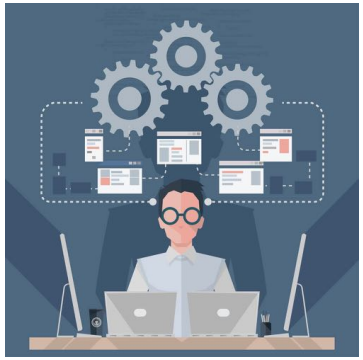
Vanilton Pinheiro

CSM, CSPO, Kanban System Designer, Test Analyst (QA)

TESTE ÁGIL



Quem?

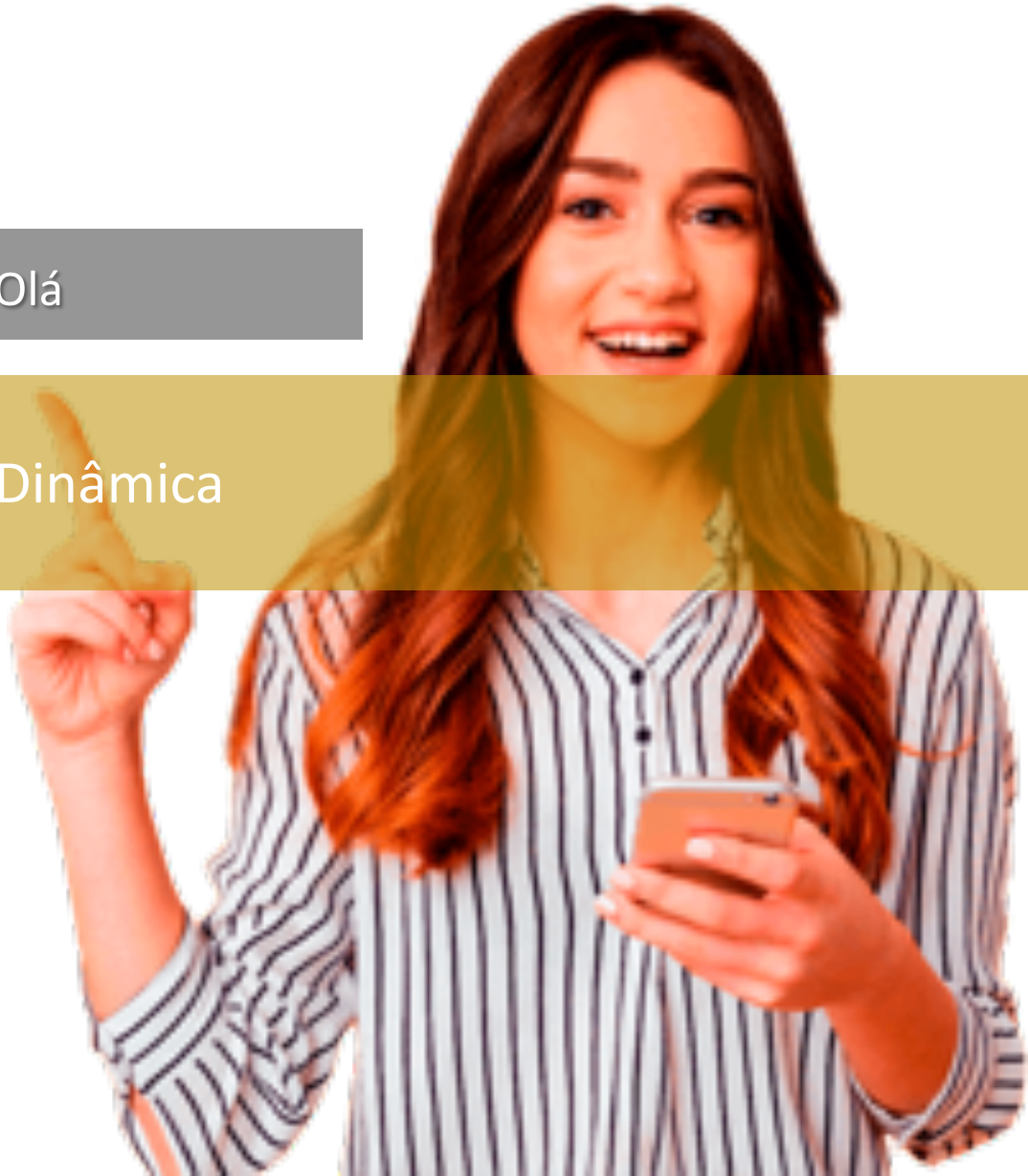
Vanilton Pinheiro

MURAL

Tópico 5

Diga Olá

Dinâmica



Breve História do Trabalho

Do Manual ao Digital

1750

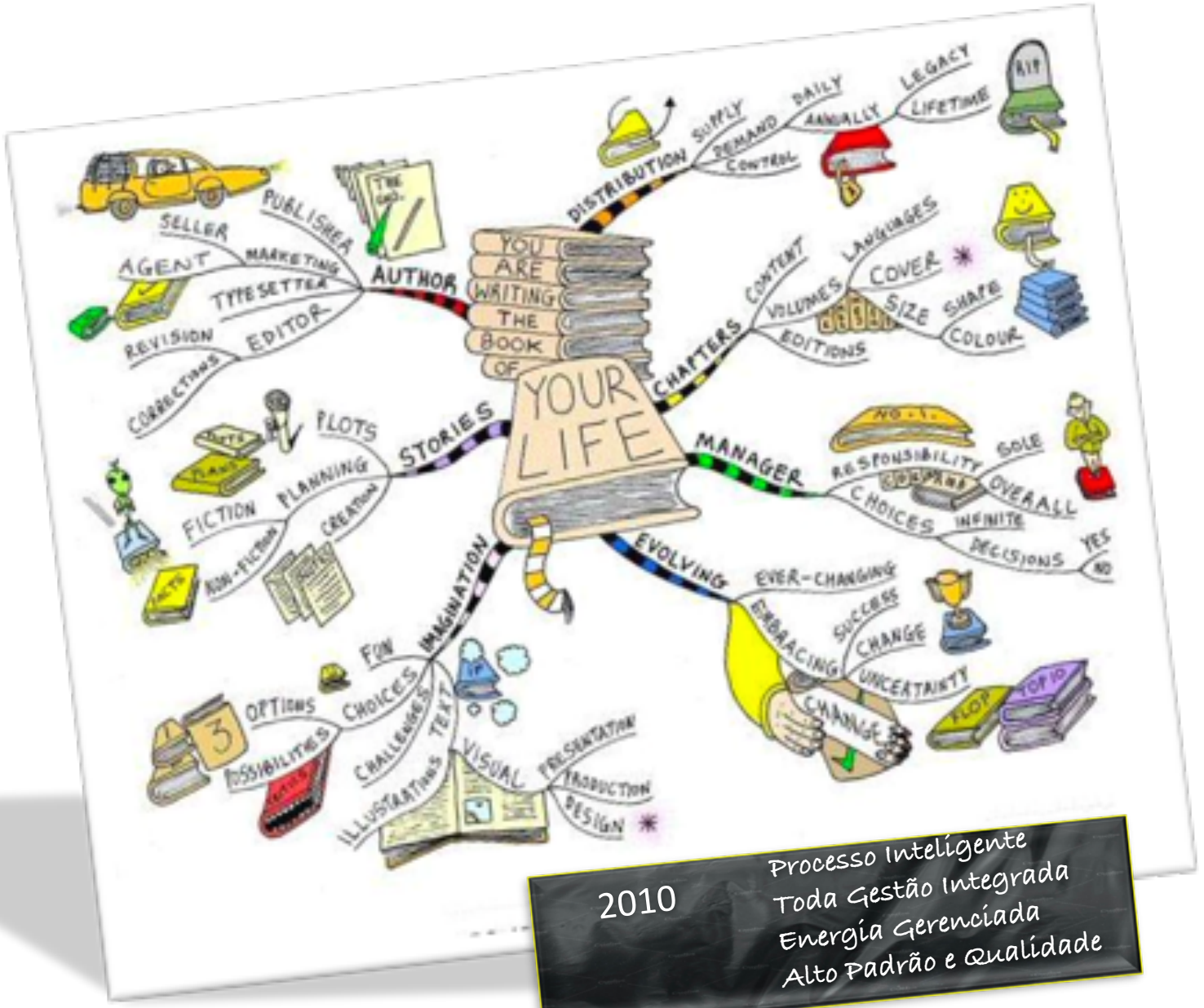
Processo Manual
Energia Precária
Sem Padronização

1880

Processo Mecanizado
Boa demanda de energia
Baixa Padronização
Comando / Eletronização

Era do emprego Match

Empregabilidade e Desenvolvimento

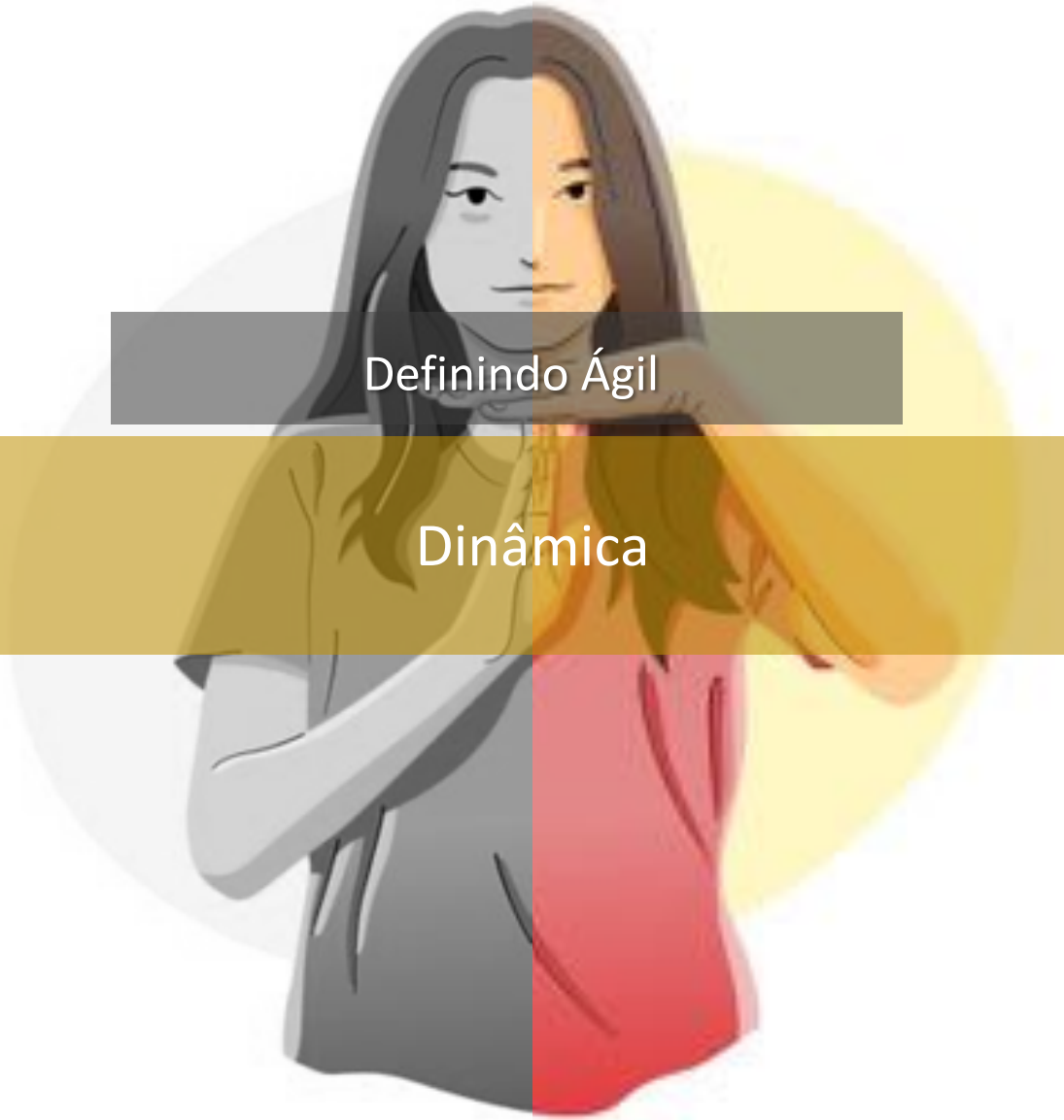


2010

Processo Inteligente
Toda Gestão Integrada
Energia Gerenciada
Alto Padrão e Qualidade

Trabalhabilidade e Life Design

Tópico 6



Definindo Ágil

Dinâmica

Um tempo para definir ágil

Timeline Desenvolvimento

Do Waterfall ao Ágil


Programação Estruturada
Melhor sequência de comandos para máquina
i love the 70

Processos

Padronizados, eficientes, sequencias


BACK TO THE 90's
Programação Orientada a Objetos
Modelar o mundo real, abstrair
Eiffel
Delphi
ABAP
I LIVE 90s


C1
Cn
A1
An
Programming Framework
Code Generation Manager
Analysis Manager
UI Manager
Validation
Model Adapter
Parameters Provider
Messages Manager
Models
Borland
Delphi


Made in the 90s
Envolvimento do cliente
ROI
Adaptação contínua
Entrega rápida
Foco no problema
Multi Disciplinaridade
Daily Scrum
Product Backlog
Sprint Tasks
Completed Features
XP
EXTREME PROGRAMMING

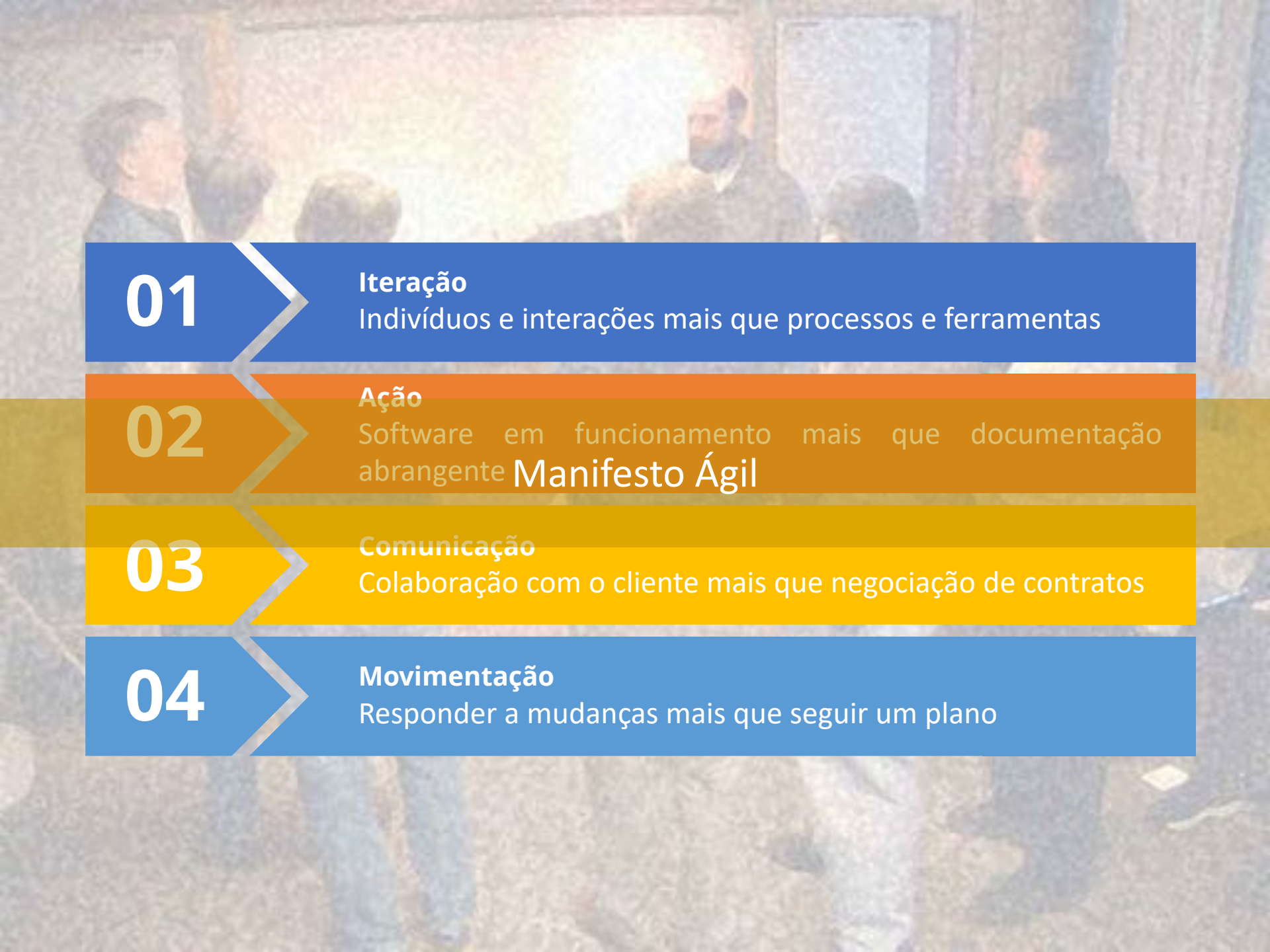

XP
EXTREME PROGRAMMING
Made in the 90s
Programação em Pares
Code Review
Integração Continua
Teste de aceitação
TDD (Test-Driven Development)
KANBAN
Card Wall
Medir Leadtime/Cycle Time
Controle de WIP
(Work in Progress)
SCRUM
Planejamento
Revisão
Retrospectiva
Reunião Diária
Product Backlog
Sprint Backlog
Burdown charts


TODO
IN PROGRESS
I ♥ 2000s


Shopping List
Burn-Down
PBI
To Do
In Progress
Done
"Yesterday I went to a very important meeting..."
I ♥ 2000s
Product backlog
Sprint backlog
24 hours
weeks
SPRINT


Designers
Programadores
Testadores
Infra
Marketing
Estatísticos
Negócios
Trabalho Criativo


Designers
Programadores
Marketing
Testadores
Infra
Estatísticos
Negócios
Criar a partir das diferenças
Trabalho Criativo



01

Iteração
Indivíduos e interações mais que processos e ferramentas

02

Ação
Software em funcionamento mais que documentação abrangente

Manifesto Ágil

03

Comunicação
Colaboração com o cliente mais que negociação de contratos

04

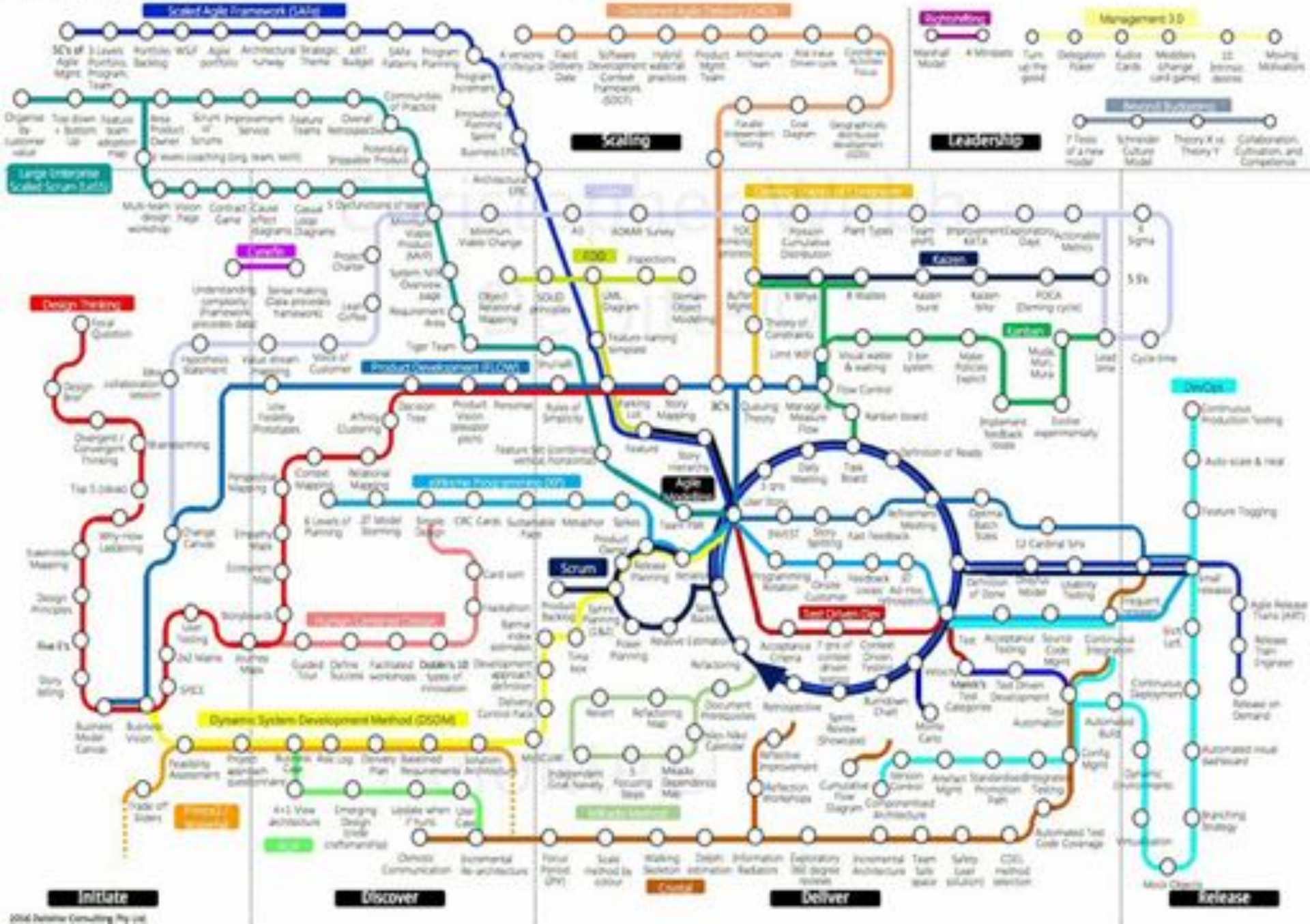
Movimentação
Responder a mudanças mais que seguir um plano

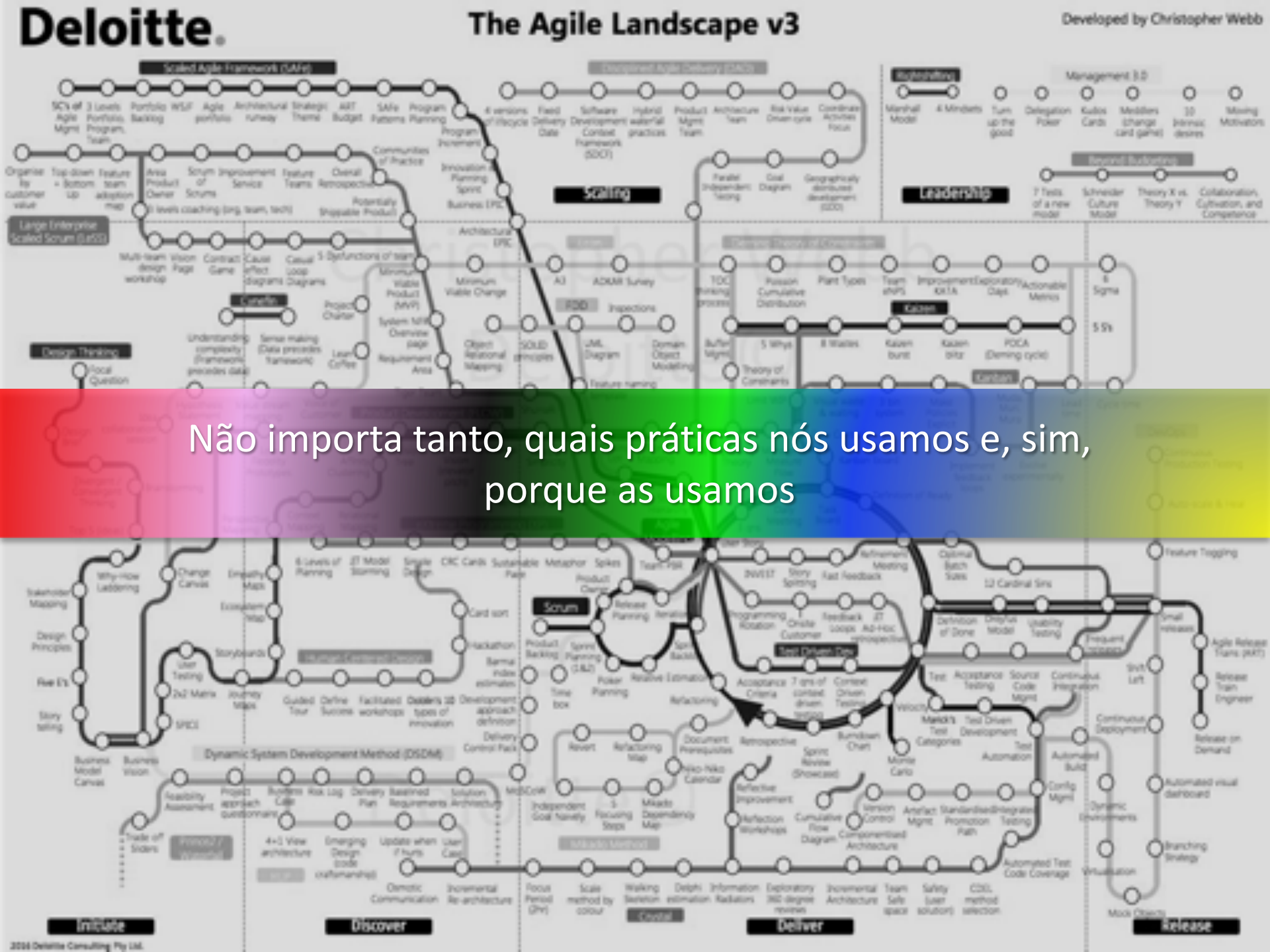
Uma atividade, essencialmente humana

Priorizar a colaboração entre as pessoas mais do que os processos e ferramentas

The Agile Landscape v3

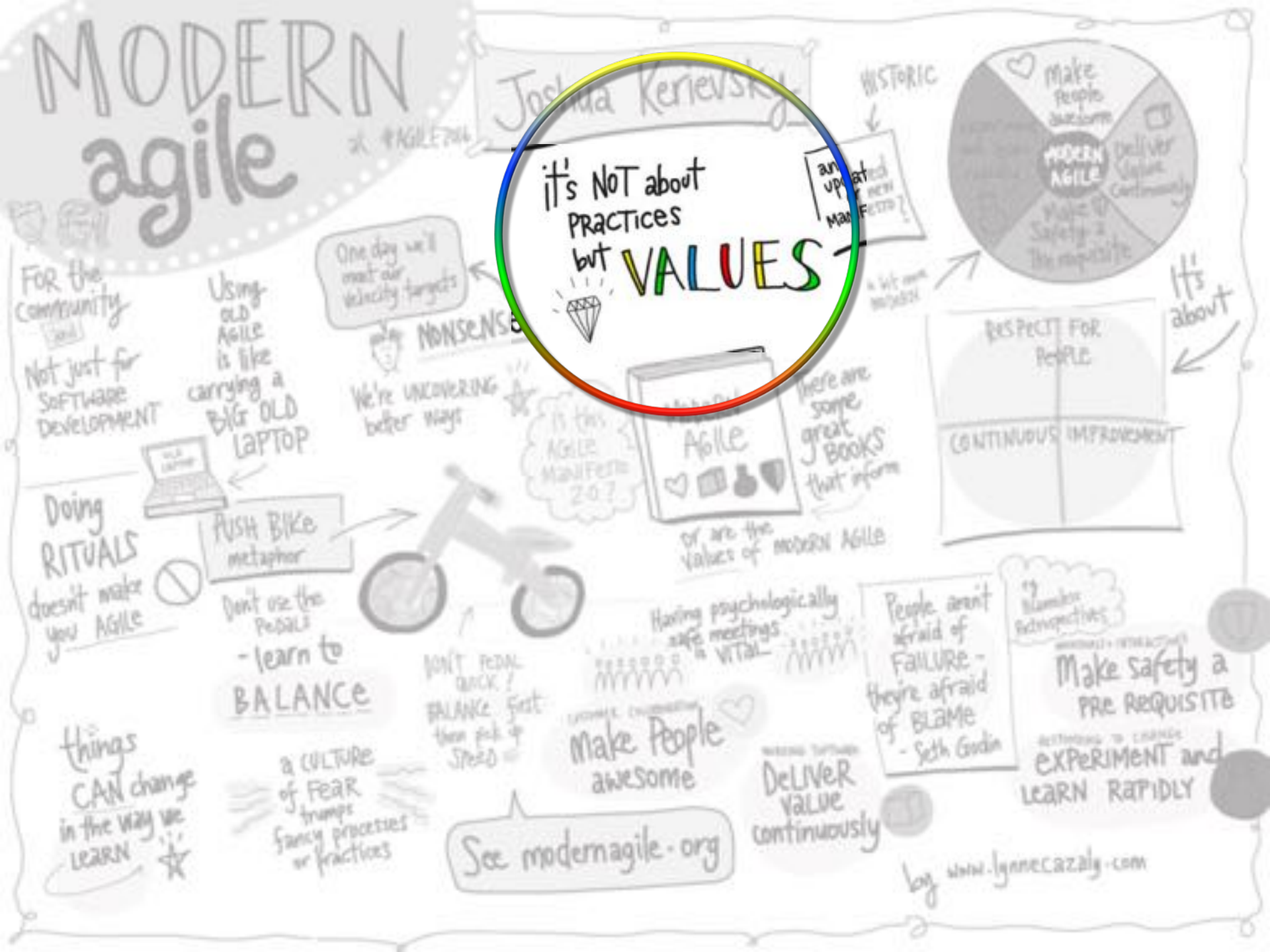
Developed by Christopher Webb





Não importa tanto, quais práticas nós usamos e, sim, porque as usamos

Não importa tanto, quais práticas nós usamos e, sim,
porque as usamos


MODERN agile
it's NOT about PRACTICES but VALUES
Joshua Kerievsky
HISTORIC
Make people awesome
Deliver Value Continuously
Make Safety a Pre-requisite
MODERN AGILE
FOR the community
Not just for SOFTWARE DEVELOPMENT
Using OLD AGILE is like carrying a BIG OLD LAPTOP
One day we'll meet our velocity targets
NONSENSE
We're UNCOVERING better ways
Doing RITUALS doesn't make you AGILE
PUSH BIKE metaphor
Don't use the PEDALS - learn to BALANCE
DON'T PEDAL QUICK! BALANCE first then pick up SPEED
things CAN change in the way we LEARN
a CULTURE of FEAR trumps fancy processes or practices
See modernagile.org
Agile
there are some great BOOKS that inform
of are the Values of modern Agile
RESPECT FOR PEOPLE
CONTINUOUS IMPROVEMENT
It's about
Having psychologically safe meetings is VITAL
Make People awesome
DeLiVeR value continuously
People aren't afraid of FAILURE - they're afraid of BLAME - Seth Godin
Blameless Retrospectives
Make safety a PRE REQUISITE
eXPERIMENT and LEARN RAPIDLY
by www.lynnecazaly.com

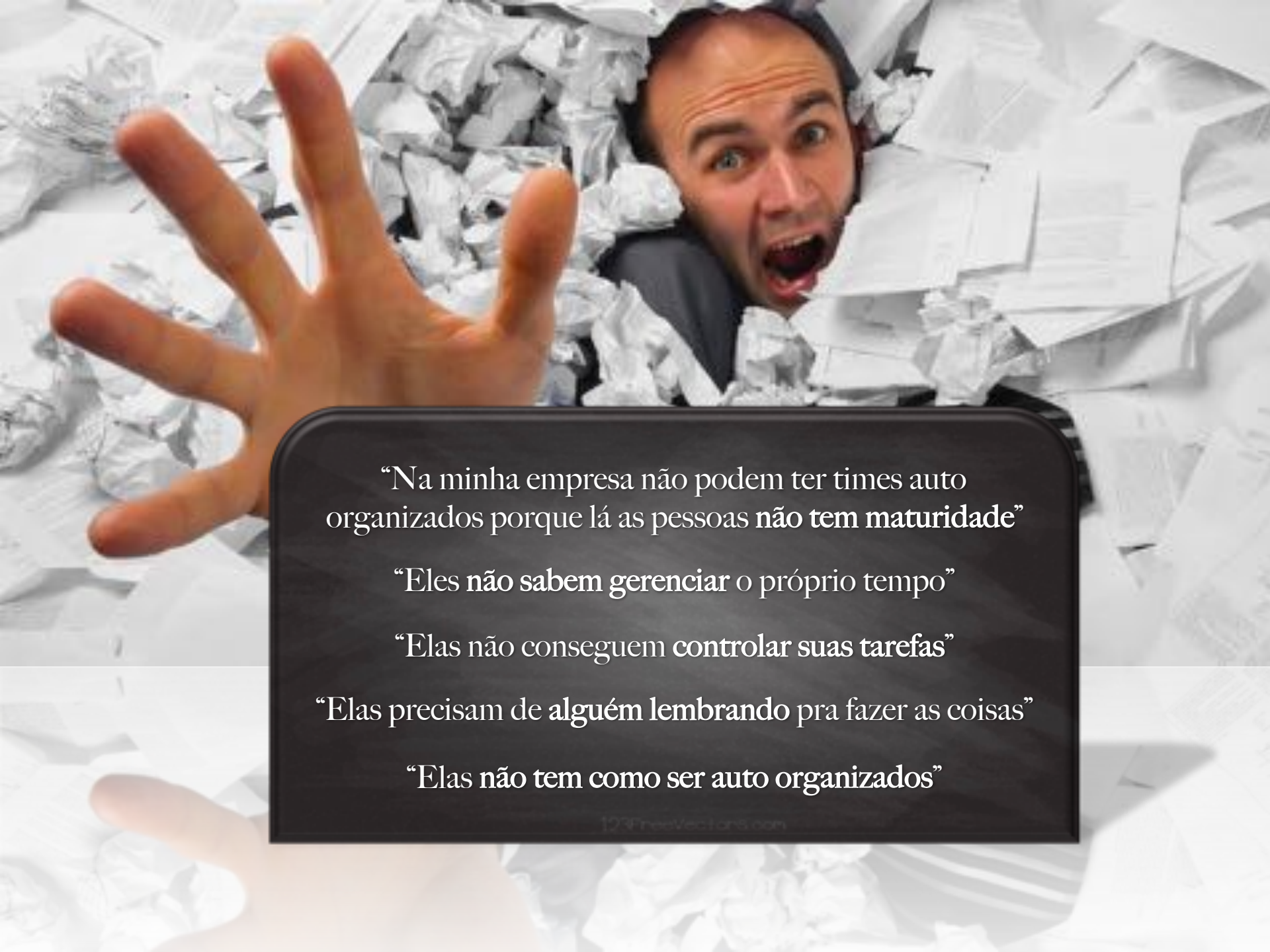
Cultural

Se você tem essa cultura, há grandes chances de você ser ágil


Compras


Cultura de
Auto Organização


Muitas vezes, isso causa confusão
Indivíduo Organizado
Times Auto Organizados



"Na minha empresa não podem ter times auto organizados porque lá as pessoas **não tem maturidade**"

"Eles **não sabem gerenciar** o próprio tempo"

"Elas não conseguem **controlar suas tarefas**"

"Elas precisam de **alguém lembrando** pra fazer as coisas"

"Elas **não tem como ser auto organizados**"

Se você pensa isso...

Você está dizendo que essas pessoas não tem o **desejo**, **interesse** ou **habilidade** de ser um indivíduo auto organizado

Você não está dizendo que ela não consegue participar de times auto organizados

Elas **não tem como ser auto organizados**"

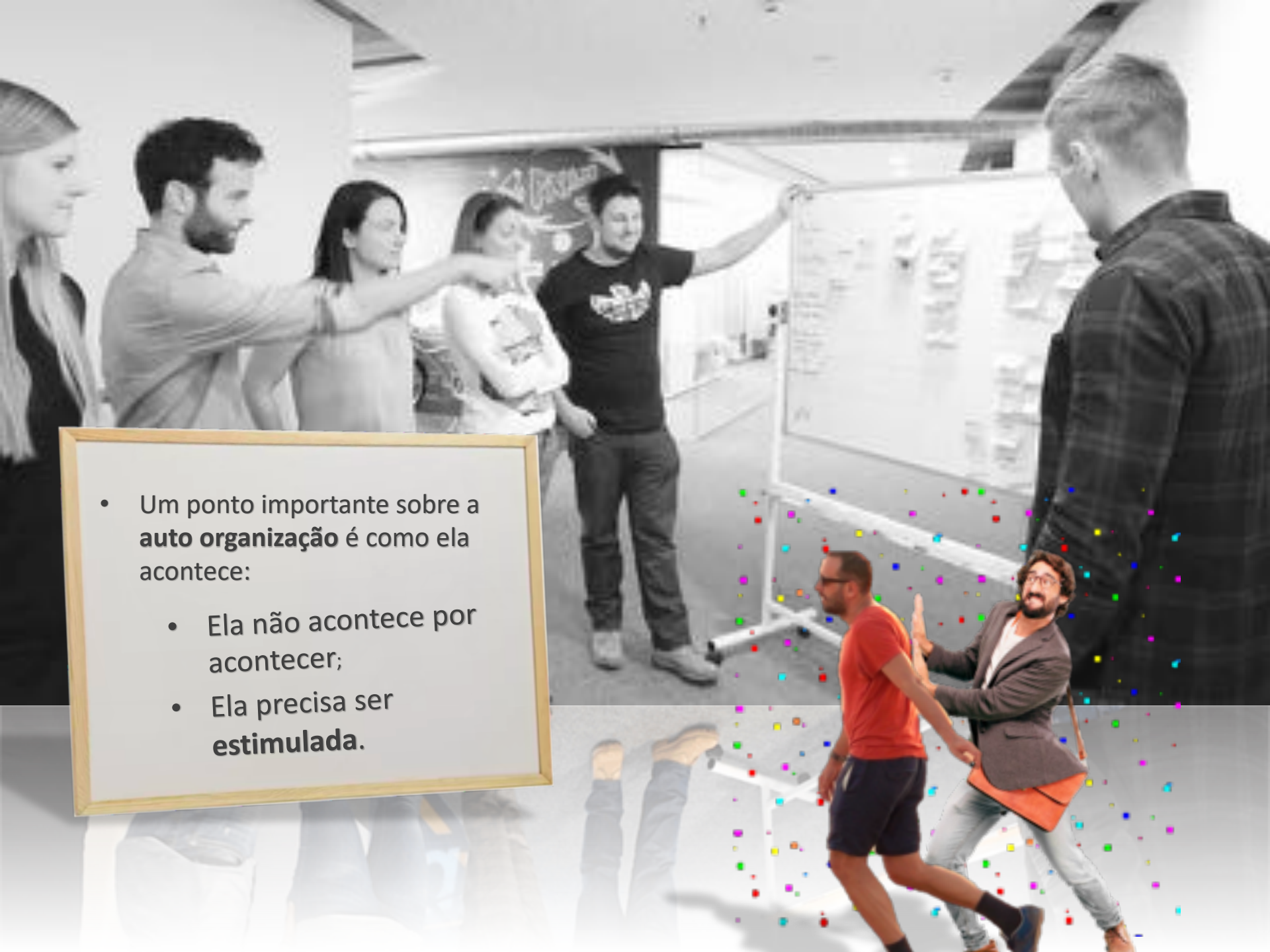
Ágil tem pessoas em **times** auto organizados

O foco da auto organização é a **equipe**

16/11

Desculpa pra viver sem chefe

- Um ponto importante sobre a **auto organização** é como ela acontece:
  - Ela não acontece **por acontecer**;

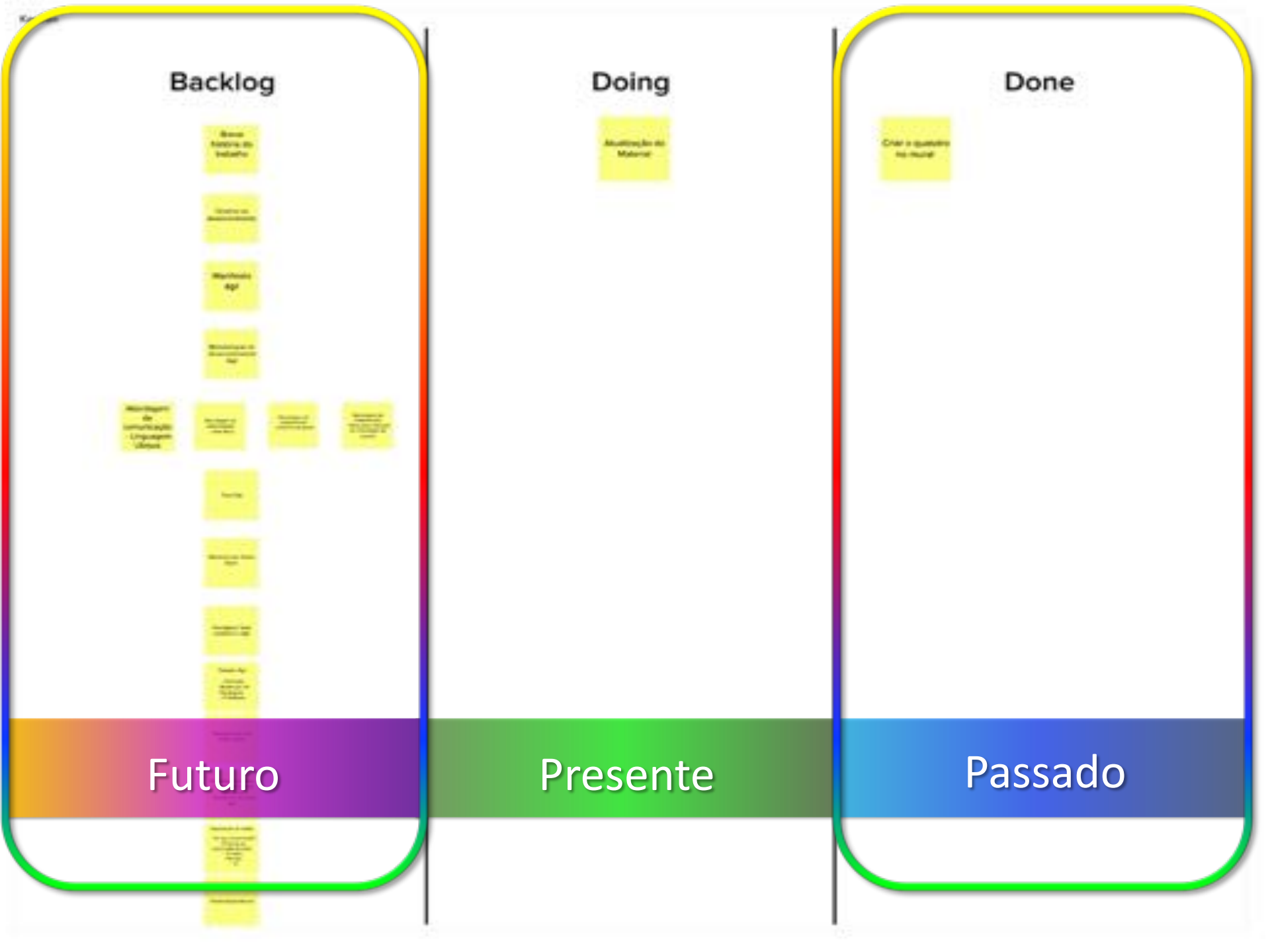


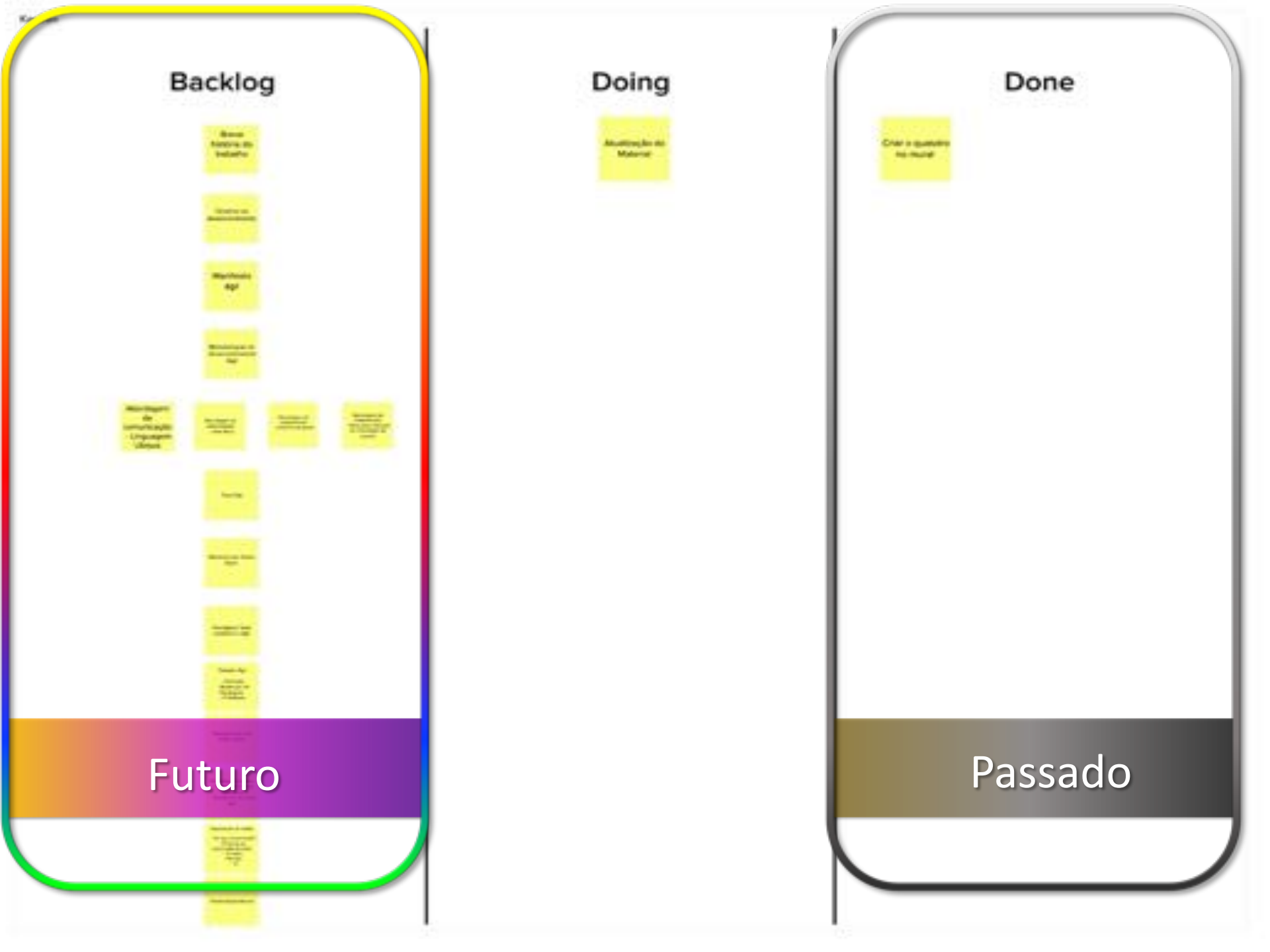
- Um ponto importante sobre a **auto organização** é como ela acontece:
  - Ela não acontece por acontecer;
  - Ela precisa ser **estimulada**.

- Grupos **auto organizam** quando:

  - Tem **necessidade**; ou

  - **Motivação**.


Backlog
Doing
Done
Futuro
Presente
Passado


Backlog
Doing
Done
Futuro
Passado

Mais do que seguir um plano, responder às mudanças

O método mais eficiente e eficaz de transmitir informações é através de uma conversa face a face.

Cultura da

Multi disciplinaridade

Sua **especialidade importa menos** que a intensidade de valor que o seu time entrega

Backlog semana

PONTO

ANUAIS (RELATÓRIOS)

HOMOLOGAÇÃO

POLÍTICA DE

Fazendo

TESTAR BENEFÍCIOS

1ª Parcela 13º Salário

Acompanhamento

ANUAIS

BENEFÍCIOS

IMPLANTAÇÃO DO MERIT MONEY

POLÍTICA DE CARGOS E SALÁ

FGTS

RH-SUMMIT

LEVANTAR HISTÓRICO DE

AÇÕES NPS-RH

DO MERIT MONEY

CARGOS E SALÁRIOS

ENVIAR E-SOCIAL

FÉRIAS

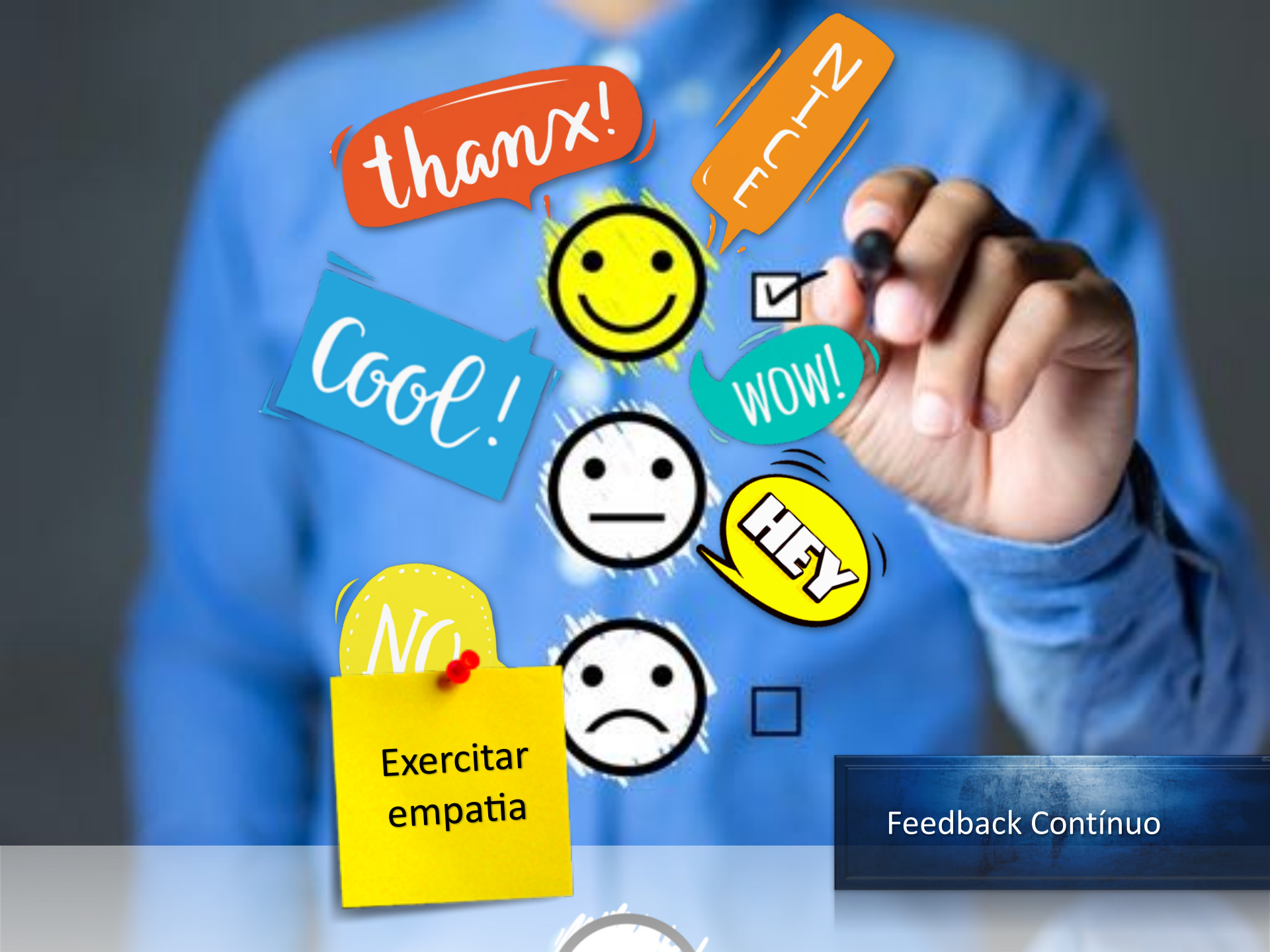

Muito menos ou nenhum
Micro gerenciamento


Foco no macro
Onde nós queremos chegar?

Autonomia

Para que as pessoas que estão imersas naquele contexto escolham a melhor forma de resolver o problema

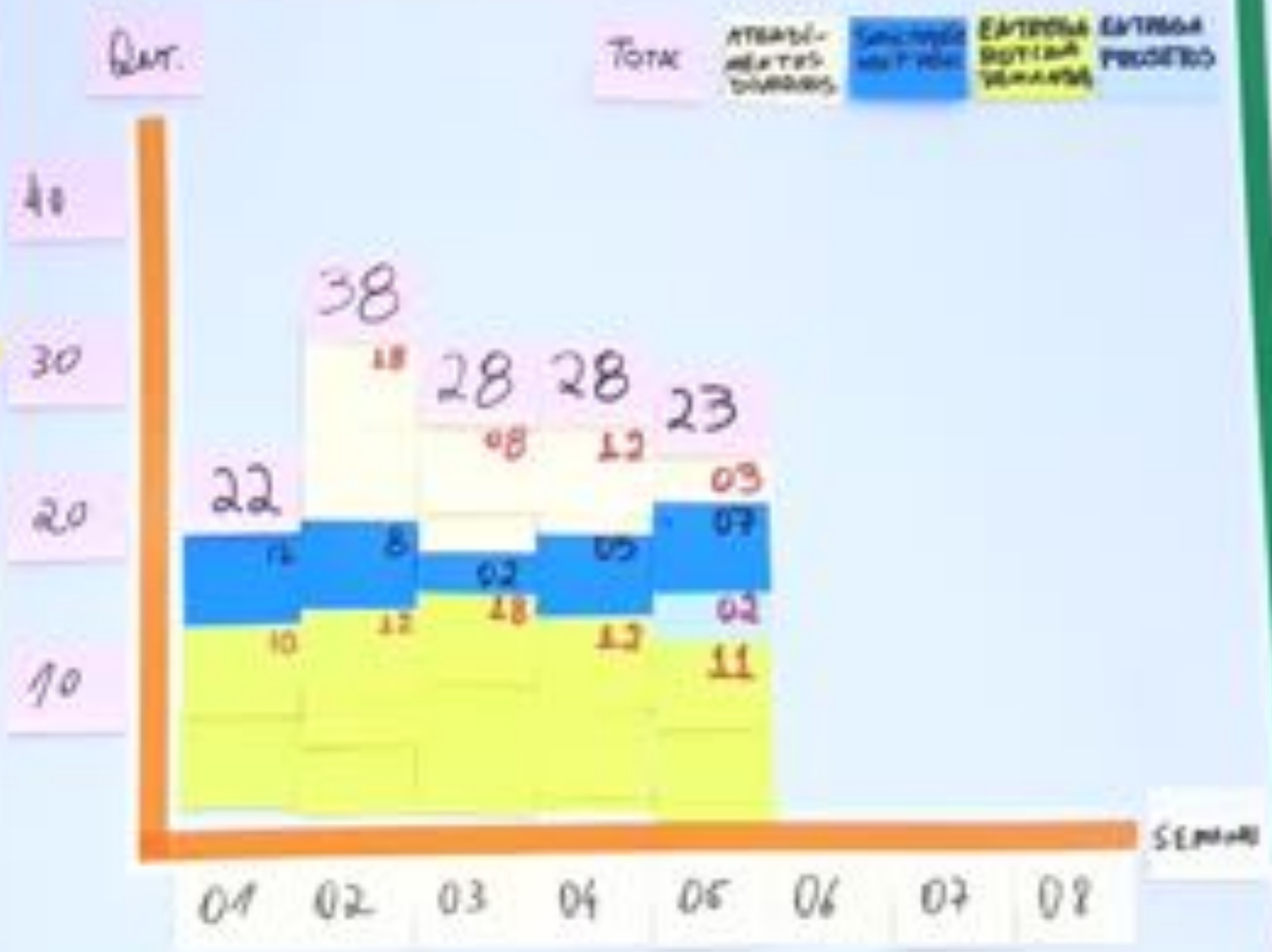

Restrições
Ao invés de Regras
BEST PRACTICE
Visibilidade das boas práticas. Se não servir, inove.


thanx!
Cool!
WOW!
HEY
Exercitar empatia
Feedback Contínuo

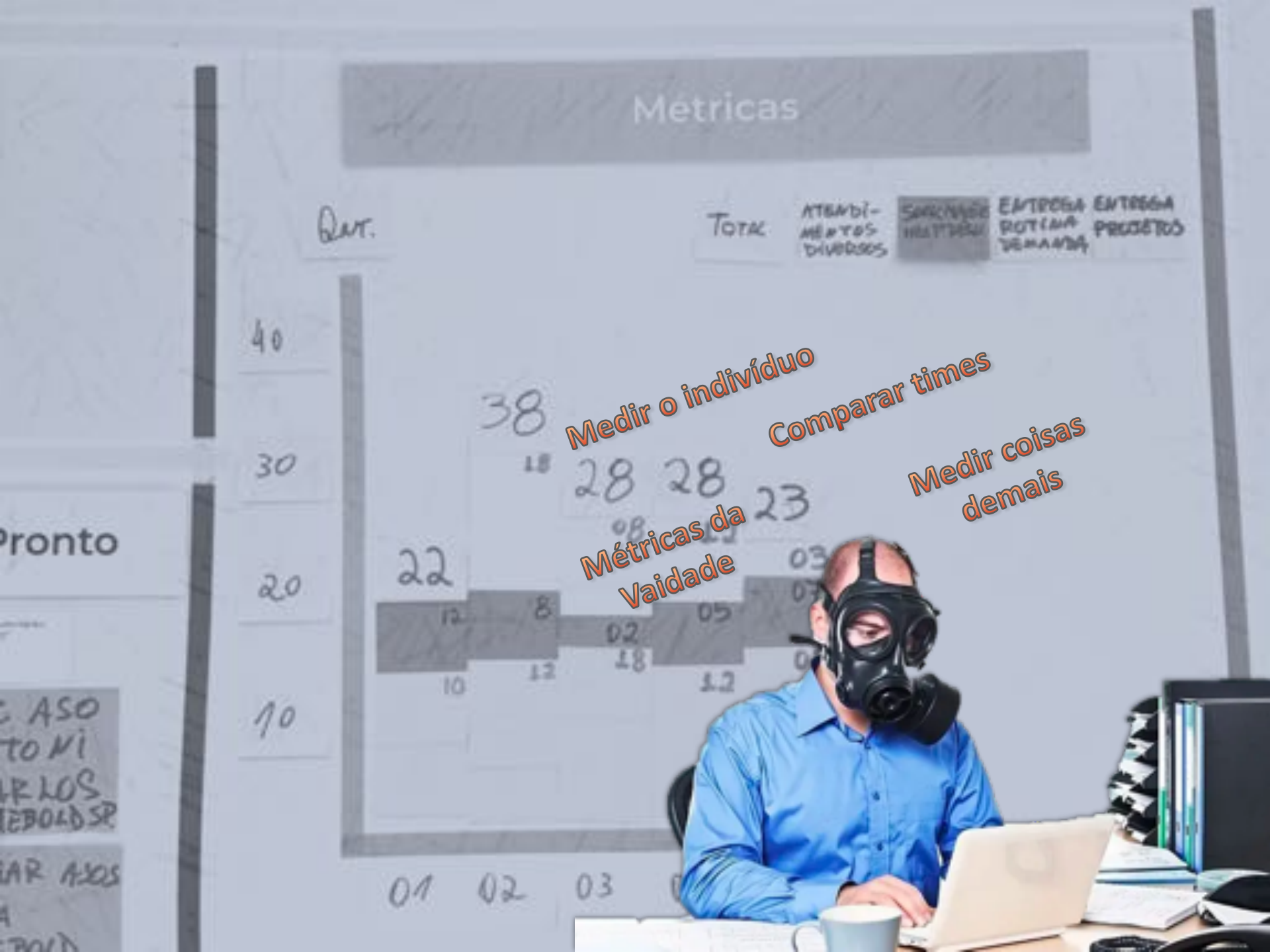

Priorize
feedbacks
face a face


Entrega frequente de valor

Métricas



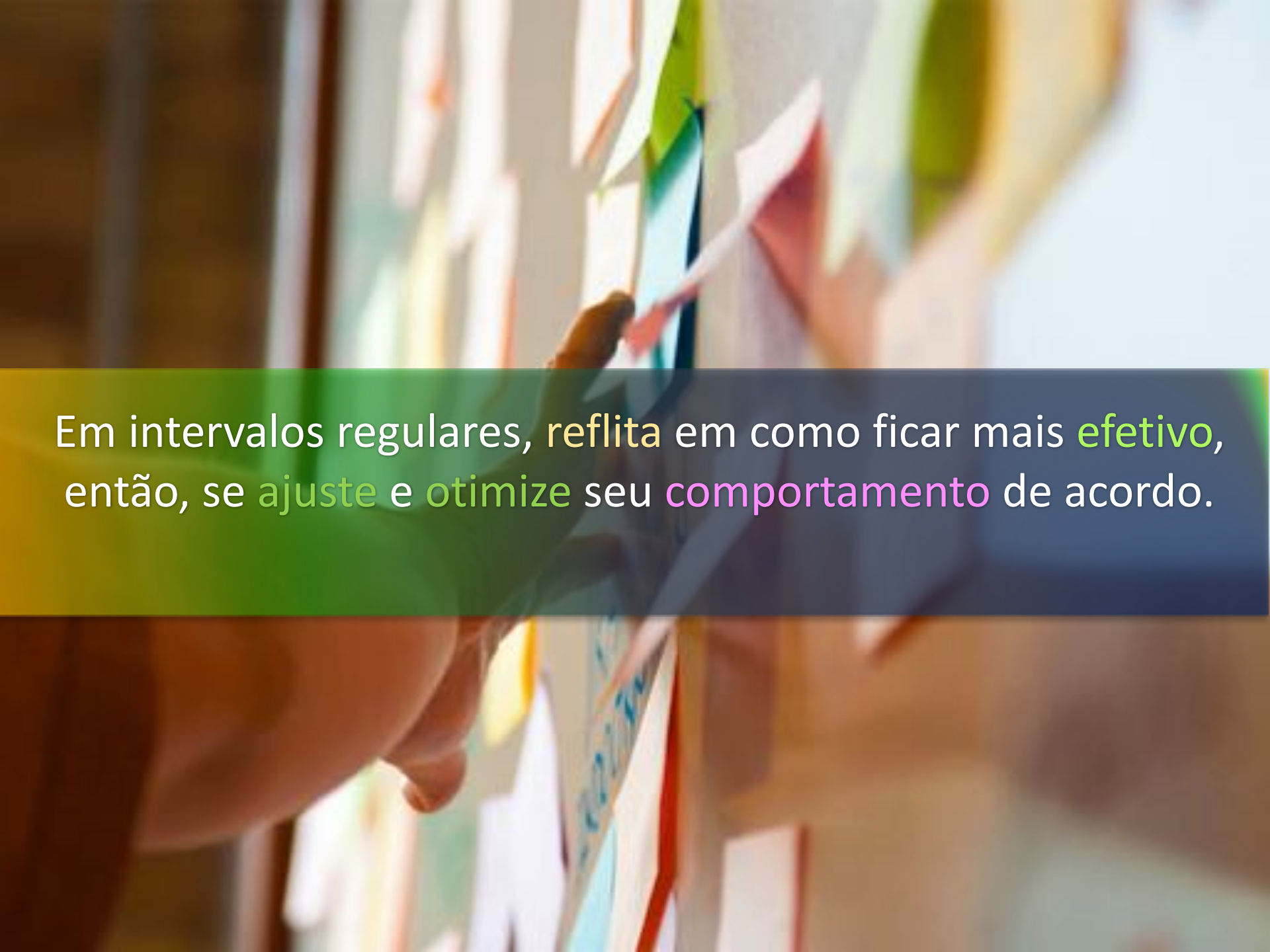
Pronto


Métricas
Qnt.
Total
Atendimentos diversos
Entrega rotina semanal
Entrega projetos
40
30
20
10
38
28
28
23
22
Pronto
01
02
03
Medir o indivíduo
Comparar times
Métricas da Vaidade
Medir coisas demais

Processos ágeis promovem um **ambiente sustentável**. As pessoas devem ser capazes de manter indefinidamente, passos constantes


CONTINUOUS
IMPROVEMENT

Agilidade tem uma relação com a velocidade com que você aprende e se adapta a partir dos feedbacks e das mudanças


Em intervalos regulares, reflita em como ficar mais efetivo,
então, se ajuste e otimize seu comportamento de acordo.


Cultura da
Experimentação

TRY

CCESS

você acerta, ou

ocê aprende

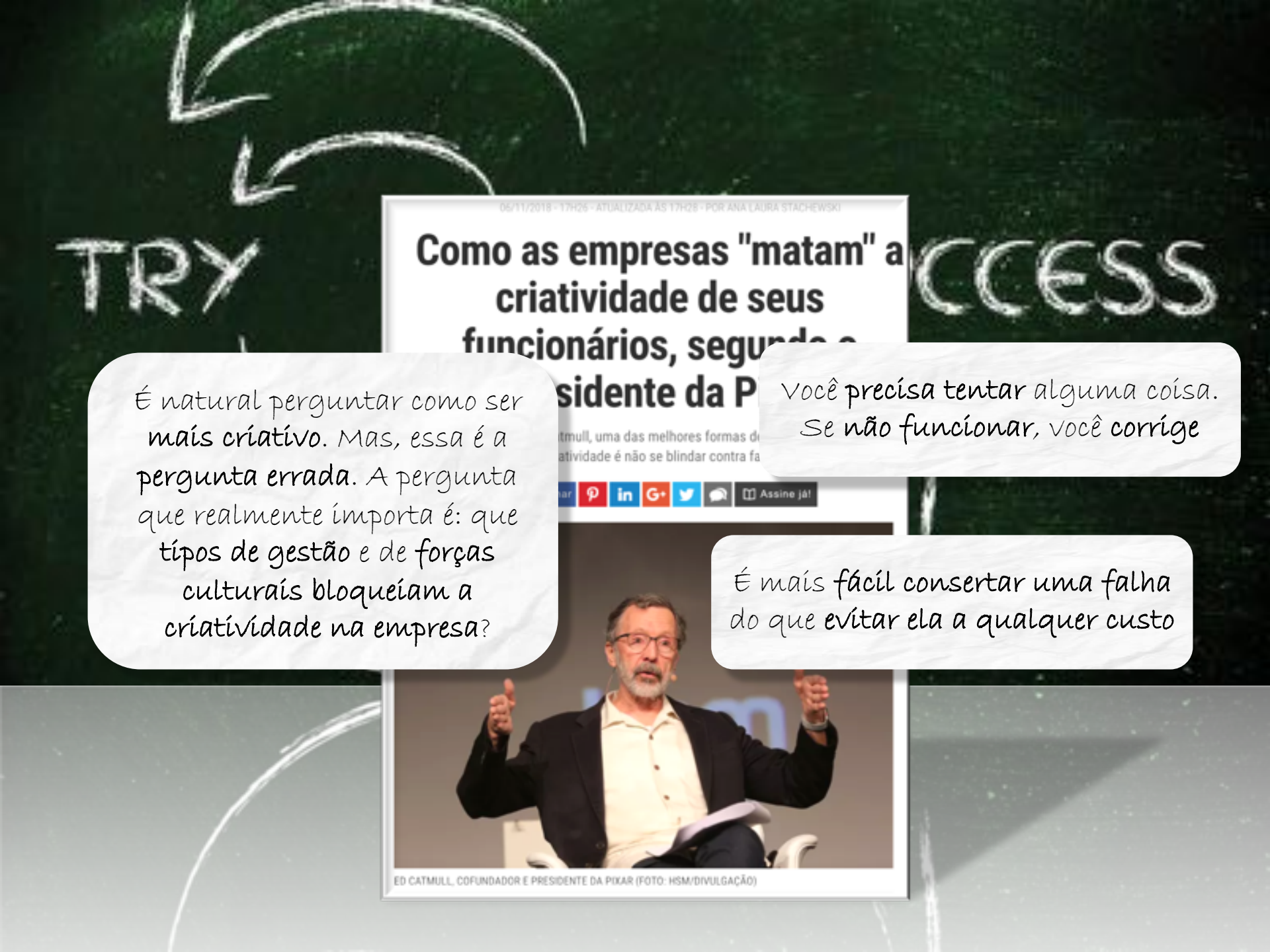
Isso redefine a noção de Progresso

06/11/2018 - 17H26 - ATUALIZADA ÀS 17H28 - POR ANA LAURA STACHEWSKI

Como as empresas "matam" a criatividade de seus funcionários, segundo o presidente da Pixar

Para Ed Catmull, uma das melhores formas de estimular a criatividade é não se blindar contra falhas

ED CATMULL, COFUNDADOR E PRESIDENTE DA PIXAR (FOTO: HSM/DIVULGAÇÃO)



06/11/2018 - 17H26 - ATUALIZADA ÀS 17H28 - POR ANA LAURA STACHEWSKI

Como as empresas "matam" a criatividade de seus funcionários, segu... sidente da P...

...tmull, uma das melhores formas d... ...atividade é não se blindar contra fa...

Assine já!

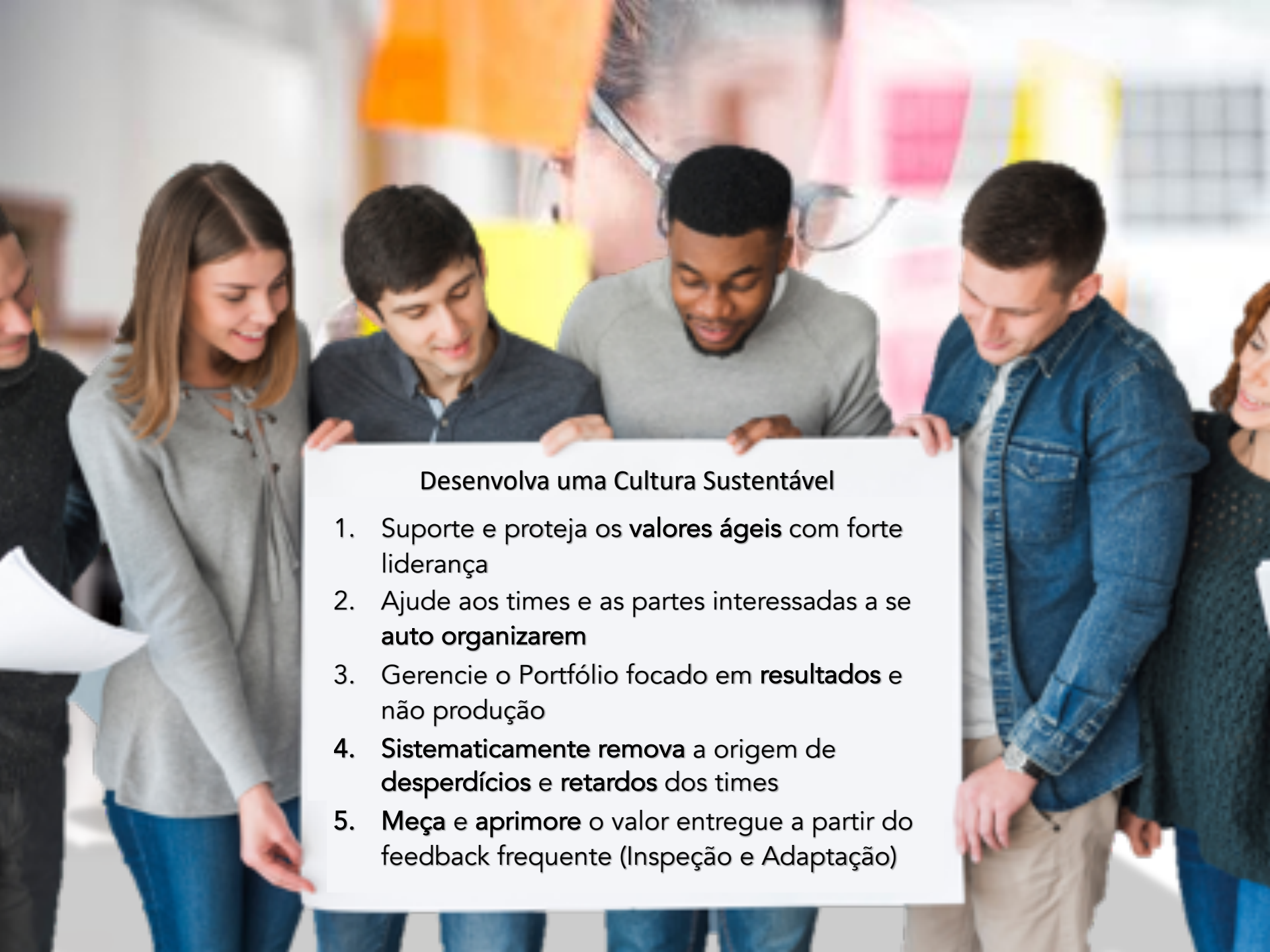
ED CATMULL, COFUNDADOR E PRESIDENTE DA PIXAR (FOTO: HSM/DIVULGAÇÃO)

É natural perguntar como ser **mais criativo**. Mas, essa é a **pergunta errada**. A pergunta que realmente importa é: que **tipos de gestão** e de **forças culturais bloqueiam a criatividade na empresa**?

Você **precisa tentar** alguma coisa. Se **não funcionar**, você **corrige**

É mais **fácil consertar uma falha** do que **evitar ela a qualquer custo**


WHAT HAVE
YOU LEARNED?


Desenvolva uma Cultura Sustentável
1. Suporte e proteja os valores ágeis com forte liderança
2. Ajude aos times e as partes interessadas a se auto organizarem
3. Gerencie o Portfólio focado em resultados e não produção
4. Sistematicamente remova a origem de desperdícios e retardos dos times
5. Meça e aprimore o valor entregue a partir do feedback frequente (Inspeção e Adaptação)

Abordagem de comunicação

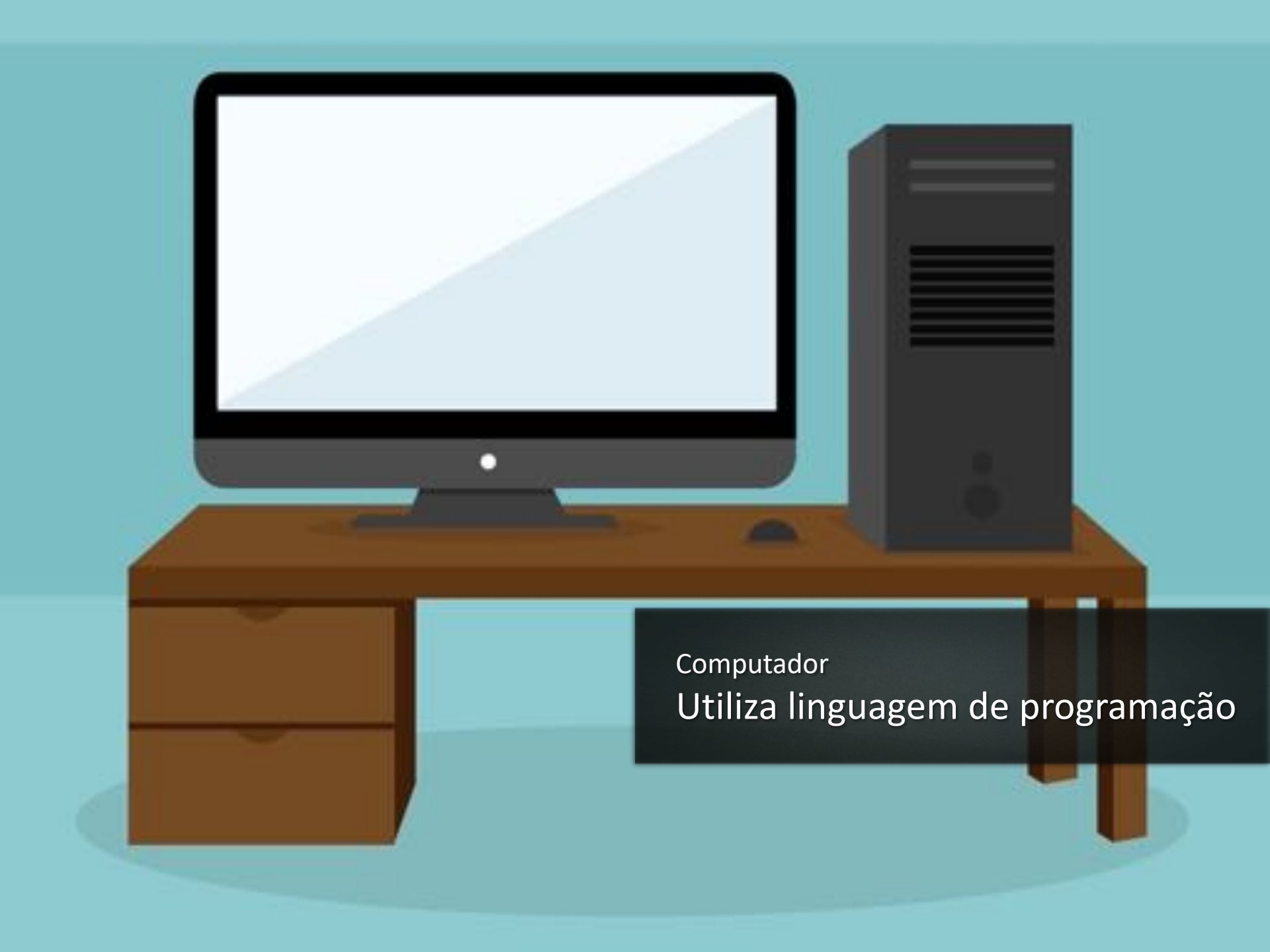
Linguagem Ubíqua

Temos **um grande problema** no desenvolvimento de um software dentro de uma equipe


MAINTENANCE
ARCHITECTURE
UPDATE
FEATURE
STANDARD
SOLUTION
DESIGN
METHODOLOGY
REQUIREMENTS
DEPLOYMENT
RESEARCH
PROTOTYPING
TESTING
SOFTWARE
API
STEP
DEVELOPMENT
UPGRADE
WEB
SPECIFICATION
PLATFORM
EMBEDDED
APPROACH
PROCESS
PROGRAMMING
FRAMEWORK
SUPPORT
LANGUAGE
ENGINEERING
LIFE CYCLE
CLIENT
Desenvolvedores
Utilizam palavras técnicas

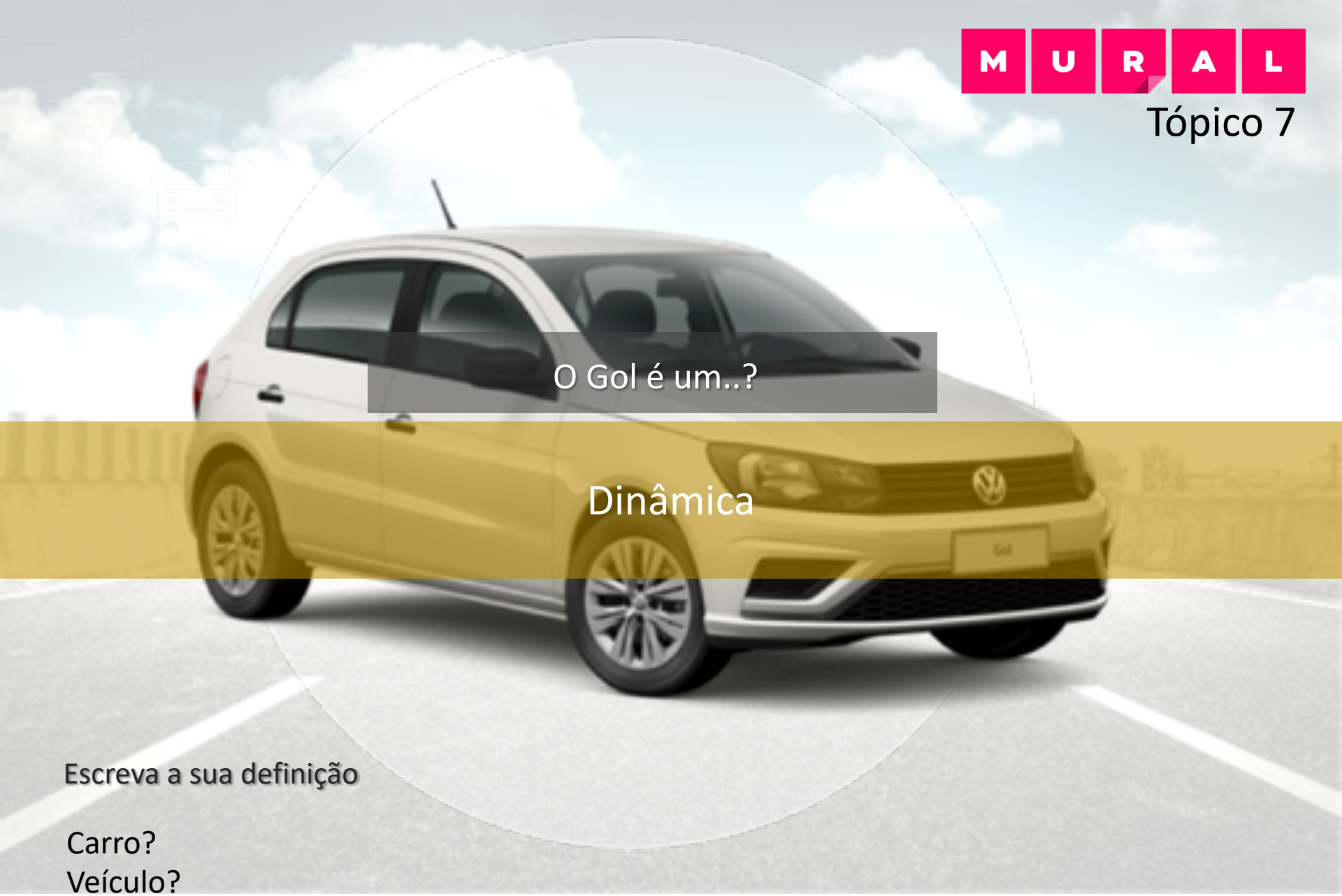
Analistas

Utilizam terminologias de sua área


Computador
Utiliza linguagem de programação

MURAL

Tópico 7



O Gol é um..?

Dinâmica

Escreva a sua definição

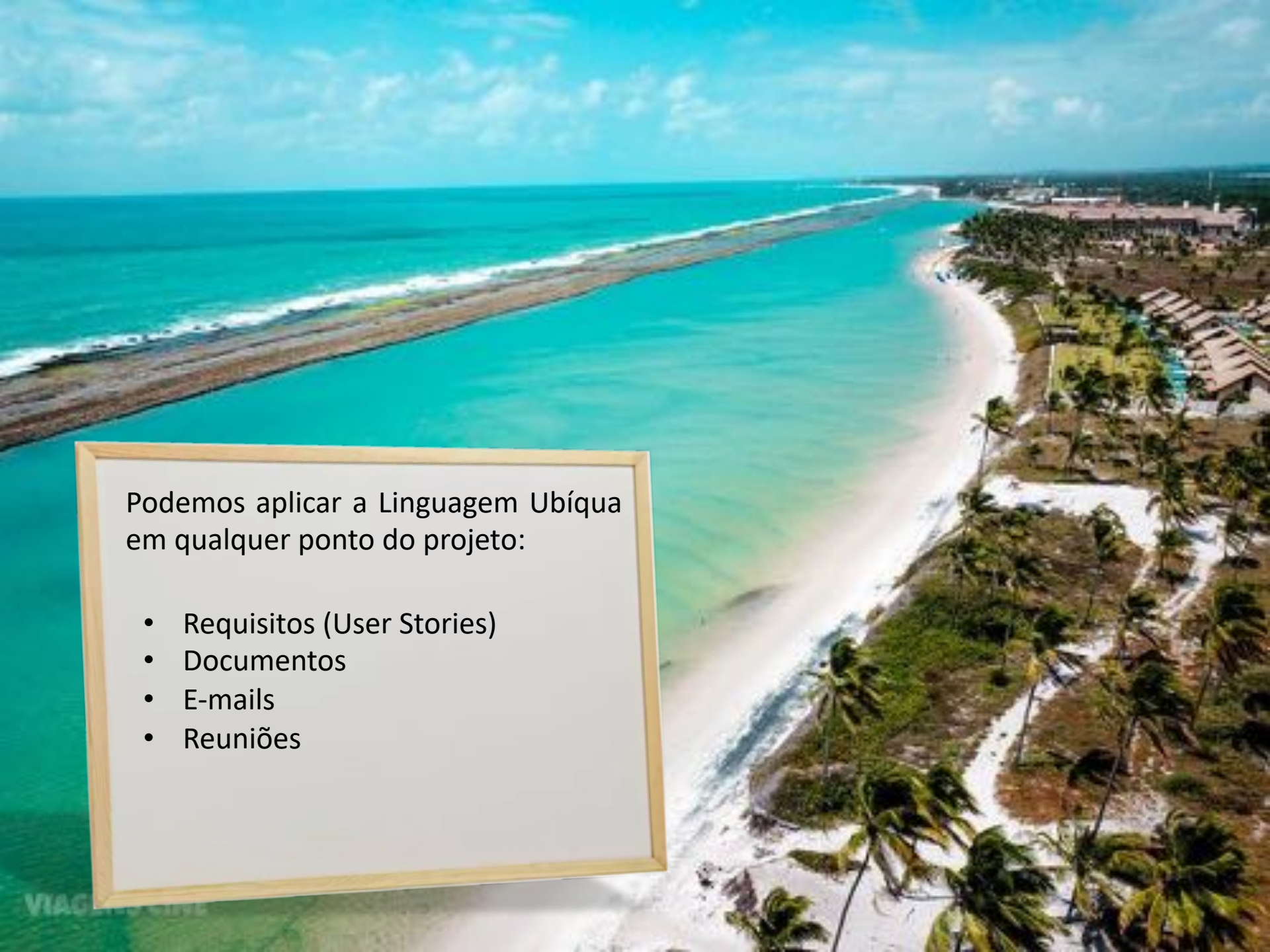
Carro?
Veículo?
Automóvel?
Modelo?

O **usuário** efetua o login com usuário e senha válido e visualiza a tela com diversos campos

O **estudante** efetua o login com usuário e senha válido e visualiza **a lista de provas** marcadas para o dia

```
String string = new StringBuffer();

public class listDAO() {

     public List<User> allData() {
       try {
         // codigo aqui
       } catch (Exception e) {
         e.printStackTrace();
       }
   }
}
```

```
String usuario = new StringBuffer();

public class listaProvasDia() {

    public List<Estudante> retornaTodosDados() {
        try {
            // codigo aqui
        } catch (NaoHaConsultultasException e) {
          e.printStackTrace();
    }
}
```



Podemos aplicar a Linguagem Ubíqua em qualquer ponto do projeto:

- Requisitos (User Stories)
- Documentos
- E-mails
- Reuniões

Vantagens de utilização

- Menos risco de falta de entendimento
- Comunicação mais rápida e direta
- Conhecimento do domínio por todos
- Entendimento/clarificação de código

Ela nos ajuda na comunicação, contextualização, identificação e delimitação de conceitos.

De forma muito resumida, linguagem ubíqua é: Todos falando a mesma língua e usando as mesmas palavras para definir as mesmas coisas.

Abordagem de comunicação

User Story

Sua definição de pronto orienta os testes necessários para a estória


Um requisito é muito mais do que uma história para poder descrever a necessidade do usuário.
Método 3C proposto por Ron Jeffries
• Card
• Conversation
• Confirmation
Estória
Conversa
Exemplos
Requisito


Quem?
Papéis
Personas
O que?
Ações
Rotinas
Porque?
Estratégia no negócio
Efeito no produto

Como um

<PAPEL>

eu posso/gostaria/devo

<FUNÇÃO>

para/de

<RESULTADO para o NEGÓCIO>

Como um **aluno EAD**, gostaria de **enviar uma dúvida ao professor de minha disciplina** para **que eu possa sanara dúvida de um exercício**

Como um **Professor**, gostaria de obter **a nota dos alunos minha disciplina** para **aprovar ou reprovar um aluno**

Como um **aluno do de pós graduação EAD**
eu gostaria de **visualizar as notas de todas as disciplinas**
Para **saber se eu posso obter meu certificado**

O que mais é importante saber?

Funcionalidade: <nome da funcionalidade>

Como um <papel/persona>
Eu quero/posso/gostaria <meta a ser alcançada>
De modo que <a razão para alcançar a meta>

NARRATIVA

-<Listar itens óbvios>
-<Listar itens que tenham relevância no software>

FORA DE ESCOPO

- <Listar o que o cliente não quer que seja desenvolvido>

O que mais é importante saber?

Como testar

Critério de Aceite

- Um Critério de Aceitação é onde expressamos como iremos garantir que um requisito (user story) será, além de testável, validado e entendido pelo cliente e qualquer pessoa do time.
- Ele utiliza a notação **Gerkin** Given–When–Then que conheceremos logo mais.

Cenário: <descrição do teste>
Dado <pré-condição>
Quando <ação>
Então <resultado esperado>

Cenário: <descrição do teste>
Dado que eu estou na página da disciplina
Quando eu clicar no link "Visualizar notas das disciplinas"
Então eu visualizo cada disciplina cursada e a respectiva nota

- Este mesmo documento pode ser utilizado por todos para:
  - O desenvolvimento da aplicação
  - Teste da aplicação
  - Aceitação da aplicação

MURAL

Tópico 8

Sistema de Triângulo

Dinâmica

- Eu sou um professor de matemática
- Meus alunos não sabem os tipos de triângulos
- Eu preciso de um sistema para apresentar os tipos de triângulos

MURAL

Tópico 13

O que é teste para vocês?

Dinâmica

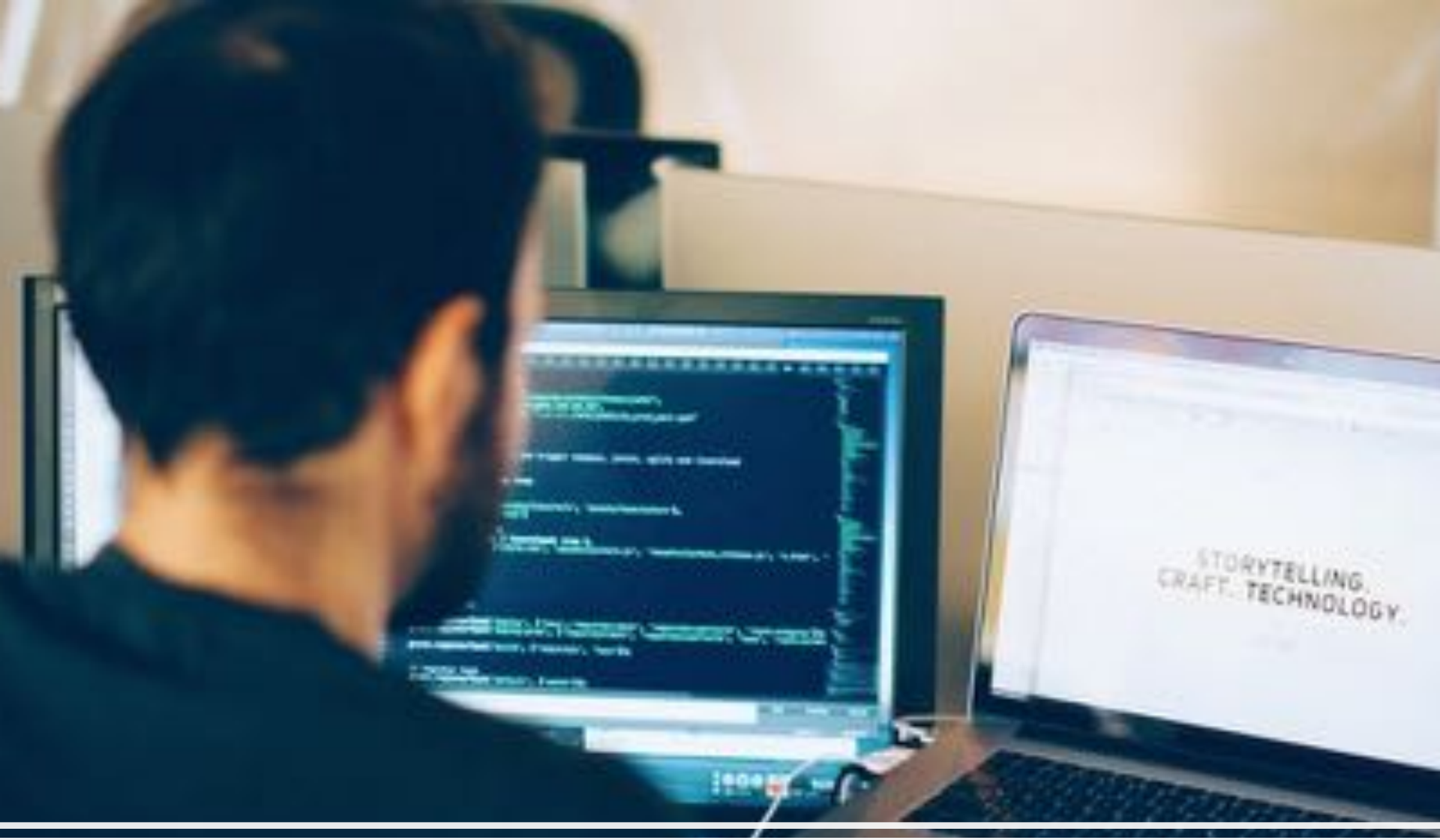
Boehm e Basili (2001) apontam que é quase improvável liberar um produto de software livre de defeitos.

Por quê testar Software?

GÊNESIS

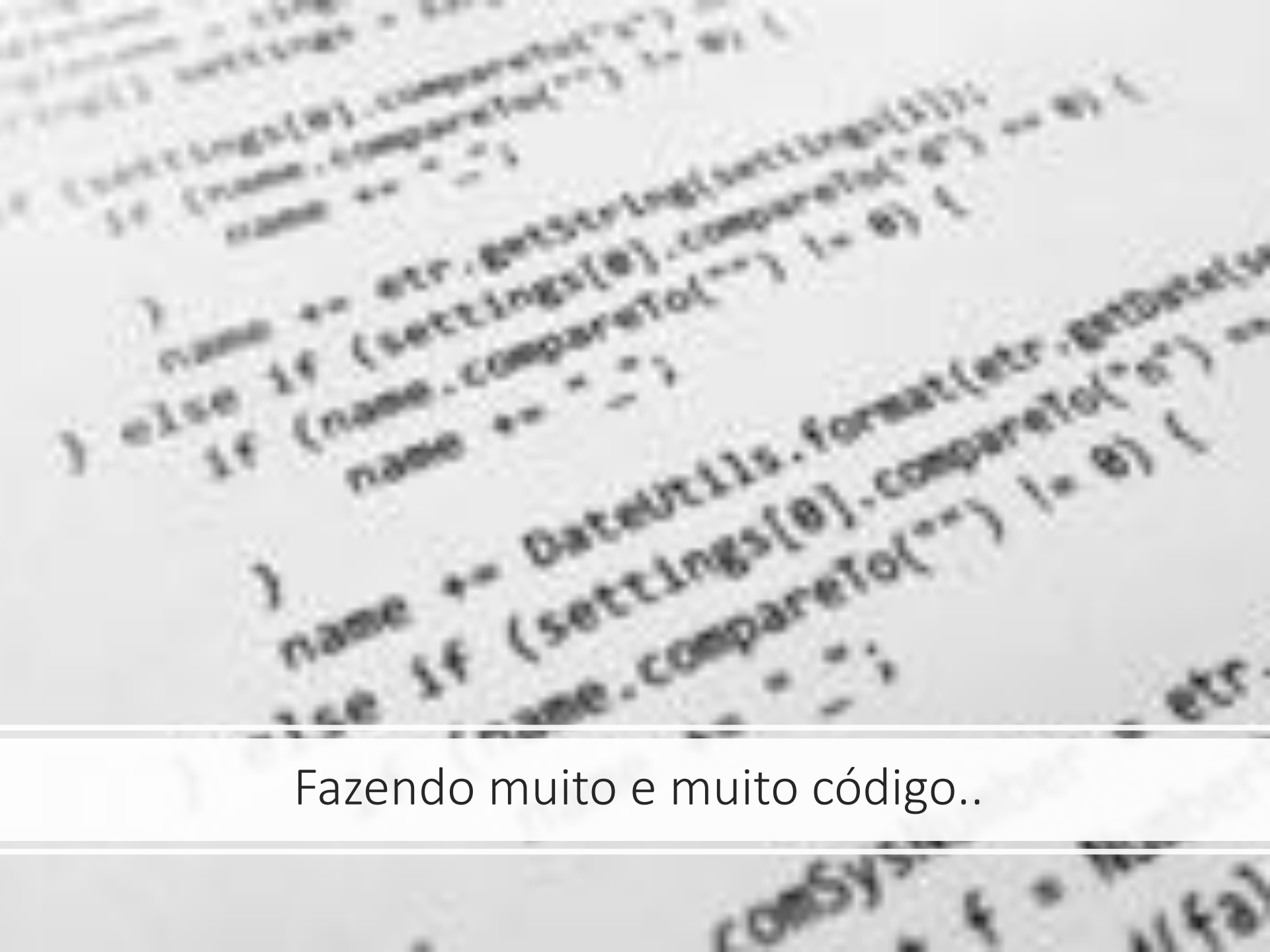
O INÍCIO DE TUDO

"No princípio criou Deus o céu e a terra"
Gênesis 1:1



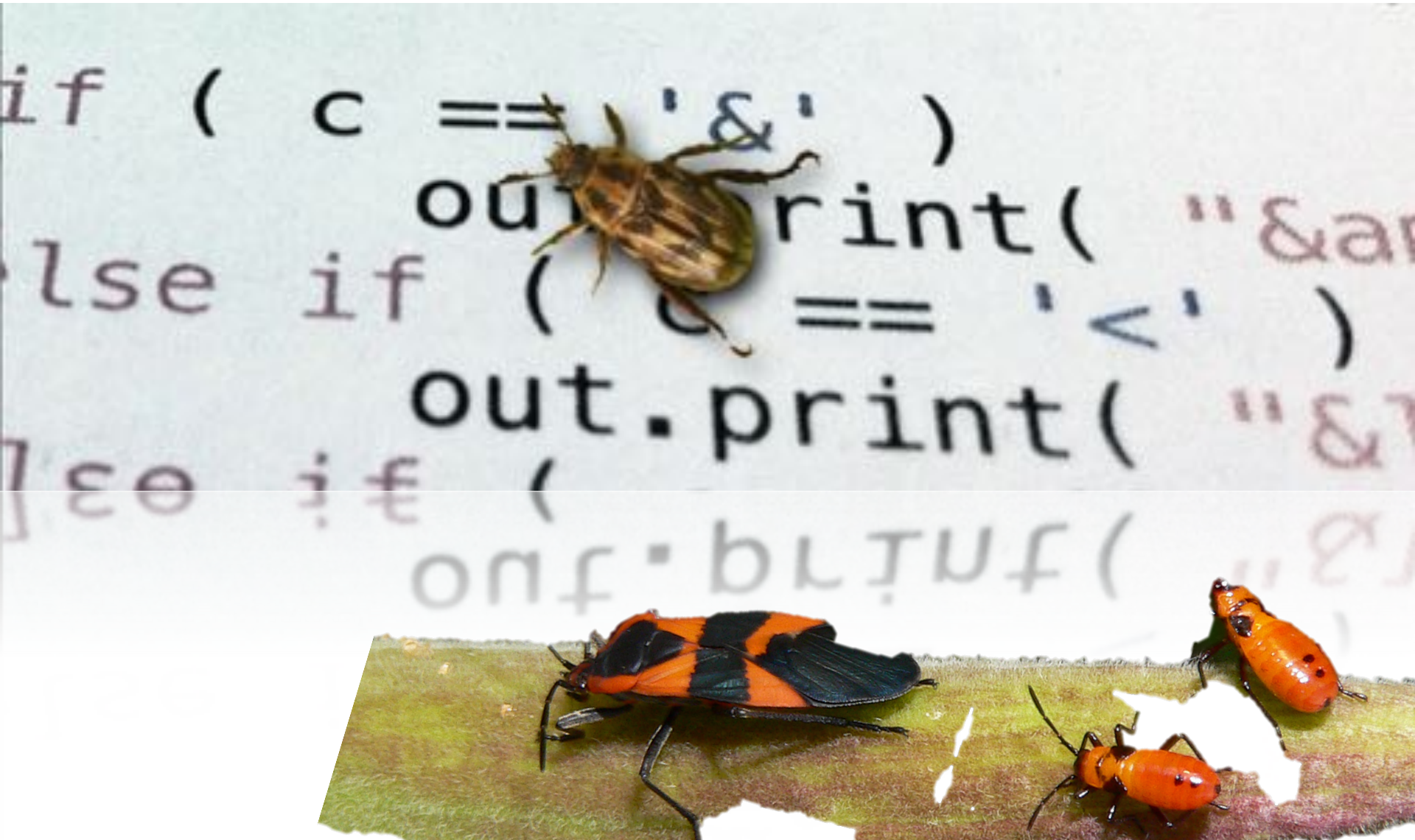
E os desenvolvedores...

E mais desenvolvedores...



Fazendo muito e muito código..

E no meio desses códigos...



Mas por que existe esses bugs?

Erro → Defeito → Falha

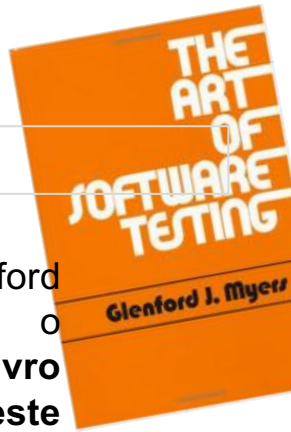
Mas por que existe esses bugs?

```
1  package uninorte.test;
2
3  public class MainExample {
4
5      public static void main(String[] args) {
6
7          int entrada = 10;
8
9          int[] array = new int[entrada];
10
11         for (int i = 0; i <= 10; i++) {
12             array[i] = i;
13             System.out.println(i);
14         }
15     }
16
17 }
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 10
at uninorte.test.MainExample.main(MainExample.java:12)
```

Então se fez a luz do Teste de Software

Breve História do Teste de Software

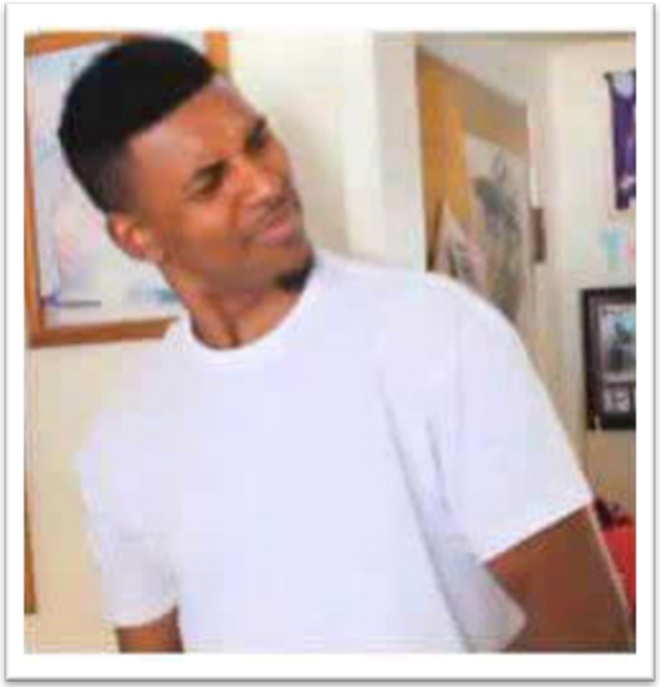


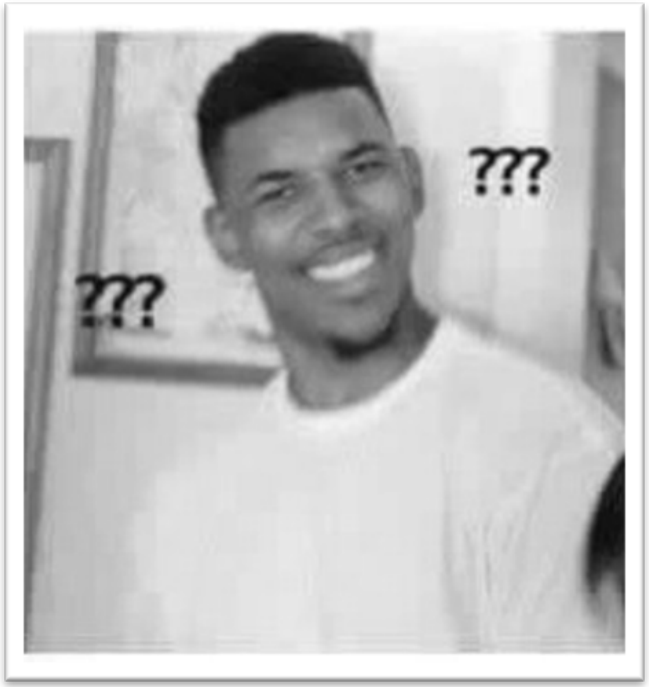
Década	Eventos		
1960 - 1980	1961 - Computer Programming Fundamentals (Leeds e Weinberg). O livro apresenta **um capítulo** sobre **teste de software**.	1969 - **"Teste mostra a presença e não a ausência de defeitos"**, Dijkstra usa essa afirmação falando em uma conferência para o comitê de ciência da OTAN na Itália.	1979 - Glenford Myers publica o **primeiro livro** somente sobre **Teste de Software** chamado "A arte de testar software".
1980 - 1990	1983 - A norma **IEEE 829**, primeira versão do padrão de documentação de teste de software é pulbicada.	1986 - Paul Book publica **modelo V**.	1990 - Taxonomia de defeitos Boris Beizer e **Paradoxo do Pesticida**

Breve História do Teste de Software

Década	Eventos		
1990 - 2000	1991 - **ISO 9126** (Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade e Portabilidade)	1995 - Daniel Mosley aplica pela primeira vez o conceito de tabelas de decisão em teste de software.	1999 - Martin Pol e Koomen lançam o modelo Test Process Impromement voltado para melhoria de processos de teste de software.
2000- 2010	2002 - Criado na Europa e atualmente com sede na Bélgica o International Software Testing Qualifications Board órgão responsável pelo exame de certificação **ISTQB Certified Tester**.	2003 - Lançado por Emerson Rios e Trayahu Moreira o livro Teste de Software que é o primeiro sobre esse assunto especificamente escrito em português.	2006 - Realizado no Brasil o primeiro exame CBTS – **Certificação Brasileira em Teste de Software**.

E um dia me perguntaram.. Já pensou em Trabalhar com Teste de Software?





E a vida de testador começa mais ou menos assim..

E os desenvolvedores?

E no final das contas..

- Plano de Testes
- Casos de Testes (muitos)
- Relatório de Teste

MURAL

Tópico 14

Plano de Teste

Dinâmica

MURAL

Tópico 17

Caso de Teste

Dinâmica

Mundo do Teste de Software

Critérios de teste

- Partição de Equivalências
- Análise Valor Limite
- Tabela de decisão

Fases ou Níveis de Teste

- Aceitação
- Sistema
- Integração
- Unidade

Tipos de Teste

- Estático
- Estrutural
- Funcional
- Segurança
- Regressão
- Usabilidade
- Desempenho
- Exploratório
- Mutação

Técnicas de Teste

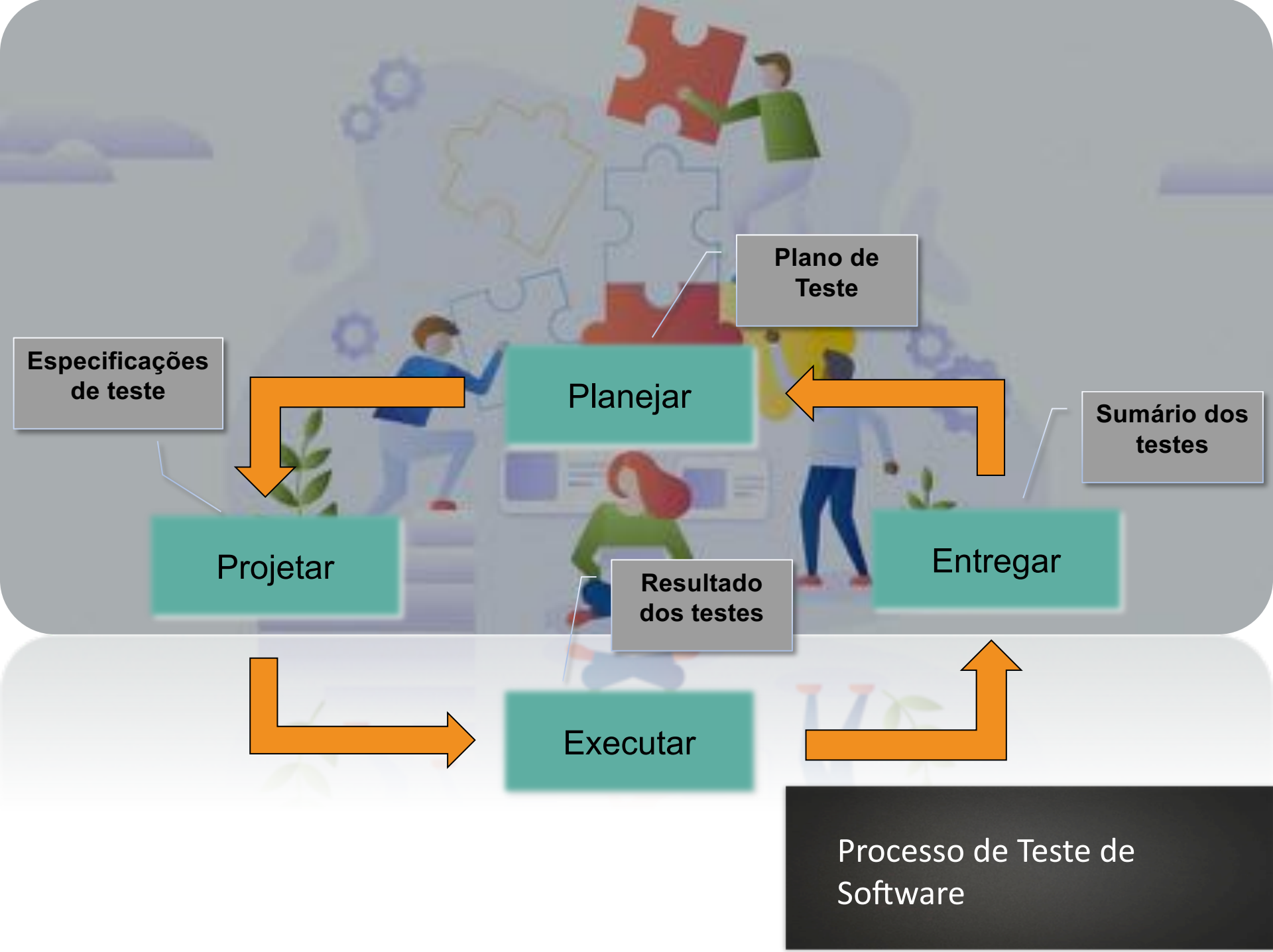
- Caixa Branca
- Caixa Preta
- Caixa Cinza

Ferramentas de Teste

- Selenium
- Appium
- JMeter
- JUnit
- SikuliX

Papéis

- Gerente de Teste
- Líder de Testes
- Arquiteto de Teste
- Analista de Teste
- Testador
- Automatizador de Teste



Processo de Teste de Software

MURAL

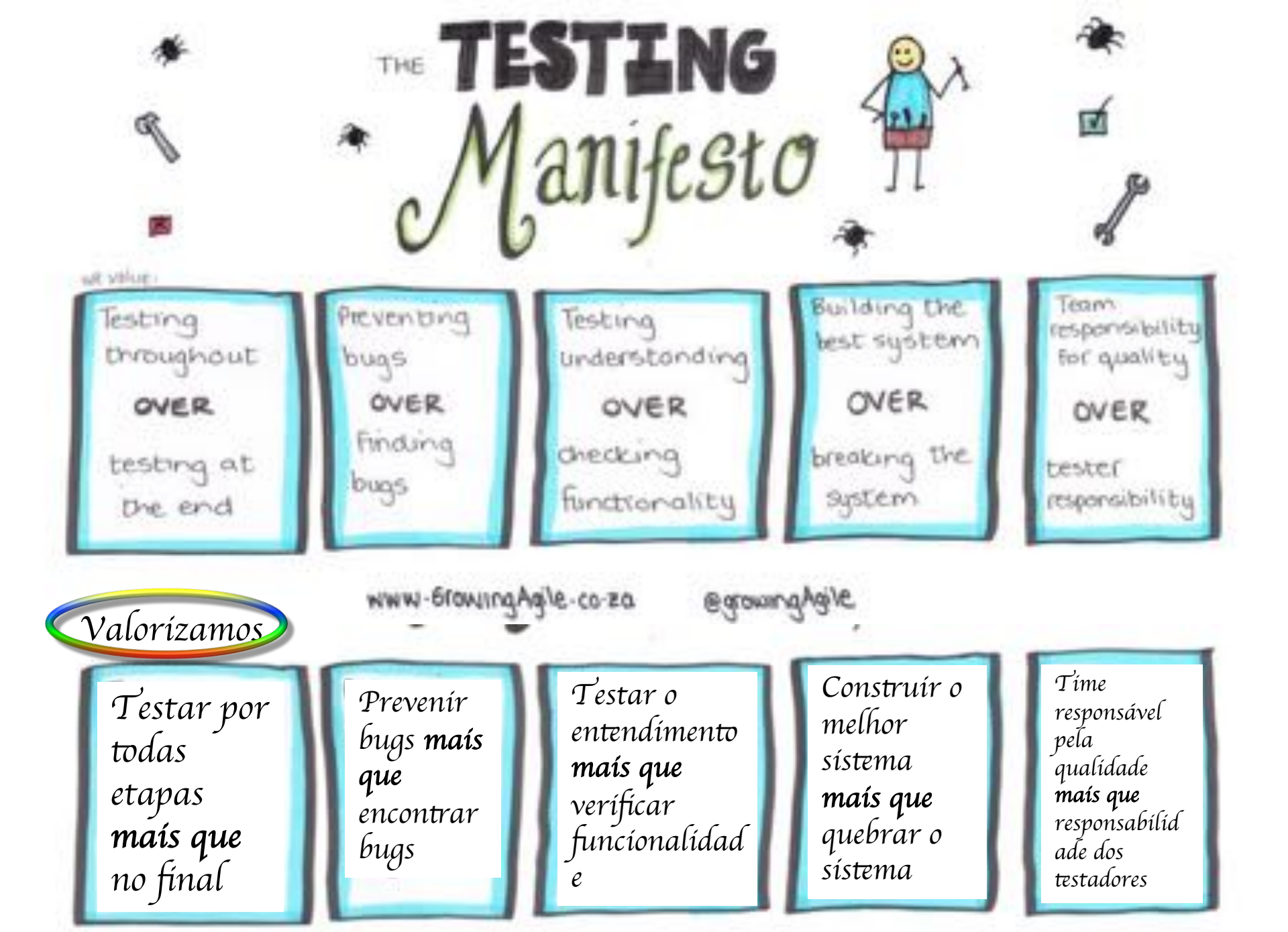
Tópico 22

Programa Identifier

Dinâmica

Teste Ágil

- Teste ágil é uma prática de Teste de Software que segue os princípios do desenvolvimento ágil

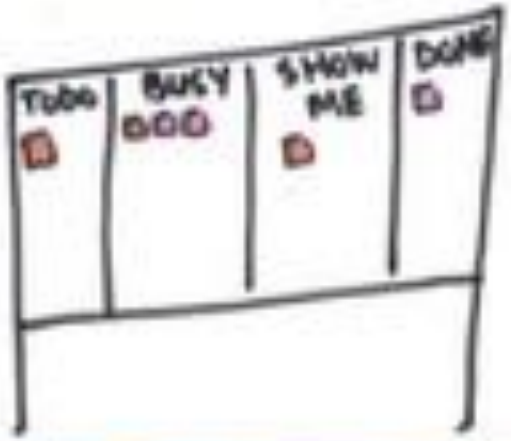

THE TESTING Manifesto
we value:
Testing throughout OVER testing at the end
Preventing bugs OVER finding bugs
Testing understanding OVER checking functionality
Building the best system OVER breaking the system
Team responsibility for quality OVER tester responsibility
www.GrowingAgile.co.za
@GrowingAgile
Valorizamos
Testar por todas etapas **mais que** no final
Prevenir bugs **mais que** encontrar bugs
Testar o entendimento **mais que** verificar funcionalidade
Construir o melhor sistema **mais que** quebrar o sistema
Time responsável pela qualidade **mais que** responsabilidade dos testadores

PREVINIR BUGS!

Teste é uma fase

Teste é uma ATIVIDADE!

Eu faço parte do quadro

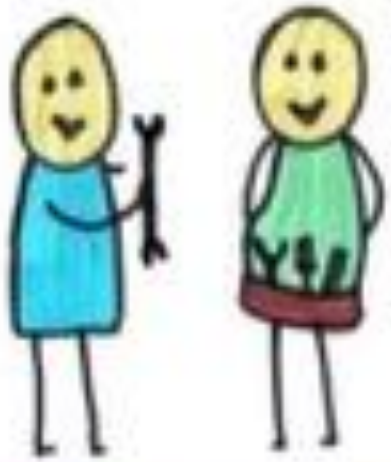


Você é um verificador?

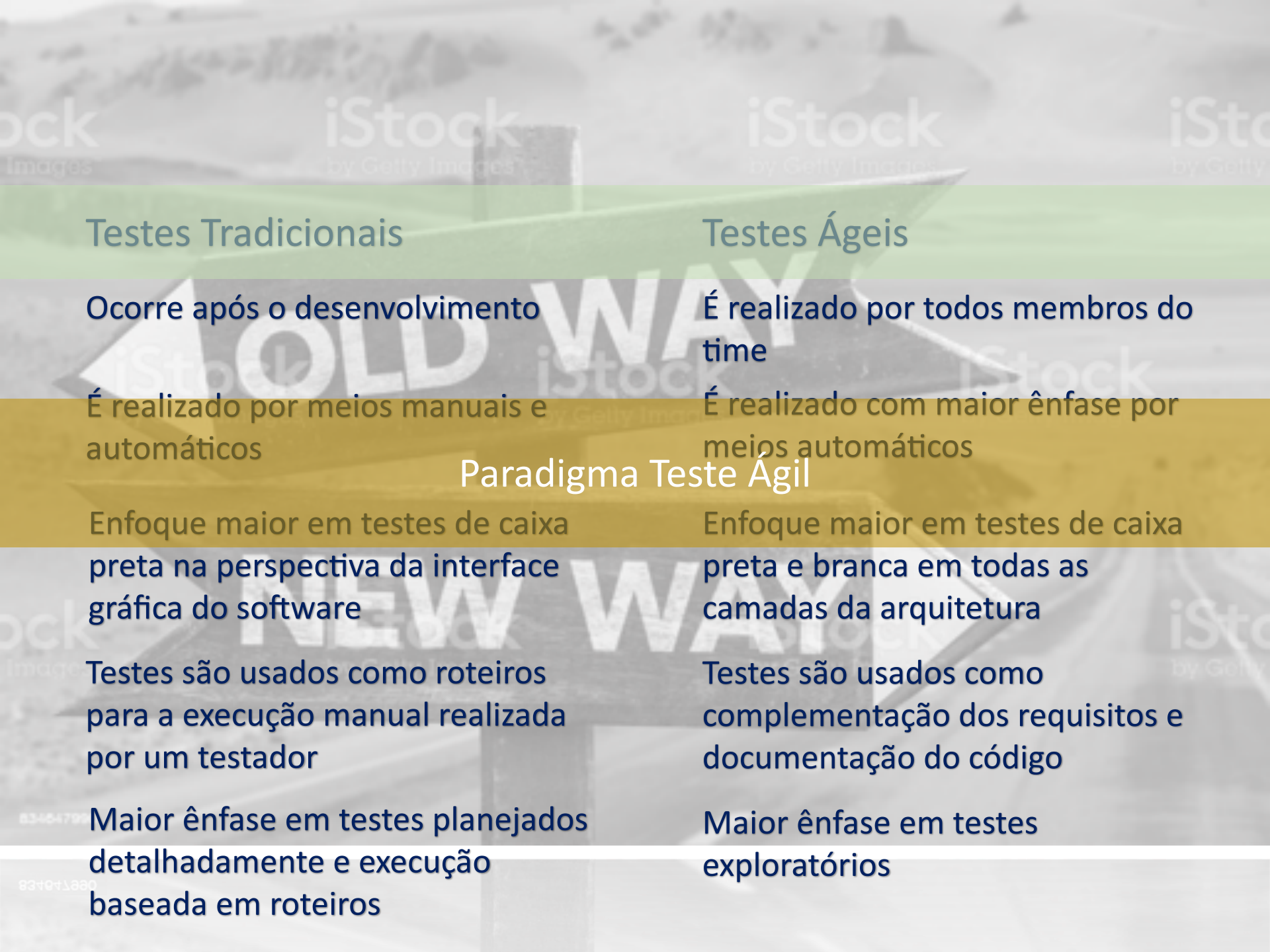
Seja um testador!

Quebrando o sistema

Ajudar a construir o MELHOR sistema



Todo time é responsável pela qualidade



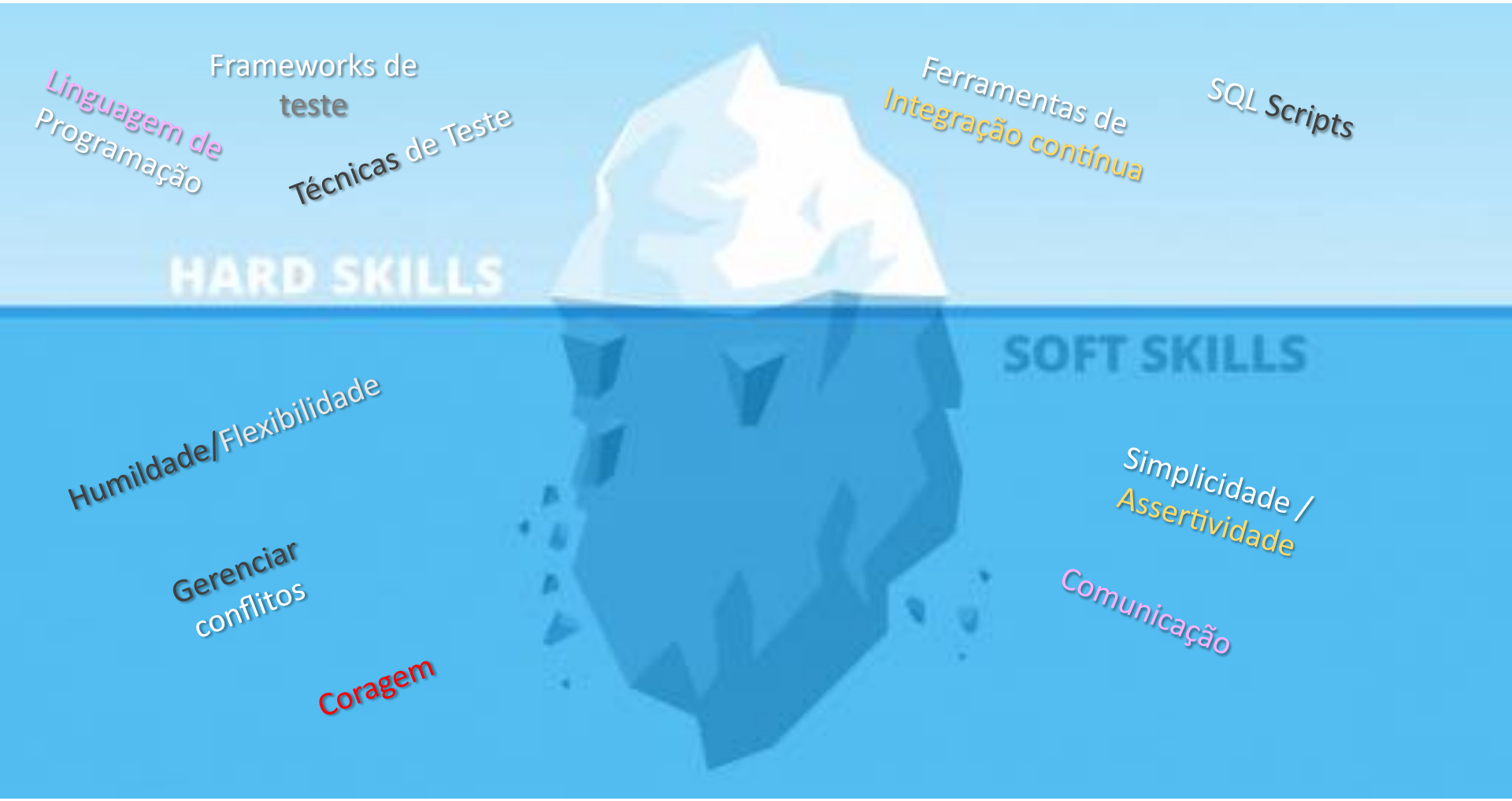
Paradigma Teste Ágil

"Nós definimos o Agile Tester nesta forma: um profissional de teste que abraça as mudanças, colabora bem com pessoas técnicas de de negócios e entende o conceito de utilizar testes para documentar os requisitos e guiar o desenvolvimento"

Lisa Crispin e Janet Gregory

- Qual é o meu papel e as minhas responsabilidades?
- Como eu posso atuar mais próximo do desenvolvedor?
- O que devo fazer manualmente ou automaticamente?
- Eu tenho que programar?
- Como posso testar em ciclos curtos de desenvolvimento?
- Como posso testar em um ambiente onde a mudança é constante?
- Como posso testar sem requisitos detalhados?

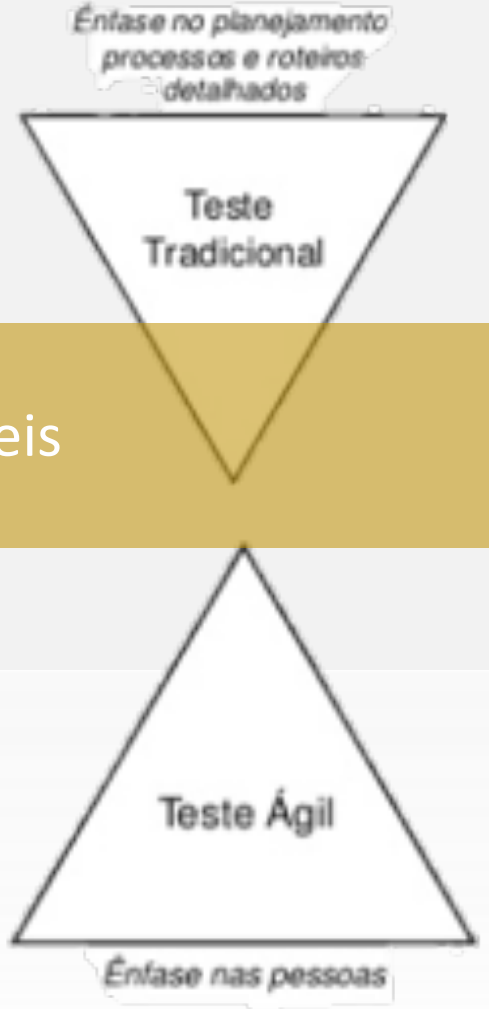
Premissas Testador Ágil



http://agile.pub/assuntos-diversos/as-5-maiores-soft-skills-para-um-agile-tester/

Planejamento dos Testes Ágeis

- Guias e diretrizes (foco na intenção do que vai ser testado)
- Planilhas
- Checklists
- Conversa cara a cara



- Pode-se planejar focando na exploração por meios de:
  - Mapas mentais
  - Testes de aceitação
  - Automatização ao invés de testes tradicionais com roteiros

ACEITAÇÃO

Dado que eu sou um cliente do Brasil

Quando eu comprar um produto da categoria Eletrônico

Então eu vejo o frete grátis

TESTE

Dado que eu sou um cliente do Brasil

Quando eu clico no menu Eletrônicos

E eu adiciono um produto no carinho

Então e vejo o carrinho de compras

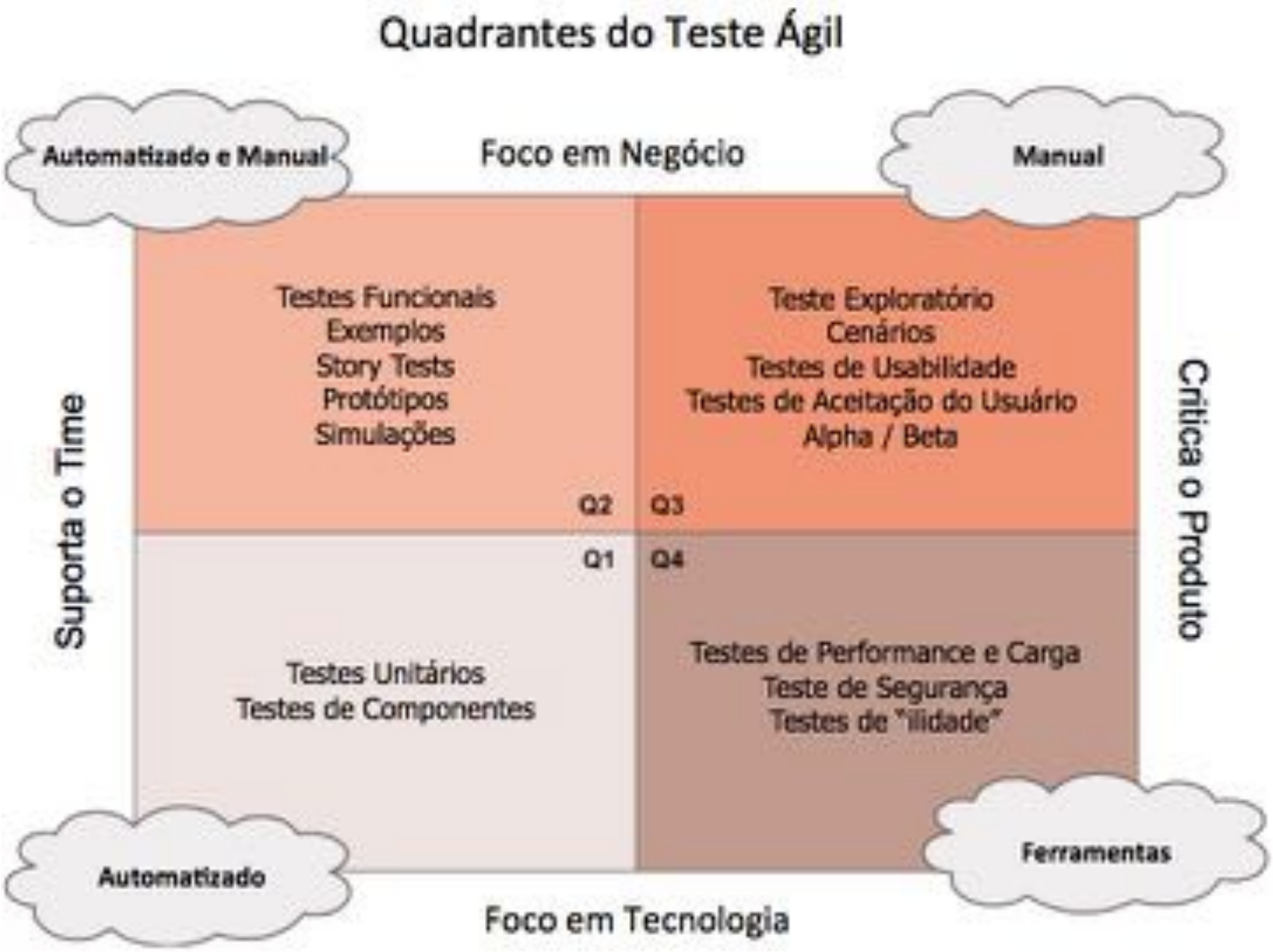
Quando eu informo as condições de pagamento

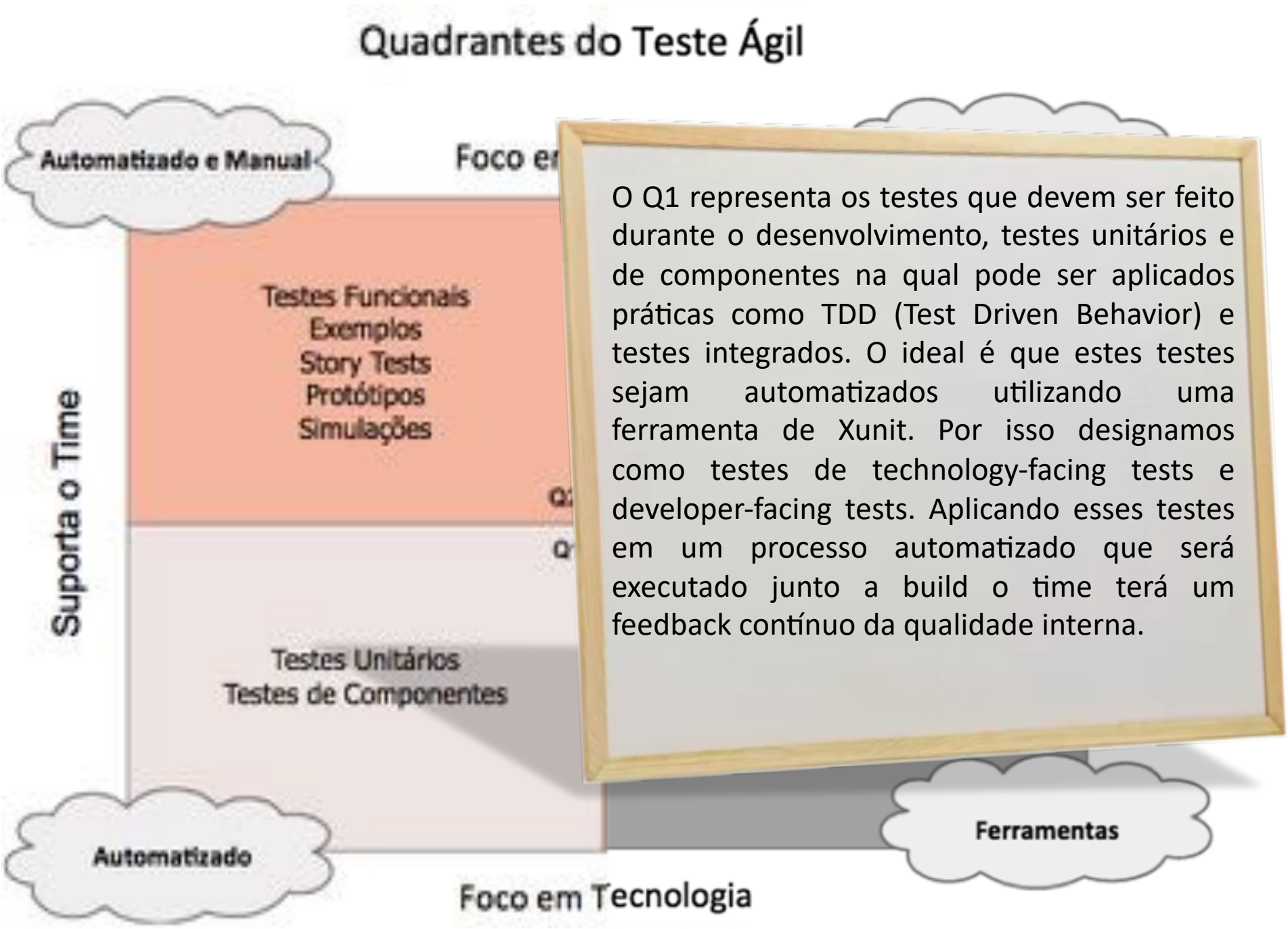
Então eu vejo que o frete é grátis

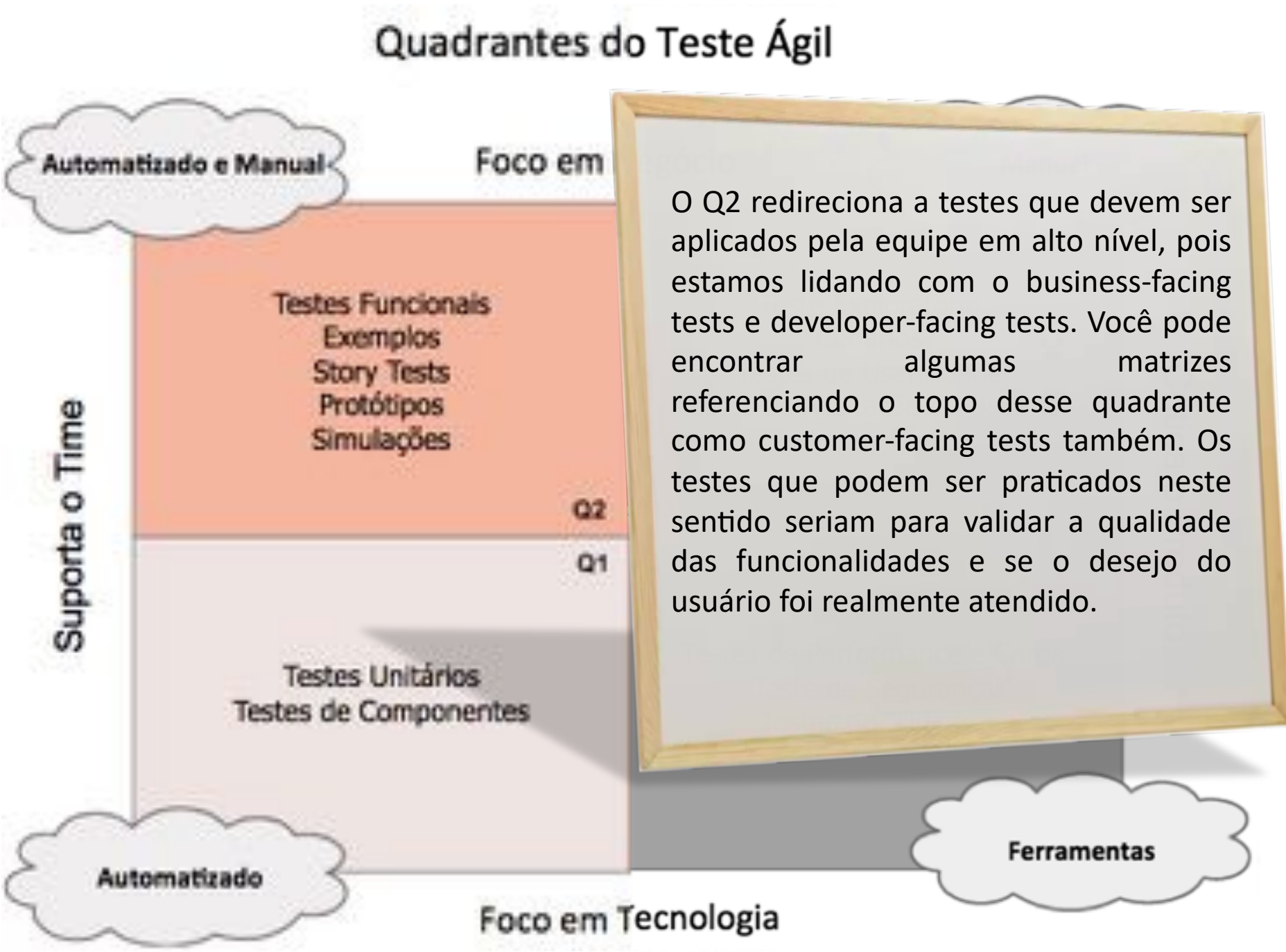
Estratégia Testes Ágeis x Testes Tradicionais

- Os objetivos são os mesmos
  - Para confirmar se o software faz o que ele deve fazer
  - Para confirmar se o software não faz o que ele não deveria fazer
  - Para aferir o atendimento a um atributo de qualidade (implícito e explícito)
  - Encontrar defeitos
- A diferença é a abordagem (mais leve, mais fluída, mais proativa)

A chave é a mudança do comportamento

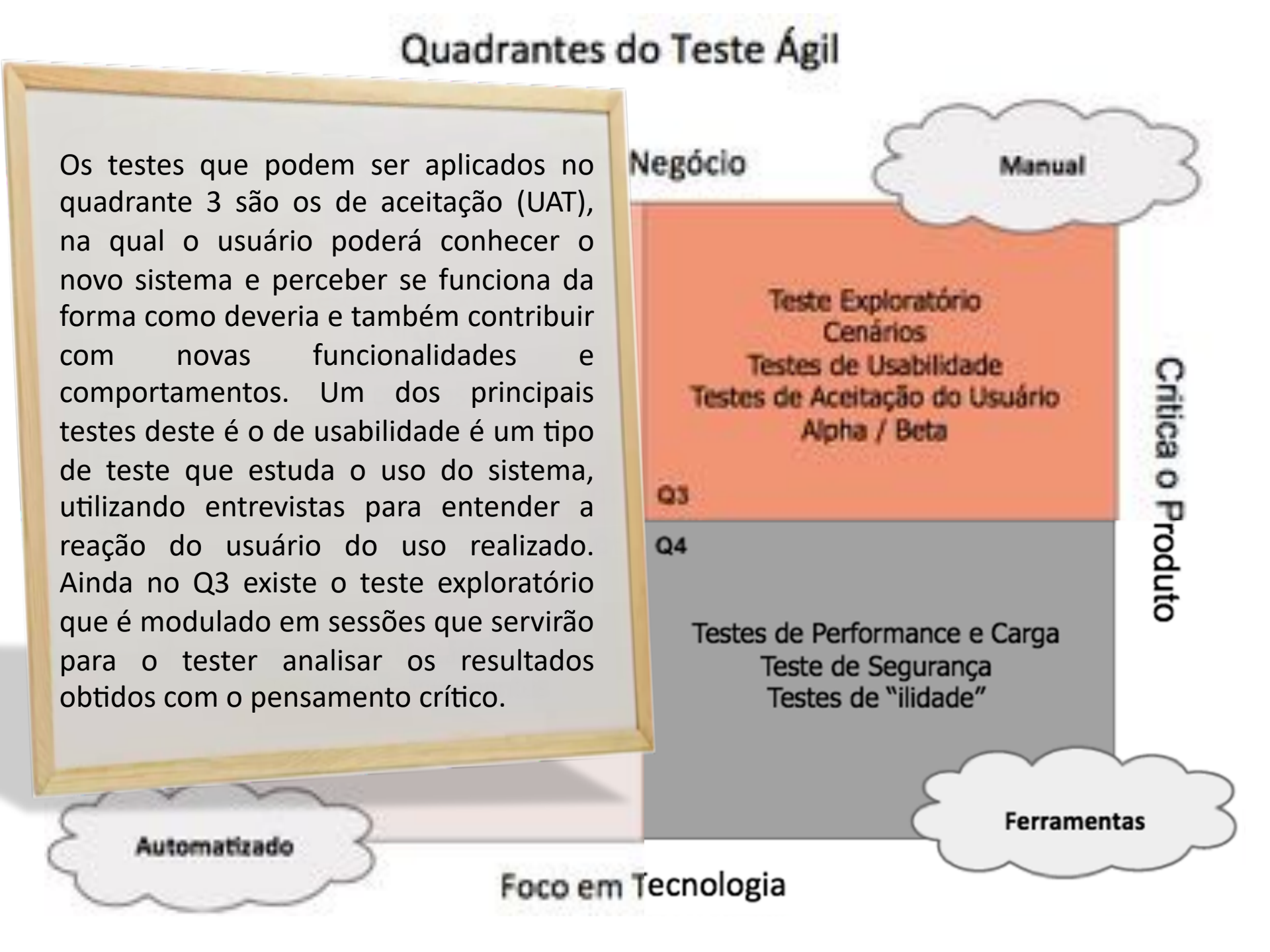

Quadrantes do Teste Ágil
Automatizado e Manual
Foco em Negócio
Manual
Testes Funcionais
Exemplos
Story Tests
Protótipos
Simulações
Teste Exploratório
Cenários
Testes de Usabilidade
Testes de Aceitação do Usuário
Alpha / Beta
Q2
Q3
Q1
Q4
Suporta o Time
Critica o Produto
Testes Unitários
Testes de Componentes
Testes de Performance e Carga
Teste de Segurança
Testes de "ilidade"
Automatizado
Ferramentas
Foco em Tecnologia


Quadrantes do Teste Ágil
Automatizado e Manual
Foco em
Testes Funcionais
Exemplos
Story Tests
Protótipos
Simulações
Suporta o Time
Testes Unitários
Testes de Componentes
Automatizado
Ferramentas
Foco em Tecnologia
O Q1 representa os testes que devem ser feito durante o desenvolvimento, testes unitários e de componentes na qual pode ser aplicados práticas como TDD (Test Driven Behavior) e testes integrados. O ideal é que estes testes sejam automatizados utilizando uma ferramenta de Xunit. Por isso designamos como testes de technology-facing tests e developer-facing tests. Aplicando esses testes em um processo automatizado que será executado junto a build o time terá um feedback contínuo da qualidade interna.


Quadrantes do Teste Ágil
Automatizado e Manual
Foco em
Testes Funcionais
Exemplos
Story Tests
Protótipos
Simulações
Q2
Q1
Suporta o Time
Testes Unitários
Testes de Componentes
Automatizado
Ferramentas
Foco em Tecnologia
O Q2 redireciona a testes que devem ser aplicados pela equipe em alto nível, pois estamos lidando com o business-facing tests e developer-facing tests. Você pode encontrar algumas matrizes referenciando o topo desse quadrante como customer-facing tests também. Os testes que podem ser praticados neste sentido seriam para validar a qualidade das funcionalidades e se o desejo do usuário foi realmente atendido.

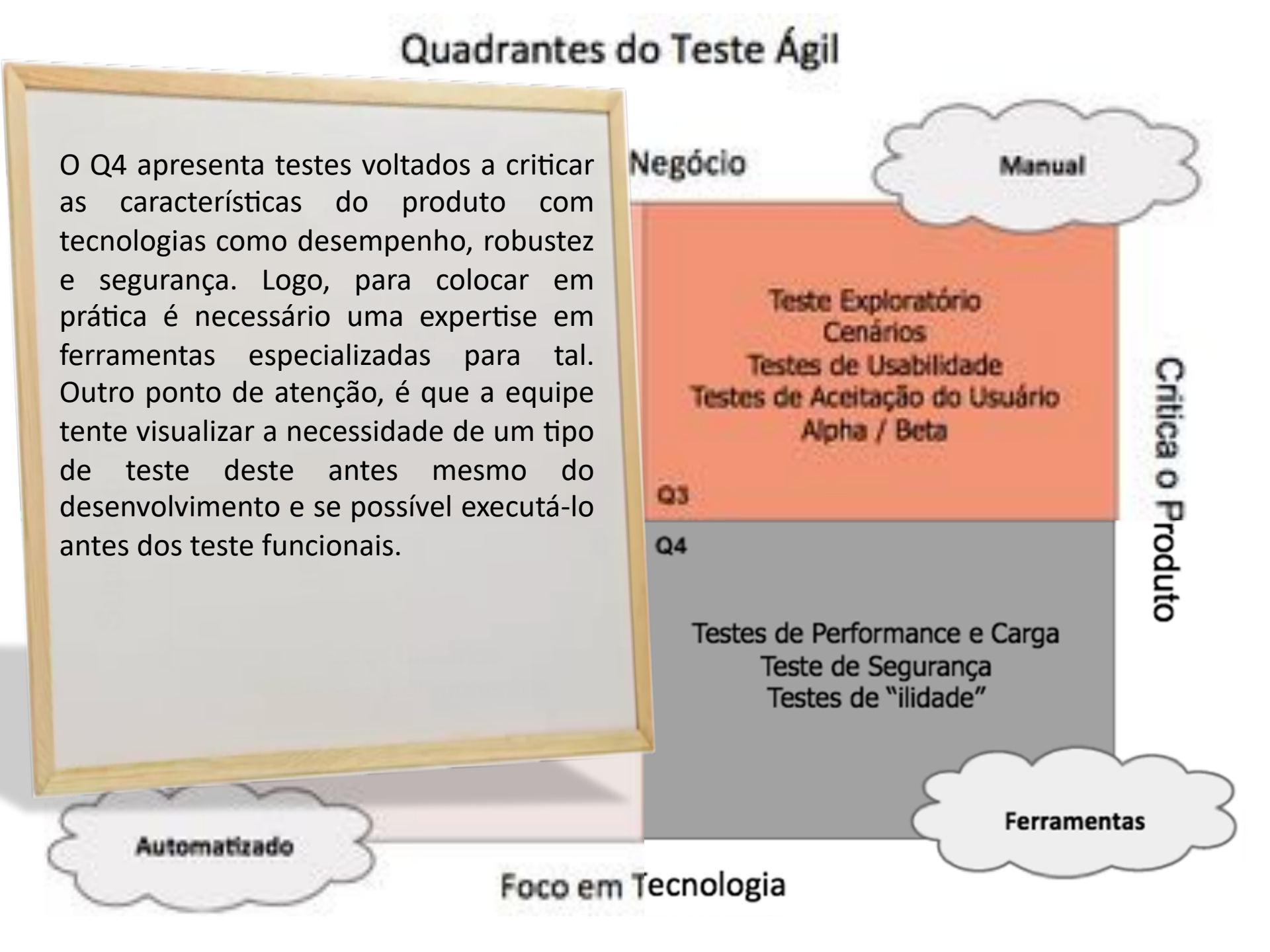
Quadrantes do Teste Ágil

Os testes que podem ser aplicados no quadrante 3 são os de aceitação (UAT), na qual o usuário poderá conhecer o novo sistema e perceber se funciona da forma como deveria e também contribuir com novas funcionalidades e comportamentos. Um dos principais testes deste é o de usabilidade é um tipo de teste que estuda o uso do sistema, utilizando entrevistas para entender a reação do usuário do uso realizado. Ainda no Q3 existe o teste exploratório que é modulado em sessões que servirão para o tester analisar os resultados obtidos com o pensamento crítico.



Quadrantes do Teste Ágil

O Q4 apresenta testes voltados a criticar as características do produto com tecnologias como desempenho, robustez e segurança. Logo, para colocar em prática é necessário uma expertise em ferramentas especializadas para tal. Outro ponto de atenção, é que a equipe tente visualizar a necessidade de um tipo de teste deste antes mesmo do desenvolvimento e se possível executá-lo antes dos teste funcionais.




Testes Exploratórios
Testes de Aceitação
Testes de Integração de Sistema
Testes de Unidade
Testes de Integração em Par
Análise estática de código
Integração Contínua – Automação de testes
Baixo Nível
Alto Nível

MURAL

Tópico 28

Teste Ágil Quadrantes

Dinâmica

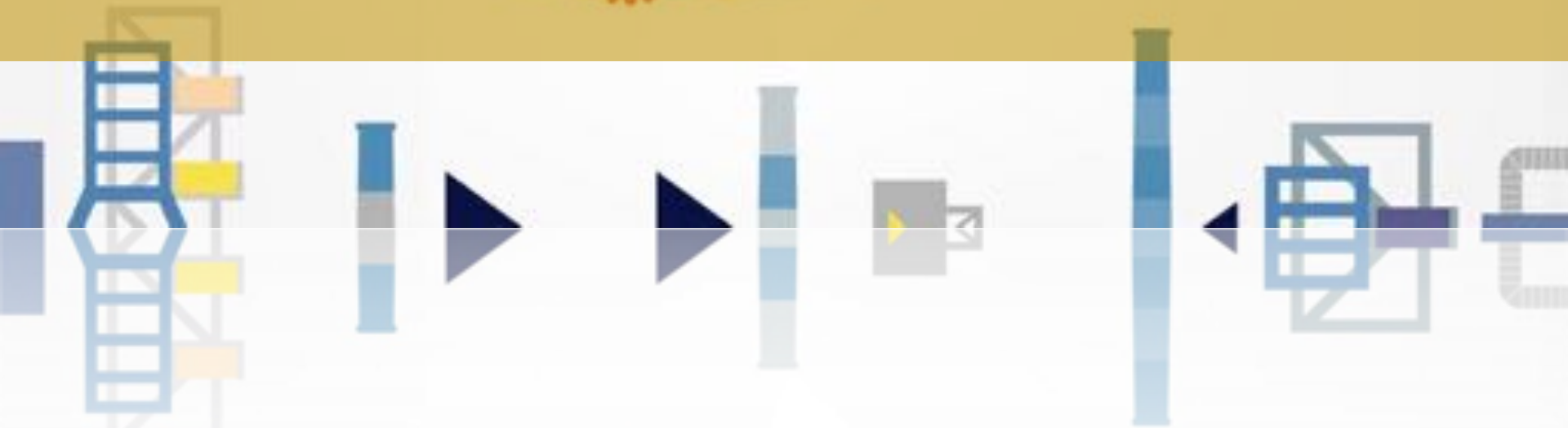
MURAL

Tópico 29

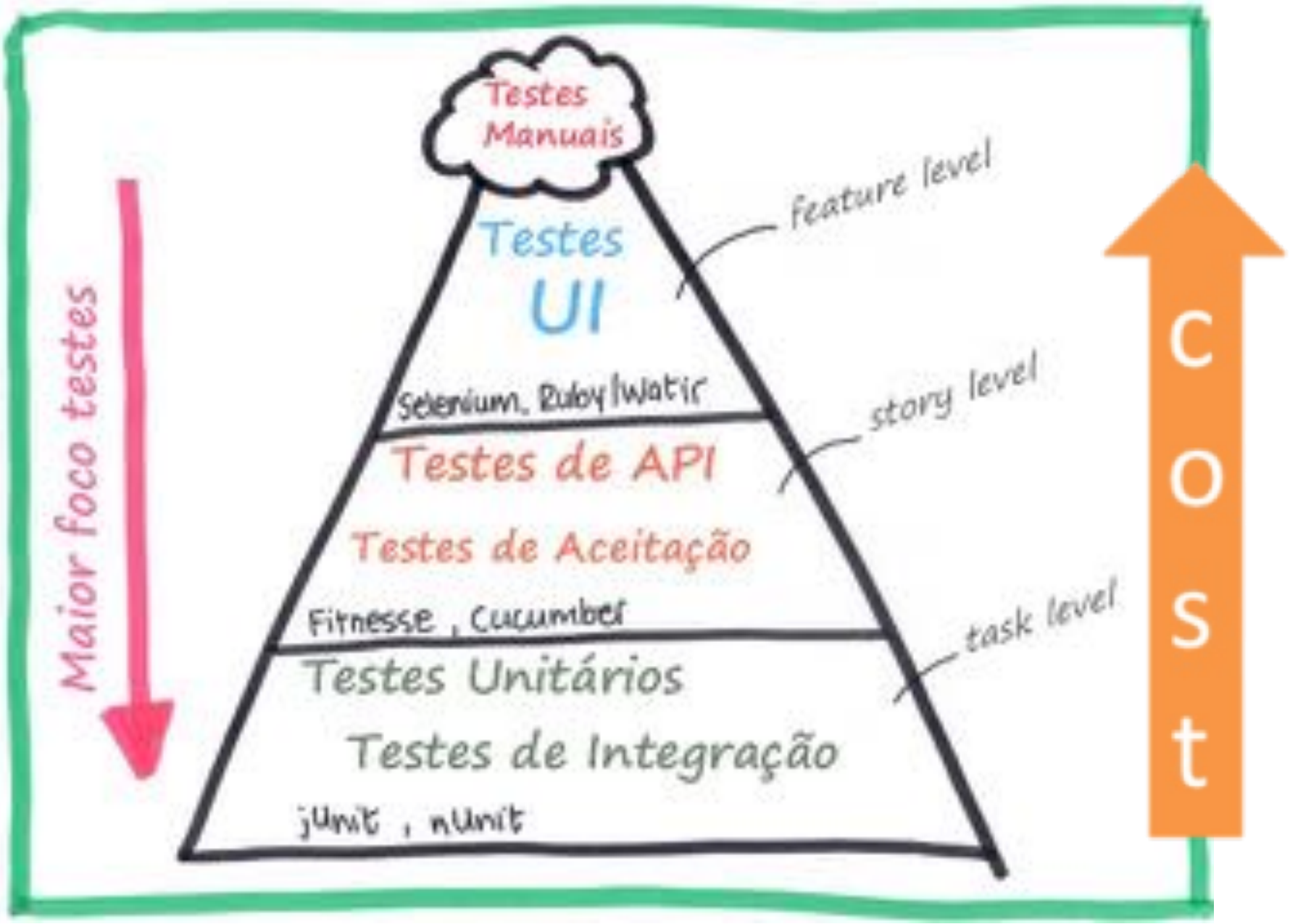
Teste Ágil Automação

Dinâmica

Automação de testes



- A automação viabiliza ciclos curtos de entrega
- A automação pode fazer parte de um ciclo de integração contínua fornecendo feedback contínuo
- A automação oferece uma rede de segurança por meio de regressões completas
- A automação permite a implementação do conceito DRY (Don’t Repeat Yourself) e libera as pessoas para realizarem tarefas mais criativas ao invés de terem que executar testes manuais, enfadonhos e repetitivo


Testes Manuais
Testes UI
Selenium, Ruby/Watir
Testes de API
Testes de Aceitação
Fitnesse , Cucumber
Testes Unitários
Testes de Integração
junit , nUnit
feature level
story level
task level
Maior foco testes
cost

Tópico 30

Teste Ágil Participação

Dinâmica


backlog grooming
mindmap + fluxo de valor
entendimento
Porque (3w)
opções
Como testar (regras)
Planejamento da sprint
1
Entendimento
exemplos
Como será testado
afeta regressão
2
tarefas de teste
automação
Em qual nível
dados de teste/ambiente


Durante a sprint
Coluna de review (show me)
qualquer bug
definition of done
pareamento com dev
reuniões diárias
questionar tarefas
procurar falta de entendimento
tarefas de teste primeiro
review
usabilidade
resolver problemas de negócio

MURAL

Tópico 30

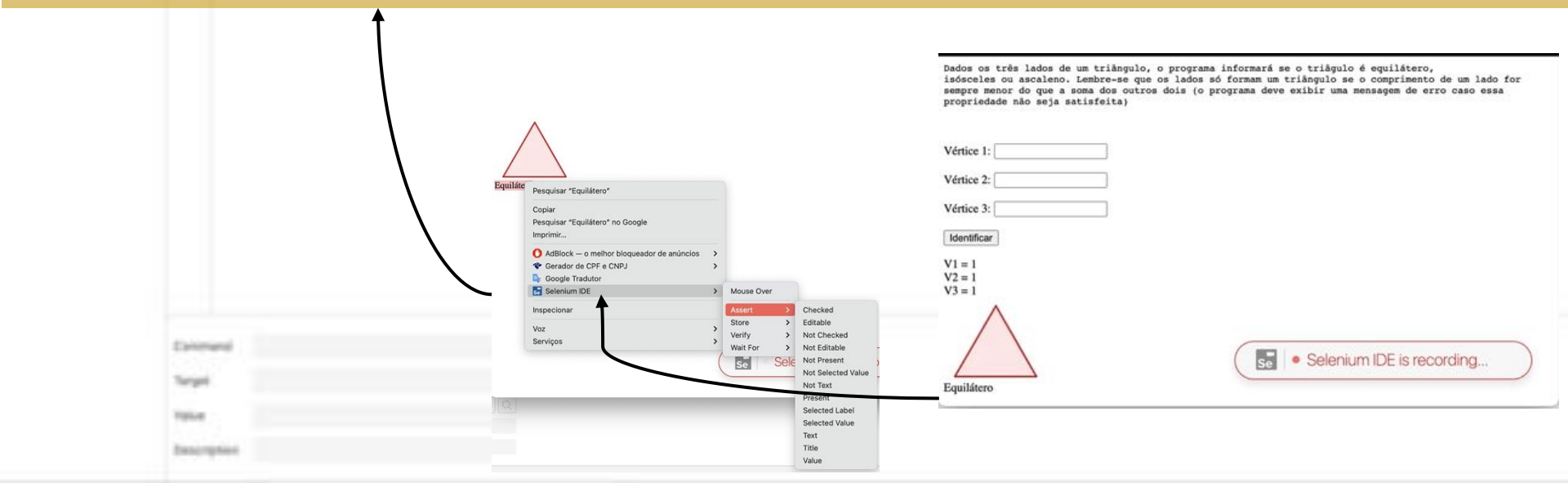
Teste UI - Triângulo

Dinâmica

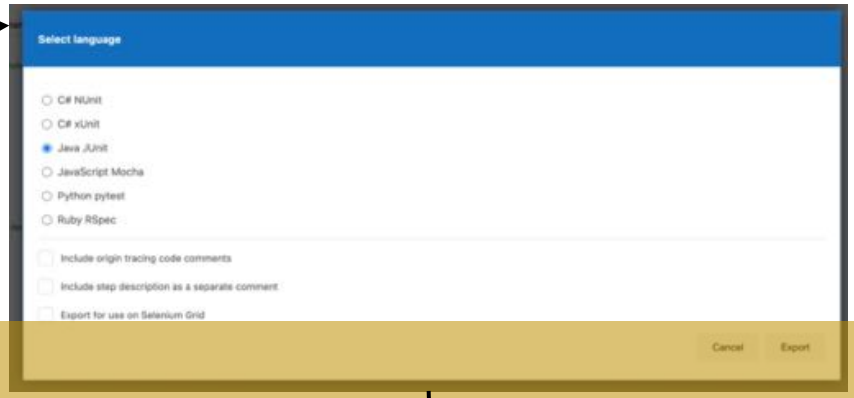
<- Criar Teste

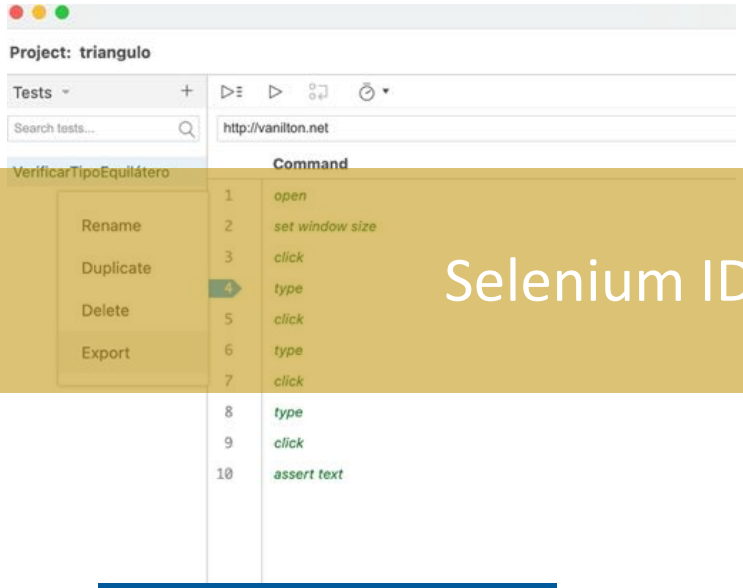
Gravar Teste ->

Selenium IDE – Chrome Extension



Selenium IDE – Exportando Teste



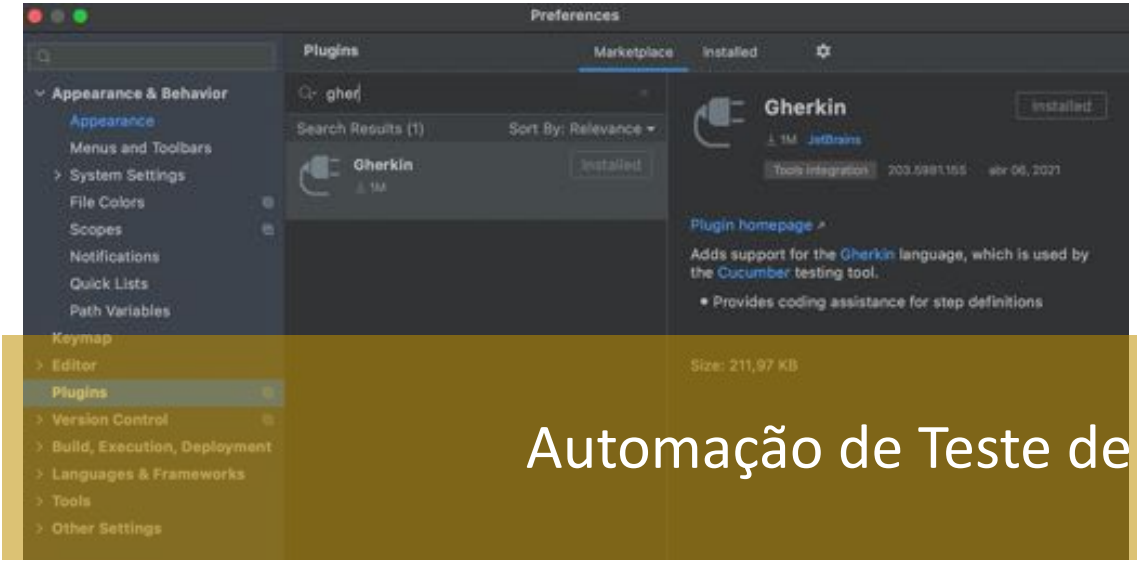


```java
public class VerificarTipoEquilteroTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void verificarTipoEquiltero() {
    driver.get("http://vanilton.net/triangulo/");
    driver.manage().window().setSize(new Dimension(839, 900));
    driver.findElement(By.name("V1")).click();
    driver.findElement(By.name("V1")).sendKeys("1");
    driver.findElement(By.name("V2")).click();
    driver.findElement(By.name("V2")).sendKeys("1");
    driver.findElement(By.name("V3")).click();
    driver.findElement(By.name("V3")).sendKeys("1");
    driver.findElement(By.cssSelector("input:nth-child(13)")).click();
    assertThat(driver.findElement(By.cssSelector(".vsc-initialized > div:nth-child(7)")).getText(), is("Equilátero"));
  }
}
```

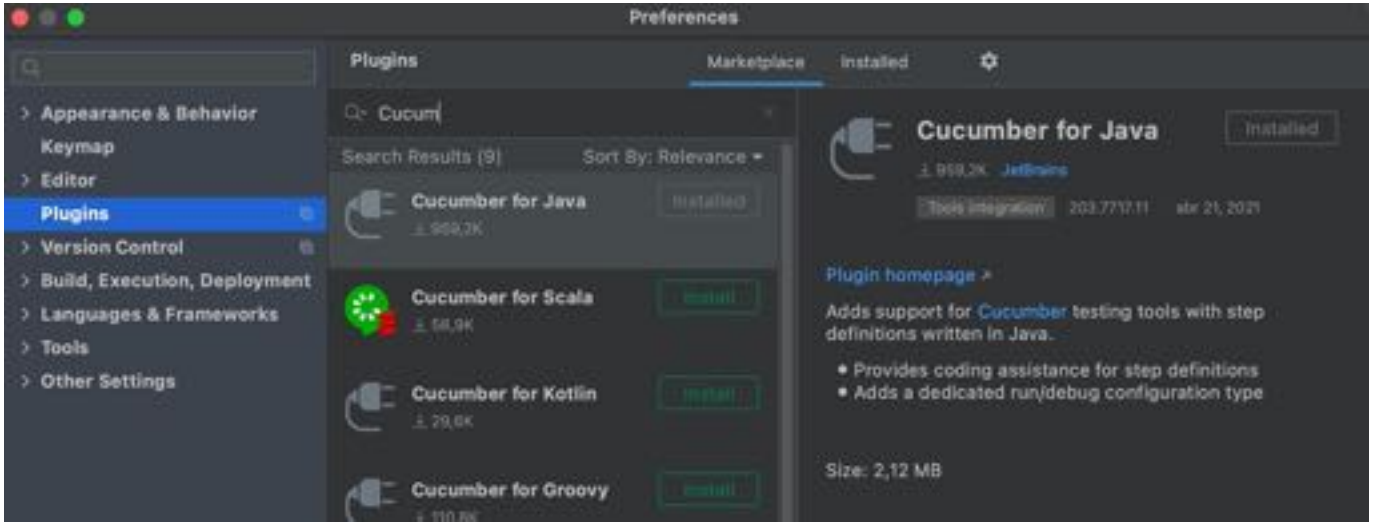

Selenium Java – Implementando Teste Funcional

SikuliX – Implementando Teste Funcional

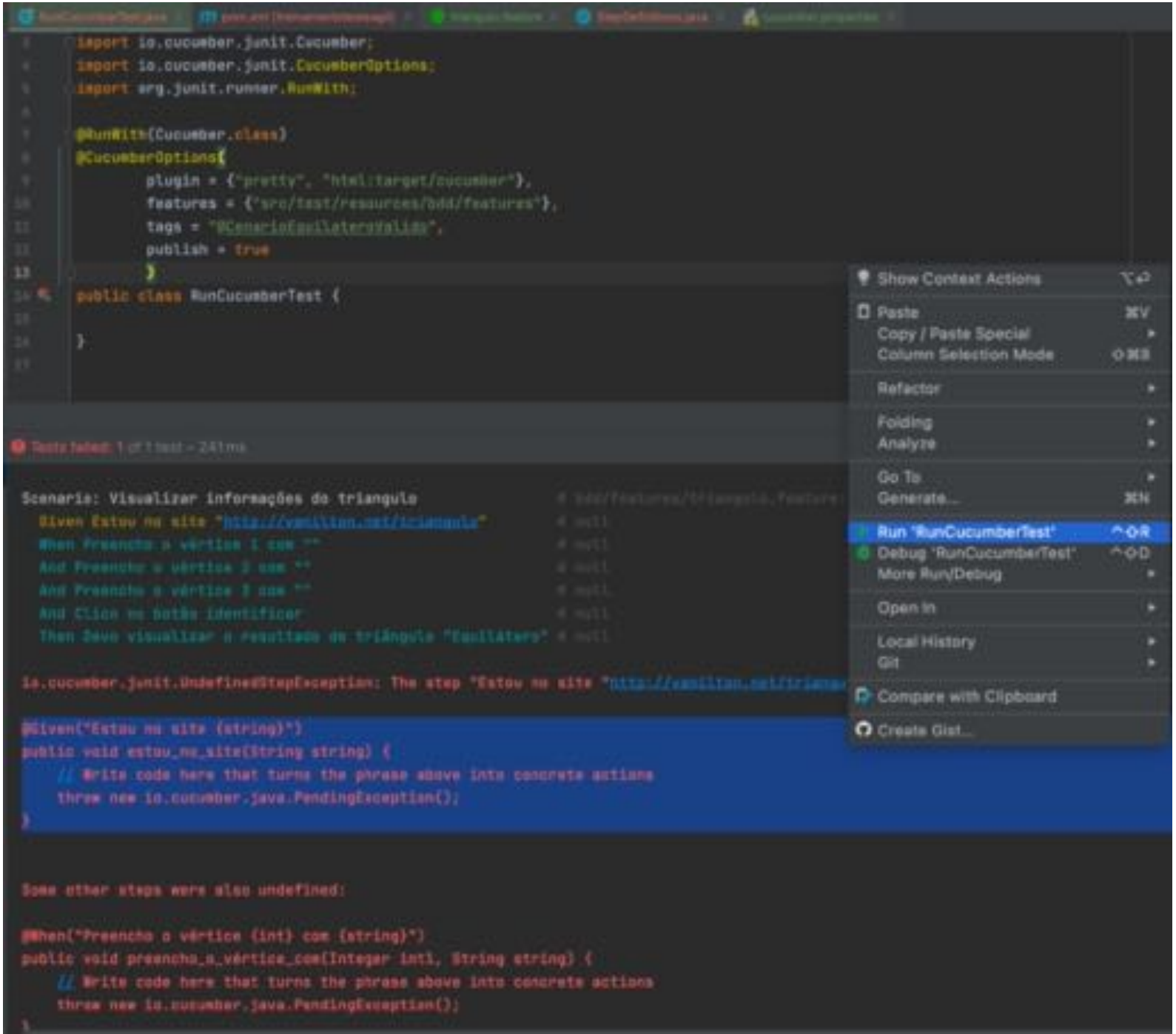
- Implementar os testes para do Programa Triângulo em http://vanilton.net/triangulo
- Utilizar os casos de testes já elaborados



Automação de Teste de Aceitação



```
Feature: Como um PROFESSOR eu gostaria de IDENTIFICAR OS TIPOS DE TRIÂNGULOS para ENSINÁ-LOS AOS MEUS ALUNOS.

  @Cenario1
  Scenario: Visualizar informacoes do triangulo
    Given Estou no navegador "Chrome"
    And Acesso o site "http://vanilton.net/triangulo"
    When Preencho o vértice 1 com "3"
    And Preencho o vértice 2 com "3"
    And Preencho o vértice 3 com "3"
    And Clico no botão identificar
    Then Devo visualizar o resultado de triângulo "Equilátero"

  @Cenario2
  Scenario: Visualizar informacoes do triangulo
    Given Estou no navegador "Chrome"
    And Acesso o site "http://vanilton.net/triangulo"
    When Preencho triângulo
      |vertice|valor |
      |1      |3     |
      |2      |3     |
      |3      |3     |
    And Clico no botão identificar
    Then Devo visualizar o resultado de triângulo "Equilátero"
```


import io.cucumber.junit.Cucumber;
import io.cucumber.junit.CucumberOptions;
import org.junit.runner.RunWith;
@RunWith(Cucumber.class)
@CucumberOptions(
plugin = {"pretty", "html:target/cucumber"},
features = {"src/test/resources/bdd/features"},
tags = "@CenarioEquilatero",
publish = true
)
public class RunCucumberTest {
}
Show Context Actions
Paste
Copy / Paste Special
Column Selection Mode
Refactor
Folding
Analyze
Go To
Generate...
Run 'RunCucumberTest'
Debug 'RunCucumberTest'
More Run/Debug
Open In
Local History
Git
Compare with Clipboard
Create Gist...
Tests failed: 1 of 1 test – 241 ms
Scenario: Visualizar informações do triangulo
Given Estou no site "http://emilian.net/triangulo"
When Preencho o vértice 1 com ""
And Preencho o vértice 2 com ""
And Preencho o vértice 3 com ""
And Clico no botão identificar
Then Devo visualizar o resultado do triÂngulo "EquiLatero"
io.cucumber.junit.UndefinedStepException: The step "Estou no site "
@Given("Estou no site (string)")
public void estou_no_site(String string) {
// Write code here that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();
}
Some other steps were also undefined:
@When("Preencho o vértice (int) com (string)")
public void preencho_o_vértice_com(Integer int1, String string) {
// Write code here that turns the phrase above into concrete actions
throw new io.cucumber.java.PendingException();

JUnit – Implementando Teste Unitário

- Implementar os testes da classe Identifier localizadas no projeto
- https://github.com/Vanilton18/treinamento_teste_agil

Vamos montar um conjunto de casos de teste para testar o método validateIdentifier(String s). Lembrando que para cada string de teste deve ser verificado o resultado no tipo boolean.

Um identificador válido deve começar com uma letra e conter apenas letras ou dígitos. Além disso, deve ter no mínimo um e no máximo seis caracteres de comprimento.

JMeter – Implementando Teste Serviço

- Criar um projeto com Jmeter e avaliar a performance de acesso a página http://fpftech.com, para 5, 100 e 1000 usuários simultâneos.
- Criar assertiva para avaliar se as requisições atingiram um tempo esperado de 1 segundo para carregamento da página.

Simultâneamente ....

... aprender sobre o software

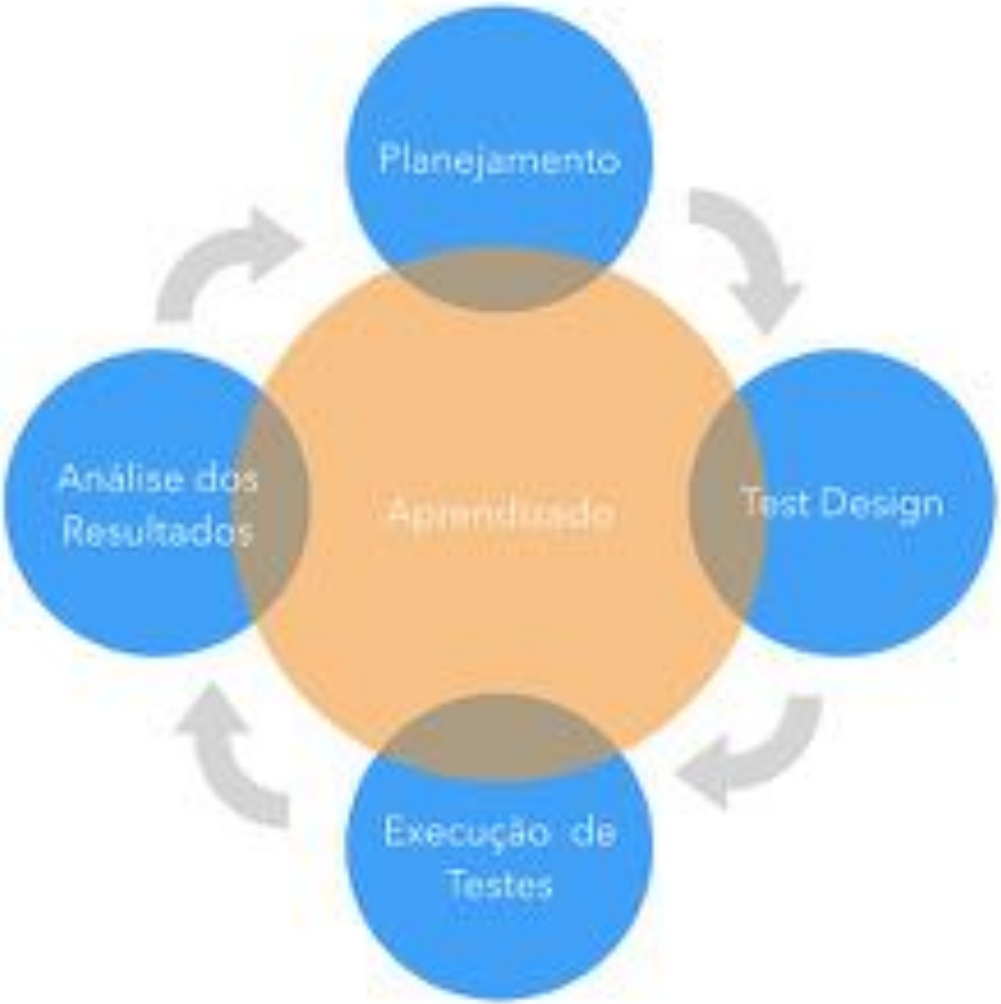
... desenvolver mais testes

... executar testes

Teste Exploratório

Usando o feedback do último teste para executar o próximo!

Ao invés de realizar uma análise sequencial de necessidades, seguida pela concepção, documentação e execução de casos de teste, o teste exploratório foi definido, por **James Bach (2009)**, como um processo em que o testador aprende, elabora o design do teste e executa simultaneamente enquanto explora o produto.



Quando Usar

Para **Bach** (2009) o teste exploratório deve ser utilizado nas seguintes situações

- Necessidade em fornecer feedback rápido sobre um novo produto ou serviço;
- Necessidade de aprender o produto rapidamente; · Diversificar os testes e melhorá-los;
- Encontrar o erro mais importante no menor espaço de tempo;
- Investigar e isolar um defeito específico;
- Investigar a situação de risco em especial, a fim de avaliar a necessidade de testes com script nessa área;

Quando Usar

- Realização de testes quando não existem requisitos;
- Realização de testes quando existe pouco tempo disponível;
- Realização de testes quando não se conhece o aplicativo a ser testado;
- Realização de testes em ambientes pouco testados pelos testes convencionais;
- Identificação dos passos para tentar reproduzir um defeito aleatório;
- Diagnóstico de comportamentos inesperados;
- Investigação de efeitos colaterais;
- Investigação de defeitos semelhantes;
- Medição de riscos;

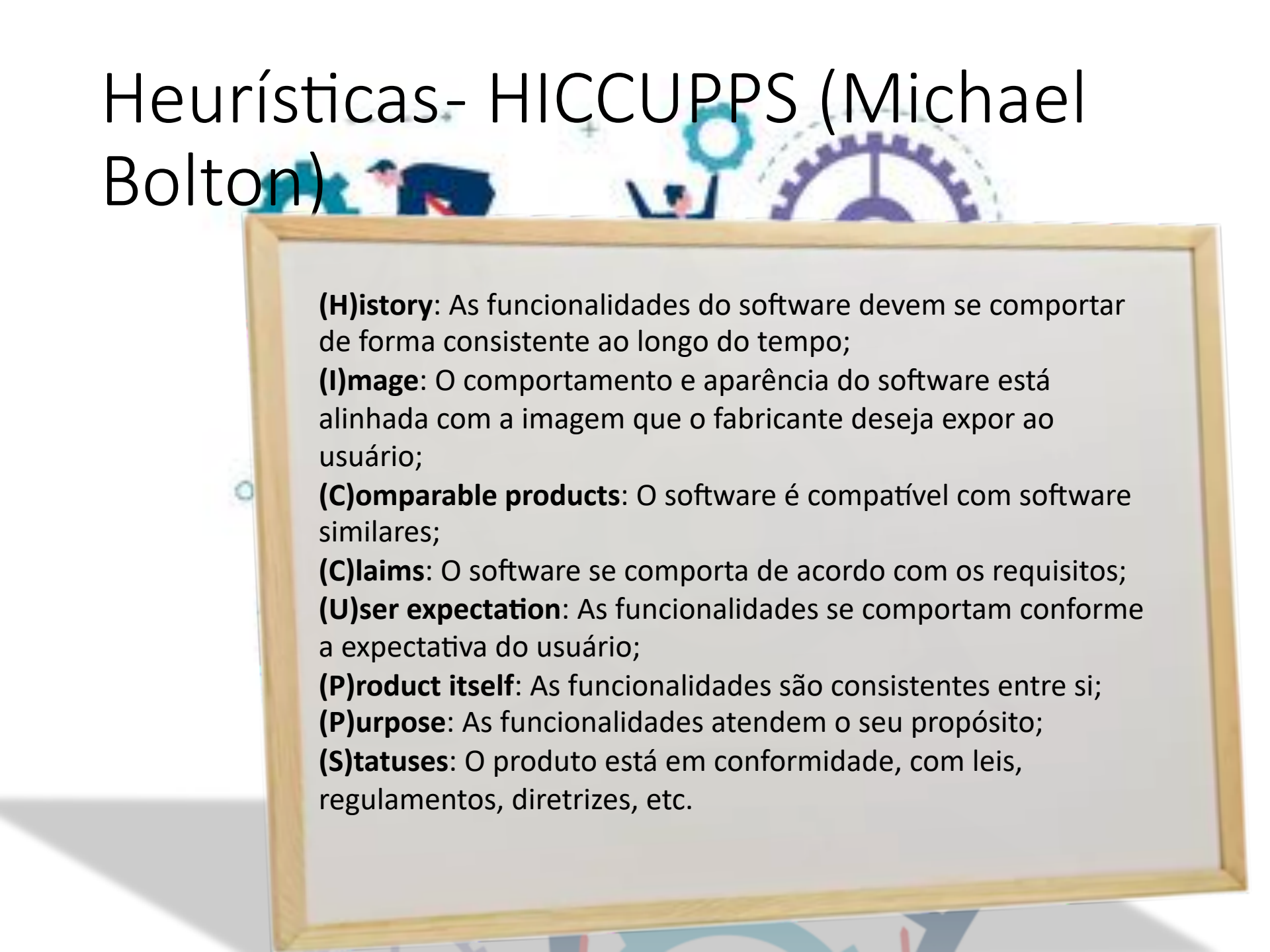
Por quê usar?

- Os testes não são criados com antecedência
- Não segue um roteiro rígido (segue guias e diretrizes)
- É baseado em pensamento estruturado e exploração livre
- É adaptativo e flexível
- Enfoca o aprendizado em paralelo
- A execução do teste é guiada/aprimorada com base em execuções anteriores
- Exige profissionais experientes
- Expande o escopo dos testes tradicionais baseados em roteiros (Injeta/introduz variação aos casos de testes)
- Fluxo imediato de feedback (e correção de curso)
- Amplifica a cobertura dos testes

Guias e Diretrizes

Exploração do produto: descobrir e registrar o objetivo e as funções do produto, tipos de dados processados e áreas de potencial instabilidade. Esta fase depende do entendimento da tecnologia utilizada, informações sobre o produto e usuários, e a quantidade de tempo disponível para fazer o trabalho;
Projeto de teste: determinar as estratégias de operação, observação e avaliação do produto;
Execução do teste: operar o produto, observar o comportamento e utilizar informações para formar hipóteses de como o produto funciona;
Heurísticas: são guias ou regras que ajudam o testador a decidir o que fazer, o que deverá ser testado e como testá-lo;
Resultados: é o resultado do teste. Este é finalizado uma vez que o testador produziu resultados que atendam os requisitos especificados.

Heurísticas- HICCUPPS (Michael Bolton)



(H)istory: As funcionalidades do software devem se comportar de forma consistente ao longo do tempo;
(I)mage: O comportamento e aparência do software está alinhada com a imagem que o fabricante deseja expor ao usuário;
(C)omparable products: O software é compatível com software similares;
(C)laims: O software se comporta de acordo com os requisitos;
(U)ser expectation: As funcionalidades se comportam conforme a expectativa do usuário;
(P)roduct itself: As funcionalidades são consistentes entre si;
(P)urpose: As funcionalidades atendem o seu propósito;
(S)tatuses: O produto está em conformidade, com leis, regulamentos, diretrizes, etc.

Heurísticas- IOSC

(I)nput: Exploração do comportamento do software sob a perspectiva das entradas de dados;
(O)utput: Exploração do comportamento do software sob a perspectiva das saídas de dados;
(S)torage: Exploração do comportamento do software sob a perspectiva dos dados armazenados;
(C)omputation: Exploração do comportamento do software sob a perspectiva da computação/processamento dos dados.

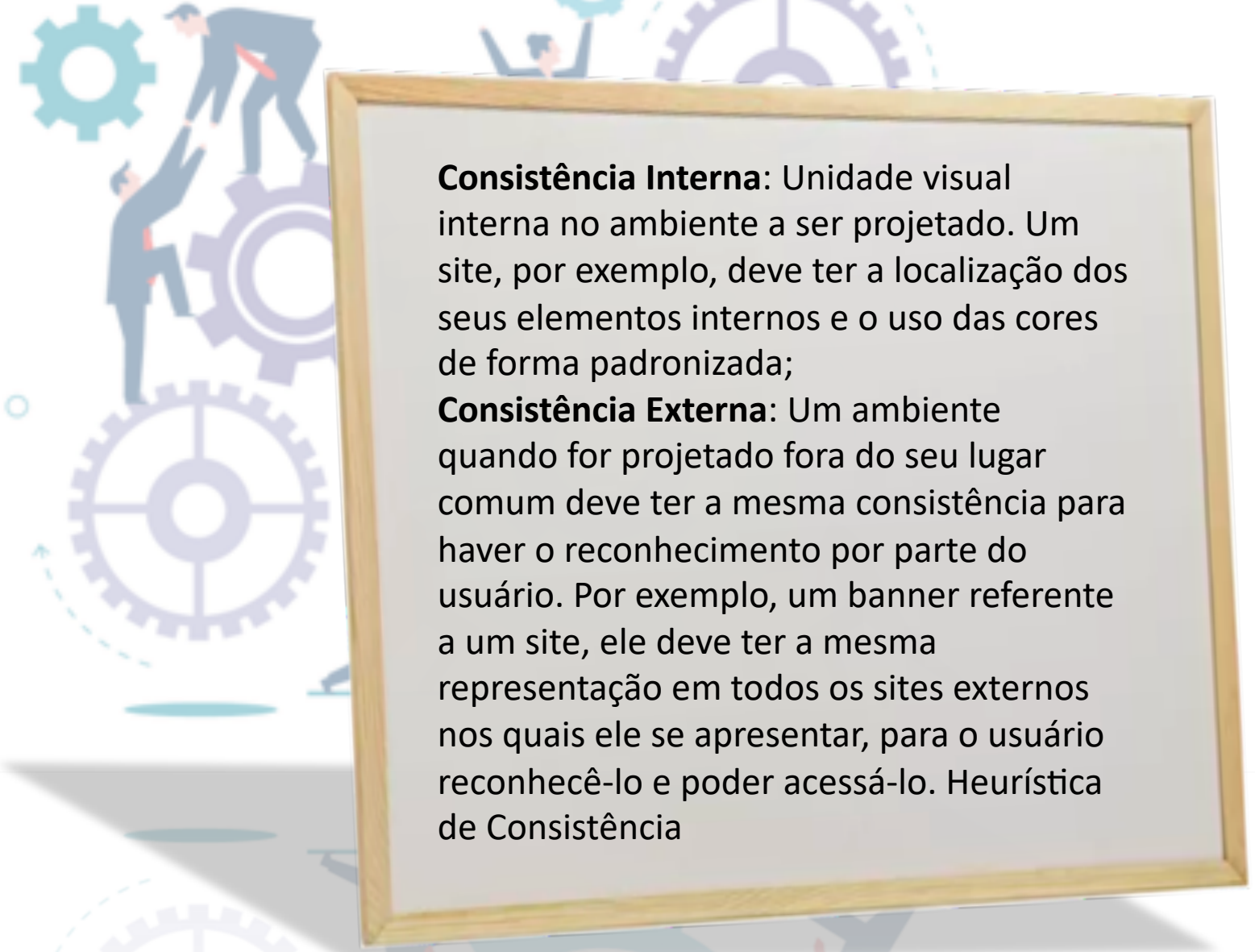
Heurísticas- Consistência

- Consistência é uma das Heurísticas de Nielsen e significa que o mesmo comando ou ação deve ter sempre o mesmo efeito e se localizar sempre no mesmo lugar.
- A consistência melhora o reconhecimento, o aprendizado, a memorização e transmite até mais credibilidade ao usuário do sistema. Existem quatro espécies de consistências:

Consistência Estética: O estilo e a aparência aumentam o reconhecimento e comunicam de uma forma bem mais efetiva a mensagem. Um exemplo de consistência estética seria a mesma escala de cores usadas por empresas do mesmo ramo, facilitando o reconhecimento por parte do usuário/cliente;

Consistência Funcional: Mesmas ações com significados iguais melhoram a usabilidade e a funcionalidade para que os usuários aproveitem ao máximo a interface. Controles remotos e aparelhos tocadores de mp3 possuem a mesma linguagem, havendo consistência de informação

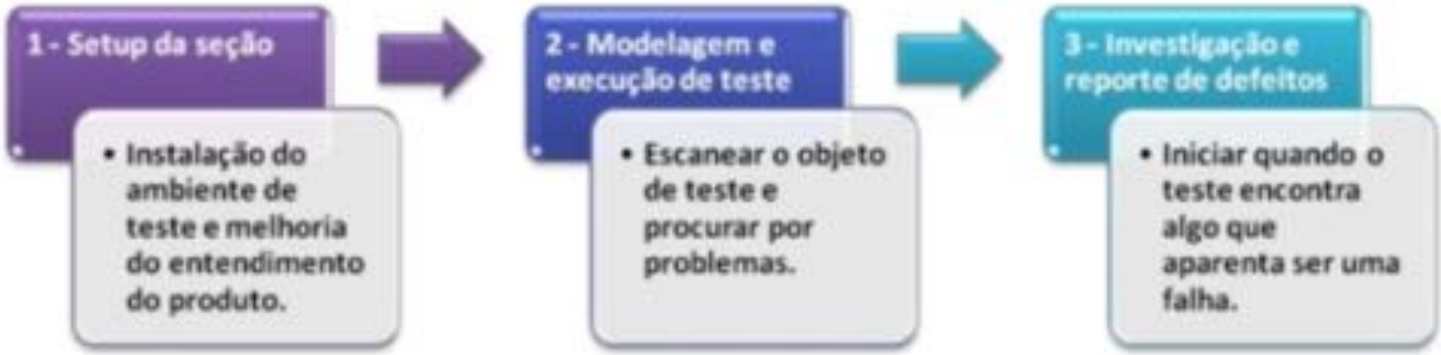
Heurísticas- Consistência



Consistência Interna: Unidade visual interna no ambiente a ser projetado. Um site, por exemplo, deve ter a localização dos seus elementos internos e o uso das cores de forma padronizada;

Consistência Externa: Um ambiente quando for projetado fora do seu lugar comum deve ter a mesma consistência para haver o reconhecimento por parte do usuário. Por exemplo, um banner referente a um site, ele deve ter a mesma representação em todos os sites externos nos quais ele se apresentar, para o usuário reconhecê-lo e poder acessá-lo. Heurística de Consistência

Heurísticas de Nielsen



Planejamento

- Testes Baseados em Seção
- Durante a jornada de estudos sobre o teste exploratório, Bach (2009) percebeu que testadores faziam muitas coisas durante o dia além de testar. Para monitorar os testes, seria preciso encontrar uma forma de distinguir o teste de qualquer outra coisa. Foi quando surgiu a idéia das seções. Uma seção não é um caso de teste ou registro de um bug, mas uma unidade básica de teste. Um bloco ininterrupto, revisável e objetivo de um esforço de teste. As seções de teste são divididas em três tipos de tarefas (Bach 2009) (Crispin 2009): Design e Execução, Investigação e Report de bugs e Setup da sessão, que são chamadas de TBS (Task Breakdown Structure). Após, é necessário que os testadores estimem o tempo gasto para cada tipo de tarefa. Isso inclui, também, os tempos gastos em missões associadas às seções e o tempo de teste gasto em oportunidades, ou seja, o tempo despendido a qualquer teste que não esteja descrito na missão da sessão.

Característica – Teste Baseado em Seção

Missão: Objetivo do Teste. Ex.: analisar uma função, ou procurar um problema particular, ou verificar um conjunto de bugs consertados.

Tempo da Seção: Média de 45 à 120 minutos. Sendo que 45 minutos é classificado como seção curta e mais de 02 horas classificado como seção longa.

Passos: Não há.

Registros: Para saber quais tarefas foram realizadas durante o teste, é necessário que os testadores as reportem de forma genérica. O testador precisa registrar a atividade executada, reações do sistema, dados utilizados, condições, diagnósticos ou idéias.

Característica – Missão

Missão: Objetivo do Teste. Ex.: analisar uma função, ou procurar um problema particular, ou verificar um conjunto de bugs consertados.

Não deve ser muito específica nem muito genérica;

A missão determina o que deve ser testado (não como o teste deve ser realizado);

Ao final da seção de teste exploratório, novas idéias, oportunidades ou problemas encontrados pelo testador podem ser usados para a criação de novas missões;

Importante que sempre tenha uma reunião de alinhamento após a conclusão das missões para discutirem os vídeos gravados, anotações feitas, possíveis erros identificados.

Etapas da seção

Preparação (Setup): Preparação do ambiente de testes, configuração de massa de dados, leitura de manuais, requisitos, diagramas, etc.

Especificação (Design): Definição (modelo mental) dos casos de testes (hipóteses) baseados em heurísticas, idéias, checklists, etc.

Execução (Execution): Execução propriamente dita do teste exploratório para demonstrar se as hipóteses/expectativas foram atendidas (ou não).

Oportunidades (Opportunities): Tempo gasto em atividades, explorações, investigações que não estão no escopo ou foco da missão.

Relato de defeitos (Bug investigation/Report): Investigação e registro de defeitos.

- Ad-hoc
- Baseado em Sessão


I'm gonna hug you!

Links de apoio

- http://agilemanifesto.org/iso/ptbr/manifesto.html
- https://pt.slideshare.net/Qualister/mini-curso-de-testes-ageis
- http://www.agilemodeling.com/artifacts/userStory.htm
- https://ronjeffries.com/xprog/articles/expcardconversationconfirmation/
- http://qualidade-de-software.blogspot.com/2010/02/plano-de-teste-padrao-ieee-829-1998.html
- http://agile.pub/assuntos-diversos/as-5-maiores-soft-skills-para-um-agile-tester/
- https://www.dextra.com.br/blog/agile-testing-quadrants/
- https://cucumber.io/docs/guides/
- https://www.selenium.dev/
- https://pt.slideshare.net/alancarlos29/alm-testes-exploratrios-26949951

Contatos:
vanilton.pinheiro@fpf.br
vanilton18@gmail.com

Linkedin e Instagram:
/vaniltonpinheiro

Facebook:
/vanilton.pinheiro

Site:
vanilton.net