
FIAP
FIAP
SLIDER

COMPLIANCE & QUALITY ASSURANCE

Prof. M.Sc. Felipe Desiglo Ferrare
proffelipe.ferrare@fiap.com.br

Introdução aos Testes de Software e Processos
Aula 3

Cronograma

- Conceitos de Testes de Software
- Modelos de Maturidade

Conceitos Básicos
Terminologia

Testes Funcionais

Envolve testes que avaliam as funções que o sistema deve executar.

Alguns tipos de suítes de testes Funcionais:

- Teste de regressão
- Teste de confirmação
- Teste exploratório
- *Smoke Test*
- *Sanity Test*

Conceitos Básicos

Terminologia

- **Testes Exploratórios:** técnica em que testes informais (não pré-definidos) são modelados, executados, registrados e avaliados dinamicamente durante a execução da tarefa. Um bom teste exploratório é planejado, interativo e criativo. É frequentemente usado para complementar outras técnicas de teste e servir como base para o desenvolvimento de casos de teste adicionais.

Conceitos Básicos

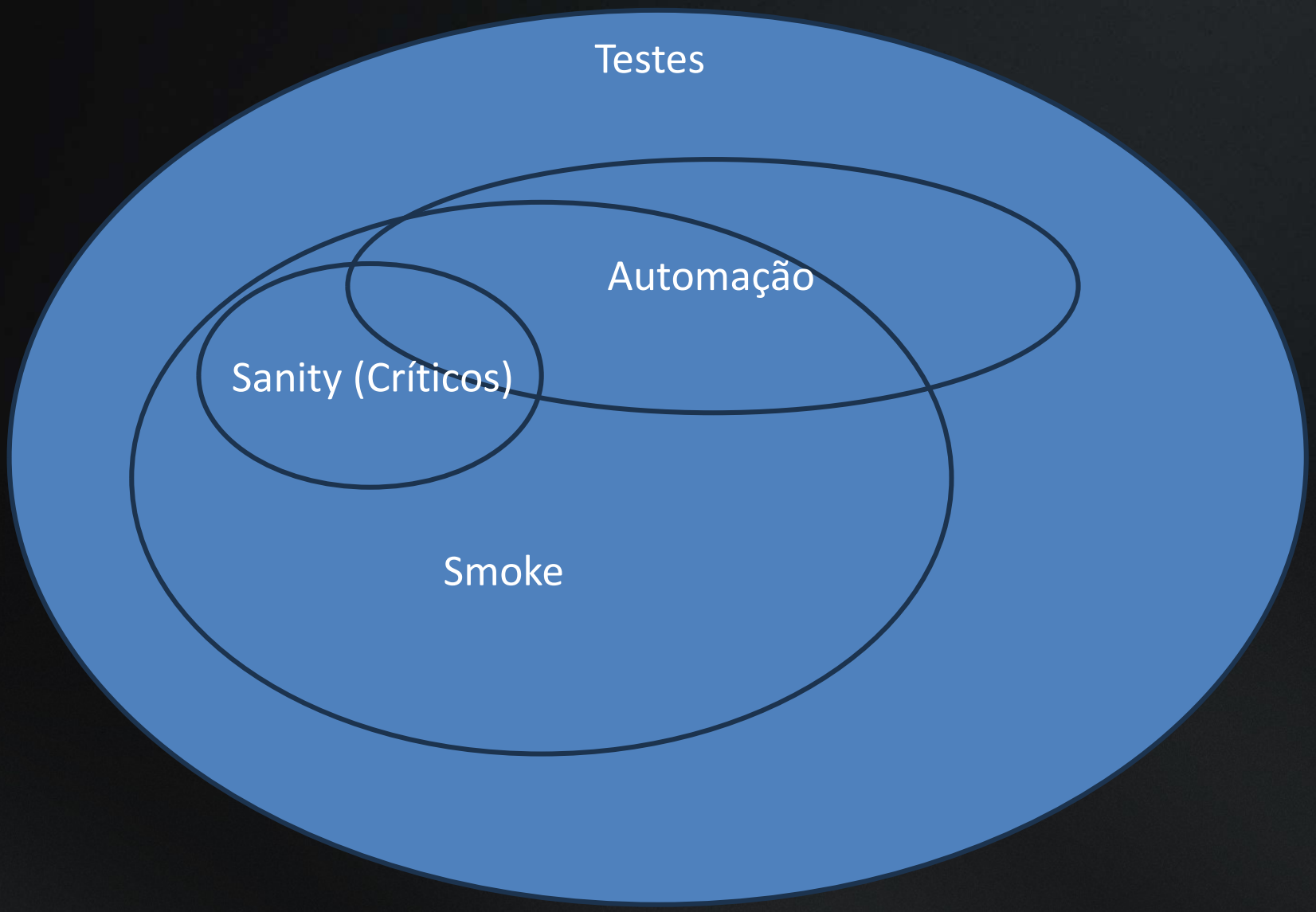
Terminologia

- **Teste de Confirmação:** Depois que um defeito é corrigido, todos os casos de teste que falharam devido ao defeito devem ser executados novamente na nova versão. No mínimo, as etapas para reproduzir as falhas causadas pelo defeito devem ser executadas novamente, com a finalidade de confirmar se o defeito original foi corrigido com sucesso.
- **Teste de Regressão:** É possível que uma alteração feita em uma parte do código (ou mesmo no ambiente), seja uma correção ou nova implementação, possa afetar acidentalmente o comportamento de outras partes do sistema. Tais efeitos colaterais não intencionais são chamados de regressões. Portanto, o teste de regressão envolve a execução de um conjunto testes básicos que abrangem as principais funcionalidades do sistema, com o intuito detectar eventuais efeitos colaterais indesejados. Os conjuntos de testes de regressão são executados muitas vezes (em toda nova release), portanto é um forte candidato à automação.

Conceitos Básicos

Terminologia

- **Smoke Tests:** pequeno grupo de testes que verifica se as funcionalidades básicas do sistema estão operacionais em uma determinada release. Eles devem ser sempre os primeiros testes executados no escopo, sendo que a aprovação neles é a condição para a continuidade das demais atividades de teste programadas.
- **Sanity Tests:** o teste de sanidade é feito quando o tempo disponível para validação é muito curto, sendo então um conjunto pequeno de testes simples, mas que deve cobrir todas as funcionalidades principais do sistema. Pode ser considerado um subconjunto do Teste de Regressão. Ao contrário do *Smoke Test* e do Teste de Regressão, que são os testes iniciais de um escopo maior, o *Sanity Test* normalmente é o único escopo possível para a release, dada a restrição de tempo.


Testes
Automação
Sanity (Críticos)
Smoke

Tipos de Cobertura de Testes

- **Cobertura de instrução:** técnica que testa as instruções executáveis do código. A cobertura é uma porcentagem do número de instruções executadas pelos testes em relação número total de instruções executáveis existentes, e o objetivo deve ser sempre a cobertura total (embora na prática isso nem sempre seja possível).
- **Cobertura de decisão:** técnica que testa as decisões existentes no código e percorre o código implementado com base nos resultados da decisão. Decisões são pontos no código onde o fluxo de controle escolhe um de dois ou mais resultados possíveis (IF, ou SWITCH/CASE). Os testes de decisão não consideram como uma decisão com múltiplas condições é tomada e podem falhar na detecção de defeitos causados por combinações dessas condições.
- **Cobertura de condição/decisão modificada:** técnica que considera como uma decisão é tomada quando inclui múltiplas condições. Verifica se cada uma das condições únicas afeta de forma independente e correta o resultado da decisão global, proporcionando, portanto, um nível de cobertura mais forte do que a instruções e decisões (quando há múltiplas condições).

Técnicas de Teste

- **Partição de equivalência:** usada para criar partições de equivalência (geralmente chamadas classes de equivalência) a partir de conjuntos de valores que precisam ser processados da mesma maneira. Ao selecionar um valor representativo de uma partição, a cobertura para todos os itens na mesma partição é assumida. Alguns exemplos de classes de equivalência são: entradas válidas e não válidas; letras, números e caracteres especiais; valores segmentados por faixa, etc.).
- **Análise de valor limite:** também chamada de BVA (*Boundary Value Analysis*), a técnica é usada para testar o manuseio adequado dos valores existentes nos limites das partições de equivalência ordenadas. O comportamento nos limites das partições de equivalência tem maior probabilidade de estar incorreto do que o comportamento dentro das partições. Normalmente usa-se o teste com 3 valores: o próprio valor limite, um valor antes e um valor depois.

Técnicas de Teste

FIAP

- **Tabela de decisão:** é uma representação tabular de um conjunto de condições e ações relacionadas, regras expressas que indicam qual ação deve ocorrer para qual conjunto de valores de condição. Condições e ações formam as linhas da tabela. Cada coluna corresponde a uma regra de decisão (combinação de condições que resultam na execução de determinadas ações) e deve ter um teste associado a ela.

Condições	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5	Teste 6	Teste 7
É Eletrônico	S	S	S				
É Eletro-doméstico				S	S	S	
Outra categoria							S
Preço >= 500,00	S	S	N				
Preço >= 1000,00				S	S	N	
Cliente VIP	S	N	-	S	N	-	
Ações							
Aplicar desconto de 10%		X			X		
Aplicar desconto de 15%	X			X			
Não aplicar desconto			X			X	X

Técnicas de Teste

FIAP

- **Testes Positivos:** Testes em que existe o sucesso da ação (“Caminho feliz”)
- **Testes Negativos:** Testes em que acontecem exceções, situações não previstas e erros.

Modelos de Maturidade

Os modelos de maturidade desempenham um papel fundamental no campo da engenharia de software, fornecendo uma abordagem estruturada para avaliar e melhorar os processos de desenvolvimento de software de uma organização.

Eles servem como uma ferramenta de benchmarking, permitindo que as organizações avaliem a maturidade de seus processos atuais e identifiquem áreas de melhoria.

Os modelos de maturidade ajudam a alcançar consistência, previsibilidade e qualidade no desenvolvimento de software, levando, em última análise, a uma maior satisfação do cliente e à redução de riscos.

CMMI

- Inicialmente desenvolvido na década de 80 como CMM

- Níveis de capacidade dos processos

Criado pela Software Engineering Institute (SEI) da Carnegie Mellon University, atualmente desenvolvido pela ISACA.

- CMM inspirou a ISO/IEC 15504 (SPICE)

CMMI
Níveis

FIAP

1 - Inicial

Os processos são ad-hoc e muitas vezes caóticos.
Há uma falta de consistência nos processos e, frequentemente, dependência da habilidade individual.

2 - Gerenciado

Os processos são planejados e executados de acordo com políticas bem definidas. Há um controle básico sobre os processos e uma conscientização de gestão.

3 - Definido

Os processos são padronizados e documentados. Há um foco na padronização dos processos dentro da organização.

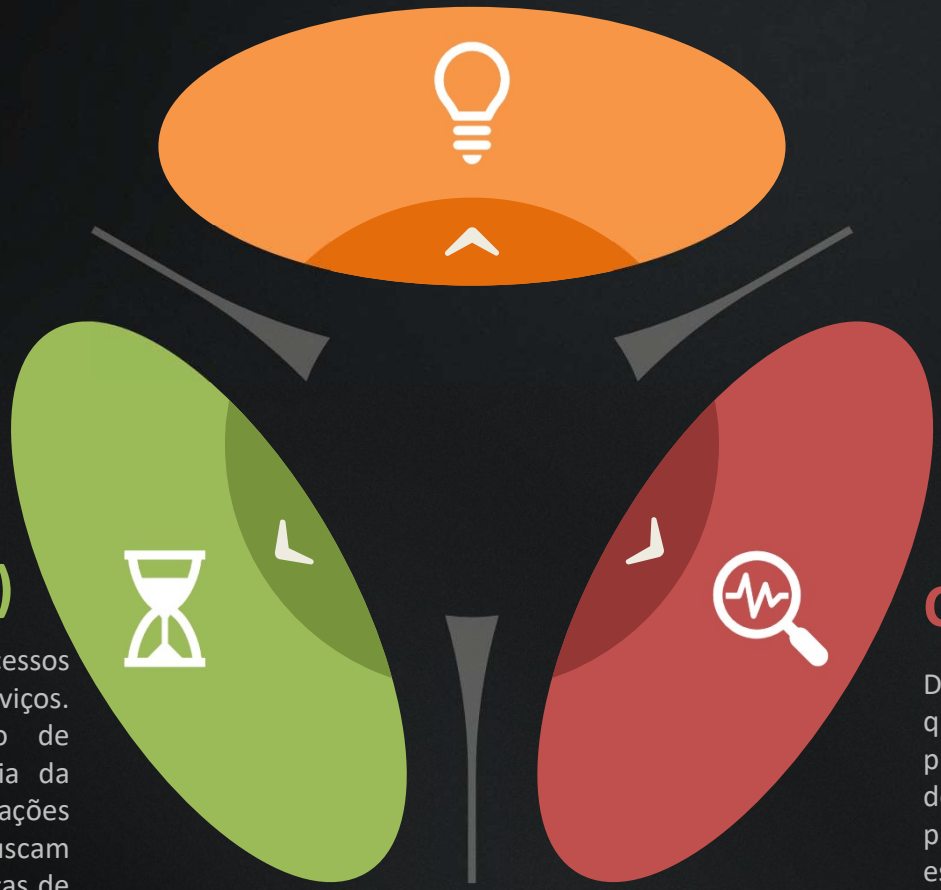
4 – Quantitativamente Gerenciado

A organização quantifica seu desempenho. Os processos são controlados usando técnicas estatísticas e outras formas de gerenciamento quantitativo.

5 – Em otimização

A organização busca continuamente melhorar seus processos por meio de inovação e otimização. Há um foco na melhoria contínua e na introdução de práticas inovadoras.

CMMi v1.3 - Constelações



CMMi - Dev

Tem como objetivo principal melhorar os processos de desenvolvimento de software nas organizações. Ele fornece um conjunto de práticas e diretrizes para ajudar as organizações a alcançarem maturidade em seus processos de desenvolvimento.

CMMI-ACQ (Aquisição)

Projetada para aprimorar os processos relacionados à aquisição de produtos e serviços. Ele abrange áreas como gerenciamento de fornecedores, gestão de contratos, garantia da qualidade na aquisição, fornecendo orientações específicas para organizações que buscam melhorar a eficácia e eficiência de suas práticas de aquisição.

CMMI-SVC (Serviços)

Dedicada à melhoria de processos em organizações que prestam serviços. Ele fornece diretrizes e práticas específicas para áreas como gerenciamento de serviços, entrega de serviços, resolução e prevenção de problemas, proporcionando uma estrutura para melhorar a qualidade e eficiência na entrega de serviços.

Exemplo
CMMI-DEV 1.3

REQUIREMENTS DEVELOPMENT

An Engineering Process Area at Maturity Level 3

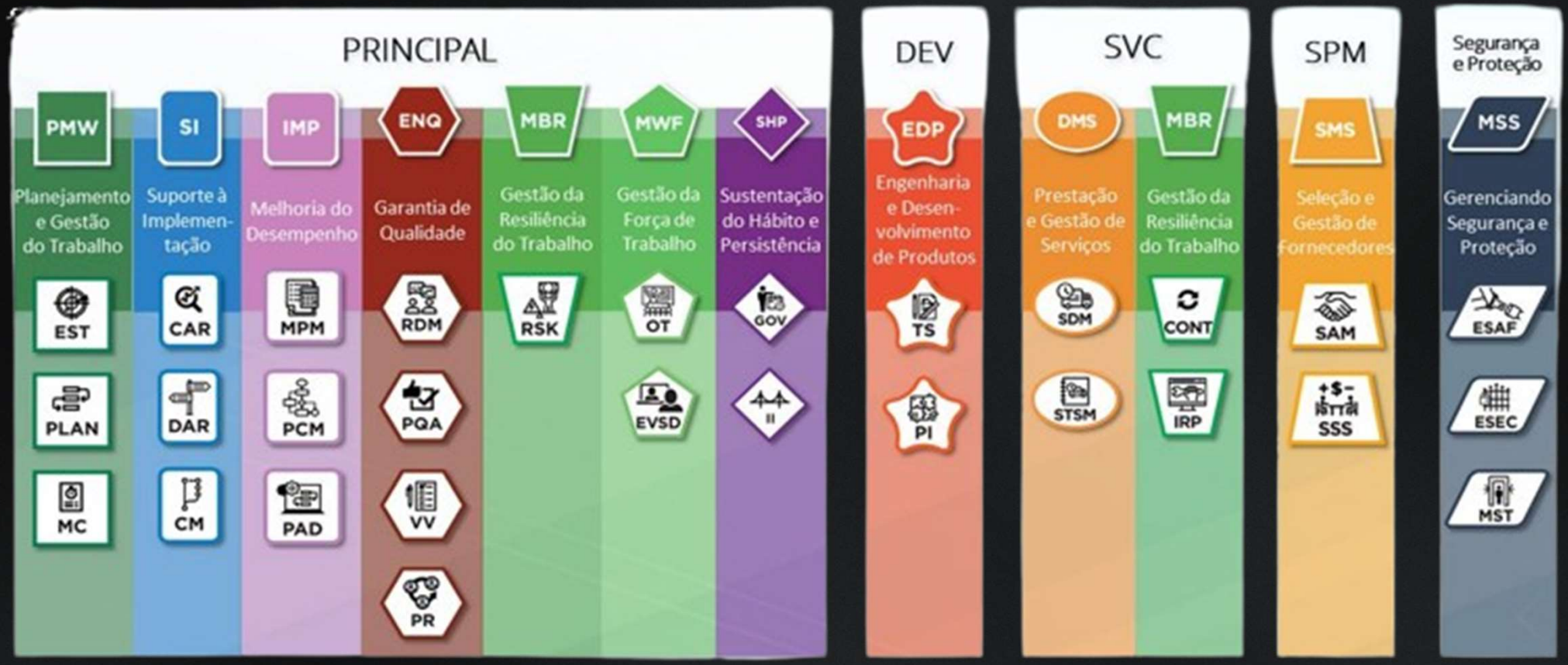
Specific Goal and Practice Summary

SG 1 Develop Customer Requirements
- SP 1.1 Elicit Needs
- SP 1.2 Transform Stakeholder Needs into Customer Requirements

SG 2 Develop Product Requirements
- SP 2.1 Establish Product and Product Component Requirements
- SP 2.2 Allocate Product Component Requirements
- SP 2.3 Identify Interface Requirements

SG 3 Analyze and Validate Requirements
- SP 3.1 Establish Operational Concepts and Scenarios
- SP 3.2 Establish a Definition of Required Functionality and Quality Attributes
- SP 3.3 Analyze Requirements
- SP 3.4 Analyze Requirements to Achieve Balance
- SP 3.5 Validate Requirements

CMMI 2.0 / 3.0 – Áreas:



TMMI

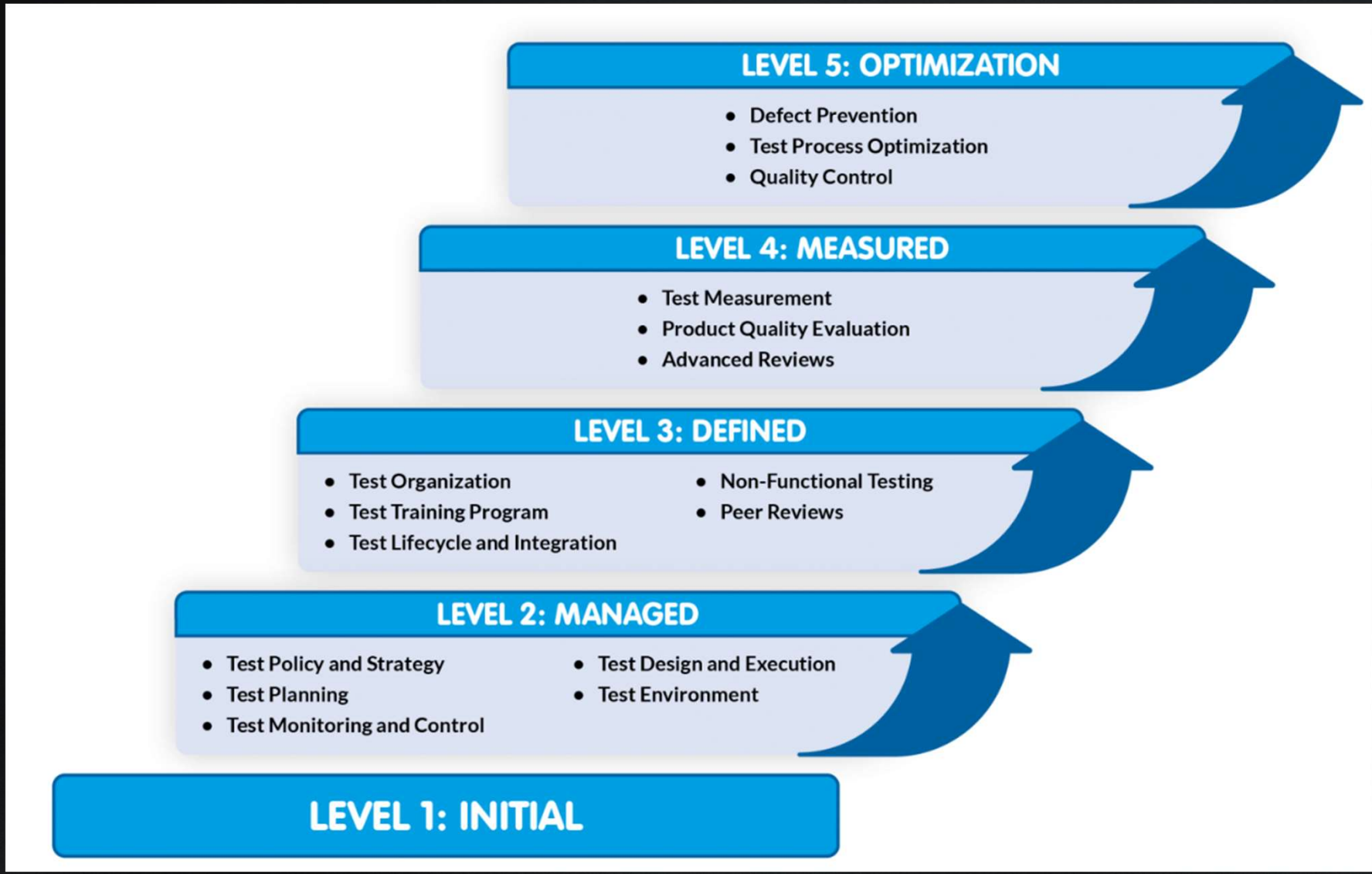
- Testing Maturity Model integration (TMMI)

Baseado no CMM

Busca avaliar os processos de testes e qualidades da Organização para a melhoria de processos

TMMI

FIAP

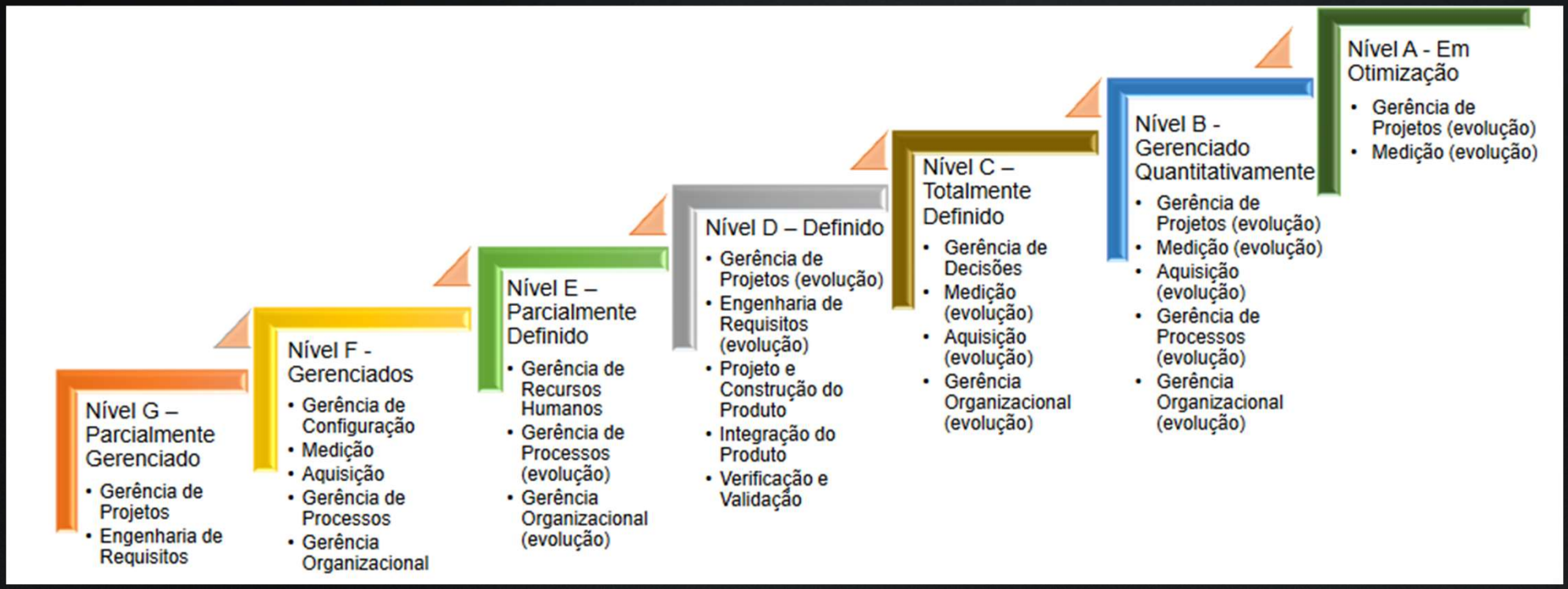

LEVEL 5: OPTIMIZATION
Defect Prevention
Test Process Optimization
Quality Control
LEVEL 4: MEASURED
Test Measurement
Product Quality Evaluation
Advanced Reviews
LEVEL 3: DEFINED
Test Organization
Test Training Program
Test Lifecycle and Integration
Non-Functional Testing
Peer Reviews
LEVEL 2: MANAGED
Test Policy and Strategy
Test Planning
Test Monitoring and Control
Test Design and Execution
Test Environment
LEVEL 1: INITIAL

MPS.BR

- Modelo Brasileiro de Qualidade de Processos

-Concorrente ao CMMI

MPS.BR
Exemplo DEV



Obrigado!


FIAP