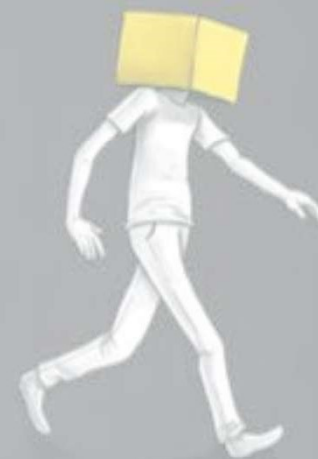




**PENSANDO QA DOS CONCEITOS A PRÁTICA, O QA FORA DA CAIXA. QADEVOPS MITO OU REALIDADE?**

NÃO ADIANTA  
SAIR DA CAIXINHA,  
SE ELA NÃO  
SAIR DE VOCÊ.



**Facilitadora: Alessandra M. Martins**

## ALESSANDRA MONTEIRO MARTINS

### MINI BIO:

Formada Em Licenciatura em Informática pela Universidade do Estado do Amazonas - UEA, Especialista em Governança de TI pela Universidade Católica de Brasília – UCB, ITIL v3, COBIT 5, ISO 27002 Foundation Certified, CTFL, Scrum Master e KMP I certificada.

Atuando no mercado de TI desde 2004, comecei a carreira por infraestrutura e segurança, em 2011 entrei para o gerenciamento de serviços de TI e Projetos, em 2013 comecei a atuar com Inteligência de Negócios, Projetos Ágeis de SDLC E BI, atuando principalmente em fábricas de software, consultoria e serviços de Qualidade de Software e Boas Práticas de Desenvolvimento Seguro. Em 2014, comecei a estudar sobre forense computacional, me tornando entusiasta e autodidata em temas relacionados a engenharia de software, segurança, auditoria e perícia digital.

**Belém – Manaus – São Paulo**

# Roteiro:

QA DevOps Mito ou Realidade?

At the Beginning  
Conceitos

01



Criatividade é importante

02

Processos

03

Modelos



04

Técnicas



Conhecer

Ferramentas,  
Frameworks, Flows e  
Técnicas é preciso

05

Ferramentas

09

Papel QA



Aprender

Com os testes, erros e  
acertos, tudo pode servir  
de base de conhecimento

08

Mudanças

07

Defeitos

Experimentar

Sempre, validar hipóteses,  
necessário para avaliar as  
ferramentas e técnicas mais  
aderentes



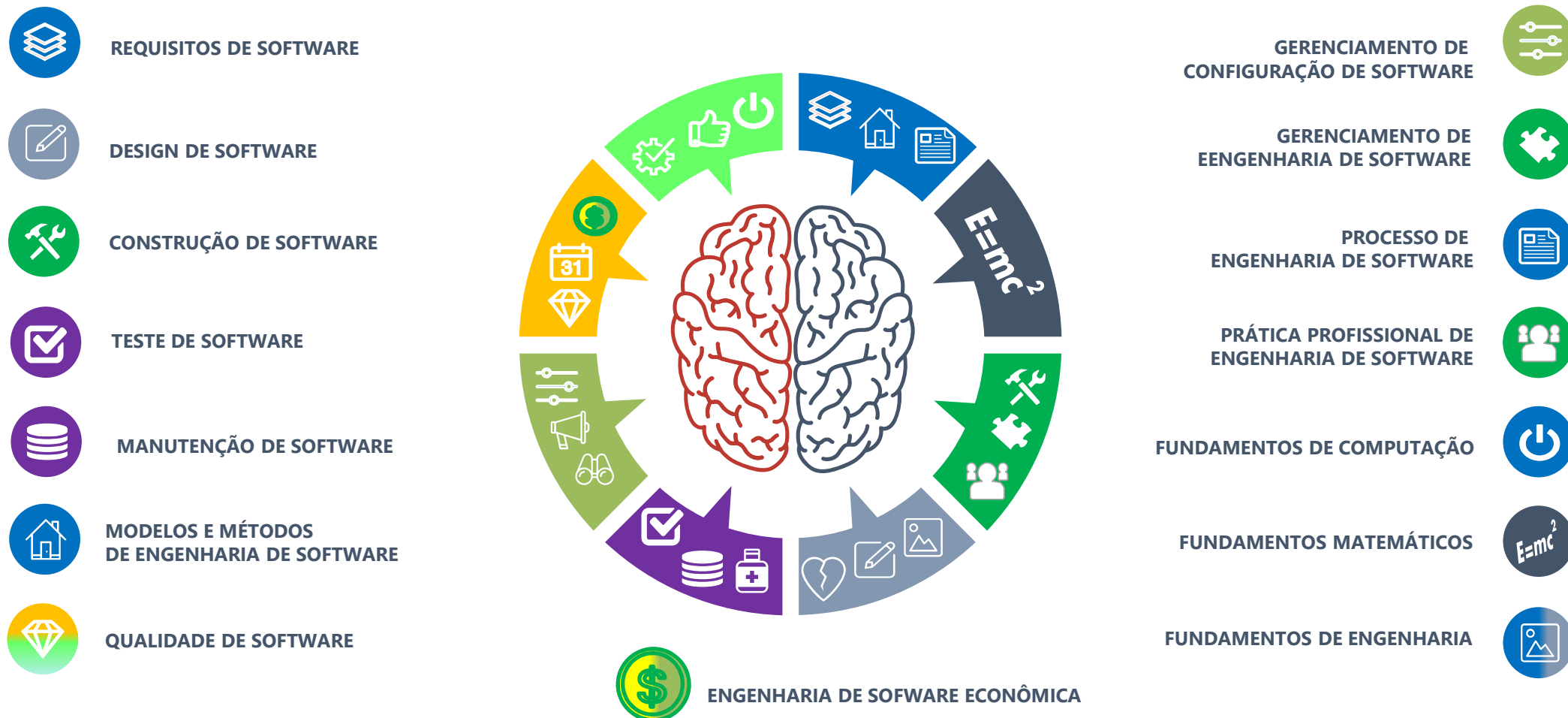
Pipeline

06



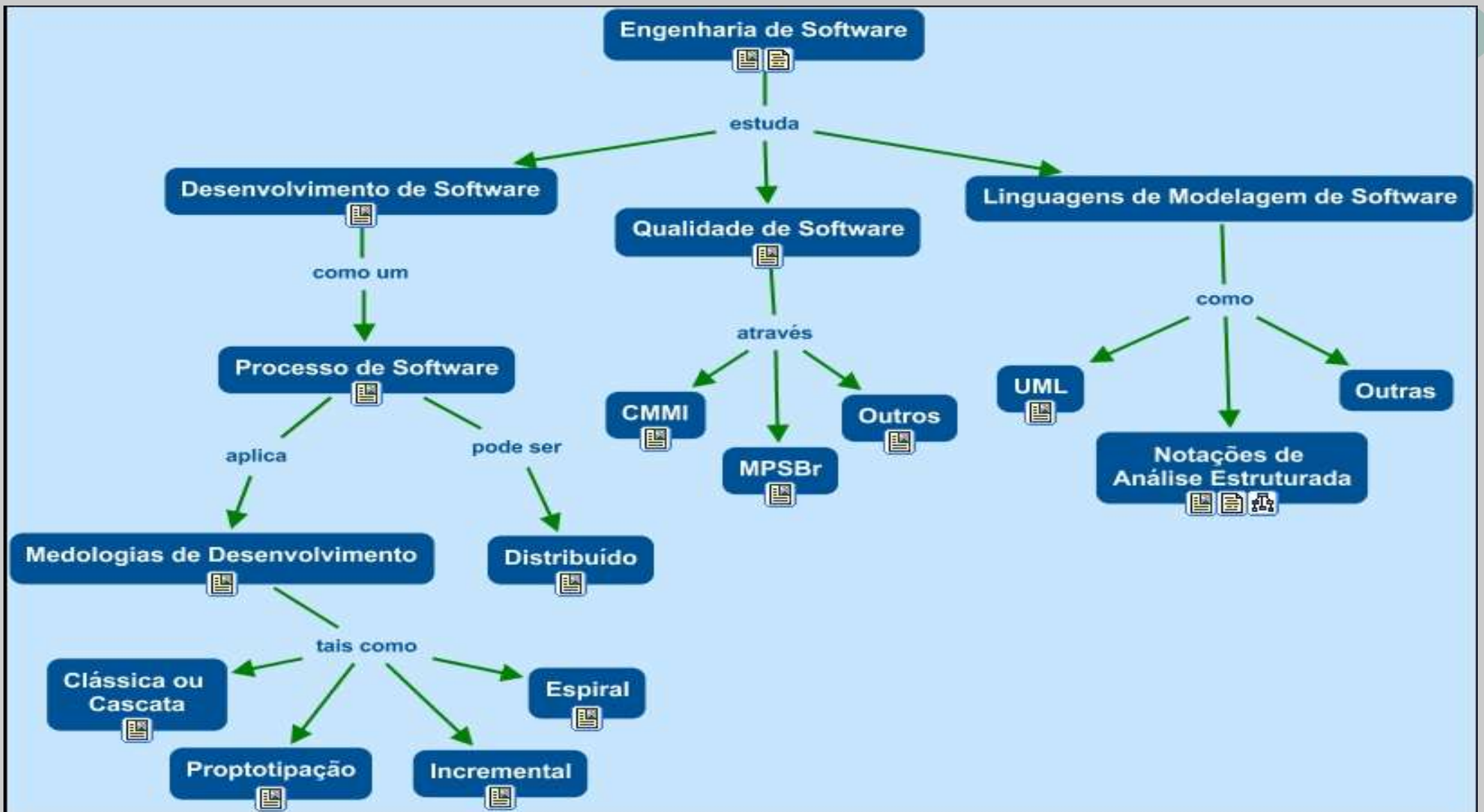
# Knowledge Area (KA) SWEBOK

Áreas de Conhecimento da Engenharia de Software



# 1. Engenharia de Software

- **O principal objetivo da engenharia do software é auxiliar na construção de produtos de qualidade.**
- Com o intuito de melhorar a qualidade dos softwares em geral e aumentar a produtividade no desenvolvimento de tais produtos, surgiu a engenharia de software. Esta é um conjunto de três elementos: métodos, ferramentas e procedimentos que possibilitam ao gerente controlar o processo de desenvolvimento e, ao profissional, oferece base para construção de um produto com alta qualidade, de forma produtiva.



# 1.Qualidade de Software

- Recomendado tanto pelo Syllabus do BSTQB (Brazilian Software Testing Qualification Board), quanto por metodologias ágeis, onde para todas as atividades do desenvolvimento há uma atividade de teste correspondente, modelos de ciclo de vida de desenvolvimento diferentes necessitam de abordagens diferentes para testar.
- **Qualidade é atender perfeitamente, de forma confiável (sem defeitos), acessível (de baixo custo), segura e no tempo certo as necessidades ou requisitos do cliente.**
- A qualidade também pode ser entendida como sendo a conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido.

- A qualidade é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir a conformidade de processos e produtos, prevenindo e eliminando defeitos.



# 1. Controle de Qualidade

- **O controle de qualidade** pode ser definido como um método de comparação entre o produto e os requisitos apresentados no projeto. É feita uma verificação determinando se o produto está dentro dos níveis aceitáveis. Diferentes níveis de teste podem ser usados para esse controle de qualidade.
- **FOCO:** Descobrir defeitos em produtos de trabalho gerados ao longo do projeto.
- "Controle de qualidade é definido como os processos e métodos usados para monitorar o trabalho e os requerimentos envolvidos. É focado nas revisões e remoção de defeitos antes da entrega do produto. Controle de qualidade deve ser responsabilidade da unidade de produção do produto dentro da organização..." (p.64, COSTA, NETO, COSTA NETO, & JUNIOR, 2013)



# 1. Garantia de Qualidade – Software Quality Assurance

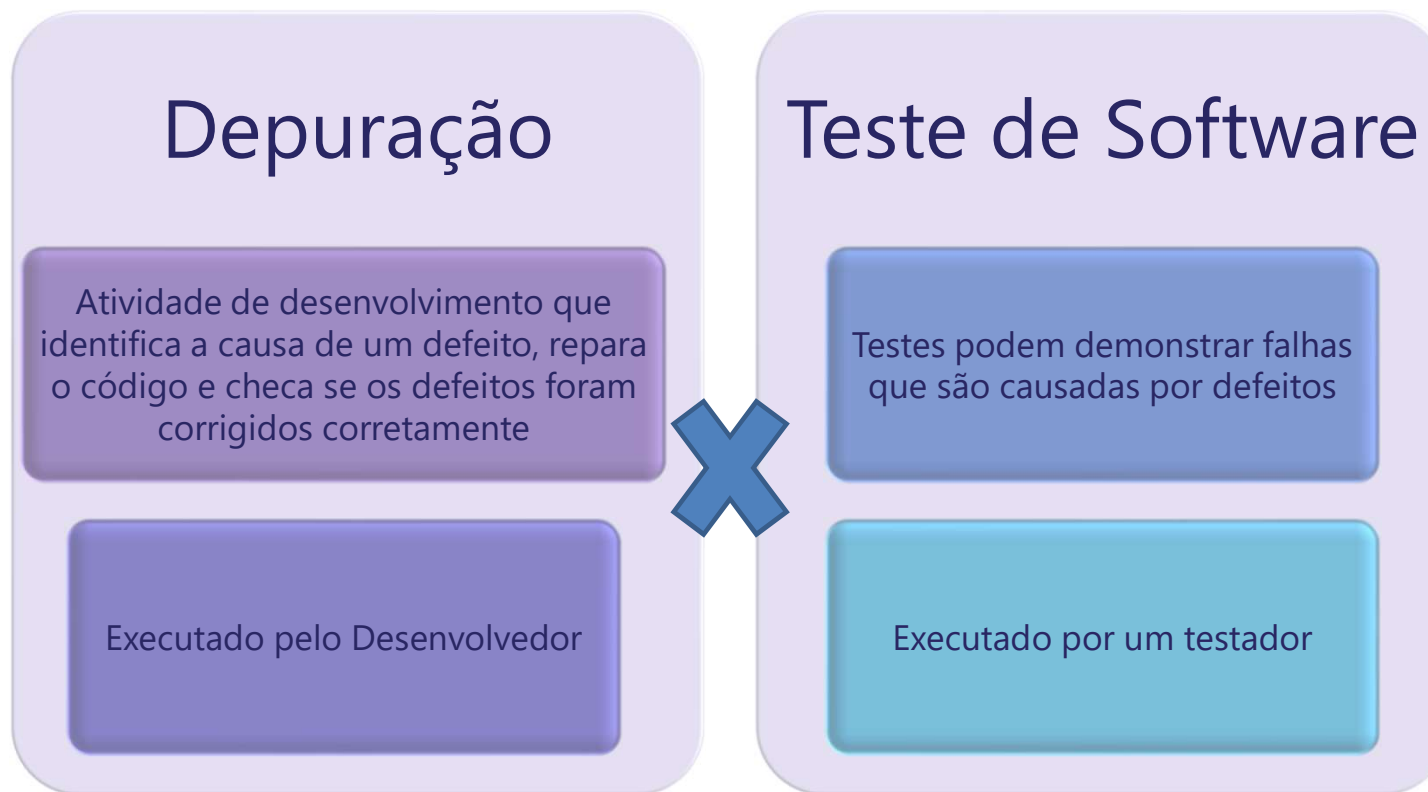
- O **Software Quality Assurance** ou **Garantia de Qualidade de Software**, refere-se a um conjunto de atividades que visa assegurar que todos os esforços serão feitos para garantir que os produtos de software tenham a qualidade desejada.
- A execução dessas atividades deve minimizar o número de defeitos; criar mecanismos para controlar o desenvolvimento e manutenção de forma a preservar os prazos e custos; garantir que o produto possa ser utilizado no mercado; melhorar a qualidade de versões futuras do produto ou de novos produtos.
- O rastreamento da qualidade de software é considerado um objetivo importante de SQA é para avaliar o impacto das mudanças metodológicas e procedimentais sobre a qualidade do software. Para isso seja realizado, **métricas de qualidade de software devem ser coletadas**. A anotação e manutenção de registros para a garantia de qualidade de software oferecem procedimentos para coleta e disseminação de informações de SQA. Os resultados de revisões, auditorias, controle de mudanças, testes e outras atividades SQA devem tornar-se parte do registro histórico de um projeto e devem ser levados ao conhecimento do pessoal de desenvolvimento (p.56, COSTA, NETO, COSTA NETO, & JUNIOR, 2013)

# 1. Garantia de Qualidade

- **A garantia da qualidade** avalia a aderência das atividades executadas e dos produtos de trabalho, de acordo com padrões, processos, procedimentos e requisitos estabelecidos e aplicáveis. Ela assegura que a qualidade que foi planejada não seja comprometida. Busca identificar desvios o quanto antes, eliminando possíveis retrabalhos e melhorando custos e prazos.
- **FOCO:** Garantir que o projeto emprega todos os processos e padrões necessários para atender aos requisitos.
- Utiliza métodos, procedimentos e padrões para comparar previsto com realizado.
- É orientada a processo, visando a prevenção de defeitos.
- Cuida da monitoração e melhoria dos processos e padrões empregados.
- Assegura que se faz de maneira correta.

# Não Confundir

## O Que é Testar/ teste de software?



# 1. Ciclo de Vida de Software

Como todo produto industrial, o software tem um ciclo de Vida:

01

## PROJETOS - PROJECTS

GERENCIAMENTO DE PROJETOS| STAKEHOLDERS (PARTES INTERESSADAS)

02

## REQUISITOS - REQUIRMENTS

CARACTERÍSTICAS| ESPECIFICAÇÃO| ENGENHARIA DE REQUISITOS| GESTÃO DE REQUISITOS

03

## PRAZOS & CUSTOS – COSTS & TERMS

REALISMO DE PRAZOS E CUSTOS| PLANEJAMENTO DE PROJETOS| CONTROLE DE PROJETOS| GESTÃO DE CONTRATOS | RISCOS

04

## QUALIDADE - QUALITY

CONFORMIDADE COM REQUISITOS| GARANTIA DA QUALIDADE

05

## GESTÃO DE CONFIGURAÇÕES – CONFIGURATION MANAGEMENT

ARTEFATOS| SCM – SOURCE CONTROL MANAGEMENT | DOCUMENTOS

06

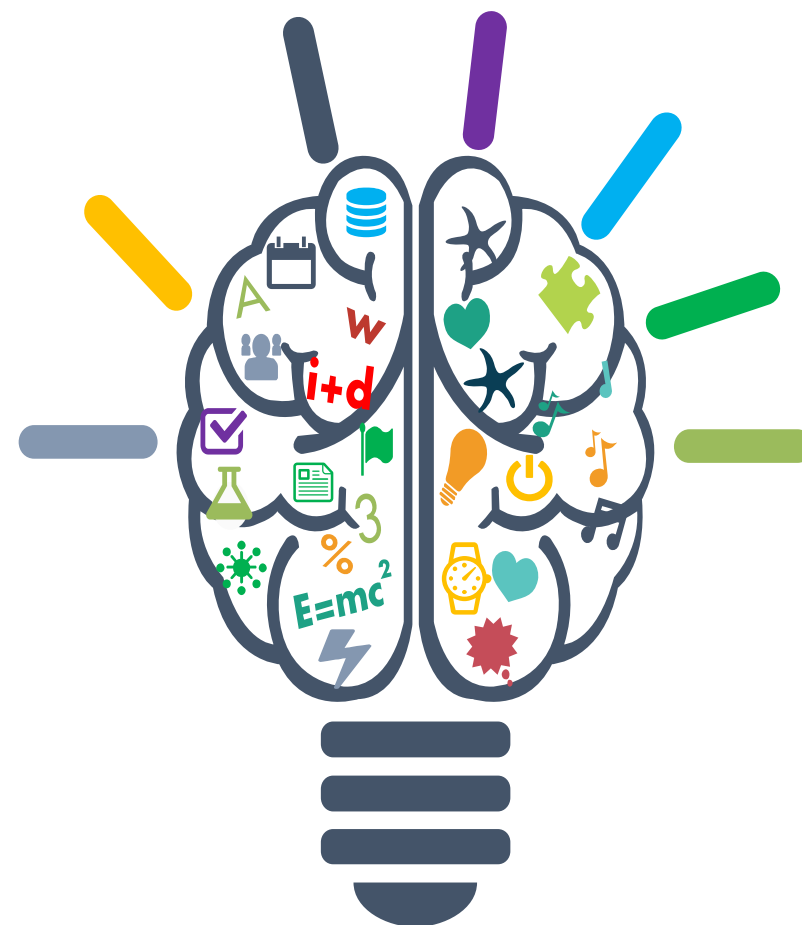
## DESENHO - DESIGN

EXPLÍCITO| DOCUMENTADO

07

## MATURIDADE - MATURITY

MODELOS DE MATURIDADE| CMMI| SW-CMM| MPS-BR



# 1. Conceitos

## O que é DevOps ?

É um movimento cultural que muda o modo como os indivíduos pensam sobre o seu trabalho, valoriza a diversidade do trabalho realizado, suporta processos intencionais que aceleram a taxa pela qual as empresas realizam valor e mede o efeito da mudança social e técnica.

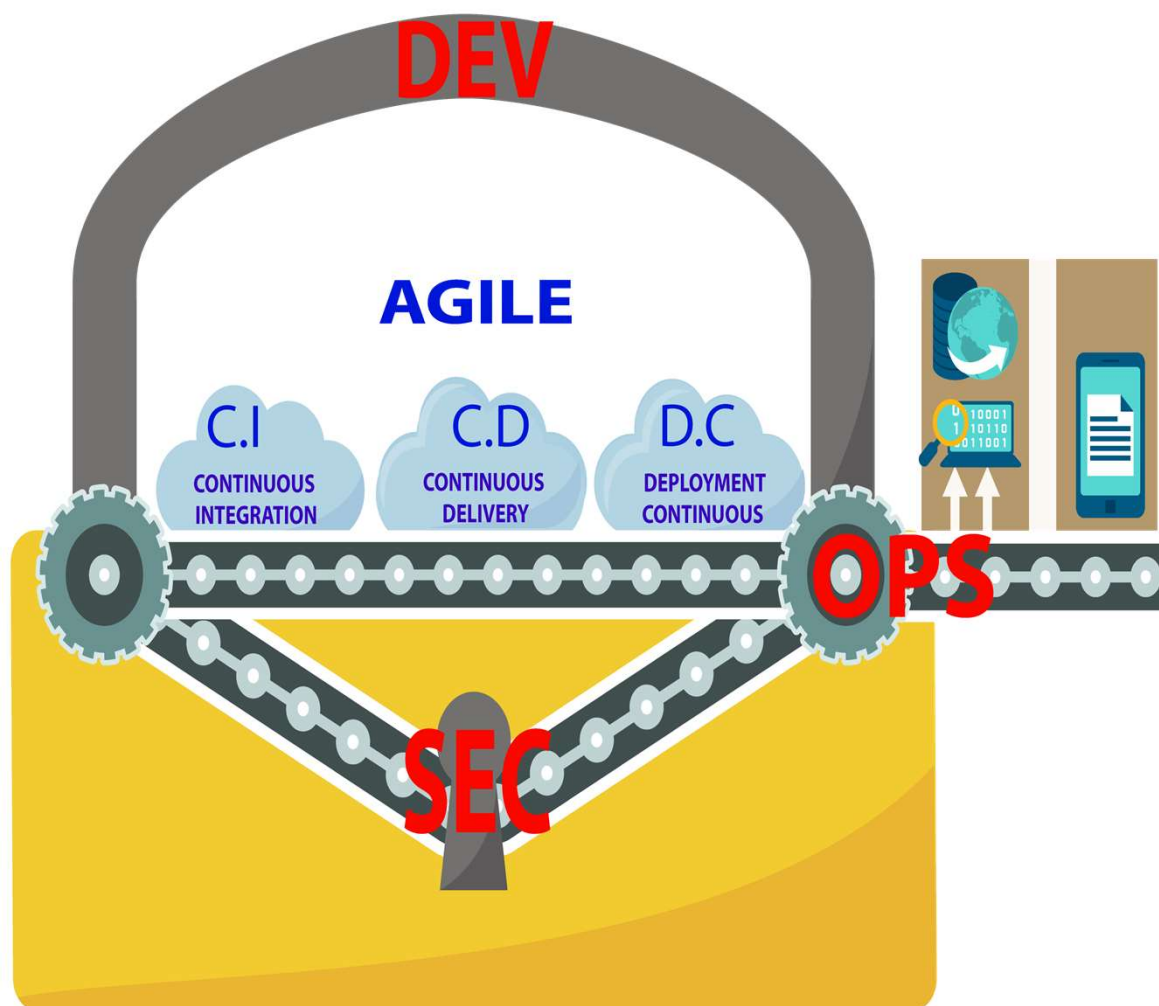
É uma maneira de pensar e um modo de trabalhar que permite que indivíduos e organizações desenvolvam e mantenham práticas de trabalho sustentáveis.

É um quadro cultural para compartilhar histórias e desenvolver empatia, permitindo que pessoas e equipes para praticar seus ofícios de forma eficaz e duradoura.



# 1. Conceitos

O que é DevSecOps ?

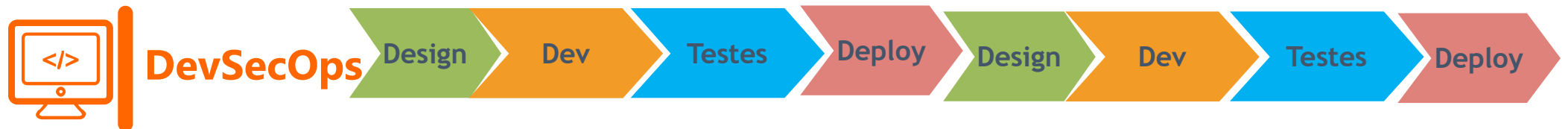


DevSecOps é um termo criado para descrever um conjunto de práticas para integração entre os times de Desenvolvimento de Software, Segurança e Operações e a adoção de processos automatizados para produção rápida e segura de aplicações e serviços

## 2 . Modelos de Processos

15

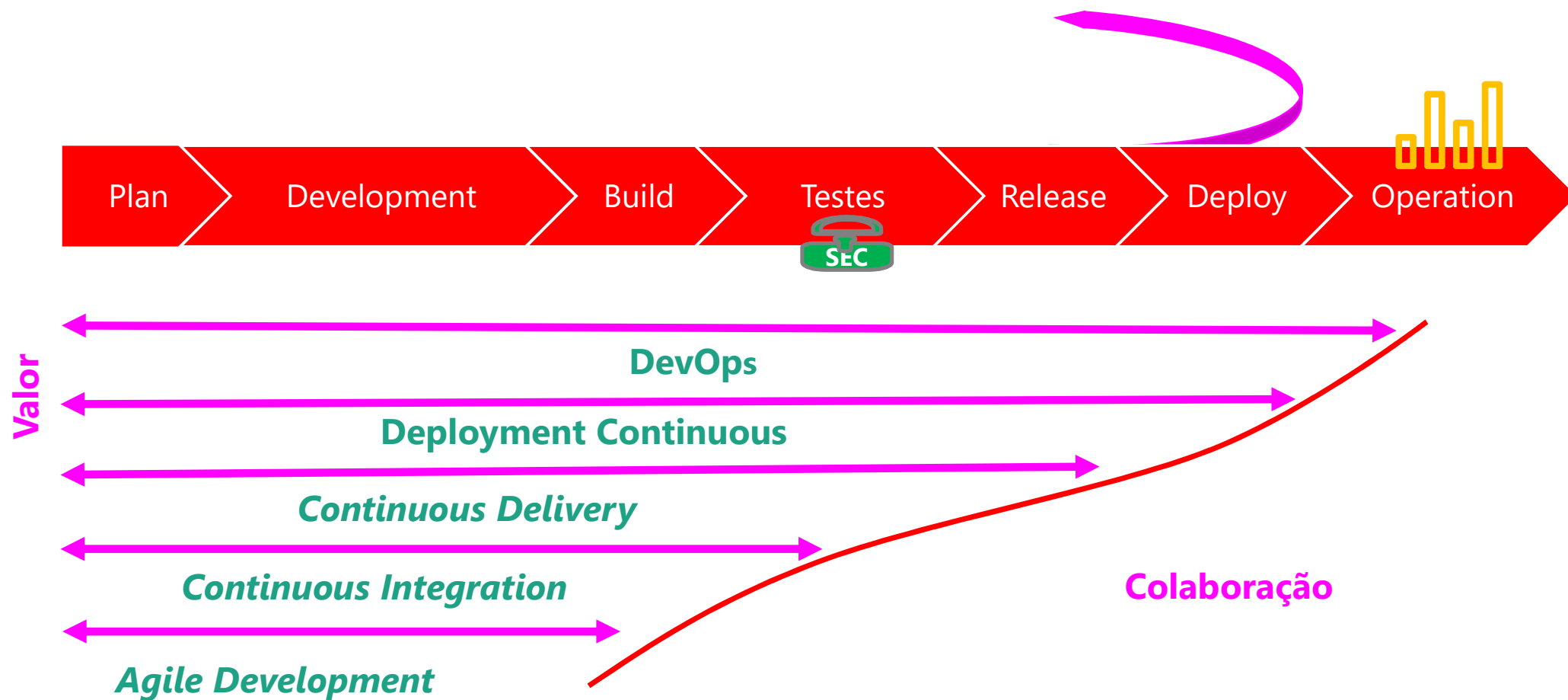
DevSecOps vs Outros Modelos





## 2. Conceitos + Modelo

DevSecOps + Agile



- Com base na documentação ou artefatos é que os profissionais de teste membros do *development team* irão definir os cenários, fluxos e escrever os casos de testes.
- Dessa forma, logo que os artefatos forem apresentados, a equipe já deverá trabalhar para validá-los, participando do refinamento e eliminando possíveis brechas, pensando além do descrito nos documentos. Para isso, os testadores devem ler, entender e analisar as *user stories*.
- Nessa função deve-se ter sempre em mente a visão do usuário e quais são os possíveis casos

### Avaliação da Documentação

### Criação da documentação de testes

- Com os artefatos revisados, melhorados e atualizados, deve-se dar início à escrita de casos de testes. A documentação de testes deve ser simples, direta e clara na medida do possível e deve descrever o passo-a-passo para validação dos critérios de aceite. Fazem parte dessa atividade a criação, manutenção e atualização dos documentos, além da criação de *scripts*.
- A documentação normalmente é criada em alguma ferramenta utilizada pela empresa e fica armazenada como histórico, servindo posteriormente como base de testes e criação de templates para funcionalidades mais complexas. Todos na equipe devem ser capacitados para criar a documentação de testes. É uma boa prática sempre realizar uma "validação" dos casos de testes escritos para verificar se a cobertura está completa e/ou se é necessário atualizar, acrescentar ou remover algum teste ou cenário que seja desnecessário. Essa validação pode ser feita por outro membro de testes do *development team*.

- Após os casos de testes e demais artefatos estarem concluídos, os profissionais devem fazer a preparação do ambiente que será utilizado nos testes.
- Essa preparação poderá incluir a montagem do ambiente, que engloba configuração no banco de dados, instalação e configuração de máquinas virtuais e servidores.
- Além disso, tem-se também a criação de massa de dados e parametrização no sistema, com o objetivo de deixar o ambiente completo e preparado para receber as atualizações e alterações para o teste.
- Além dos testadores, pode haver times específicos que trabalham para a disponibilização do ambiente.

### Preparação do ambiente de testes

### Execução dos testes

- Após a criação dos casos de testes e demais artefatos, e a verificação de que o ambiente está disponível e com as alterações das *user stories* aplicadas, os testadores podem iniciar a execução. A instalação ou atualização do sistema pode ser de responsabilidade da equipe de testes caso não exista uma equipe específica alocada para esse fim. Os testes são executados seguindo os artefatos criados e demais documentações.
- Após a execução, o testador deverá observar se algum outro cenário ou caso não foi coberto pelos artefatos, por isso, testes exploratórios são muito importantes. Sempre deve haver uma boa comunicação entre o time de testes e desenvolvimento dentro do *development team*, possibilitando a troca de informações, análise do que foi alterado, dúvidas e outras questões que podem ser tratadas.

- Quando algum comportamento estranho é identificado, é reportado ao desenvolvedor para que possa ser corrigido e depois retestado.
- Os resultados obtidos também devem ser categorizados conforme um template, se houver, seguindo um padrão, que pode ser: "Teste Passou" - execução sem falha; "Teste Falhou" - execução com falha (onde deve ser informado o passo-a-passo, resultados obtidos e esperados e qual a falha); "Bloqueado" - impossibilidade de execução do caso de teste devido a um pré-requisito não atendido ou motivo que não possibilite avaliar o item desejado e "Não executado".
- As *user stories* somente são aprovadas após todos os defeitos priorizados serem corrigidos e retestados. Após isso, cabe ao *product owner* e/ou analista funcional realizar o aceite.

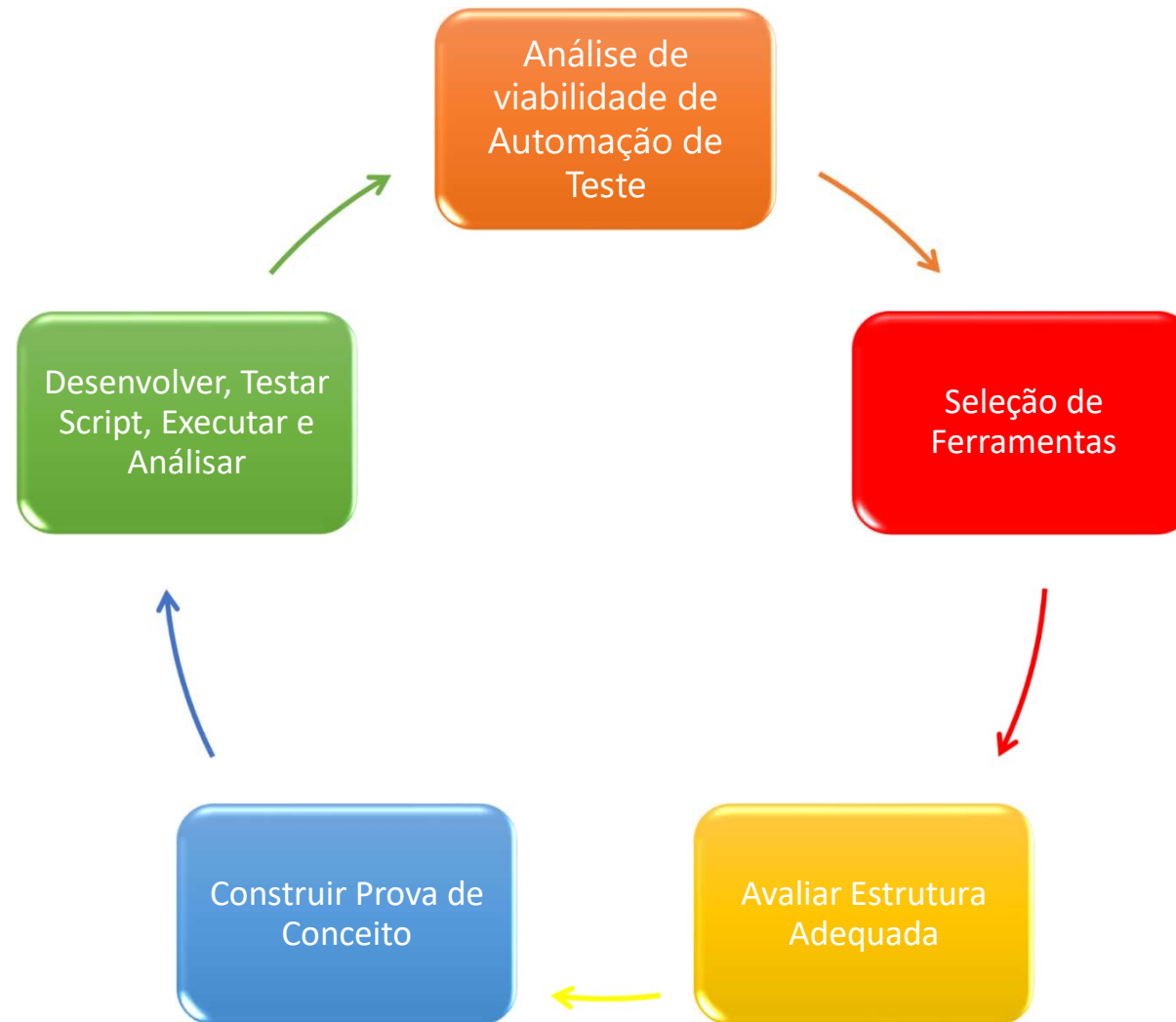
### Aceite e reprovação dos testes

## 2. Processo de Teste de Software

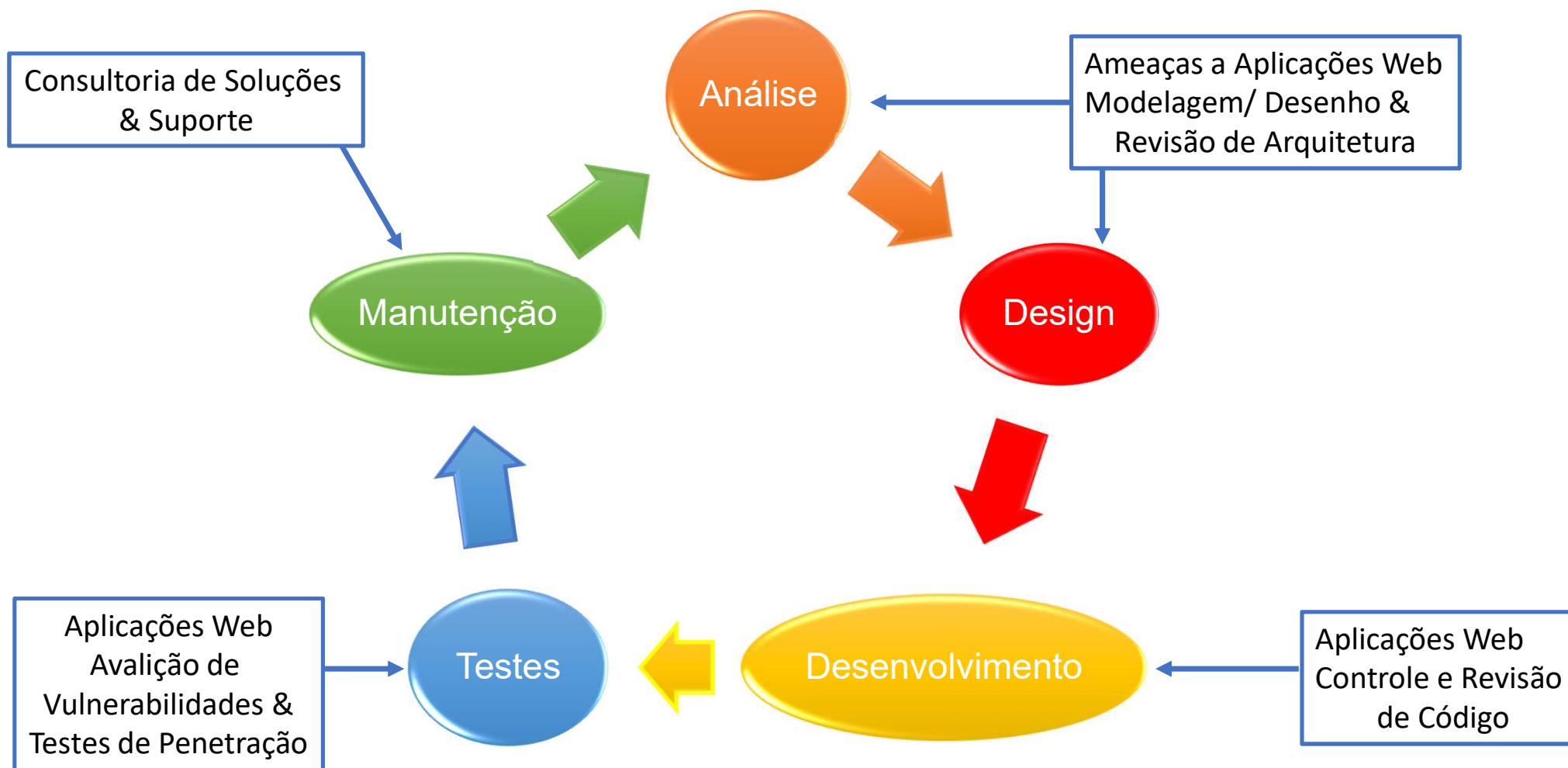
### Modelo V

## 2. Processo de Automação de Testes

18



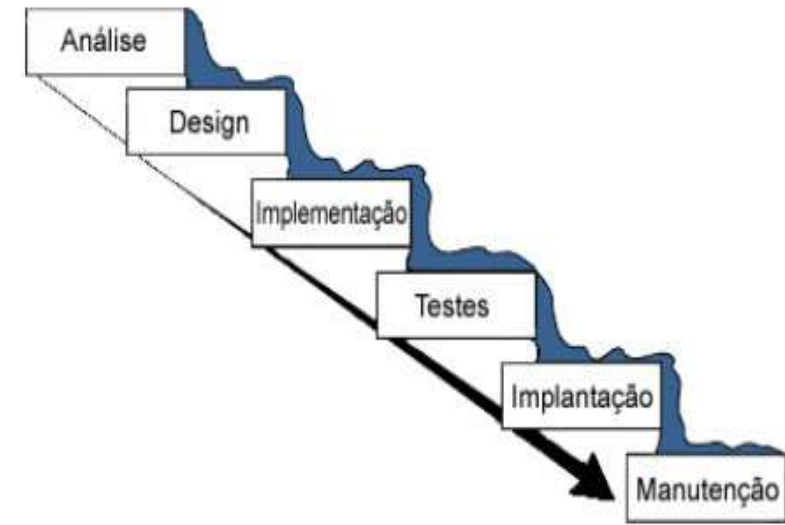
## 2. Processo Testes de Segurança



# 3. Testes durante o Ciclo de Vida do Software

## Modelo Cascata – Waterfall

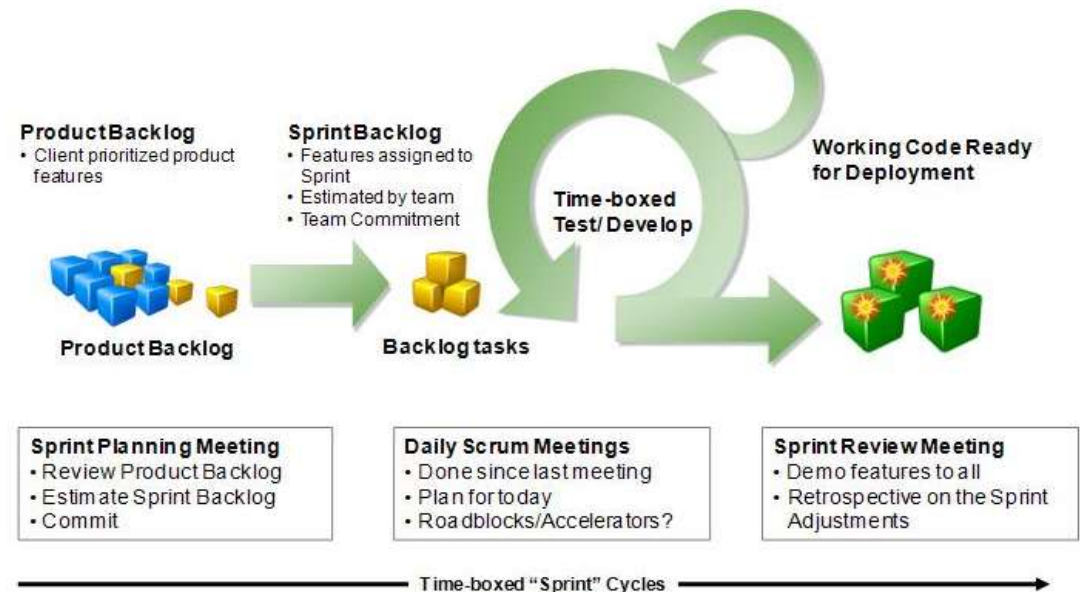
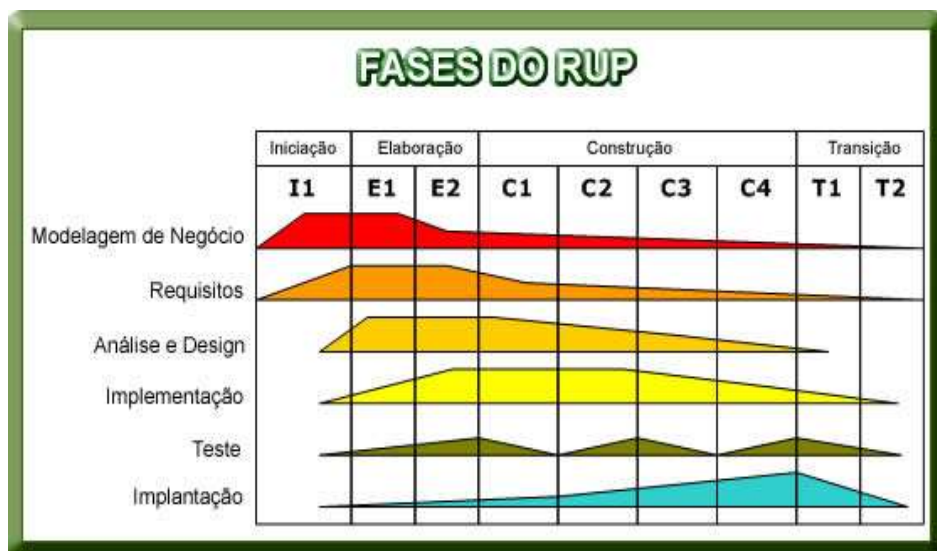
- Fases sequenciais
- Fases dependentes da completa finalização e aceitação do cliente da anterior para iniciar as próximas
- Cliente é participante ativo do projeto e sabe bem o que quer
- Teste pode iniciar com a revisão da documentação
- Acompanhamento do desenvolvimento
- Validação dos testes funcionais e de aceitação



# 3. Testes durante o Ciclo de Vida do Software

## Modelos Iterativos e Ágeis de Desenvolvimento

Estabelece os requisitos, modelagem, construção e teste de um sistema, realizada como uma série de desenvolvimentos menores.



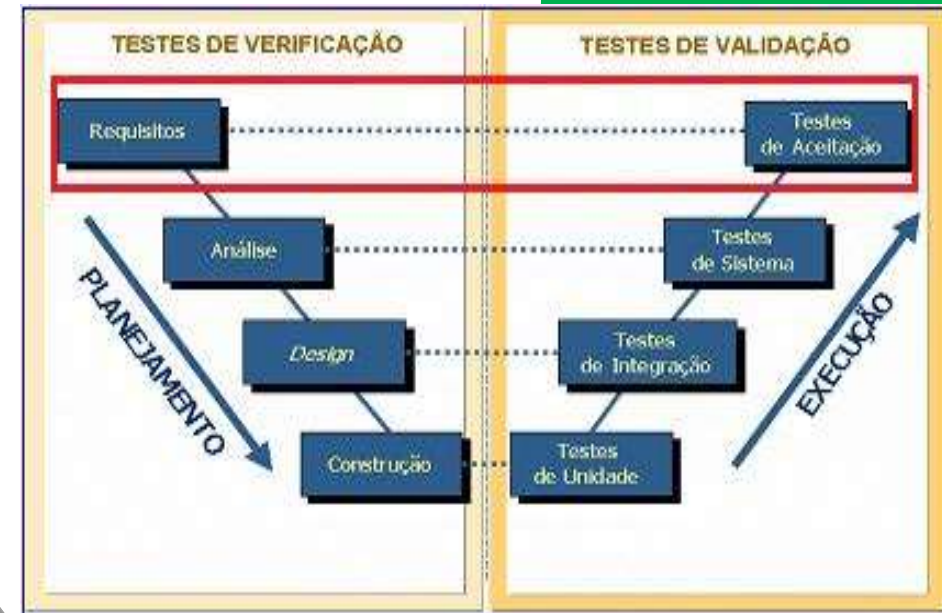
# 3. Testes durante o Ciclo de Vida do Software

## Modelo V

Programadores e a equipe de testes devem garantir que todos os aspectos do projeto foram implementados corretamente no código.

Os quatro níveis usados no syllabus são:

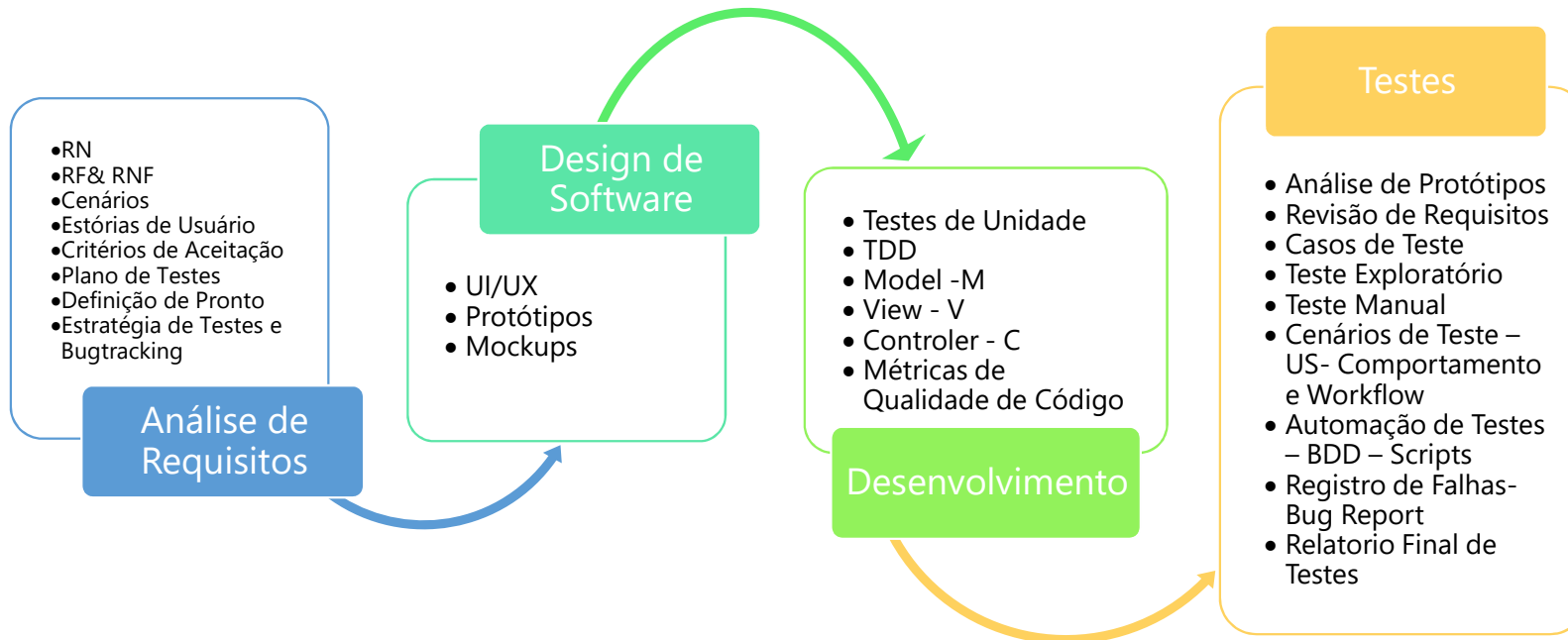
- Teste de Componente (unidade)
- Teste de Integração
- Teste de Sistema
- Testes de Aceitação





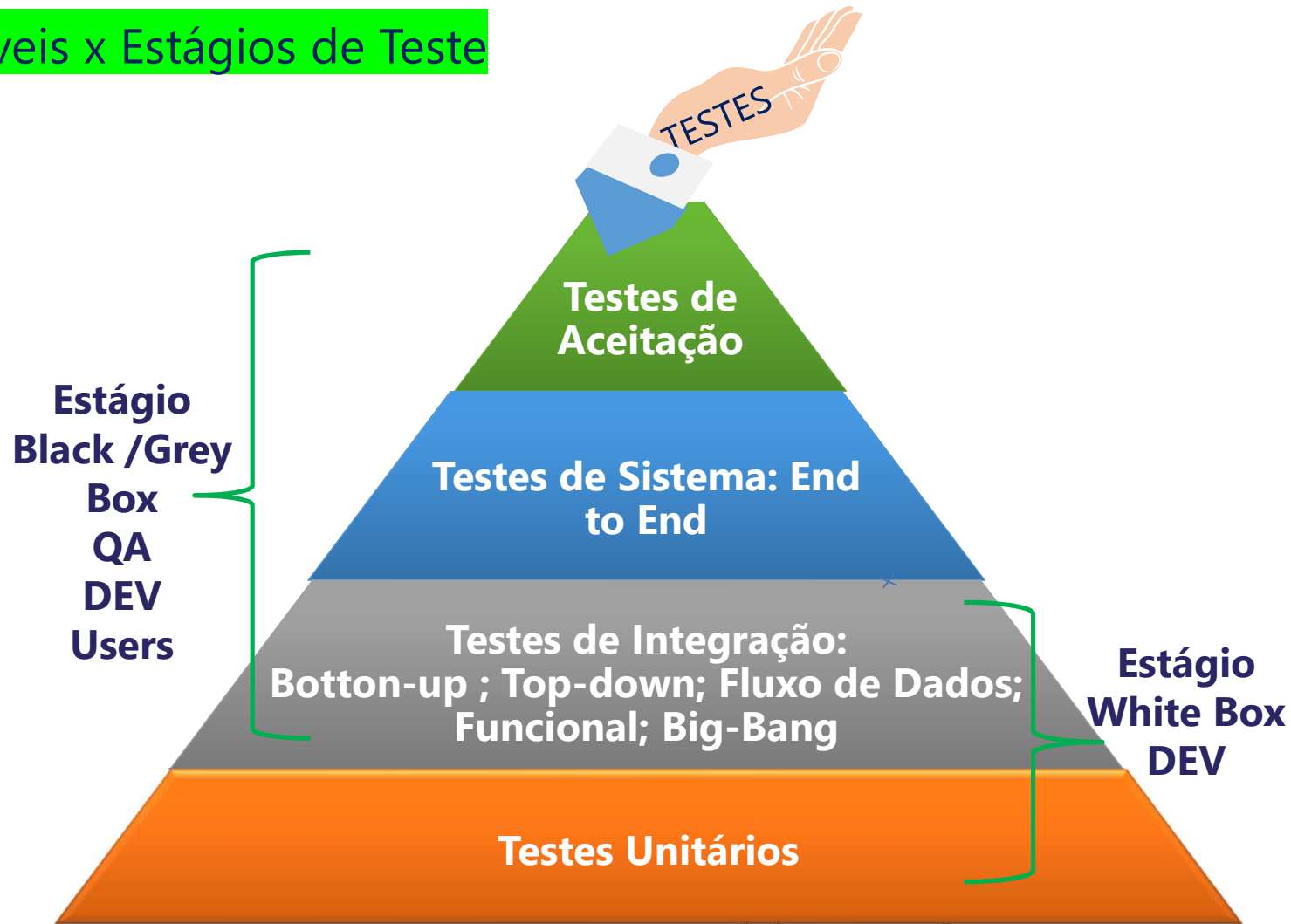
# 3. Testes durante o Ciclo de Vida do Software

## Modelo V



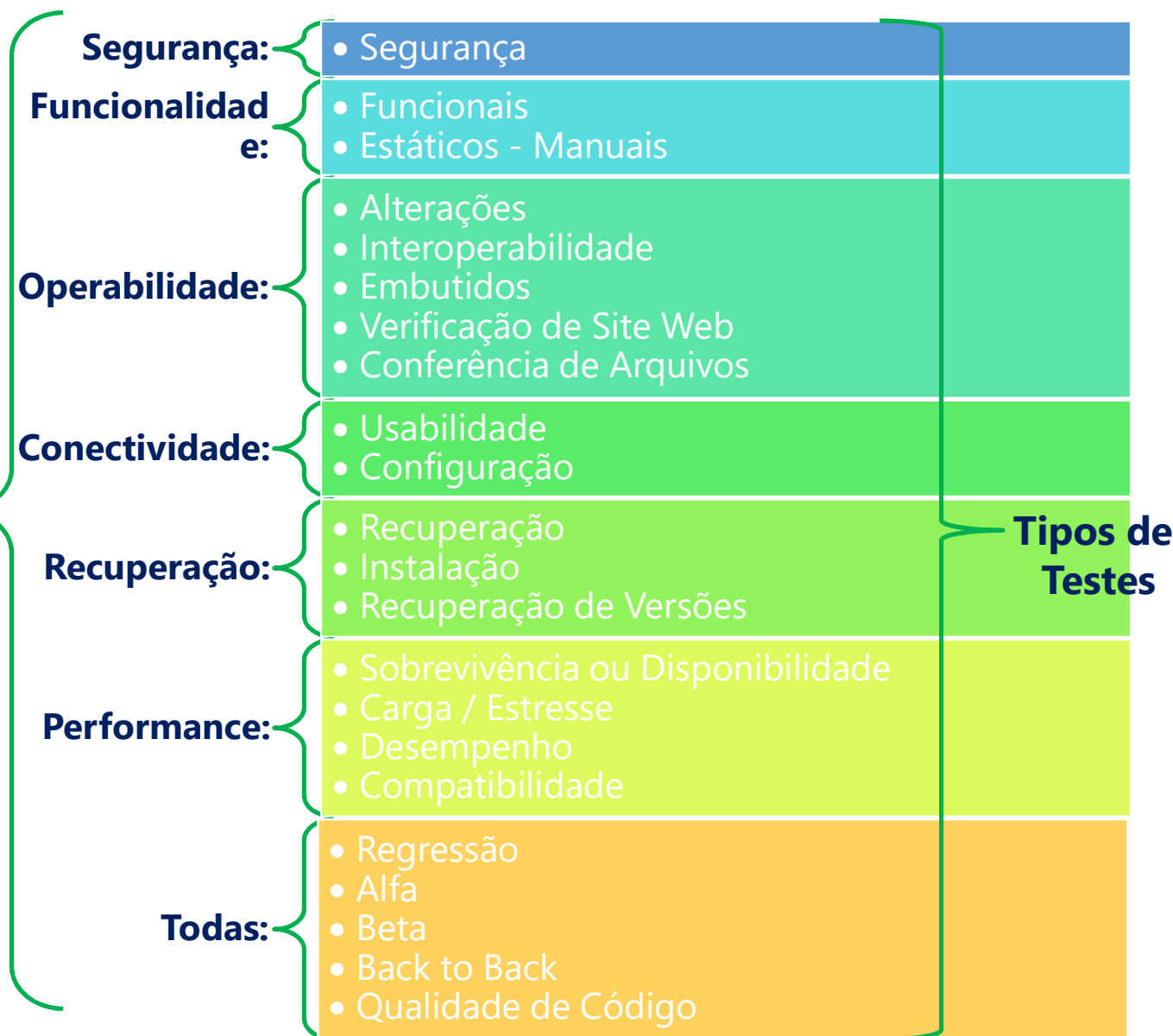
# 4. Técnicas de Teste

## II. Níveis x Estágios de Teste



## Tipos de Teste

### Características de Qualidade:

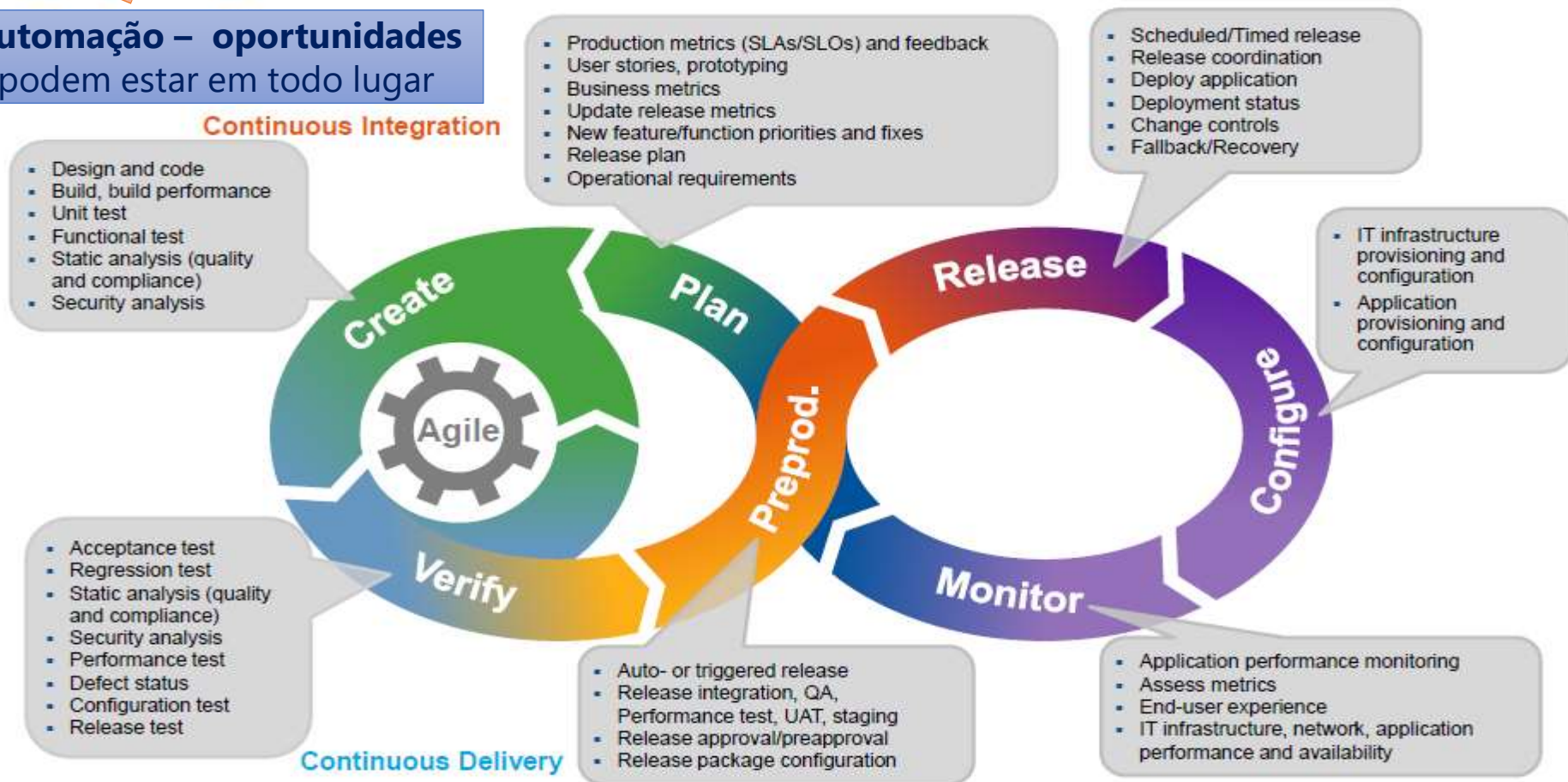


## 4. Técnicas de Teste

# 4 .Técnicas & Práticas

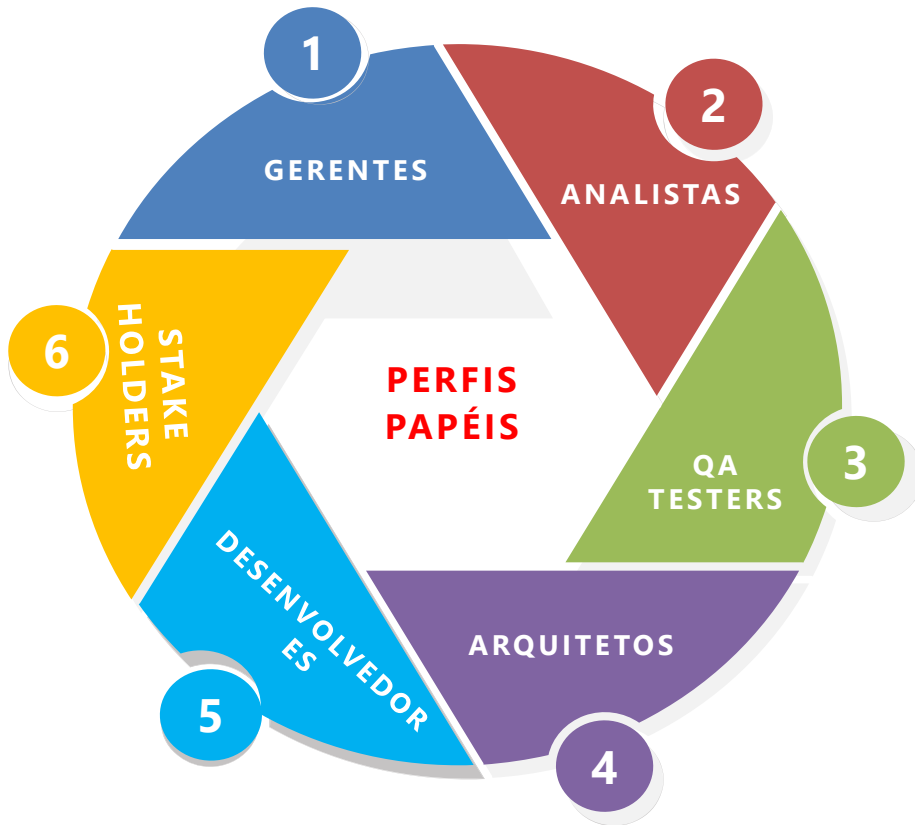
Segundo o Gartner

**Automação – oportunidades  
podem estar em todo lugar**



# Profissionais envolvidos em Projetos de Software

## 1 – Cultura: Pessoas e Papéis



### GERENTES

DE PROJETOS, DE SISTEMAS, DE PORTFÓLIO, SCRUM MASTER, PRODUCT OWNER, INFRAESTRUTURA

### QA

ENGENHEIRO DE QUALIDADE, TESTADORES, ANALISTAS DE TESTES, ANALISTAS DE QUALIDADE

### DESENVOLVEDORES

ENGENHEIROS DE SISTEMAS, ENGENHEIROS DE SOFTWARE, ENGENHEIROS DEVOPS, GATEKEEPER, ENGENHEIROS DE CONFIABILIDADE, DESENVOLVEDORES

### ANALISTAS

DE NEGÓCIOS, REQUISITOS, SEGURANÇA, PROCESSOS, SISTEMAS, DEVOPS

### ARQUITETOS

ARQUITETOS DE SOFTWARE, DE DADOS, DE SEGURANÇA, DE INFRAESTRUTURA, CLOUD

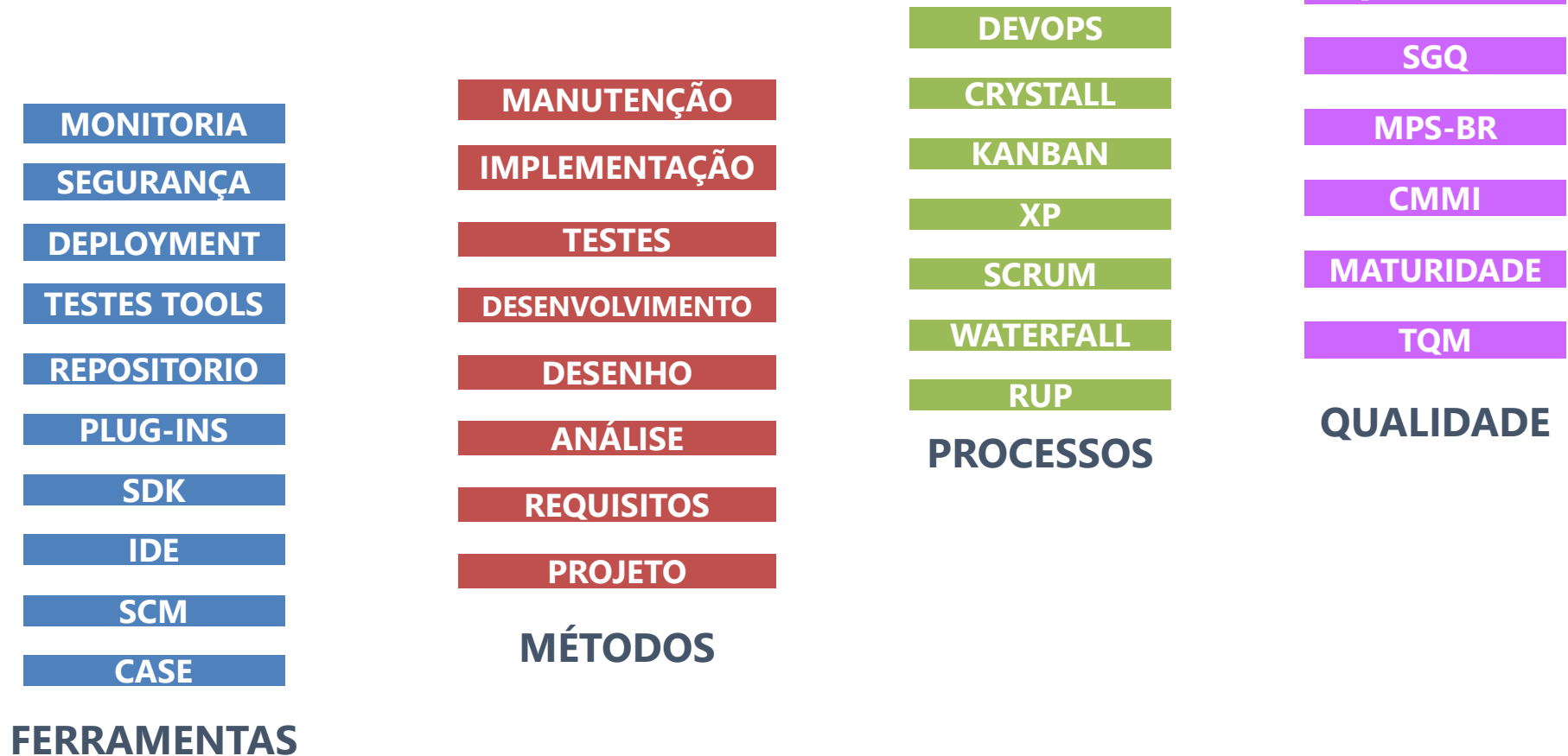
### STAKEHOLDERS

USUÁRIOS CHAVE, CLIENTES FINAIS, SPONSORS

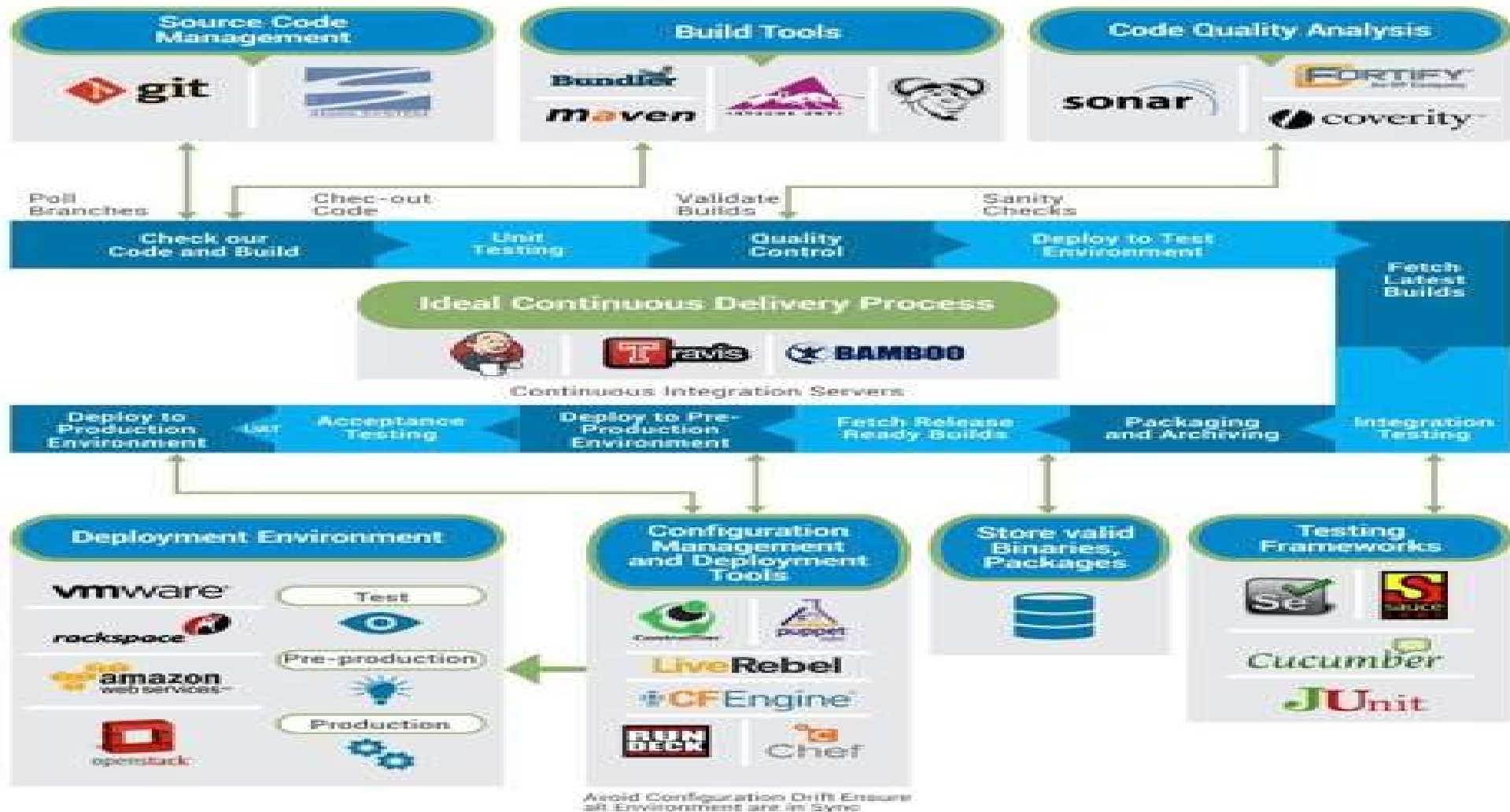
2 – Controle  
3 – Automação  
4 -Visibilidade

# Engenharia de Software

CAMADAS DE DESENVOLVIMENTO DE SOFTWARE









## 5. Ferramentas QA DevOps => Exemplo



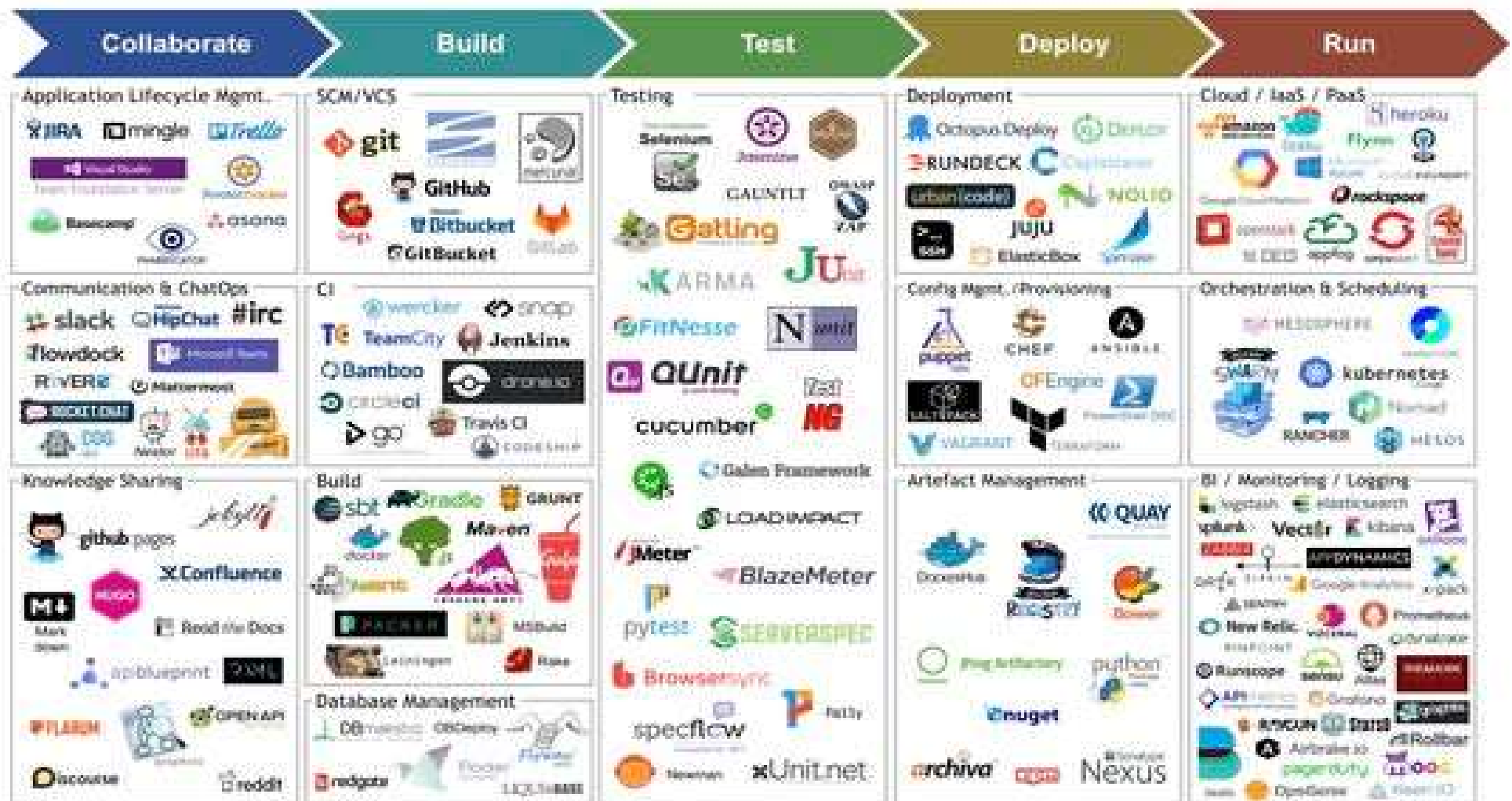


## 5. Comparação de Ferramentas para automação de Testes 30

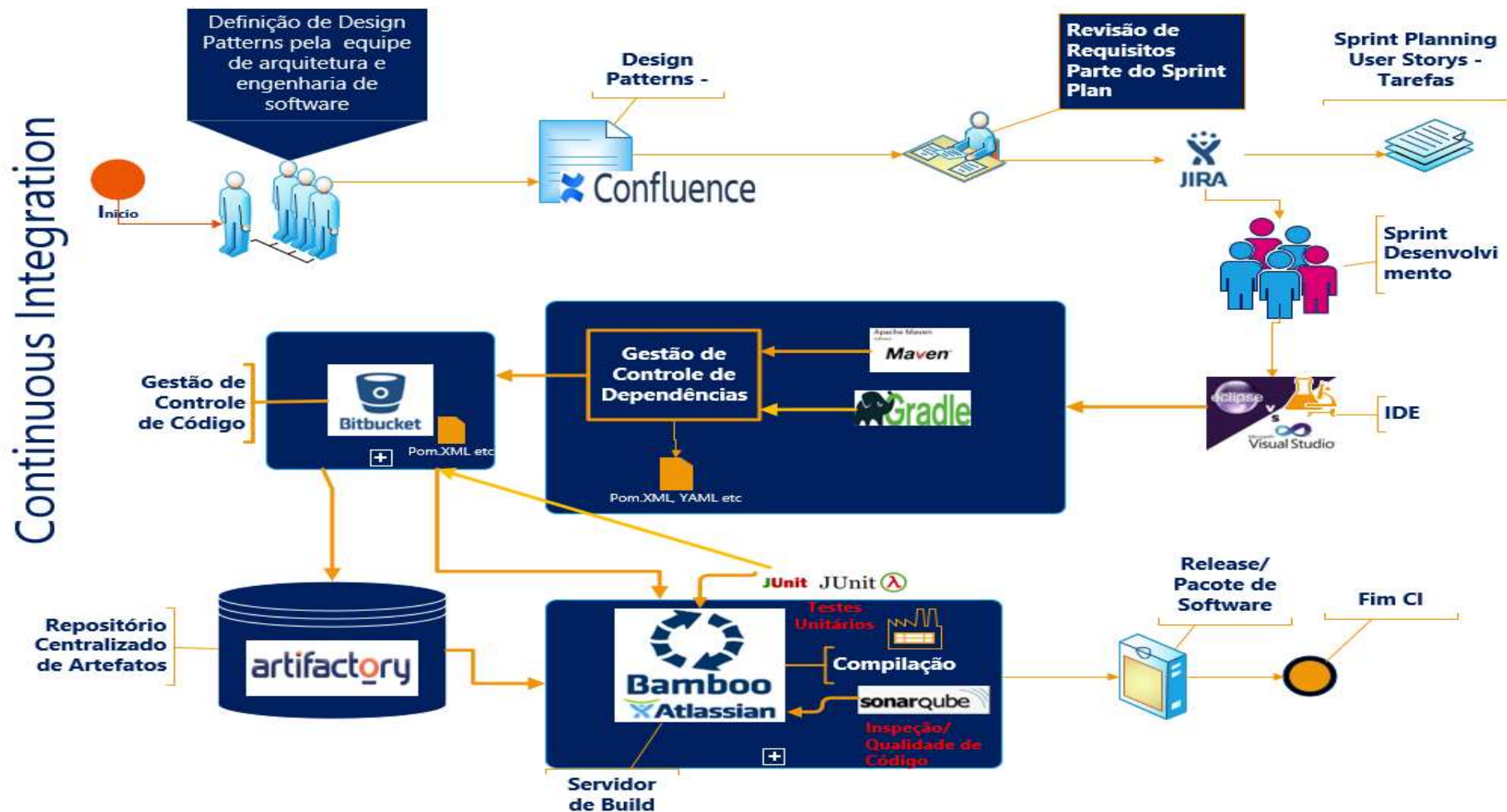
 Ferramentas	 Preço	 Plataforma	 Linguagens Suportadas	 Teste Apps	 Code Skills
Selenium	FREE	Windows  Linux   MacOS	Java, Ruby, Python, PHP, C#, Perl, JavaScript	Web, Mobile (com Appium)	Avançado
TestComplete	\$4600/9000	Windows	VB, Jscript, RubyRails, Delphi, Angular, C#,C++ , JavaScript	Web, Mobile Desktop	Mínimo/ Avançado
Tricentis Tosca	Negociável de acordo com numero de usuários	Windows	JavaScript	Desktop, Web, Mobile	Mínimo/ Avançado
Katalon	FREE	Windows  MacOs	Groovy, Java, Ruby	Web, Mobile	Mínimo/ Avançado
UFT	\$2500/3500	Windows	VB Script	Web, Mobile Desktop	Mínimo/ Avançado
Watir	FREE	Windows	Ruby, C#, Java	Web	Mínimo

## 5. Visão Geral das Possíveis Ferramentas a serem utilizadas:

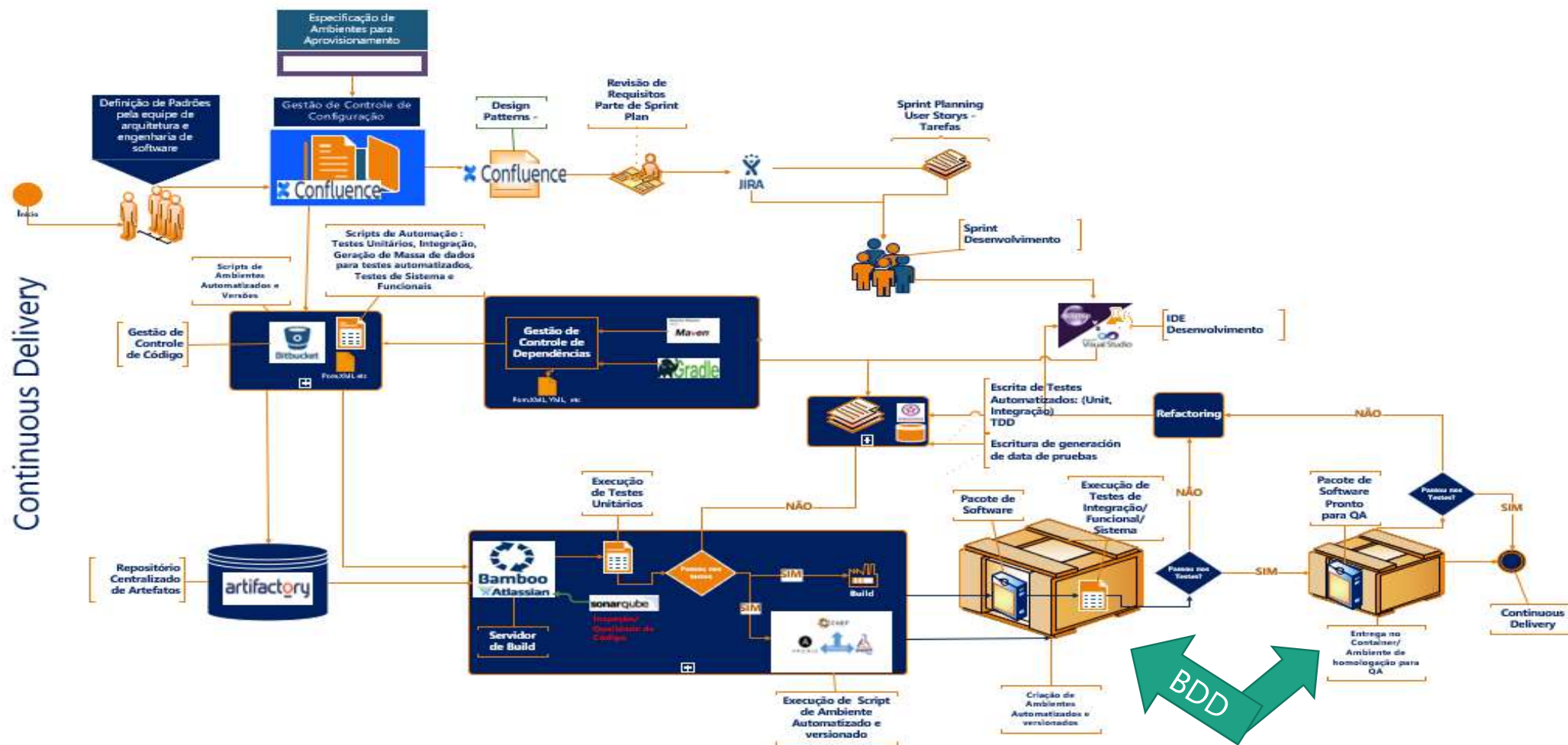
31



## 6. DevOps Pipeline=> Conceitos e Fases



## 6. DevOps Pipeline



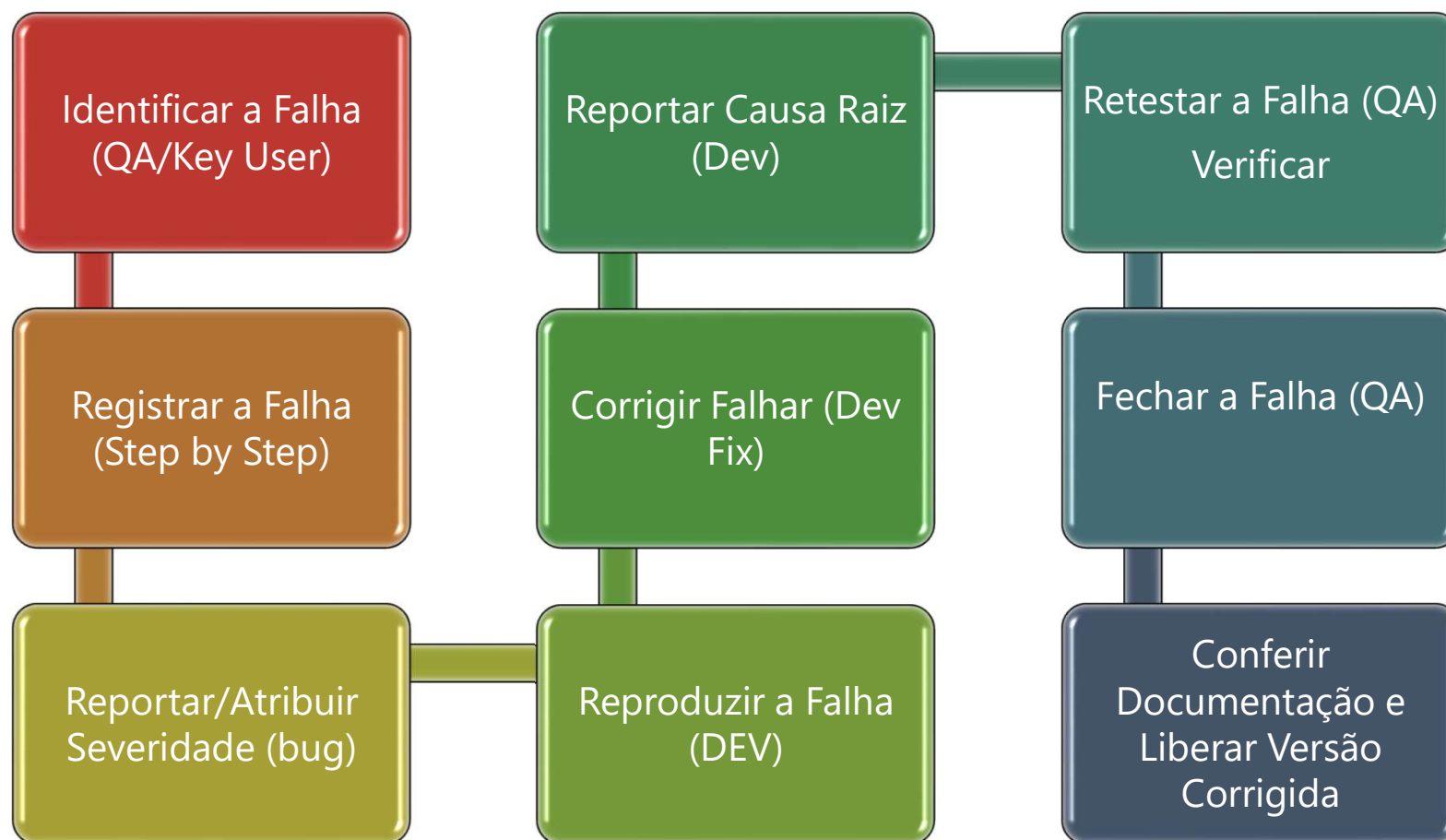
## 7. Gestão de Defeitos

### Bug Tracking Prioridade X Severidade

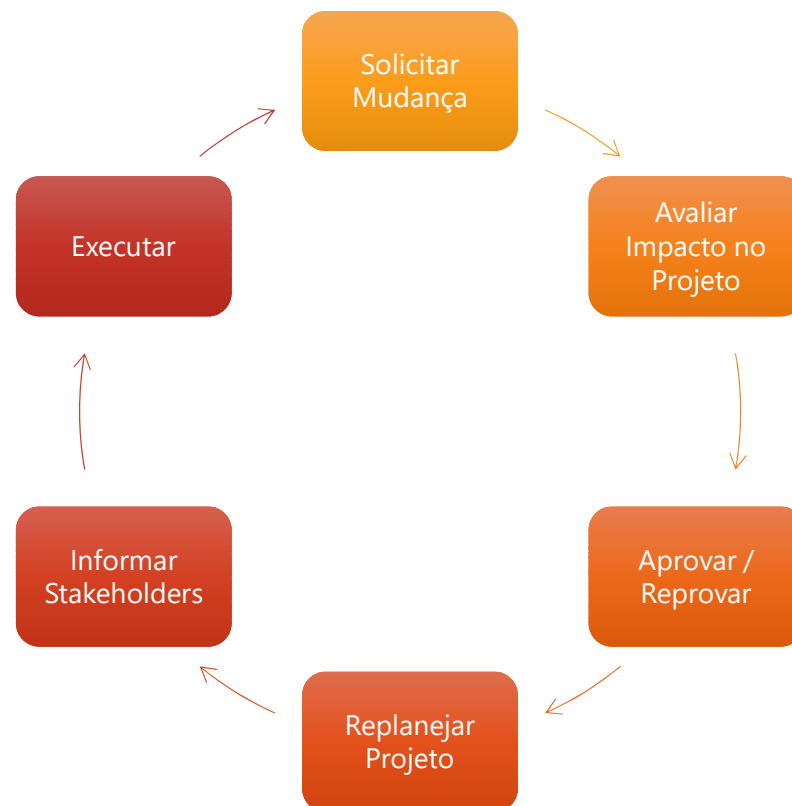


## 7. Gestão de Defeitos

### Gestão de Falhas e Incidentes



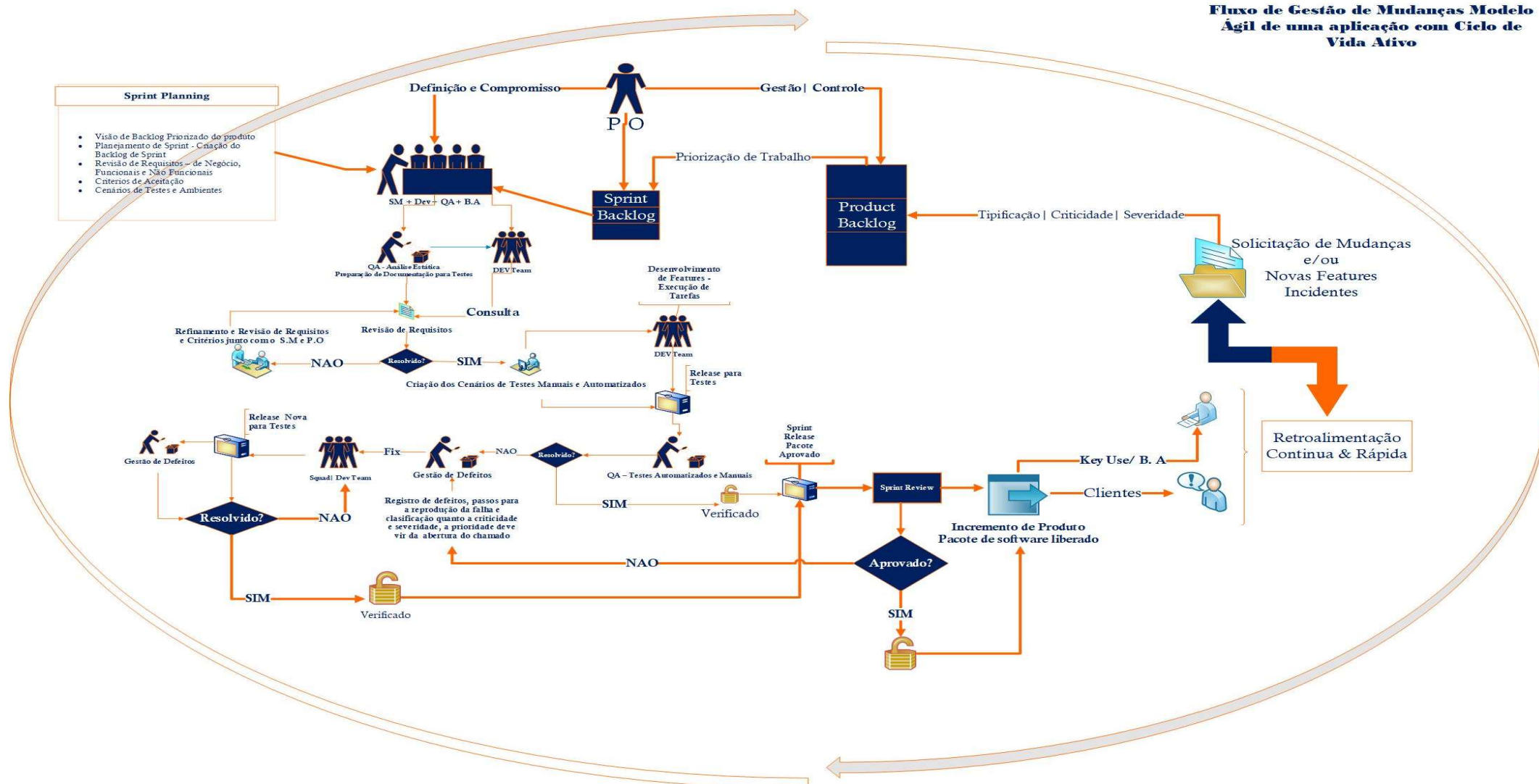
## 8. Gestão de Mudanças:



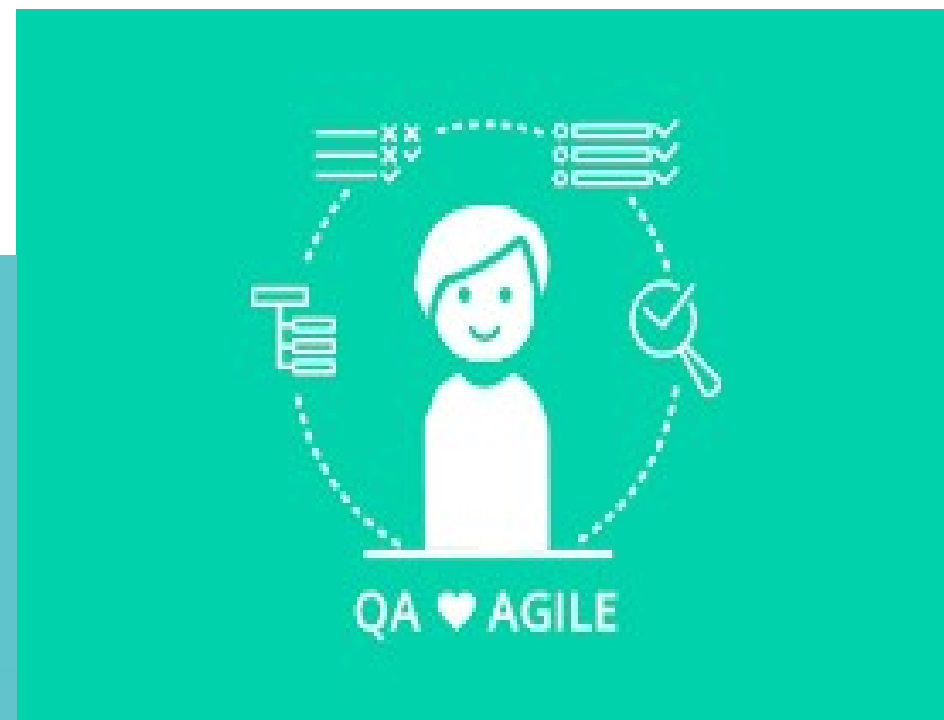


# 8. Gestão de Mudanças:

**Fluxo de Gestão de Mudanças Modelo Ágil de uma aplicação com Ciclo de Vida Ativo**



## 9. Papel do QA:



# QA DevOps => O que deve ser automatizado:

- Testes de Regressão
- Smoke Tests
- Tarefas Repetitivas
- Funcionalidades Críticas do Software
- Testes com Cálculos Matemáticos



## Atenção e Cuidado:

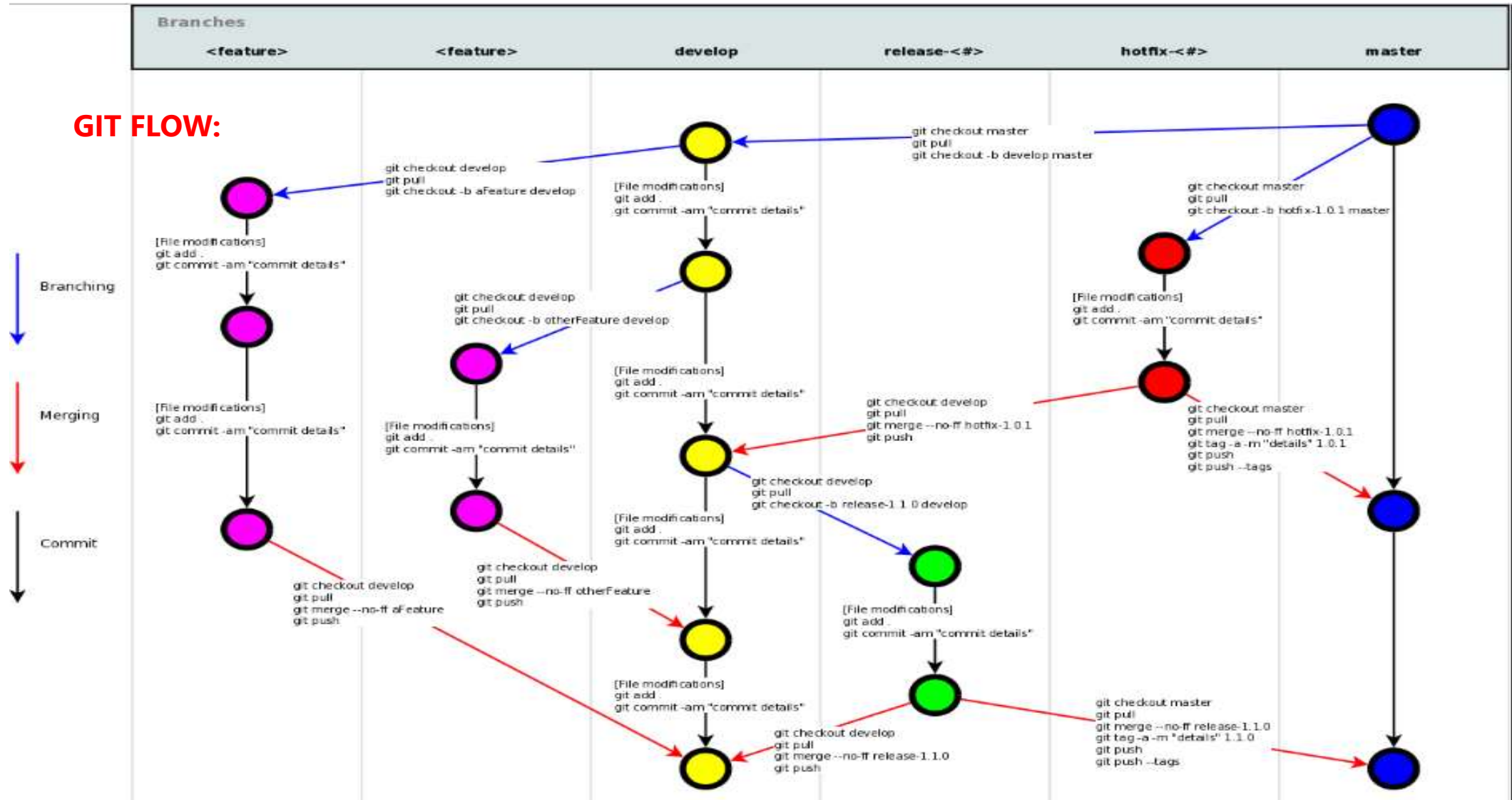
A automatização de testes não é algo simples, quando não aplicada da forma correta pode trazer mais problemas que benefícios.

- Dependência de Ferramentas
- Exige tempo demais para criação dos Testes
- Necessidade de Profissionais especializados
- Mudanças contínuas de interface geram refactoring nos scripts
- Podem ser executados a qualquer horas, repetidas vezes
- Menos sujeitos a falhas humanas

# 10. Boas Práticas:

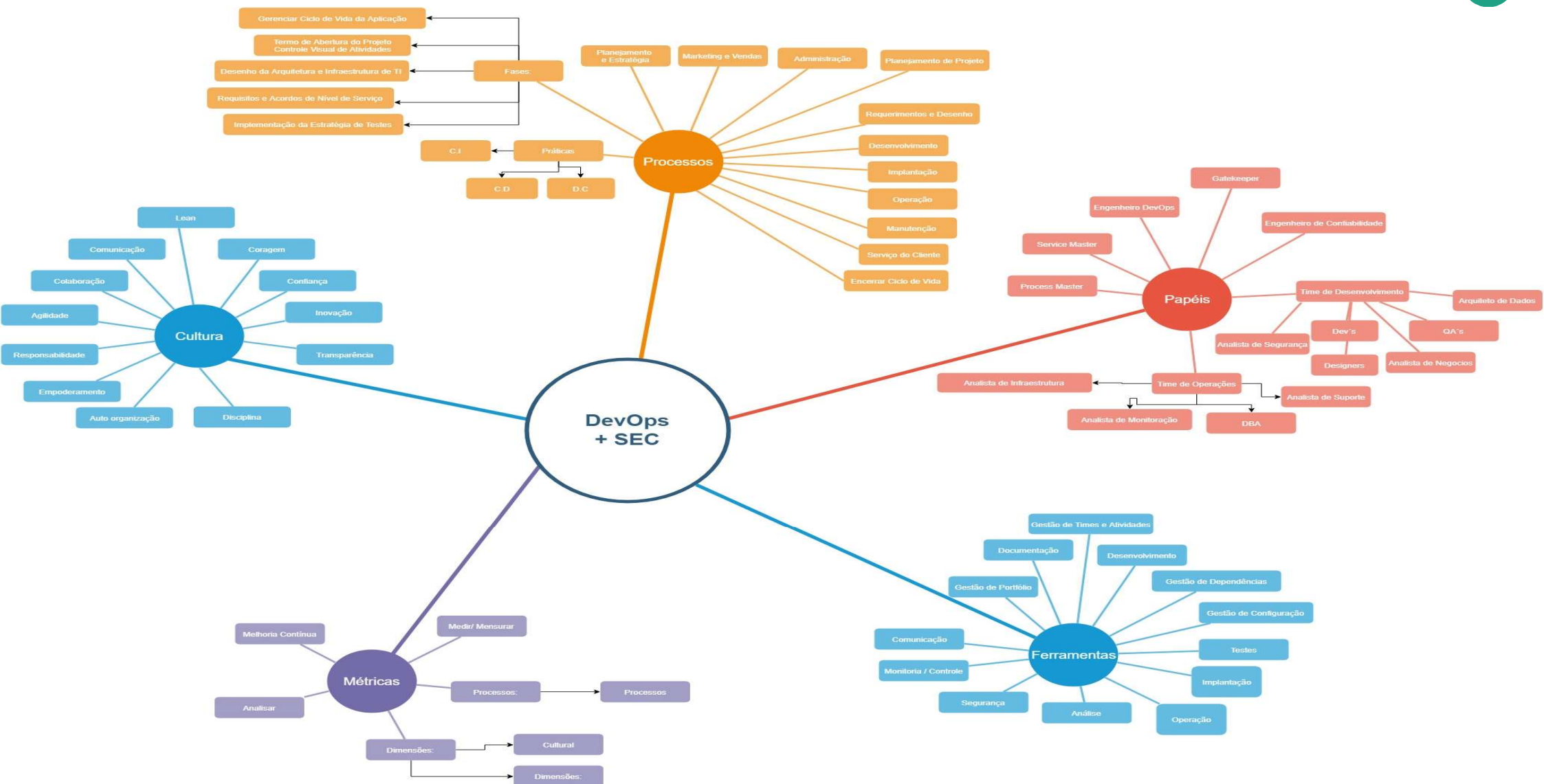
40

## GIT FLOW:



# 10. Boas Práticas:

41



## 10. Boas Práticas: Gestão de Configuração

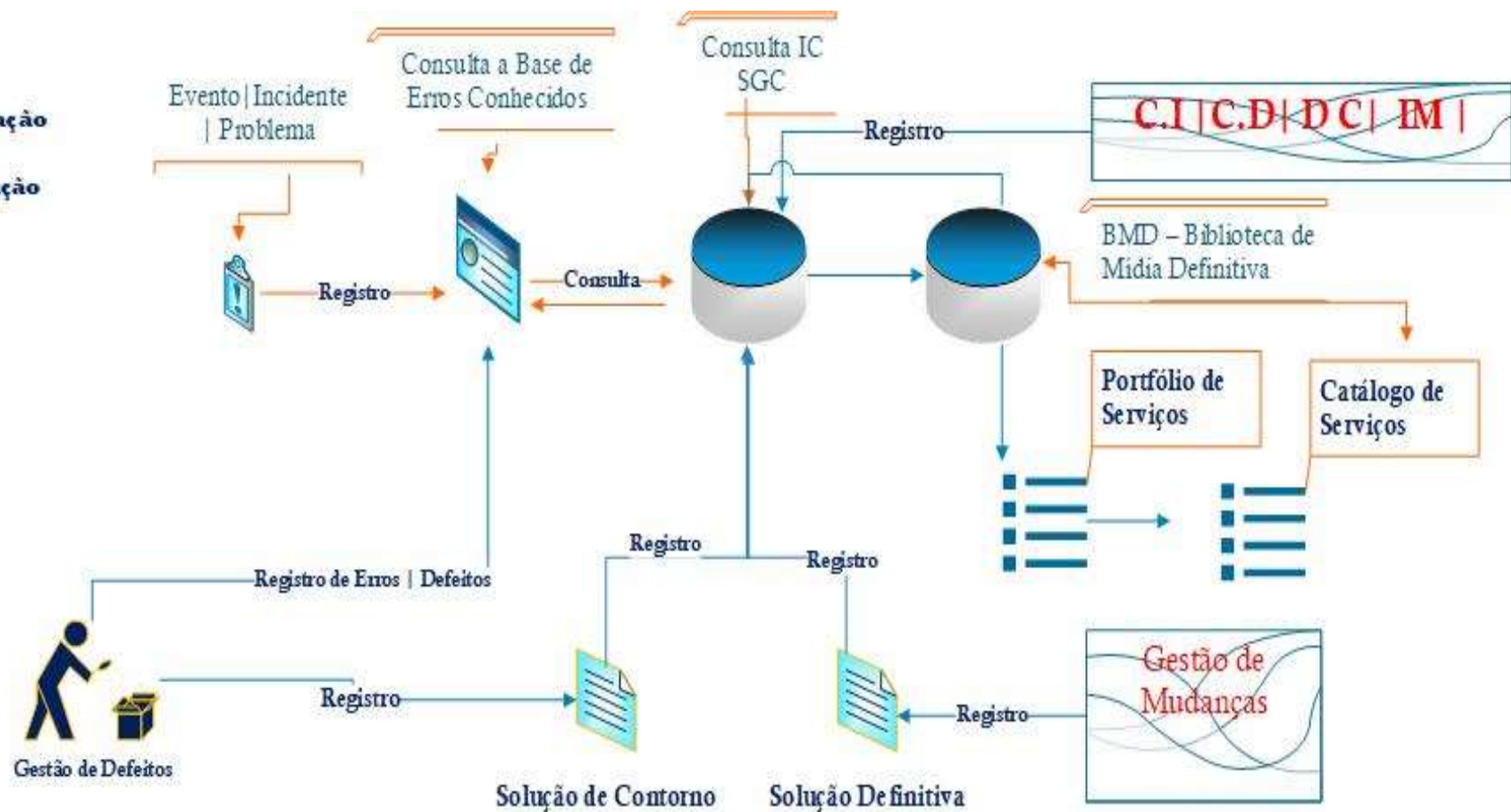
- **GCS** ou em inglês *Software Configuration Management (SCM)*, que é um conjunto de atividades de apoio que permitem controlar as mudanças que ocorrem no desenvolvimento de software, mantendo a estabilidade na evolução do projeto.
- A **Gerência de Configuração de Software** responde a algumas questões como: Quais mudanças aconteceram no sistema? Por que essas mudanças aconteceram? O sistema continua íntegro mesmo após as mudanças?
- Para se entender a Gerência de Configuração de Software é interessante definir-se o que vem a ser configuração.
- Configuração de um sistema é uma coleção de versões específicas de itens de configuração como hardware ou software que são combinados de acordo com procedimentos específicos de construção para servir a uma finalidade particular. A Gerência de Configuração de Software por sua vez é uma disciplina que identifica a configuração de um sistema em diferentes pontos no tempo com a finalidade de controlar sistematicamente as mudanças realizadas, mantendo a integridade e rastreabilidade da configuração através do ciclo de vida do sistema.



# 10. Boas Práticas: Gestão de Configuração

43

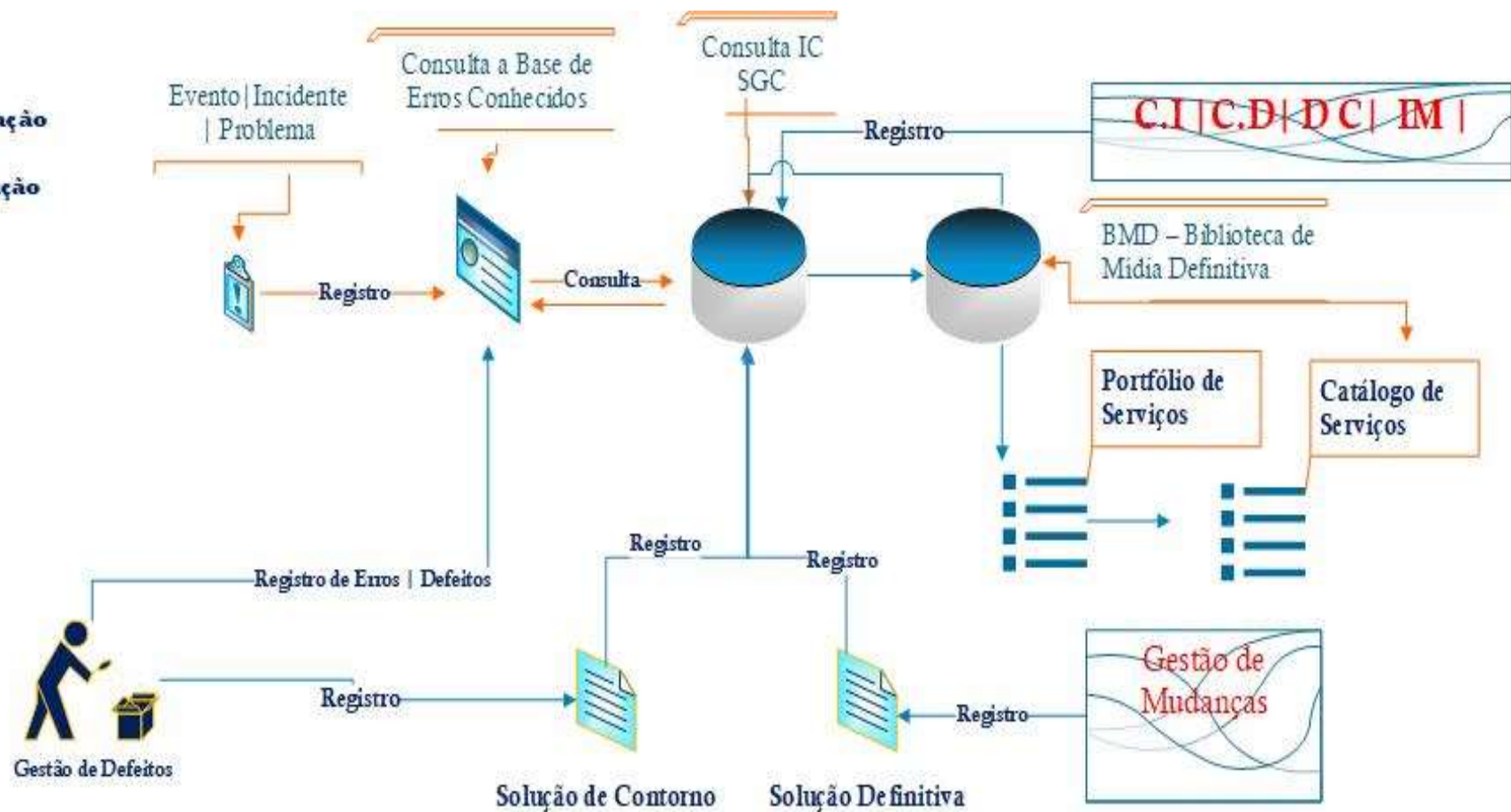
**Fluxo de Gestão de Configuração**  
**Modelo Agil**  
**Ciclo de Vida Ativo | Produção**



# 10. Boas Práticas: Gestão de Configuração

44

**Fluxo de Gestão de Configuração**  
**Modelo Agil**  
**Ciclo de Vida Ativo | Produção**





## Referências:

- <https://www.siteware.com.br/gestao-de-equipe/gestao-de-mudancas/>
- <https://www.contino.io/insights/they-call-it-royale-with-cheese-what-gdpr-means-for-australia>
- <https://www.f5.com/services/resources/white-papers/using-docker-container-technology-with-f5-products-and-services>
- Exin White Paper
- MORAIS, Gleicon - "CAIXA DE FERRAMENTAS DEVOPS – Casa do Código, 2017 São Paulo, SP.
- [https://www.ibm.com/developerworks/community/blogs/a9ba1efe-b731-4317-9724-a181d6155e3a/entry/accelerating\\_maximo\\_development\\_with\\_continuous\\_delivery?lang=en](https://www.ibm.com/developerworks/community/blogs/a9ba1efe-b731-4317-9724-a181d6155e3a/entry/accelerating_maximo_development_with_continuous_delivery?lang=en)
- <https://www.gp4us.com.br/backlog-do-produto/>
- <http://www.datacenterdynamics.com.br/focus/archive/2015/02/empresas-brasileiras-faturam-29-mais-ao-adotar-devops-diz-pesquisa-da-ca-techn>
- <https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/beyond-agile-reorganizing-it-for-faster-software-delivery>
- [https://devops.com/integrating-itol-change-management-and-devops/?\\_\\_hstc=82446857.f17651f71ffeced04f16650f088009bd.1524442511418.1524442511418.1524442511418.1&\\_\\_hssc=82446857.1.1524442511420&\\_\\_hsfp=3548929705](https://devops.com/integrating-itol-change-management-and-devops/?__hstc=82446857.f17651f71ffeced04f16650f088009bd.1524442511418.1524442511418.1524442511418.1&__hssc=82446857.1.1524442511420&__hsfp=3548929705)
- <https://gaea.com.br/tudo-que-voce-precisa-saber-sobre-integracao-itol-e-devops/>
- <https://novacontext.com/azure-strategy-and-implementation/>
- <http://www.forumdaconstrucao.com.br/conteudo.php?a=0&Cod=1860>
- <https://www.mandic.com.br/artigos/devops-significado-do-termo>
- <https://www.computer.org/web/education/associate-certifications>
- <https://www.computer.org/ieeecs-swebokdelivery-portlet/swebok/SWEBOKv3.pdf?token=t6dRawnbArvLuPu7rAi142X5Xqwe4FFH>
- <http://www.nuvemlab.com.br/blog/etapas-mais-comuns-do-desenvolvimento-de-um-website/>
- <https://www.iso.org/committee/45086/x/catalogue/>
- <http://sites.computer.org/ccse/SE2004Volume.pdf>

