CTEDS

Capacitação Tecnológica em Engenharia e Desenvolvimento de Software

Métodos Ágeis

eXtreme Programming (XP)

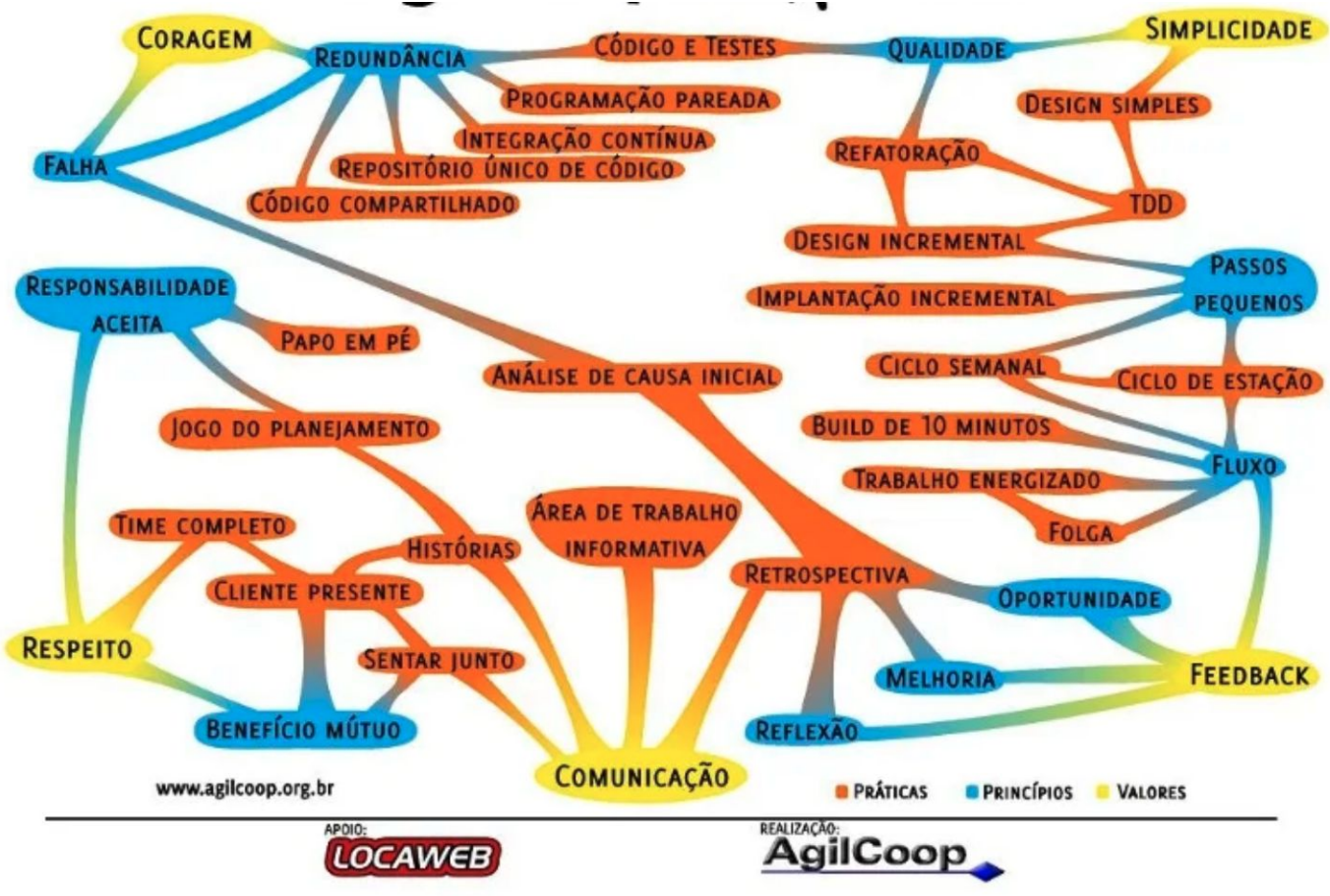
Professor: Carla Rocha [UnB] *caguiar@unb.br*

15 de Agosto de 2022

Agenda

- XP - práticas
- Princípios da Gestão Ágil

Programação Extrema



Princípios básicos de XP

Feedback rápido

Simplicidade é o melhor negócio

Mudanças incrementais

Carregue a bandeira das mudanças/não valorize o medo (Embrace change)

Alta qualidade do código

4 variáveis do desenvolvimento

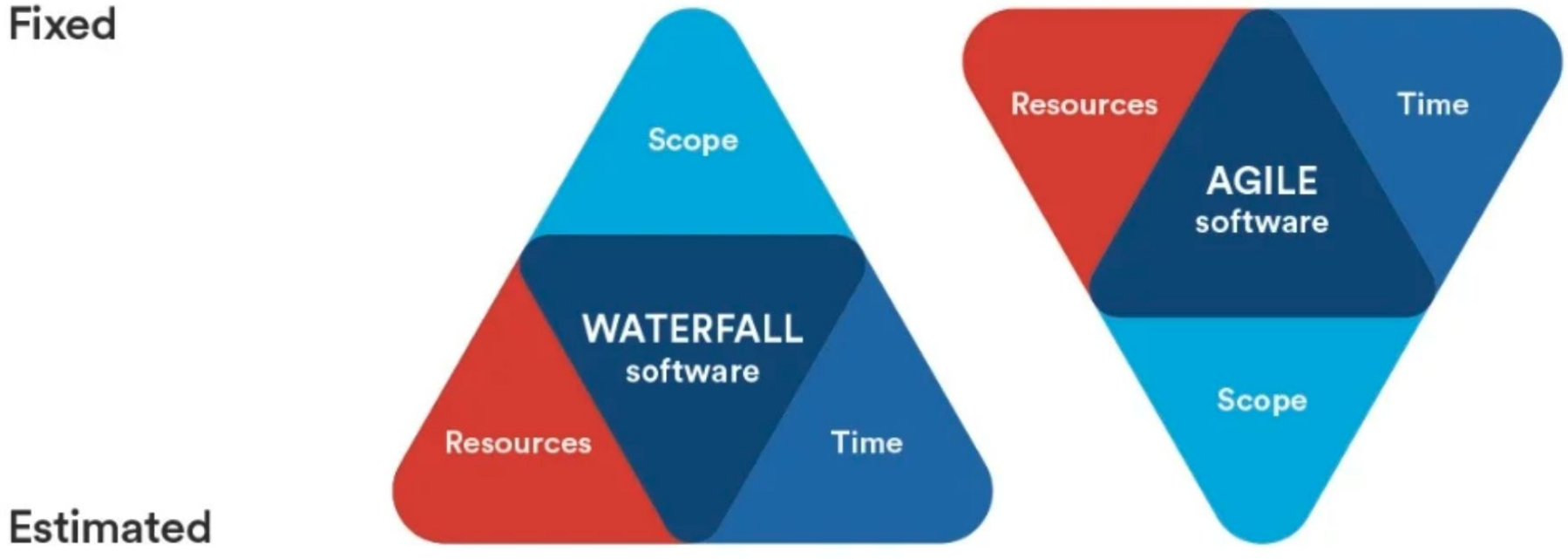
Tempo

Custo

Qualidade

Escopo (foco principal de XP)

4 VARIÁVEIS E O TRIÂNGULO DE FERRO

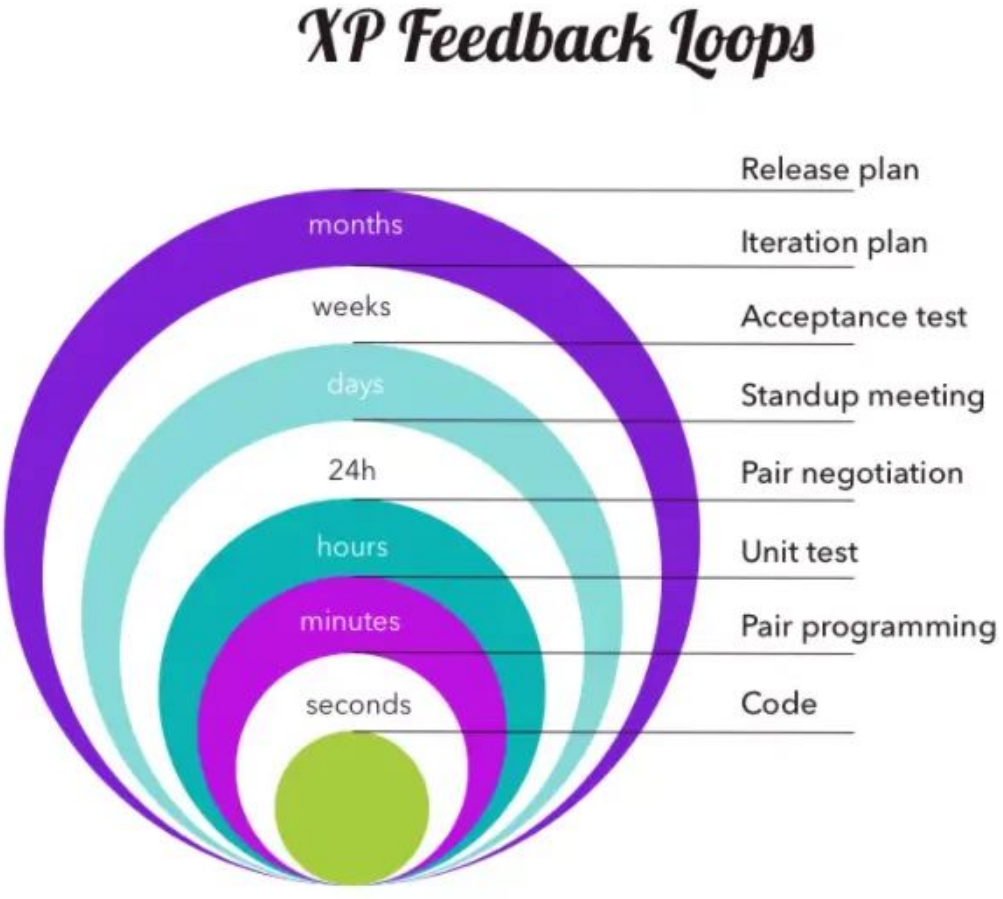


https://www.atlassian.com/agile/agile-iron-triangle/sectionWrap/00/column/00/moreContent/05/imageBinary/waterfall-v-agile-iron-triangle-v03.png

Um projeto XP

- Fase de Exploração
  - Duração: 1 a 6 meses
  - Termina quando a primeira versão do software é enviada ao cliente
  - Clientes escrevem "histórias"(story cards)
  - Programadora/es interagem com clientes e discutem tecnologia
  - Não só discutem, **experimentam** diferentes tecnologias e arquiteturas para o sistema

XP


XP Feedback Loops
months
weeks
days
24h
hours
minutes
seconds
Release plan
Iteration plan
Acceptance test
Standup meeting
Pair negotiation
Unit test
Pair programming
Code

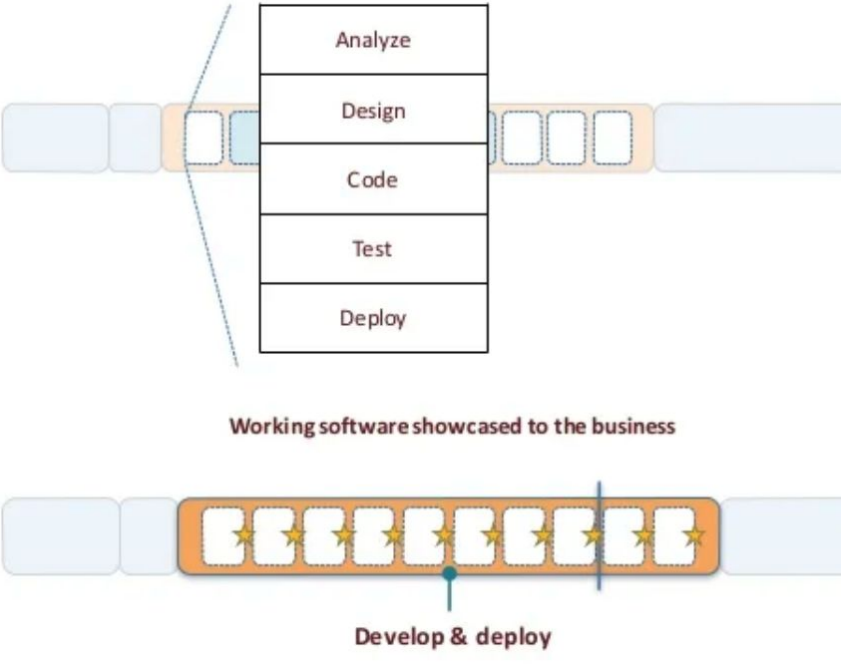
Um dia na vida do/a programador/a XP

Um dia na vida do/a XP

- Escolhe a próxima história mais importante do cliente
- Procura um par livre
- Seleciona um "cartão de história" contendo uma tarefa claramente relacionada a uma característica (*feature*) desejada pelo cliente

Um dia na vida do/a XP

- Discute modificações recentes no sistema
- Discute história do cliente (*Business Analysis*)
- Testes (*Test-Driven Development*)
- Implementação
- Projeto (*design*)
- Integração do código



Um dia na vida do/a XP

- Atos constantes no desenvolvimento:
  - Executa Testes antigos
  - Busca oportunidades para simplificação
  - Modifica desenho e implementação incrementalmente baseado na funcionalidade exigida no momento
  - Escreve testes
  - Enquanto todos os testes não estão 100%, o trabalho não está terminado
  - Integra novo código ao repositório

Testes

- Fundamento mais importante do XP
- É o que dá segurança e coragem ao grupo
- Testes de Unidade (*Unit Tests*)
    - Escritos, por quem desenvolve, para testar cada elemento do sistema (ex: cada método de cada caso)
- Testes de Funcionalidades (*Functional tests*)
    - Escritos pelos clientes para testar a funcionalidade do sistema (Testes de Aceitação)

Testes

- Unidade
- Integração
- Interface de Usuário
- Aceitação
- Mutação
- Desempenho
- Estresse
- Segurança

Quando escrever testes

- Antes de escrever o código (TDD)
- Durante a implementação
- Antes da refatoração
- Quando um erro for encontrado: Escreva um teste que simule o erro e depois corrija-o
- Quando um teste de aceitação falha, pode ser um sinal que testes de unidade estão faltando
- Quando um usuário ou cliente encontra um erro, é sinal que estão faltando testes de unidade e de aceitação

Quando executar testes

- A execução deve ser ágil
- Sempre que uma nova porção de código é criada
- À noite ou em uma máquina dedicada
- Assim que alterações são detectadas no repositório
- Lembre-se: é melhor escrever e executar testes incompletos do que não executar testes completos

Por que a automação importa em Agile?

- Entre as práticas mais importantes
- A mais importante em muitos contextos
- Muitas das práticas dependem de testes automatizados:
  - Refatoração
  - Integração contínua
  - Propriedade Coletiva do Código
  - Ainda: design simples, releases pequenos

Métricas

- **Métrica:** medida quantitativa do grau em um sistema, um componente, ou processo possui um determinado atributo
- **Métrica de produto:** "Métrica usada para medir uma característica de qualquer produto intermediário ou na resultante do processo de desenvolvimento de software"
- **Métrica de Processo:** "Métrica usada para medir características de método, técnicas e ferramentas empregadas no desenvolvimento, implementação e manutenção de sistemas de software"

Automação de Métricas

- Existem iniciativas de automação da coleta de diferentes tipos de métricas
- Maioria destinada à coleta de métricas de produto a partir do código fonte
  - Code climate (https://codeclimate.com)
  - SonarQube (https://www.sonarqube.org)

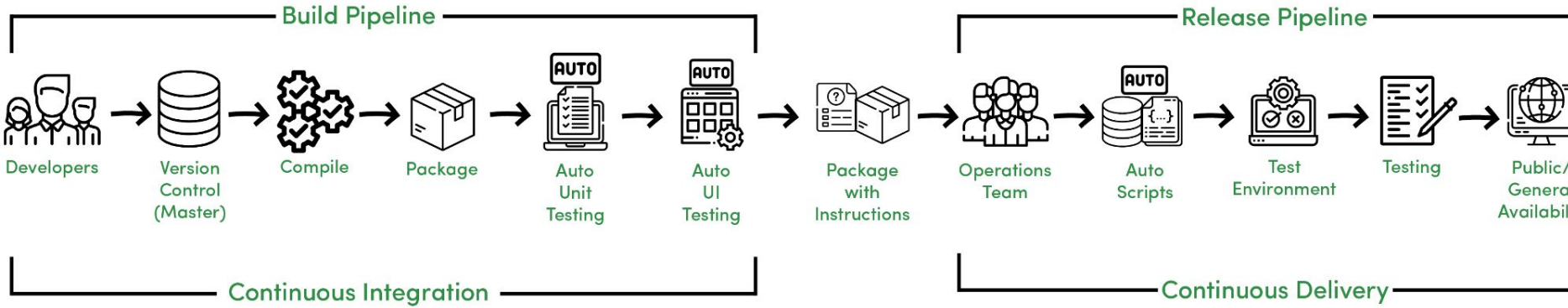
Automação de Análise Estática

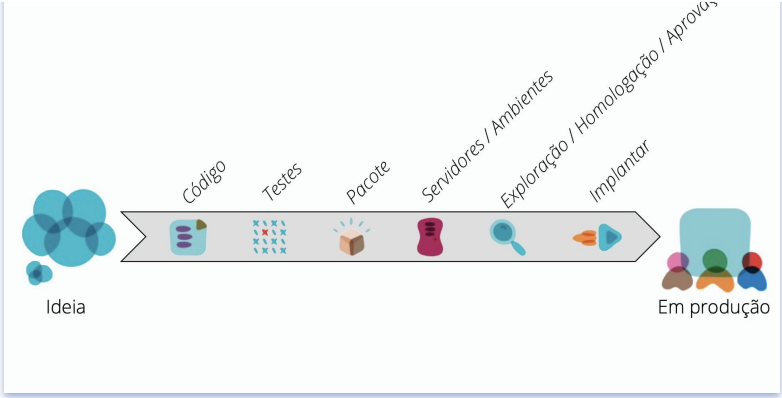
- Eficaz para
  - Verificações rápidas e baratas de propriedades simples
    - Exemplo: análises de fluxo de dados simples podem identificar padrões anômalos
  - Verificações caras necessárias para propriedades críticas
    - Exemplo: Ferramenta de verificação de estado finito para encontrar falhas de sincronização

Métrica e ferramenta: cobertura de código

- Métrica Cobertura de Código (*code coverage)*
- Útil para guiar que testes fazer e os quais não fazer

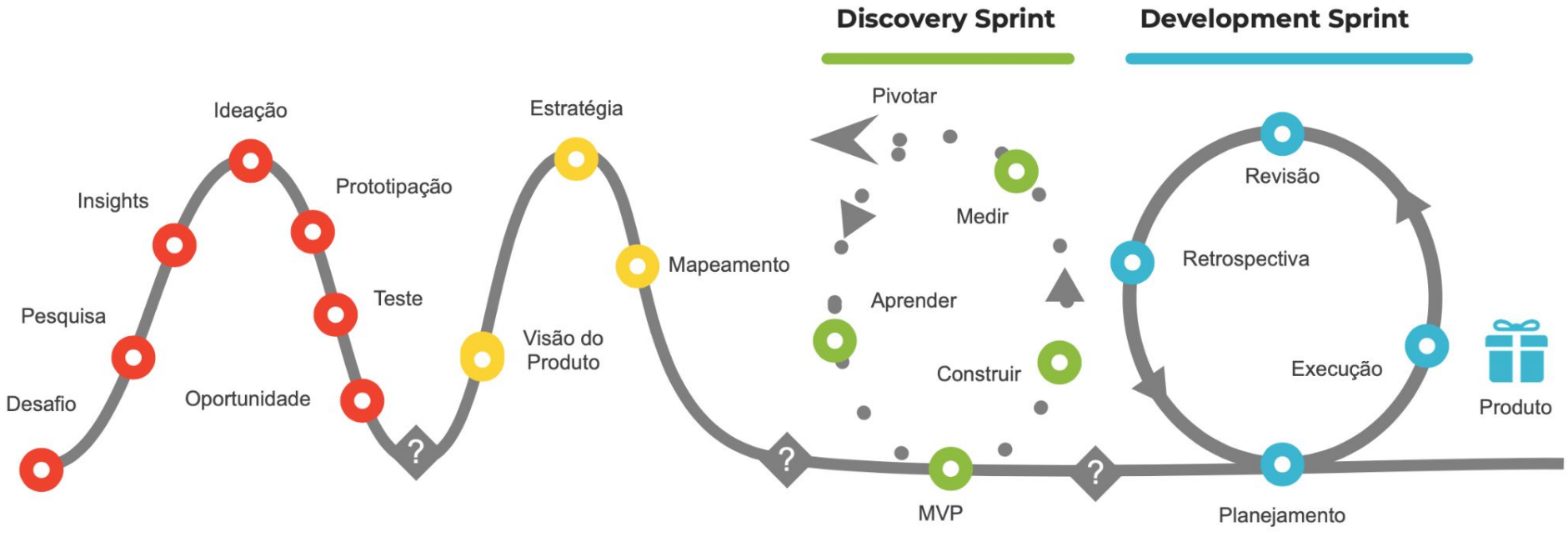
Integração Contínua e pipelines



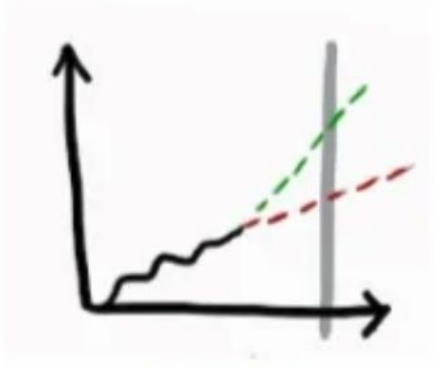


Princípios da Gestão ágil

FRAMEWORK AGILE DESIGN

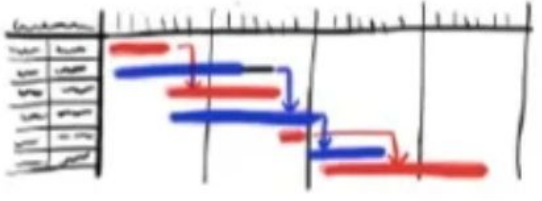


100% predictability = 0% innovation



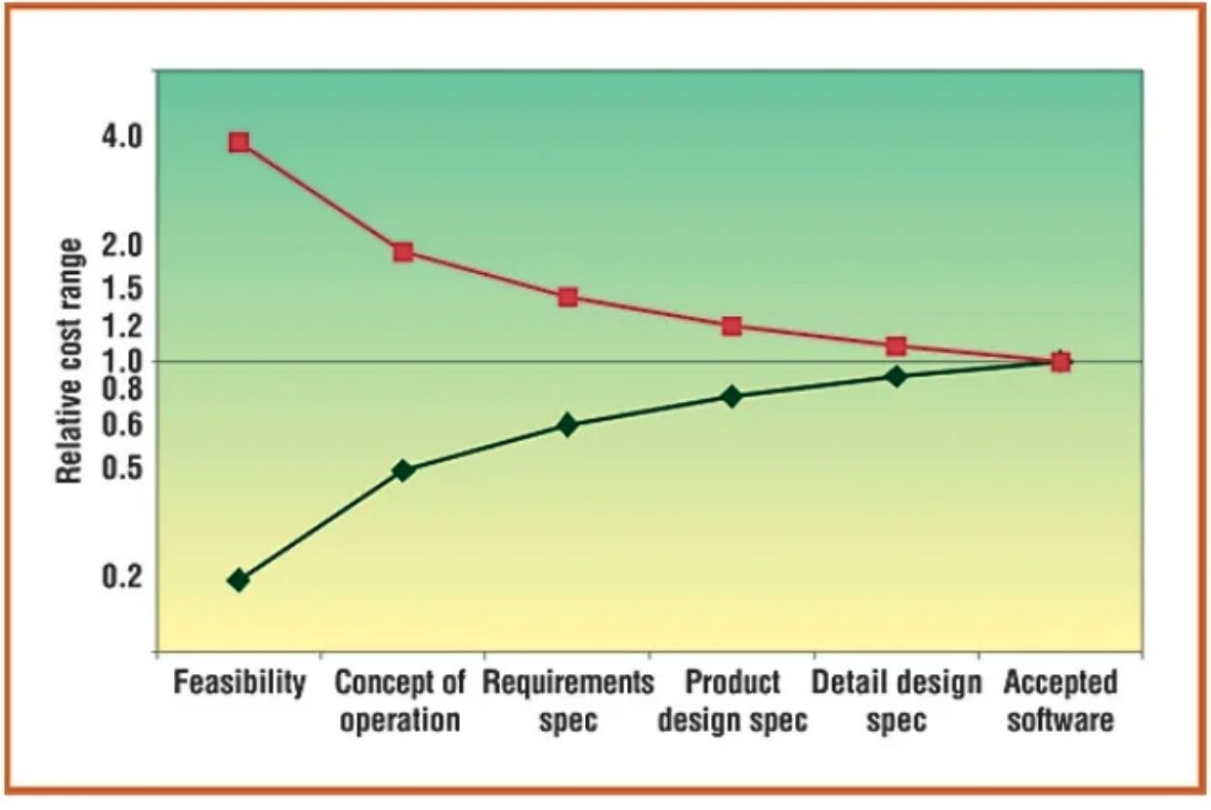
Focus on
Innovation

Focus on
Predictability



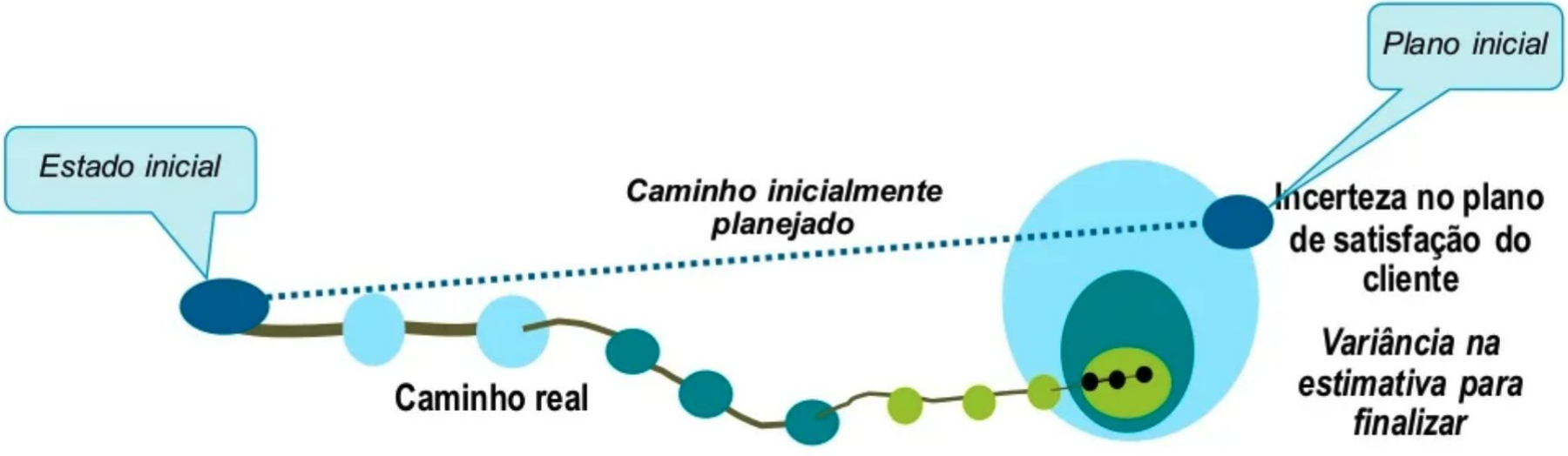
Value delivery > Plan fulfillment

Cone da Incerteza

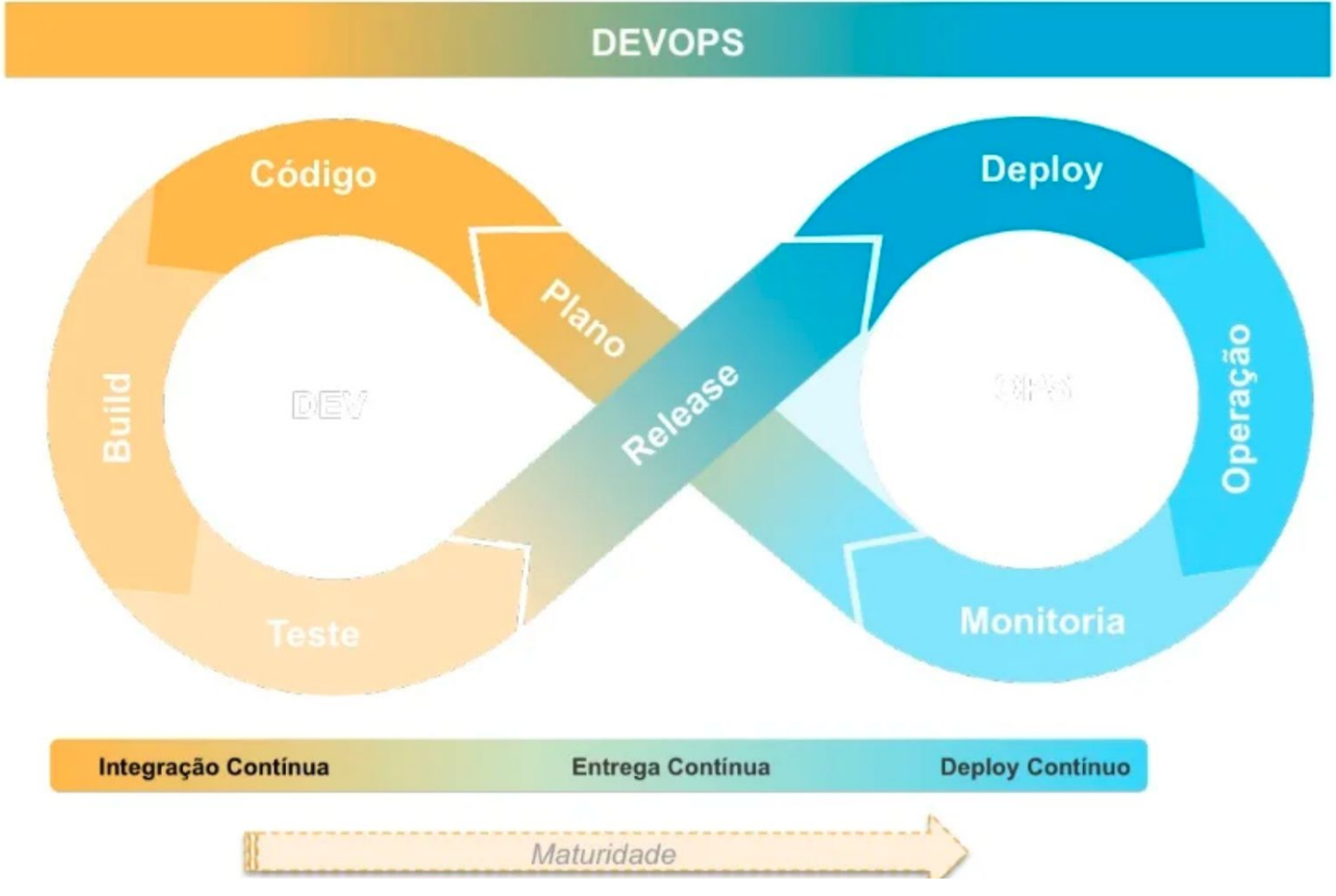


https://doi.ieeecomputersociety.org/cms/Computer.org/dl/mags/so/2006/03/figures/s30484.gif

Incerteza


Plano inicial
Estado inicial
Caminho inicialmente planejado
Incerteza no plano de satisfação do cliente
Variância na estimativa para finalizar
Caminho real

DevOps


DEVOPS
Código
Plano
Build
DEV
Release
Deploy
Operação
Monitoria
Teste
Integração Contínua
Entrega Contínua
Deploy Contínuo
Maturidade

Gestão de Projetos Ágeis

"A gestão de projetos ágeis requer uma abordagem diferente, no qual é adaptado para o desenvolvimento incremental e os pontos fortes dos métodos ágeis.

No seu *core*, a **gestão de projetos ágeis é sobre gestão do impacto da complexidade e incerteza de um projeto**, reconhecendo:

A necessidade de **diminuir a janela de tempo entre planejamento e execução**

O **planejamento de uma atividade não provê todos os detalhes de sua implementação**

A **criatividade e o aprendizado são necessário**s para fazer sentido do ambiente

T. Dybå, T. Dingsøyr, N.B. Moe, "Agile Project Management," book chapter in G. Ruhe and C. Wohlin (Eds.), Software Project Management in a Changing World, Berlin: Springer Verlag, 2014, pp. 277-300.

Gestão de Projetos Ágeis

Um projeto ágil espera que as coisas irão mudar ao longo do projeto

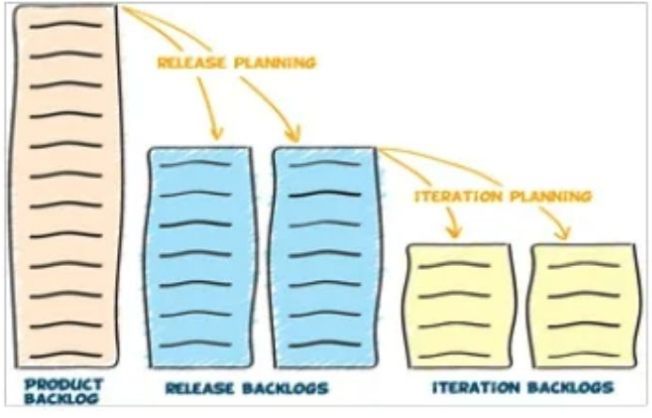
- Mudanças de requisitos
- Mudanças de design
- Mudanças de tecnologia
- Mudanças de pessoas

Por isso, requerem **outra forma de planejamento**

Seis níveis de planejamento

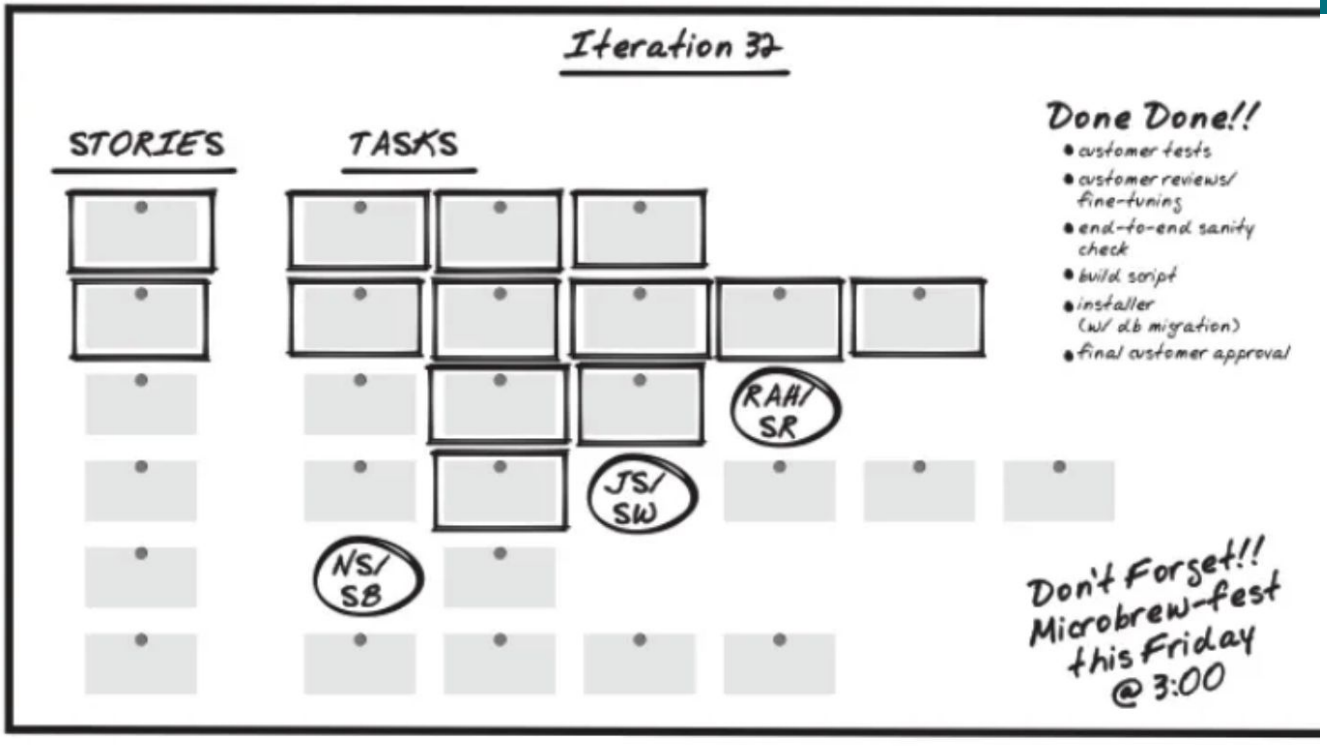
https://www.mountaingoatsoftware.com/uploads/presentations/agile-project-management.png

Planejamento



Planejamento de Release com Story Mapping

Planejamento de Iteração



4 Princípios da Gestão ágil de projetos

- O princípio da Especificação Mínima
- O princípio de Times autônomos
- O princípio de redundância
- O princípio do Feedback e Aprendizado

1. O Princípio de Especificação Mínima

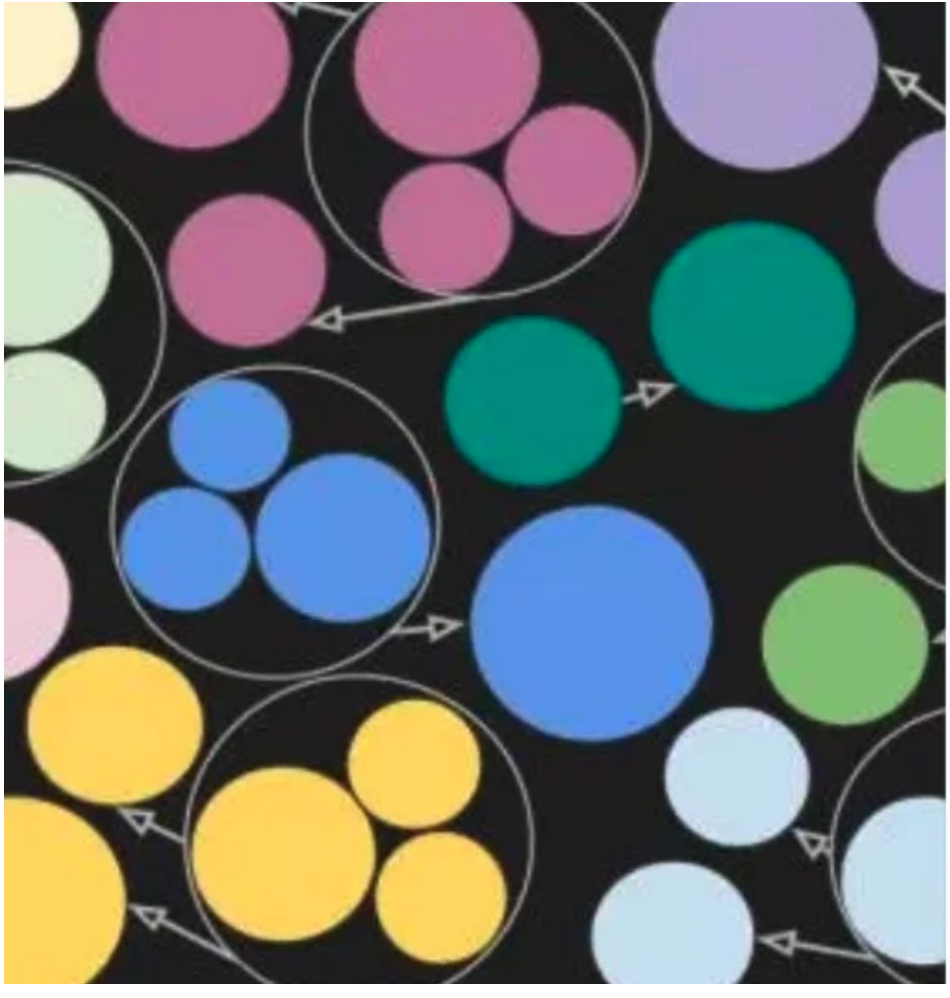
This principle is oriented towards the analysis and understanding of the nature of the overall problems:

- *Identify what is critical to overall success*
- *Specify no more than what is absolutely essential*
- *Keep the use of rules, standards, and predefined procedures to an absolute minimum*
- *Focus on the larger system requirements and boundary conditions, leaving as many design decisions as possible to those closest to the work*
- *The project's kick-off meeting is crucial*

"Plans are nothing;
Planning is everything"

Dwight D. Eisenhower

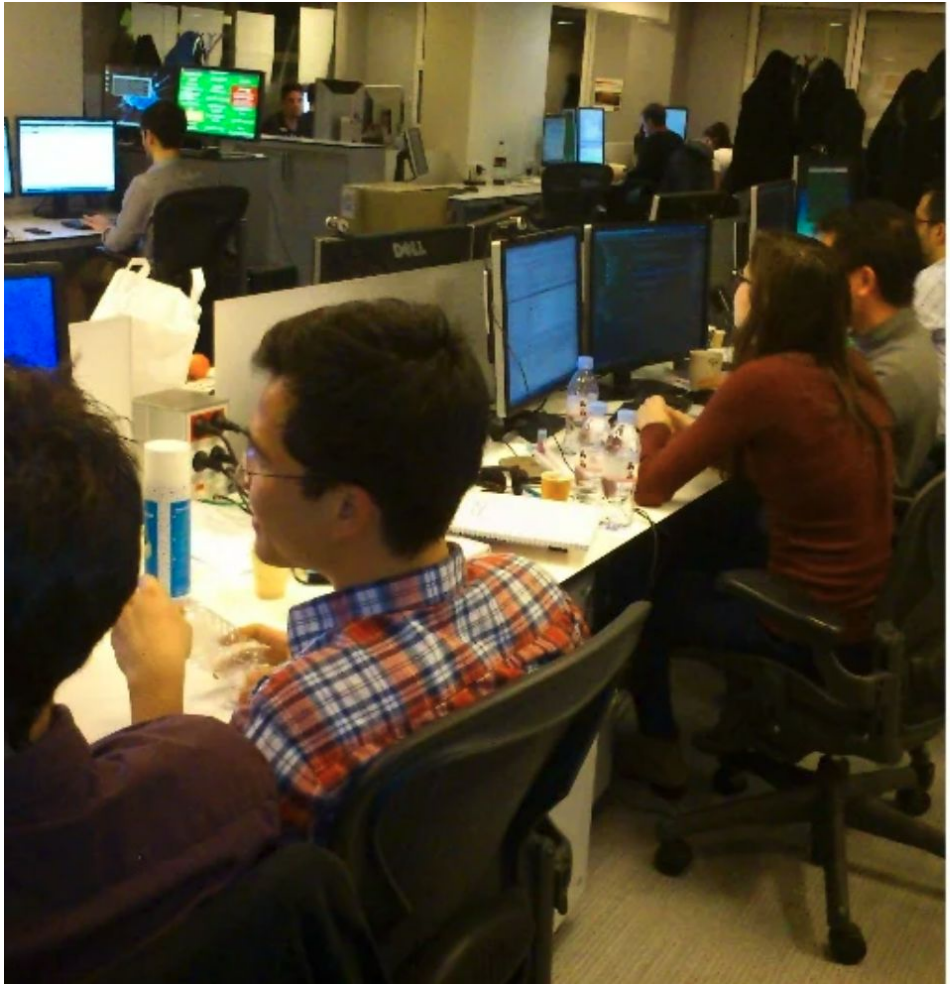
Focus on Impact! Not dates or deliverables



2. O Princípio de Times Autônomos

Autonomous, or self-managing, teams bring decision-making authority to the level of operational problems and uncertainties:

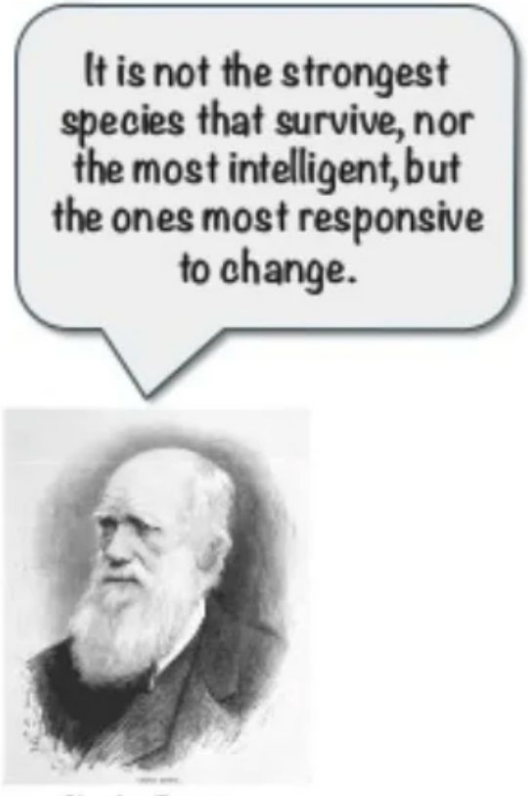
- *Members of autonomous teams are responsible for managing and monitoring their own processes and executing tasks*
- *They typically share decision authority jointly,*
- *For autonomous teams to thrive, it is necessary to build trust and commitment in the whole organization, avoiding any controls that would impair creativity and spontaneity*
- *Successfully dealing with this principle, we must understand the context surrounding the team*



3. O Princípio da Redundância

This principle is concerned with the overlap in individuals' knowledge and skills:

- *Each member of the team should be skilled in more than one function, making the project more flexible and adaptive*
- *With increased redundancy within the team, individuals will find it easier to share new knowledge*
- *Cross-trained team members increase the project's functional redundancy and thus the flexibility of the team in dealing with personnel shortages*
- *Redundancy is also critical in turbulent environments where people need to work on tasks assigned by priority rather than the competence of team members*



4. O Princípio do Feedback e Aprendizado

Without feedback and learning, agile project management is not possible:

- *The focus on project execution rather than on up-front planning, leads to an*
  - *intertwinement of learning and work, and of*
  - *problem specification and solution.*
- *The complexity and unpredictability of software problems are typical of 'wicked' problems, which are difficult to define until they are nearly solved.*
- *To deal with this principle, project activities have to be performed in an iterative and incremental way.*

Gestão de Projetos Ágeis

O **desafio** em fazer a gestão ágil de projetos de software é **encontrar o equilíbrio entre o planejamento adiantado e o aprendizado.**

Planejamento proporciona disciplina e um conjunto concreto de atividades e contingências que podem ser codificadas, executadas e monitoradas.

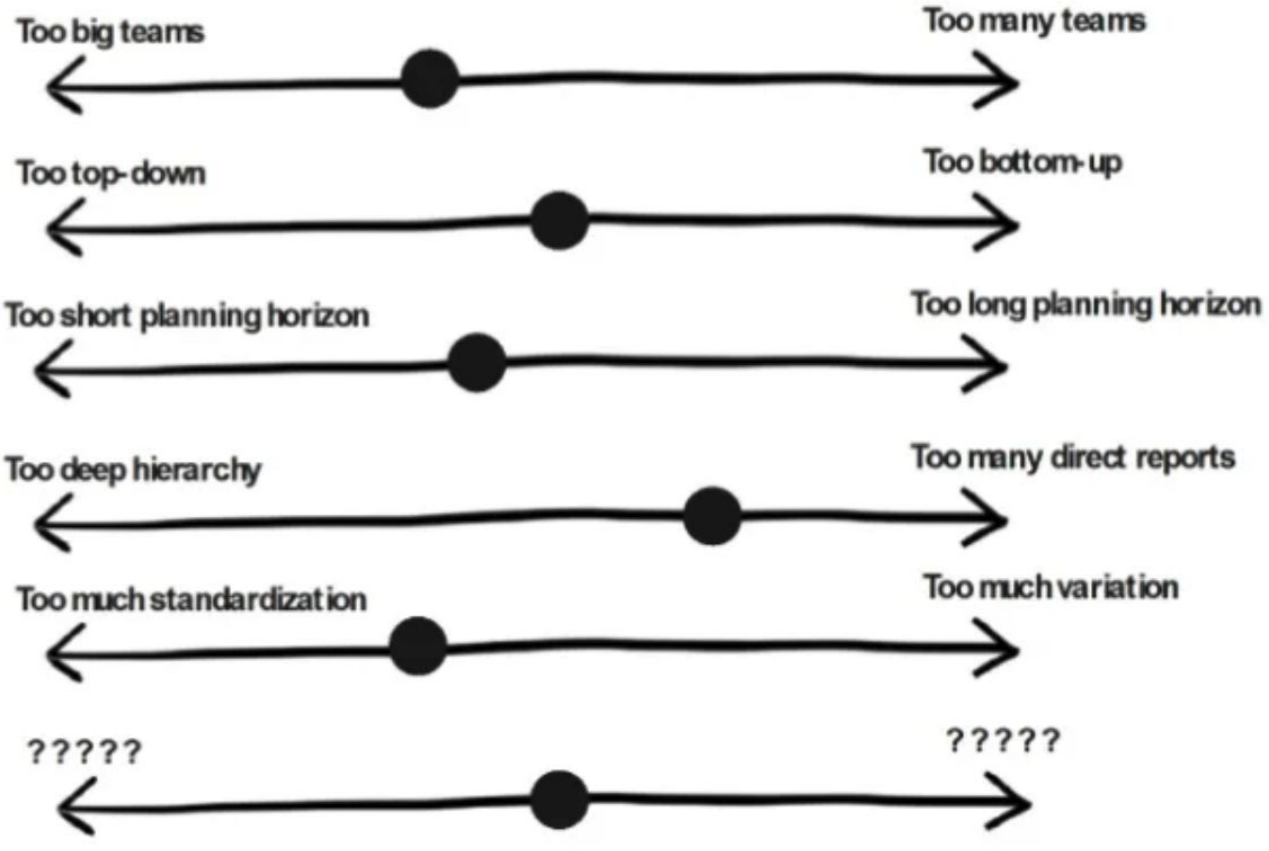
Aprendizagem permite adaptação a eventos imprevistos e caóticos.

Ambos requerem um tipo diferente de estilos de gestão e infraestrutura de projeto

Projetos com baixo grau de complexidade e incerteza permite planejamento mais detalhado, no qual **projetos com alto grau de complexidade e incerteza requer maior ênfase no aprendizado**

T. Dybå, T. Dingsøyr, N.B. Moe, "Agile Project Management," book chapter in G. Ruhe and C. Wohlin (Eds.), Software Project Management in a Changing World, Berlin: Springer Verlag, 2014, pp. 277-300.

Não tem certo ou errado. São tradeoffs



Henrik Kniberg, What is Scrum?

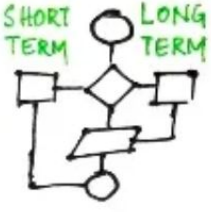
Liderança Ágil

Agilistas

QAspire.com TANMAY VORA @tnvora

NINE ROLES FOR GREAT LEADERSHIP..

ARCHITECT



PLANNER

EXECUTOR

CONDUCTOR
INSPIRE POSSIBILITIES

TEACHER

STEWARD

INNOVATOR

EXPERT

THINKER

SENIOR LEADER

OVER SIGHT
DIRECTOR

PRODUCT OWNER +TEAM

ENCOURAGE & **MONITOR**

REMOVE BIG IMPEDIMENTS

IMPLEMENT and DELIVER

SUPPORT

PLAN

-PREPARE-

CREATE THE VISION

PARTNER

Learn

DISCOVERING

SHORT TERM | MID TERM | LONG TERM

Hands On

Referências

- https://www.slideshare.net/hotpop/introduo-qualidade-e-testes-geis-de-software
- https://www.slideshare.net/hotpop/princpios-da-gesto-gil

CTEDS

Capacitação Tecnológica em Engenharia e Desenvolvimento de Software

Métodos Ágeis

Professor: Carla Rocha [UnB] *caguiar@unb.br*

15 de Agosto de 2022