

# Metodologias de desenvolvimento de SW

# Processo de desenvolvimento de SW



# Visão Geral

Modelo em Cascata

Modelos de Processo Evolutivos

- Modelo baseado em Protótipos
- Modelo em Espiral

Modelo Incremental

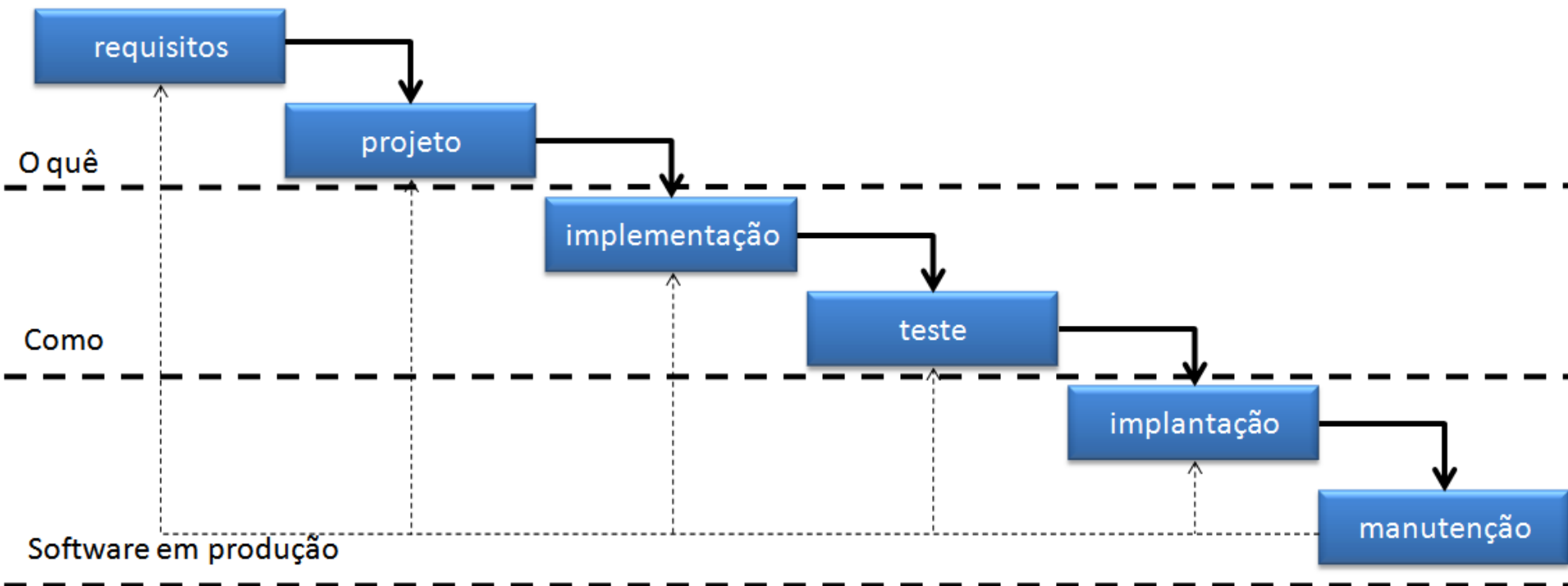
Modelo Unificado

Modelos Ágeis

# Metodologias Tradicionais

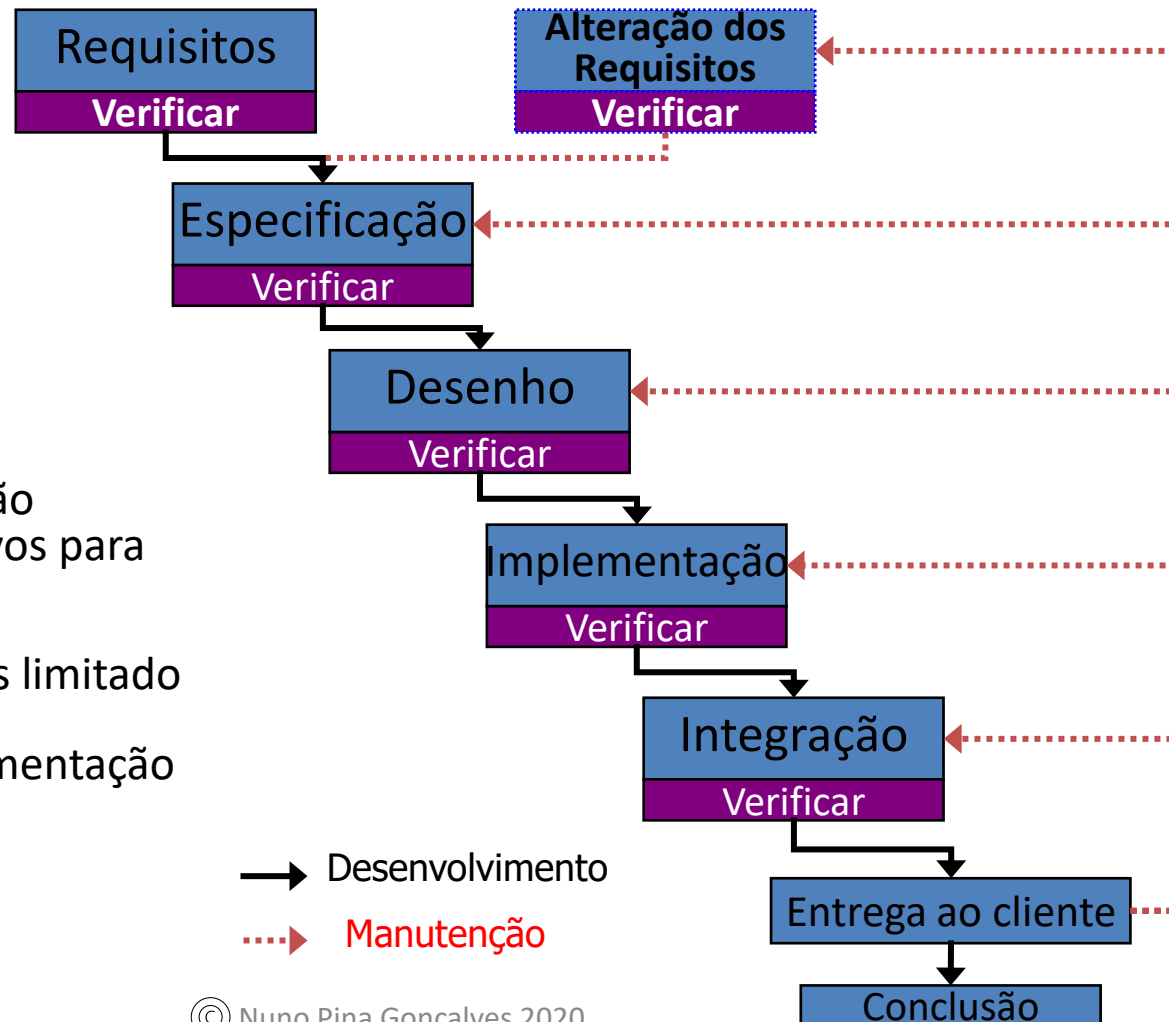
## Modelo Cascata

- Linear/Sequencial



# Metodologias Tradicionais

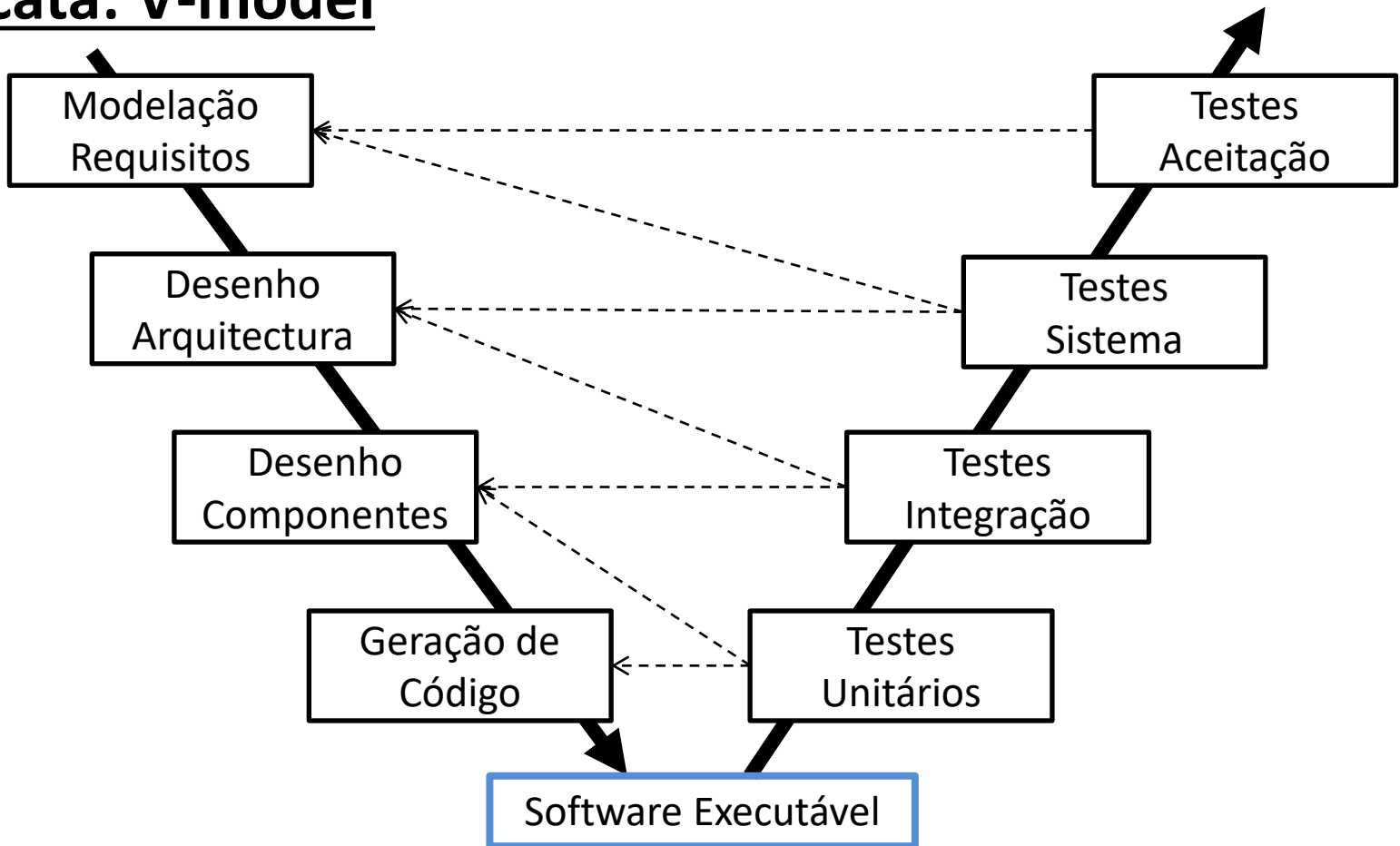
## Modelo Cascata



- Cada fase é visitada sequencialmente
- No fim de cada fase são verificados os objectivos para avaliar a progressão
- Retrocesso entre fases limitado
- Cada fase inclui documentação

# Metodologias Tradicionais

## Cascata: V-model



# Metodologias Tradicionais

## Modelos evolutivos: Prototipagem



- Adequado quando o cliente tem uma necessidade mas não consegue especificar os detalhes
- A construção de protótipos é enquadrada na fase de requisitos
- A especificação suporta-se no feedback relativo ao protótipo

# Metodologias Tradicionais

## Baixa Fidelidade

A hand-drawn sketch of a web interface titled "My Contact List". In the top right corner, it says "User: John Doe". Below the title is a section labeled "Edit Contact". This section contains four input fields: "Name:", "Phone:", "Email:", and "Notes:". At the bottom right of the "Edit Contact" section are two buttons labeled "Save" and "Cancel". The entire sketch is enclosed in a simple rectangular border.

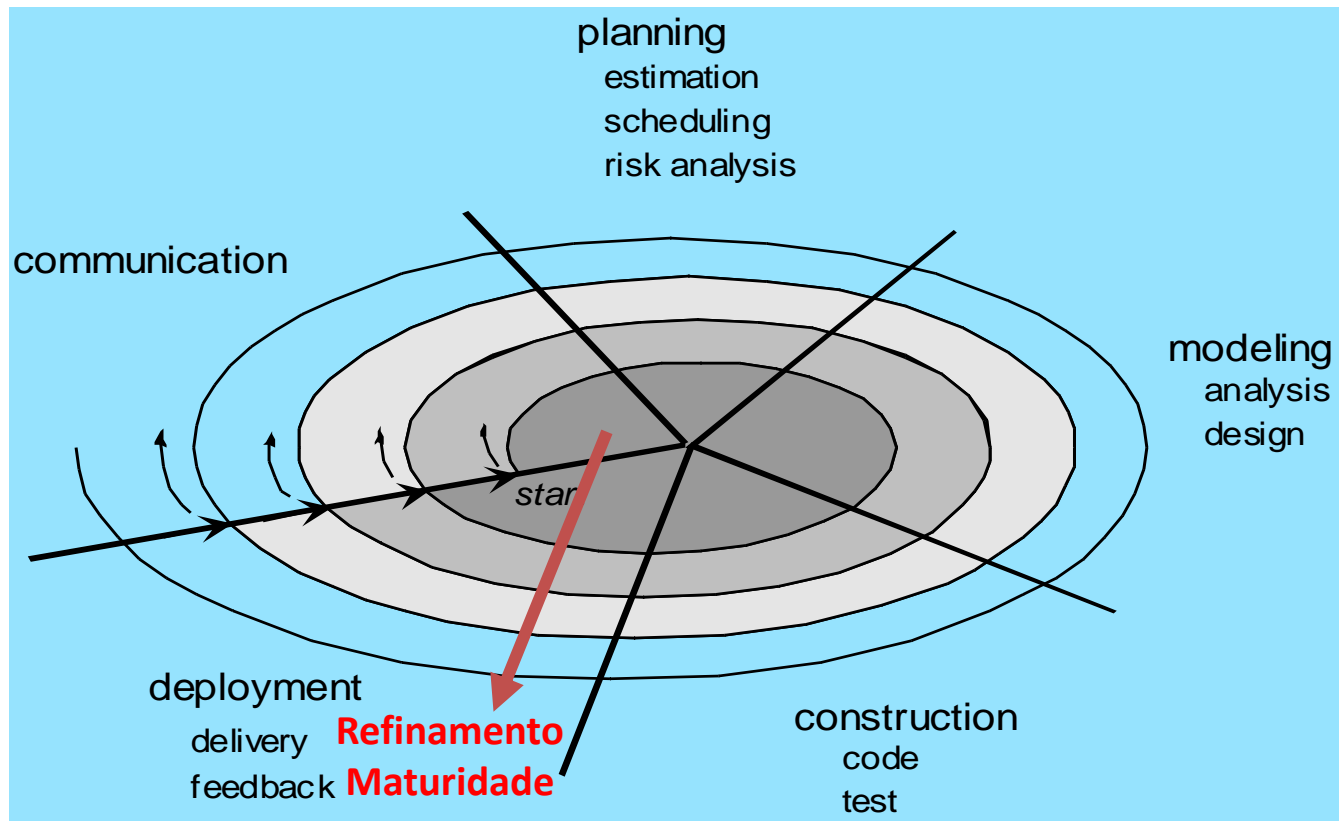
## Alta Fidelidade

A pixelated mockup of a web browser window. The browser's title bar says "Horochovec Blog". The address bar contains the URL "http://www.horochovec.com.br/". The main content area displays a registration form titled "Cadastro". Below the title is the instruction "Informe os dados abaixo para seu cadastro". The form includes four input fields: "Nome:", "Email:", "Senha:", and "Senha:". At the bottom of the form are three buttons: "Cadastrar", "Limpar", and "Efetuar Login". The browser window has standard navigation buttons (back, forward, stop, home) and a search icon.



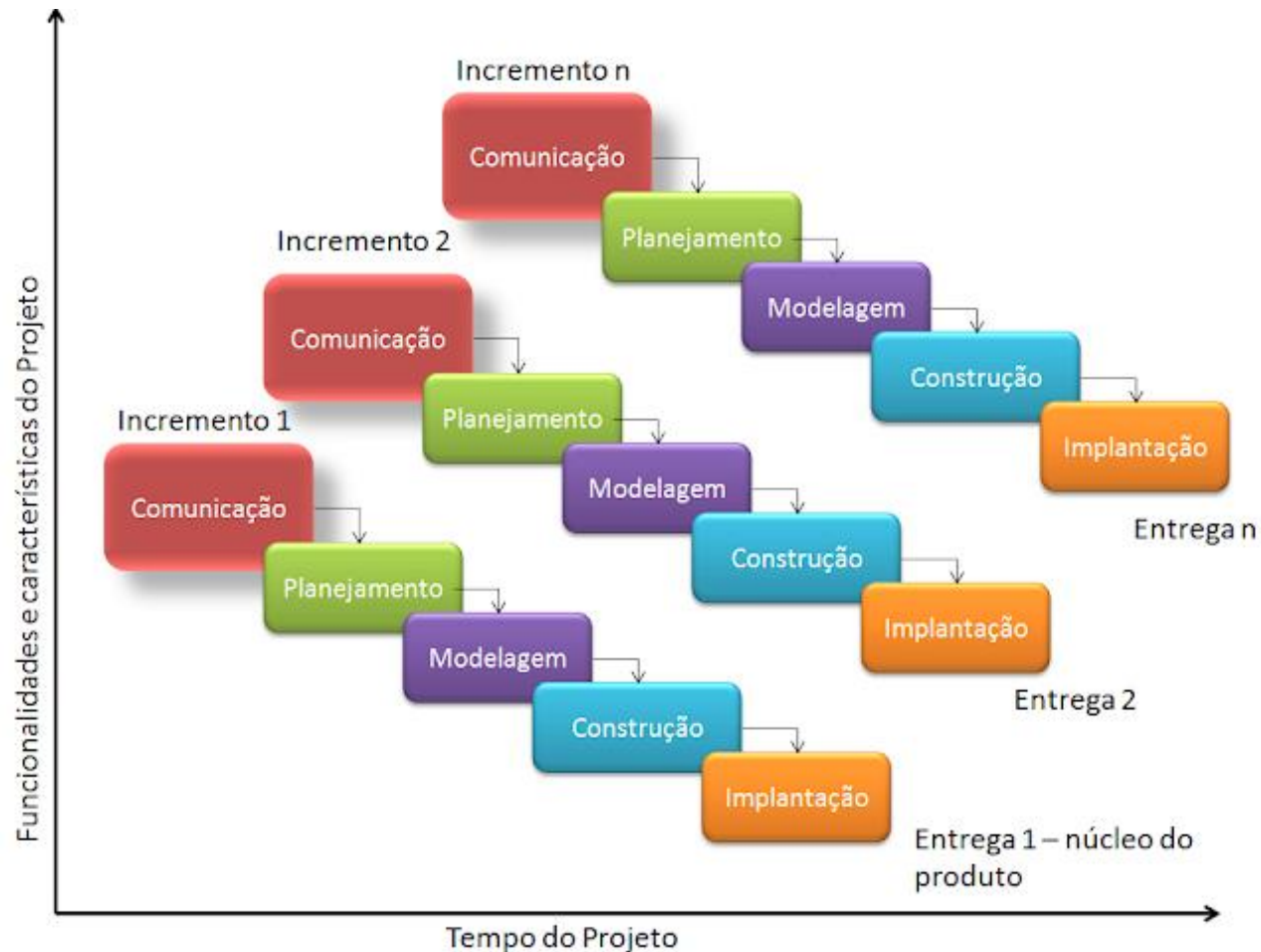
# Metodologias Tradicionais

## Modelos evolutivos: Espiral



# Metodologias Tradicionais

## Modelo incremental

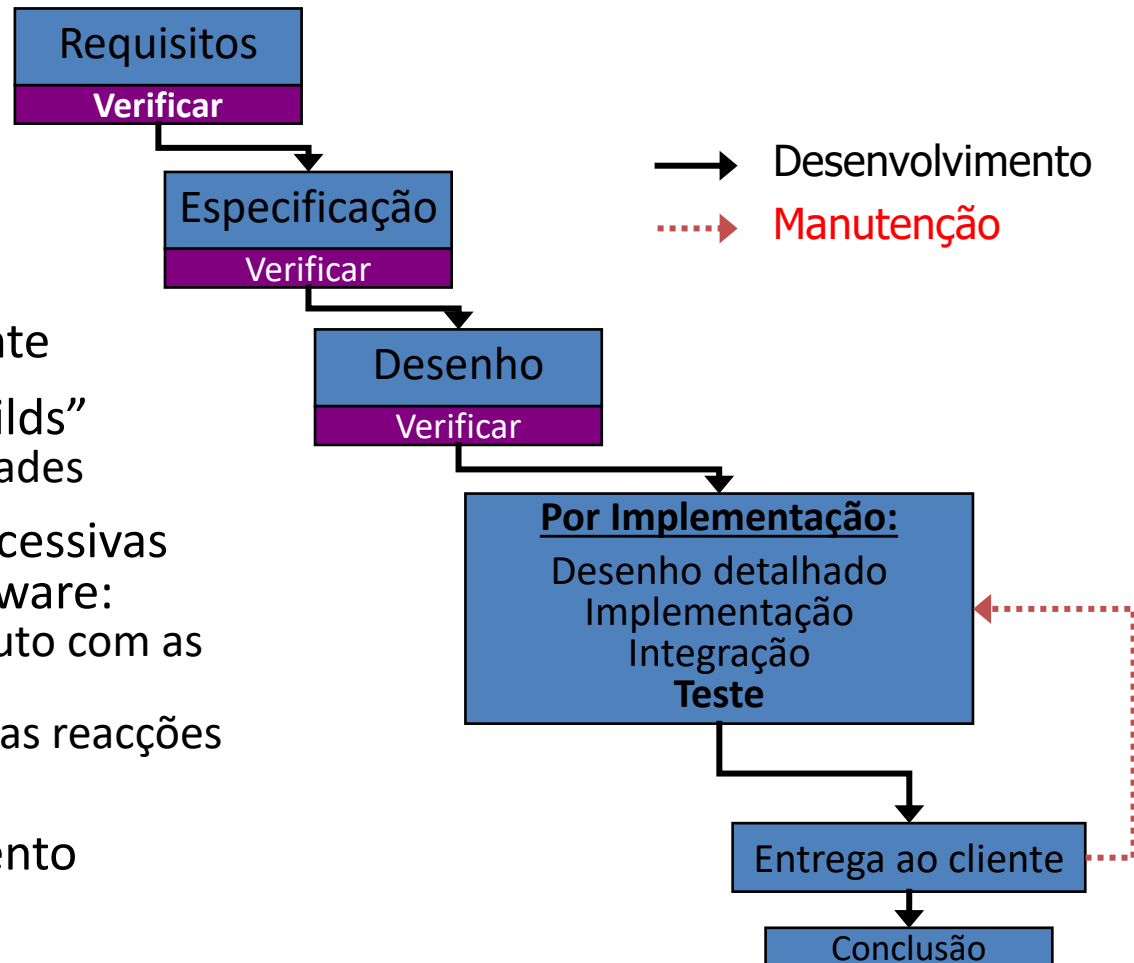


# Metodologias Tradicionais

## Modelo incremental

### Modelo Incremental

- Entrega progressiva ao cliente
- Divisão do projecto em “builds”
  - cada adiciona funcionalidades
- Do desenvolvimento das sucessivas versões entregáveis do software:
  - o 1º incremento é o produto com as funcionalidades de base
  - novas versões baseadas nas reacções dos utilizadores
- Capitalização no conhecimento adquirido anteriormente



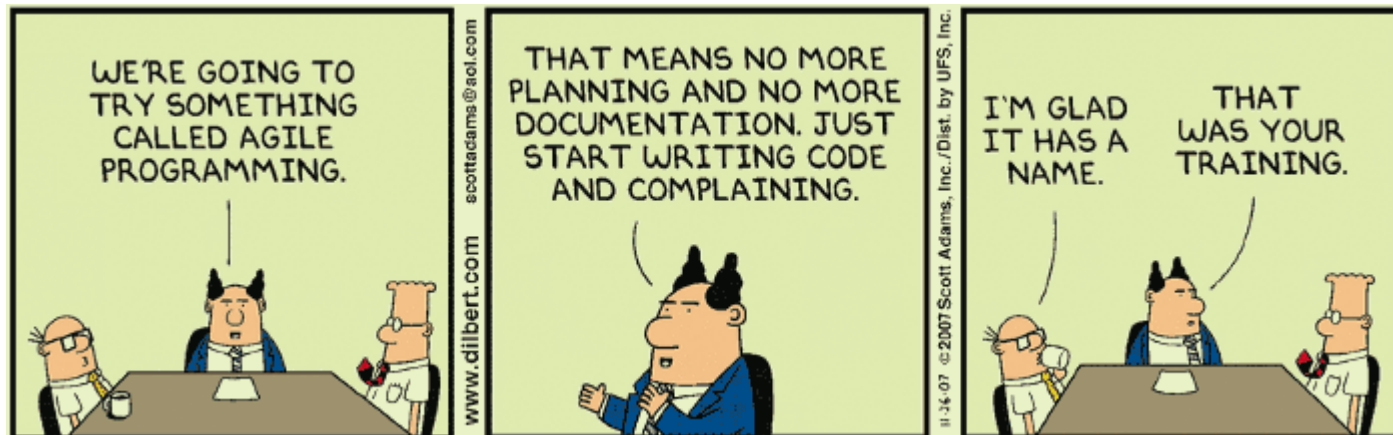
# Metodologias Tradicionais

Modelo	Pontos Fortes	Pontos Fracos
<b>Cascata</b>	<ul style="list-style-type: none"><li>- Segurança na passagem entre fases</li><li>- Linear</li><li>- Controlo</li><li>- Documentação</li></ul>	<ul style="list-style-type: none"><li>- Rígido</li><li>- Risco de insatisfação final</li></ul>
<b>Prototipagem</b>	<ul style="list-style-type: none"><li>- Permite ao cliente visualizar a ideia relativa ao produto final</li><li>- Revisitasse com facilidade fases anteriores</li><li>- Assegura que os requisitos do cliente são satisfeitos: Construtivista</li></ul>	<ul style="list-style-type: none"><li>- Pode ser complicado definir os protótipos</li><li>- Custo e tempo associados</li></ul>
<b>Espiral</b>	<ul style="list-style-type: none"><li>- Ênfase na análise de riscos</li><li>- Incorpora actividades dos restantes modelos</li><li>- Construtivista</li></ul>	<ul style="list-style-type: none"><li>- Adequado para projectos maiores (e longos!)</li><li>- ... e com equipas experientes</li><li>- Complexo de usar</li></ul>
<b>Incremental</b>	<ul style="list-style-type: none"><li>- Vai de encontro as necessidades do cliente progressivamente</li><li>- Ciclos + rápidos com entregas operacionais</li><li>- Divide complexidade (Modular/Escalavel)</li></ul>	<ul style="list-style-type: none"><li>- Varias versões (controlo!!)</li><li>- Obriga a uma formação continua dos utilizadores</li><li>- Se mal usada pode degenerar em “construir e modificar”</li></ul>

# Metodologias Ágeis



# Metodologias Ágeis



# Metodologias Ágeis



# Metodologias Ágeis

## Métodos Preditivos vs Adaptativos

- Os **métodos tradicionais** focam-se no planeamento das actividades futuras. A equipa deve ser capaz de reportar exactamente quais as tarefas que estão planeadas para todo o processo de desenvolvimento  
» **Métodos Preditivos**
- Os **métodos ágeis** focam-se na resposta à mudança. A equipa não consegue reportar exactamente quais as tarefas previstas para daqui a e.g. 6 meses, mas apenas o objectivo da versão a desenvolver e o custo/esforço estimado da mesma  
» **Métodos Adaptativos**



# Metodologias Ágeis



# Metodologias Ágeis

**Indivíduos  
e interações**

mais que  
processos e  
ferramentas

1

2

**Software em  
funcionamento**

mais que  
documentação  
abrangente

**Responder  
a mudanças**

mais que  
seguir um  
plano

4

3

**Colaboração  
com o cliente**

mais que  
negociação  
de contratos

# Metodologias Ágeis



Providenciam uma estrutura conceptual para reger projetos de engenharia de software, baseada no seguinte conjunto de princípios:

- Valorização dos indivíduos e das suas interações em detrimento da valorização dos processos e ferramentas
- Valorização do desenvolvimento de software funcional ao invés de um maior ênfase no desenvolvimento de modelos e documentação
- Valorização da colaboração com clientes ao invés de especificação rígida e altamente formal
- Valorização da capacidade de adaptação à mudanças em detrimento da capacidade de seguir um plano.

# Metodologias Ágeis

## Considerações Gerais

- As metodologias ágeis tendem ser adoptadas de forma mais eficiente com equipas de pequena/média
  - e.g. até 10 elementos
- Surgem no entanto algumas dificuldades para:
  - Equipas com membros distribuídos geograficamente
  - Quando estamos perante culturas organizacionais do tipo: *comando e controlo*.

# Metodologias Ágeis

## Processo Ágil

- Contextos em que reconhecidamente os planos têm uma “vida-curta”
  - resposta efectiva à medida que as alterações ocorrem
- Efectiva comunicação com todos os intervenientes (*stakeholders*)
- É orientado por descrições do cliente sobre o que é pretendido (cenários)
  - o cliente faz parte da equipa
- O software é desenvolvido iterativamente com ênfase nas actividades de construção
  - entrega de múltiplos “incrementos de software”
- Organização muito própria das equipas de forma a controlar efectivamente o trabalho

*resumindo ...*

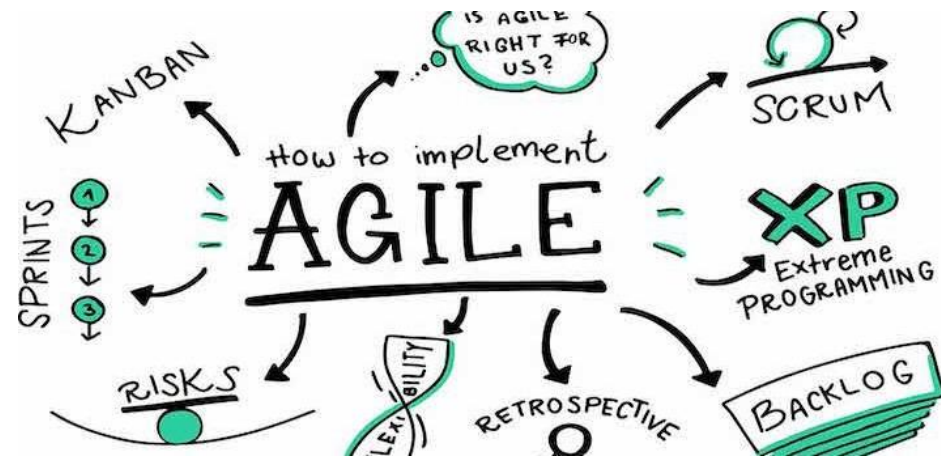
**Entrega rápida e incremental de software/componentes operacionais,  
acomodando a adaptação entre entregas**



# Metodologias Ágeis

## Alguns exemplos

- **SCRUM**
- **KANBAN**
- **XP – extreme Programming**
- DSDM -dynamic systems development method
- Adaptive Software Development
- Crystal Clear
- Feature-Driven Development
- Pragmatic Programming



# Metodologias Ágeis

## SCRUM



### Origem de SCRUM

- O Scrum foi criado por Jeff Sutherland e Ken Schwaber em 1995, e apresentado na conferência Ospsla em Austin no Texas. Neste mesmo ano foi publicado o artigo “SCRUM Software Development Process”.
- Os autores herdaram o termo “Scrum” do artigo “The New Product Development Game”, publicado por Takeuchi e Nonaka em 1986. Neste artigo eles utilizam o termo “Scrum” obtido do jogo de Rugby, onde fazem uma analogia entre o processo de desenvolvimento de um produto com as táticas de jogo de Rugby, que valorizam o trabalho em equipe.

# Metodologias Ágeis

## Origem de SCRUM

Em Rugby, SCRUM é composta por uma equipa de oito elementos que trabalham em conjunto para levar a bola adiante no campo.



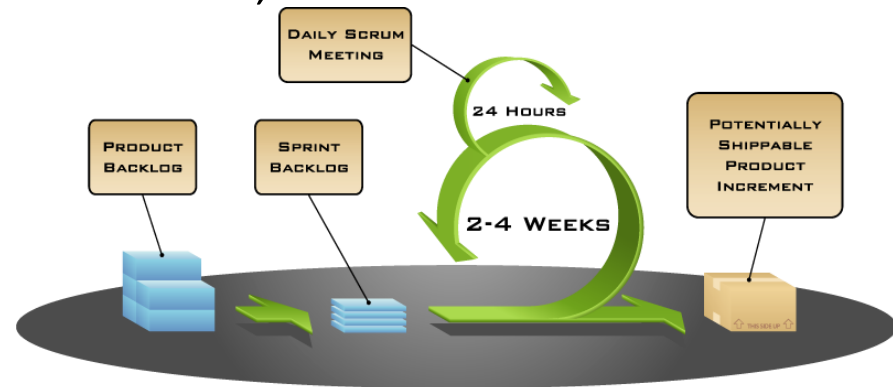
Uma equipa que trabalha como uma unidade altamente integrada com cada membro desempenhando um papel bem definido e toda a equipa focando-se num único objectivo comum.



# Metodologias Ágeis

## SCRUM - Definições

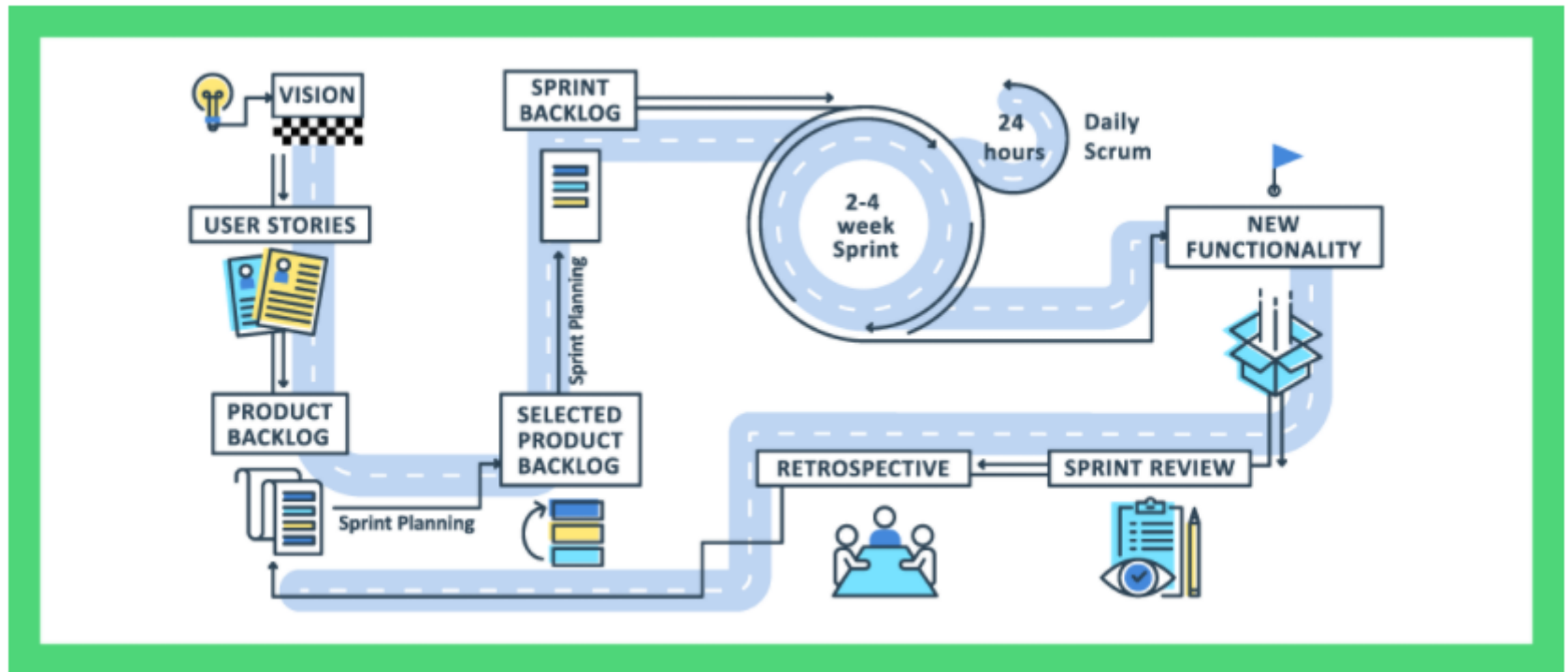
- SCRUM define um processo para construir software incrementalmente em contextos complexos, onde os requisitos não são claros ou mudam com muita frequência.
- SCRUM é um *framework* ágil de gestão de projetos usado para entregar aos clientes, de forma iterativa, incrementos de produto de alto valor.



COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE

# Metodologias Ágeis

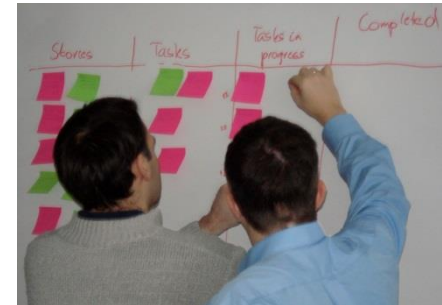
## Processo SCRUM



# Metodologias Ágeis

## Processo SCRUM

- O trabalho a realizar é registado no **Product Backlog**, que é uma lista de tarefas (incluindo associadas à implementação dos requisitos/funcionalidades) ou de user stories
- O projeto desenvolve-se numa série de iterações, chamadas incrementos (**sprints**)
- No início de cada incremento é feita uma Reunião de Planeamento de Incremento (**Sprint Planning Meeting**) na qual o Dono do Produto (**Product Owner**) define as prioridades do *Product Backlog*
- A Equipa (**Scrum Team**) selecciona as tarefas que completará durante o próximo *Incremento*
- Essas tarefas são então transferidas do *Product Backlog* para o **Sprint Backlog**



# Metodologias Ágeis

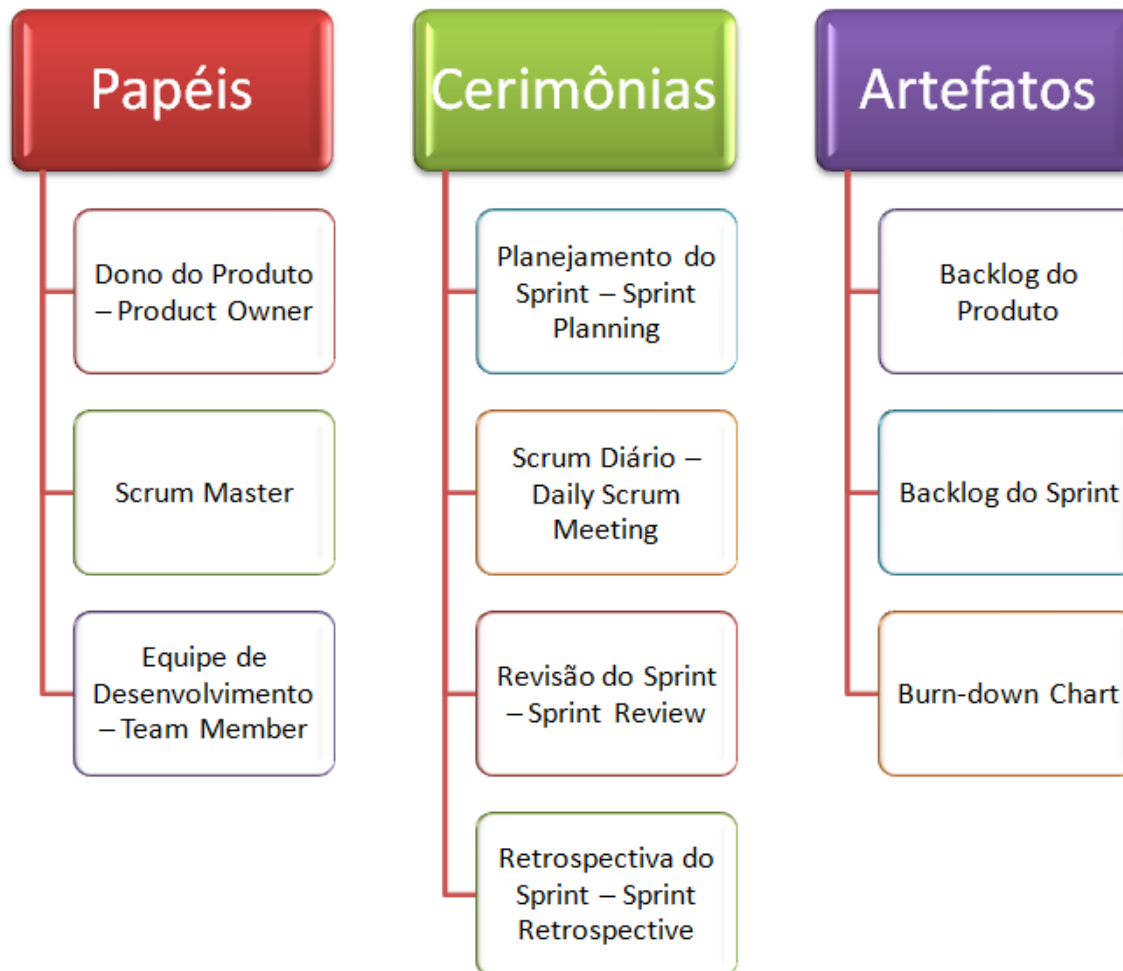
## Processo SCRUM

- Durante o incremento, são conduzidas curtas reuniões diárias chamadas de Scrum Diário (**Daily Scrum**), que ajudam a equipa a sincronizar-se e manter-se coesa e direccionada
- Ao final de cada incremento a equipa demonstra a funcionalidade concluída, na **Sprint Review Meeting**

Tipicamente:

- O desenvolvimento é dividido em Sprints de 30 dias
- Equipas são pequenas
- O *Daily Scrum* segue o formato de uma reunião de 15 minutos onde a equipa reporta sobre o que fez e expõe o que será feito no próximo dia e identifica os potenciais factores de impedimento ao progresso

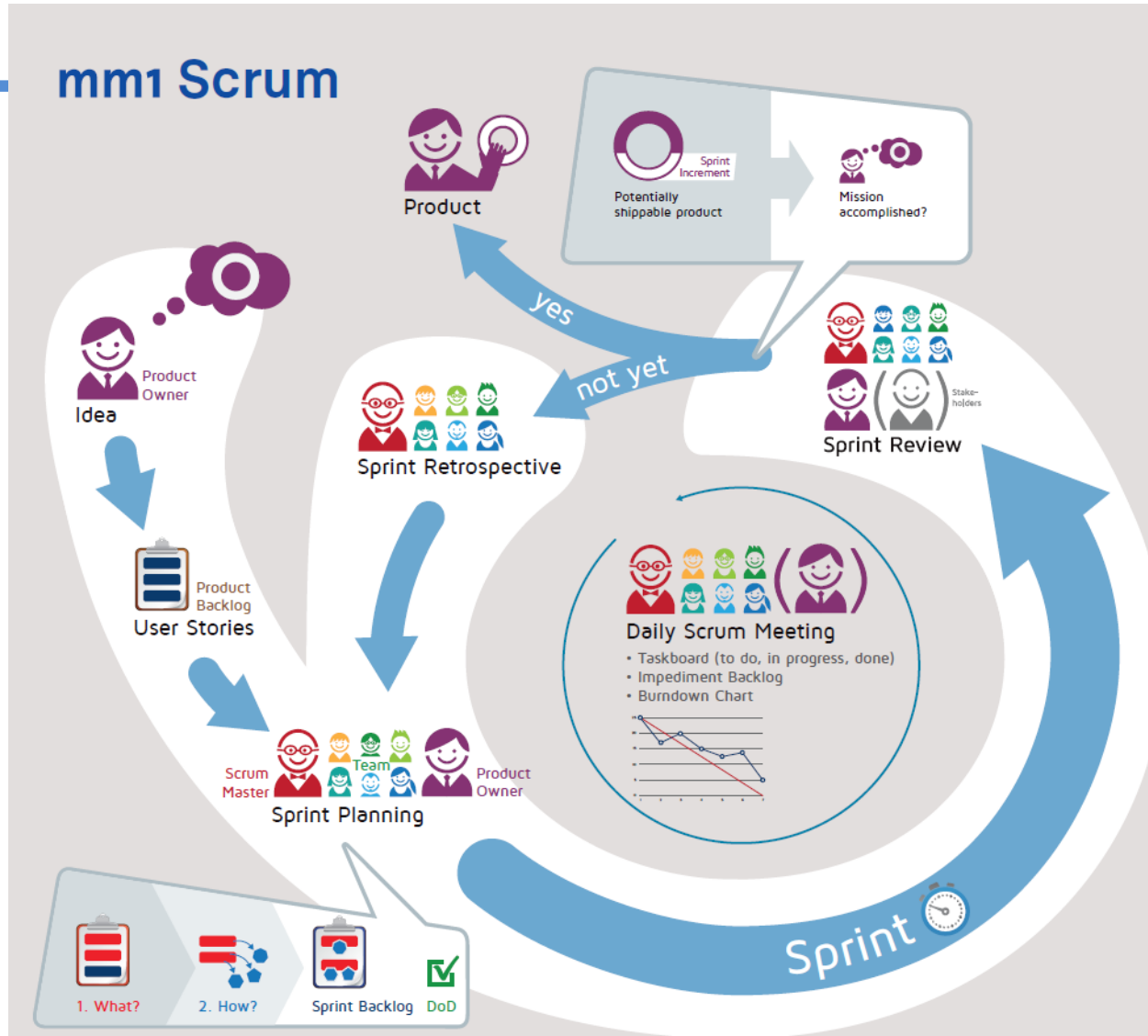
# Metodologias Ágeis



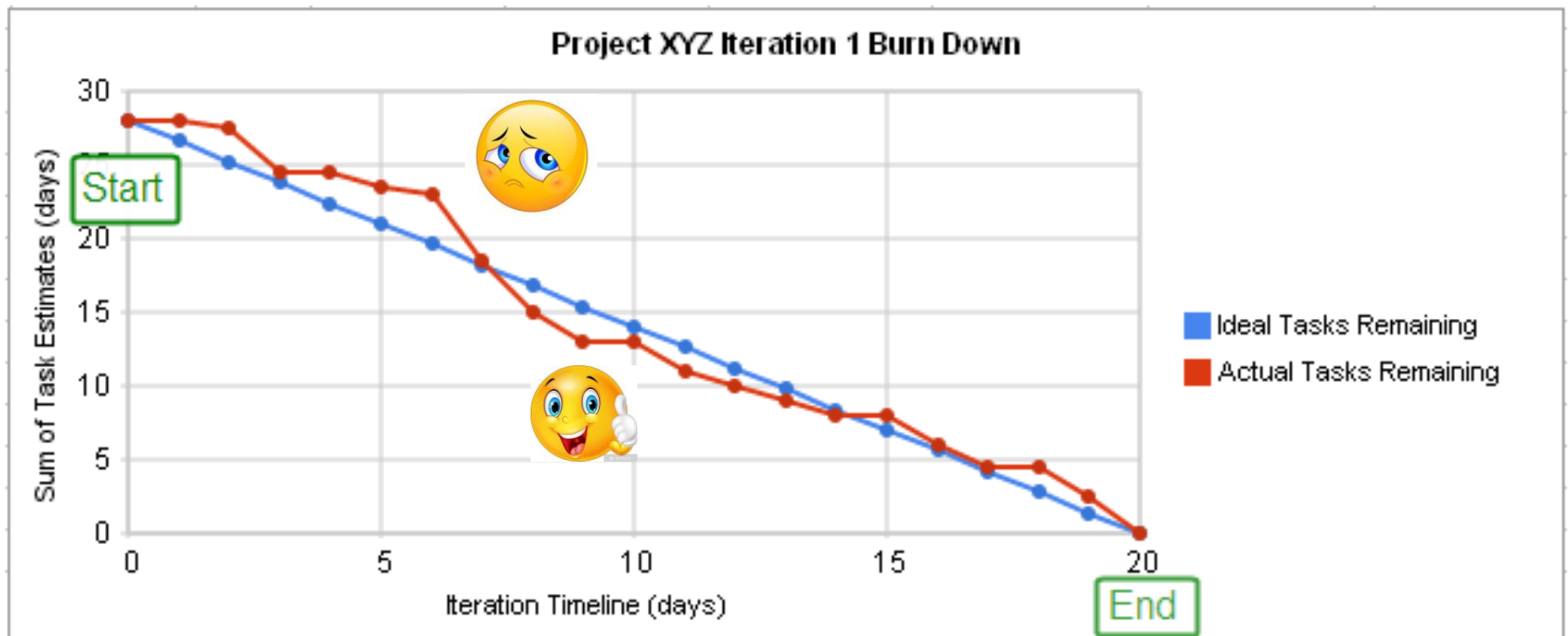
# Eventos SCRUM

Event	Inspection	Adaptation
Sprint Planning	<ul style="list-style-type: none"><li>• Product Backlog</li><li>• (Commitments Retrospective)</li><li>• (Definition of Done)</li></ul>	<ul style="list-style-type: none"><li>• Sprint Goal</li><li>• Forecast</li><li>• Sprint Backlog</li></ul>
Daily Scrum	<ul style="list-style-type: none"><li>• Progress toward Sprint Goal</li></ul>	<ul style="list-style-type: none"><li>• Sprint Backlog</li><li>• Daily Plan</li></ul>
Sprint Review	<ul style="list-style-type: none"><li>• Product Increment</li><li>• Product Backlog (Release)</li><li>• Market-business conditions</li></ul>	<ul style="list-style-type: none"><li>• Product Backlog</li></ul>
Sprint Retrospective	<ul style="list-style-type: none"><li>• Team &amp; collaboration</li><li>• Technology &amp; engineering</li><li>• Definition of Done</li></ul>	<ul style="list-style-type: none"><li>• Actionable improvements</li></ul>

# SCRUM



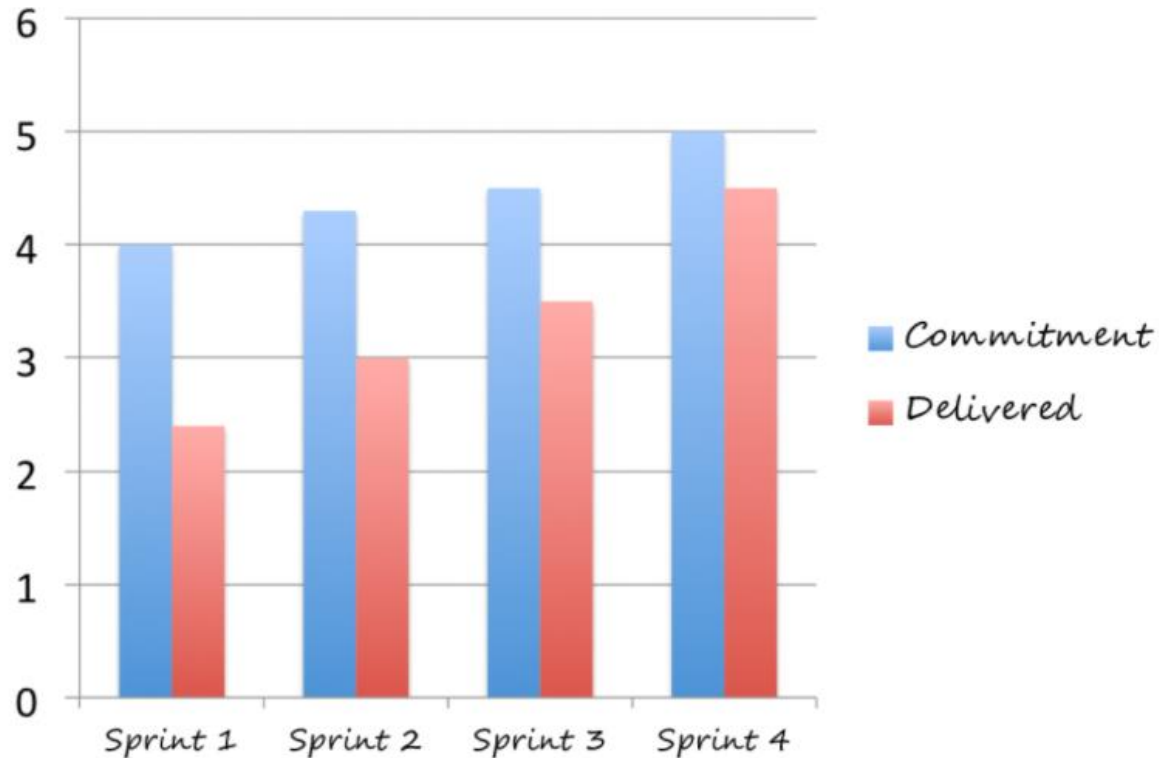
# Burndown Chart





# Velocity Chart

## VELOCITY CHART



# Outros reports

Jira Software

O seu trabalho

Projetos

Filtros

Painéis

People

Aplicações

Criar

Search

?

ING

MADS

Projeto de software clássico

Voltar para o projeto

Relatórios

Todos os relatórios

Ágil

quadro MADS

Painel

Gráfico Burndown

Gráfico de burnup

Relatório de Sprint

Gráfico de velocidade

Diagrama de fluxo cumulativo

Relatório de versão

Relatório de epic

Controlar Gráfico

Burndown de epic

Burndown de versão

ANÁLISE DA PROBLEMAS

Relatório de média de idade

Relatório de problemas cria...

Relatório de gráfico circular

Projetos / MADS

Todos os relatórios

Ágil

**Gráfico Burndown**

Acompanhe o trabalho restante total e projete a probabilidade de alcançar o objetivo da sprint. Isso ajuda seu time a gerenciar seu progresso e responder adequadamente.

**Gráfico de burnup**

Acompanhe o trabalho concluído independentemente do trabalho total realizado. Isso ajuda sua equipe a gerenciar seu progresso e entender melhor o efeito das mudanças de escopo.

**Relatório de Sprint**

Acompanhe o trabalho concluído ou colocado de volta no backlog em cada sprint. Isso ajuda a determinar se sua equipe está sobrecarregada ou se houve um aumento excessivo de escopo.

**Gráfico de velocidade**

Acompanhe a quantidade de trabalho concluída sprint a sprint. Isso ajuda a determinar a velocidade do time e estimar o trabalho que o time pode realisticamente realizar em sprints futuros.

**Diagrama de fluxo cumulativo**

Mostra os estados dos itens ao longo do tempo. Isso ajuda você a identificar potenciais gargalos que precisam ser investigados.

**Relatório versão**

Acompanhe a data de release prevista para uma versão. Isso ajuda você a monitorar se a versão será lançada no prazo, para que você possa tomar medidas se o trabalho estiver ficando atrasado.

**Relatório épico**

Entenda o progresso para completar um épico ao longo do tempo. Isso ajuda você a gerenciar o progresso do seu time, acompanhando o trabalho restante incompleto/não estimado.

**Controlar Gráfico**

Mostra o tempo de ciclo para o seu produto, a versão vs sprint. Isso ajuda você a identificar se os dados do processo atual podem ser usado para determinar o desempenho futuro.

**Burndown Épico**

Acompanhe o número projetado de sprints necessários para completar o épico (otimizado para Scrum). Isso ajuda você a monitorar se o épico será lançado no prazo, assim, você possa tomar medidas se o trabalho está ficando atrasado.

**Release Burndown**

Acompanhe a data de release prevista para uma versão (otimizado para Scrum). Isso ajuda você a monitorar se a versão será lançada no prazo, assim, você pode tomar medidas se o trabalho está ficando atrasado.

# Metodologias Ágeis

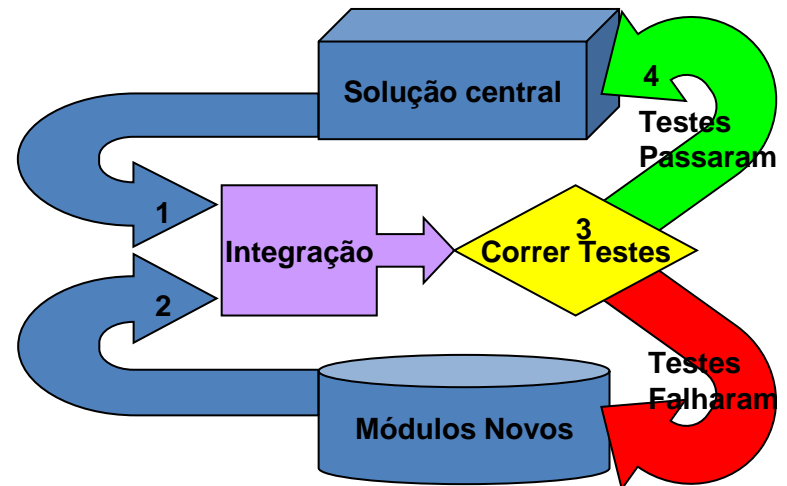
## eXtreme Programming - XP

- XP agrega uma colecção de práticas do desenvolvimento de software
- As práticas recomendadas pela XP não são novas individualmente
  - A contribuição emerge na conjugação e no ênfase extremo que se coloca na sua utilização
- Objectiva software de alta qualidade
  - Redução de bugs
  - Adaptável, modular e escalável

# Metodologias Ágeis

## XP – Processo

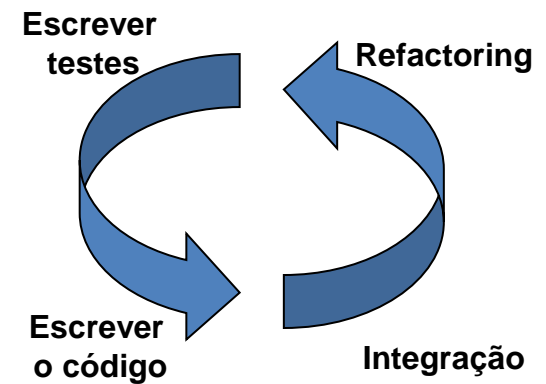
- Integração contínua dos incrementos
  - O código escrito é integrado na solução completa (processo progressivo)
  - A solução completa é compilada (minimiza as falhas por dificuldade de integração dos módulos)
  - Todos os testes são executados
  - Mantém uma versão testável
- Reuniões em pé (stand up meetings: SUM)
  - discussão do progresso do dia anterior e do plano de actividade para o dia (máximo ½ hora)
- Ênfase no combate ao erro (*abugging*)
  - Passo sustentável (tempo de trabalho fixo)
    - O cansaço causa erros: menos horas; maior intensidade
    - A concentração intensa aumenta a qualidade
  - Desenho simples
    - não incluir no código nada que não seja essencial.
  - Seguir standards de codificação
- “Releases curtas”
  - para obter feedback do cliente.



# Metodologias Ágeis

## XP – Principais Práticas

- **Spiking**
  - investigar e experimentar tecnologias necessárias à solução
  - O aprofundar de conhecimentos deve ser transmitido/partilhado
- **Desenvolvimento orientado por testes (TDD)**
  - é a prática de escrever testes unitários antes de escrever o código
  - evitar os enviesamentos dos testes realizados *à posteriori*
  - antecipar/evitar esforço em processo de *debug*
  - recurso a ferramentas (e.g. *NUnit*: <http://www.NUnit.org>)
- **Refactoring**
  - é o processo de alterar/optimizar a estrutura do código (e.g. remoção de código desnecessário)
  - atenção aos critérios para encetar o processo de refactoring
- **Pair Programming**
  - proporciona uma revisão contínua do código, ajuda a disseminar conhecimento e favorece a comunicação



# Metodologias Ágeis

## Programação em Pares

- Prevê 2 papéis: o programador e o colega.
- O programador declara qual a função que pretende programar.
- Quanto o programador acaba, declara-o.
- O colega classifica
  - e.g. dá uma pontuação de 0 a 10
  - por cada ponto abaixo de 10 o colega tem de explicar como obter o 10
- Trocam de papéis e o processo recomeça



# Metodologias Ágeis

## Programação em Pares

- Recomendações para o colega:
  - Assegurar-se que o feedback é positivo
  - Não fazer comentários pessoais, manter o foco no programa e não no programador
- Recomendações para o programador:
  - Não esquecer que os comentários se destinam a melhorar a qualidade do código
  - Manter uma atitude de gratidão pela ajuda prestada pelo colega



# Metodologias Ágeis

## KANBAN





# Metodologias Ágeis

## KANBAN

Kanban (“KAHN-BAHN”) – Japanese word meaning “signboard or billboard” – a scheduling system to ensure that only what is needed is produced

- O Kanban foi criado por criado Taiichi Ohno como o sistema de planeamento de produção da Toyota (TPS).
- Para comunicar em tempo real os níveis de capacidade, os trabalhadores passavam um cartão, ou “Kanban”, entre as equipas ou aos fornecedores
- Quando uma caixa de material terminava na linha de produção, um Kanban era passado para o armazém, descrevendo as características e quantidade do material em falta.

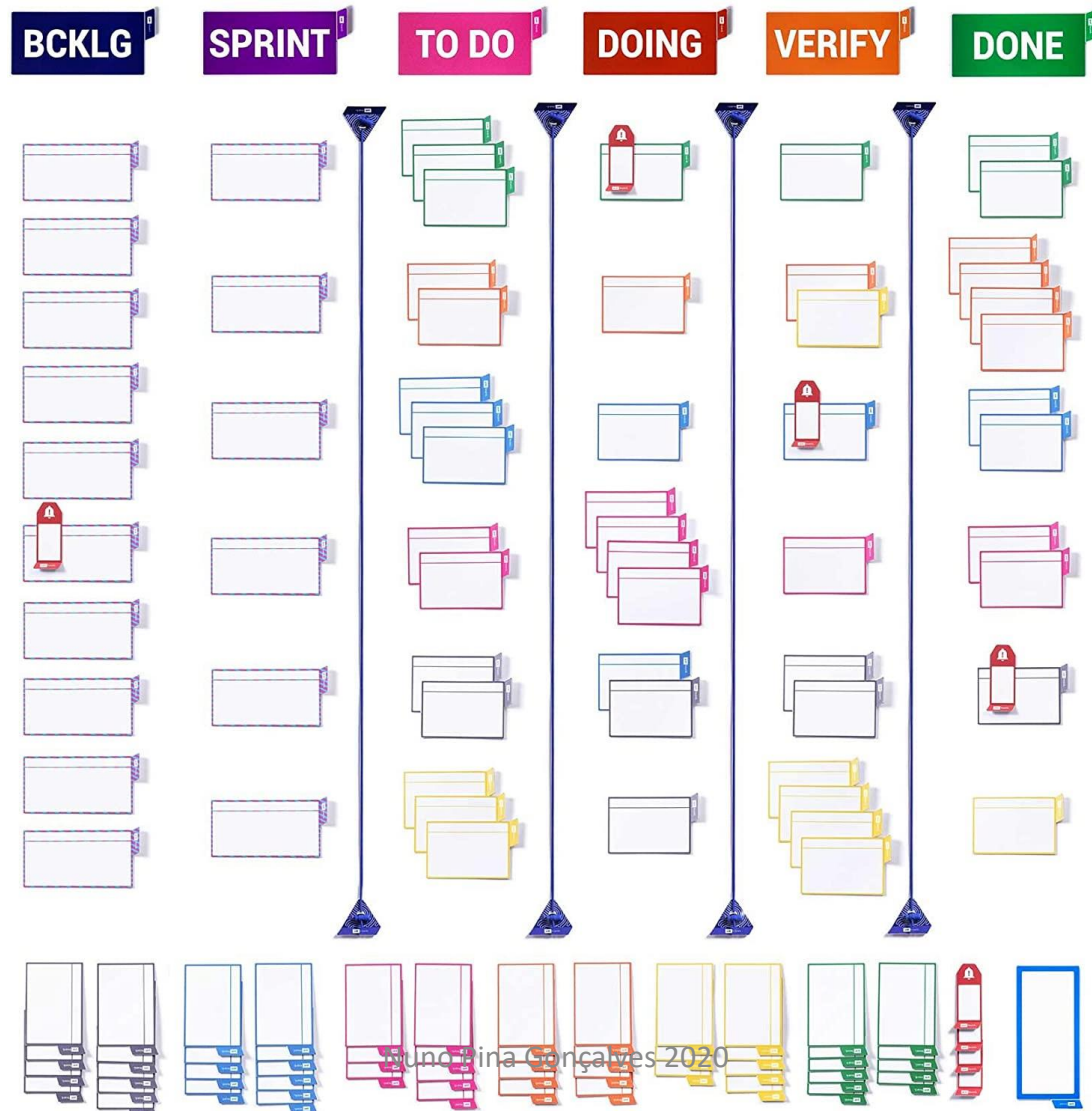
# Metodologias Ágeis

## KANBAN

- O armazém deveria de imediato enviar uma nova caixa cheia de material para a linha de produção e enviar um Kanban ao fornecedor a notificar a necessidade.
- O fornecedor, por sua vez, responderia de imediato enviando uma caixa desse material para o armazém da fábrica da Toyota. Este mesmo processo ainda está no centro do processo de fabricação “just in time”.
- No início do século XXI, o Kanban deixou de estar restrito à indústria automóvel para ser aplicado com sucesso a outros sectores complexos, como o IT, desenvolvimento de software e o marketing.
- O que hoje reconhecemos como o Método Kanban, com todos os seus principais elementos, emergiu com David J. Anderson, o primeiro a aplicar o conceito ao trabalho de IT e desenvolvimento de software.

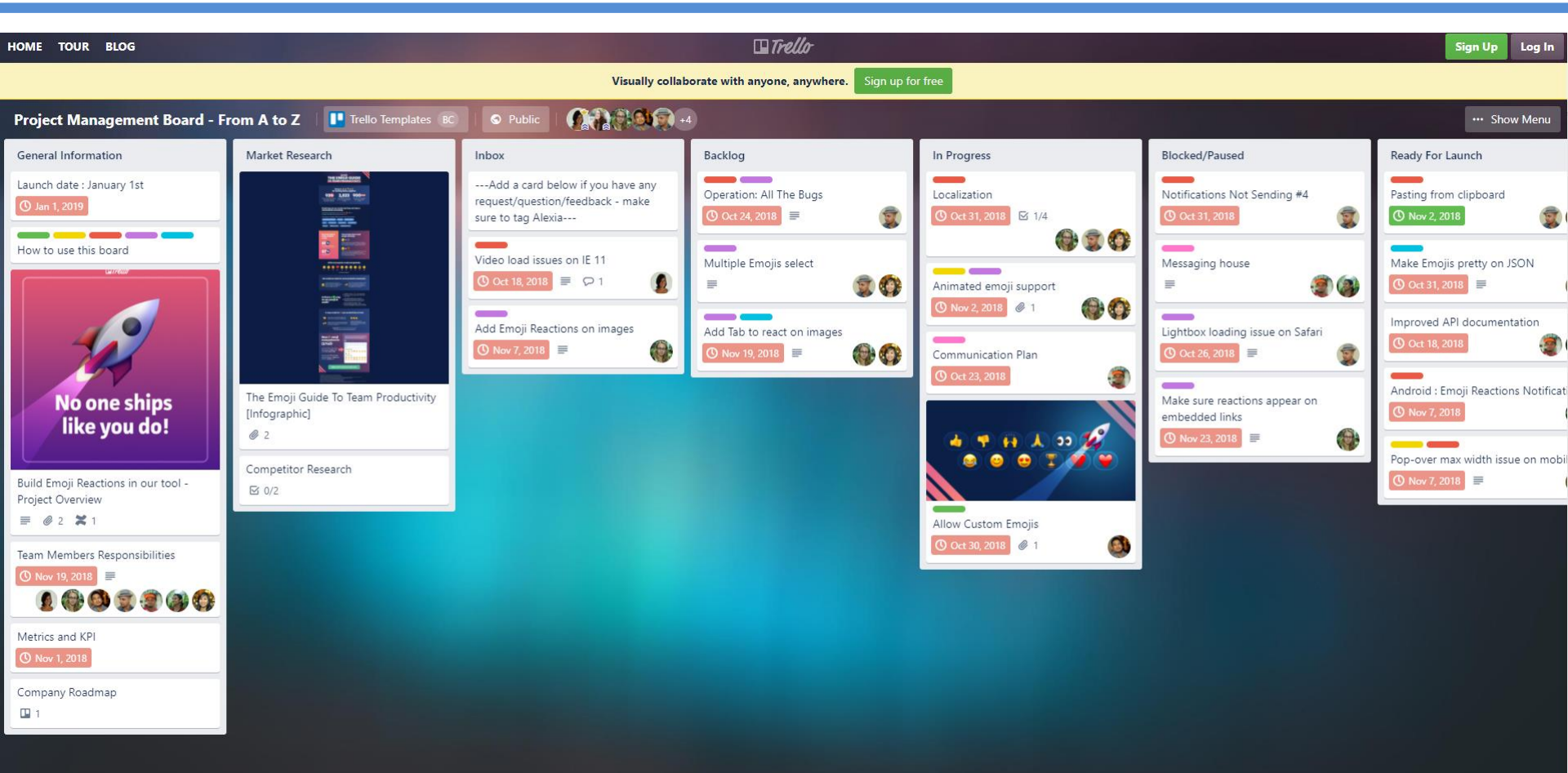
# Metodologias Ágeis

## KANBAN





# Metodologias Ágeis

## KANBAN



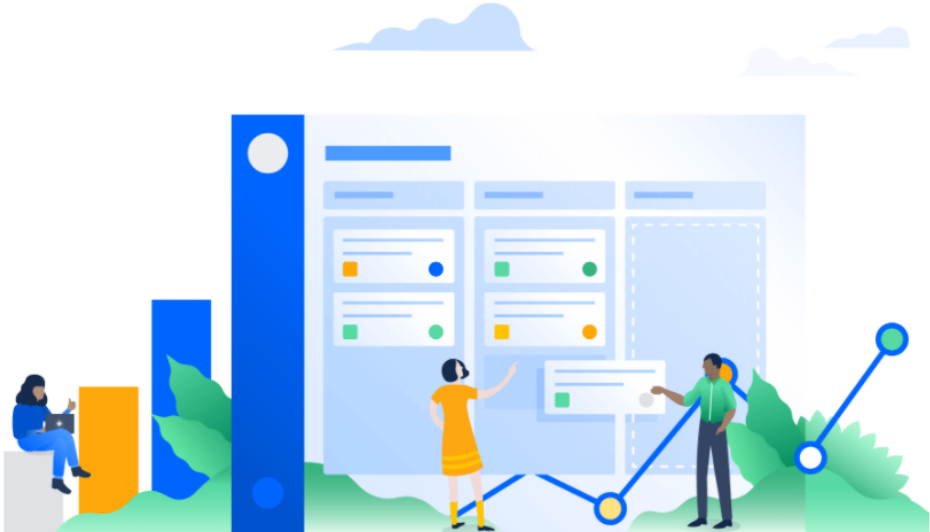
# Ferramentas

 [Products](#) [For teams](#) [Support](#) [Buy now](#) [Log in](#)

 [Features](#) [Product guide](#) [Pricing](#) [Enterprise](#) [Get it free](#)

The #1 software development tool used by agile teams

[Get it free](#)



The best software teams ship early and often.

Jira Software is built for every member of your software team to plan, track, and release great software.

# Ferramentas






Products ▾



For teams ▾

Support ▾




## PLAN, TRACK, & SUPPORT

-  **Jira Software**  
Project and issue tracking
-  **Jira Align**  
Enterprise agile planning
-  **Jira Core**  
Essential business management
-  **Jira Service Desk**  
Collaborative IT service management
-  **Opsgenie**  
Modern incident response
-  **Statuspage**  
Incident communication



## COLLABORATE

-  **Confluence**  
Document collaboration
-  **Trello**  
Collaborate visually on any project

## CODE, BUILD, & SHIP

-  **Bitbucket**  
Git code management
-  **Sourcetree**  
Git and Mercurial desktop client
-  **Bamboo**  
Integration and release management

## SECURITY & IDENTITY

-  **Atlassian Access**  
Security and control for cloud
-  **Crowd**  
User management for self-managed environments

# Ferramentas

The screenshot displays the Jira interface for the 'Teams in Space' project. The left sidebar contains navigation options: Backlog, Board (selected), Reports, Releases, Components, Issues, Repository, Add item, and Settings. The main area is titled 'Board' and shows a Kanban-style workflow with four columns: TO DO (5 items), IN PROGRESS (5 items), CODE REVIEW (2 items), and DONE (8 items). Each item includes a title, a team label (e.g., SPACE TRAVEL PARTNERS, LOCAL MARS OFFICE), a status icon, a count, and a due date. A 'Release' button is visible in the top right corner.

Column	Item	Team	Status	Count	Due Date
TO DO (5)	Engage Jupiter Express for outer solar system travel	SPACE TRAVEL PARTNERS	✓	5	TIS-25
	Create 90 day plans for all departments in the Mars Office	LOCAL MARS OFFICE	+	9	TIS-12
	Engage Saturn's Rings Resort as a preferred provider	SPACE TRAVEL PARTNERS	+	3	TIS-17
	Enable Speedy SpaceCraft as the preferred	SPACE TRAVEL PARTNERS	+	3	TIS-17
IN PROGRESS (5)	Requesting available flights is now taking > 5 seconds	SEESPACEEZ PLUS	+	3	TIS-8
	Engage Saturn Shuttle Lines for group tours	SPACE TRAVEL PARTNERS	✓	4	TIS-15
	Establish a catering vendor to provide meal service	LOCAL MARS OFFICE	+	4	TIS-15
	Engage Saturn Shuttle Lines for group tours	SPACE TRAVEL PARTNERS	+	4	TIS-15
CODE REVIEW (2)	Register with the Mars Ministry of Revenue	LOCAL MARS OFFICE	+	3	TIS-11
	Draft network plan for Mars Office	LOCAL MARS OFFICE	✓	3	TIS-15
DONE (8)	Homepage footer uses an inline style - should use a class	LARGE TEAM SUPPORT	+	1	TIS-68
	Engage JetShuttle SpaceWays for travel	SPACE TRAVEL PARTNERS	+	5	TIS-23
	Engage Saturn Shuttle Lines for group tours	SPACE TRAVEL PARTNERS	✓	1	TIS-15
	Establish a catering vendor to provide meal service	LOCAL MARS OFFICE	+	1	TIS-15



# Outras metodologias ágeis (MIX)





---

# Seleção da Metodologia

# Metodologias Ágeis

## Seleção do Modelo de Desenvolvimento

	Sequencial	Iterativa
<b>Requisitos</b>	Estáveis	Instáveis ou pouco entendidos
<b>Desenho</b>	Familiar e repetível	Complexo e desafiante
<b>Equipa</b>	Conhece o contexto	Estranha ao contexto
<b>Risco</b>	Reduzido	Elevado
<b>Previsibilidade</b>	Importante	Negligenciável
<b>Alterações</b>	Caras	Baratas

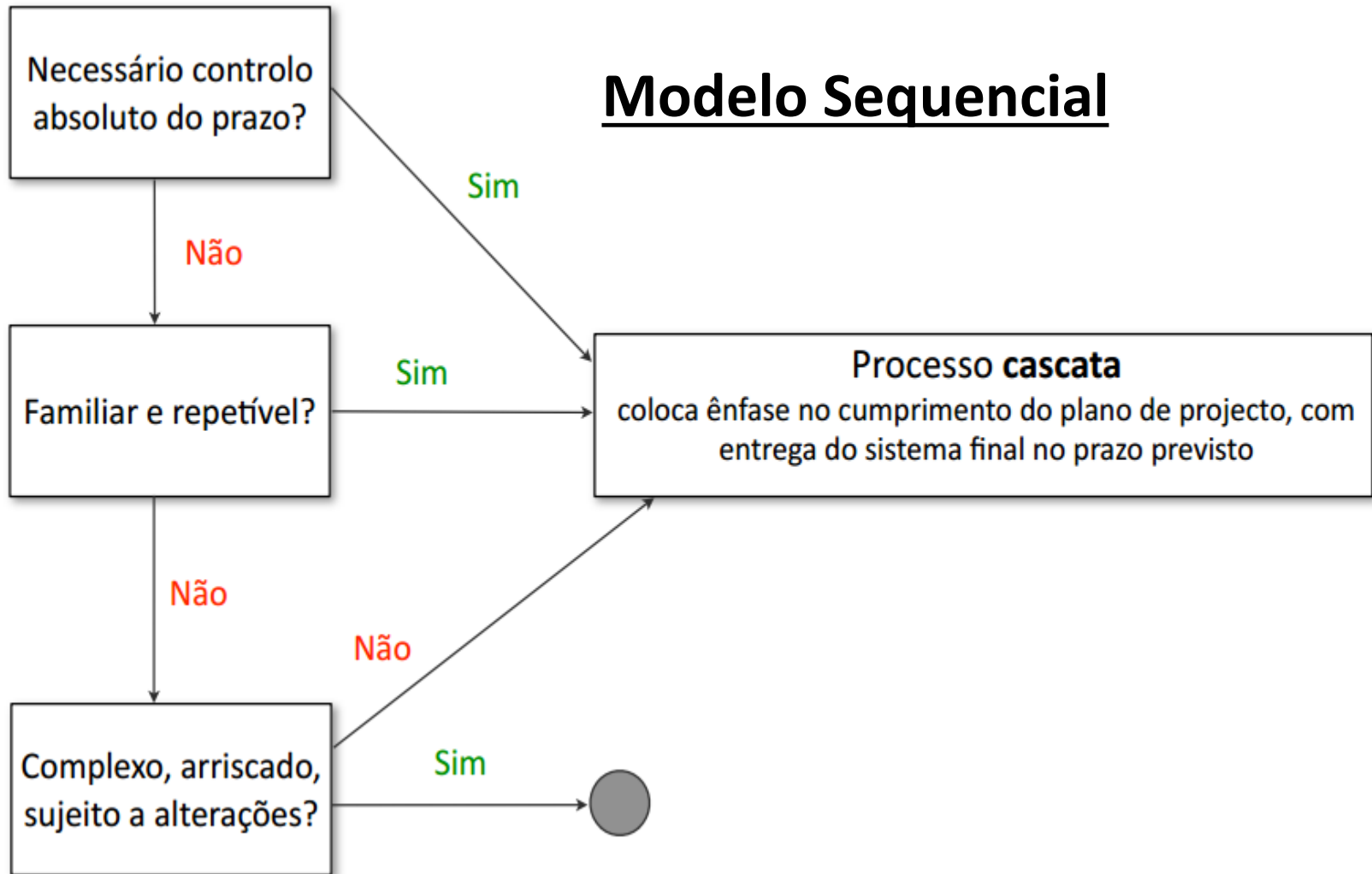


### **Modelo sequencial linear**

coloca ênfase no cumprimento do plano de projeto, com entrega do sistema final no prazo previsto

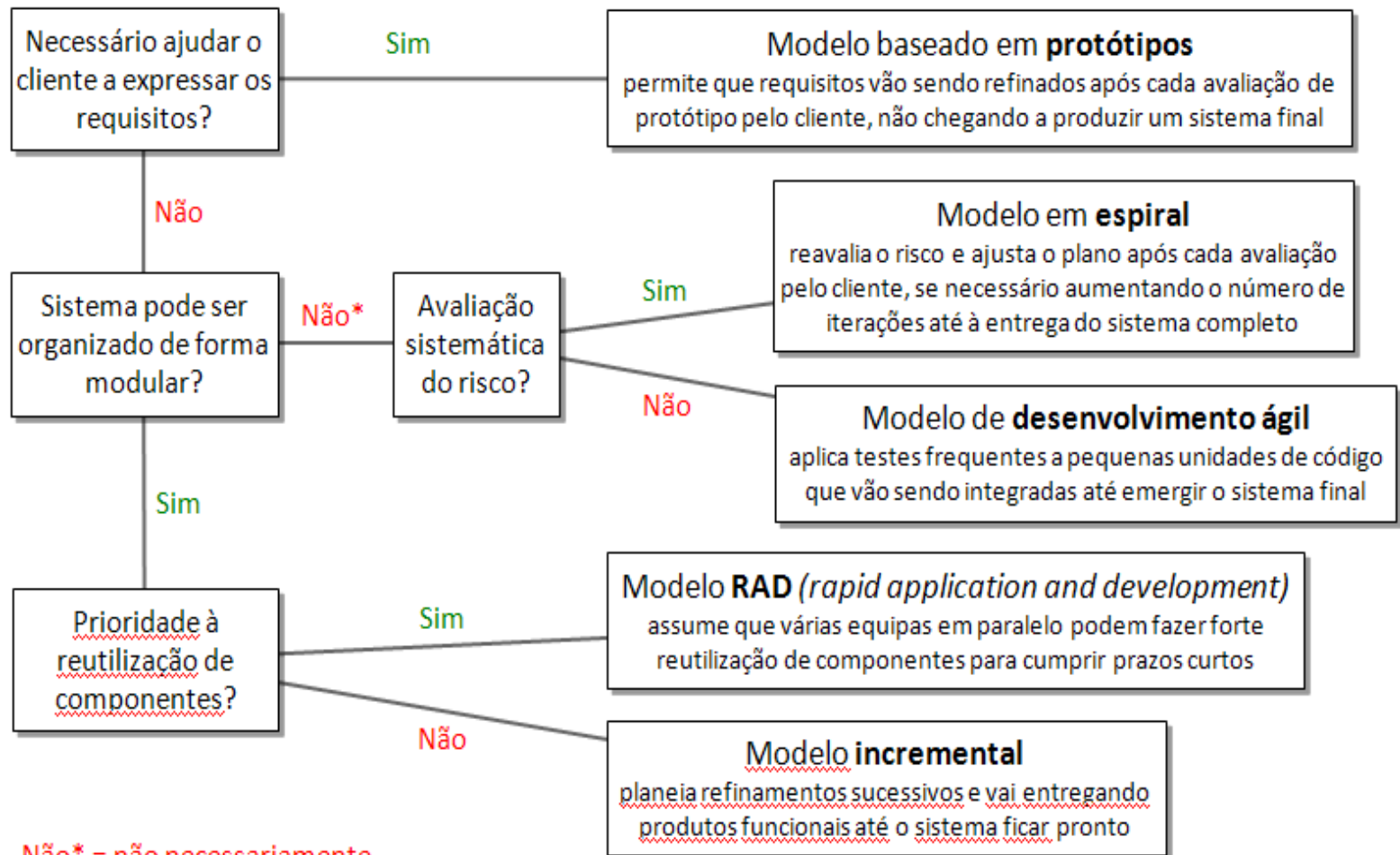
# Metodologias Ágeis

## Modelo Sequencial



# Metodologias Ágeis

## Modelos Iterativos



Não\* = não necessariamente

# Metodologias de desenvolvimento de SW