Métodos ágeis e o SCRUM

Ilídio Oliveira

v2020/12/17,

universidade de aveiro
departamento de eletrónica,
telecomunicações e informática

deti

Objetivos de aprendizagem

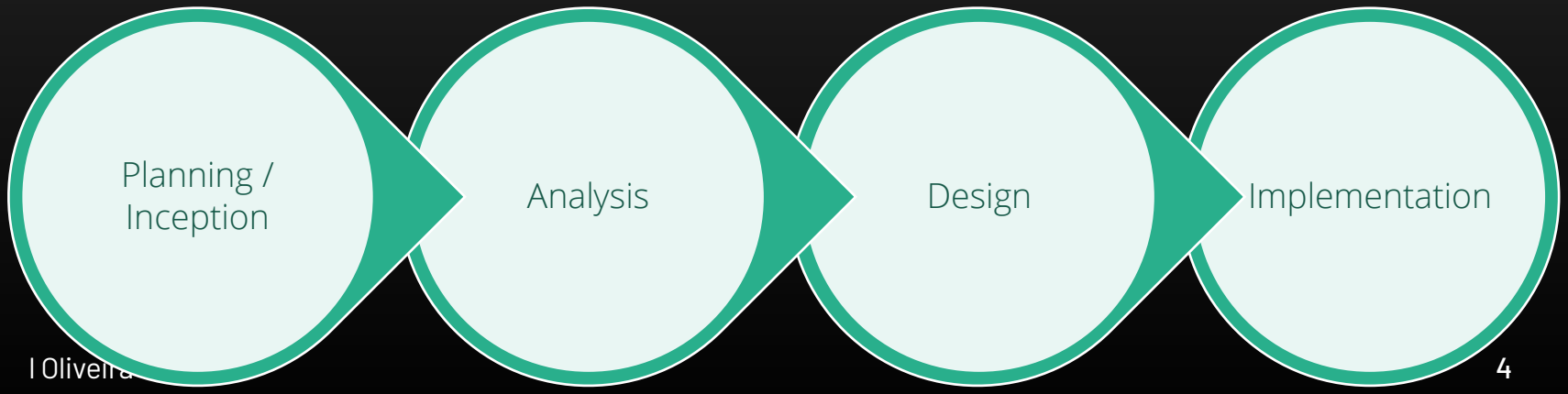
- Identificar vantagens de estruturar um projeto em iterações, produzindo incrementos frequentes.
- Caracterizar os princípios da gestão do *backlog* em abordagens ágeis.
- Identificar os papéis numa equipa de Scrum e as principais "cerimónias"
- Descrever a complementaridade entre o processo de software (e.g.: OpenUP) e uma metodologia de gestão de equipas (e.g.: Scrum)

Duas abordagens paradigmáticas do SDLC

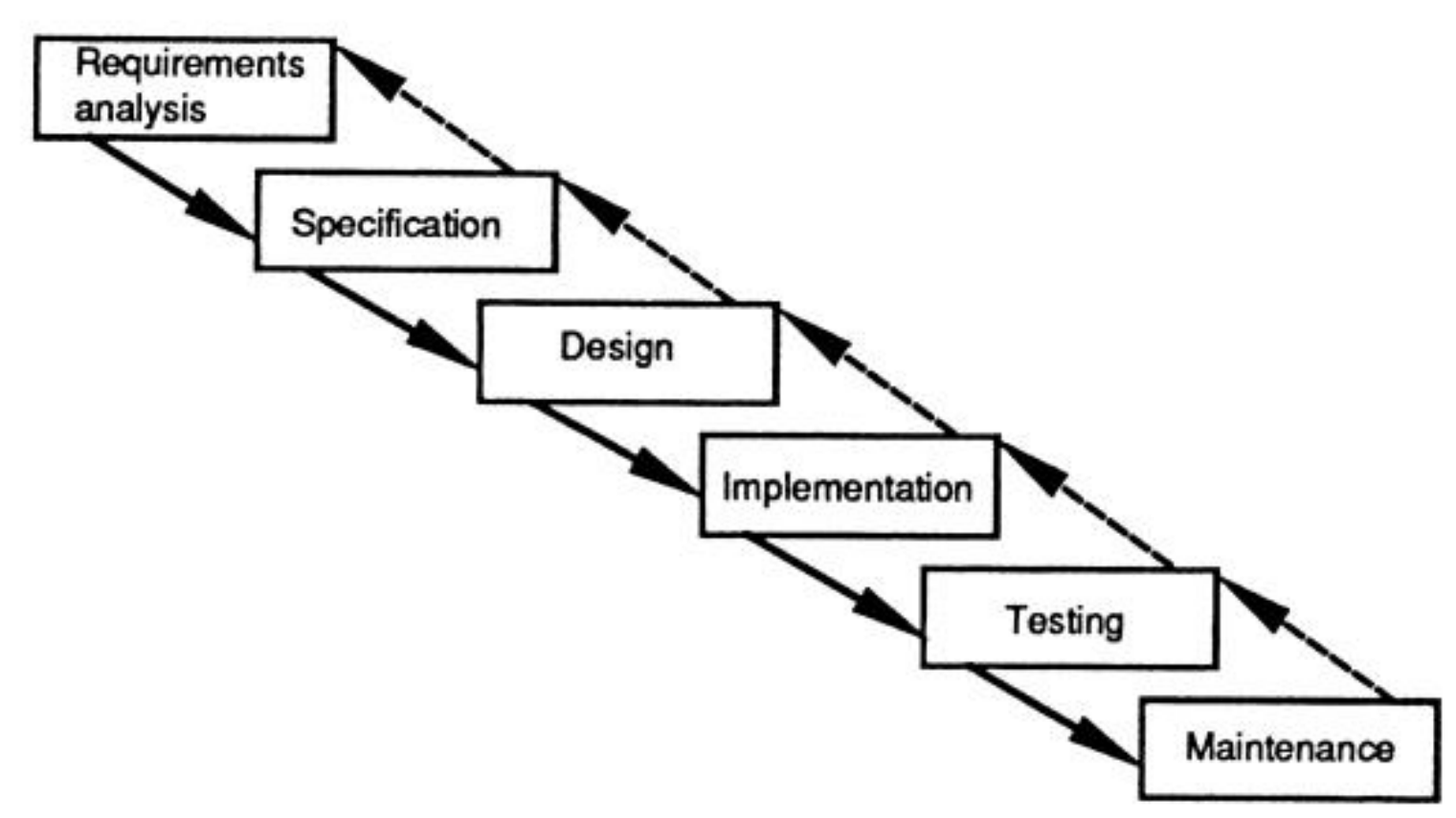
SDLC (ciclo de vida)

Quatro fases fundamentais: planeamento, análise, desenho técnico e implementação. Diferentes projetos podem abordar as fases de diferentes formas, mas todos os projetos têm elementos destas quatro fases.

Cada fase é composta por uma série de atividades, que usam disciplinas técnicas/competências adequadas, para produzir os resultados (*outcomes*).

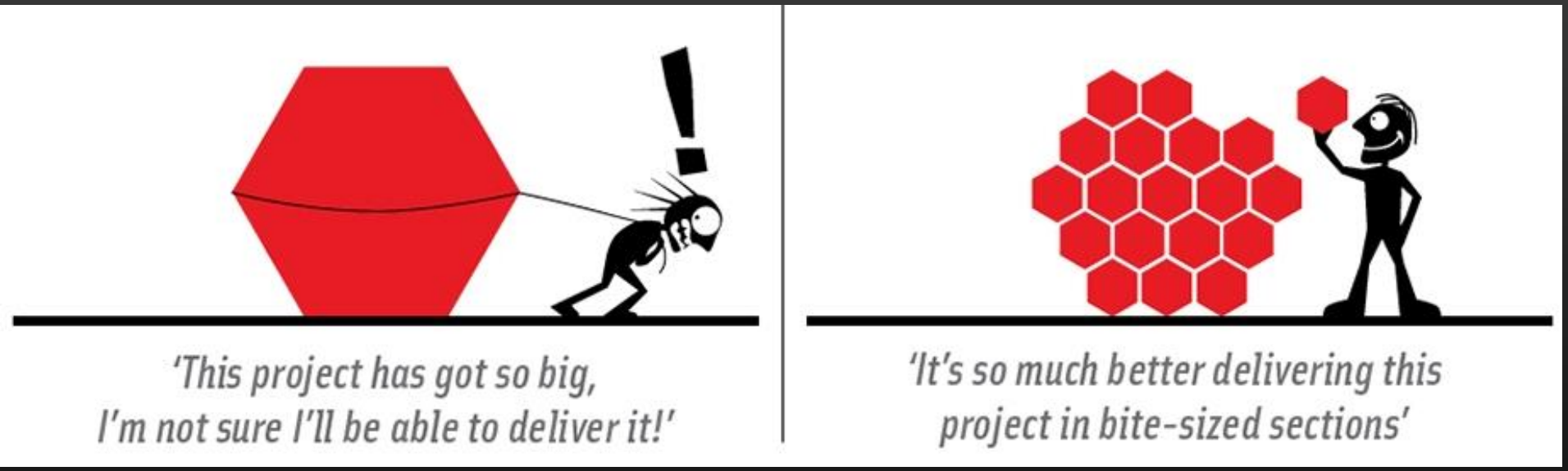


Abordagem "clássica" tipificada pelo **Waterfall model**

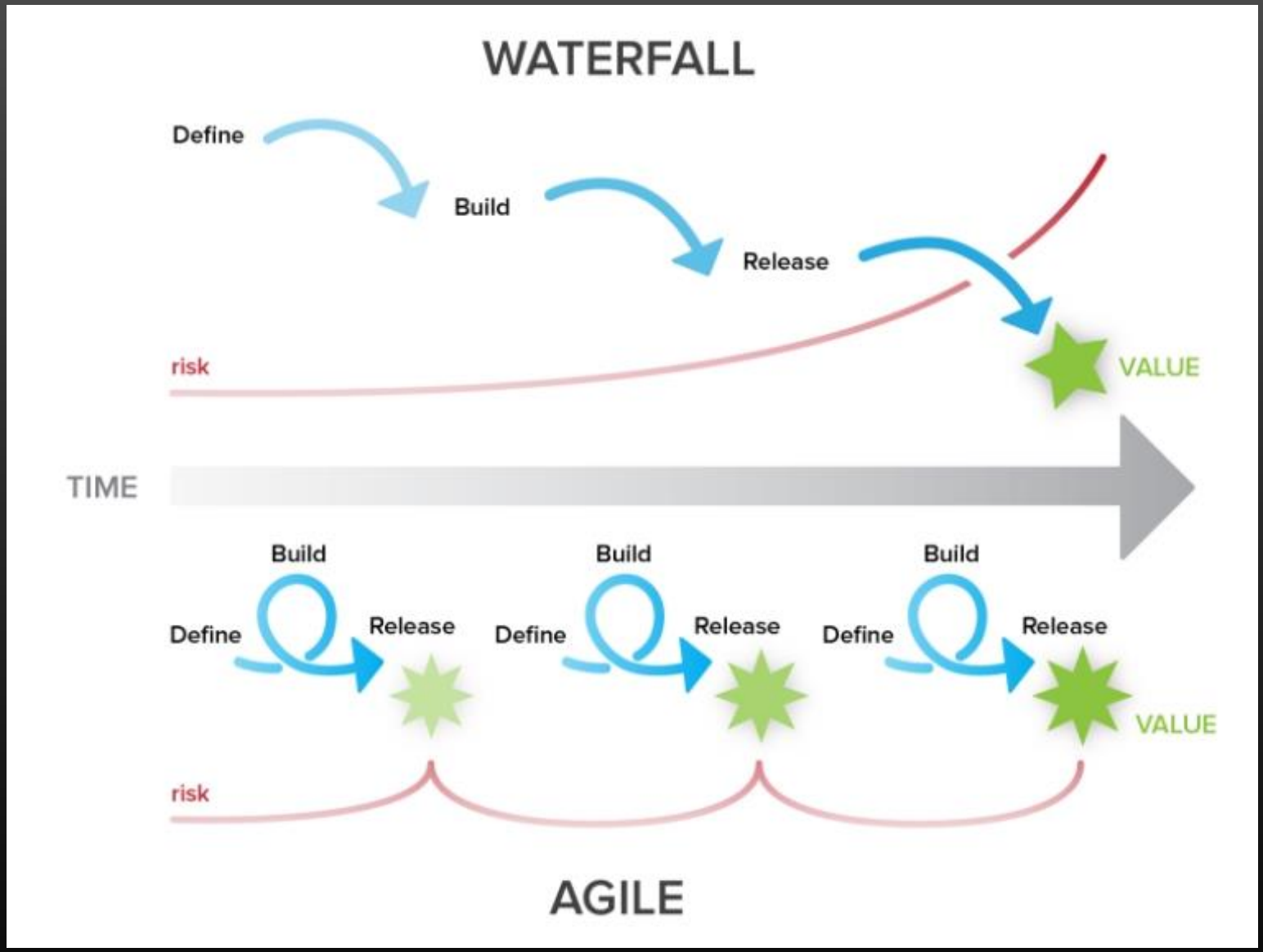


W. Royce, "Managing the Development of Large Software Systems," *Proc. Westcon*, IEEE CS Press, 1970, pp. 328-339.

Um grande projeto? Vários pequenos “projetos”?

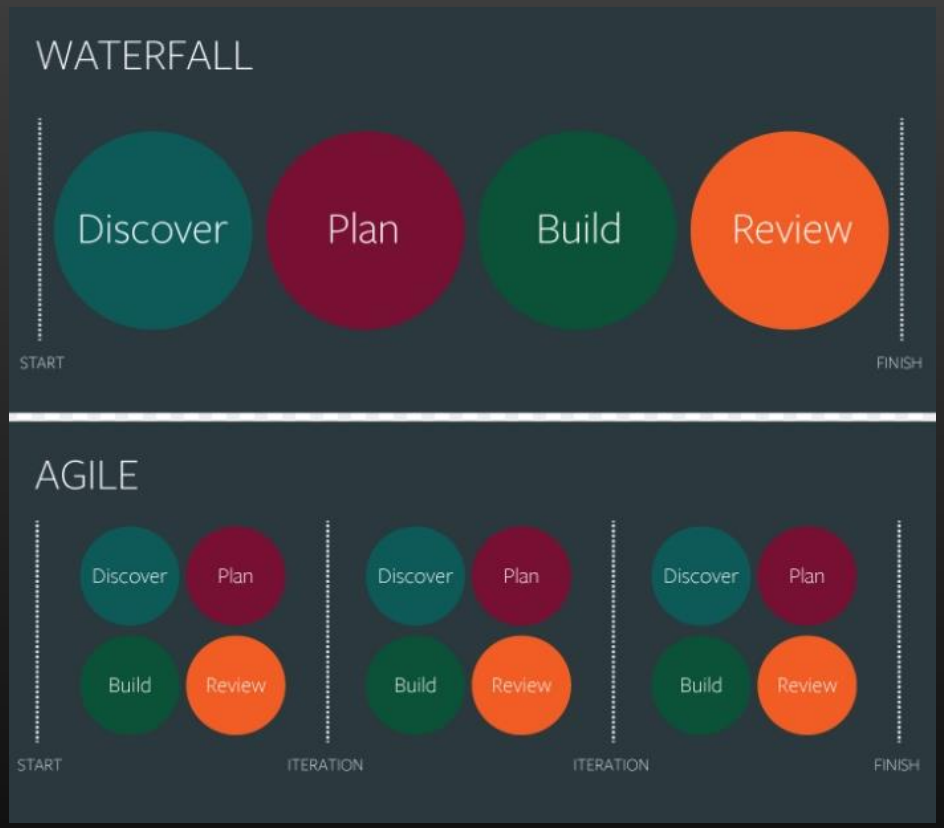


https://blog.ganttpro.com/en/waterfall-vs-agile-with-advantages-and-disadvantages/



https://blog.ganttpro.com/en/waterfall-vs-agile-with-advantages-and-disadvantages/

O que priorizar?



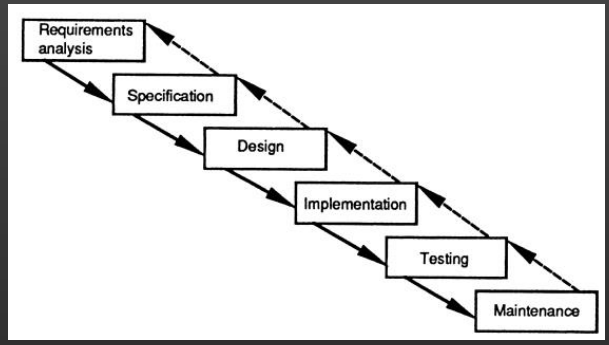
Desenvolvimento iterativo (ciclos curtos) *vs* desenvolvimento linear

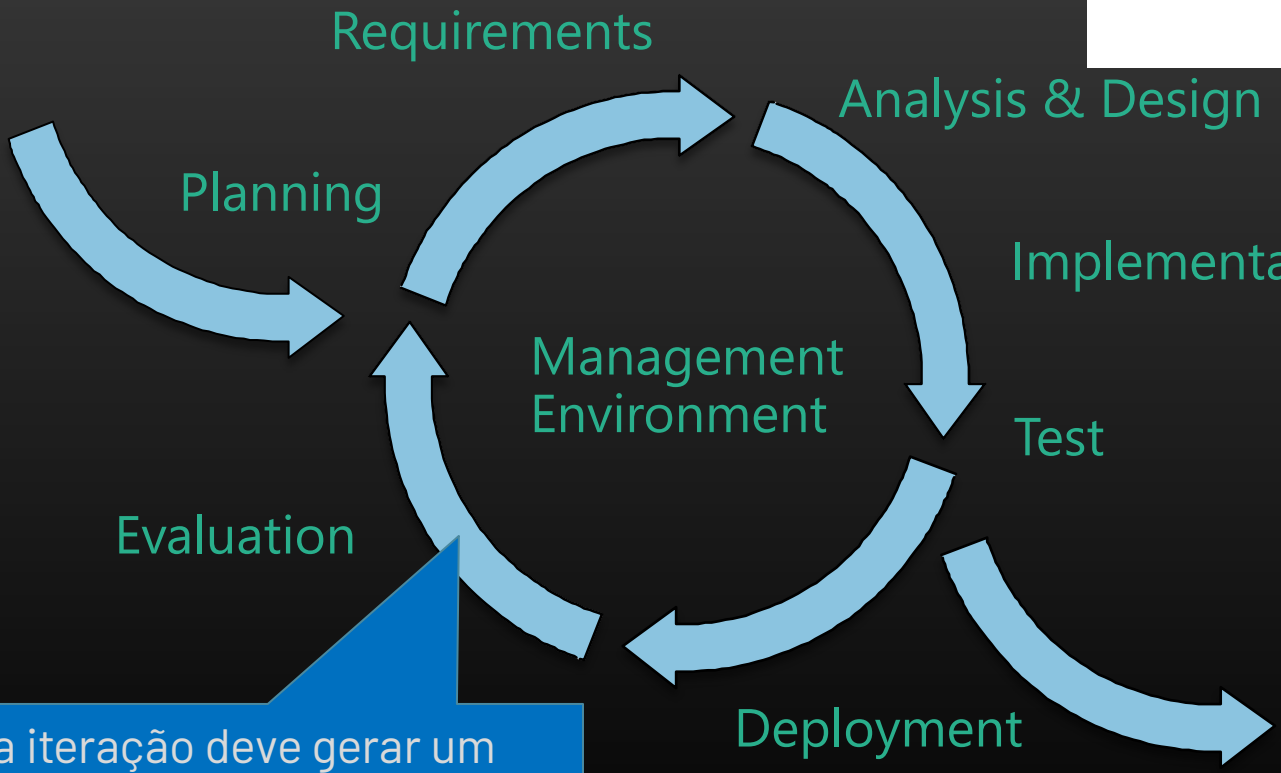
Interação frequente com o promotor/negócio *vs* flutuações na participação dos *stakeholders*

Melhor ter uma boa colaboração ou um bom plano?

Alterações são acolhidas para mitigar o risco *vs* evitar alterações para controlar o risco

O desenvolvimento iterativo foca a entrega de valor orientada por ciclos curtos



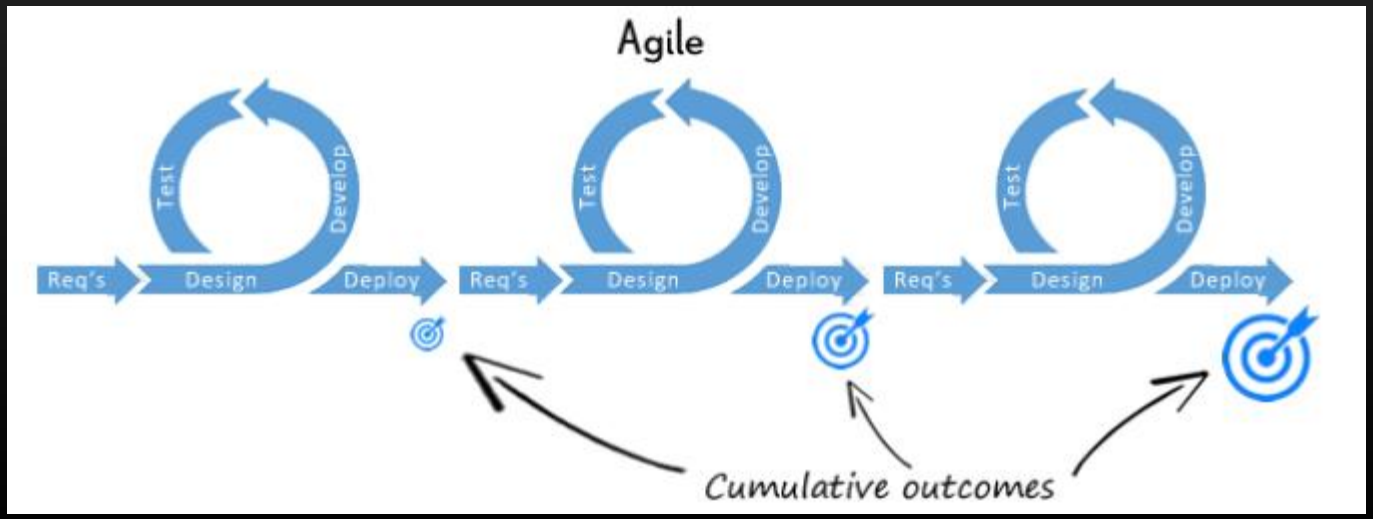


Cada iteração deve gerar um incremento functional (entrega de valor)

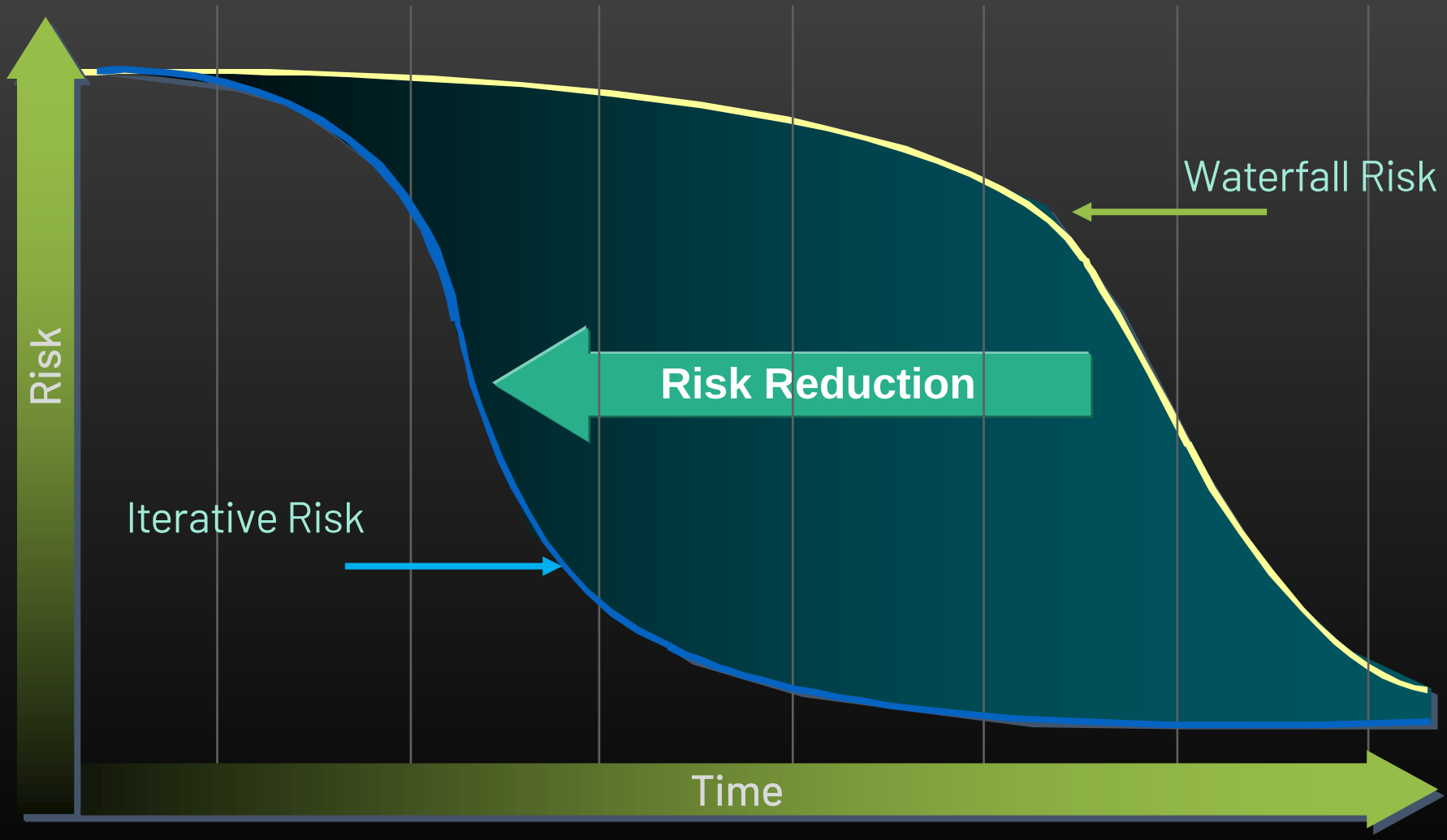
Desenvolvimento iterativo

Cada iteração envolve escolher um pequeno subconjunto dos requisitos, para projetar/desenhar, implementar e testar.

- Desenvolvimento em ciclos curtos
- Cada ciclo dá um incremento executável (parcial)
- Cada incremento é testado e integrado
- O feedback de cada iteração leva ao requinte e adaptação da próxima.



Entregas frequentes, integração em contínuo
→ **redução de risco**



Questões...

O sucesso de um projeto (de desenvolvimento de um SI) depende do seu tamanho?

As abordagens ágeis funcionam melhor que as abordagens sequenciais?

A abordagens ágeis funcionam melhor em certos tipos de projeto (e.g.: quanto ao tamanho)?

https://www.infoq.com/articles/standish-chaos-2015/

→ http://bit.ly/3moxw1e

A “normalização” do *agile*

“Embrace change”

Em vez de:

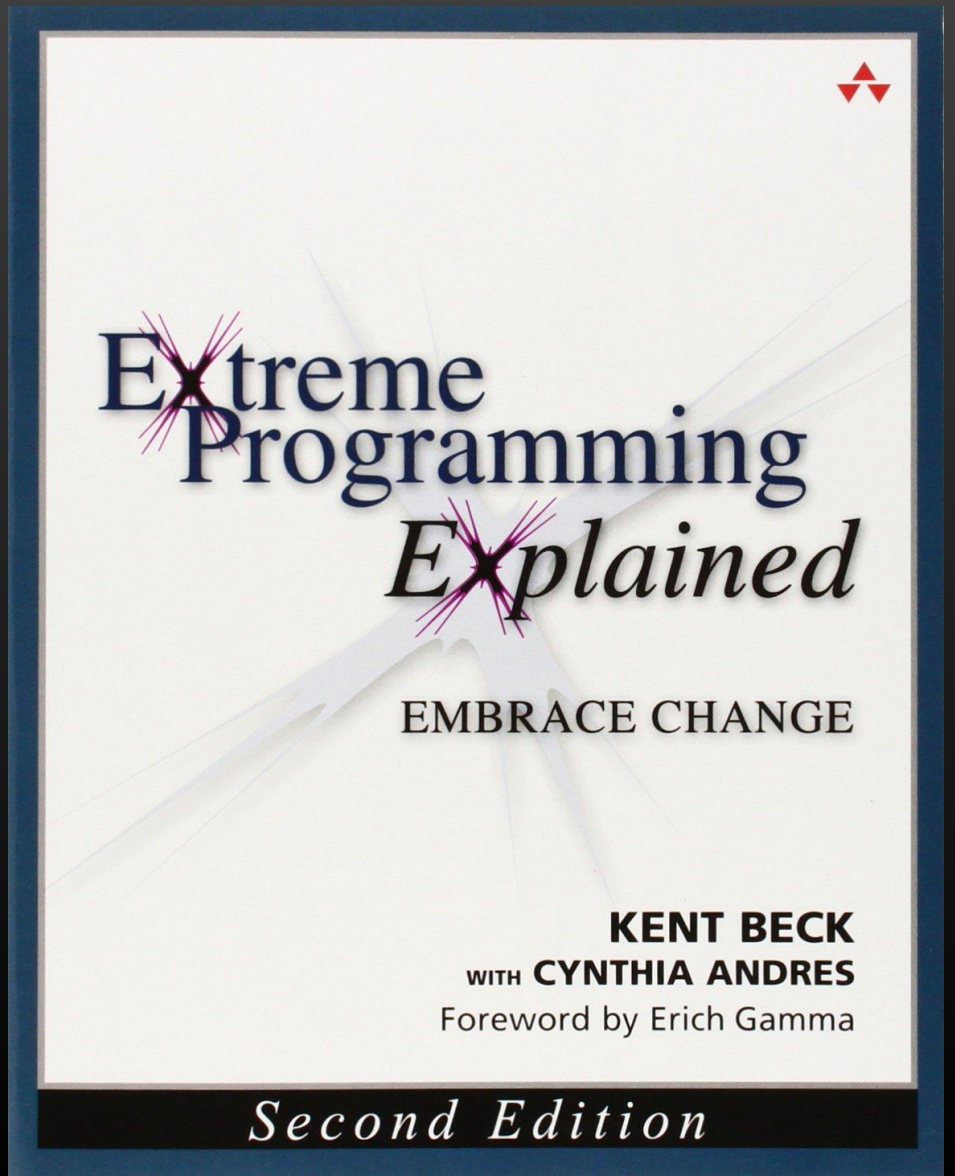
→ Lutar contra a inevitável a mudança que ocorre no desenvolvimento de software

→ Tentando (sem sucesso) especificar, congelar e "assinar" num conjunto de requisitos congelados e conceber antes da implementação

desenvolvimento iterativo e evolutivo:

→ Baseia-se numa atitude de abraçar a mudança e a adaptação como fatores inevitáveis e, na verdade, essenciais.

→ Mas: isto não quer dizer que o desenvolvimento iterativo incentive um processo descontrolado e reativo.



Manifesto para o Desenvolvimento Ágil de Software.

Ao desenvolver e ao ajudar outros a desenvolver software,
temos vindo a descobrir melhores formas de o fazer.
Através deste processo começámos a valorizar:

Indivíduos e interacções mais do que processos e ferramentas

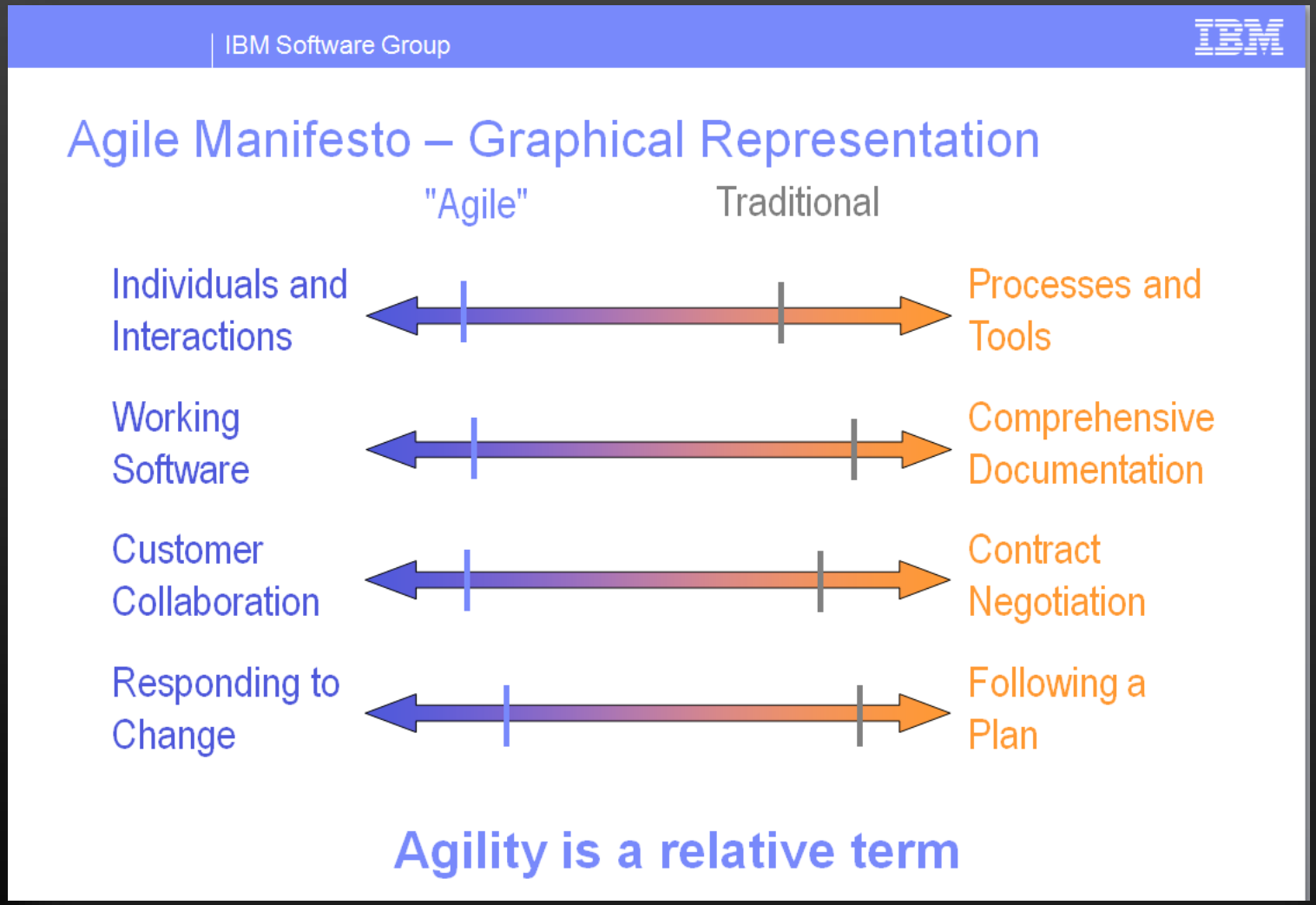
Software funcional mais do que documentação abrangente

Colaboração com o cliente mais do que negociação contratual

Responder à mudança mais do que seguir um plano

Ou seja, apesar de reconhecermos valor nos itens à direita,
valorizamos mais os itens à esquerda.

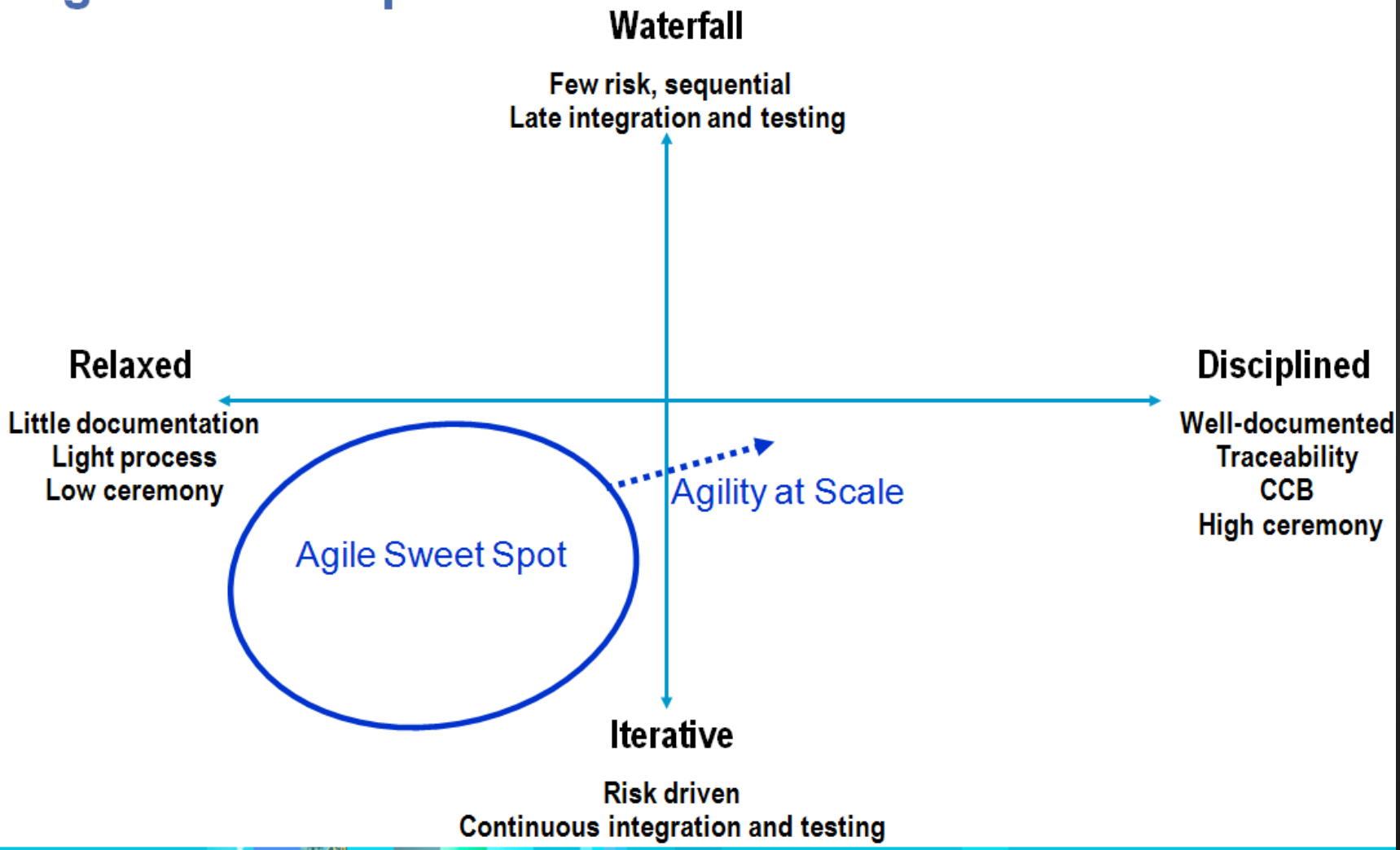
O desenvolvimento ágil de software



http://agilemanifesto.org

Credit: Per Kroll (IBM)

Agile Sweet Spot



Credit: Per Kroll (IBM)

Doze princípios para clarificar os valores

1. A nossa maior prioridade é, desde as primeiras etapas do projeto, a satisfação do cliente através da entrega rápida e contínua de valor (software implementado).

2. Aceitar alterações de requisitos, mesmo numa fase avançada do ciclo de desenvolvimento. Os processos ágeis potenciam a mudança em benefício da vantagem competitiva do cliente.

3. Fornecer frequentemente software pronto a funcionar. Os períodos de entrega devem ser de poucas semanas a poucos meses, dando preferência a períodos mais curtos.

4. As pessoas da área do negócio e a equipa de desenvolvimento devem trabalhar juntos, diariamente, durante o decorrer do projeto.

5. Desenvolver projetos com base em indivíduos motivados, dando-lhes o ambiente e o apoio de que necessitam, confiando que irão cumprir os objetivos.

6. O método mais eficiente e eficaz de passar informação para e dentro de uma equipa de desenvolvimento é através interações face-a-face (conversas diretas).

7. A principal medida de progresso é a entrega de software a funcionar.

8. Os processos ágeis promovem o desenvolvimento a um ritmo sustentável. Os promotores, a equipa e os utilizadores deverão ser capazes de manter um bom ritmo de trabalho, indefinidamente.

9. A atenção permanente à excelência técnica e um bom desenho da solução aumentam a agilidade.

10. Simplicidade – a arte de maximizar a quantidade de trabalho que não é feito – é essencial.

11. As melhores arquiteturas, requisitos e desenhos emergem das equipas que se auto-organizam.

12. A equipa reflete regularmente sobre o modo de se tornar mais eficaz, fazendo os ajustes e adaptações necessárias.

http://agilemanifesto.org/

Destacando algumas caraterísticas da abordagem "ágil"

Objetivos:

Satisfação do cliente

Mitigação de risco

Ritmo na equipa

Para atingir a agilidade é necessário:

- Ciclos curtos e entrega de valor frequente
- Envolvimento do "negócio"
- Resposta rápida à alteração de condições (e.g. alteração de requisitos)
- Colaboração e comunicação na equipa
- Aplicar práticas de construção de software evolutivas: integração em contínuo, desenvolvimento orientado por testes,...

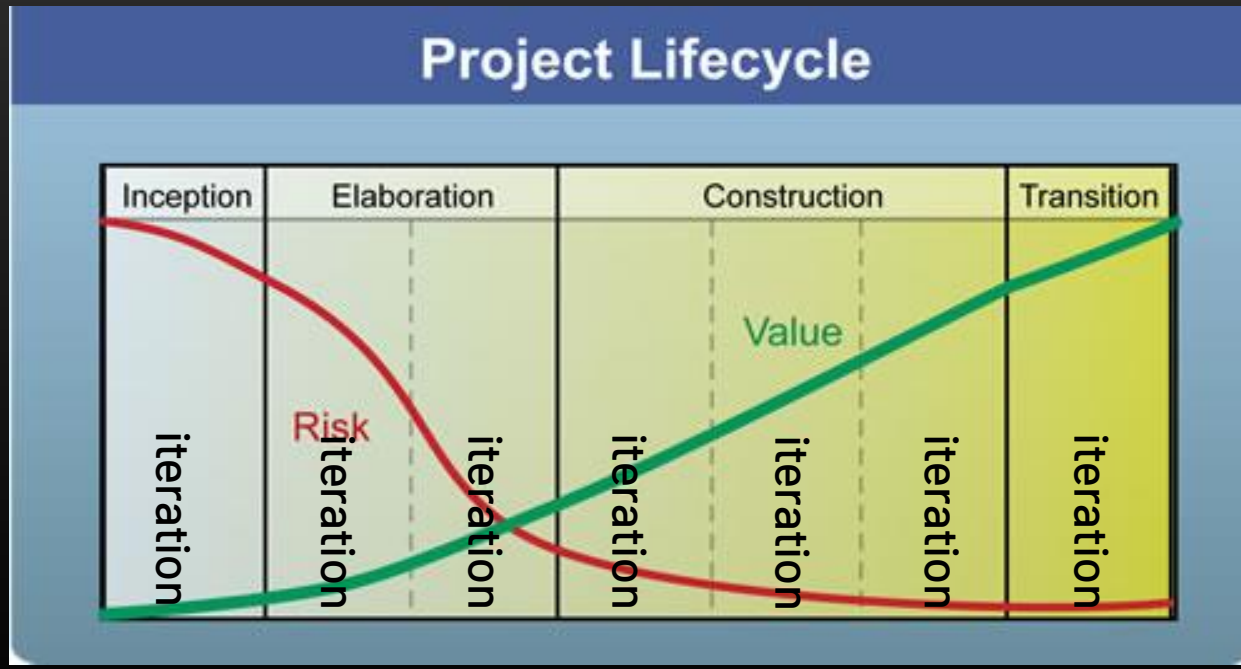
Como é que o Unified Process encaixa?

Estrutura do **Unified Process**

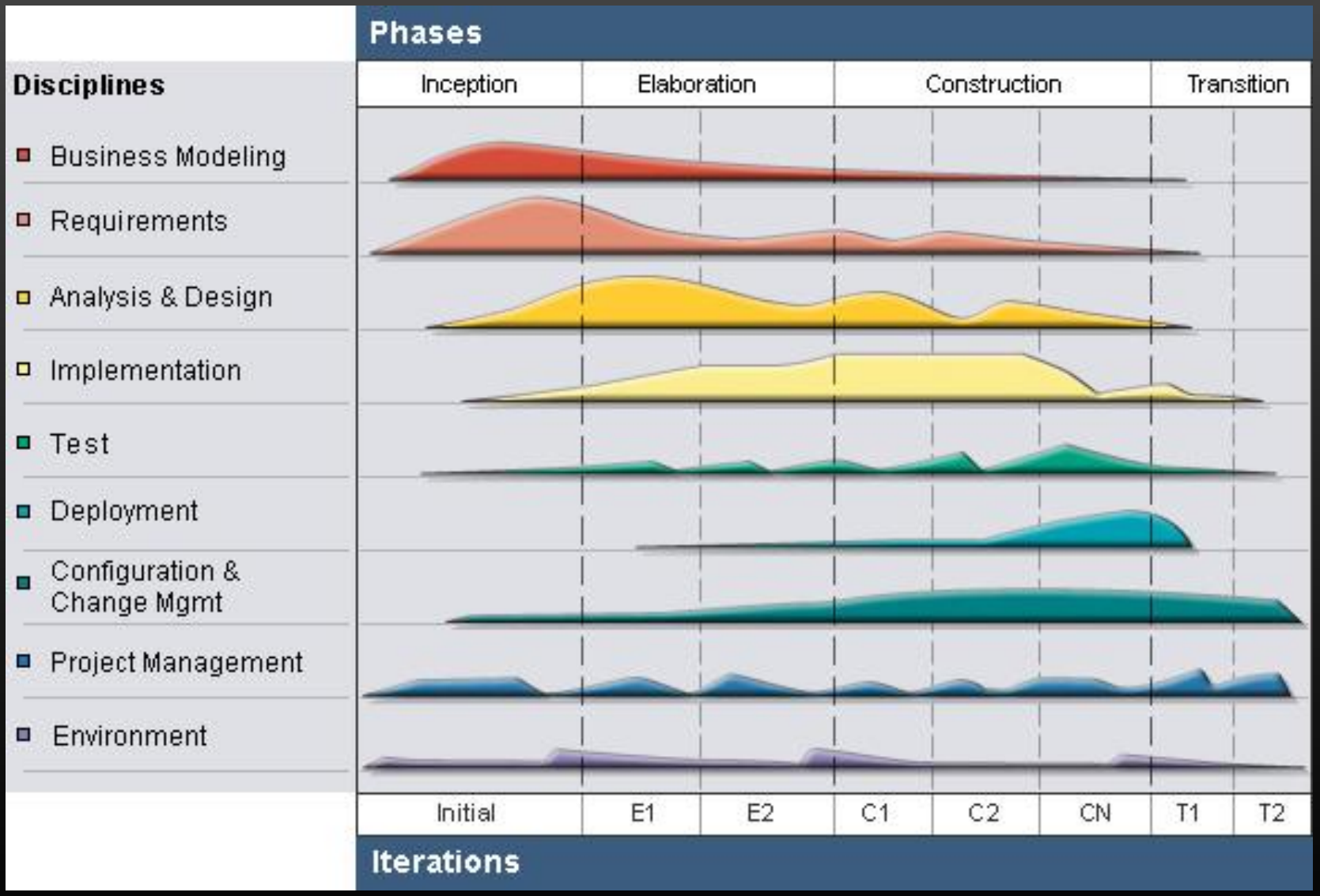
Interativo e incremental

Orientado pelos casos de utilização

Focado na arquitetura



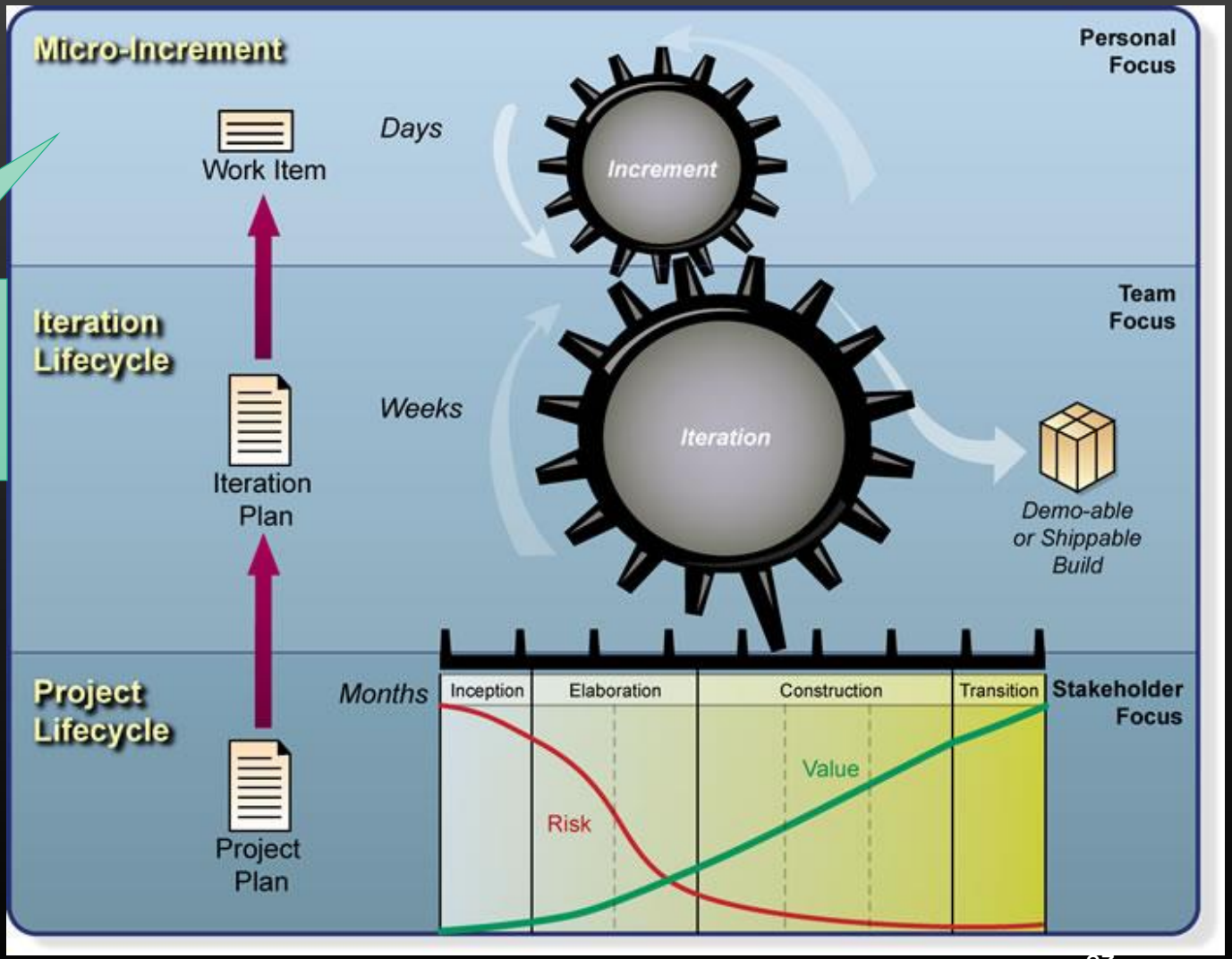
Ciclo de vida do Unified Process



OpenUP prevê o trabalho por iterações

→ OpenUP wiki

Apesar de ser clara a abordagem incremental, como é que se organiza a equipa do dia-a-dia?



Como organizar a equipa?

A metáfora do Scrum...

"restarting play in rugby football that involves players packing closely together with their heads down and attempting to gain possession of the ball"

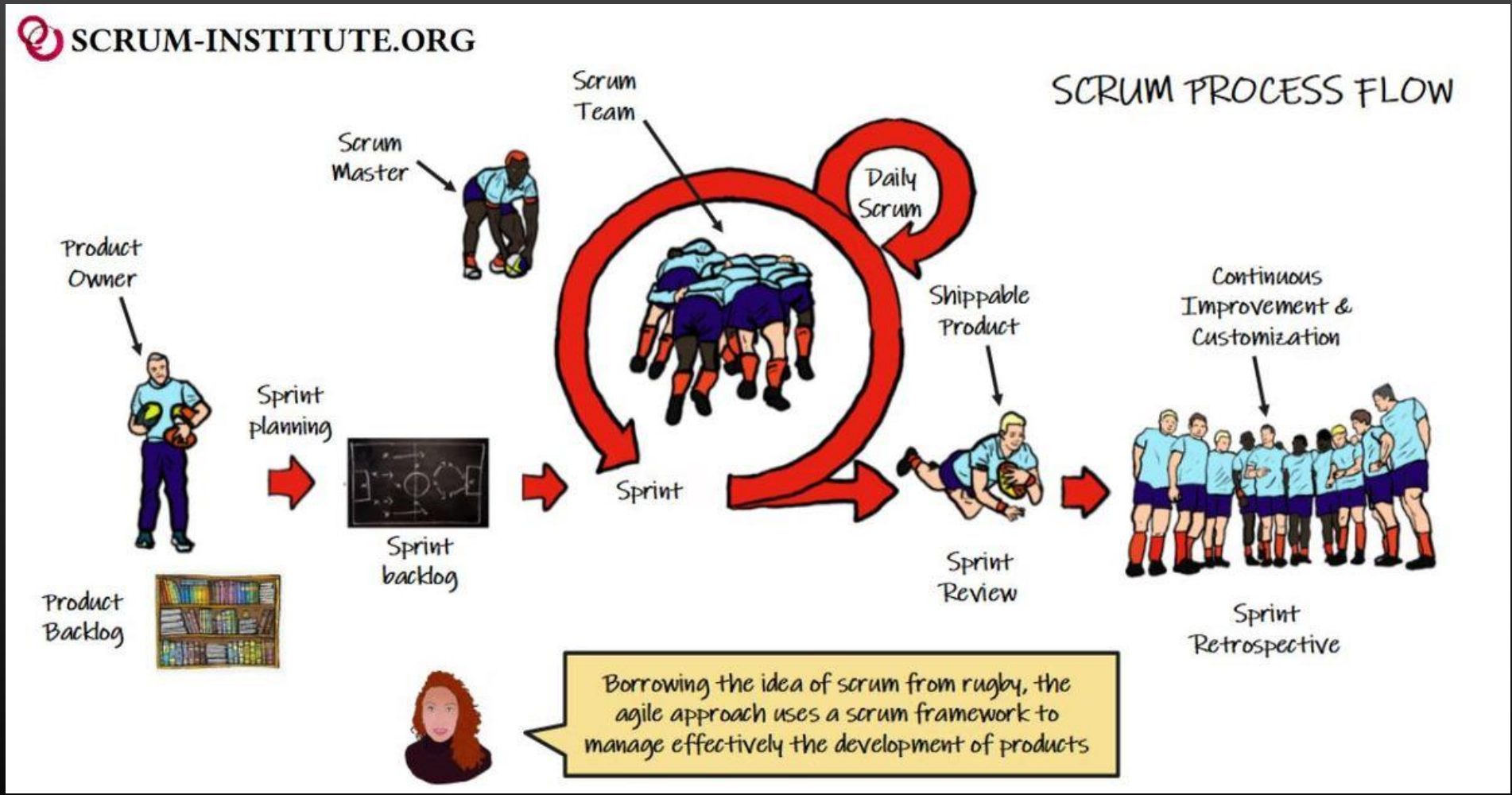
Guia SCRUM

Incluído no Moodle

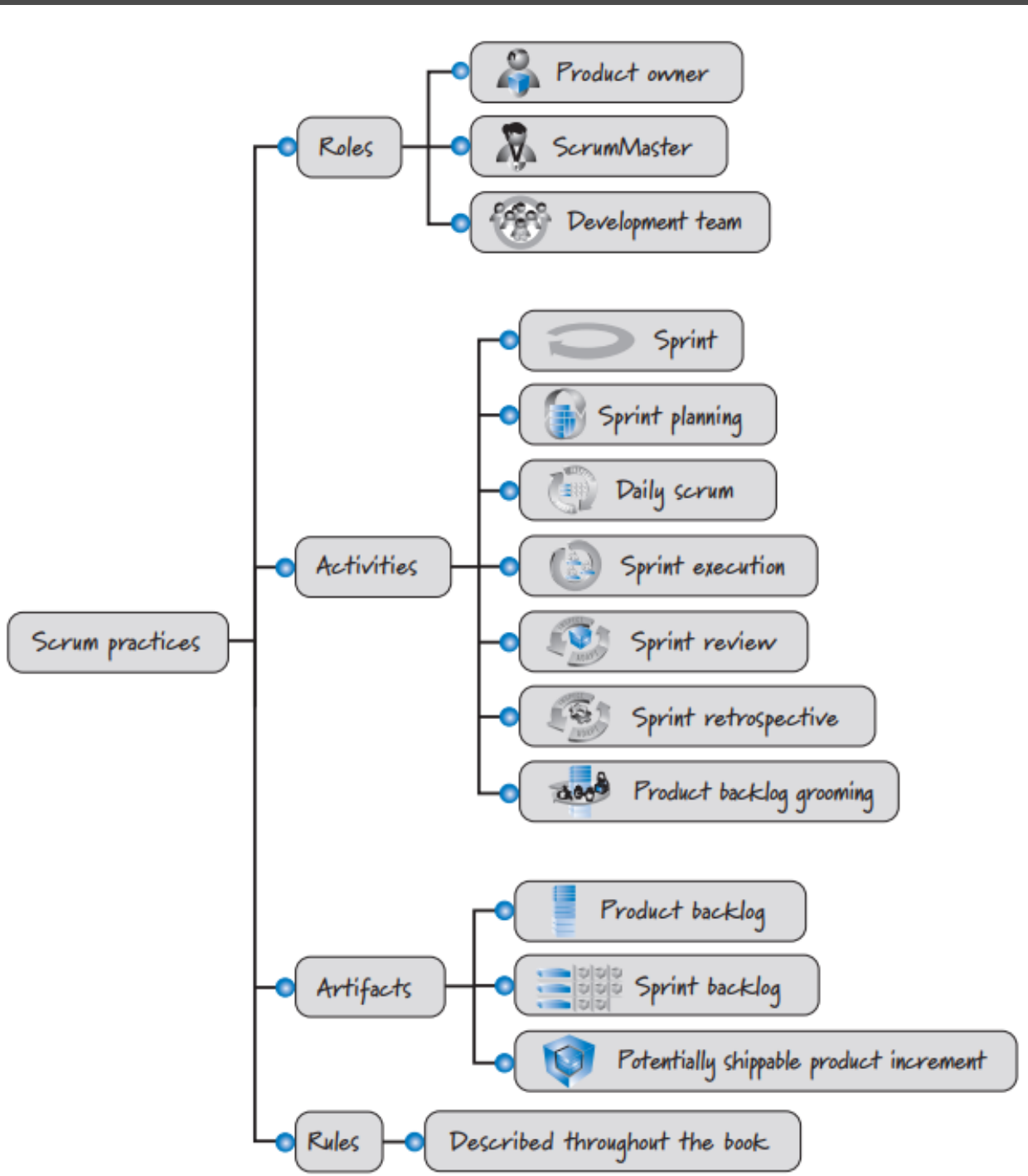
Scrum jobs?...

- Indeed.pt

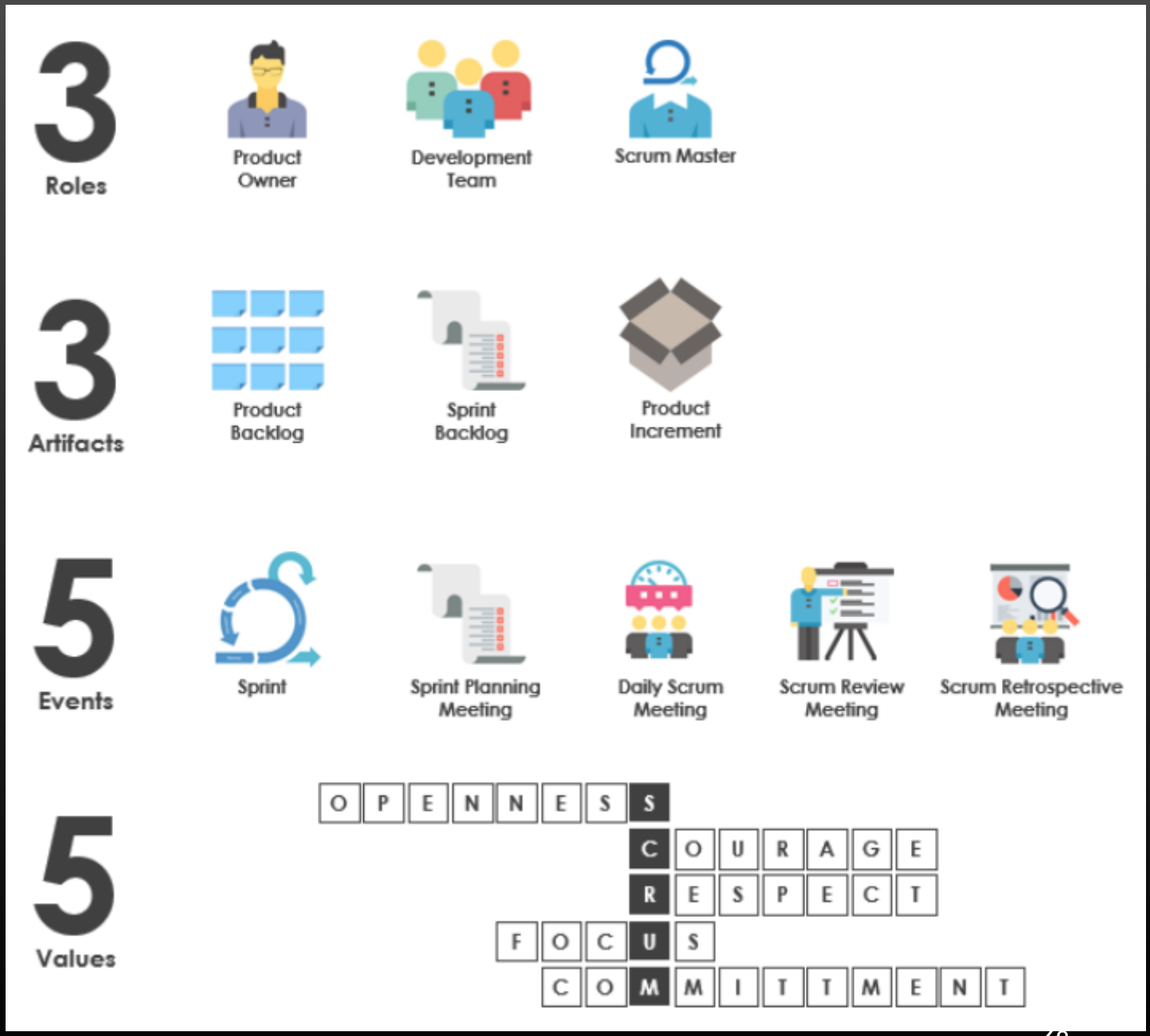
Metedologia de gestão de equipas SCRUM



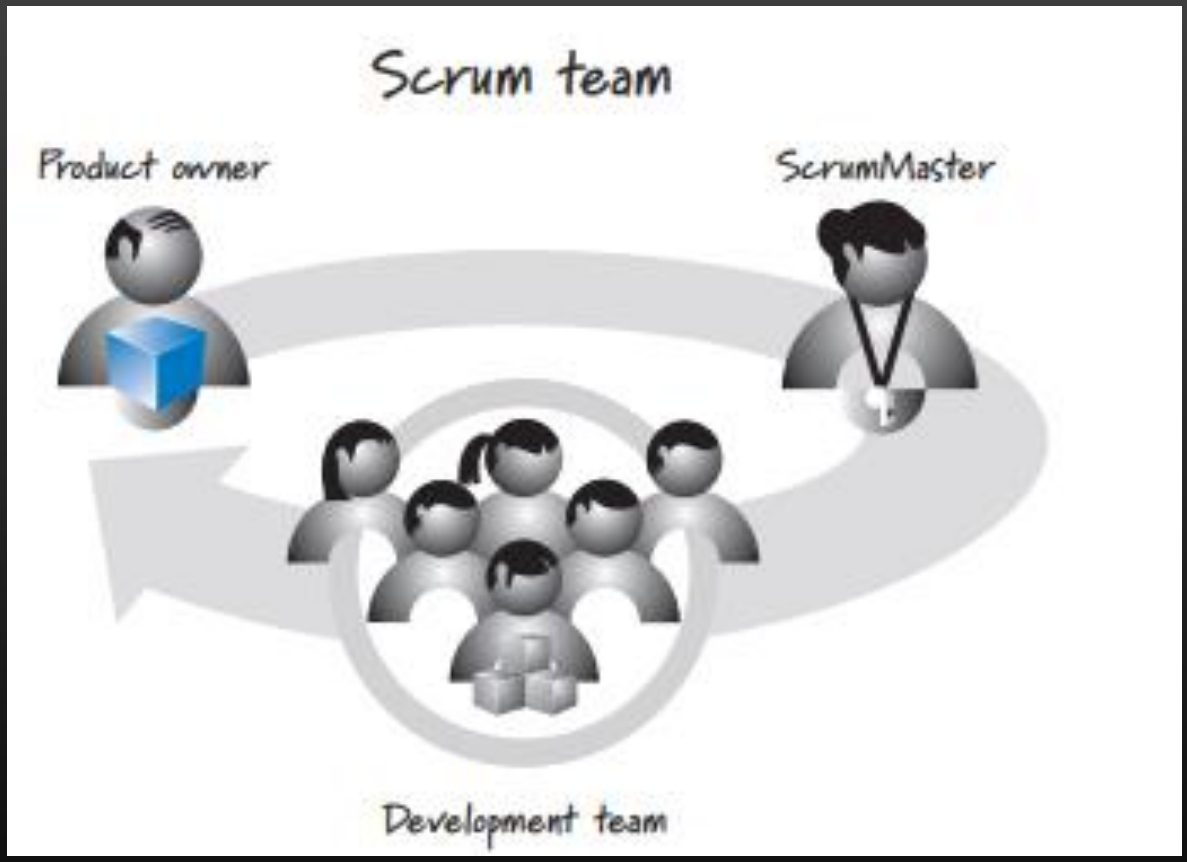
Elementos do Scrum



"3355"


3
Roles
Product Owner
Development Team
Scrum Master
3
Artifacts
Product Backlog
Sprint Backlog
Product Increment
5
Events
Sprint
Sprint Planning Meeting
Daily Scrum Meeting
Scrum Review Meeting
Scrum Retrospective Meeting
5
Values
OPENNESS
COURAGE
RESPECT
FOCUS
COMMITTMENT

Papéis previstos no Scrum



Creating a Product: Scrum: Roles & Responsibilities

Product Owner

- Holds the vision for the produtct
- Determines what needs to be done
- Sets the priorities to deliver the highest value

Scrum Master

- Help the team best use Scrum to build the product
- Protecting the Scrum process
- Prevent distractions/impediments

Development Team

- Builds the product
- Self-organizing group
- Takes on and determines how to deliver chunks of work in frequente increments

Credit: Nokia Networks.

Scrum process

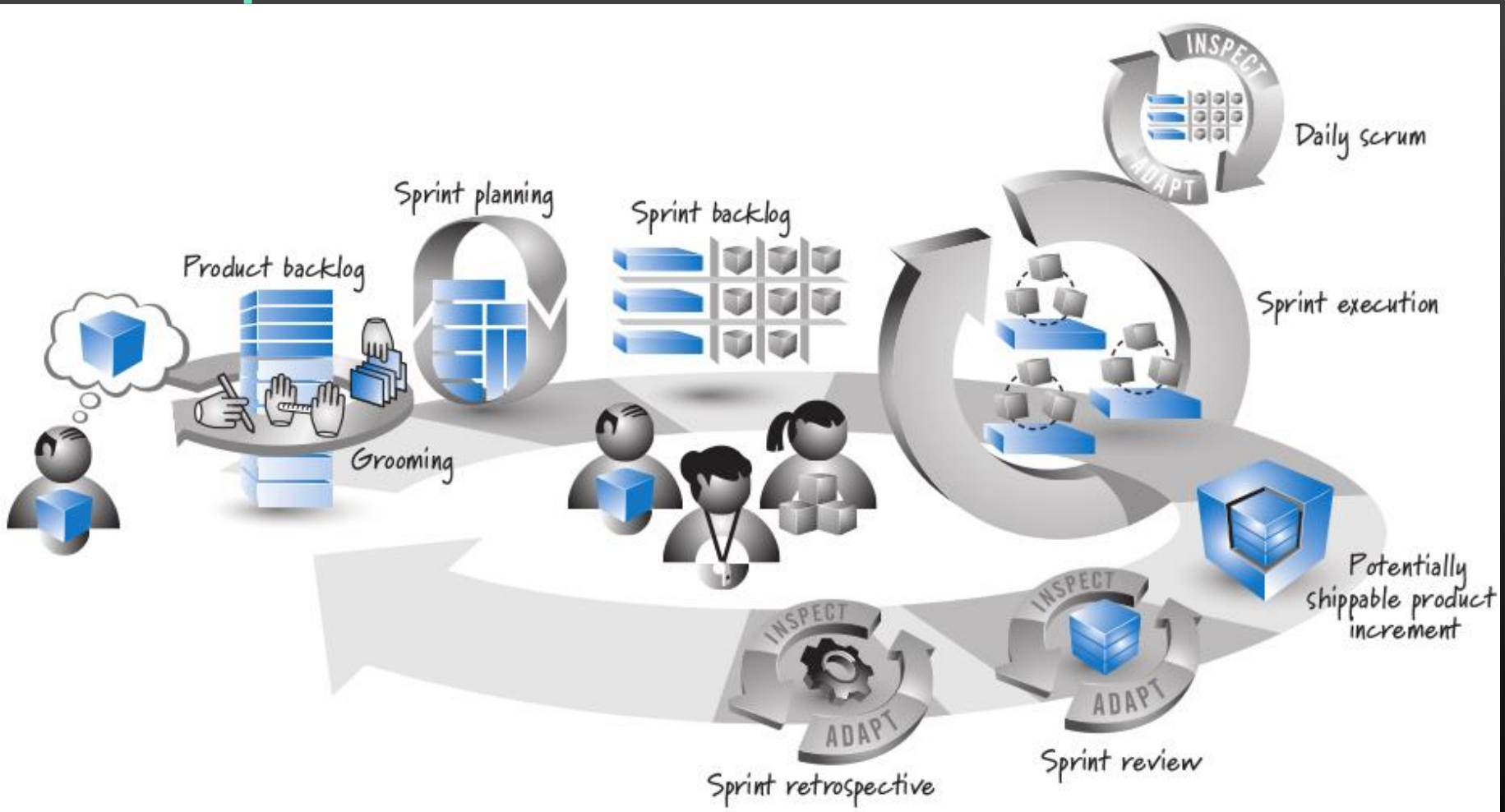
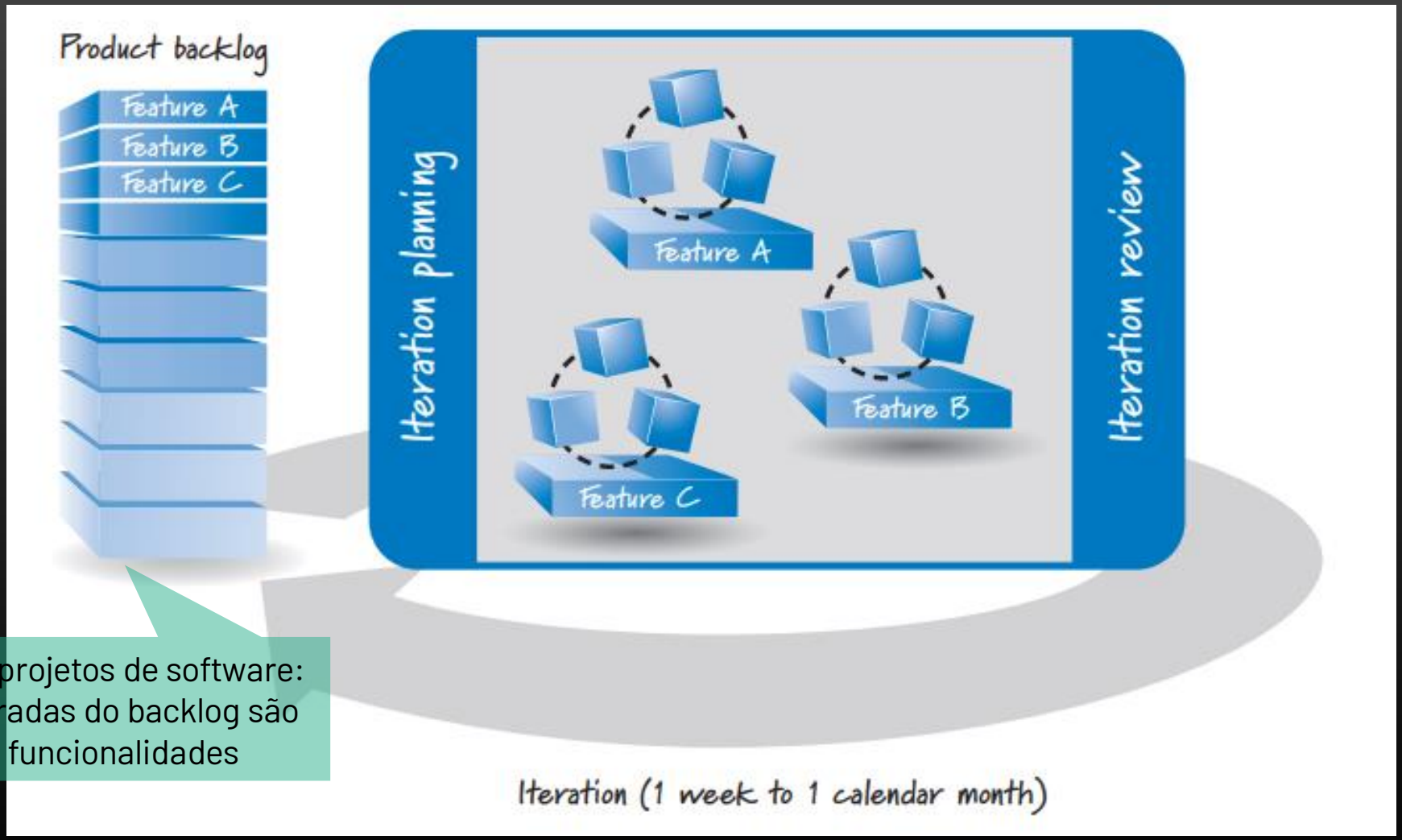


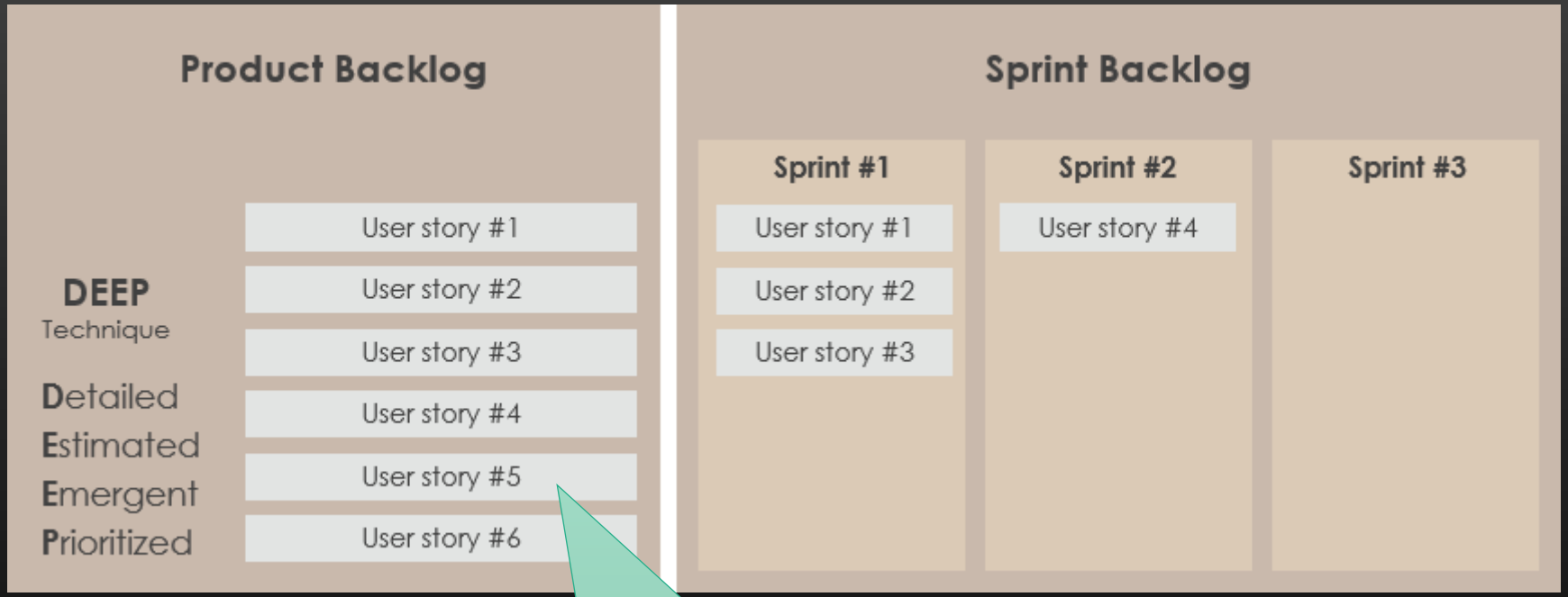
FIGURE 2.3 Scrum framework

Planeamento do trabalho e métodos ágeis



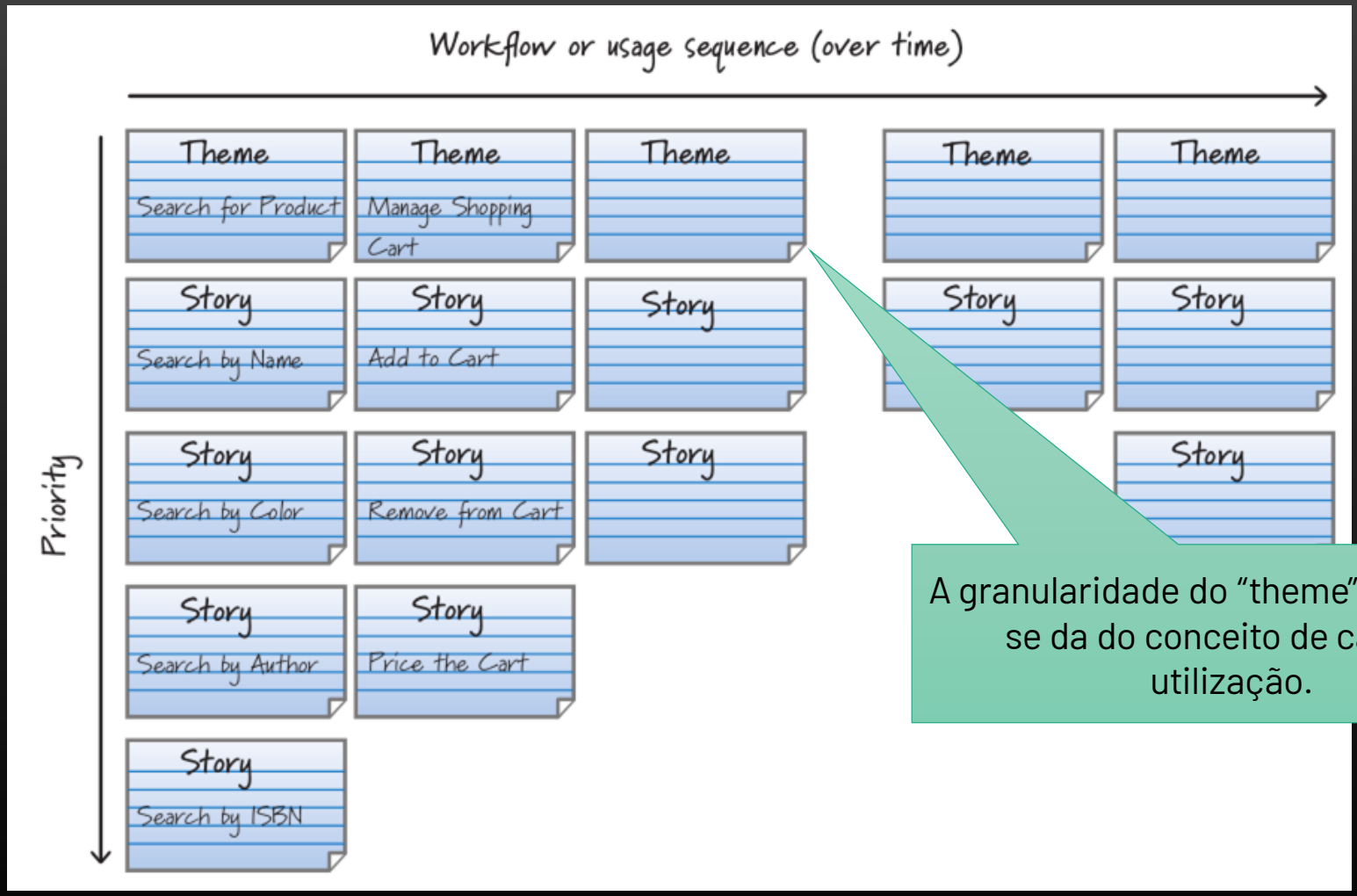
Em projetos de software: entradas do backlog são funcionalidades

Sprint planning

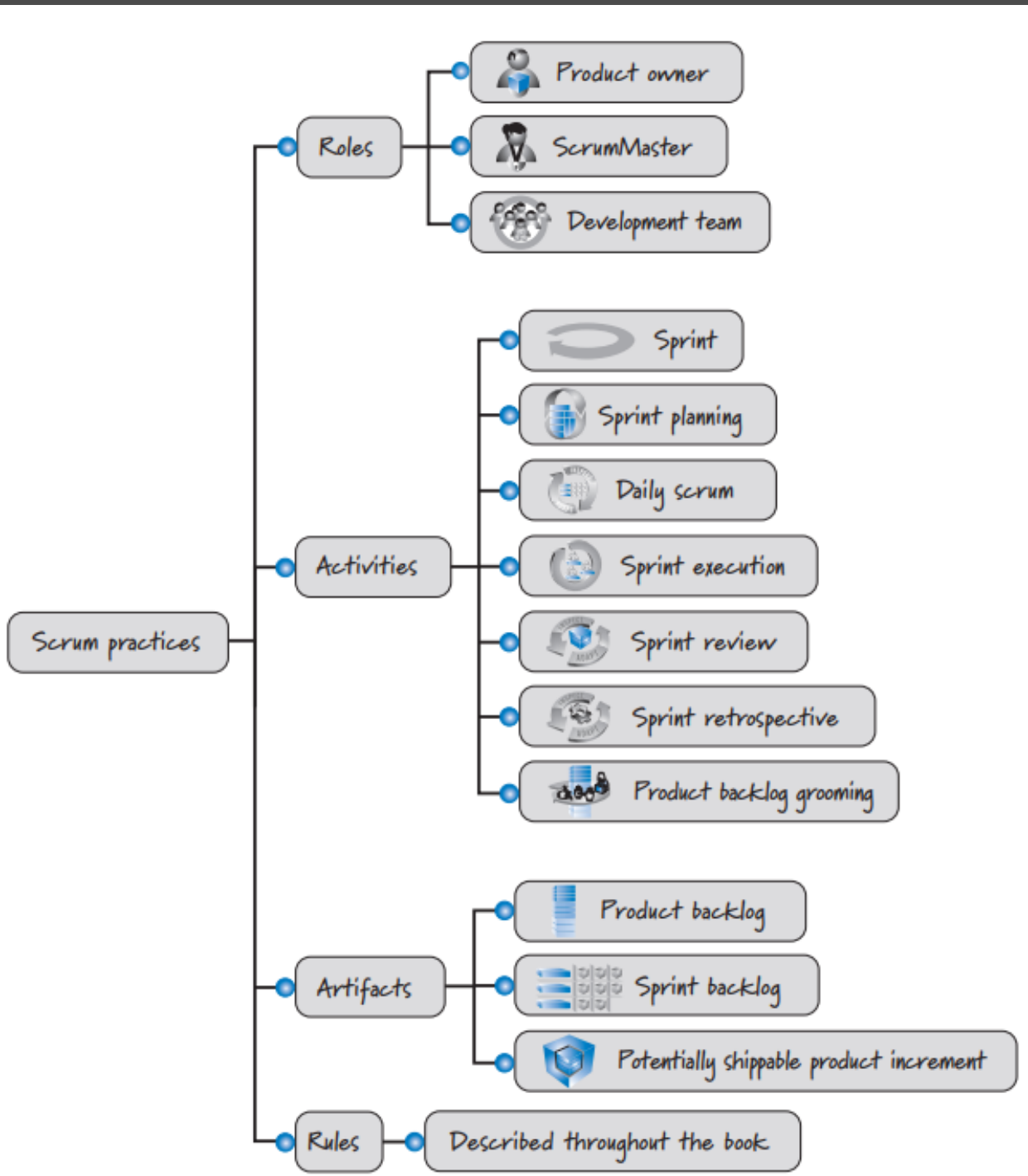


No desenvolvimento de sw, há um estilo para escrever as entradas do *backlog*, adotando o conceito de "*user story*". A história é um exemplo de utilização, ~uma forma de percorrer um caso de utilização.

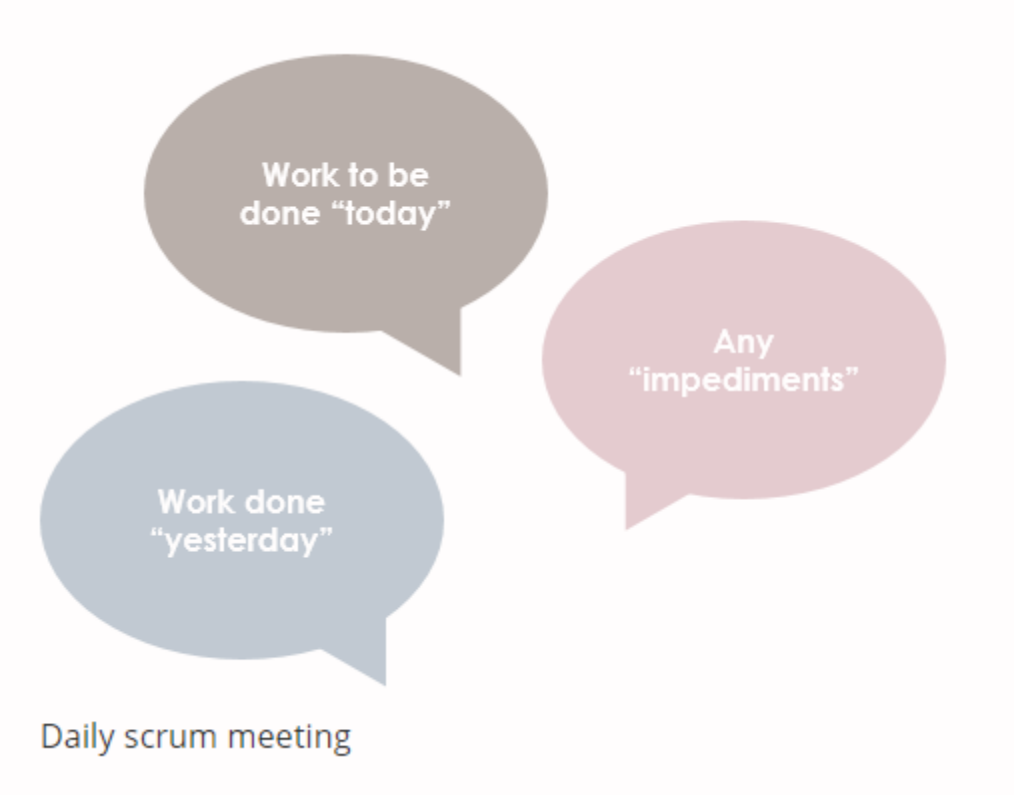
Histórias de utilização

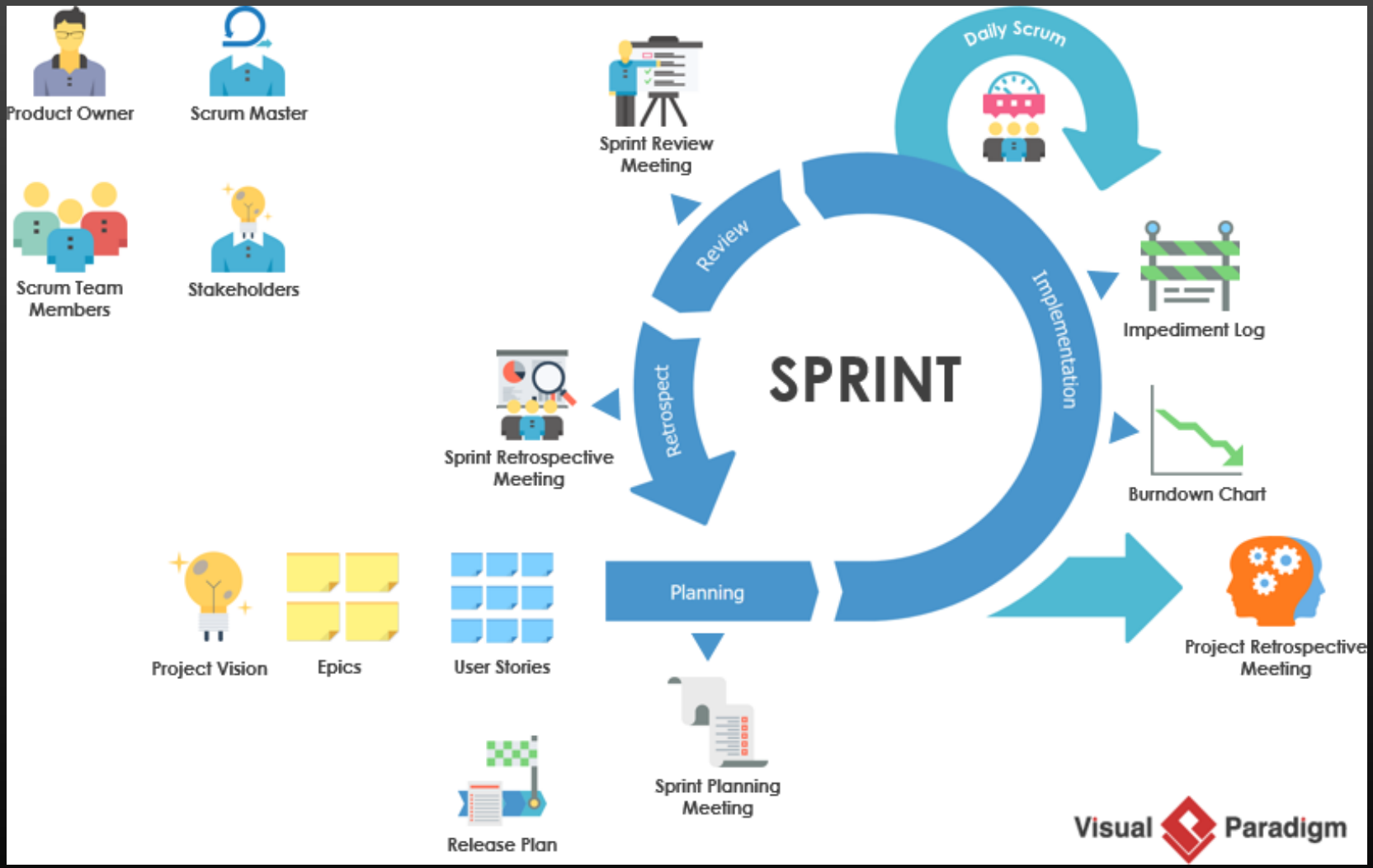


Elementos do Scrum



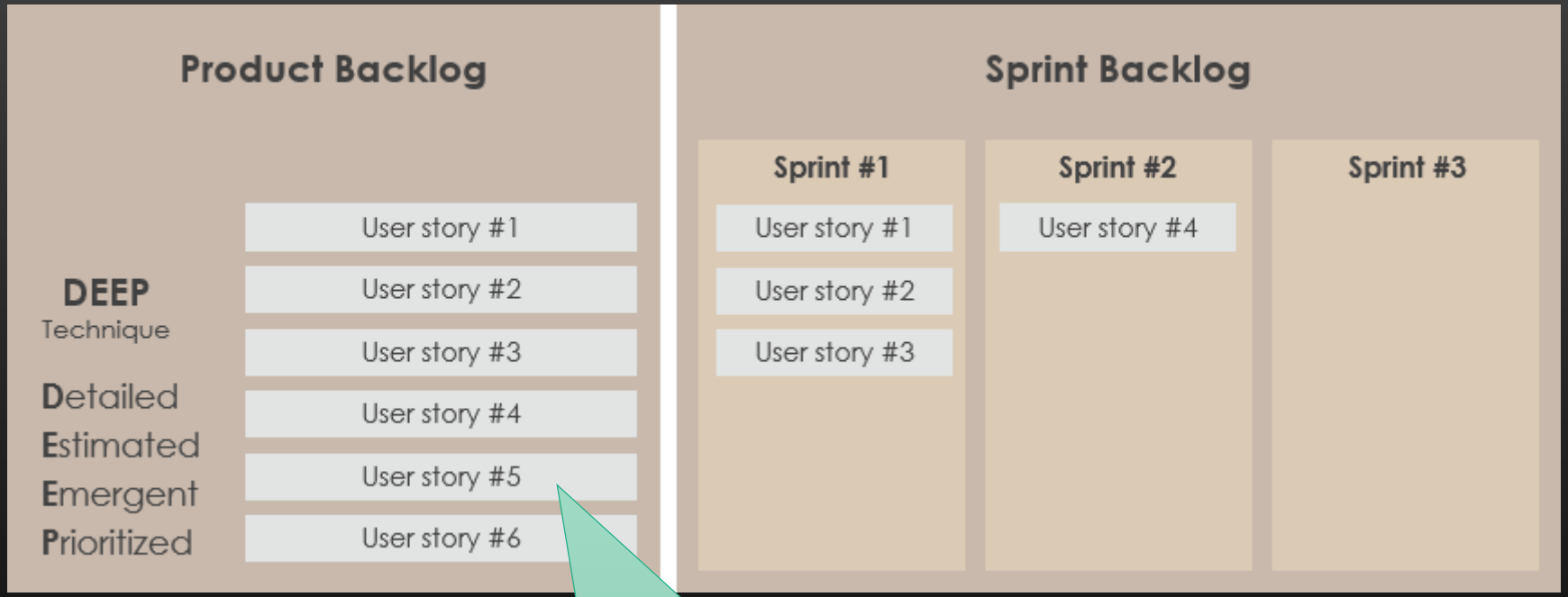
Daily Scrum


Work to be done “today”
Any “impediments”
Work done “yesterday”
Daily scrum meeting


Product Owner
Scrum Master
Scrum Team Members
Stakeholders
Sprint Review Meeting
Daily Scrum
Review
Implementation
Retrospect
SPRINT
Impediment Log
Burndown Chart
Sprint Retrospective Meeting
Project Vision
Epics
User Stories
Planning
Project Retrospective Meeting
Sprint Planning Meeting
Release Plan
Visual Paradigm

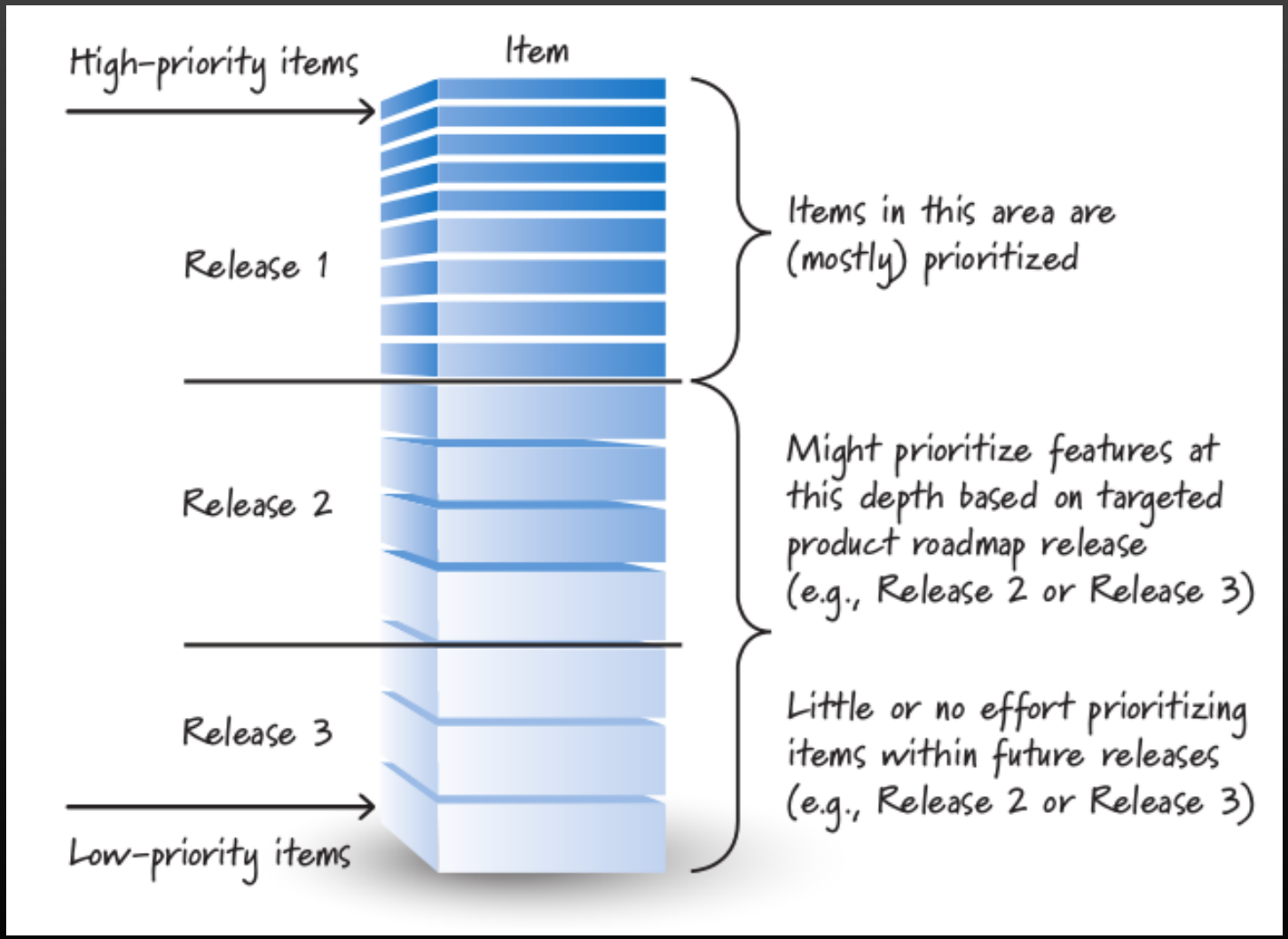
Planeamento e monitorização do progresso

Sprint planning

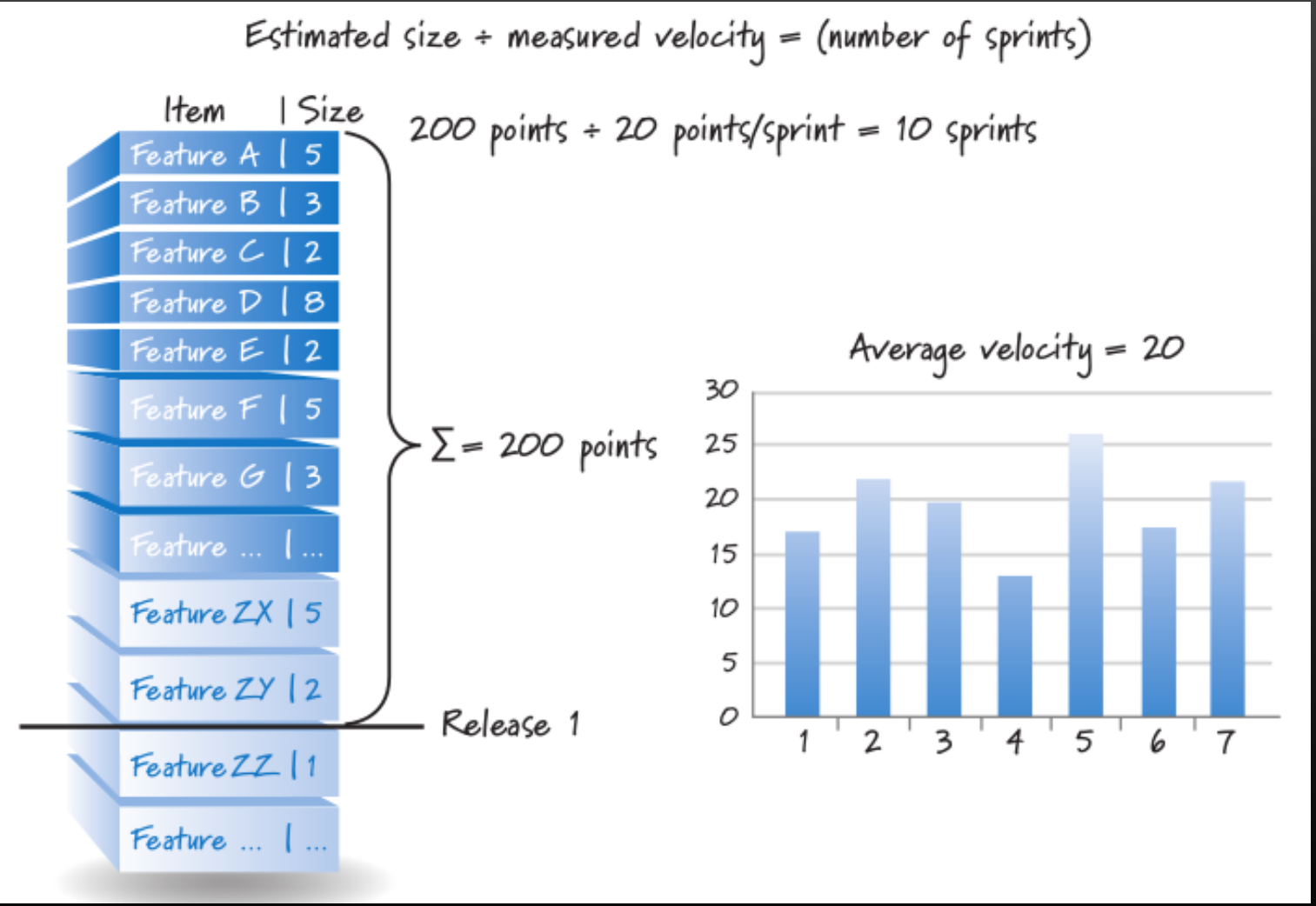


No desenvolvimento de sw, há um estilo para escrever as entradas do *backlog*, adotando o conceito de "*user story*". A história é um exemplo de utilização, ~uma forma de percorrer um caso de utilização.

Scrum: backlog must be prioritized



Scrum: Velocity



Pontuação

Funcionalidades (encomenda de comida online):

- F1: Inicio de sessão do utilizador (login)
- F2: Registo de novo utilizador na plataforma
- F3: Listar promoções em destaque do dia.
- F4: Colocar a encomenda (inclui pagamento)

Escala:

1pt: muito fácil. Direto de se implementar e âmbito reduzido.

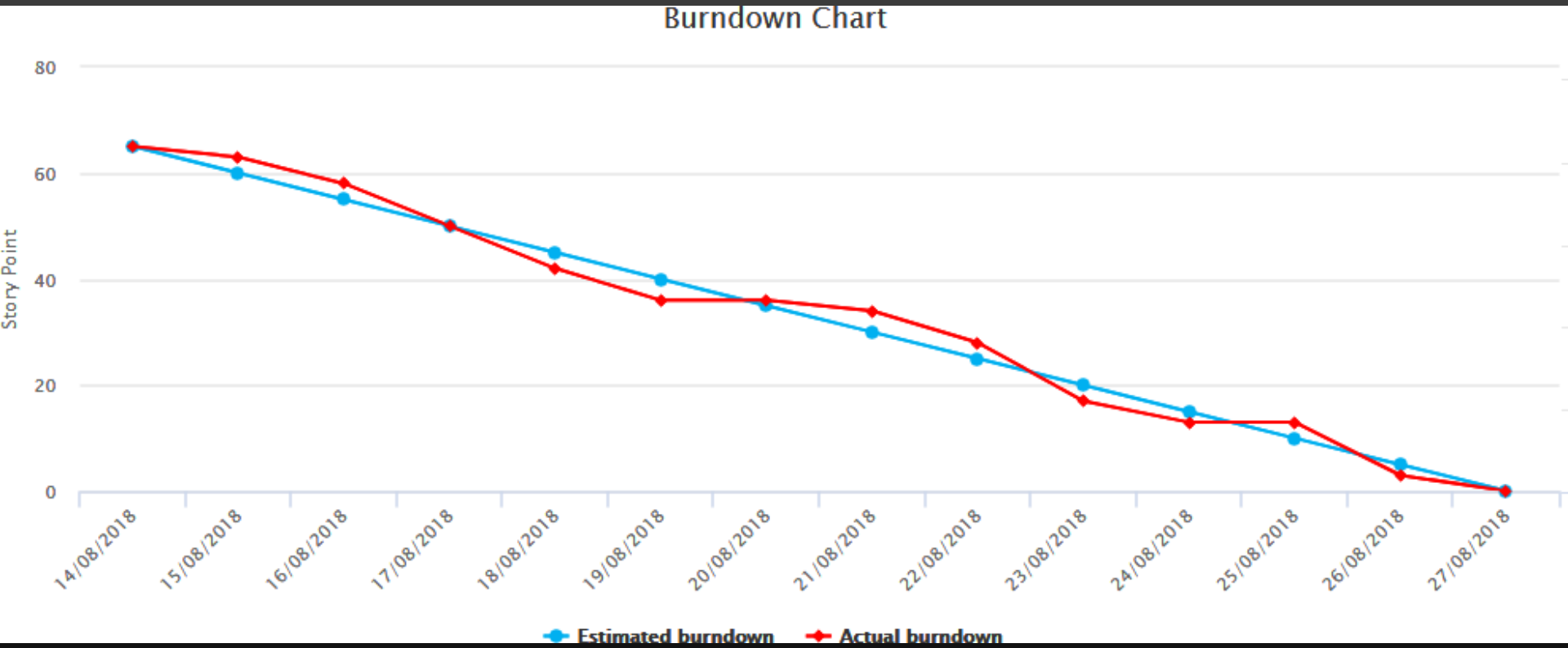
2pts: acessível; não oferece grande dificuldade.

4pts: complexo; tem várias interdependências (de outros módulos/serviços) ou um fluxo elaborado

8pts: muito complexo; requer integrações, tecnologias ou conhecimentos que não são completamente dominados

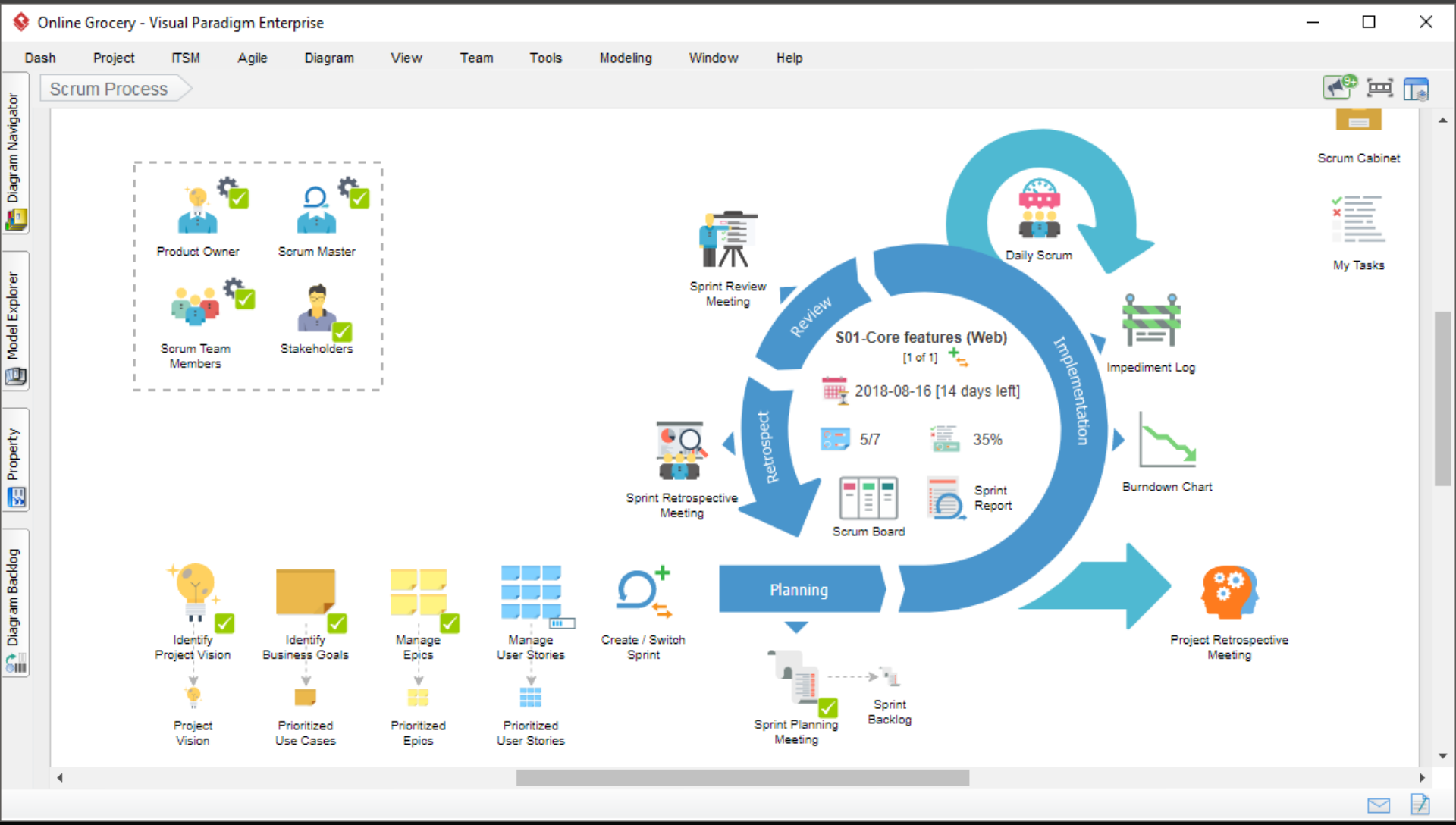
→ http://bit.ly/2IUrnMn

Monitorar o progresso com “burndown chart”



https://www.visual-paradigm.com/cn/scrum/scrum-burndown-chart/

[Optional] Scrum tutorial by VisualParadigm



https://www.visual-paradigm.com/tutorials/agile-tutorial/

Algumas ideias a reter

- Um processo de software explica o trabalho a desenvolver para construir o produto
- O processo não explica como organizer o dia-a-dia da equipa
- A Scrum oferece uma metodologia "leve" para gestão de equipas, a construir produtos complexos
- Mas... é desafiante dominar e aplicar a Scrum!

A Scrum é especialmente adequada para métodos ágeis de desenvolvimento de software

- Sprint (iteração)
- Equipa auto-organizadas e multifuncionais (comunicação)
- Foco no incremento (entrega frequente)
- Adaptação ("*embrance change*")

References

Core readings	Suggested readings
• Visual Paradigm, "What is Scrum?" • Ken Schwaber, Jeff Sutherland, "Scrum Guide". • [Dennis15] – Chap. 1	• [Pressman], Chap. 3 "Agile Development"