MÉTODOS ÁGEIS DE DESENVOLVIMENTO

MODELAÇÃO E ANÁLISE DE SISTEMAS | TP

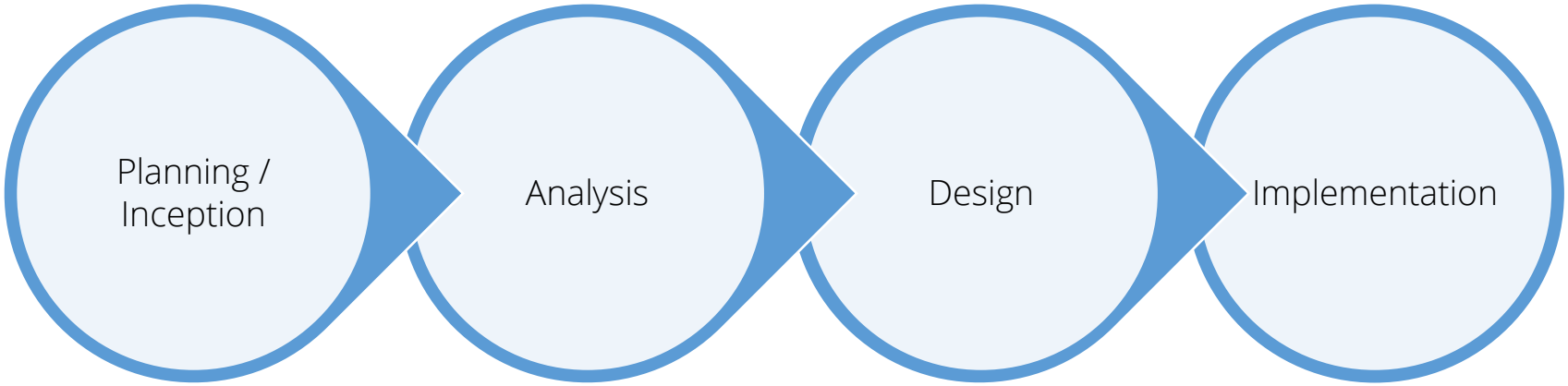
ILÍDIO OLIVEIRA ico@ua.pt
v2018-05-03

universidade de aveiro
departamento de eletrónica, telecomunicações e informática
deti

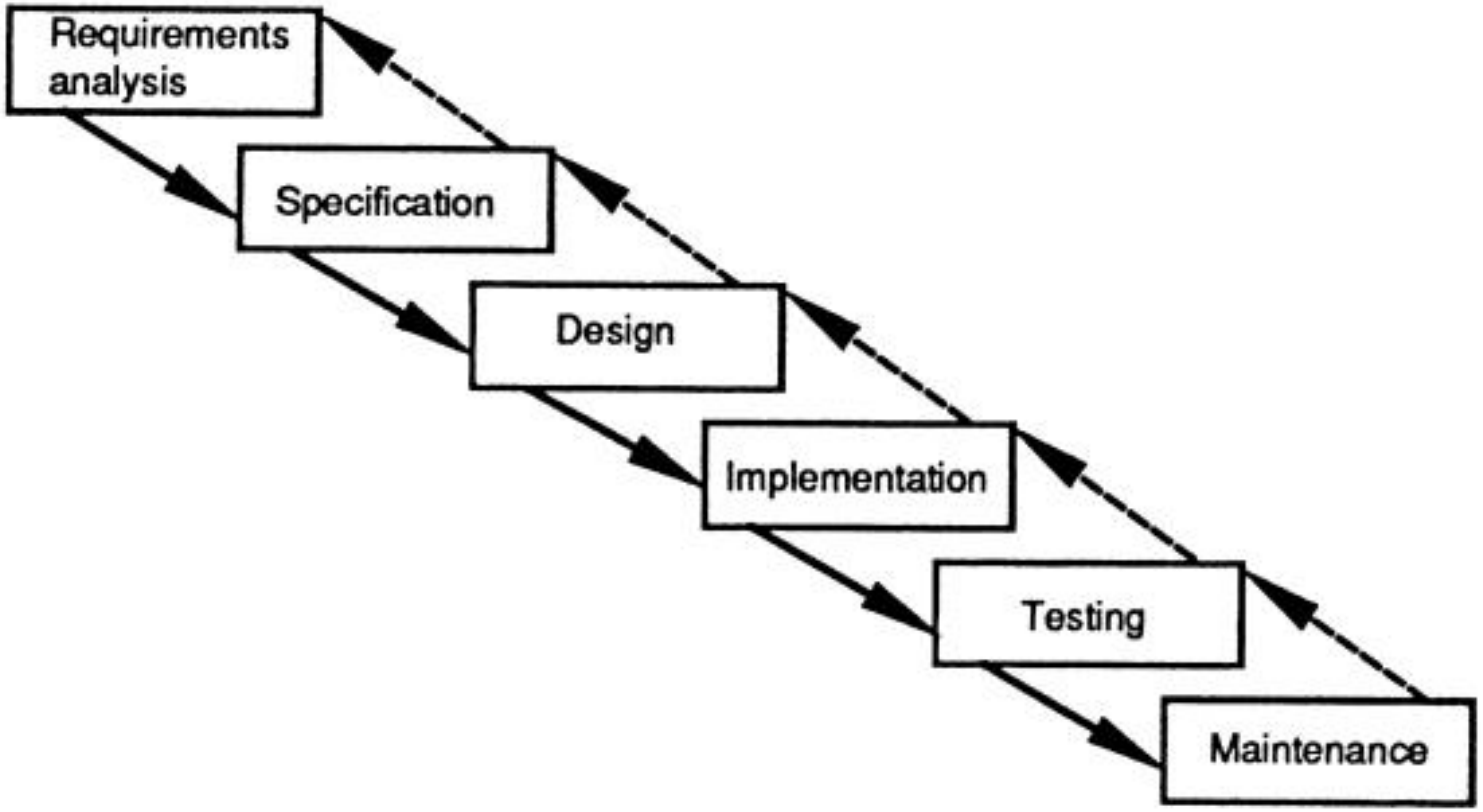
SDLC phases

Four fundamental phases: planning, analysis, design, and implementation. Different projects might approach the SDLC phases in different ways, but all projects have elements of these four phases.

Each phase is itself composed of a series of steps, which rely upon techniques that produce deliverables.



"Classical" engineering approach: **Waterfall model**



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

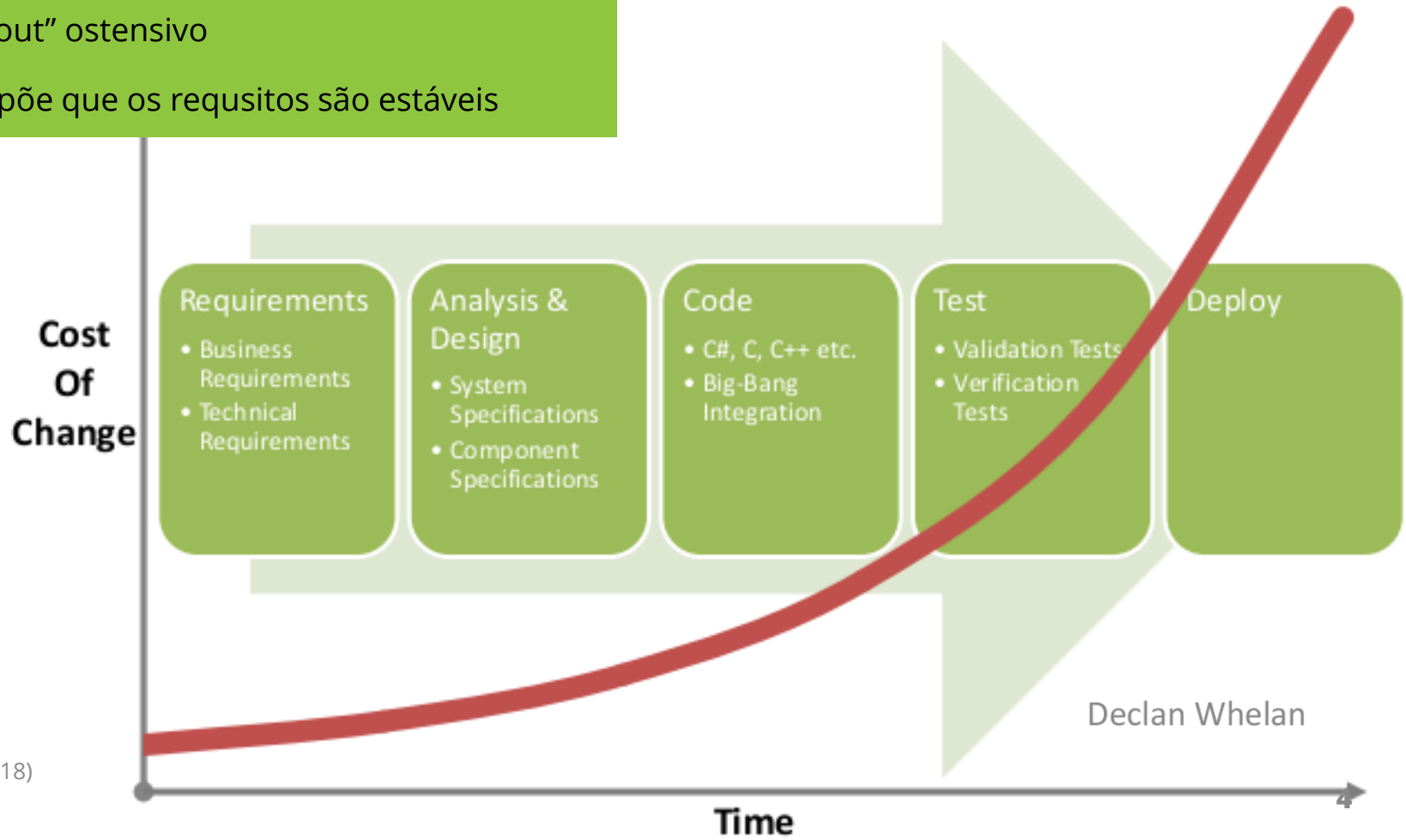
Problemas com a abordagem sequencial

Confirmação tardia de que os riscos estão controlados

Atividades de Integração e de Teste tardias

“Black out” ostensivo

Pressupõe que os requsitos são estáveis



“embrace change”

Rather than:

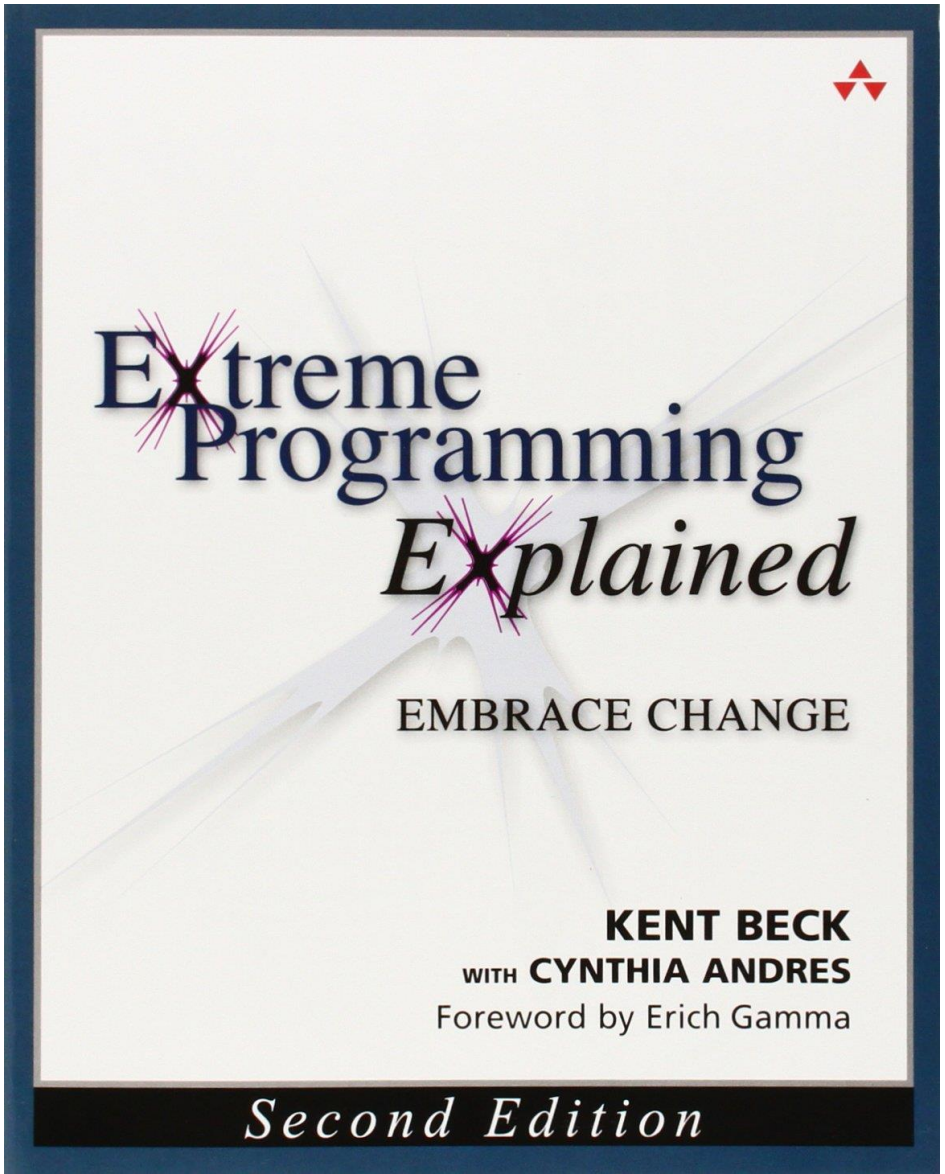
→ fighting the inevitable change that occurs in software development

→ by trying (unsuccessfully) to fully and correctly specify, freeze, and "sign off" on a frozen requirement set and design before implementation

iterative and evolutionary development :

→ is based on an attitude of embracing change and adaptation as unavoidable and indeed essential drivers.

→ this is not to say that iterative development encourage an uncontrolled and reactive process.



Abordagem “ágil”

Objetivo:

Resposta rápida à alteração de condições (e.g. alteração de requisitos)

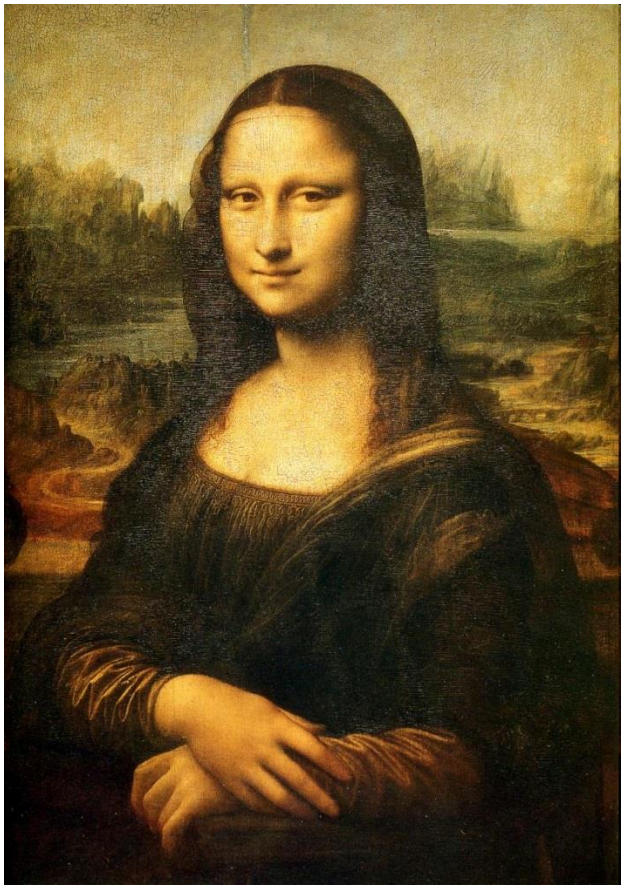
Para atingir a agilidade é necessário:

Práticas que equilibrem a disciplina e o feedback

- Ciclos curtos e entrega de valor frequente, integração em contínuo, desenvolvimento orientado por testes,...

Aplicar princípios de desenho que favorecem a construção de software flexível e evolutivo

Admitindo o objetivo...



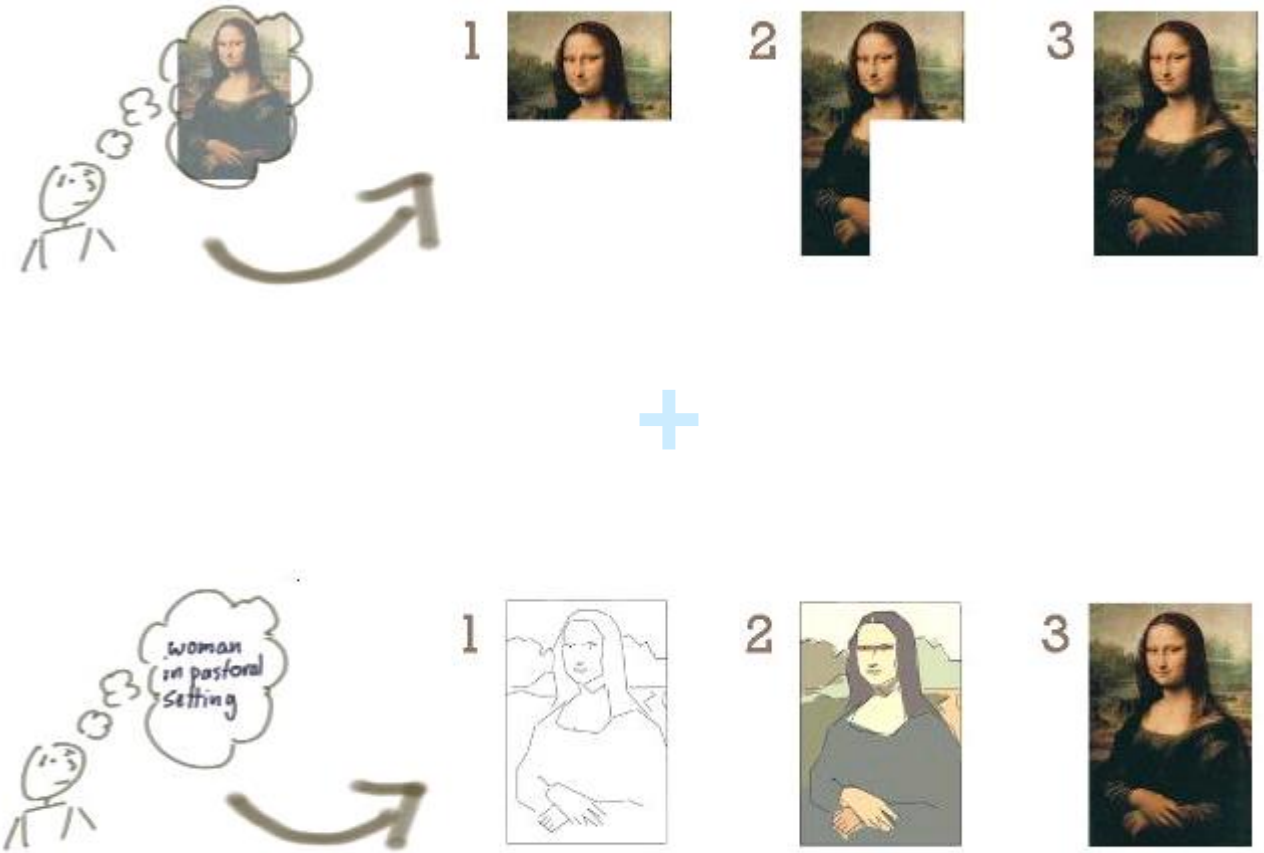
Abordagem incremental

http://jan-so.blogspot.com/2008/01/difference-between-iterative-and.html

Abordagem iterativa

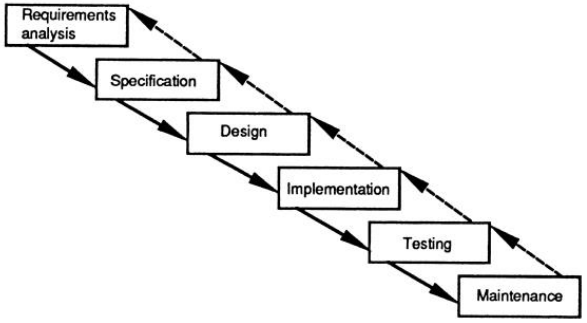
I Oliveira (2018)

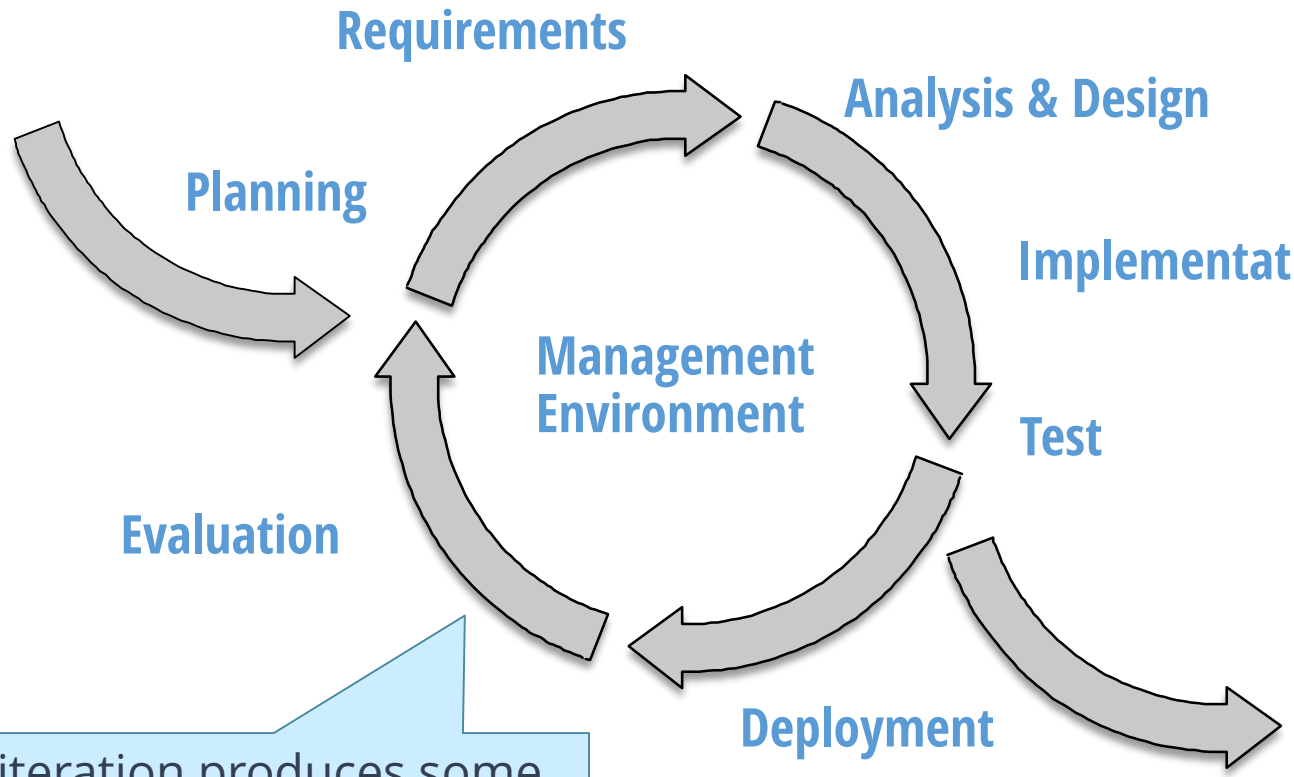
Abordagem incremental e iterativa



I Oliveira (2018)

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



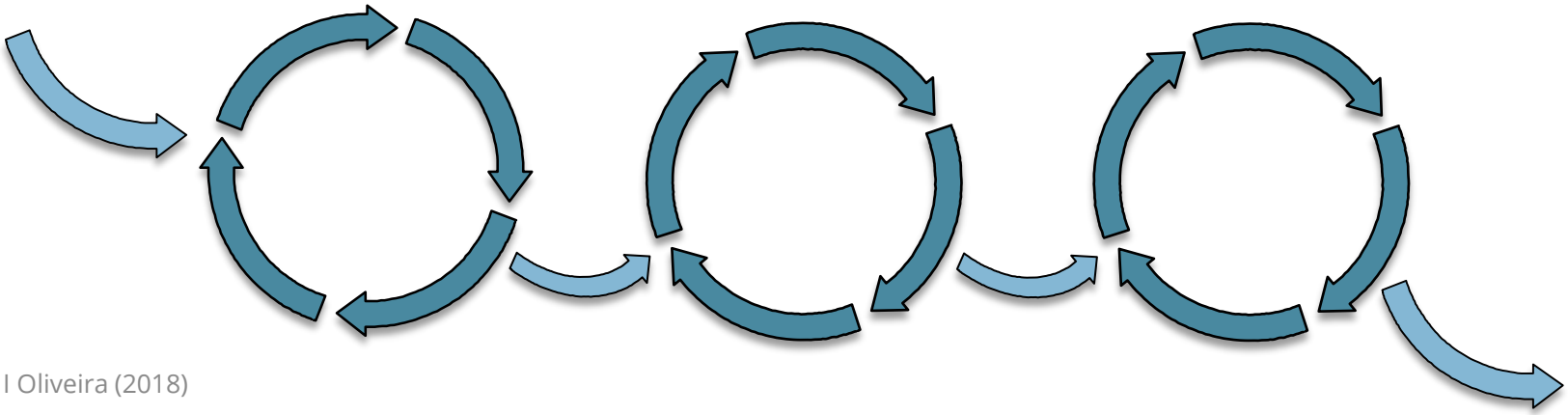


Each iteration produces some executable result

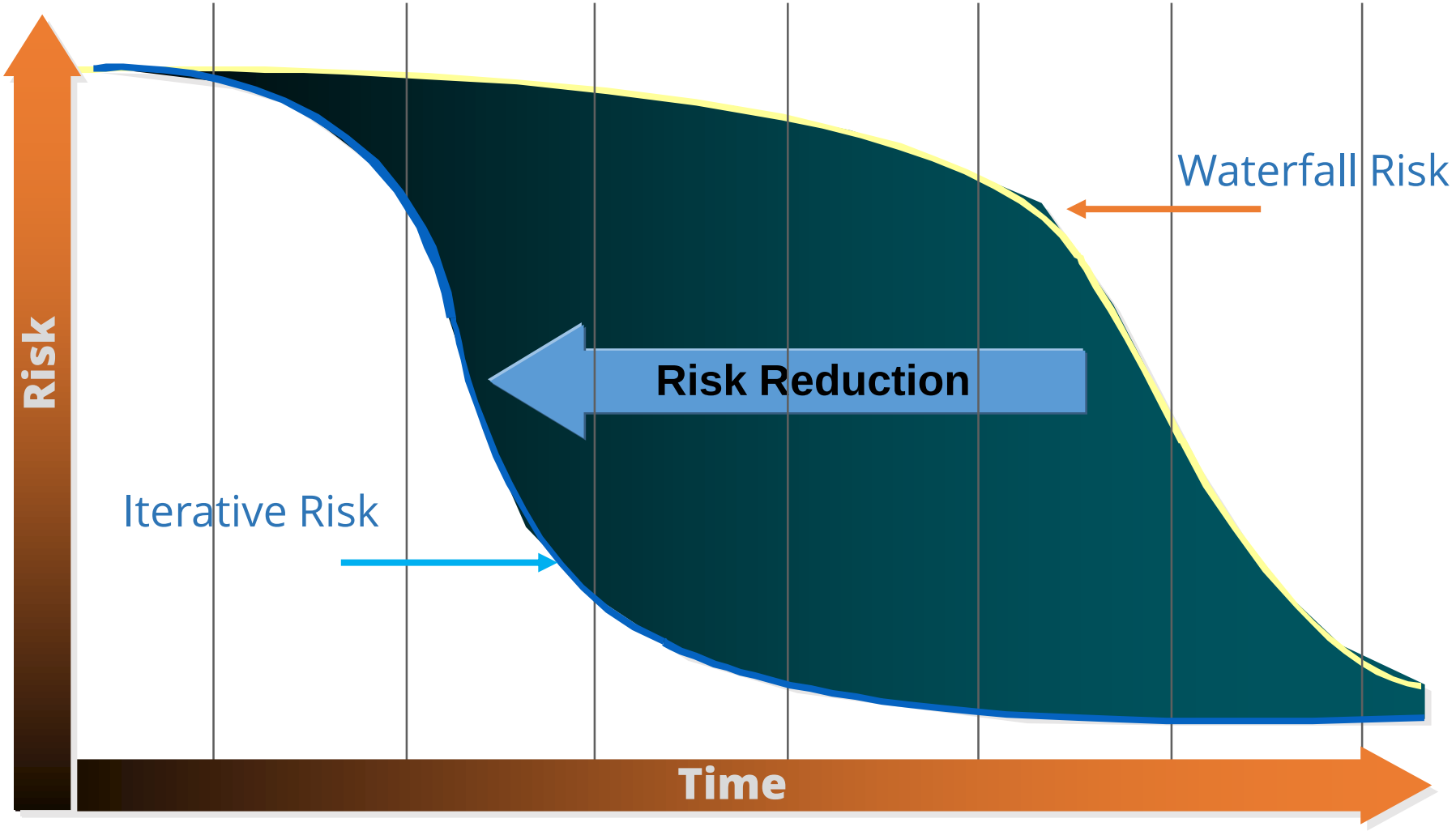
Iterative development

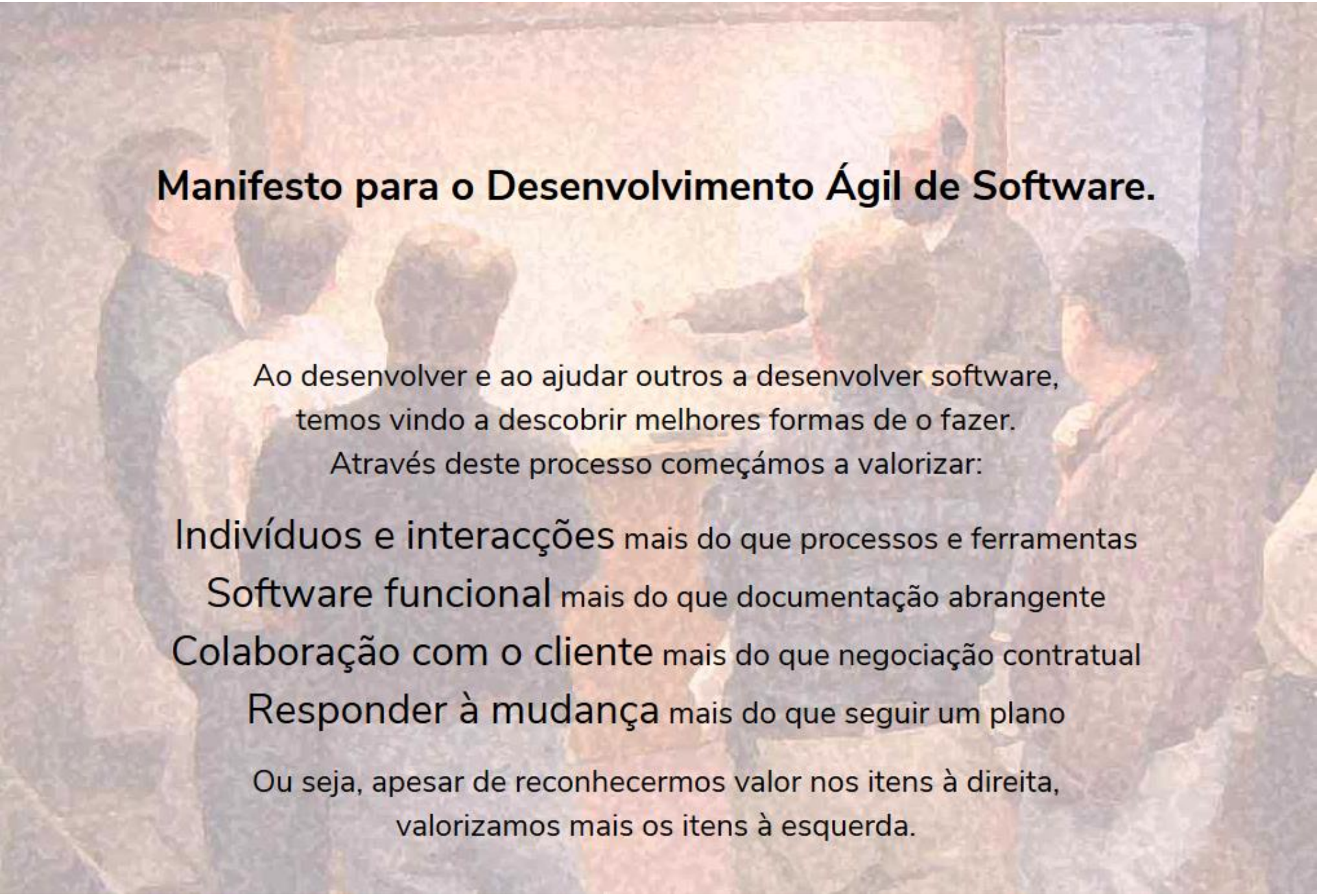
Each iteration involves choosing a small subset of the requirements, and quickly designing, implementing, and testing.

- Development in short cycles
- Each one is tested and integrated
- Each one gives an executable (partial) increment
- **Feedback** from each iteration leads to refinement and adaptation of the next.



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





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

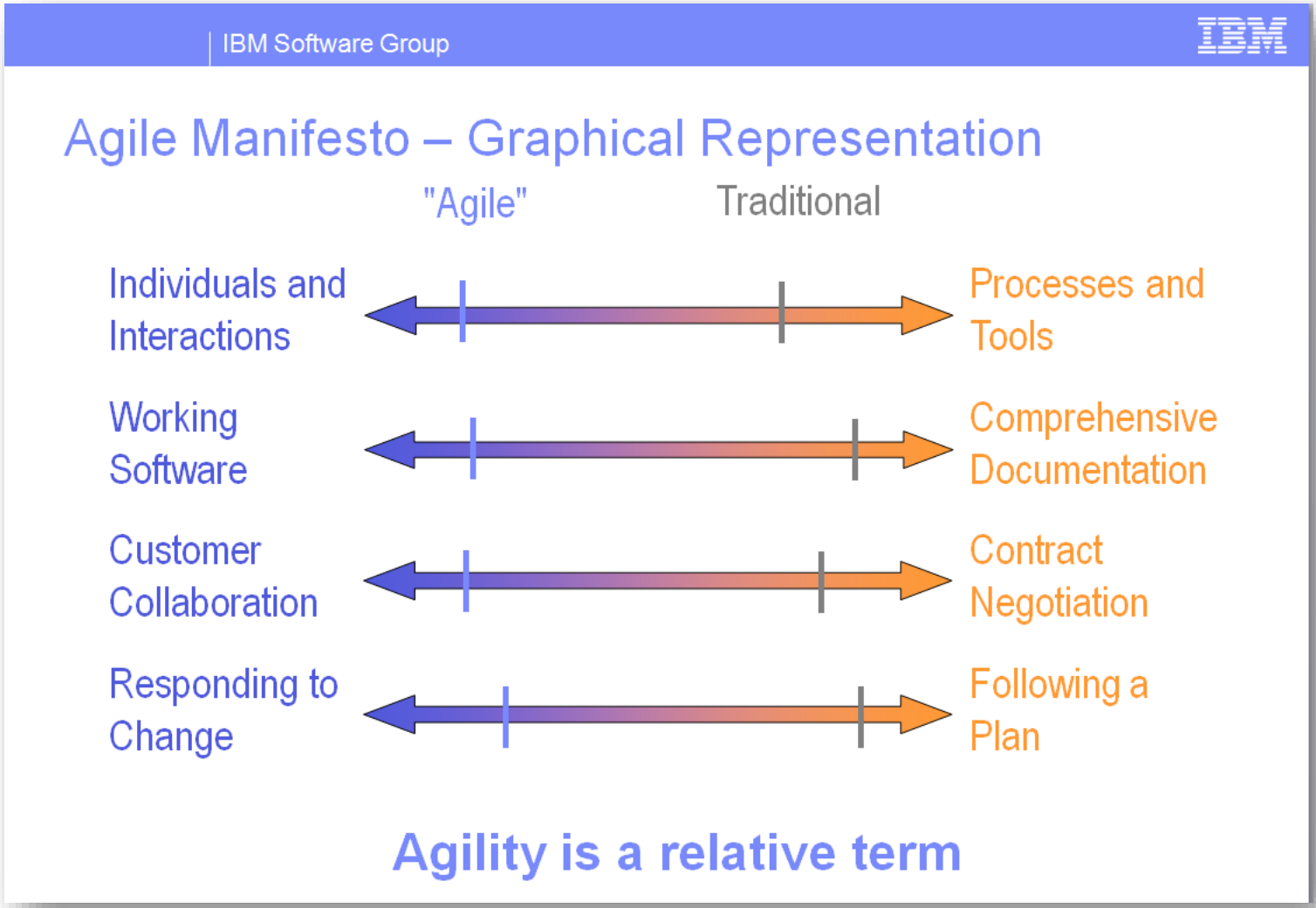
Colaboração com o cliente mais do que negociação contratual

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.

http://agilemanifesto.org/

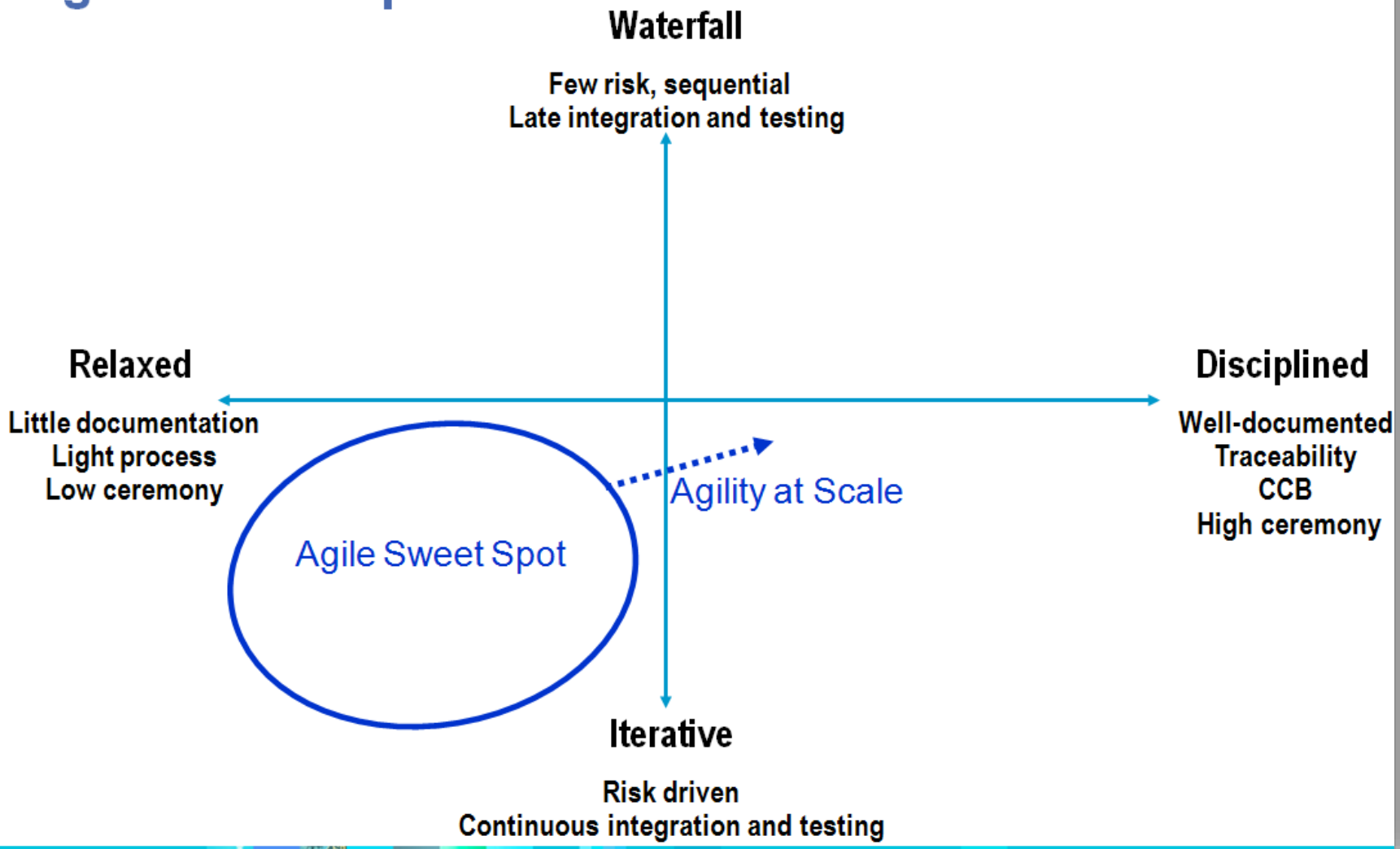
O desenvolvimento ágil de software



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 tardia 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.

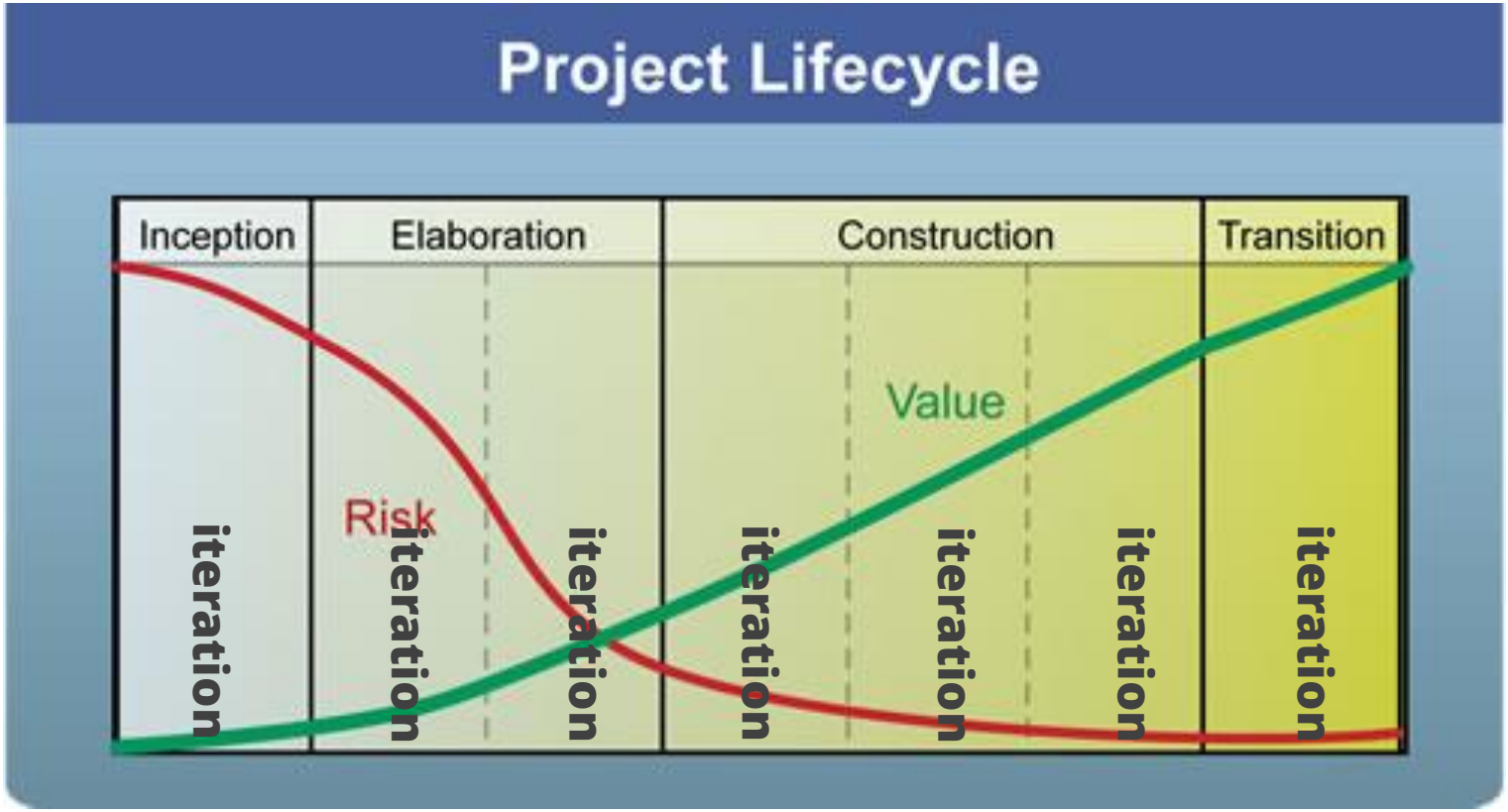
10. **Simplicidade** – a arte de maximizar a quantidade de trabalho que não é feito – é essencial.

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/

Estrutura do **Unified Process**



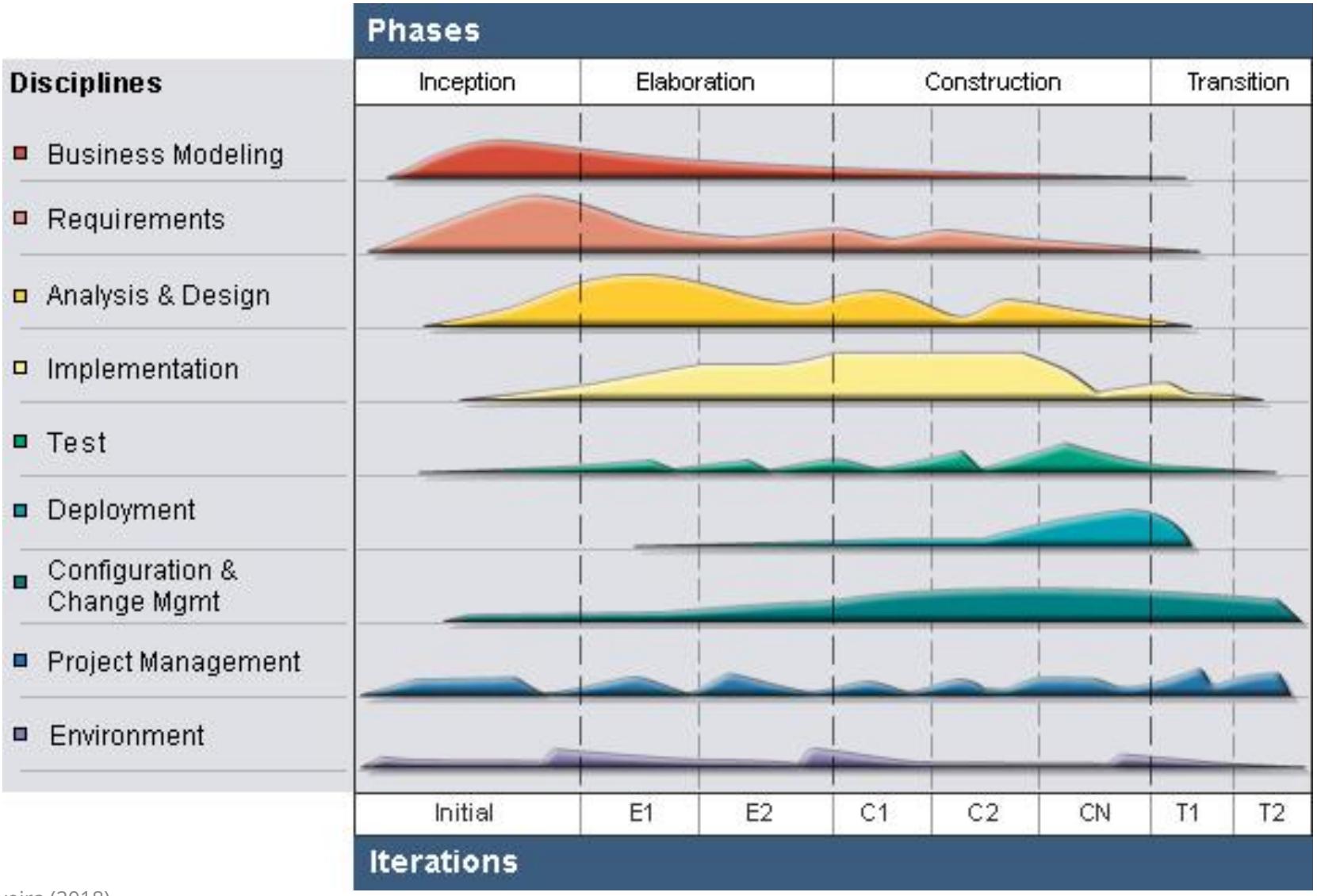
Unified Process approach

Iterative and Incremental

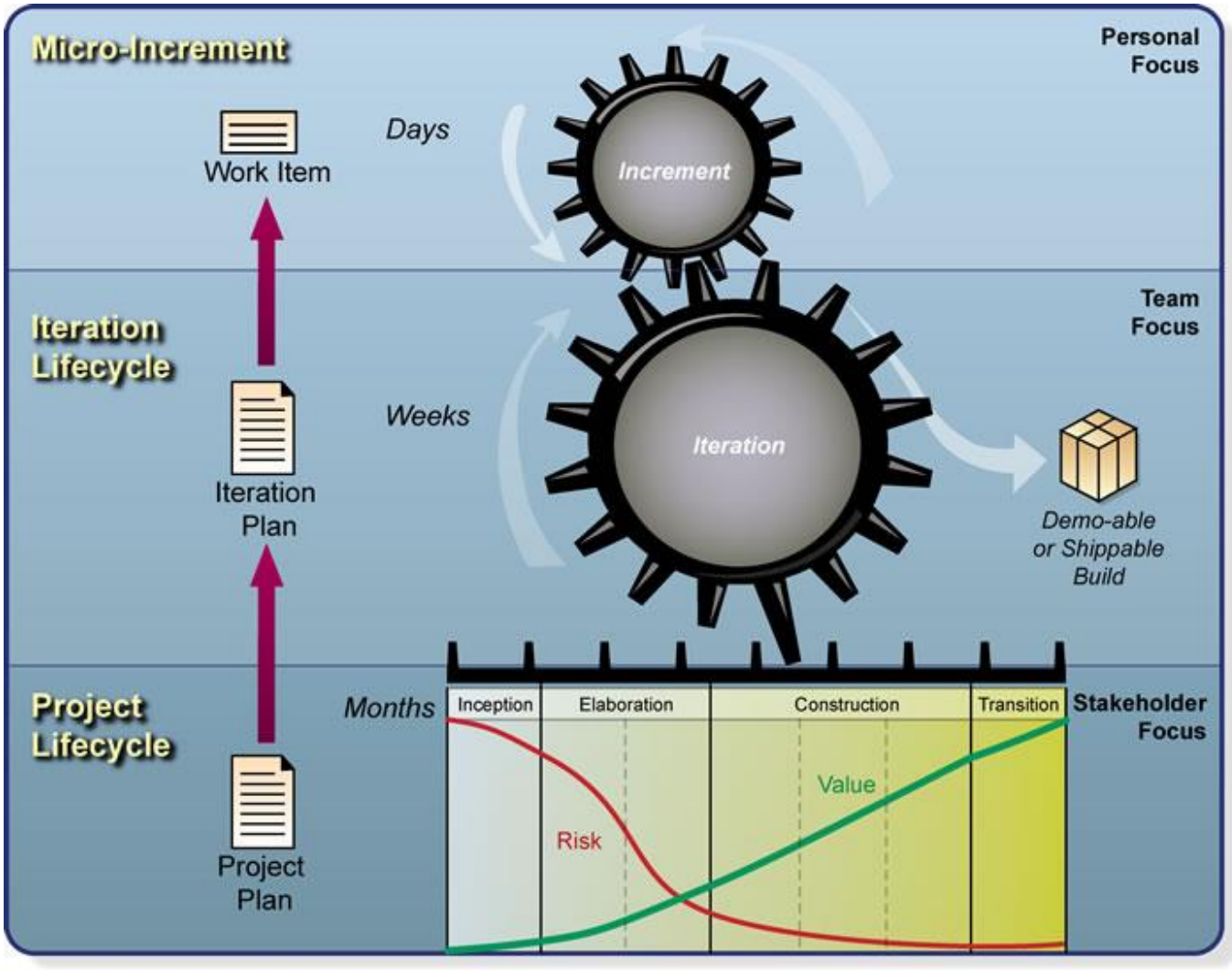
Use Case Driven

Architecture Centric

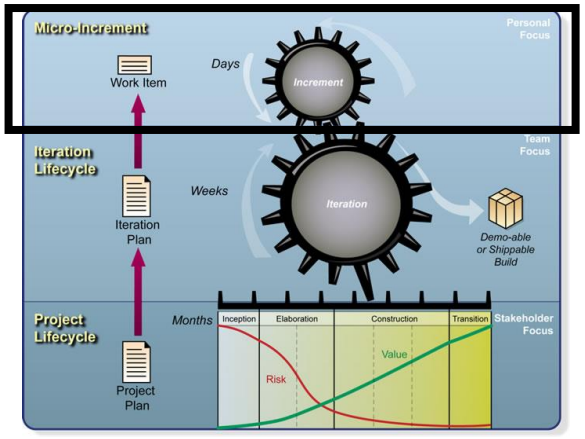
Ciclo de vida do Unified Process



Agile example: **Open Unified Process (OpenUP)**



Evolving Applications through Micro-Increments



Each micro-increment

corresponds to 0.5-3 days person days of work

is added (and removed if necessary) to the build

needs to be properly tested

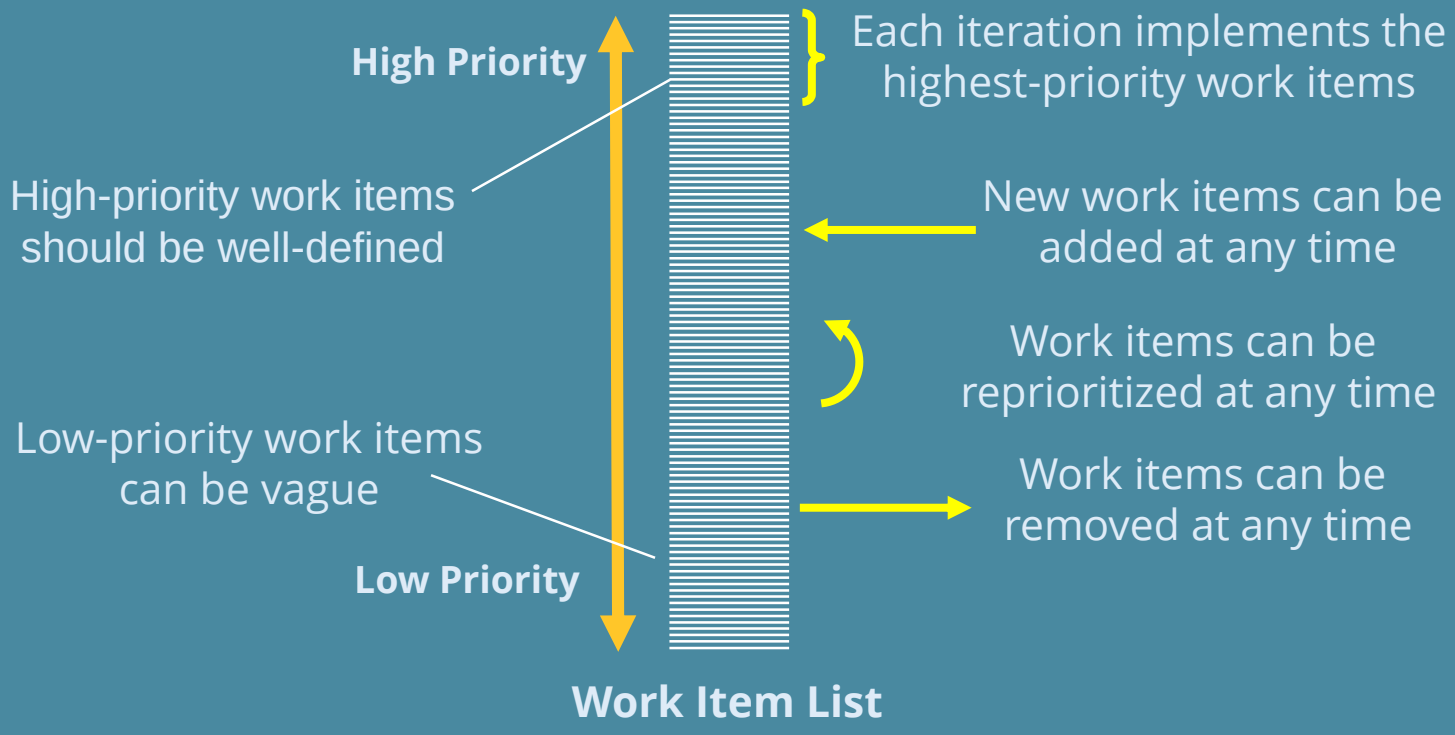
Needs to looked upon (requirements, design, implementation and testing) from a user-value perspective (use-case driven)

A month-long iteration with 20 developers would consists of ~200 micro-increments

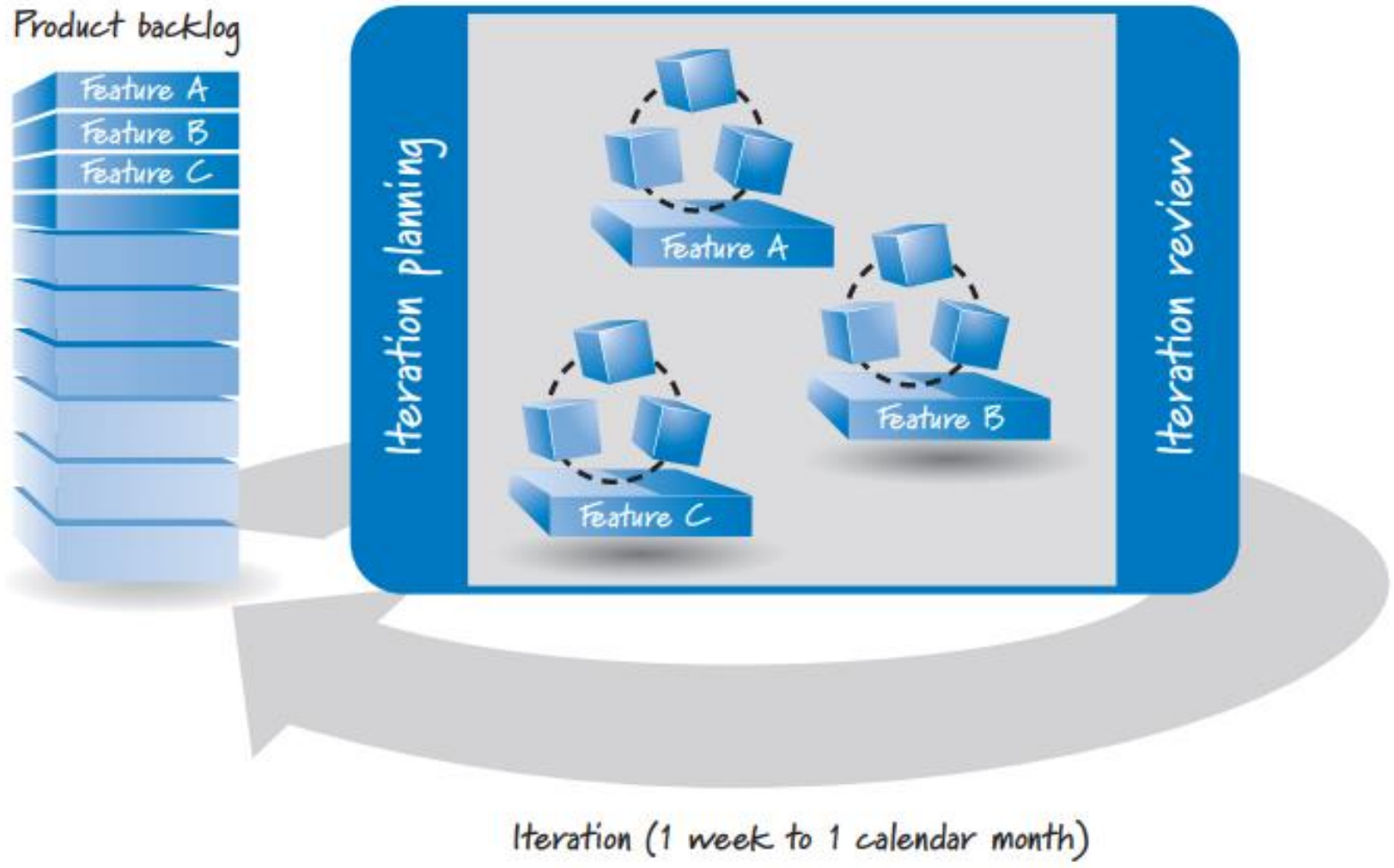
Micro-increments provides the team with the ability to manage and demonstrate continuous progress

Micro-increments applies to any type of project activity

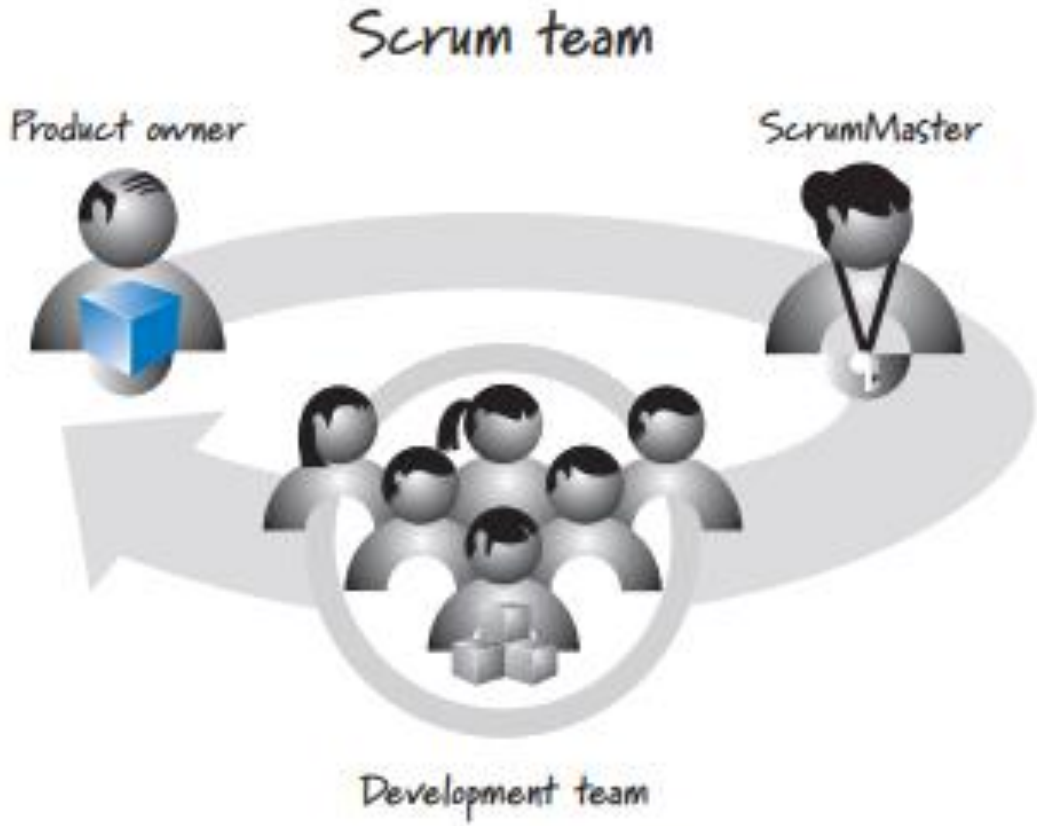
Managing Micro-Increments: **Work Items List** (a.k.a. backlog)



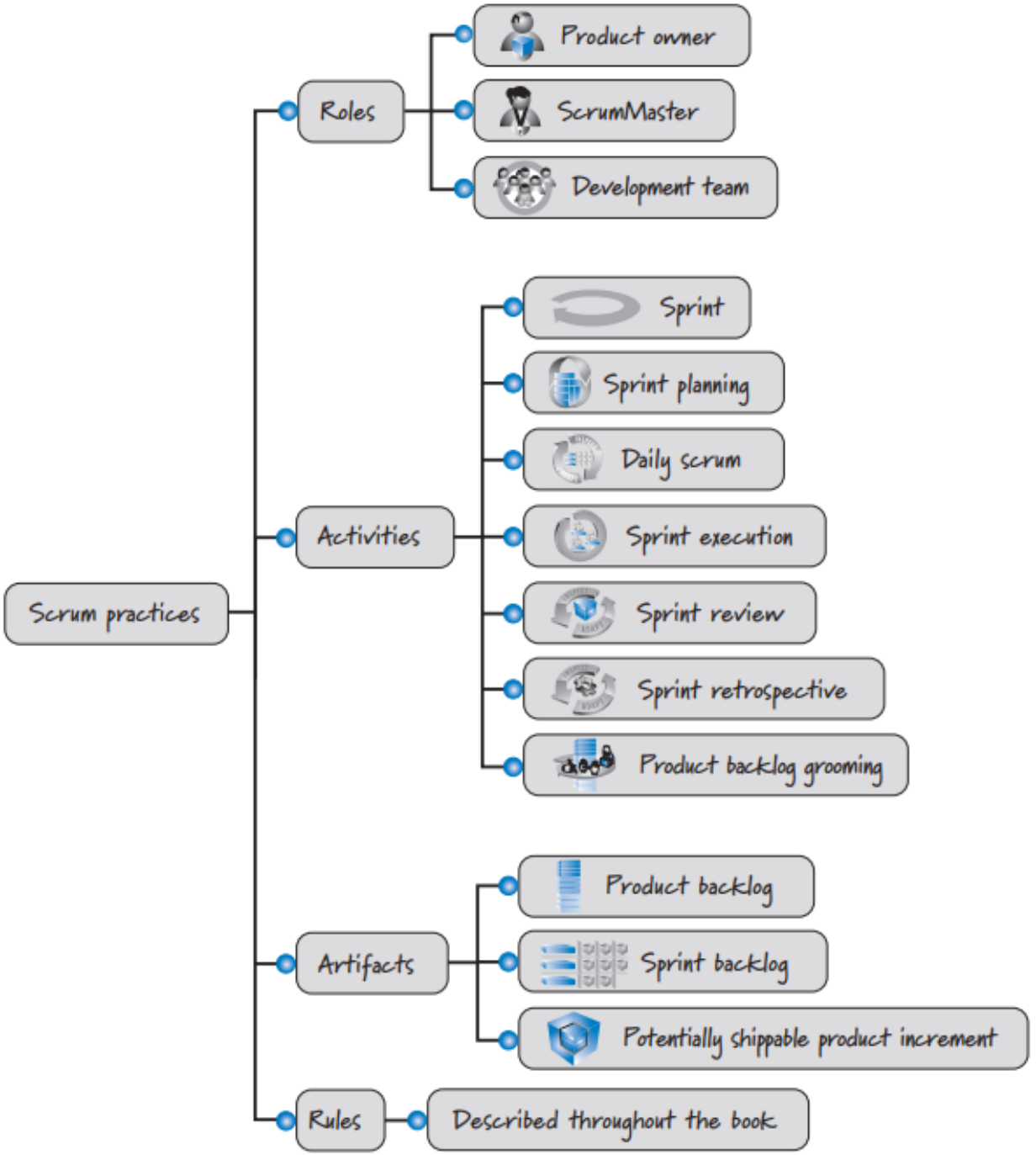
Planeamento do trabalho e métodos ágeis



Papéis previstos no Scrum



Elementos do Scrum



Scrum process

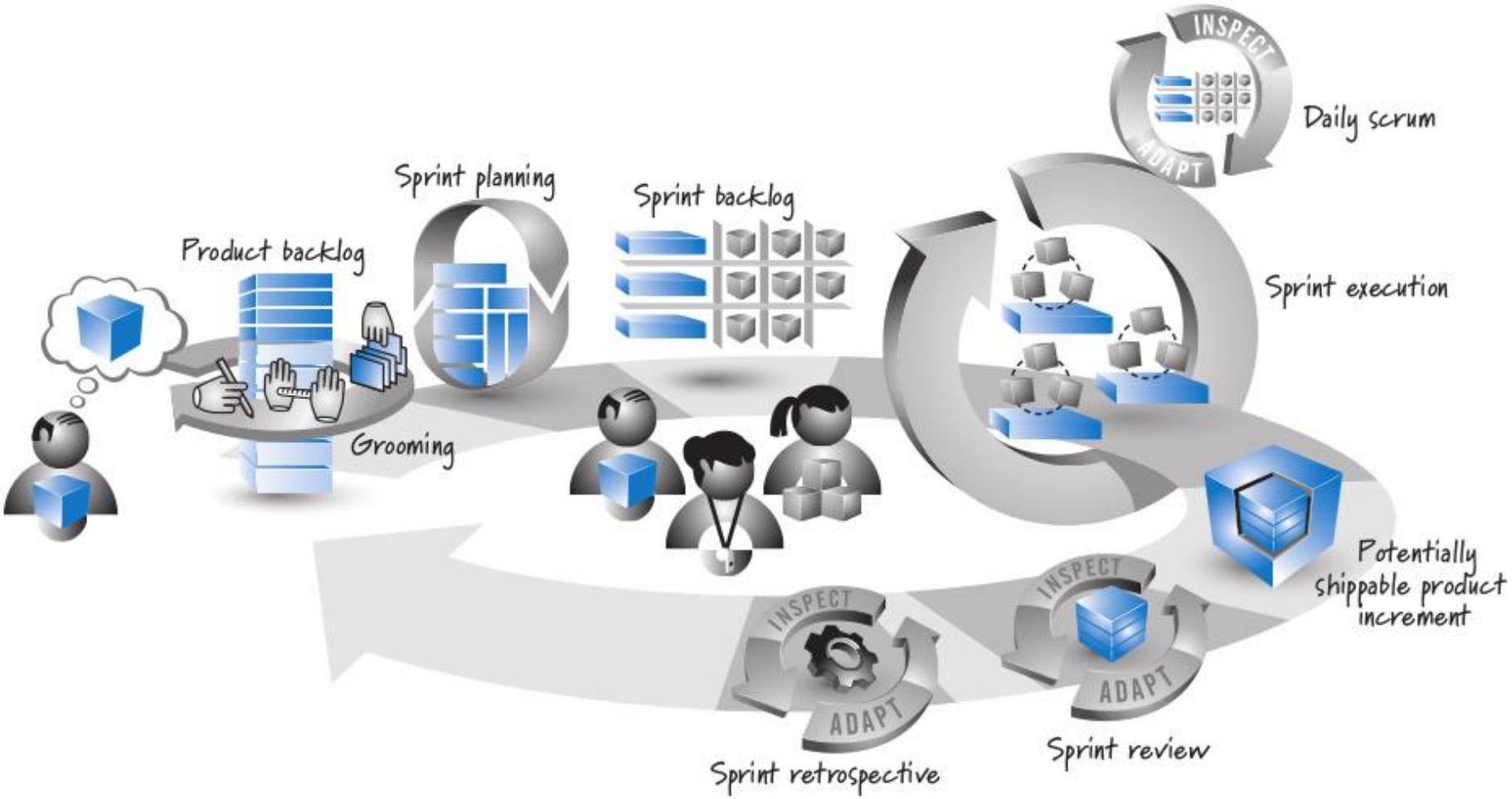
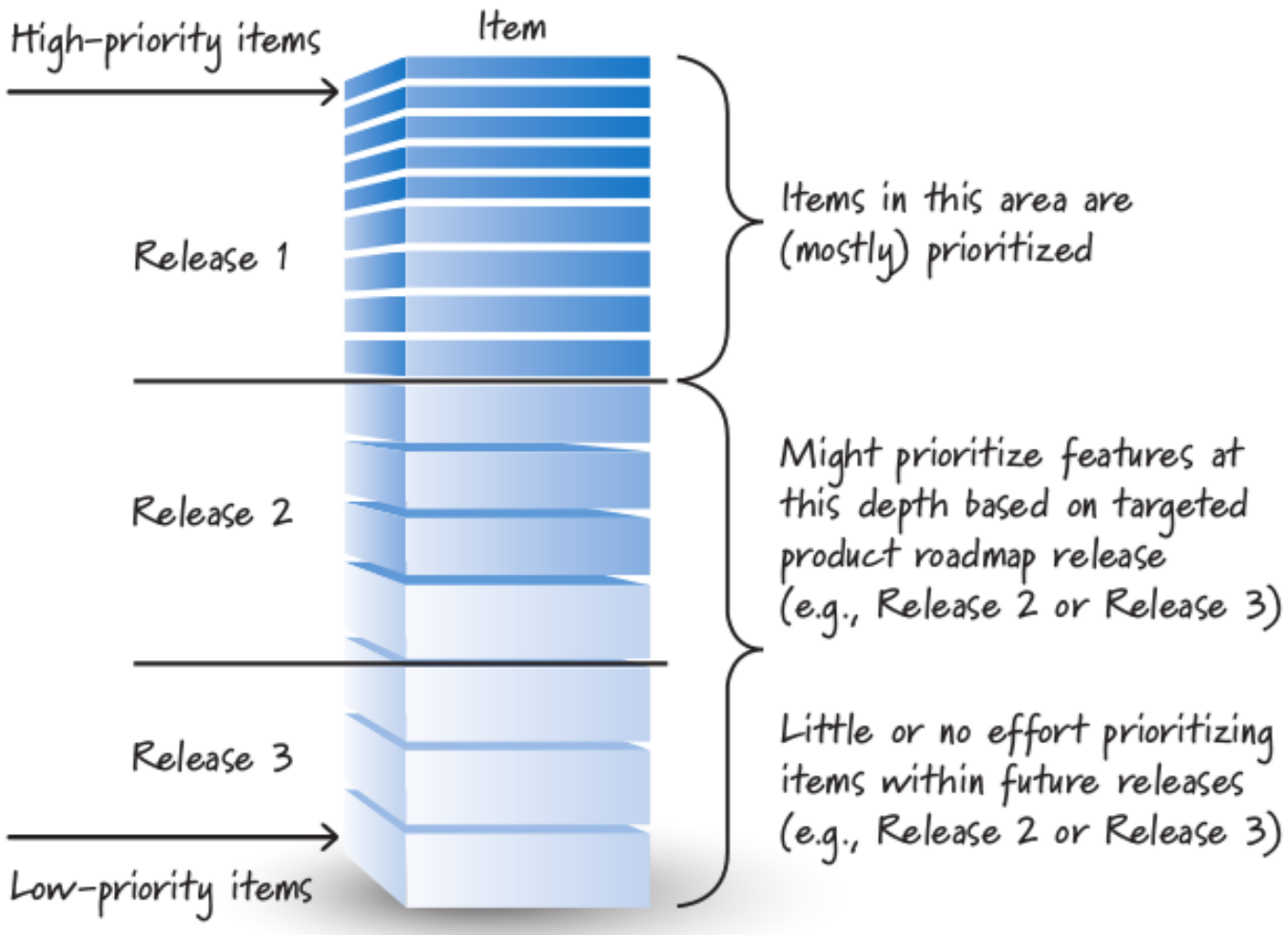
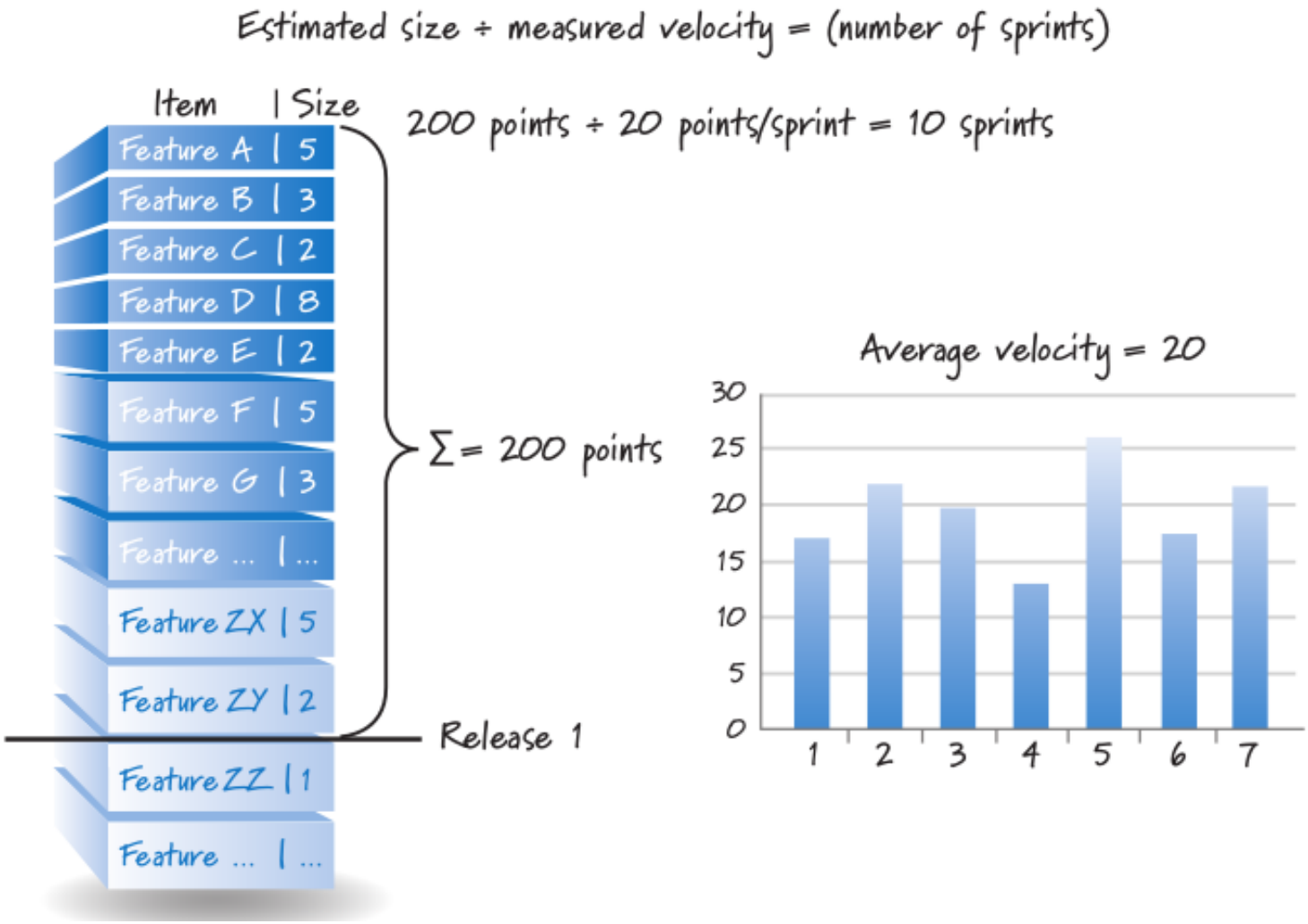


FIGURE 2.3 Scrum framework

Scrum: backlog must be prioritized



Scrum: Velocity



Referências

[LAR'12] Larman, C. (2012). *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Interative Development*. Pearson Education.

→ chap. 2

→ Rubin, Essential Scrum