ACH2028
Qualidade de Software

Aula 03 - Processos de Software

Prof. Marcelo Medeiros Eler
marceloeler@usp.br

Escola de Artes, Ciências e Humanidades - Universidade de São Paulo

Objetivos da aula

- Apresentar uma visão geral de um processo de desenvolvimento de software
- Apresentar uma visão geral de dois tipos de processos de desenvolvimento de software
  - Processos Tradicionais (Preditivos)
  - Métodos Ágeis (Adaptativos)

Crise do Software

Refere-se a uma série de problemas encontrados no desenvolvimento de software:

- Estimativas (de esforço, prazo e custo) imprecisas
- Insatisfação do cliente
- Software de baixa qualidade
- Difícil manutenção
- Gerência de projetos ineficiente
- Processo de software ineficiente

Um pouco de história

1968:

- A “NATO Science Committee” organizou uma conferência para discutir os problemas da crise do software e propor soluções
- Foi “criada” a Engenharia de Software como uma tentativa de dar um tratamento de engenharia (mais sistemático, controlado e de qualidade mensurável) ao desenvolvimento de sistemas de software complexos.

Um pouco de história

1968:

- Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968, Brussels, Scientific Affairs Division, NATOJanuary 1969. http://homepages.cs.ncl.ac.uk/brian.randell/NATO/index.html

Software Engineering

COINS III

VOLUME I

Proceedings of the Third Symposium on Computer and Information Sciences held in Miami Beach, Florida, December, 1969

Edited by

JULIUS T. TOU

Center for Informatics Research
University of Florida
Gainesville, Florida

Academic Press New York · London · 1970

Software Engineering—A New Profession

Julius T. Tou
UNIVERSITY OF FLORIDA
GAINESVILLE, FLORIDA

Throughout history engineering has continually interacted with the rest of human society in a highly determinative and influential way. It has largely freed man from physical drudgery and from the slavery of routine mental tasks. It has provided the tools of new scientific discovery as well as those of new creative art. It has provided the means of bringing to mankind the results of the creative efforts of the great artists, writers, scientists, and philosophers of both ancient and modern times. It has provided the facilities for man to communicate with his fellowmen throughout the world. It has placed men on the moon, and is now concerned with the problem of information explosion. The engineer continually seeks new imaginative solutions to many of the very real problems that confront mankind.

During the last decade, engineering has undergone a continuous transition from the practical to the theoretical. This change had made technology overlap with science. Engineering education today is quite different from engineering education twenty years ago; it bears little resemblance to engineering education at the turn of this century. Perhaps we shall not be able to recognize engineering by the year 2000. Old branches are either updated or phased out, and new branches are created out of necessity. The newest offspring is software engineering which is the theme of this volume. For decades engineering drawing was a subject required by all engineering colleges. Today, it is being replaced in many schools by digital computation and programming as a basic requirement.

Engenharia de Software

- Definição (IEEE):
  - “A aplicação de uma abordagem sistemática, disciplinada e possível de ser medida para o desenvolvimento, operação e manutenção do software”
- Objetivo inicial (tradicional):
  - Construir um software como qualquer outro produto da engenharia tradicional (civil, mecânica, aeronáutica, etc)

Engenharia de Software

Um dos benefícios da criação da Engenharia de Software foi a definição de processos de desenvolvimento de software

Neste contexto, dois tipos de processo se destacam:

- Processos tradicionais (preditivos)
- Métodos ágeis (adaptativos)

Processo de desenvolvimento de software

Antes de diferenciar os dois tipos de abordagem, é importante compreender, mesmo que de forma geral, as atividades de um processo de desenvolvimento de software.

Processo de desenvolvimento de software

Refere-se à abordagem/método/metodologia/filosofia de desenvolvimento de um software

Compreende o conjunto de atividades utilizado para desenvolver um software

Alguns objetivos de um processo de desenvolvimento são:

- Definir quais atividades devem ser executadas ao longo do projeto
- Definir quando, como e por quem tais atividades serão executadas
- Definir quais artefatos serão produzidos e usados em cada atividade
- Prover pontos de controle para verificar o andamento do desenvolvimento
- Padronizar a forma de desenvolver software em uma organização

Processo de desenvolvimento de software

Atividades técnicas

- Análise/Engenharia de Sistemas
- Análise/Engenharia de Requisitos (Especificação)
- Projeto/Arquitetura
- Codificação (Implementação)
- Teste (Validação)
- Manutenção (Evolução)

Atividades gerenciais

- Gerência de Mudanças
- Gerência de Configuração de Software
- Garantia de Qualidade de Software
- Gerência de Projetos

Engenharia de Software

Tipos de processos que se destacam:

- **Processos tradicionais (preditivos)**
- Métodos ágeis (adaptativos)

Processos tradicionais de desenvolvimento

Objetivo inicial (tradicional):

- Construir um software como qualquer outro produto da engenharia tradicional (civil, mecânica, aeronáutica, etc)

Processos tradicionais de desenvolvimento

Produtos da engenharia tradicional:

- Qual é o processo para construir um produto da engenharia tradicional (exemplo: casa, prédio ou ponte)?

Processos tradicionais de desenvolvimento

Produtos da engenharia tradicional:

- Qual é o processo para construir um produto da engenharia tradicional (exemplo: casa, prédio ou ponte)?
- Quais são as premissas de usar este tipo de abordagem para desenvolver um software?

Processos tradicionais de desenvolvimento

Premissas:

- É possível definir e especificar detalhadamente todos os requisitos de um software na fase de concepção
- Os requisitos do software permanecerão os mesmos ou serão pouco alterados durante todo o seu ciclo de desenvolvimento
- Os artefatos gerados pelas atividades de projeto são fundamentais para a codificação do software
- O cliente só usará o produto quando ele estiver totalmente finalizado

Processos tradicionais de desenvolvimento

Trouxeram disciplina para o desenvolvimento de software:

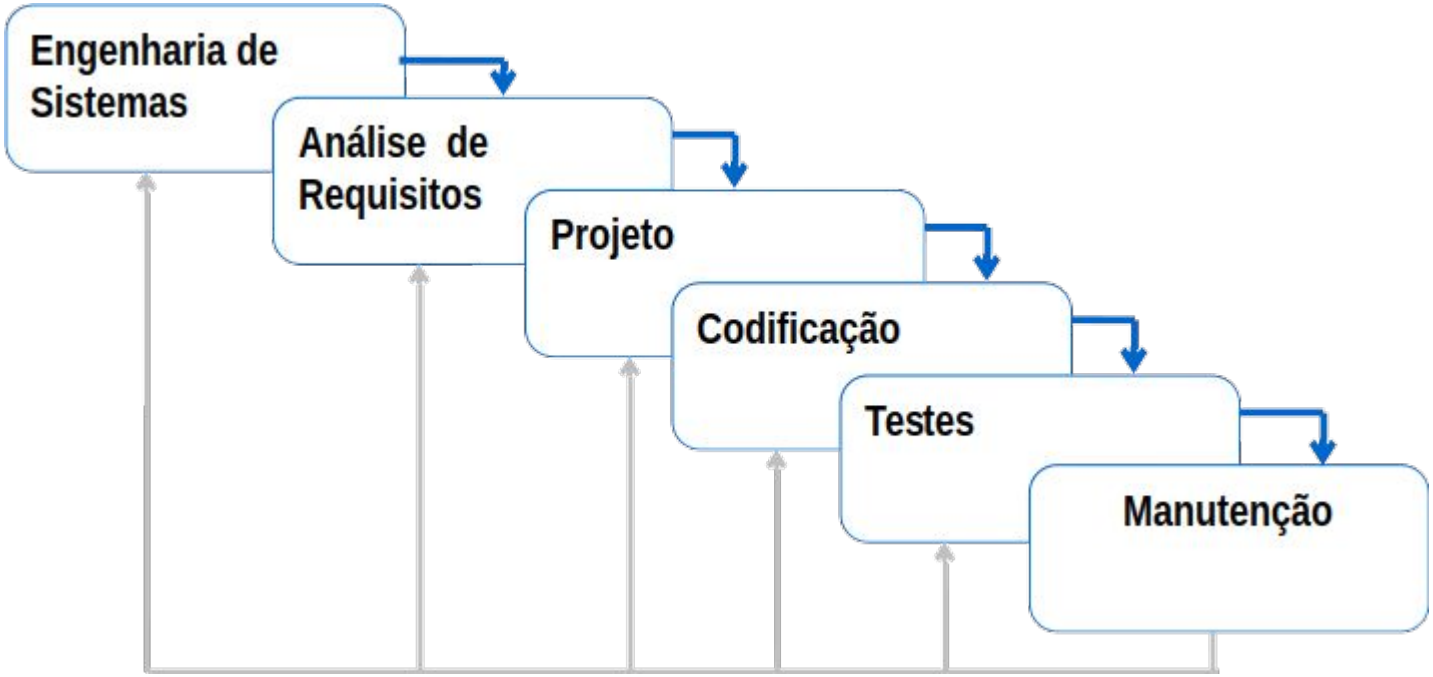
- Planejamento
- Divisão do processo em fases
- Divisão de papéis
- Sequenciamento de atividades
- Estimativas/Medição/Controle
- Documentação rigorosa
- Políticas de controle de alterações

Abordagens de desenvolvimento tradicionais

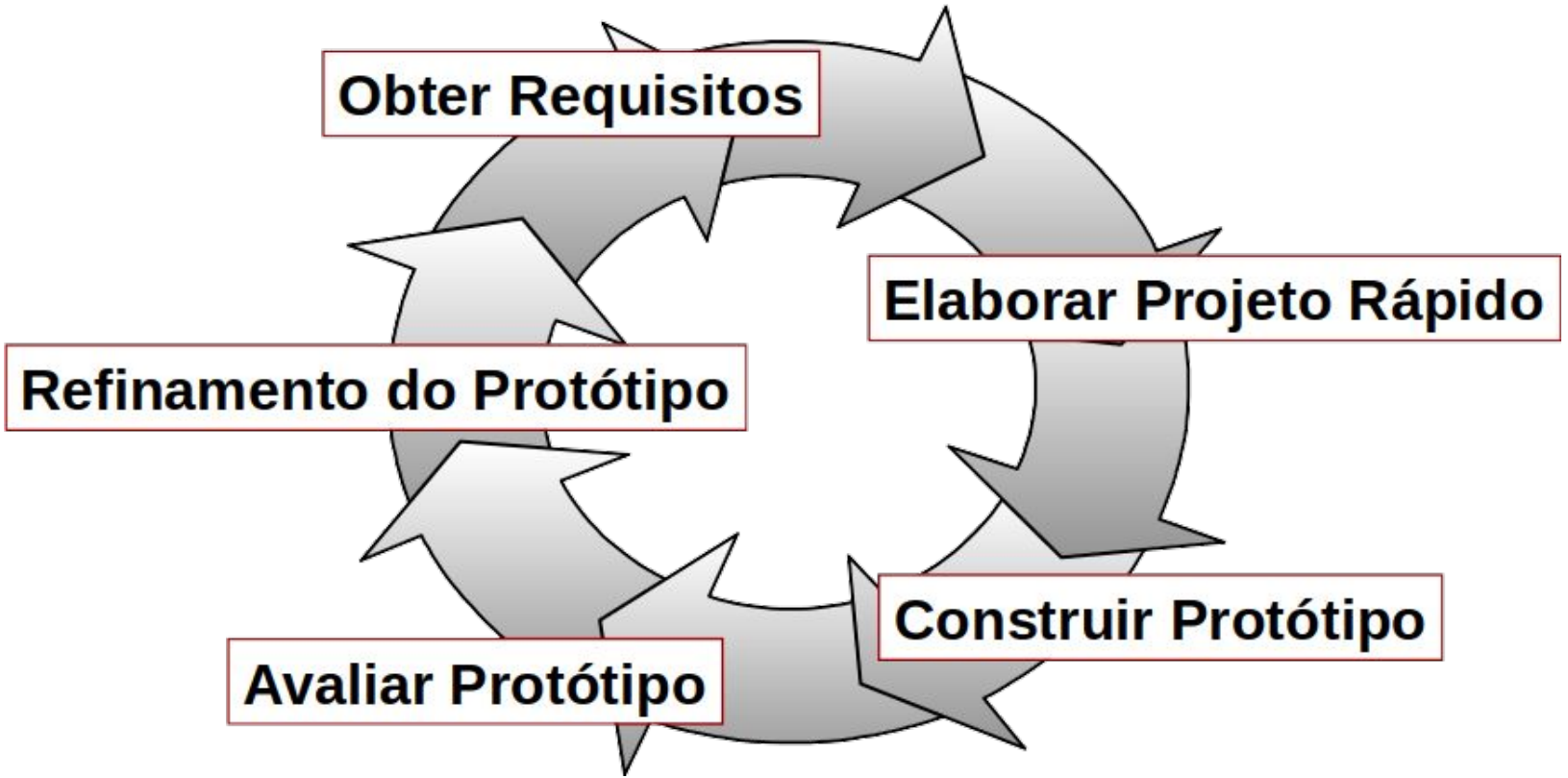
Exemplos

- Modelo Cascata
- Modelo espiral
- Modelo de Prototipação
- Modelos Evolutivos
- Processo Unificado

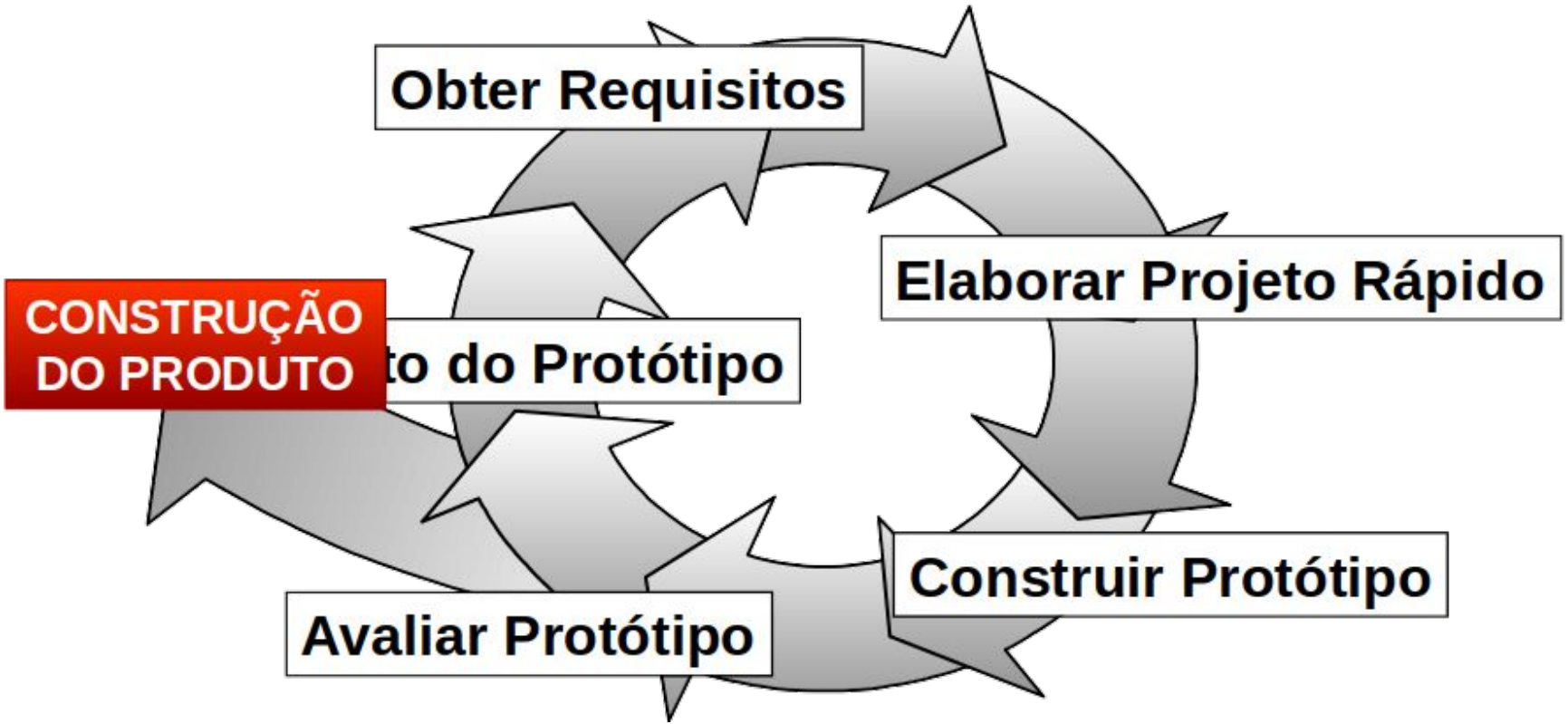
Modelo Cascata



Modelo de Prototipação



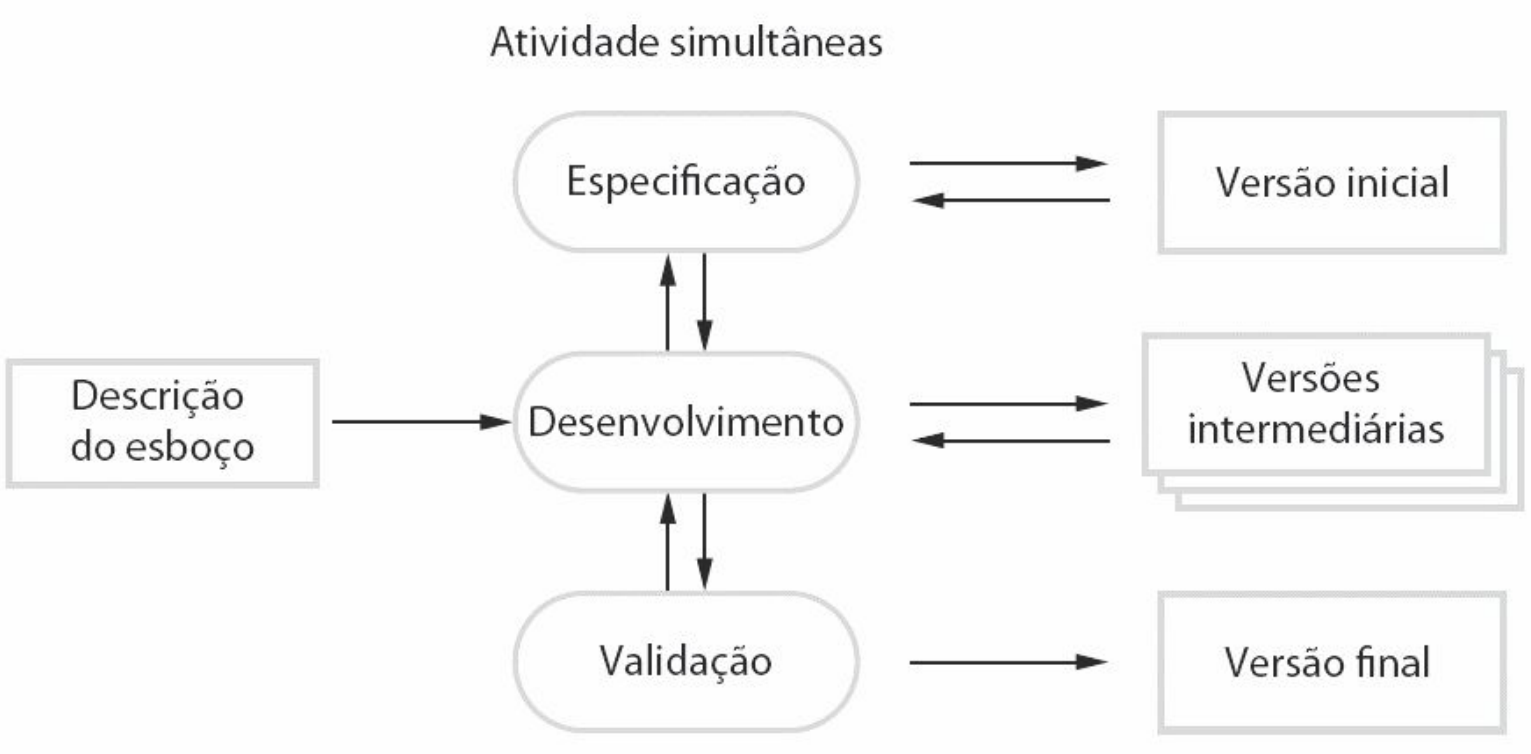
Modelo de Prototipação



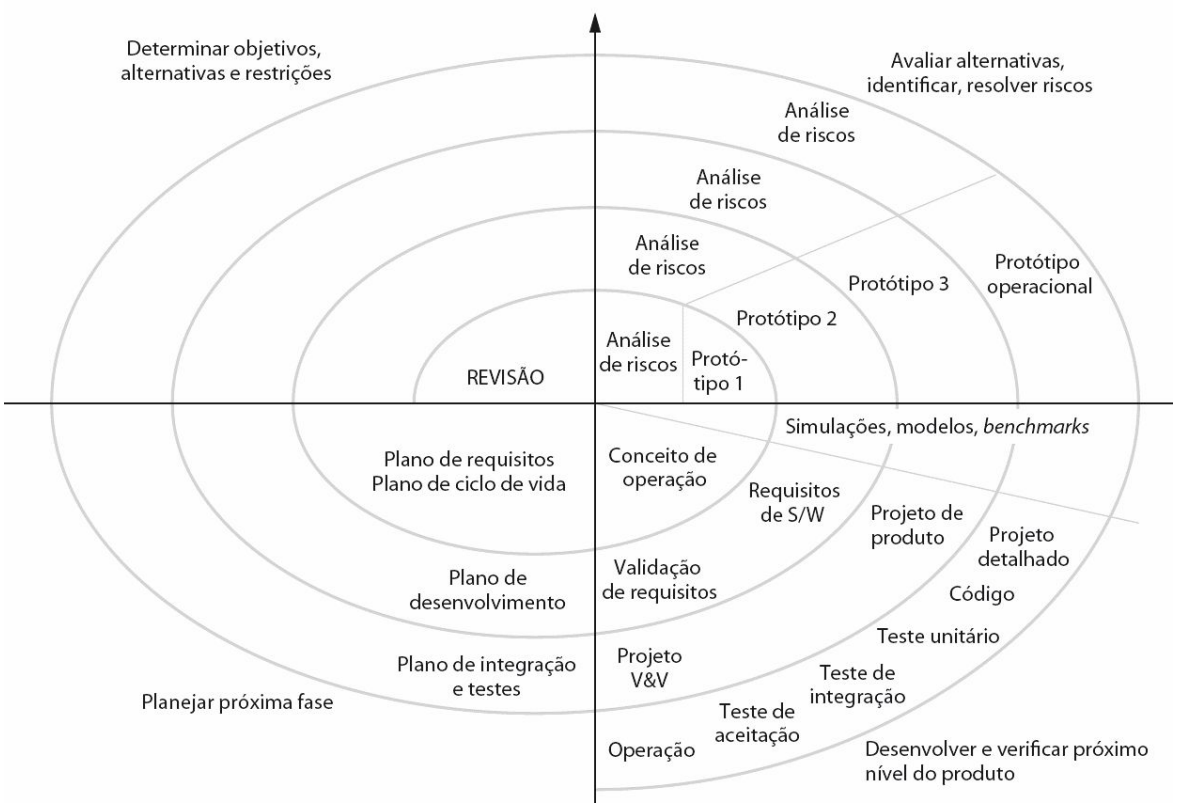
Modelo de Prototipação

- Fred Brooks disse o seguinte: “Planeje jogar uma implementação fora, pois você irá.”
- Brooks refere-se ao fato de que é mais fácil para os clientes entender o que eles querem uma vez que tenham usado um protótipo, e também é mais fácil para os engenheiros entenderem como construir um software uma vez que já o tenham feito uma vez.

Modelo Incremental



Modelo Espiral



Resumo dos métodos tradicionais

- Planejamento rigoroso
- Extensa documentação
- Etapas e papéis bem definidos
- Pontos de controle rígidos
- Foco na documentação e no processo
- Antecipação de todos os detalhes de desenvolvimento

Processos tradicionais de desenvolvimento

O que você achou deste tipo de processo? Alguma crítica?

Processos tradicionais de desenvolvimento

Críticas comuns:

- Divisão distinta de fases no projeto gera inflexibilidade uma vez que raramente os projetos seguem um fluxo sequencial.
- Requisitos totalmente especificados e “congelados” na primeira fase do projeto dificultam futuras mudanças.
- Arquitetura especificada e “congelada” na segunda fase do projeto torna a arquitetura pouco confiável diante de possíveis mudanças de requisitos.

Processos tradicionais de desenvolvimento

Críticas comuns:

- Grande dificuldade de alterações no projeto depois de decisões já tomadas.
- A excessiva documentação torna o processo “pesado”
- O excessivo planejamento “aprisiona” os envolvidos no desenvolvimento

Engenharia de Software

Tipos de processos que se destacam:

- Processos tradicionais (preditivos)
- **Métodos ágeis (adaptativos)**

Métodos ágeis

Muitos profissionais e pesquisadores não estavam contentes com a visão tradicional de que o software é um produto de engenharia tradicional

Modelos alternativos aos processos tradicionais foram criados para “abraçar” a mudança como um fato inevitável no processo de desenvolvimento

Uma das características que diferencia o software de um produto tradicional de engenharia é sua grande capacidade de mudança

Manifesto Ágil

Estamos descobrindo maneiras melhores de desenvolver software fazendo-o nós mesmos e ajudando outros a fazê-lo. Através deste trabalho, passamos a valorizar:

- **Indivíduos e interações** mais que processos e ferramentas
- **Software em funcionamento** mais que documentação abrangente
- **Colaboração do cliente** mais que negociação de contratos
- **Responder a mudanças** mais do que seguir um plano

Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

Site:

- https://agilemanifesto.org/

Princípios ágeis

- Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.
- Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.
- Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.
- Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

Princípios ágeis

- Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.
- Software funcionando é a medida primária de progresso.
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.

Princípios ágeis

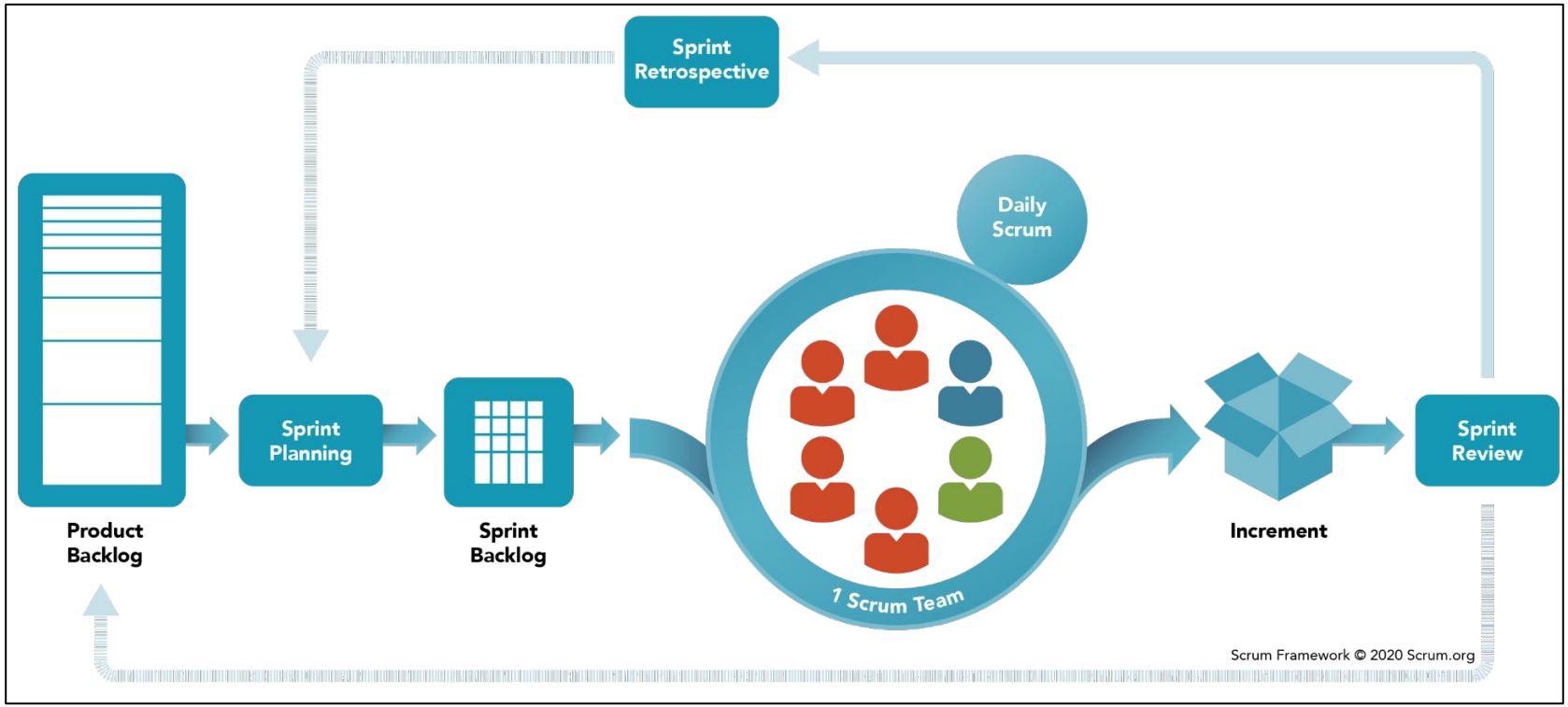
- Contínua atenção à excelência técnica e bom design aumenta a agilidade.
- Simplicidade, a arte de maximizar a quantidade de trabalho (desnecessário) não realizado, é essencial.
- As melhores arquiteturas, requisitos e designs emergem de equipes auto organizáveis.
- Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Métodos ágeis

Abordagens:

- Gerencial (e.g. Scrum, Kanban)
- Técnica (e.g. XP)

SCRUM

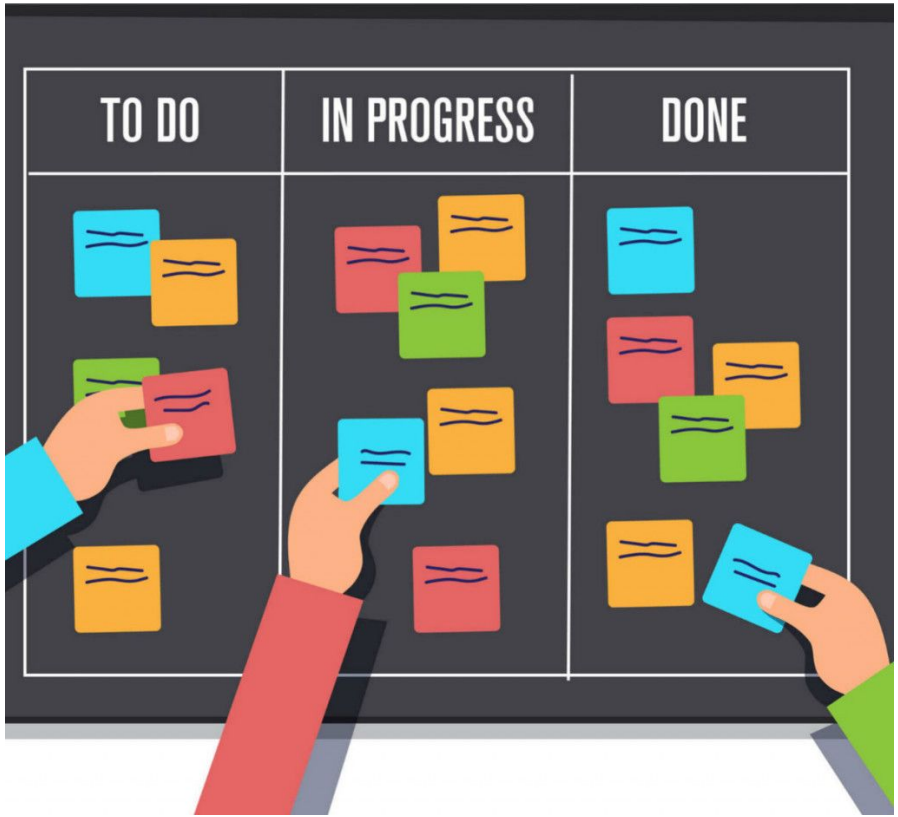

Sprint Retrospective
Daily Scrum
Product Backlog
Sprint Planning
Sprint Backlog
1 Scrum Team
Increment
Sprint Review
Scrum Framework © 2020 Scrum.org

SCRUM

Papéis:

- Scrum Master
- Product Owner
- Team

Kanban para o desenvolvimento de software



Kanban para o desenvolvimento de software

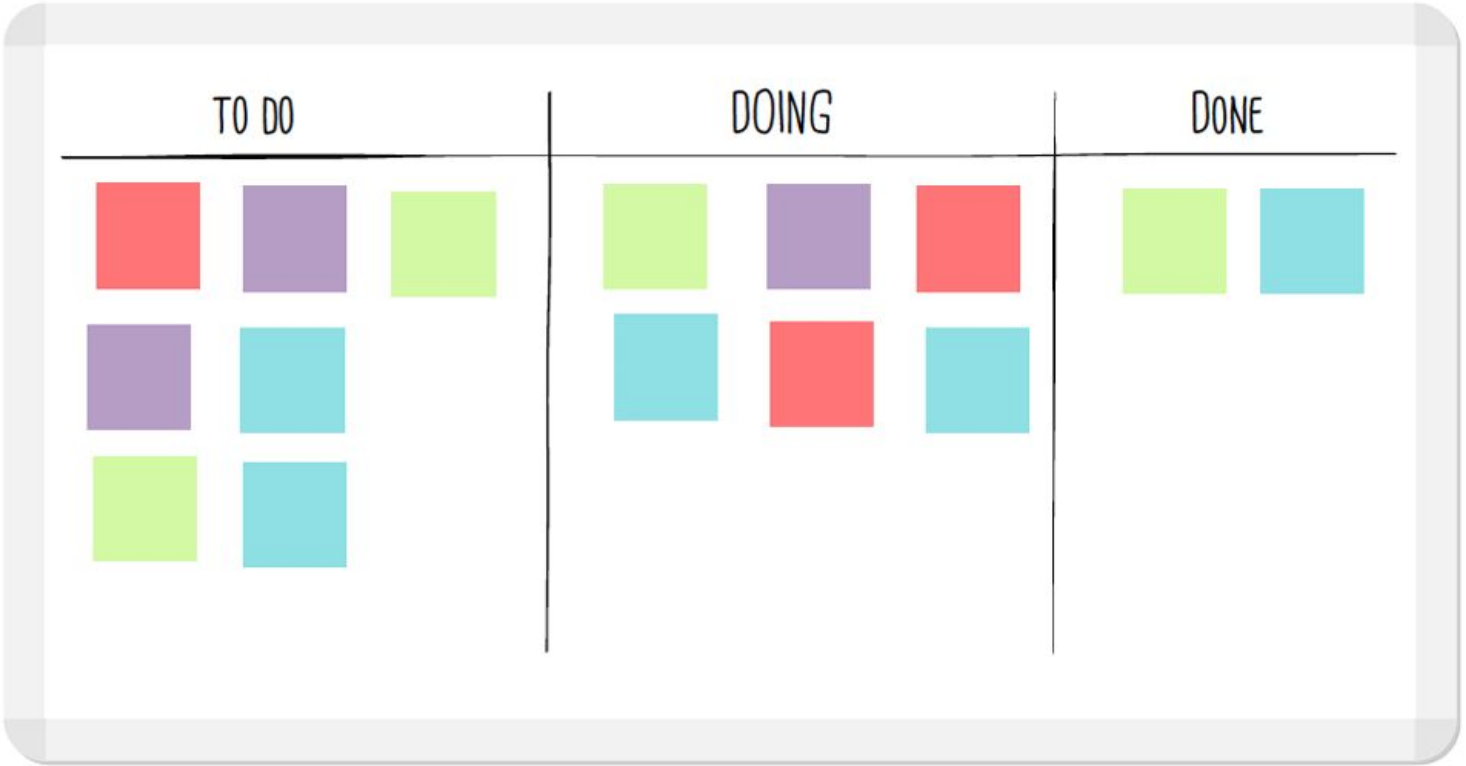
Visualize o fluxo de trabalho

- Divida o trabalho em partes, escreva cada item em um cartão e os coloque em uma parede
- Nomeie colunas para ilustrar em que parte do processo cada item está

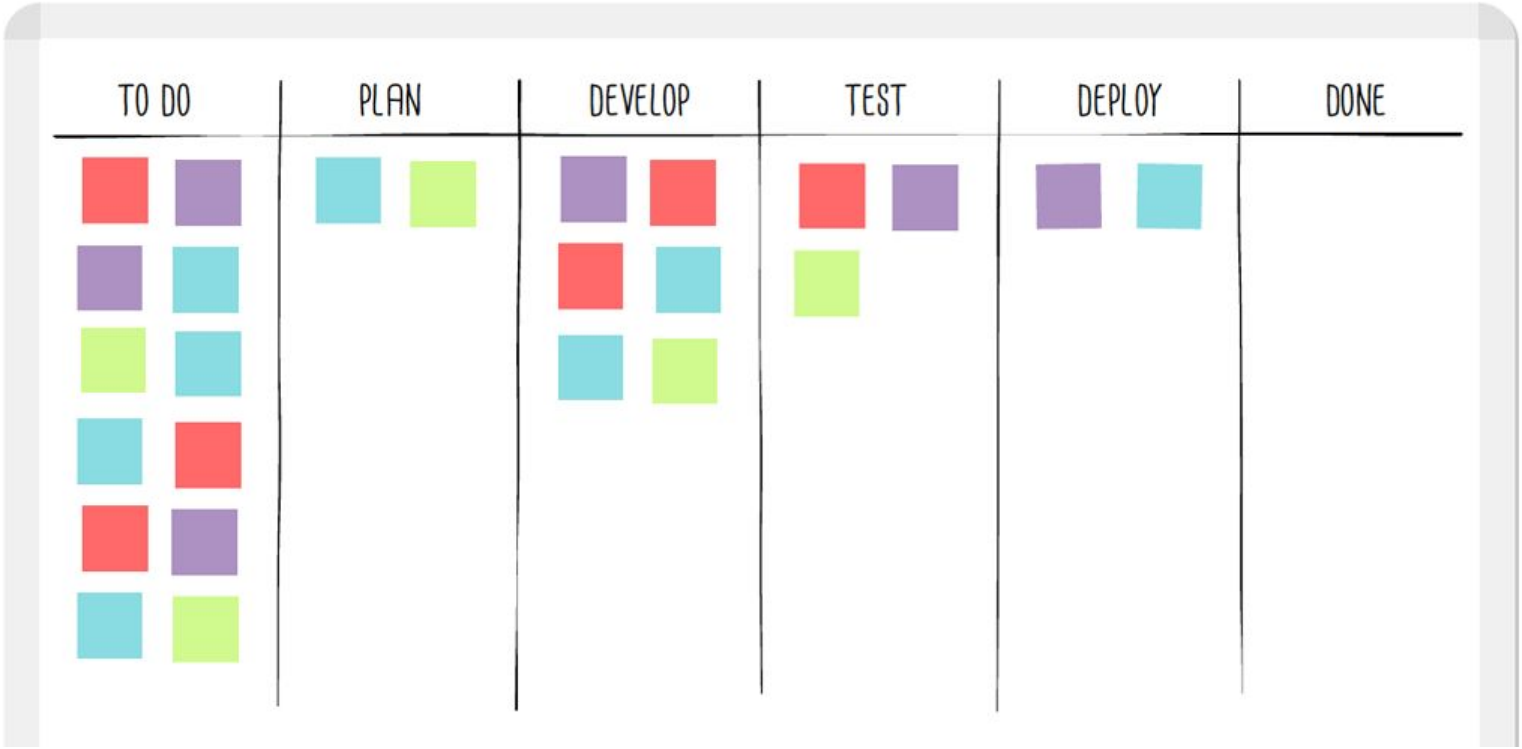
Limite o trabalho em andamento (Work In Progress - WIP)

Meça o tempo médio para a realização de cada item

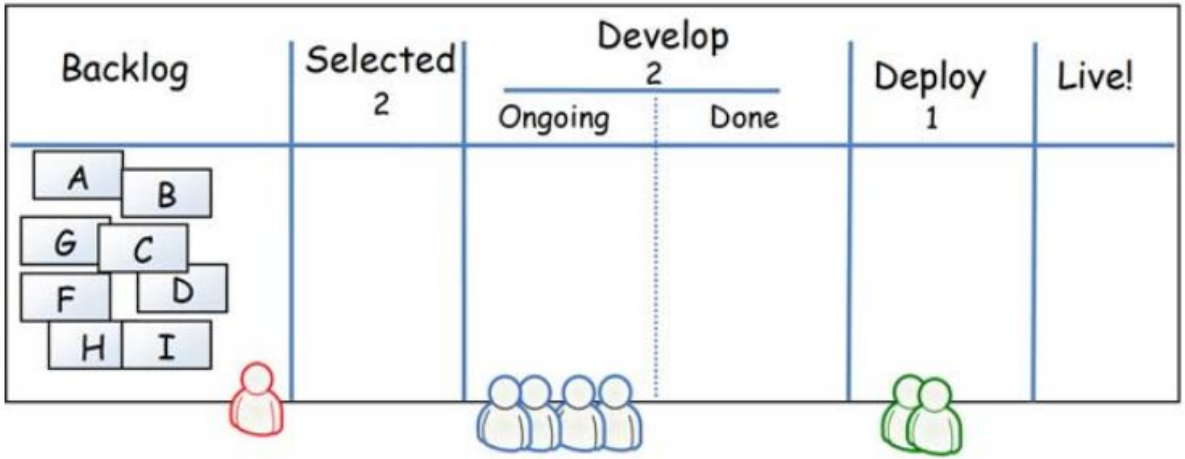
Kanban para o desenvolvimento de software

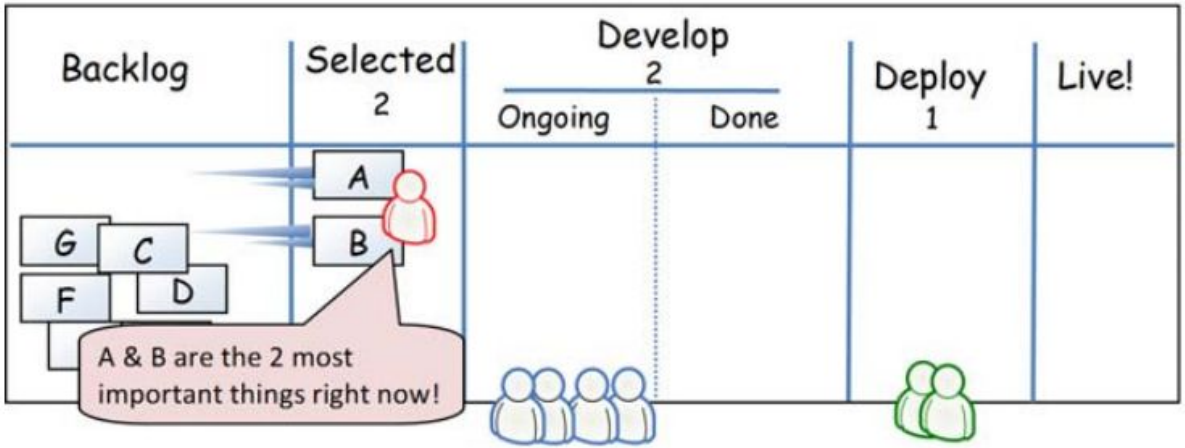


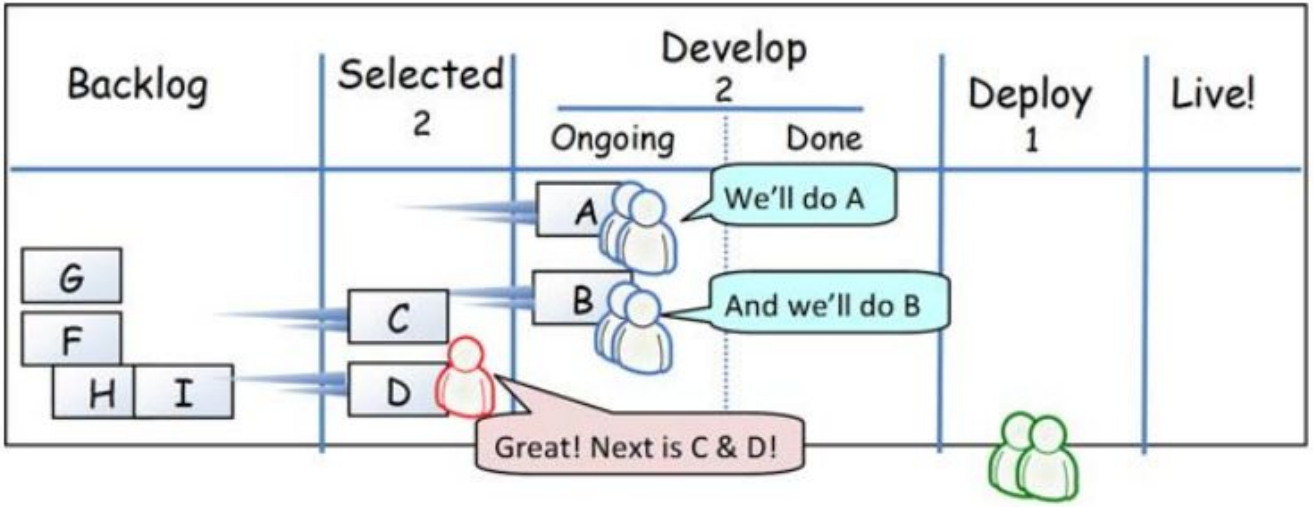
Kanban para o desenvolvimento de software

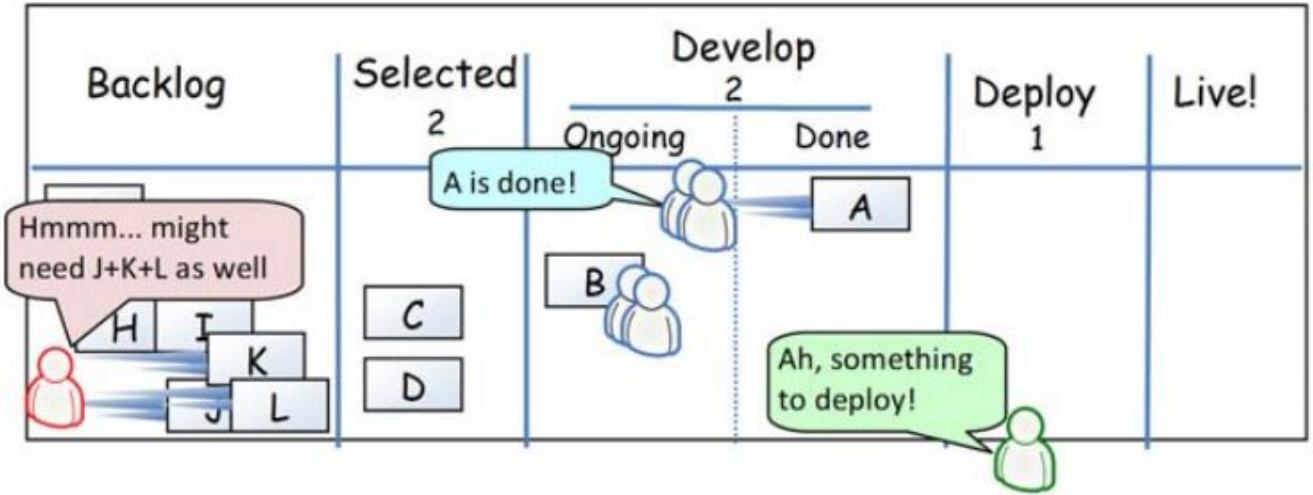


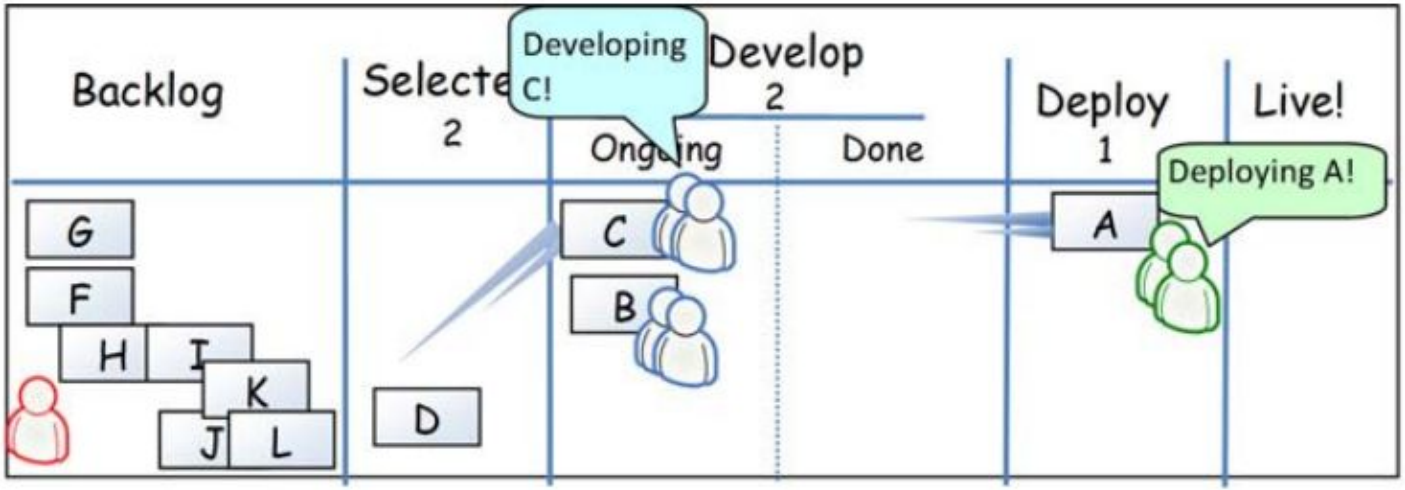
One day in Kanban-land

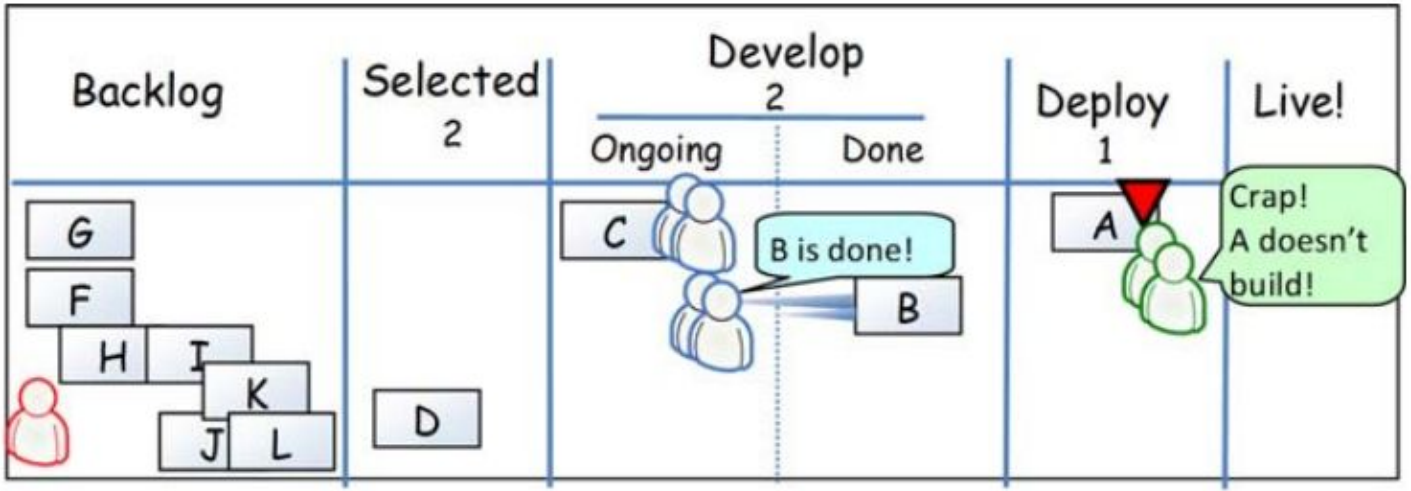


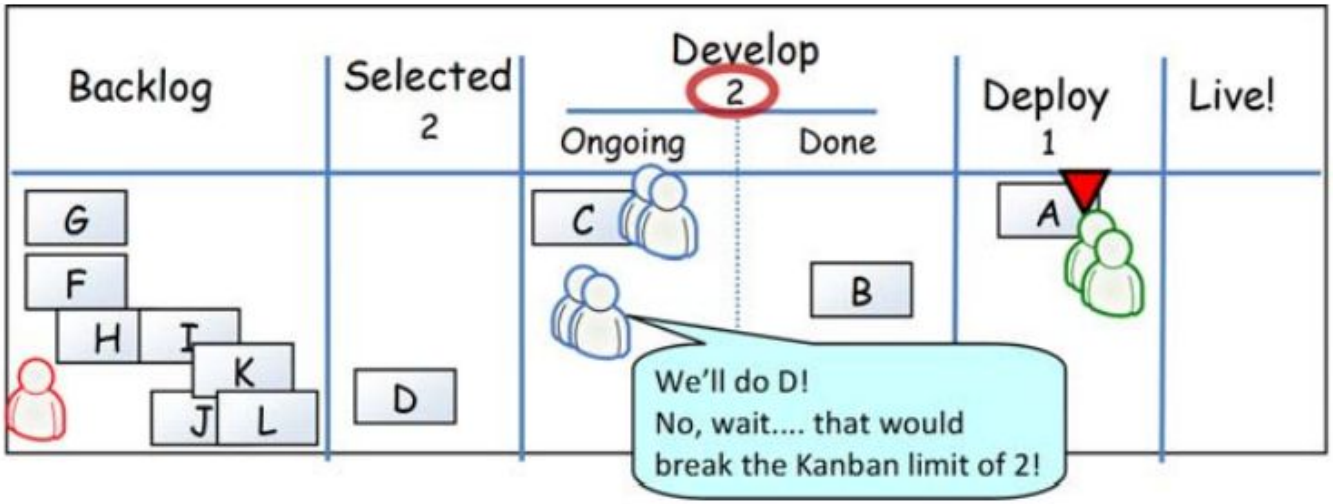


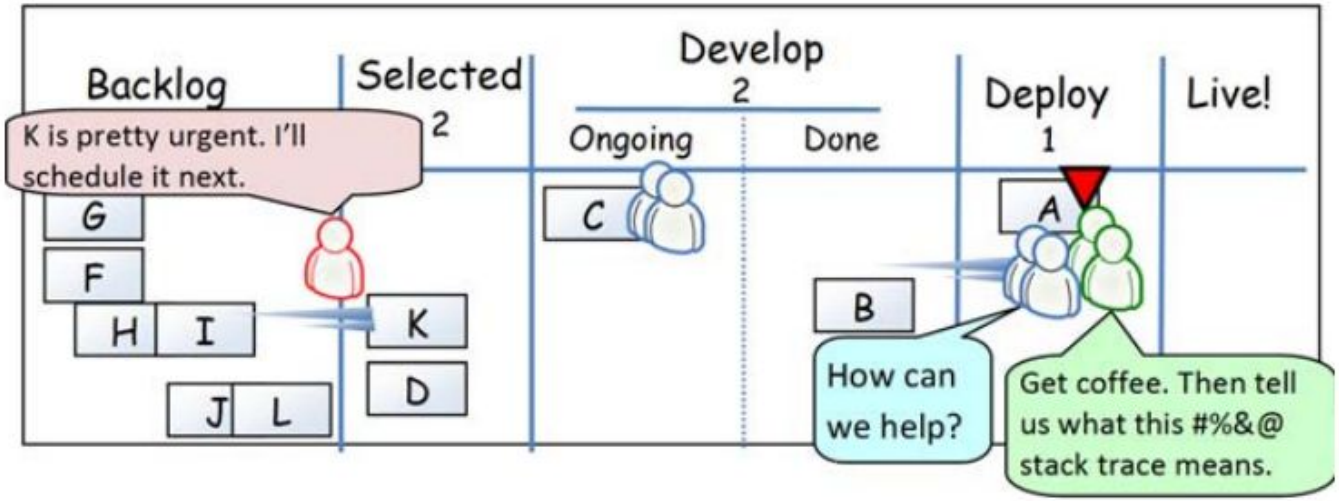

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
We'll do A
A
G
F
H
I
C
D
B
And we'll do B
Great! Next is C & D!

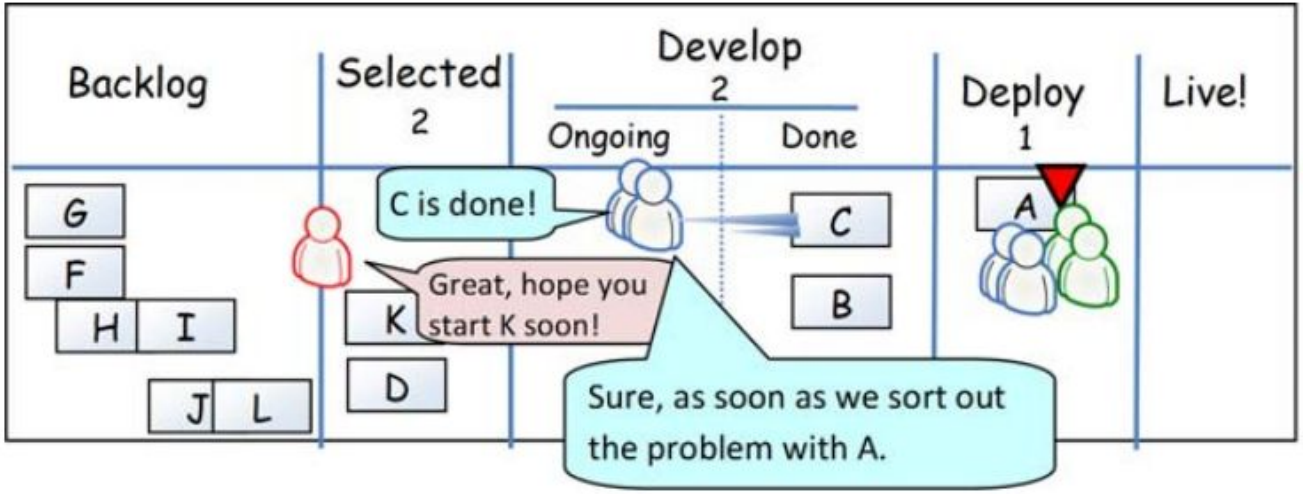

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
A is done!
A
Hmmm... might
need J+K+L as well
H
I
K
J
L
C
D
B
Ah, something
to deploy!

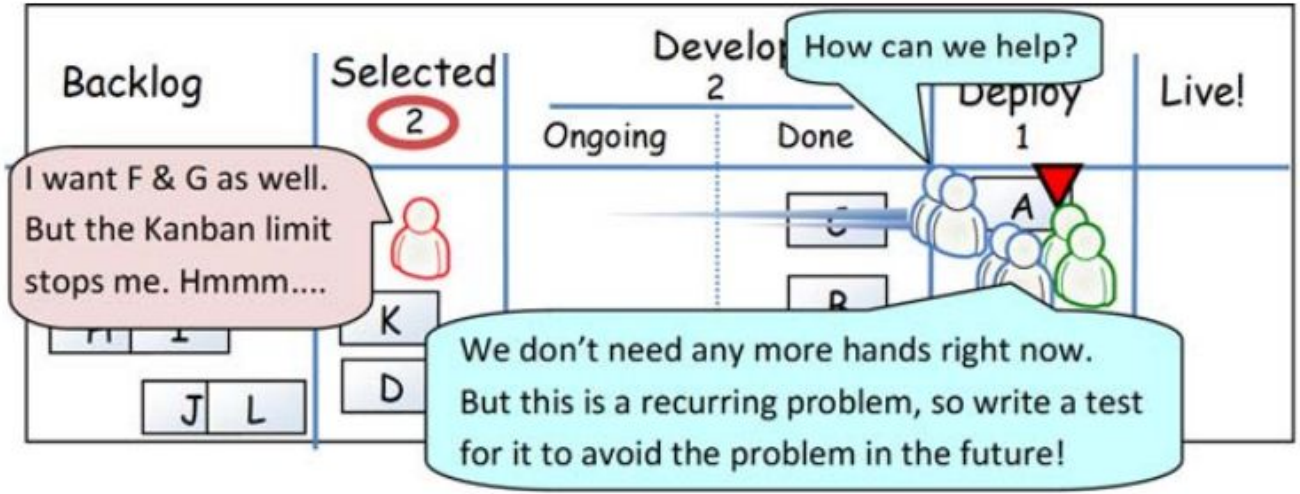

Backlog
Selecte
2
Developing
C!
Develop
2
Ongoing
Done
Deploy
1
Live!
Deploying A!
G
F
H
I
K
J
L
D
C
B
A

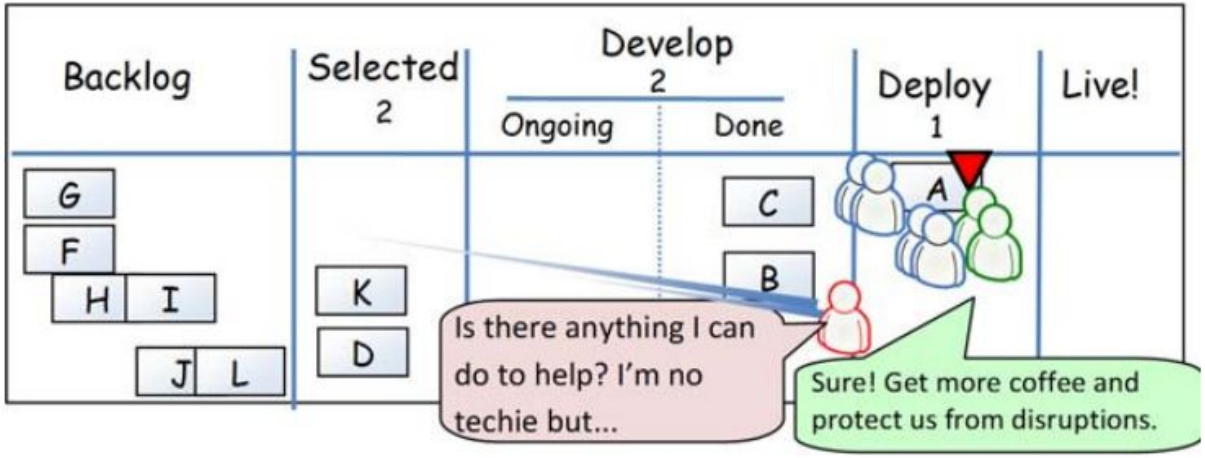

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
B is done!
Crap!
A doesn't
build!
G
F
H
I
K
J
L
D
C
B
A

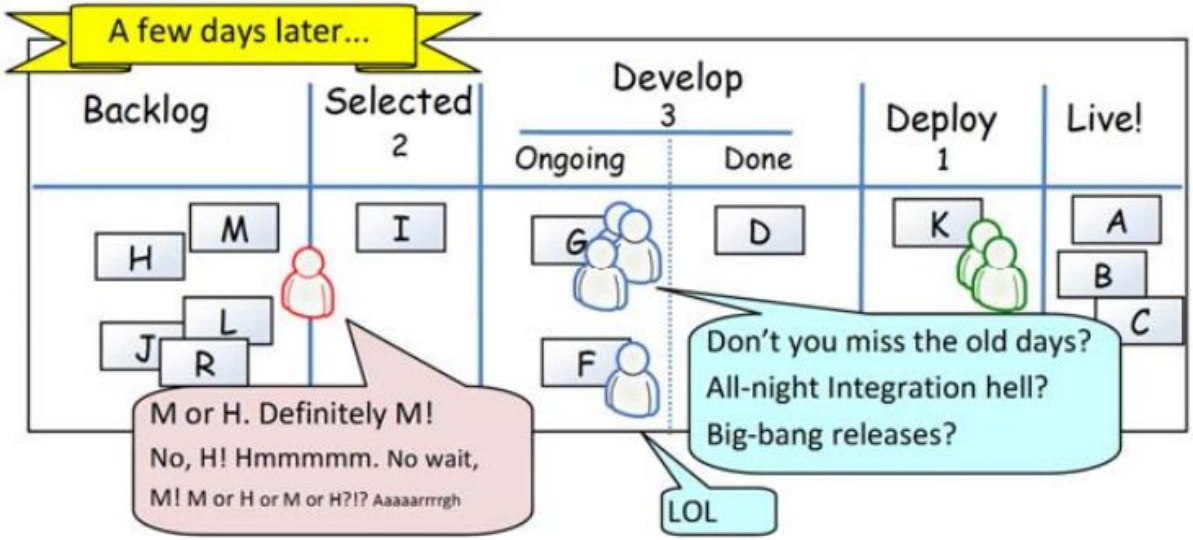

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
G
F
H
I
K
J
L
D
C
B
A
We'll do D!
No, wait.... that would
break the Kanban limit of 2!


Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
K is pretty urgent. I'll
schedule it next.
G
F
H
I
K
J
L
D
C
B
A
How can
we help?
Get coffee. Then tell
us what this #%&@
stack trace means.


Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
G
F
H
I
J
L
K
D
C is done!
C
B
A
Great, hope you start K soon!
Sure, as soon as we sort out the problem with A.

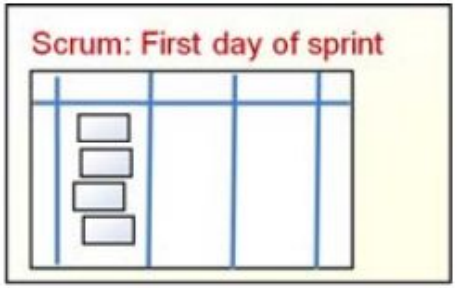

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
How can we help?
I want F & G as well. But the Kanban limit stops me. Hmmm....
K
D
J
L
A
We don't need any more hands right now. But this is a recurring problem, so write a test for it to avoid the problem in the future!

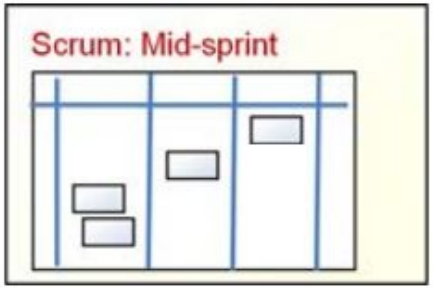

Backlog
Selected
2
Develop
2
Ongoing
Done
Deploy
1
Live!
G
F
H
I
J
L
K
D
C
B
A
Is there anything I can do to help? I'm no techie but...
Sure! Get more coffee and protect us from disruptions.

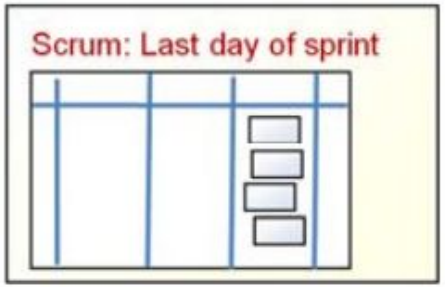

A few days later...
Backlog
Selected
2
Develop
3
Ongoing
Done
Deploy
1
Live!
M
H
L
J
R
I
G
F
D
K
A
B
C
M or H. Definitely M!
No, H! Hmmmmm. No wait,
M! M or H or M or H?!? Aaaaarrrrgh
Don't you miss the old days?
All-night Integration hell?
Big-bang releases?
LOL

Scrum e Kanban

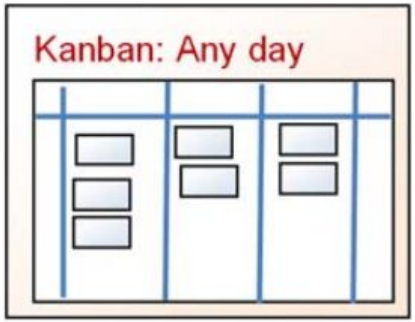
O quadro de tarefas do Scrum é “zerado” após cada Sprint





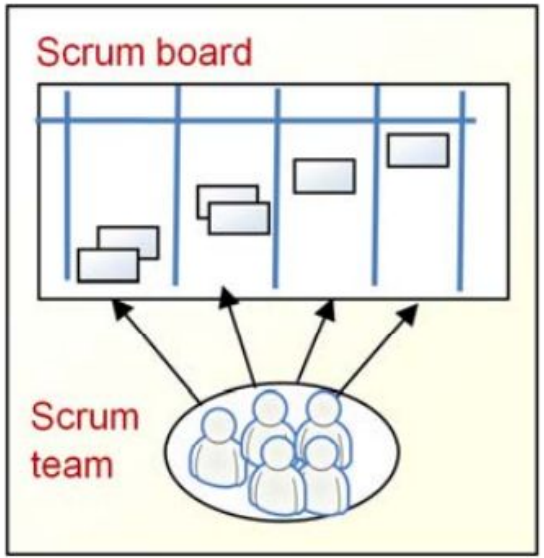


No Kanban o quadro está continuamente cheio



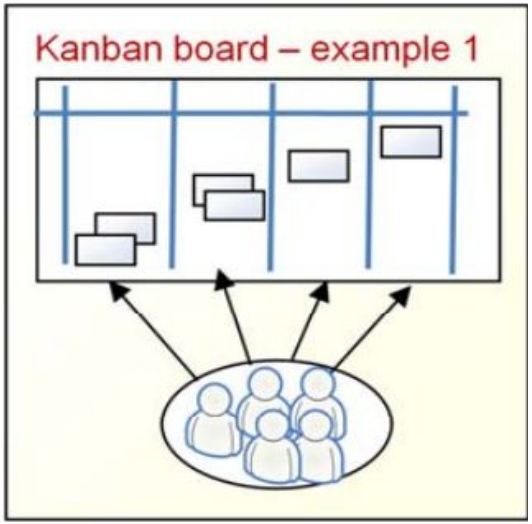
Scrum e Kanban

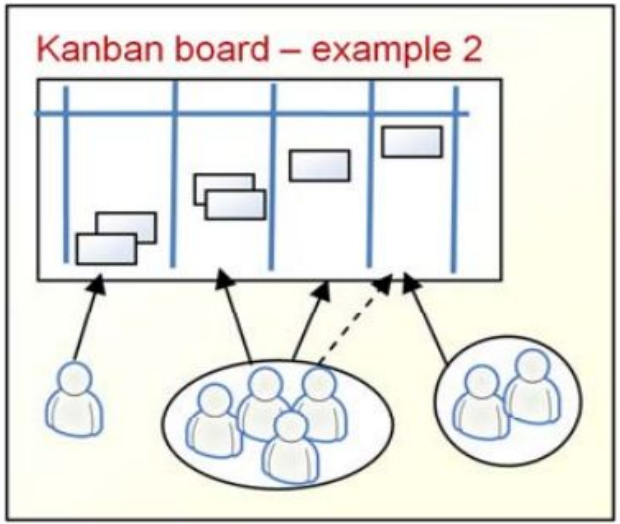
Scrum prescreve times coesos, em que a equipe possui pessoas para todas as funções básicas (analista, desenvolvedor, tester, etc)



Scrum e Kanban

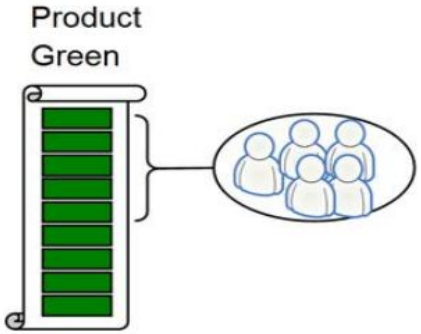
No Kanban, é possível escolher times que possuem todas as funções necessárias para o desenvolvimento de software, ou times com funções específicas

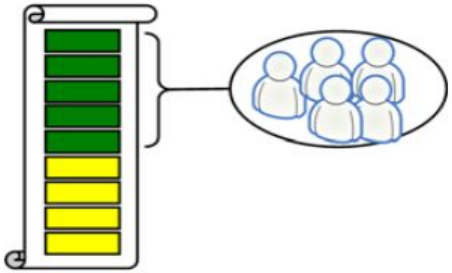


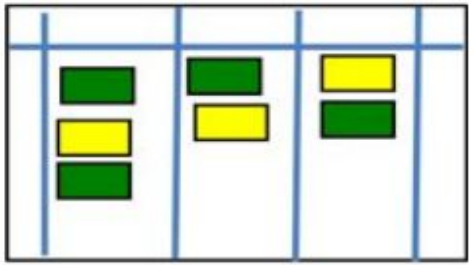


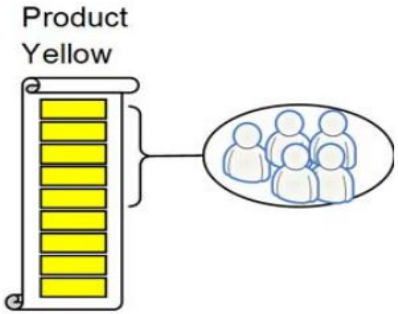
Scrum e Kanban

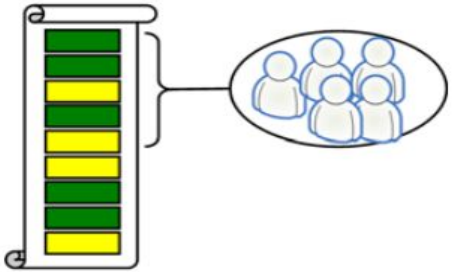
Ambos permitem que se trabalhe em dois produtos ao mesmo tempo

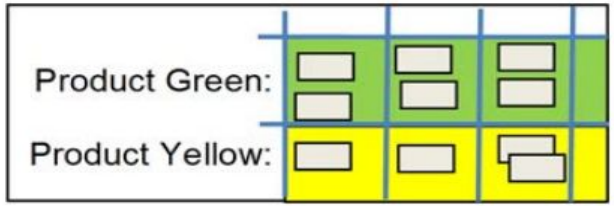












Um pouco de reflexão

Todas as empresas que dizem ser “ágeis” são, de fato, ágeis?

Que framework/métodos ágeis elas seguem?

Ou simplesmente se dizem ágeis porque não seguem nenhum modelo tradicional?

Entregar valor para o cliente em um curto espaço de tempo, de forma sustentável, com a qualidade esperada, exige comprometimento com atividades gerenciais e técnicas específicas.

ACH2028
Qualidade de Software

Aula 02 - Revisão de Engenharia de Software

Prof. Marcelo Medeiros Eler
marceloeler@usp.br

Escola de Artes, Ciências e Humanidades - Universidade de São Paulo