



Engenharia de Software - Fundamentos

Práticas Ágeis de Engenharia

Prof. Guilherme Lacerda

guilhermeslacerda@gmail.com

Roteiro

- Do individual ao Colaborativo
- Boas Práticas e Automação
- Documentação Técnica

“O talento vence jogos, mas só o trabalho em equipe vence campeonatos.”

Michael Jordan

**Do individual
ao Colaborativo**

O trabalho da pessoa desenvolvedora

■ Principais Atividades

- Design/Programação/Testes/Manutenção/Evolução
- Gestão de configuração e do trabalho
- Colaborar com outros profissionais

■ Desafios

- Dominar tecnologia(s)
- Manter-se atualizada(o)
- Ter a visão do todo
- Ser especialista/generalista
- Interagir com clientes/usuários
- Preocupar-se constantemente com a qualidade
- Se adaptar!



Padrões/Convenções

- Toda a linguagem tem
 - E por que não usamos?
 - Quais são os benefícios?
 - Prática do XP
- Componentes
 - Nomenclatura
 - Estrutura do código
 - Terminologia
 - Formatação
 - Boas Práticas
 - Exemplos

```
354 }
355 Carousel.prototype.getItemForDirection = function (direction, active) {
356   this.$items = item.parent().children();
357   return this.$items.eq(this.$activeIndex + (direction === 'prev' ? -1 : 1));
358 }
359
360 Carousel.prototype.getItemIndex = function (item) {
361   var delta = direction === 'prev' ? -1 : 1;
362   var activeIndex = this.$activeIndex;
363   var itemIndex = (activeIndex + delta) % this.$items.length;
364   return this.$items.eq(itemIndex);
365 }
366
367 Carousel.prototype.to = function (pos) {
368   var that = this;
369   var activeIndex = this.$activeIndex;
370   if (pos > (this.$items.length - 1) || pos < 0) return;
371   if (this.sliding) return this.$element.one('slid.bs.carousel', function () {
372     that.to(pos);
373   });
374   if (activeIndex == pos) return this.pause().cycle();
375   return this.slide(pos > activeIndex ? 'next' : 'prev', this.$items.eq(pos));
376 }
377
378 Carousel.prototype.pause = function (e) {
379   e || (this.paused = true);
380
381   if (this.$element.find('.next, .prev').length && $.support.transition) {
382     this.$element.trigger($.support.transition.end);
383     this.cycle(true);
384   }
385
386   this.interval = clearInterval(this.interval);
387
388   return this;
389 }
```

Exemplos

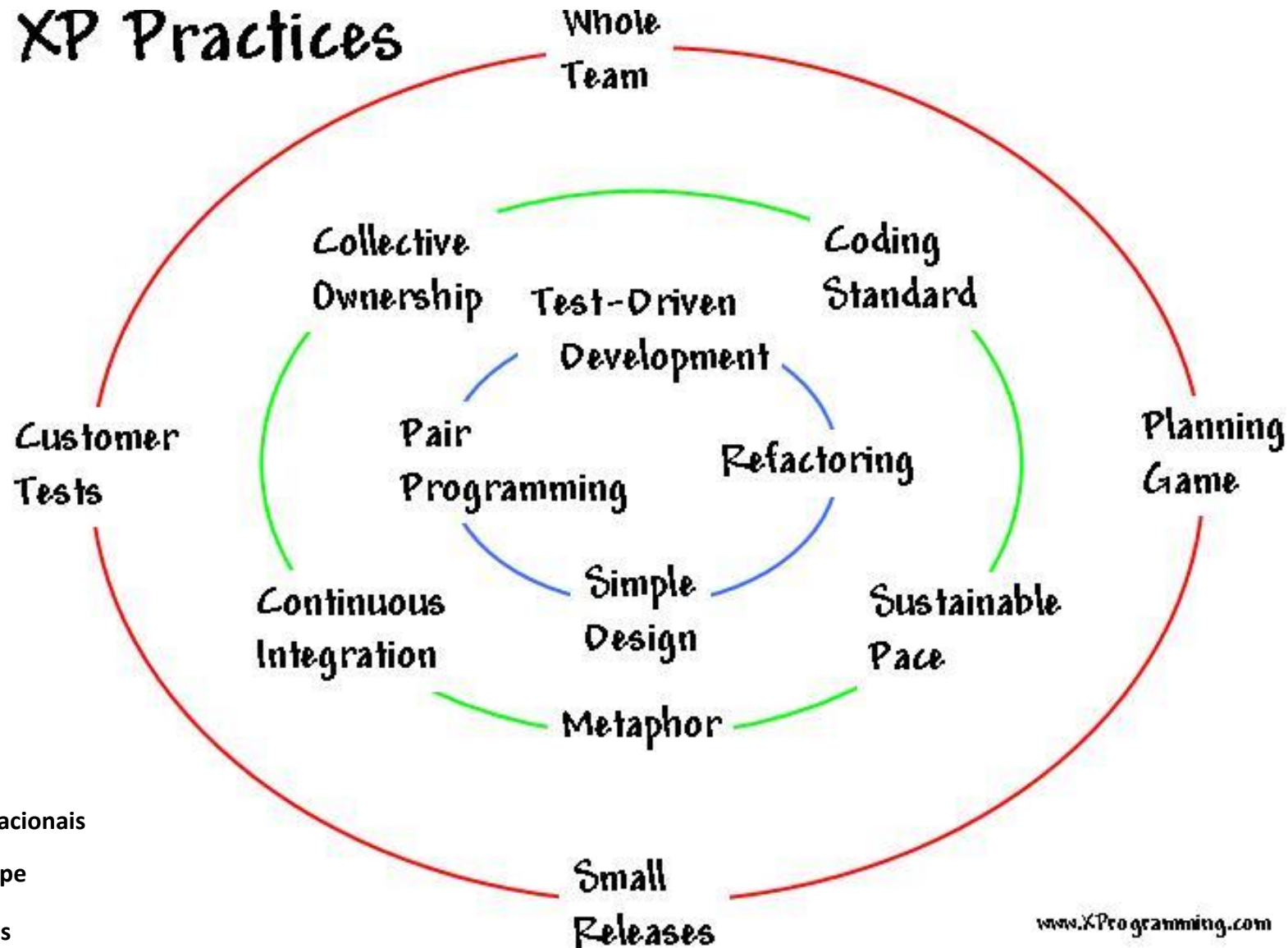
- Javascript
 - <https://github.com/standard/standard>
 - <https://github.com/airbnb/Javascript>
- Java
 - <https://google.github.io/styleguide/javaguide.html>
- Python
 - <https://www.python.org/dev/peps/pep-0008/>
- C#
 - <https://docs.microsoft.com/pt-br/dotnet/csharp/fundamentals/coding-style/coding-conventions>

```
354 }
355 Carousel.prototype.getItemForDirection = function (direction, active) {
356   this.$items = item.parent().this.$items;
357   return this.$items.eq(itemIndex || this.$items.length - 1);
358 }
359
360 Carousel.prototype.getItemForDirection = function (direction, active) {
361   var delta = direction == 'prev' ? -1 : 1;
362   var activeIndex = this.getItemIndex(active);
363   var itemIndex = (activeIndex + delta) % this.$items.length;
364   return this.$items.eq(itemIndex);
365 }
366
367 Carousel.prototype.to = function (pos) {
368   var that = this;
369   var activeIndex = this.getItemIndex(this.$active = this.$element.find('.item.active'));
370
371   if (pos > (this.$items.length - 1) || pos < 0) return;
372
373   if (this.sliding) return this.$element.one('slid.bs.carousel', function () {
374     if (activeIndex == pos) return this.pause().cycle();
375     return this.slide(pos > activeIndex ? 'next' : 'prev', this.$items.eq(pos));
376   });
377
378   return this.slide(pos > activeIndex ? 'next' : 'prev', this.$items.eq(pos));
379 }
380
381 Carousel.prototype.pause = function (e) {
382   e || (this.paused = true);
383
384   if (this.$element.find('.next, .prev').length && $.support.transition) {
385     this.$element.trigger($.support.transition.end);
386     this.cycle(true);
387   }
388
389   this.interval = clearInterval(this.interval);
390
391   return this;
392 }
```

Boas Práticas e Automação

Práticas do eXtreme Programming

XP Practices



Engenharia

- Assim como a gestão, a **engenharia** precisou se **renovar**
- Conjunto de **práticas** e **ferramentas** para apoiar o **processo**
 - Princípios de Design
 - Refatoração e heurísticas de limpeza
 - Pair/Mob Programming, Revisões de código
 - Automações
- Entender que **software** é um **produto**
 - Diferente de outros produtos (virtual, ausente de propriedades físicas)
 - Manutenção e evolução



Pair Programming e Mob Programming

- Pair Programming

- Mecânica
- Ferramentas

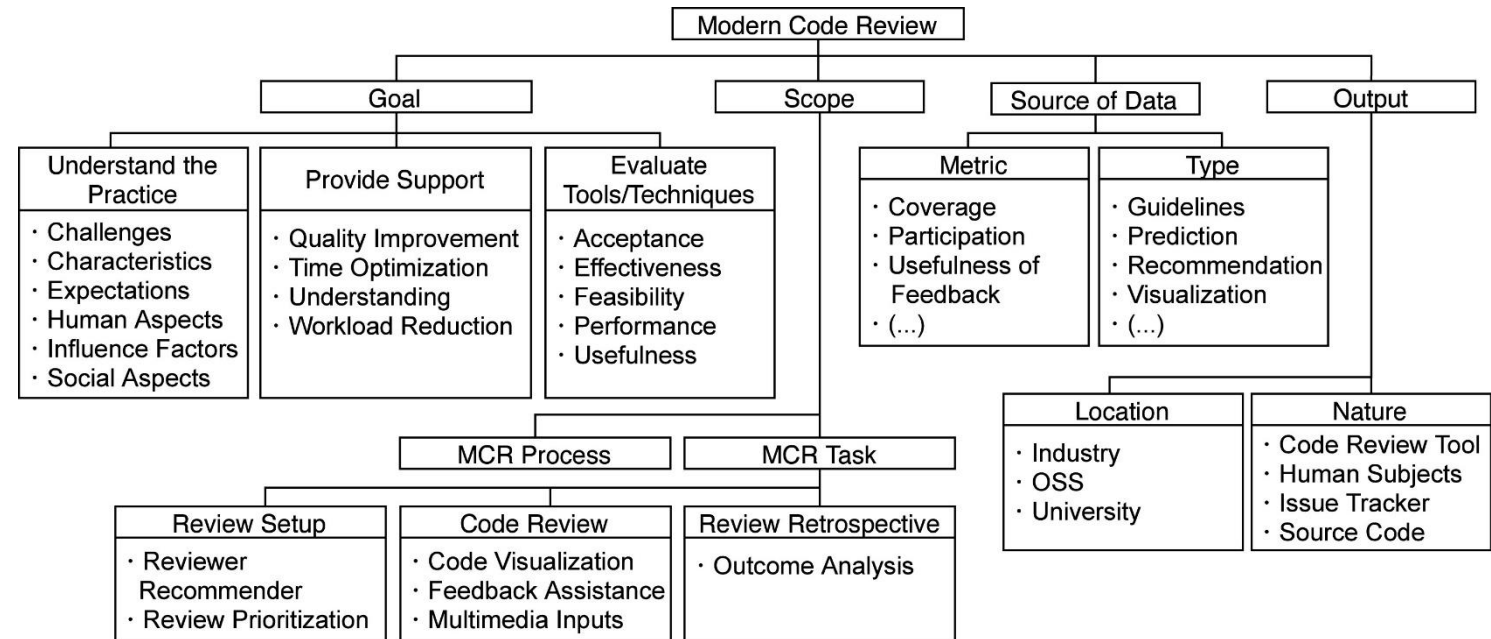
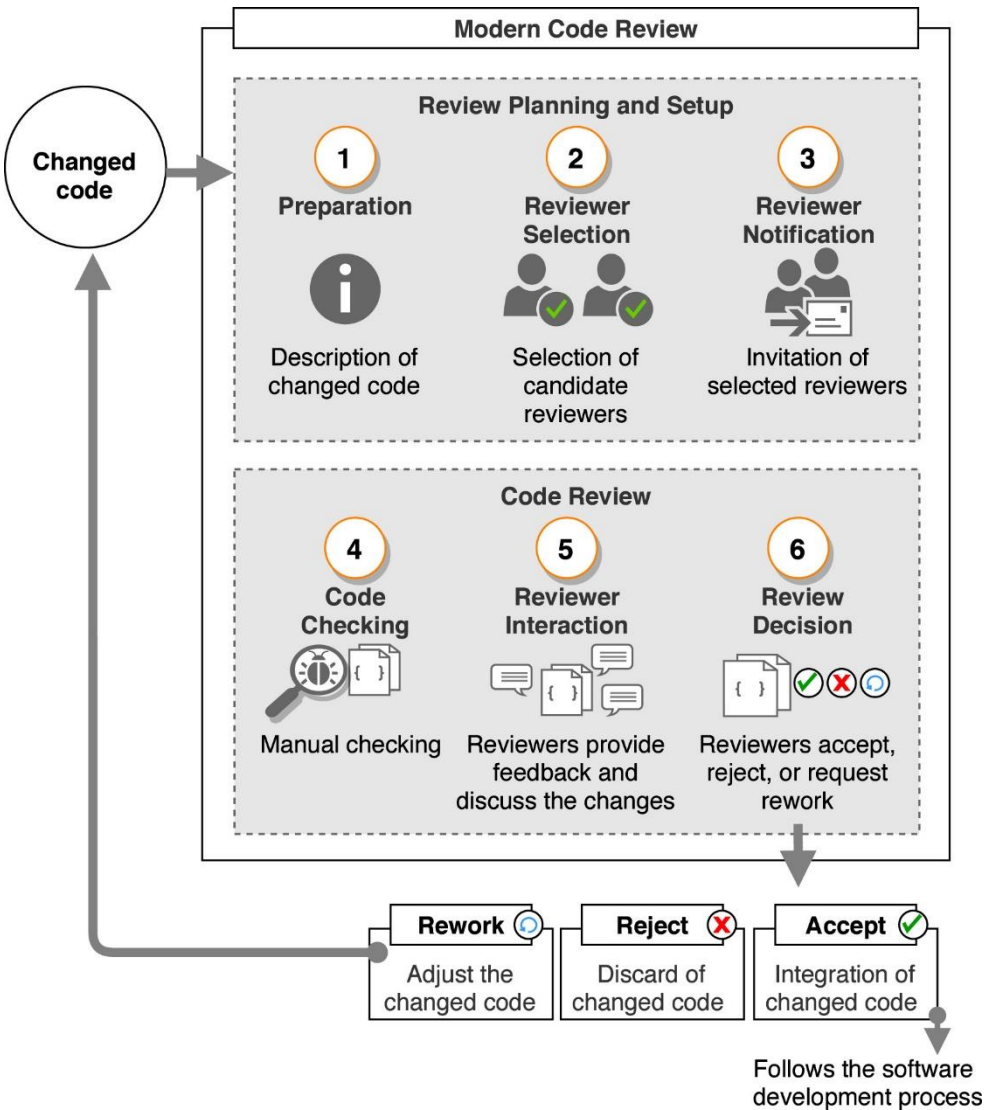
- Mob Programming

- Mecânica e Patterns

<https://github.com/michaelkeeling/mob-programming-patterns>



Modern Code Review



Code Review Pyramid

Questions to ask:

- Is the project's formatting style applied?
- Does it adhere to agreed on naming conventions
- Is it DRY?
- Is the code sufficiently "readable" (method lengths, etc.)

Questions to ask:

- Are all tests passing?
- Are new features reasonably tested?
- Are corner cases tested?
- Is it using unit tests where possible, integration tests where necessary?
- Are there tests for NFRs, e.g. performance?

Questions to ask:

- New features reasonably documented?
- Are the relevant kinds of docs covered: README, API docs, user guide, reference docs, etc.?
- Are docs understandable, are there no significant typos and grammar mistakes?

Questions to ask:

- Does it satisfy the original requirements?
- Is it logically correct?
- Is there no unnecessary complexity?
- Is it robust (no concurrency issues, proper error handling, etc.)?
- Is it performant?
- Is it secure (e.g. no SQL injections, etc.)?
- Is it observable (e.g. metrics, logging, tracing, etc.)?
- Do newly added dependencies pull their weight? Is their license acceptable?

Questions to ask:

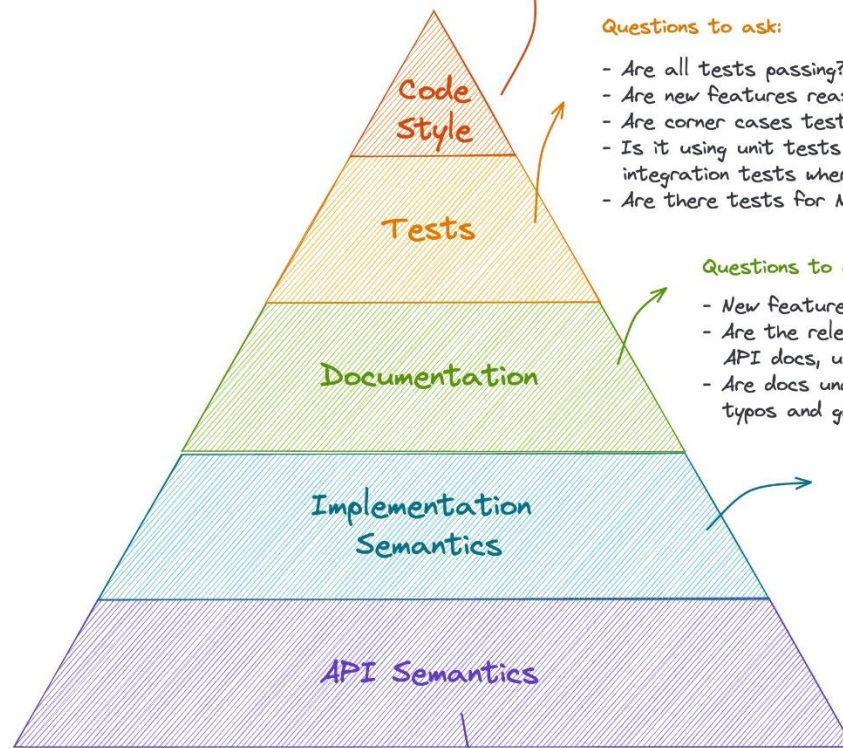
- API as small as possible, as large as needed?
- Is there one way of doing one thing, not multiple ones?
- Is it consistent, does it follow the principle of least surprises?
- Clean split of API/internals, without internals leaking in the API?
- Are there no breaking changes to user-facing parts (API classes, configuration, metrics, log formats, etc.)?
- Is a new API generally useful and not overly specific?

Smaller effort
for changes
later on

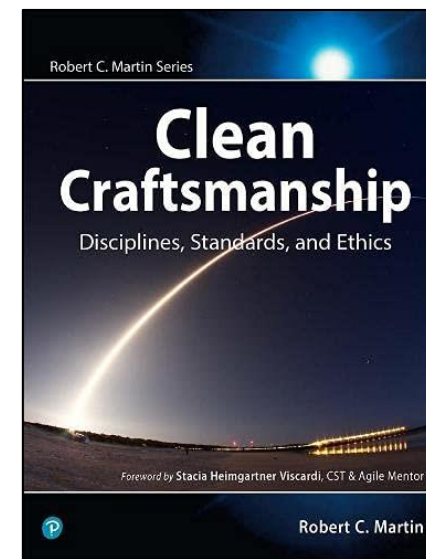
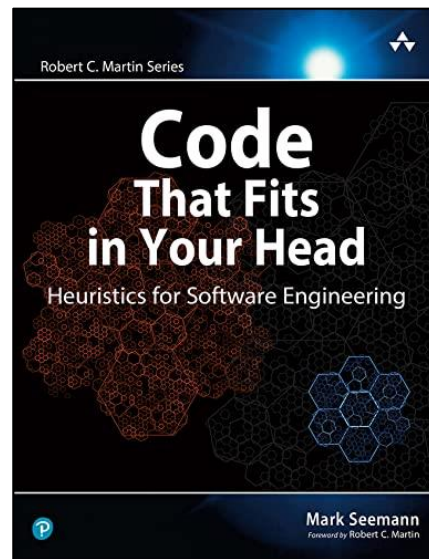
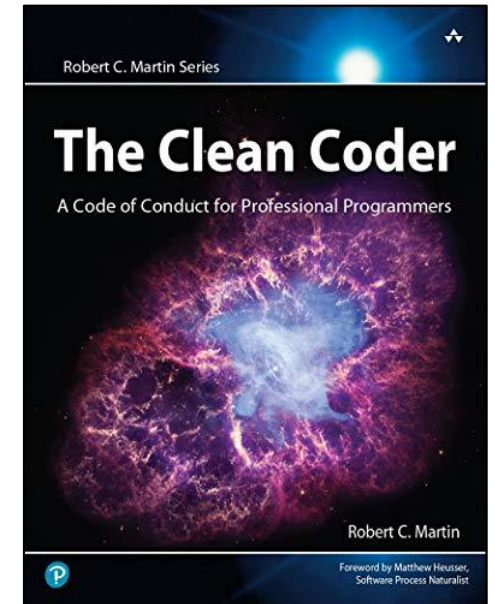
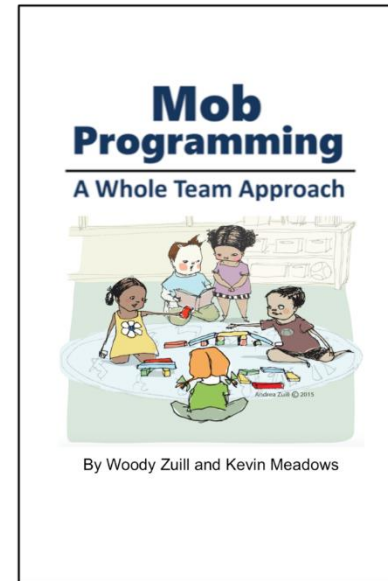
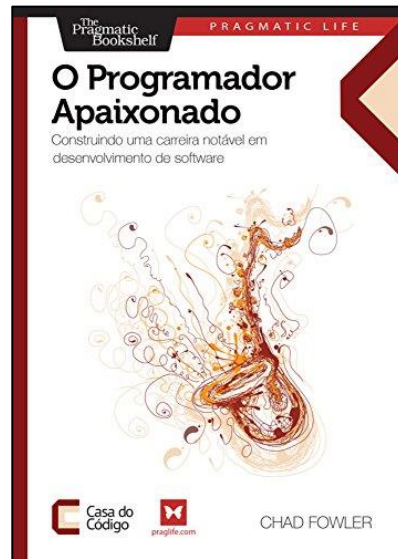
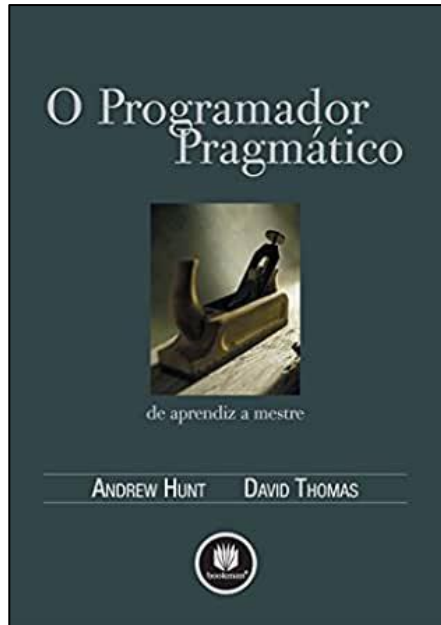
Automate here!

Focus your
review efforts
here!

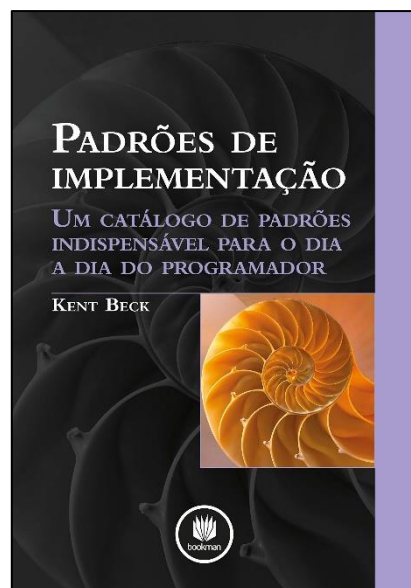
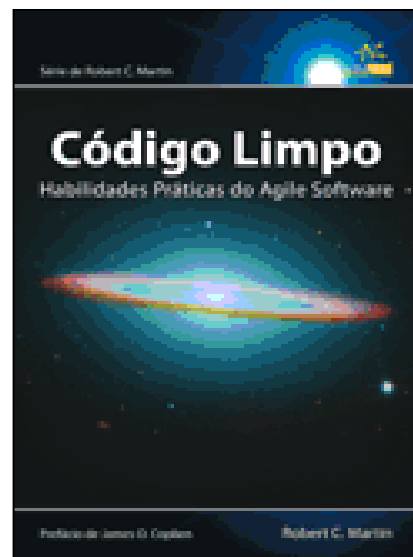
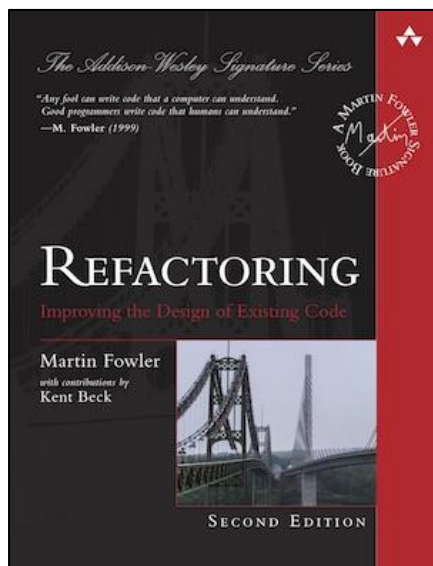
Larger effort
for changes
later on



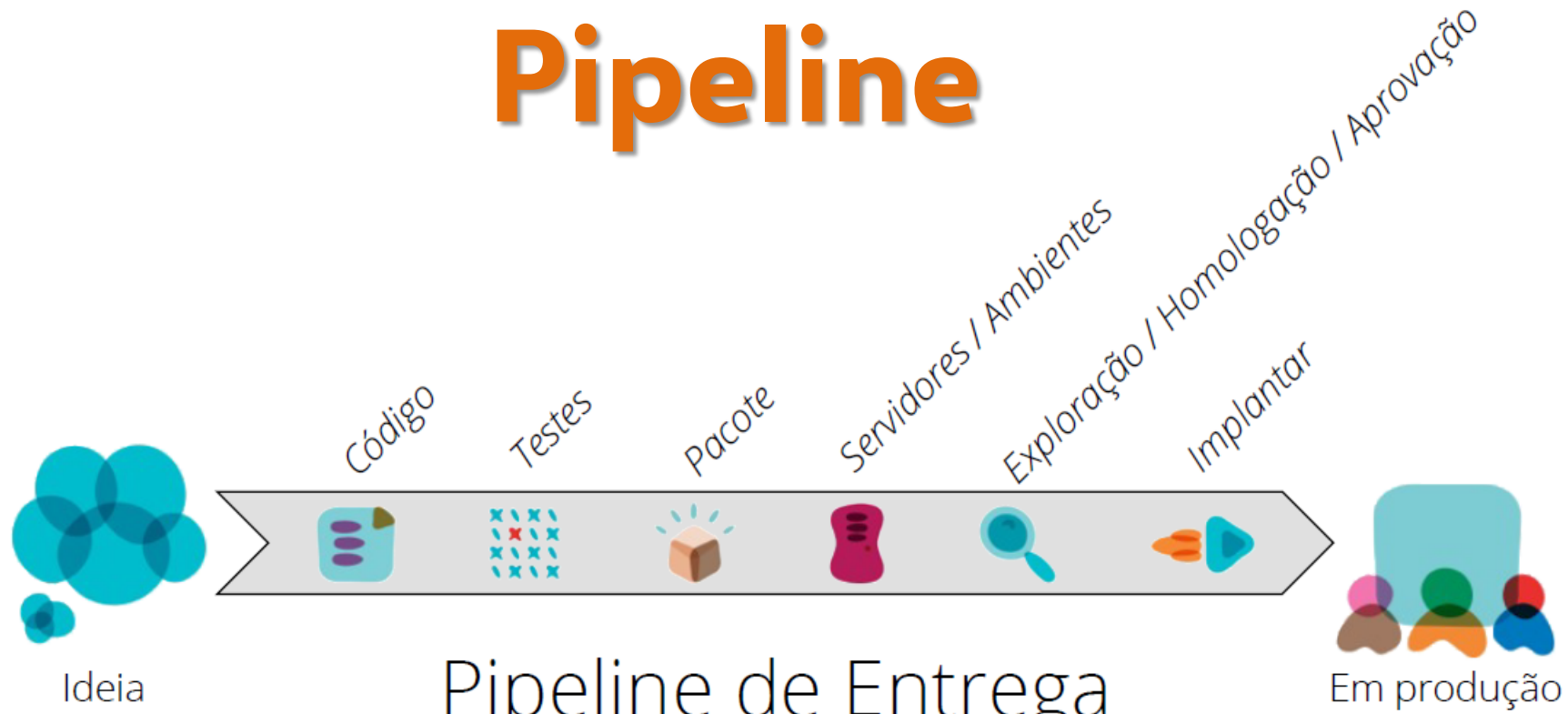
Para aprofundar os estudos...



Para aprofundar os estudos...

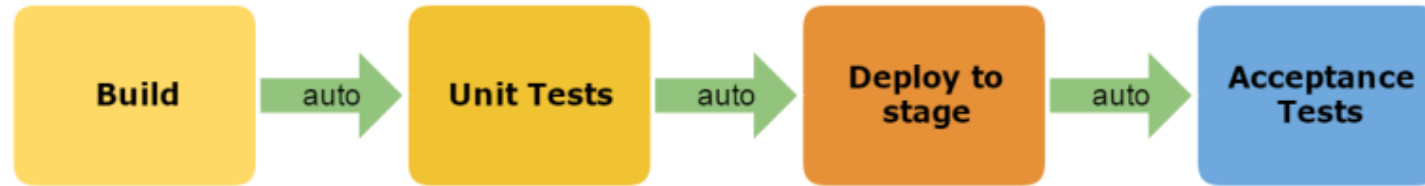


Pipeline



Integração, Deploy e Entrega Contínua

Continuous Integration



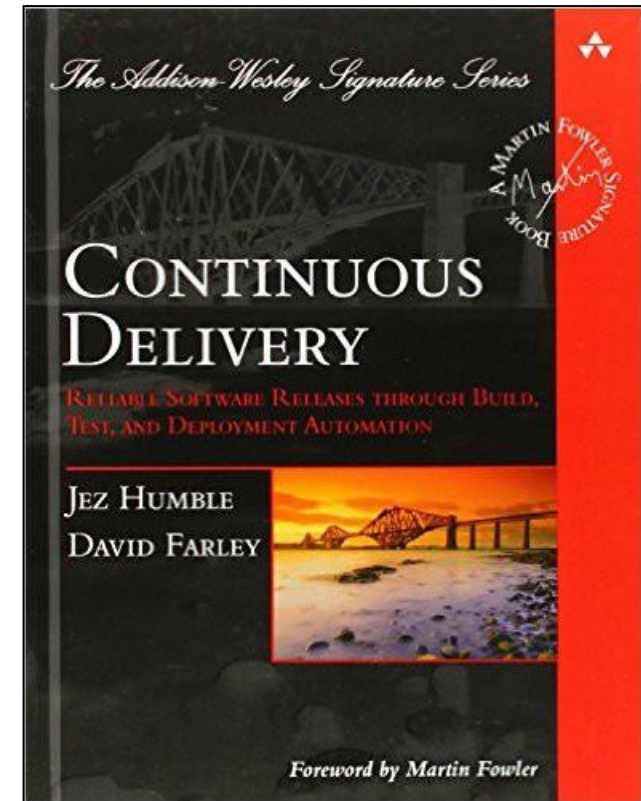
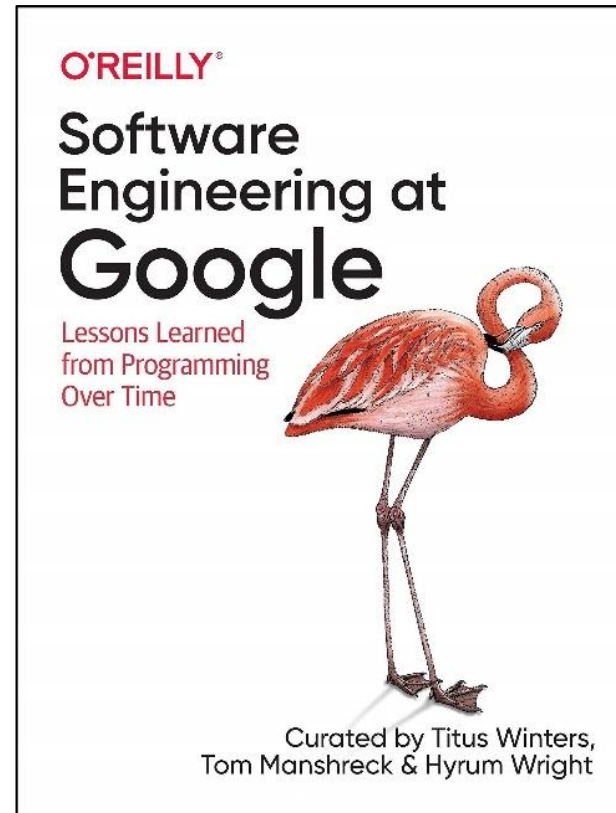
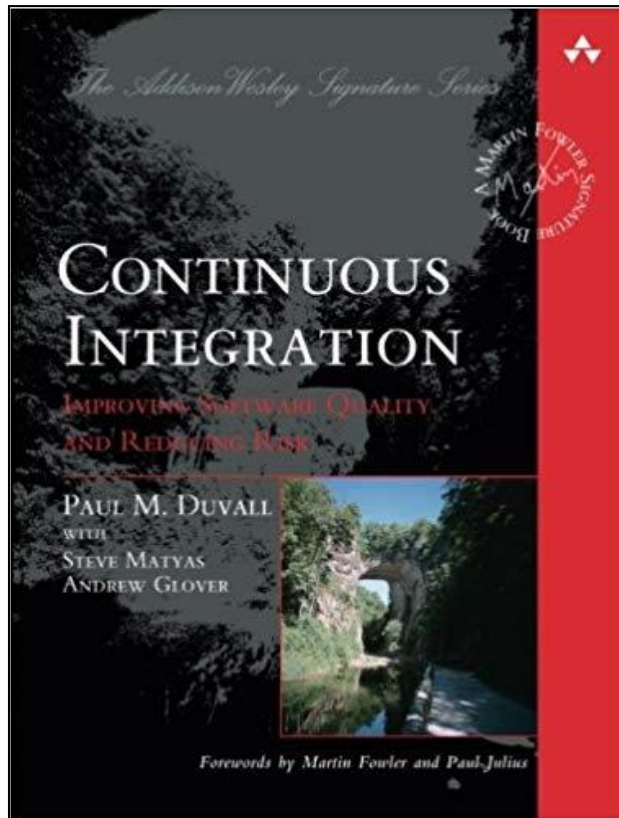
Continuous Delivery



Continuous Deployment



Para aprofundar os estudos...



Documentação Técnica

Problemas na Documentação

- Taxonomia
 - Construída a partir de mais de 800 documentos, incluindo PRs do Github e threads do StackOverflow
 - Problemas: Incompletude, falta de atualização, pouca usabilidade, pouca legibilidade
- Principais dores
 - Documentações que ferem os critérios de completude, atualidade e legibilidade
 - Falta de Corretude: documentações com informações erradas

Aghajani E. et al. (2019) **Software Documentation Issues Unveiled**. In: Proceedings of 41st ACM/IEEE International Conference on Software Engineering (ICSE 2019)

Aghajani E. et al. (2020) **Software Documentation: The Practitioner's Perspective**. In: Proceedings of 42nd ACM/IEEE International Conference on Software Engineering (ICSE 2020)



Documentação Técnica

- Estrutura
 - Conteúdo, tamanho, audiência, formato (tutorial, vídeos)
 - Contar uma história
 - Use e abuse de exemplos de código
- Elementos e exemplos
 - Diagramas (Maps, UML, entre outros)
 - Arquitetura: C4 Model, Architecture Haiku, Architecture Decision Records (ADR)
 - Mockups, protótipos navegáveis
 - Engenharia Reversa: Artefatos, Visualização de Software
 - Código: Programação Literata, comentários e documentações no código
 - Testes: documentação automatizada
 - Guias: Deployment, contribuição, instalação/uso, primeiros passos



Exercícios

- Atividade sobre boas práticas adotadas dos times de software
 - Formar duplas
 - Debata com a(o) colega sobre o que **é usado as boas práticas** discutidas em aula no trabalho do time
 - **Registre no fórum da atividade**

