Métodos de Desenvolvimento de Software

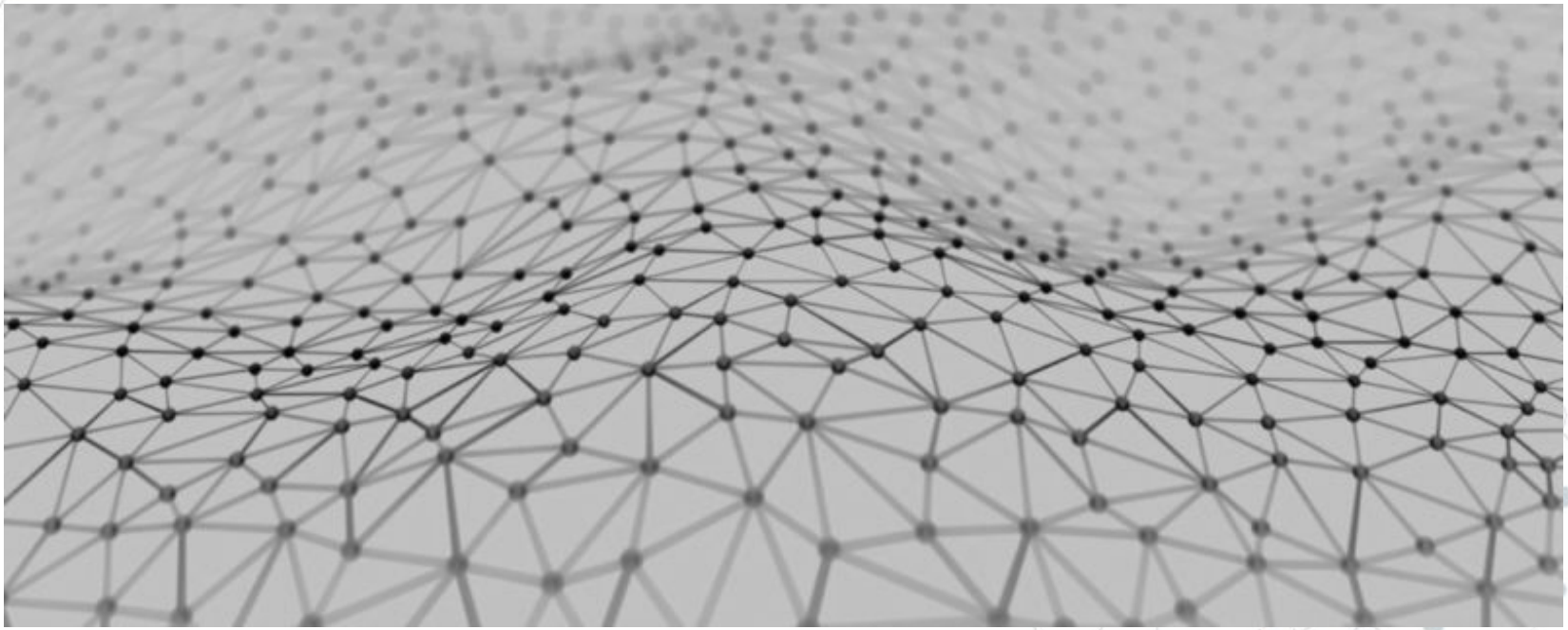
Prof. Carla Rocha

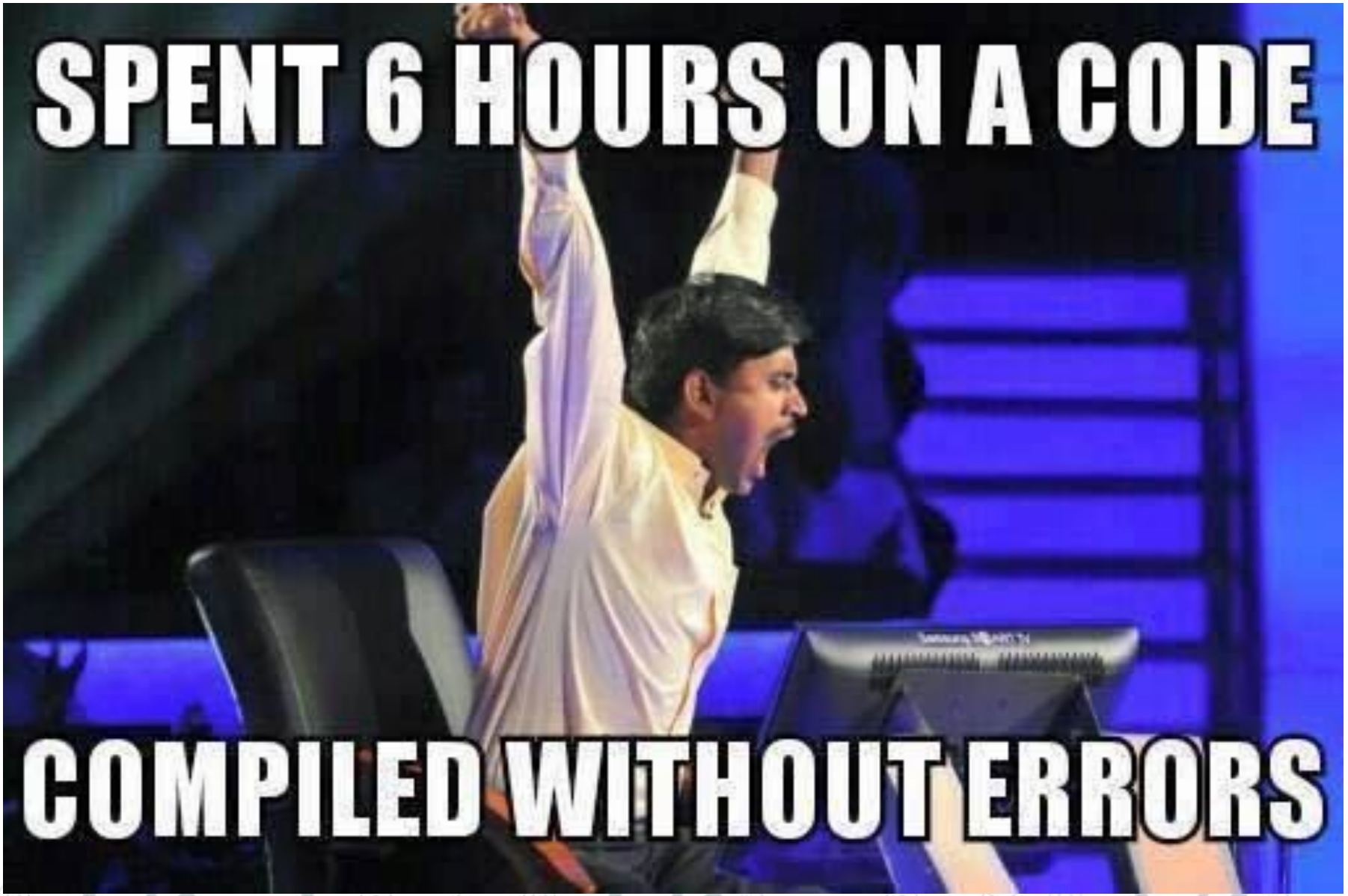
rocha.carla@gmail.com

https://github.com/fga-gpp-mds

1.

Engenharia de Software




SPENT 6 HOURS ON A CODE
COMPILED WITHOUT ERRORS

HAPPINESS IS

...when your code
runs without error.


A programmer in their natural habitat

Lição #1:

Engenharia de Software ***Não é somente*** programação!

2,000,000,000

Linhas de Código

40,000

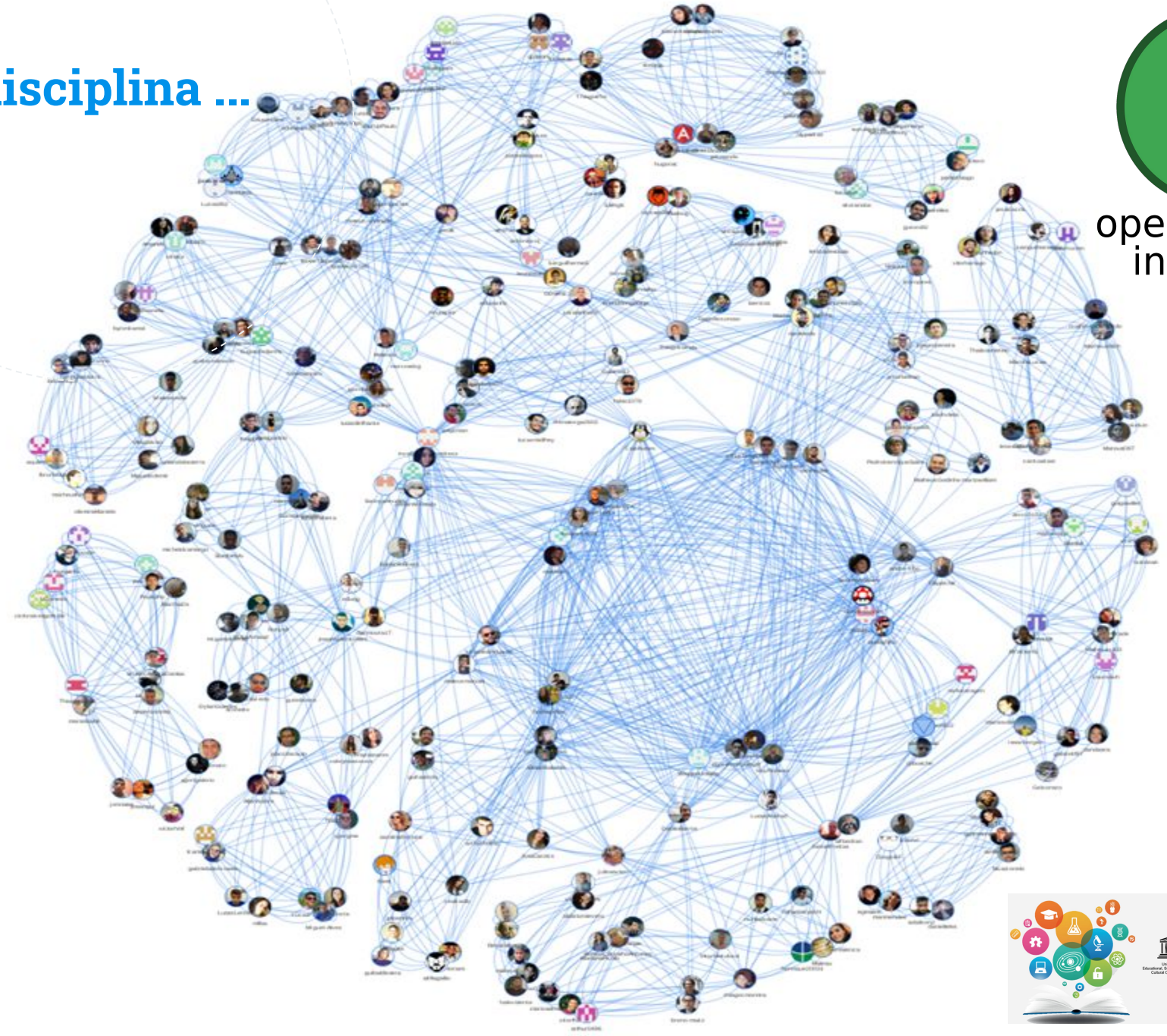
Commits por dia

Código fonte do Google

250,000

Arquivos modificado por semana!

A disciplina ...



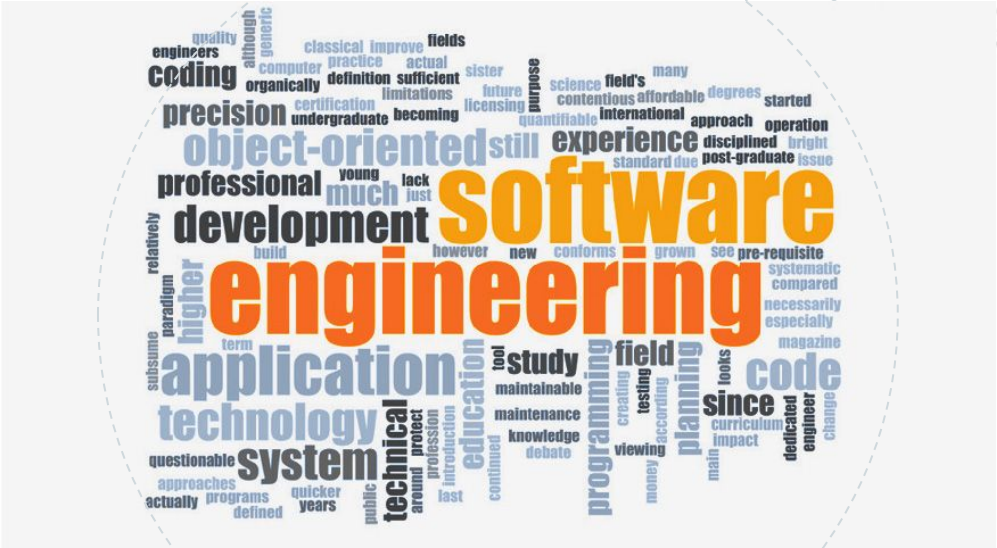
open source initiative

Open Science

O Grande Problema da Engenharia de Software

Como desenvolver sistemas de software:

- Tenha valor de mercado
- Com qualidade
- Rápidas entregas



- Fácil manutenção
- Reuso
- Modular


#3 Time

Um problema

open source initiative

Implementação

open source initiative

Mandamentos MDS

1. Se organize (não acumule trabalho!)
2. Não se apegue à linguagens (é só uma ferramenta)
3. As 6 horas semanais são sagradas!
4. Tenha calma*5
5. Ouça os Tech Leaders! (monitores)
6. Professora é facilitadora!
7. Divirta-se

Principais Problemas/Riscos na Disciplina (relatadas por grupos de outros semestres)

1. Falta de compromisso de membros (ou trabalha 6 horas por semana ou não dá para entregar o projeto)
2. Falhas na comunicação (com equipe e com os stakeholders)
3. Não ouvir a professora
4. Não assumir riscos quando necessário (reagir tardiamente)

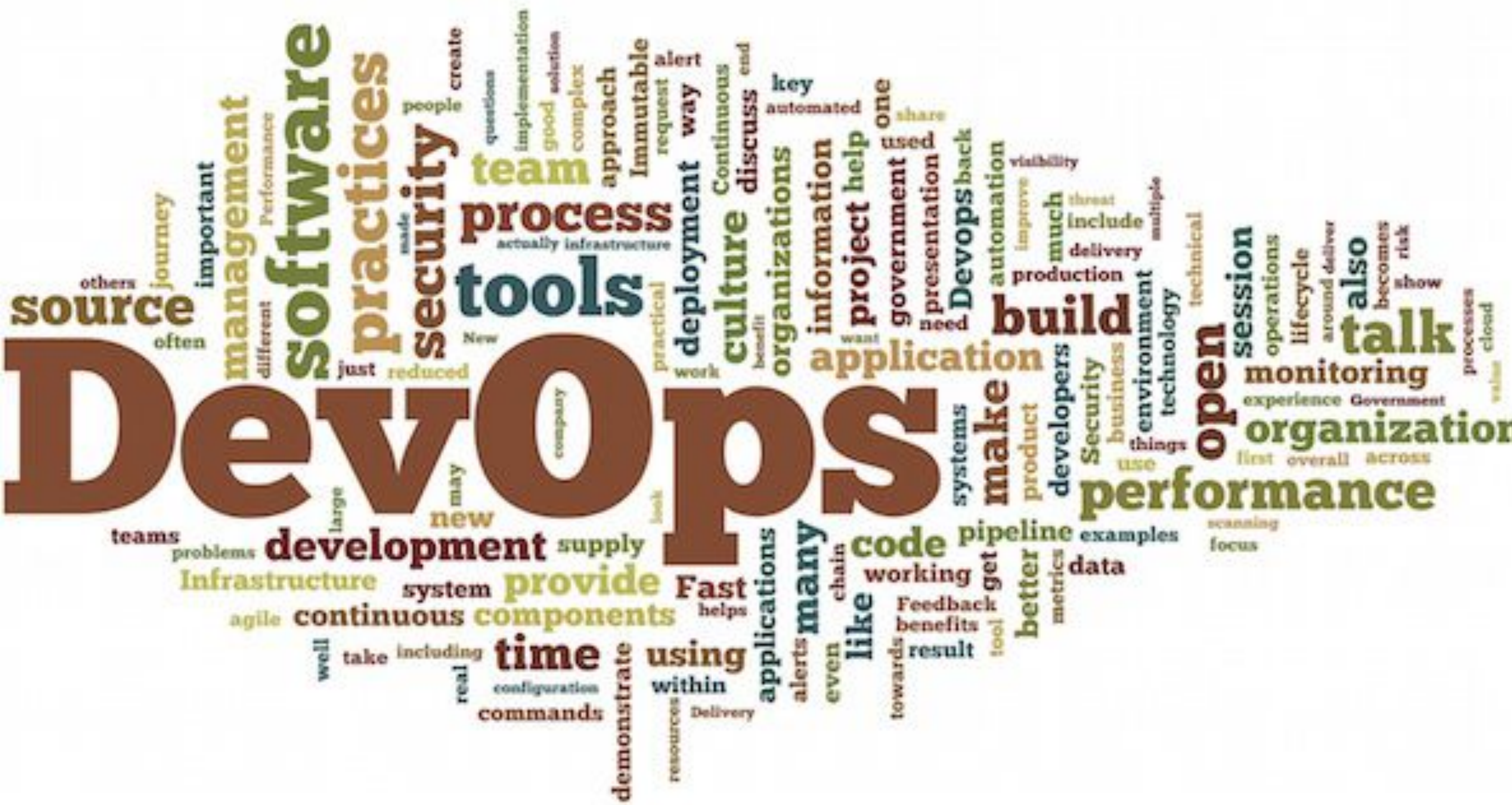
Lição #0:

"Sejamos claros: ***Sua carreira*** é ***sua*** responsabilidade, seu empregador ***não é sua Mãe***" – Robert C. Martin

Lição #2:

“Pratique uma habilidade de engenharia de software importante: ***use a ferramenta correta*** para o trabalho, mesmo que isso signifique ***aprender*** uma ***nova ferramenta*** ou uma ***nova linguagem***”

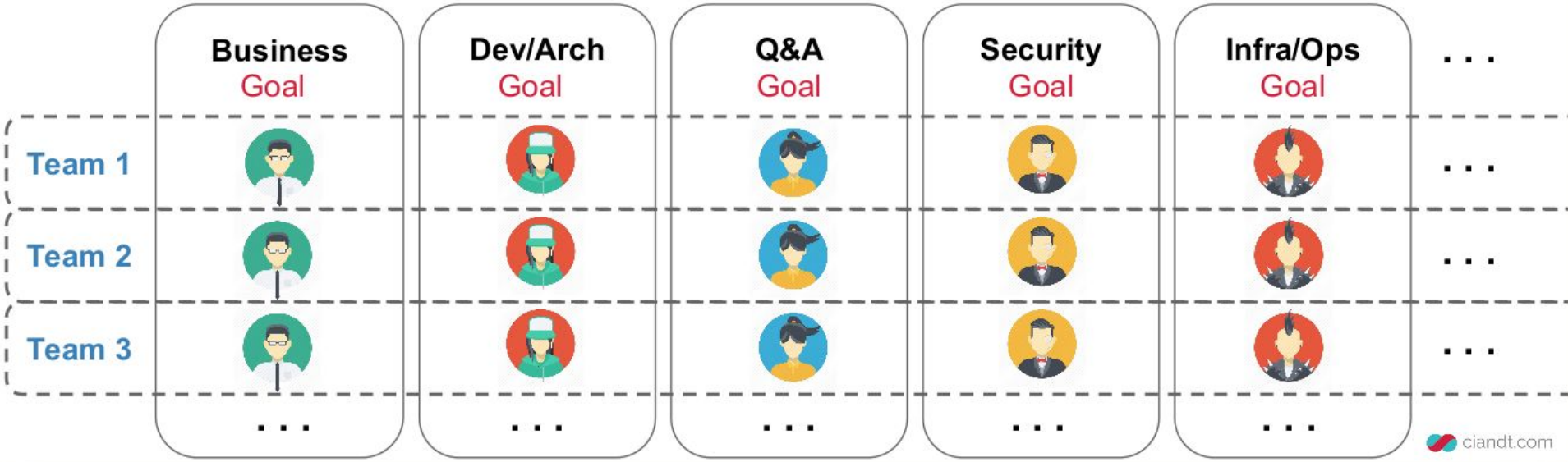

product backlog
process
burndown
speed
contract
velocity
cost
Kanban
Lean
value-driven
showcase
estimate
frequently
test-driven
iterative
Quality
continuous integration
ScrumMaster
feature
working software
self-organizing
documentation
Releasable
collaboration
retrospective
teams
Management
Done
visibility
negotiate
Agile
user story
Leadership
Adaptable
cycle
Product Owner
Simplicity
planning
Incremental
Transparency
inspect
Manifesto
daily standup
conversation


DevOps
tools
software
practices
security
process
build
talk
performance
organization
application
source
management
development
code
time
culture
deployment
team
information
project
monitoring
open
session
make
many
provide
components
continuous
Infrastructure
Fast
using
system
pipeline
applications
organizations
government
presentation
Devops
automation
production
delivery
include
much
also
lifecycle
operations
technology
environment
business
Security
developers
product
systems
better
data
working
like
get
Feedback
benefits
result
supply
new
teams
problems
agile
take
including
well
commands
within
demonstrate
approach
Immutable
way
Continuous
discuss
help
used
one
key
automated
alert
people
create
complex
important
journey
others
often
just
reduced
New
experience
Government
first
overall
across
examples
focus
metrics
towards
alerts
even
configuration
Delivery
helps
chain
things
use
processes
cloud
show
risk
becomes
around
deliver
need
want
work
practical
technical
security
visibility
threat
share
questions
implementation
solution
request
good
actually
infrastructure
Performance
different
may
large
company
real
resources
tool
scanning


OPEN
SOFTWARE
OPEN SOURCE
SOURCE
FREE
CONTENT
INFORMATION
AVAILABLE
RESEARCH
CODE
CULTURE
DEVELOPMENT
COMMUNITY
DIGITAL
INTERNET
MEDIA
TECHNOLOGY
MOVEMENT
ACCESS
COPYRIGHT
PUBLIC
COST
SCIENTIFIC
DESIGN
WORLD
PROJECTS
INCLUDING
POTENTIAL
COMMUNITIES
DISTRIBUTED
PRODUCT
MANUFACTURERS
SIMILAR
PLATFORM
ANOTHER
WORKS
COSTS
EVEN
MESSAGEBOARDS
POLITICAL
NEW
ARTICLE
MAKING
SOCIETY
BBS
COMPUTER
WORK
CITATION
PROCESS
IDEA
COMPANIES
CASE
USE
SOCIAL
USING
WELL
TERM
DATA
SYSTEMS
ONLINE
LICENSE
TECHNOLOGIES
MANY
MADE
SCIENCE
FIRST
DESCRIBE
FILM
GENERAL
CREATED
PROPERTY
FILE
INTELLECTUAL
EXAMPLES
ONE
CULTURAL
HARDWARE
INNOVATION
RAYMOND
PRODUCTION
USERS
SHARING
NETWORK
BUSINESS
CREATIVE
RESOURCES
MODEL
PRODUCTS
BLOGS
OPERATING
COMMONS
CENTURY
PROJECT
SHARED
EXAMPLE
INDIVIDUALS
BEER
LINUX
SYSTEM
ETHICS


#3 Time

Times multifuncionais (máximo de 10 pessoas por time)



Times

- PM* - Product Manager
- TL* - Tech Leader
- UX* - Designer
- QA* - Qualidade
- DV* - Devops
- DS* - Data Science
- SE** - Engenheiros de Software

O time deve ter Conhecimento e autonomia sobre determinada temática do projeto

* Desempenhado por EP

** Desempenhado por MDS


Cliente


Ter o cliente por perto agiliza o processo de decisão


É hora de rever o método

2.
Engenharia de Software

Contexto Histórico

Crise de Software (1970)

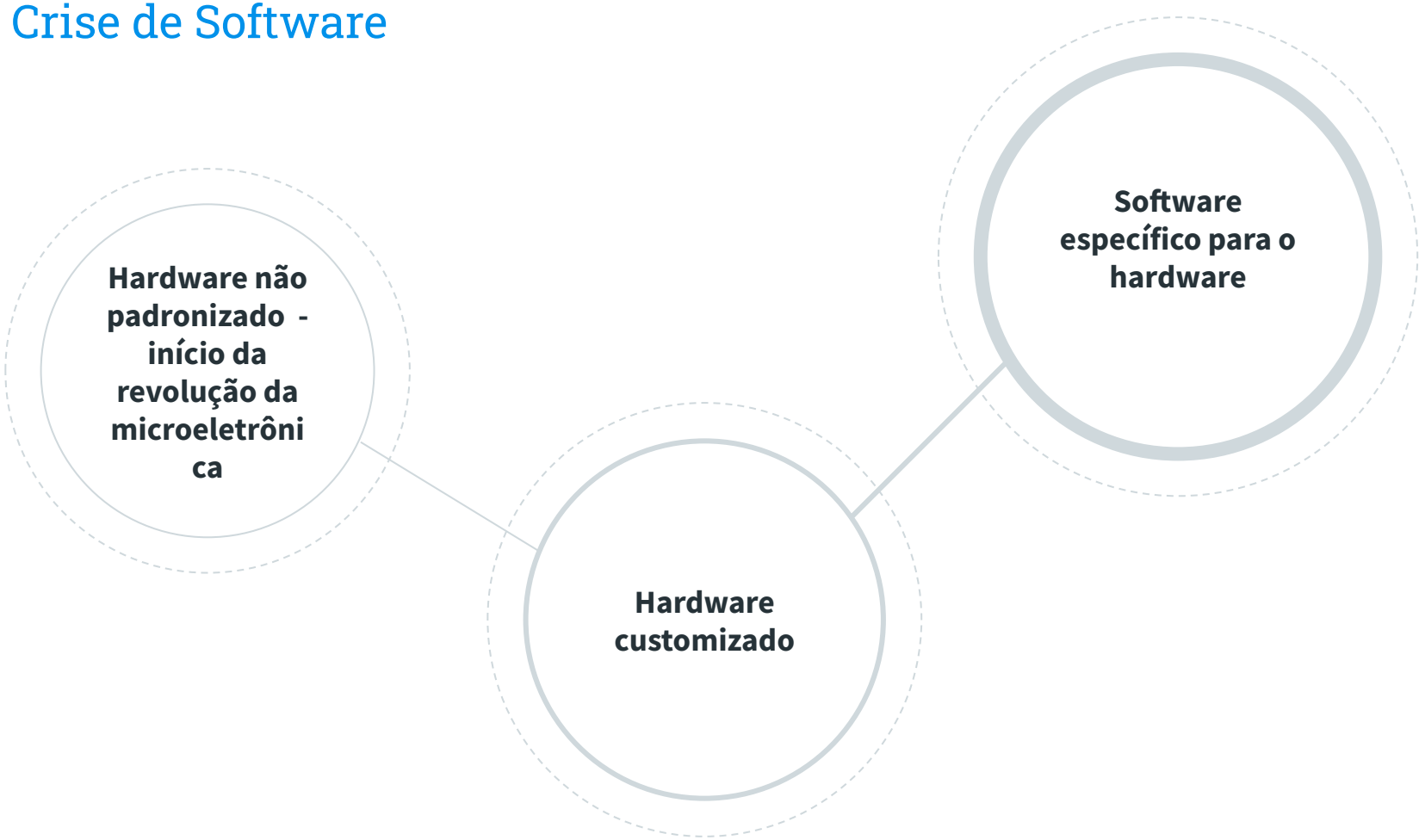
Lição #0:

"Sejamos claros: ***Sua carreira*** é ***sua*** responsabilidade, seu empregador ***não é sua Mãe***" – Robert C. Martin

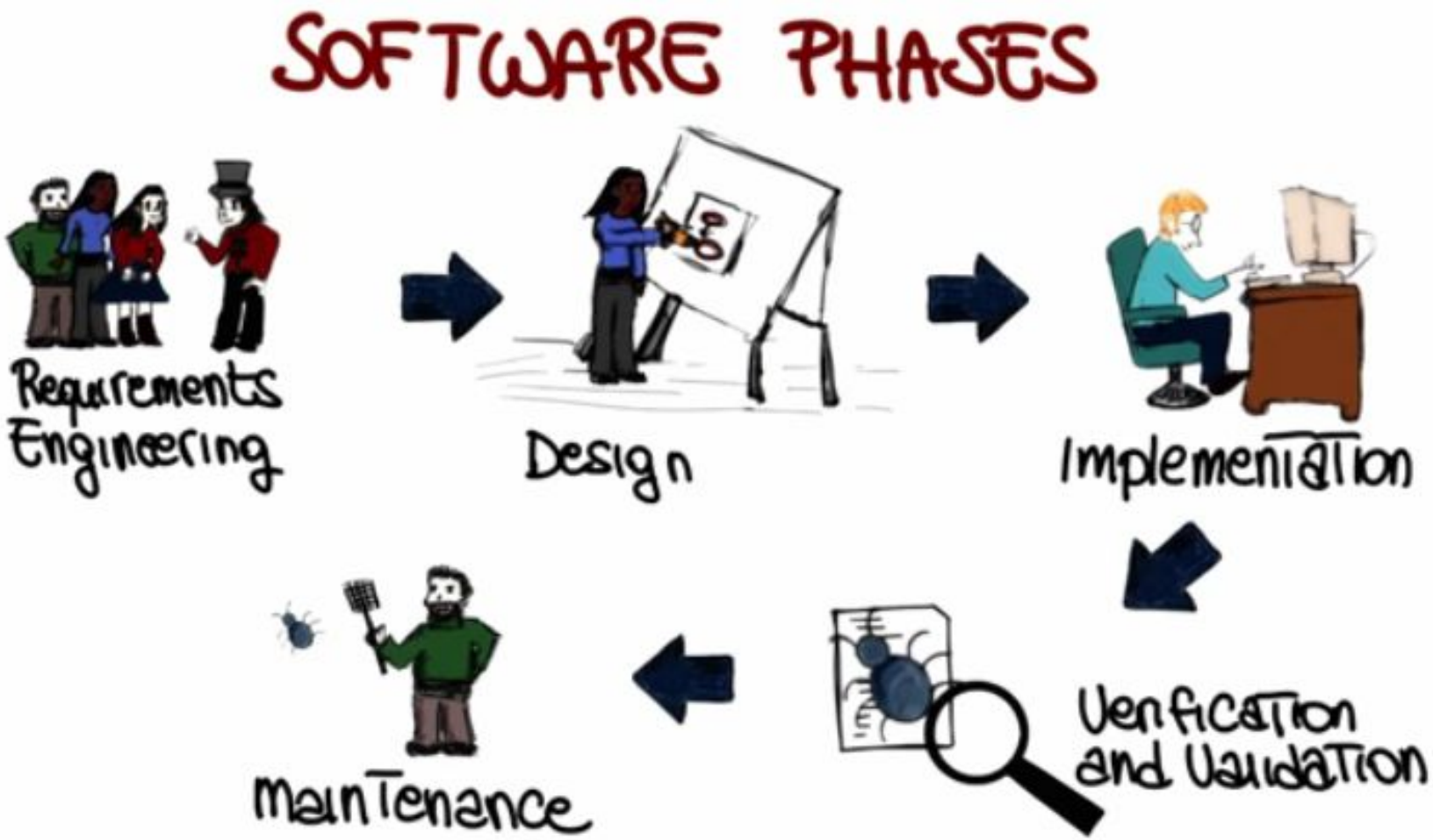
Crise de Software - Quake III Arena
Implementação da Raiz Quadrada Inversa

```
float Q_rsqrt( float number)
{
  int i;
  float x2, y;
  const float threehalfs = 1.5F;
  x2 = number * 0.5F;
  y = number;
  i = * ( int * ) &y; // evil floating point
      bit level hacking
  i = 0x5f3759df - ( i >> 1 ); // what the
      f***?
  y = * ( float * ) &i;
  y = y * ( threehalfs - ( x2 * y * y ) ); //
      1st iteration
  return y;
}
```

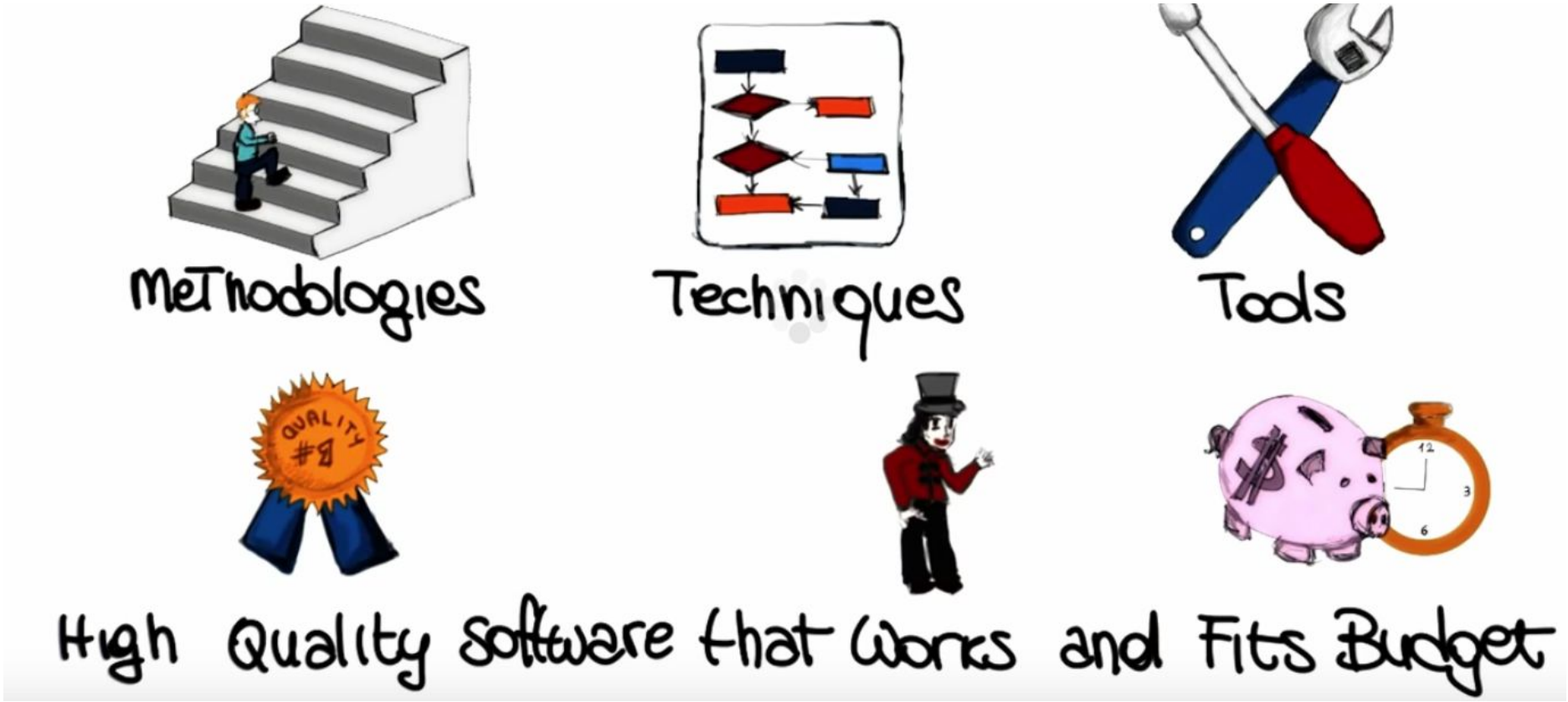
Crise de Software



Solução para a crise de Software-
Sistematizar projetos de software (Fases do Software)



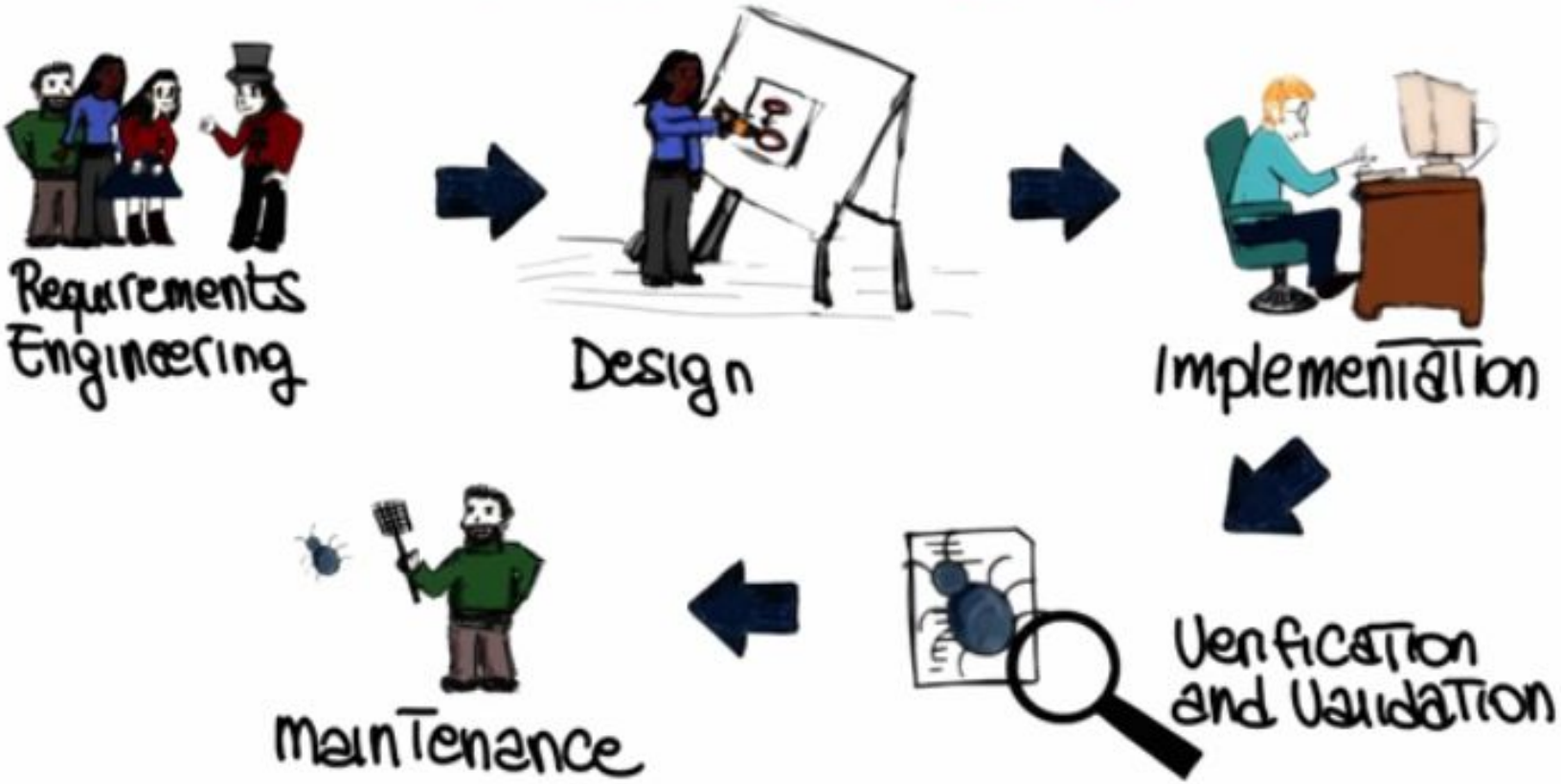
Solução para a crise de Software-
Processo de Desenvolvimento de Software



3.
Engenharia de Software

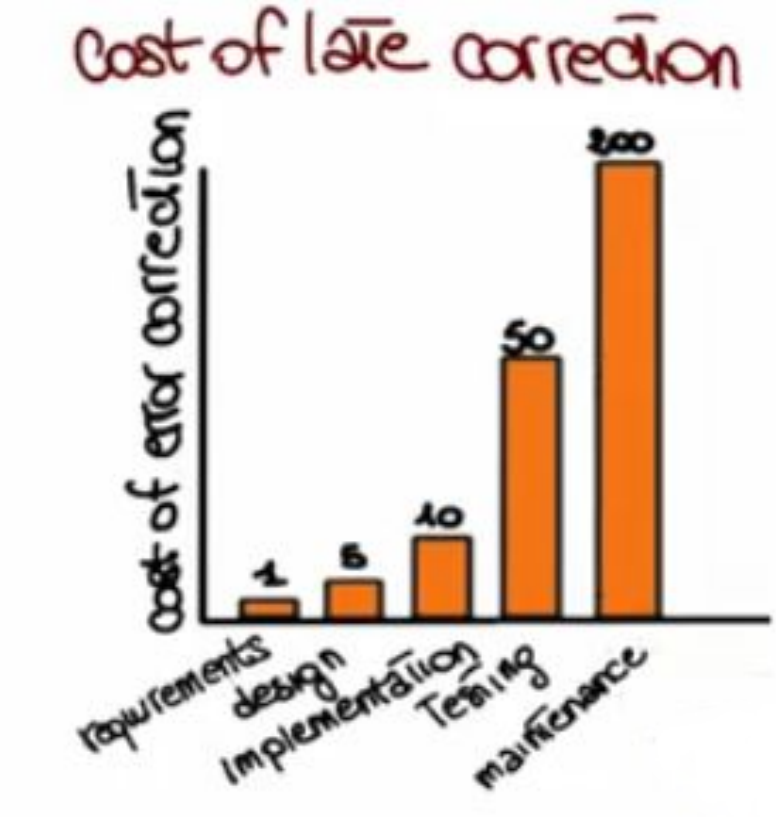
Fases de um Sistema de Software

SOFTWARE PHASES

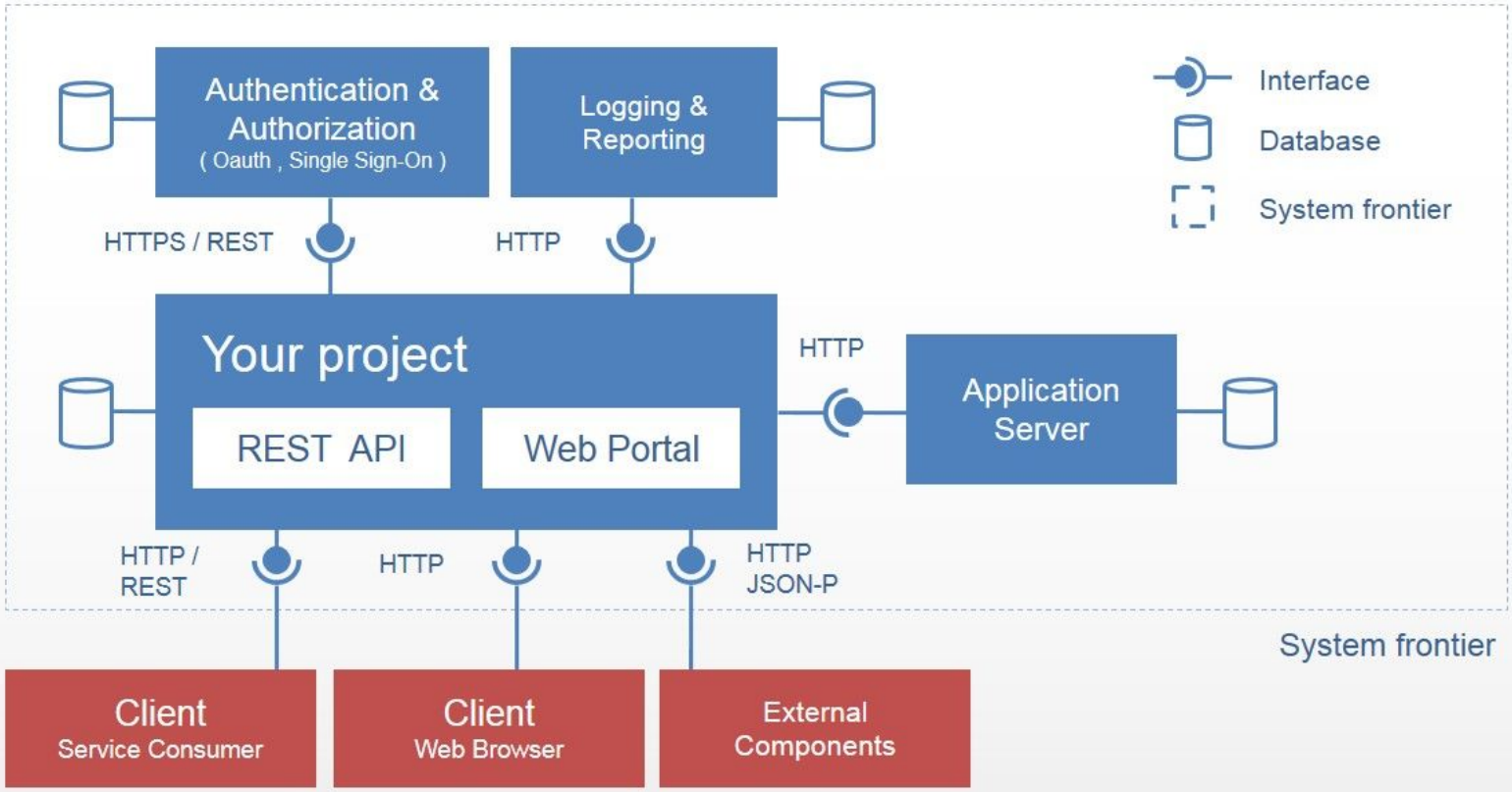


- Entender o Problema a ser solucionado
- Levantar as necessidades
- Listar Funcionalidades (features) a serem desenvolvidas

Engenharia de Requisitos

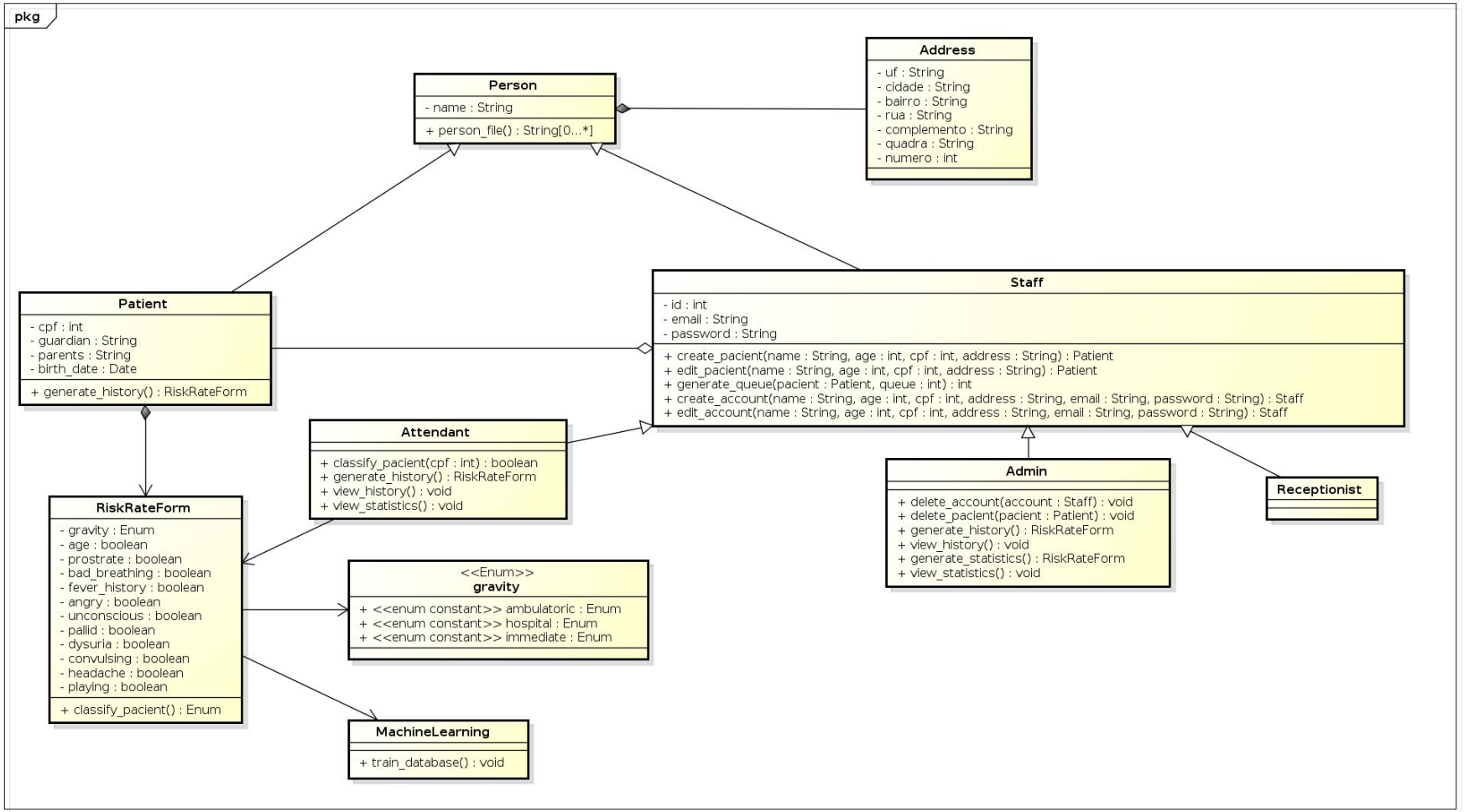


Arquitetura (Design)
Estrutura Alto nível do Sistema de Software



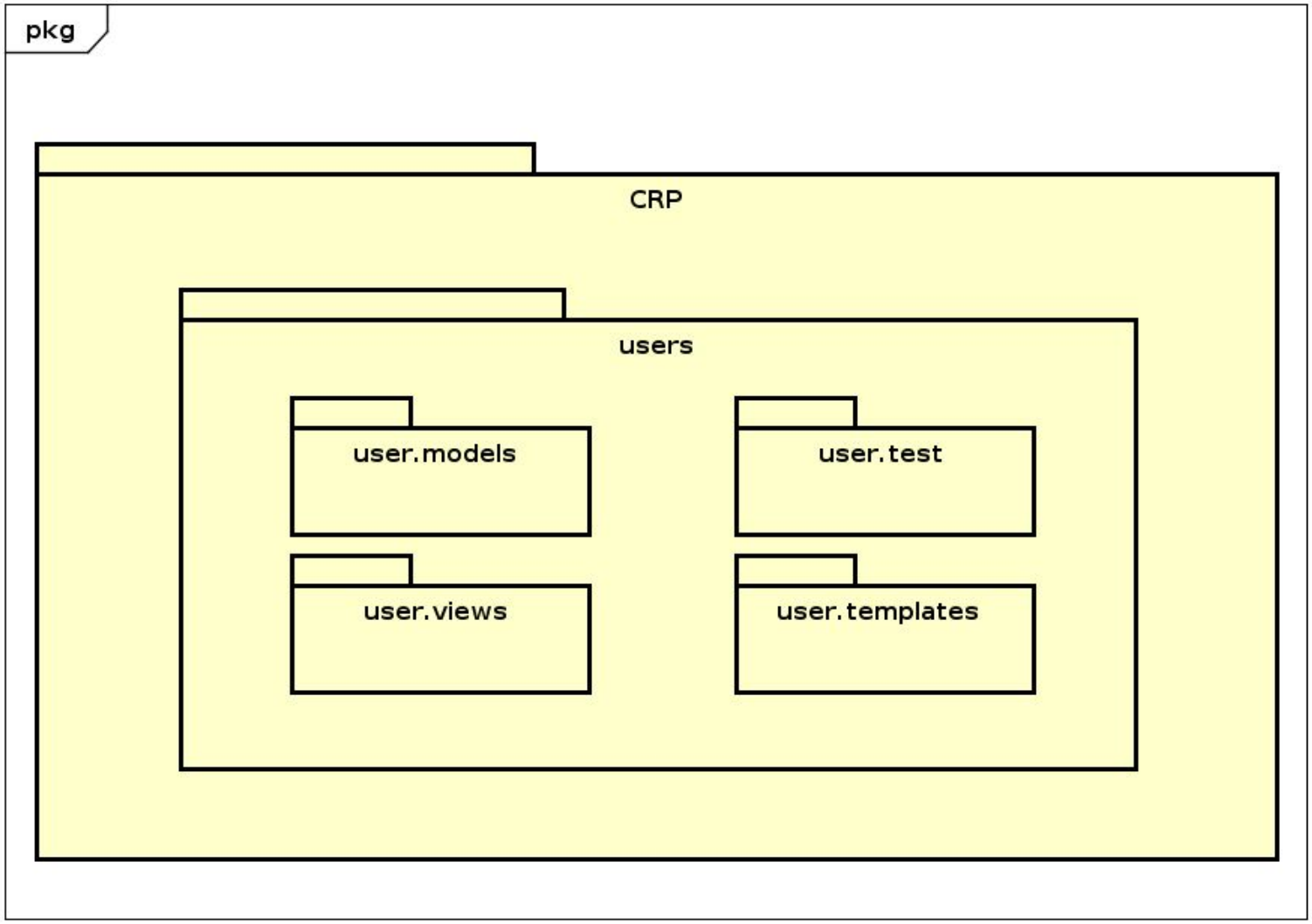
Arquitetura (Design)
Estrutura Alto nível do Sistema de Software

Diagrama de Classes



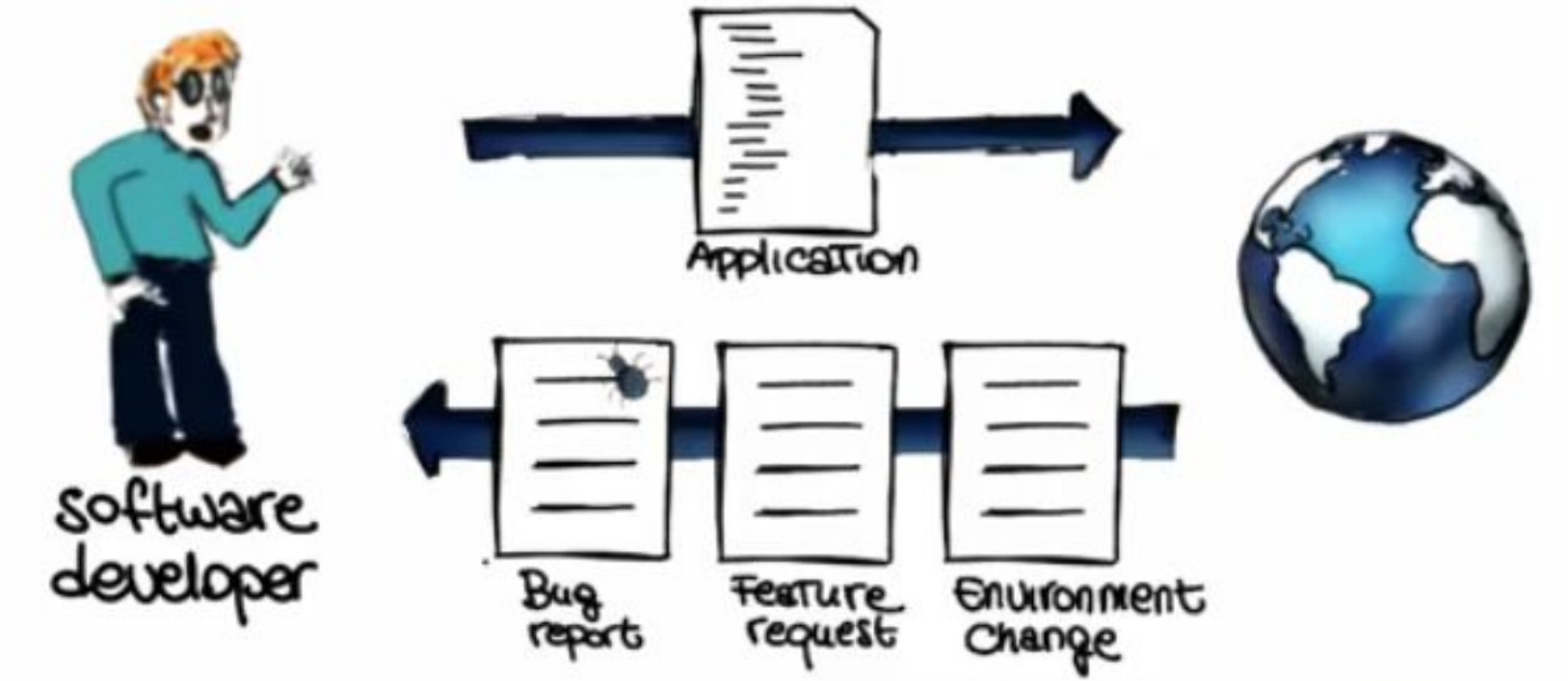
Arquitetura (Design)
Estrutura Alto nível do Sistema de Software

Diagrama de Pacotes

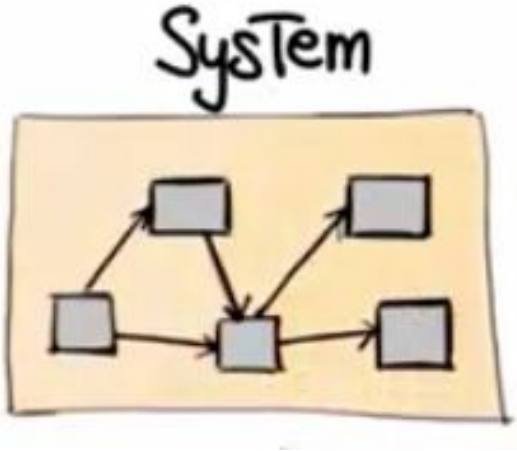


Implementação

Implementação



Verificação e Validação



Manutenção

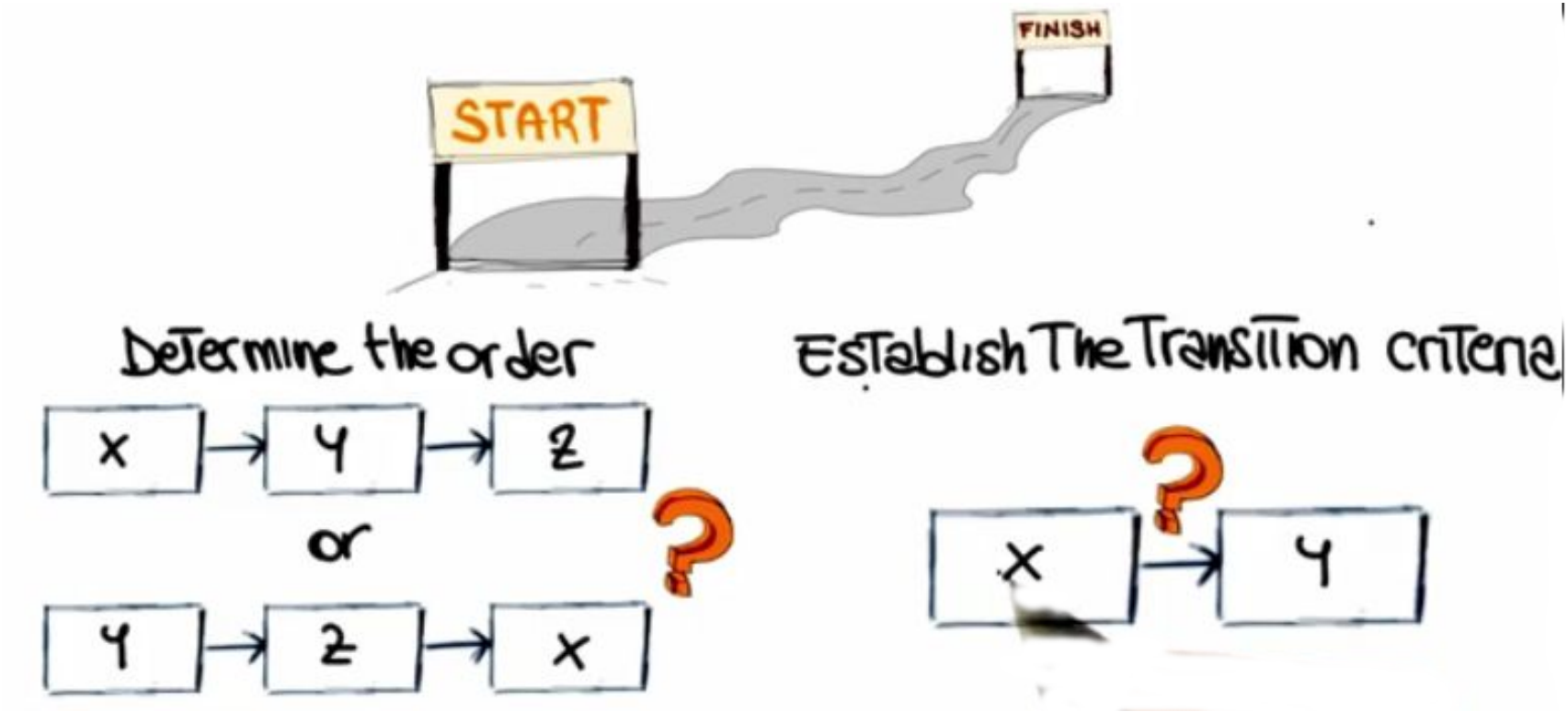

software developer
Application
Bug report
Feature request
Environment Change

4.

Engenharia de Software

Processo de Desenvolvimento

Processo de Desenvolvimento de Software - Diversas formas de executar as fases do sistema de Software



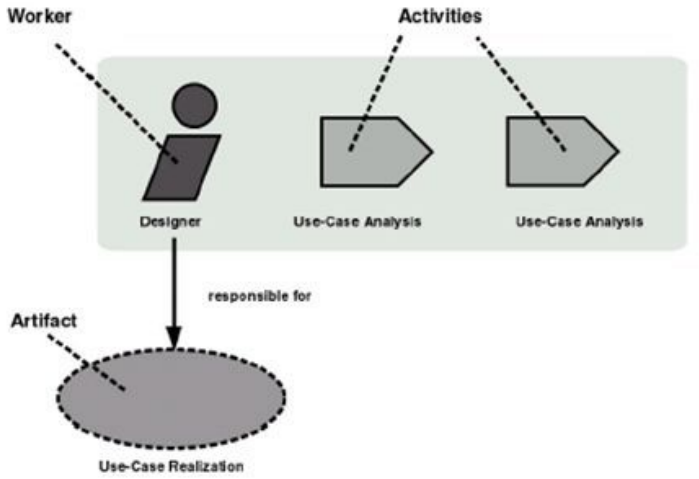
Processo *é uma série de etapas que envolve atividades que transforma entradas (insumos) em saídas (produtos)*

Método *é o caminho pelo qual fazemos algo, de maneira a atingir um objetivo; exige a organização do conhecimento e experiências prévias. (LEOPARDI, 1999)*

Processo de Software *consiste em uma série de* ***atividades***, ***práticas***, ***eventos***, ***ferramentas*** *e* ***métodos*** *que garantem, técnica e administrativa que o software pode ser desenvolvido com* ***qualidade*** *e de* ***maneira organizada***, ***disciplinada*** *e* ***previsível***

Estruturo do processo:

Intervenientes (*Workers*) - Quem? (*who*)
Atividades (*Activities*) - Como? (*how*)
Artefatos (*Artifacts*) - O Que? (*what*)
Fluxo de Trabalho (*Workflows*) - Quando? (*when*)

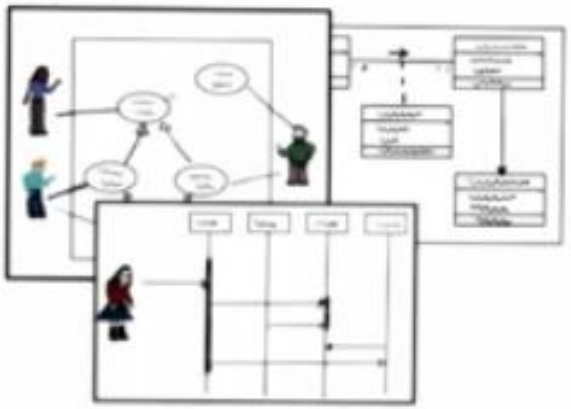



Product Manager & Developers
Grow & Work in a crossfunctional TEAM
Self organizing team
HIGHEST PRIORITY
CUSTOMER SATISFACTION
& high process product quality
visual
AGILE
MANIFESTO
© Pritam Sen
shorter development iteration
continuos inspection
sustainable development pace
measure development progress
working documentation
BUILD PRODUCT around
continuos delivery
Identify Understand and Harness the forces of change
KEEP it simple
CHANGING REQUIREMENTS ARE Welcome
effective FACE to FACE communication
motivated individuals

SOFTWARE PROCESS

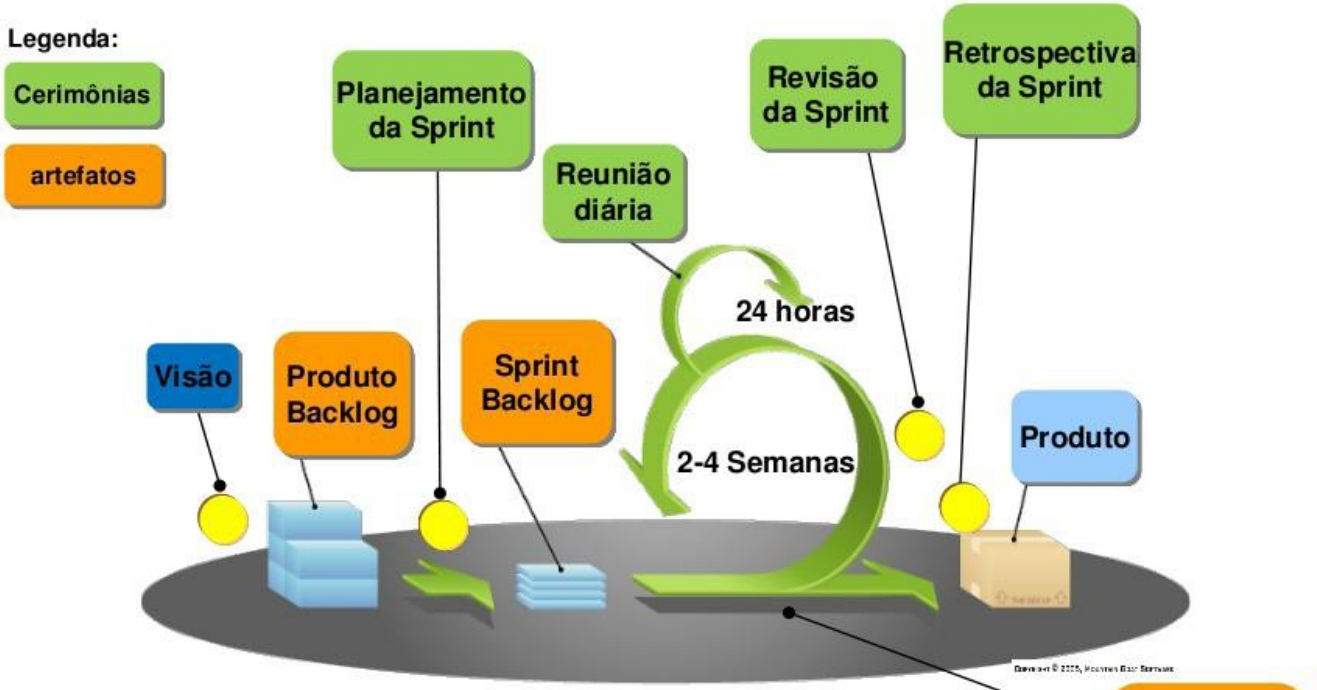
WATERFALL

EVOLUTIONARY PROTOTYPING



RUP
USP

AGILE

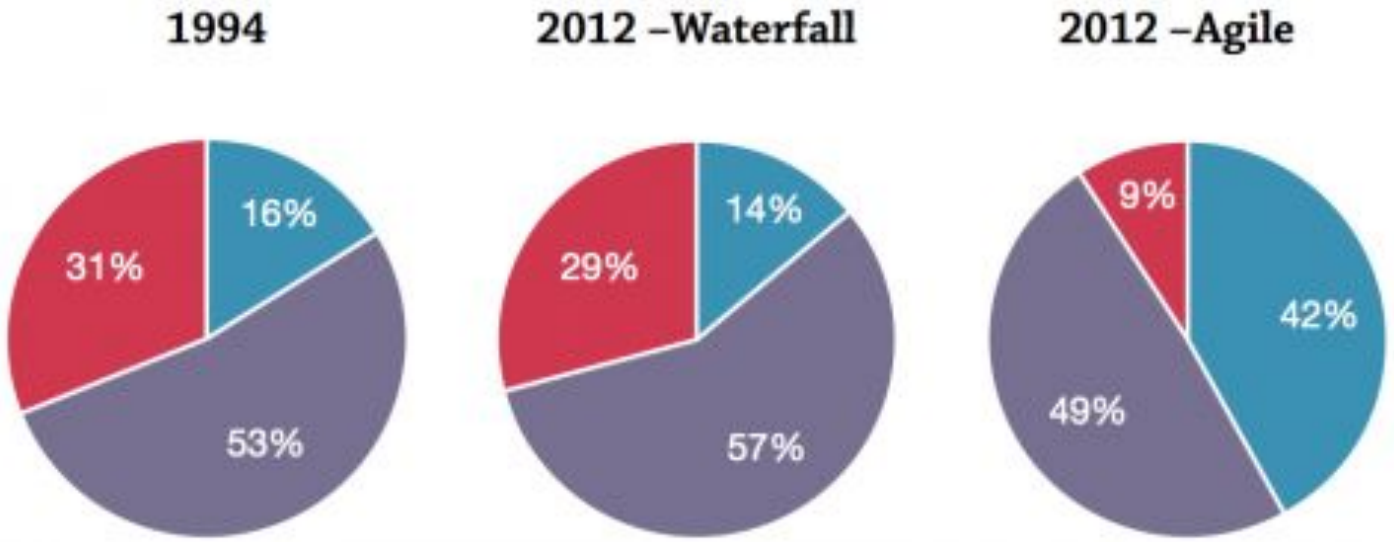

Legenda:
Cerimônias
artefatos
Planejamento da Sprint
Revisão da Sprint
Retrospectiva da Sprint
Reunião diária
24 horas
2-4 Semanas
Visão
Produto Backlog
Sprint Backlog
Produto
Papéis
• Product Owner (PO)
• ScrumMaster (SM)
• Equipe Scrum
Cerimônias
• Planejamento da Sprint
• Reunião Diária
• Revisão da Sprint
• Retrospectiva da Sprint
Artefatos
• Product Backlog
• Sprint Backlog
• Burndown (gráfico)
Burndown

Importância de processo de software - Sucesso de Projeto de Software

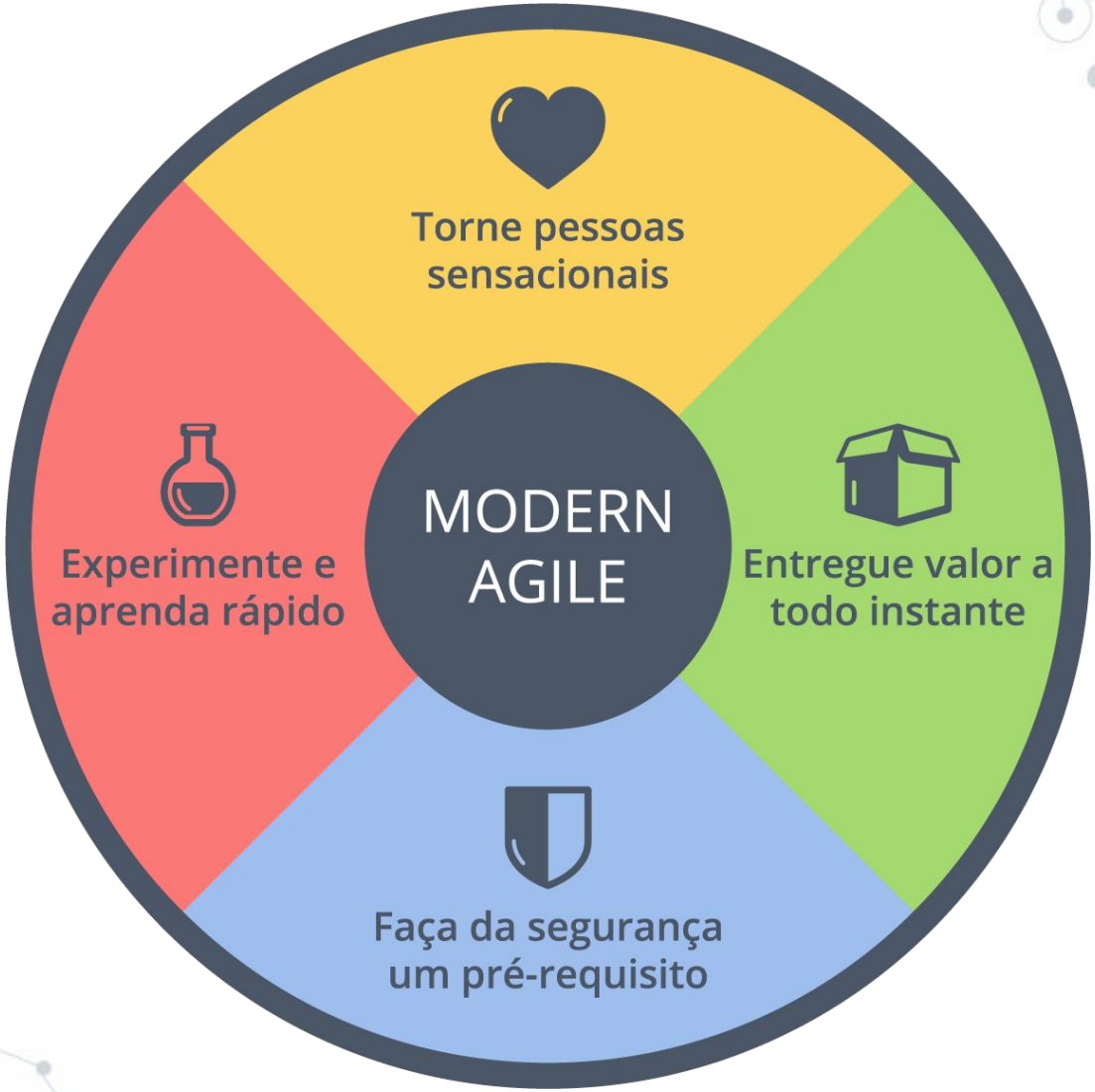
Azul - Projeto com entrega com sucesso (tempo, orçamento, escopo)

Roxo - Entrega incompleta

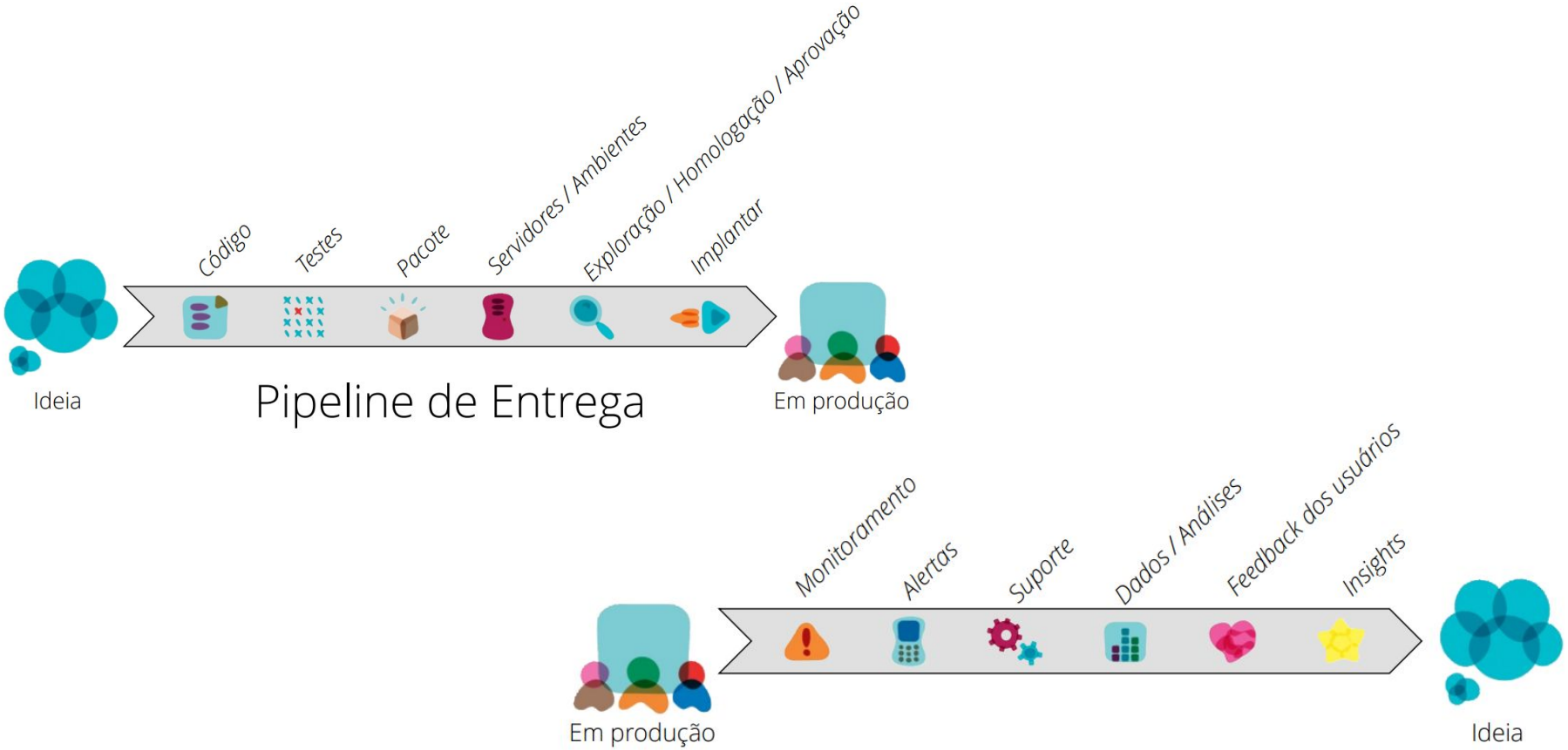
Rosa - Falha de software

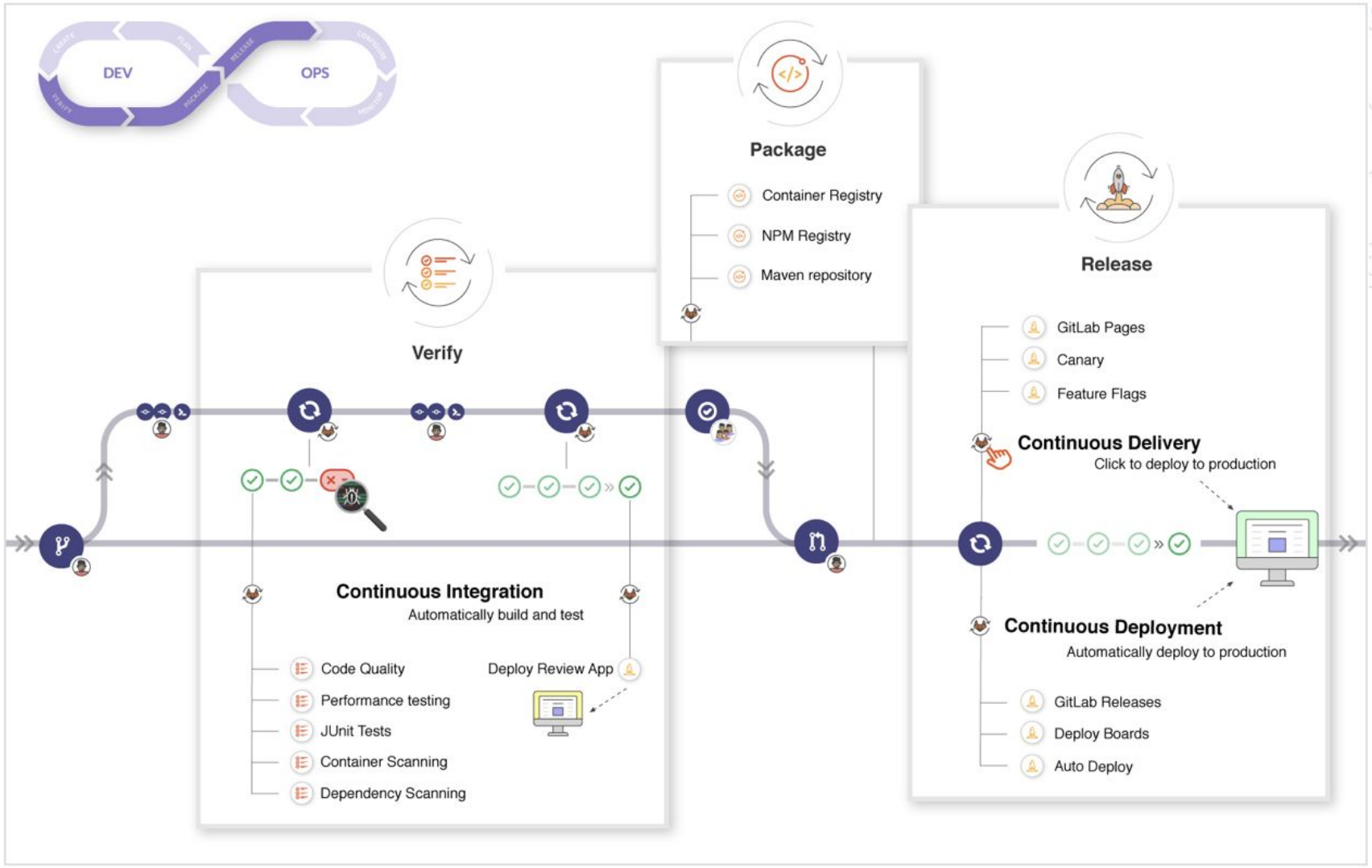


Métodos Ágeis

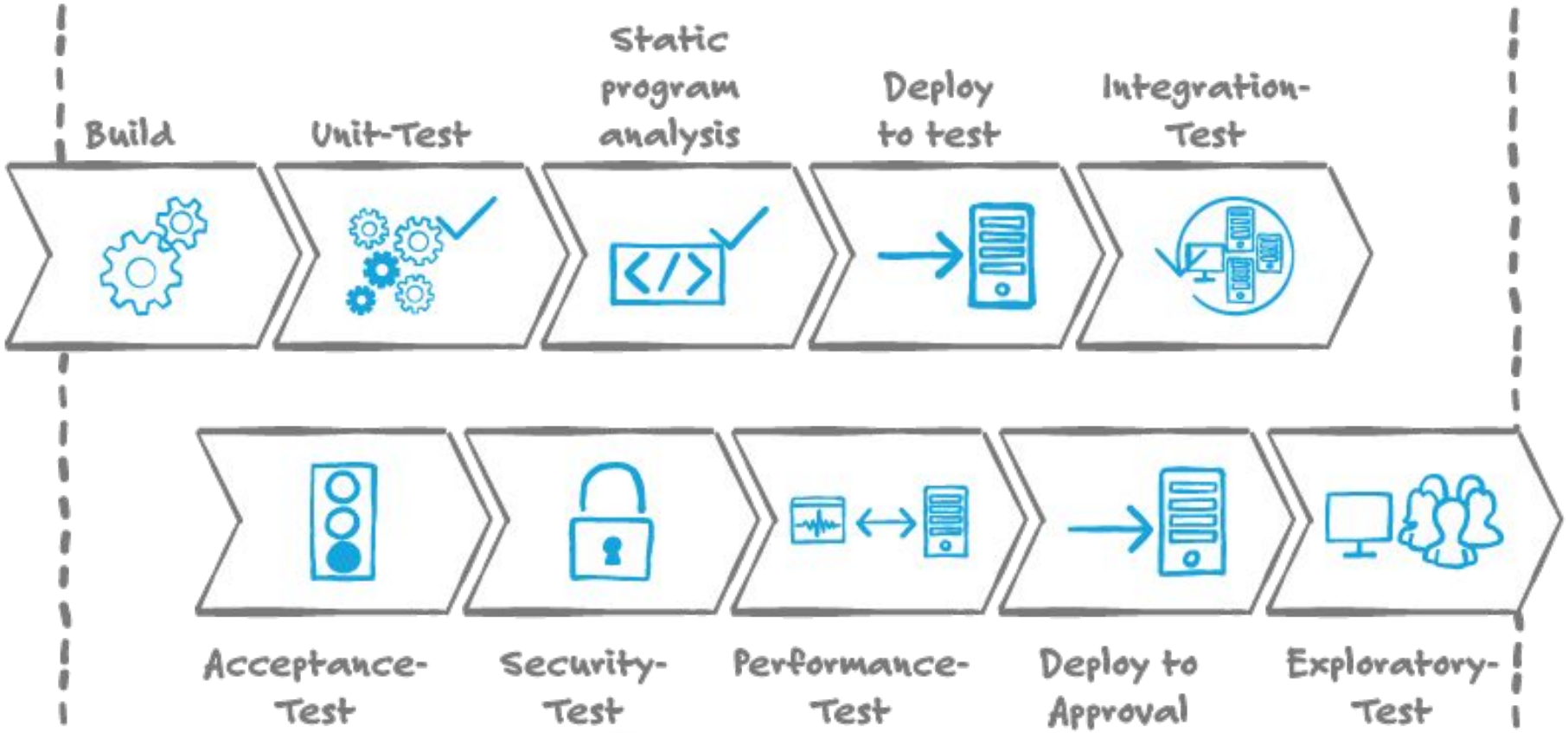


Pipeline de Entrega

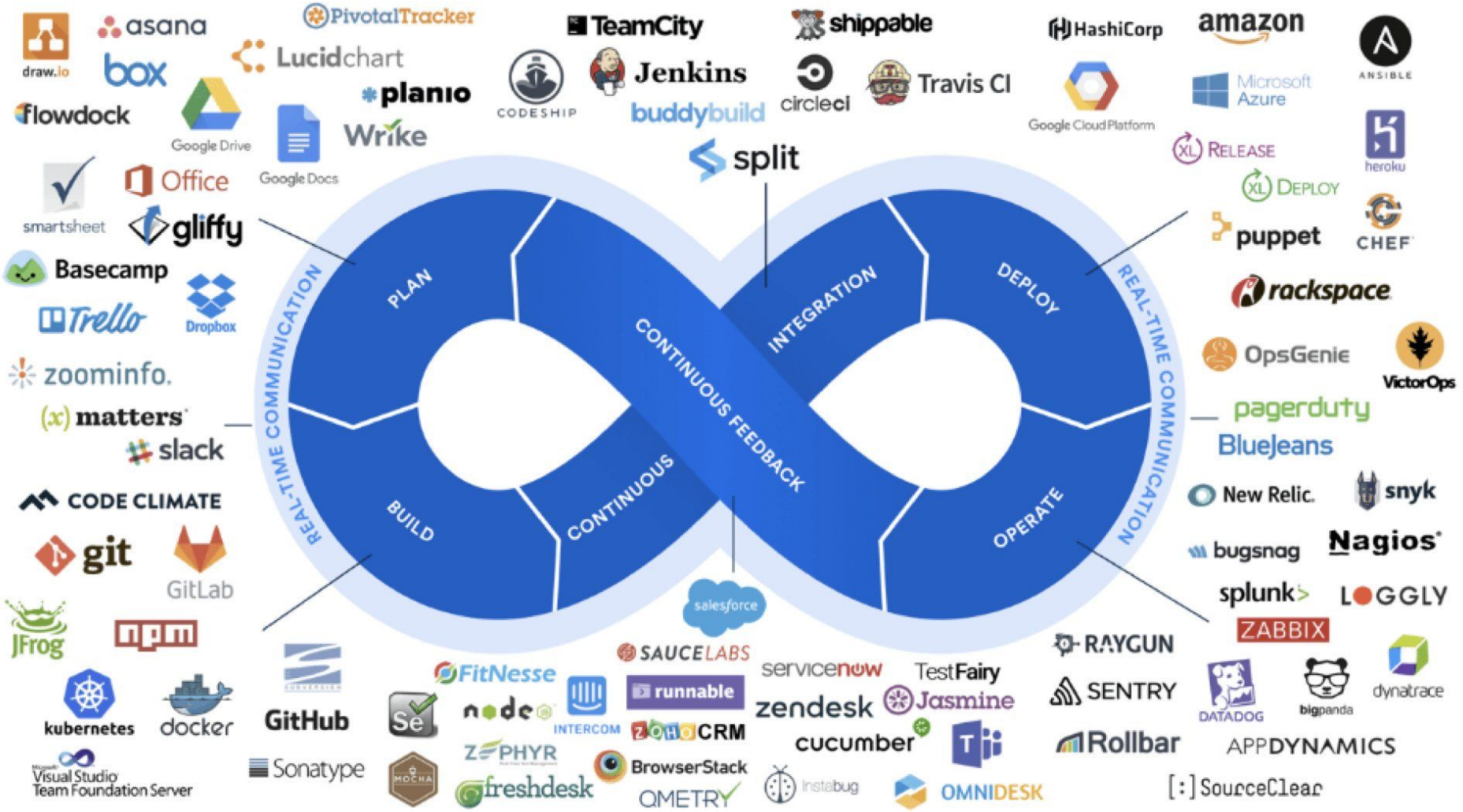



DEV
OPS
Package
Container Registry
NPM Registry
Maven repository
Release
GitLab Pages
Canary
Feature Flags
Continuous Delivery
Click to deploy to production
Verify
Continuous Integration
Automatically build and test
Code Quality
Performance testing
JUnit Tests
Container Scanning
Dependency Scanning
Deploy Review App
Continuous Deployment
Automatically deploy to production
GitLab Releases
Deploy Boards
Auto Deploy

Etapas do pipeline customizadas



Automação -
Chave da entrega contínua/ágil



Lição #2:

“Pratique uma habilidade de engenharia de software importante: ***use a ferramenta correta*** para o trabalho, mesmo que isso signifique ***aprender*** uma ***nova ferramenta*** ou uma ***nova linguagem***”

Projeto *é um conjunto* ***único*** *de atividades planejados para gerar um* ***produto específico*** *dentro de um* ***intervalo de tempo*** *e* ***recursos limitados***

As 7 Dimensões dos Projetos de Software

1. Pessoas
2. Funcionalidades
3. Qualidade
4. Ferramentas
5. Tempo
6. Valor
7. Processo

Processo

sprint planning

stand-up meetings

código fonte coletivo

velocidade

retrospectivas

planning poker

spikes

Funcionalidade

minimal marketable features

envolvimento do cliente

estórias de usuário

backlogs

demonstrações

critérios de aceitação

“inch-deep, mile-wide”

Ferramentas

sticky notes

quadros brancos

builds diários

controle de versão

integração contínua

testes automatizados

burn charts

Pessoas

cross-functional

colocation

interações

respeito

times pequenos

colaboração

responsabilidade

confiança

auto-organização

Qualidade

test-driven development

excelência técnica

definição de pronto

design emergente

refactoring

programação em par

simplicidade

Tempo

rolling wave planning

timeboxes

potentially shippable products

planejamento de release

iterações

sprints

ritmo sustentável

Valor

embracing change

incrementos

priorização

mapeamento de valor

feedback

entregas frequentes

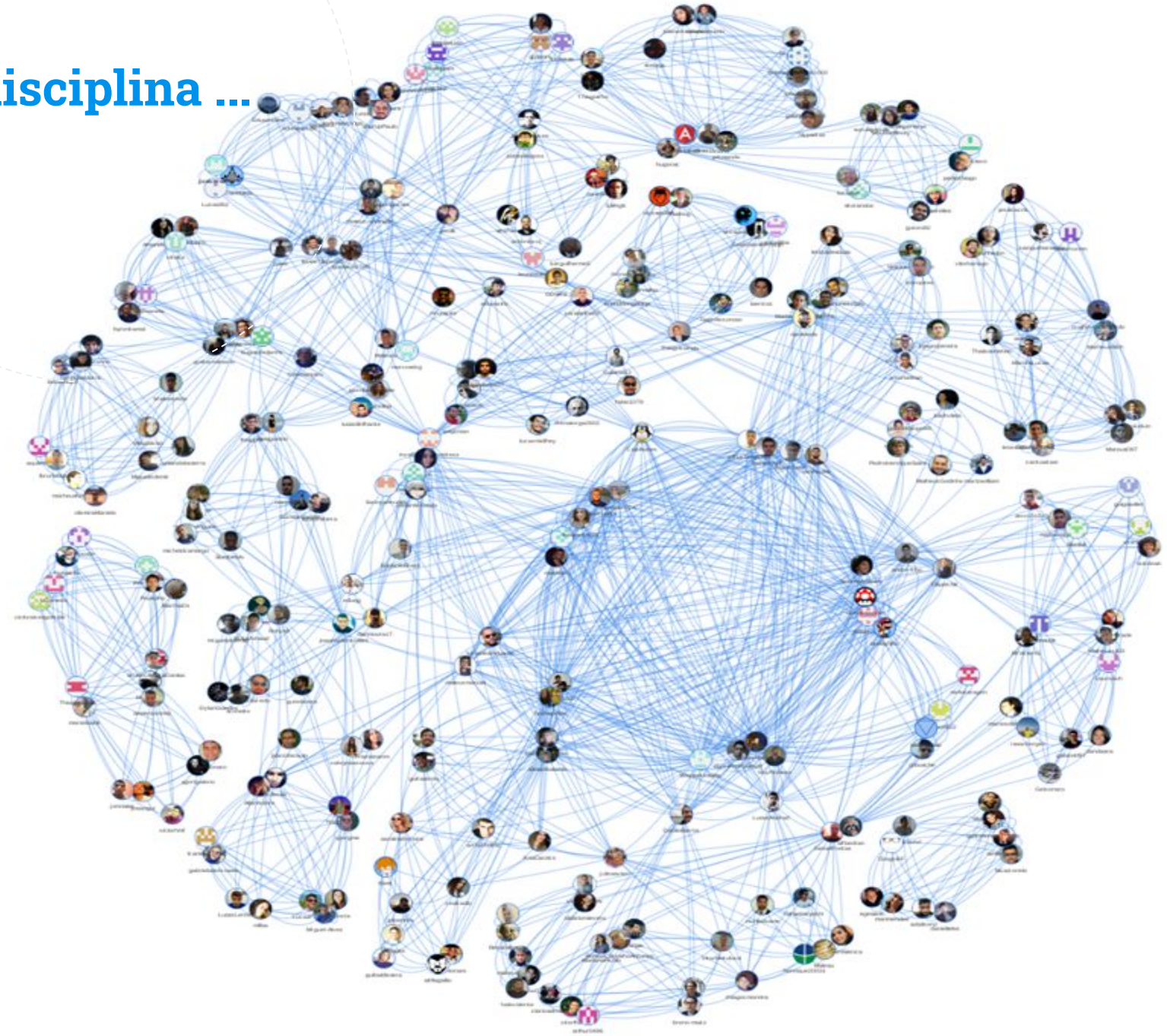
value streams

Dinâmica

[3 minutos por pessoa]

- Seu nome
- Uma curiosidade sobre você
- Uma coisa que você é ótima/o
- O que espera da disciplina?

A disciplina ...

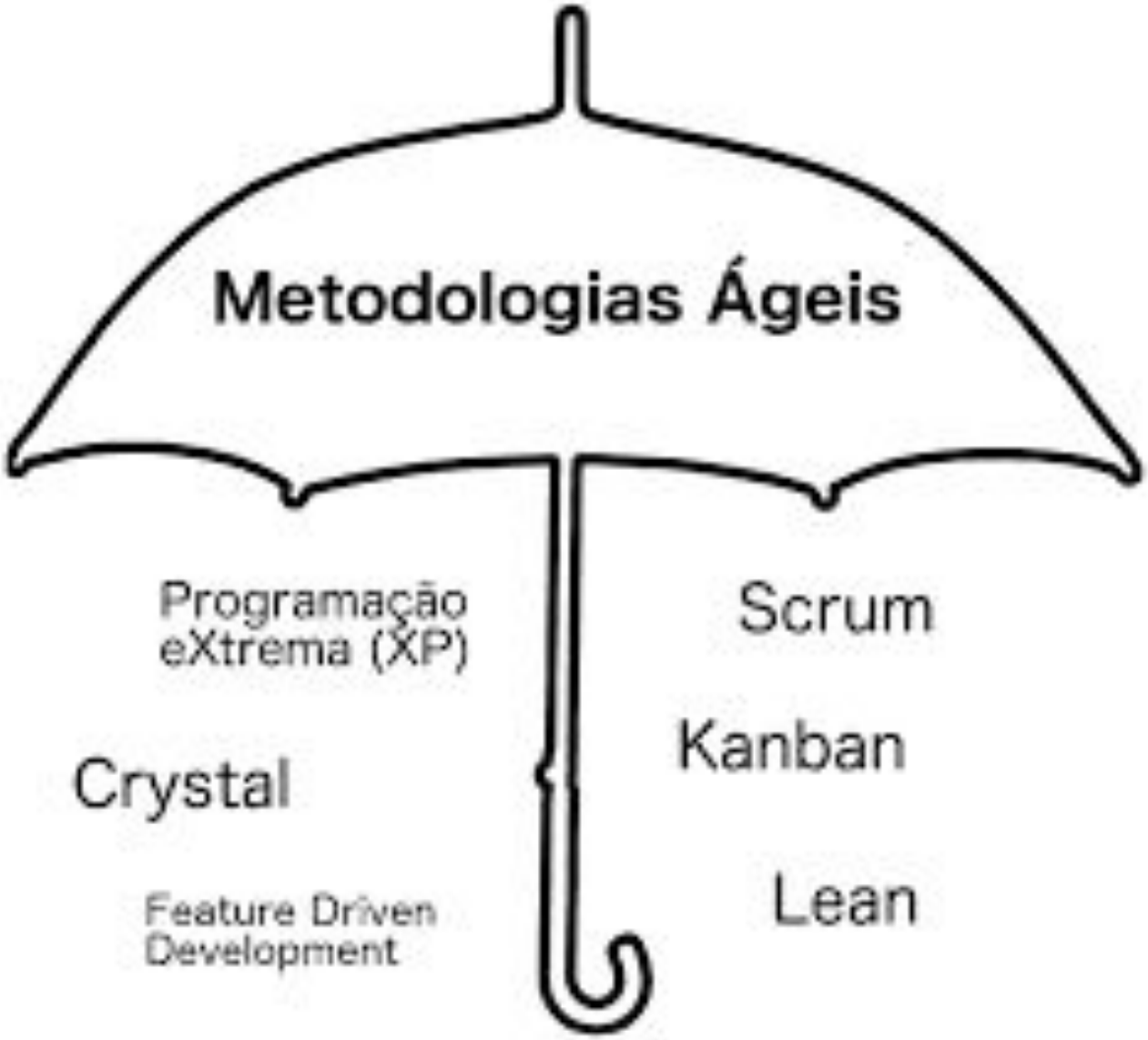


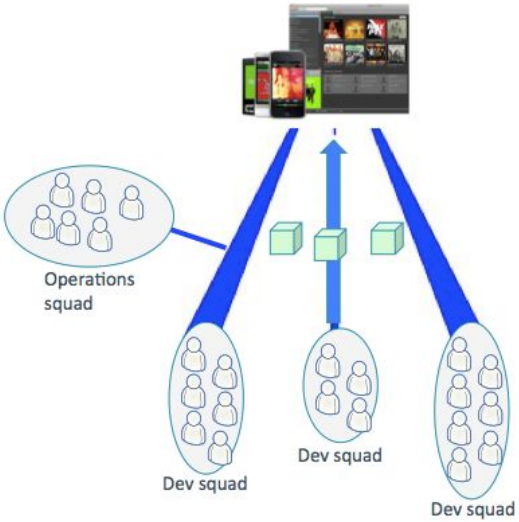
Lição #3:

"Sejamos claros: ***Sua carreira*** é ***sua*** responsabilidade, seu empregador ***não é sua Mãe***" – Robert C. Martin

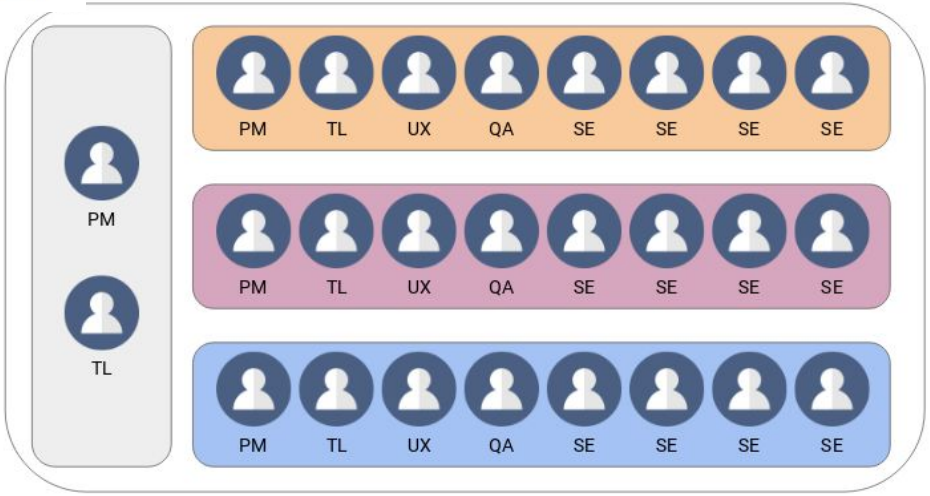
Modelo de Processo

- Vamos aplicar métodos e práticas ágeis
- Vamos adaptar lean startup, scrum, XP, Kanban
- Vamos aplicar práticas XP, práticas Devops
- Vamos aderir a alguns artefatos do RUP/PMBOK (que se adequa ao problema)
- Vamos aplicar arquitetura de Microsserviços

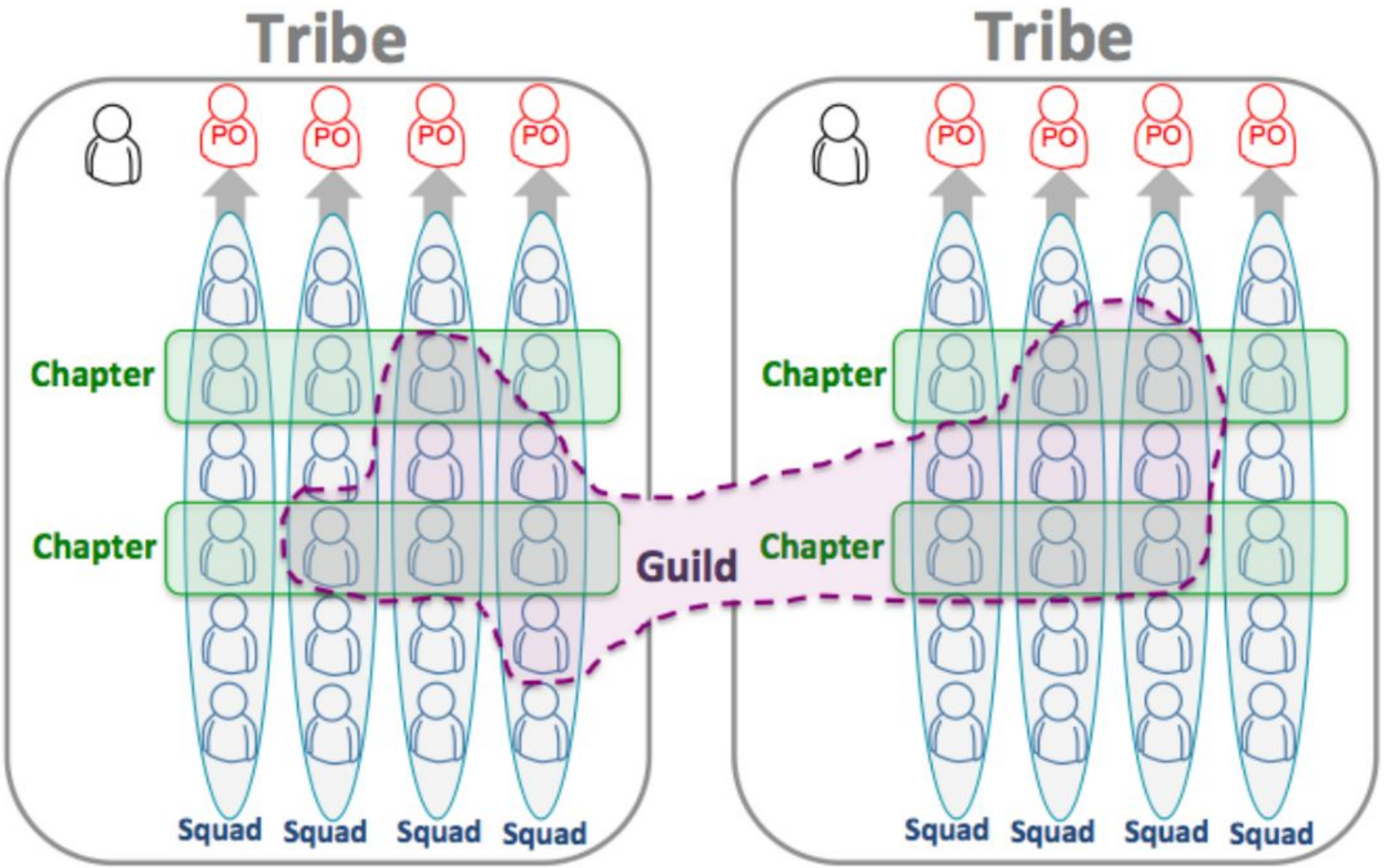

Metodologias Ágeis
Programação eXtrema (XP)
Scrum
Kanban
Crystal
Lean
Feature Driven Development

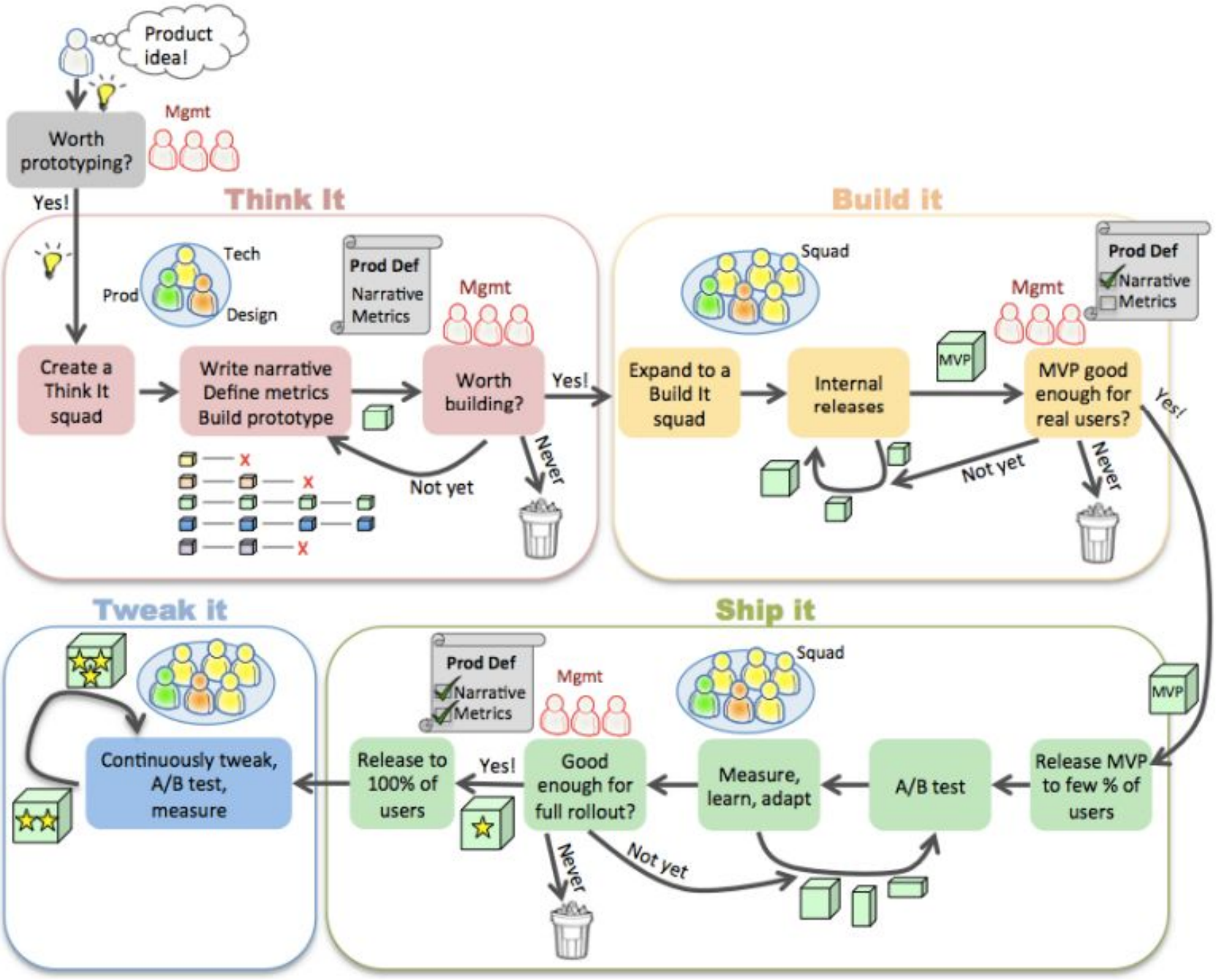


Squad

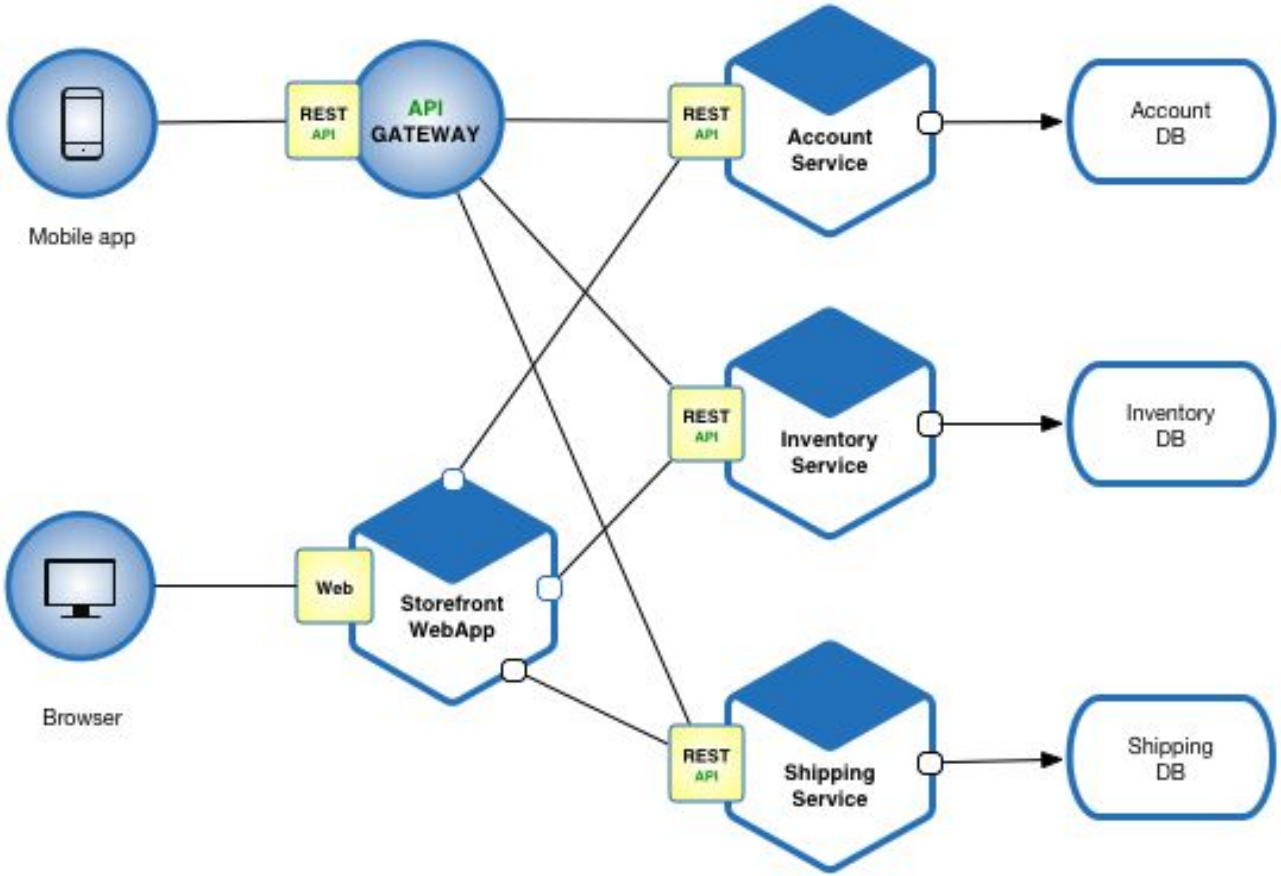


http://shipit.resultadosdigitais.com.br/blog/escalando-o-time-de-gestao-de-produtos-na-rd/


Tribe
Tribe
PO
PO
PO
PO
PO
PO
PO
PO
Chapter
Chapter
Chapter
Chapter
Guild
Squad
Squad
Squad
Squad
Squad
Squad
Squad
Squad


Product idea!
Worth prototyping?
Mgmt
Yes!
Think It
Tech
Prod
Design
Prod Def
Narrative
Metrics
Mgmt
Create a Think It squad
Write narrative
Define metrics
Build prototype
Worth building?
Yes!
Not yet
Never
Build it
Squad
Prod Def
Narrative
Metrics
Mgmt
MVP
Expand to a Build It squad
Internal releases
MVP good enough for real users?
Yes!
Not yet
Never
Tweak it
Continuously tweak, A/B test, measure
Ship it
Prod Def
Narrative
Metrics
Mgmt
Squad
MVP
Release to 100% of users
Yes!
Good enough for full rollout?
Measure, learn, adapt
A/B test
Release MVP to few % of users
Never
Not yet

Arquitetura MicroServiços



Regras Gerais GPP/MDS 2018.1

Quantidade de Sprints

Serão ao total X sprints

Horas trabalhadas

Deve-se planejar, no mínimo, 10 horas por semana por integrante. Durante TODA a disciplina, deve-se manter o monitoramento das horas trabalhadas por integrante

https://www.bitrix24.com.br/features/tasks.php

Registro das Sprints

O registro de acompanhamento das sprints devem ser disponibilizados na wiki do projeto e nas paredes da sala de aula

Artefatos

- Documento de Visão
- Documento de Arquitetura
- Folha de Estilo
- Protótipos
- Especificação Suplementar
- Acompanhamento das Sprints

Releases

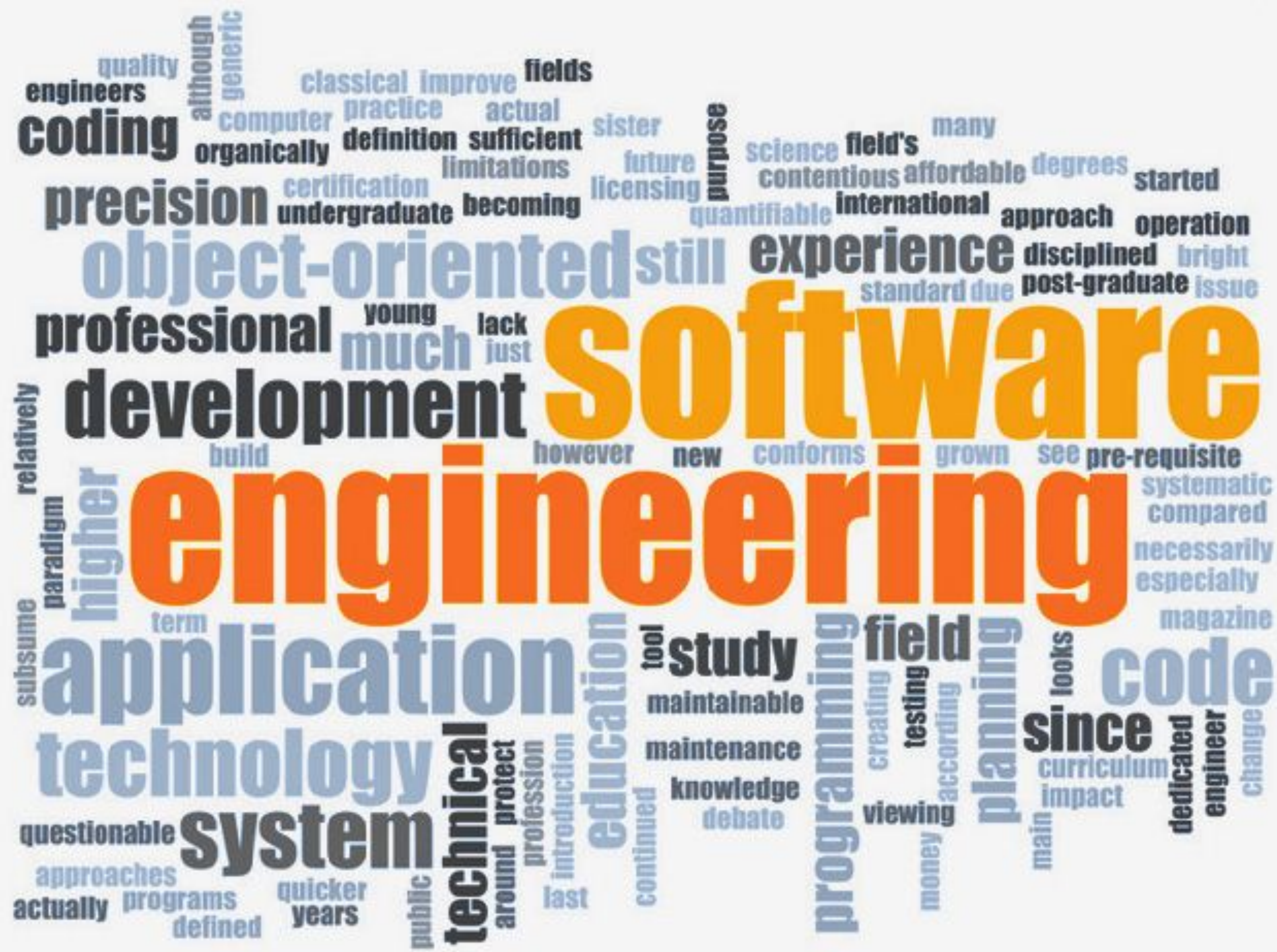
Serão realiadas 3 Releases

- R1:
- R2:
- R3:

Critérios de Avaliação

- Provas
- Projeto
- Contribuição na WIki da Disciplina

Maiores detalhes, olhar o plano de ensino no repositório da disciplina


quality
although
generic
engineers
classical
improve
fields
coding
computer
practice
actual
sister
purpose
many
organically
definition
sufficient
science
field's
limitations
future
contentious
affordable
degrees
started
precision
certification
undergraduate
becoming
licensing
quantifiable
international
approach
operation
object-oriented
still
experience
disciplined
bright
standard
due
post-graduate
issue
professional
young
much
lack
just
software
relatively
development
build
however
new
conforms
grown
see
pre-requisite
higher
paradigm
engineering
systematic
compared
necessarily
especially
subsume
term
application
education
tool
study
programming
field
planning
looks
magazine
code
maintainable
creating
testing
according
since
technology
technical
protect
profession
introduction
maintenance
curriculum
dedicated
engineer
change
knowledge
impact
questionable
system
around
continued
debate
viewing
money
main
approaches
public
last
actually
programs
quicker
years
defined


clients
flow
finishing
release
team
growth
quality
technology
fast
testing
cycle
flexible
system
developer
requirements
information
programmer
work
change
test
process
collaboration
business
organization
plan
code
innovation
strategy
software
Agile
project
review
integration
computer
programming
web
continuous
data
response
Development
leadership
management
implementation
brainstorming
lifecycle
improvement
methodology
analysis
evolution
deployment
product
analytics
engineering
backlog
coding
model
control
iterative
method
speed
progress
meeting
division
master
www.shutterstock.com · 638814157

Leituras sugeridas

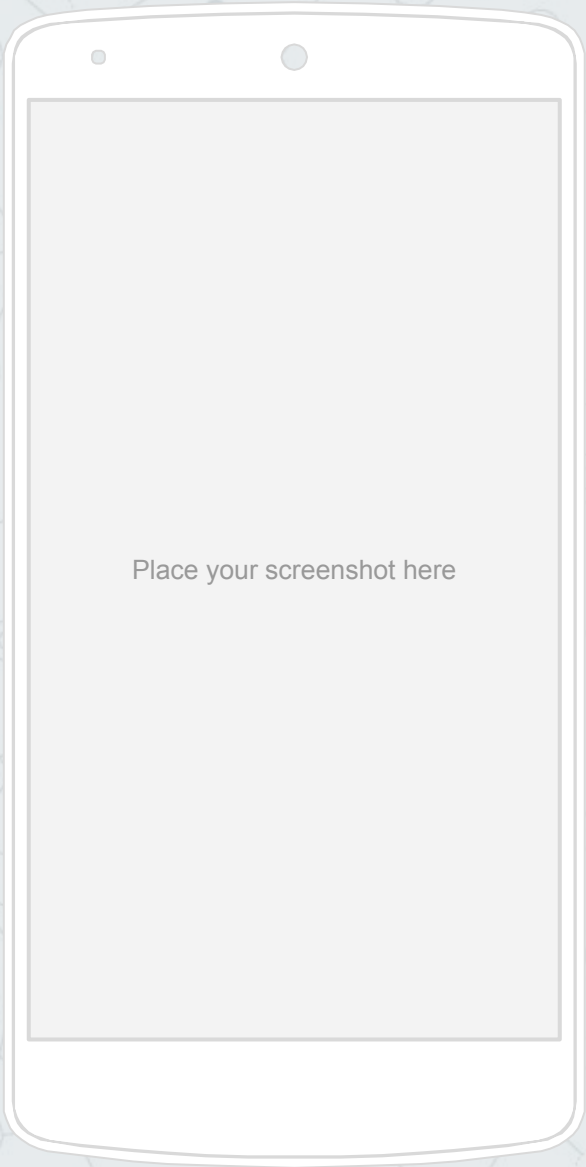
- https://medium.com/design-rd/por-que-todo-produto-deveria-investir-em-forma%C3%A7%C3%A3o-de-h%C3%A1bito-fbba2fb53c6b

Your audience will listen to you or read the content, but won’t do both.


Perguntas?

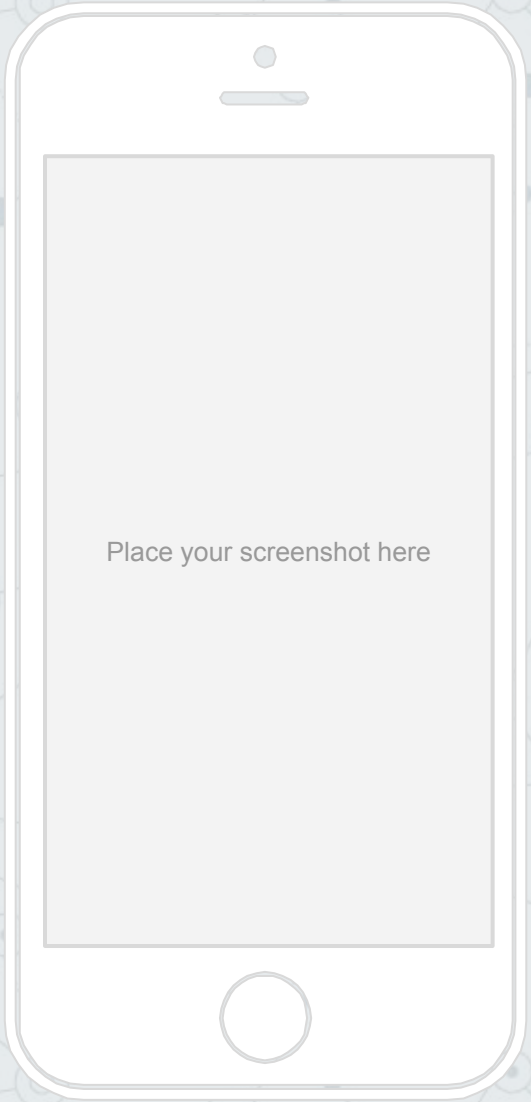
Android project

Show and explain your web, app or software projects using these gadget templates.



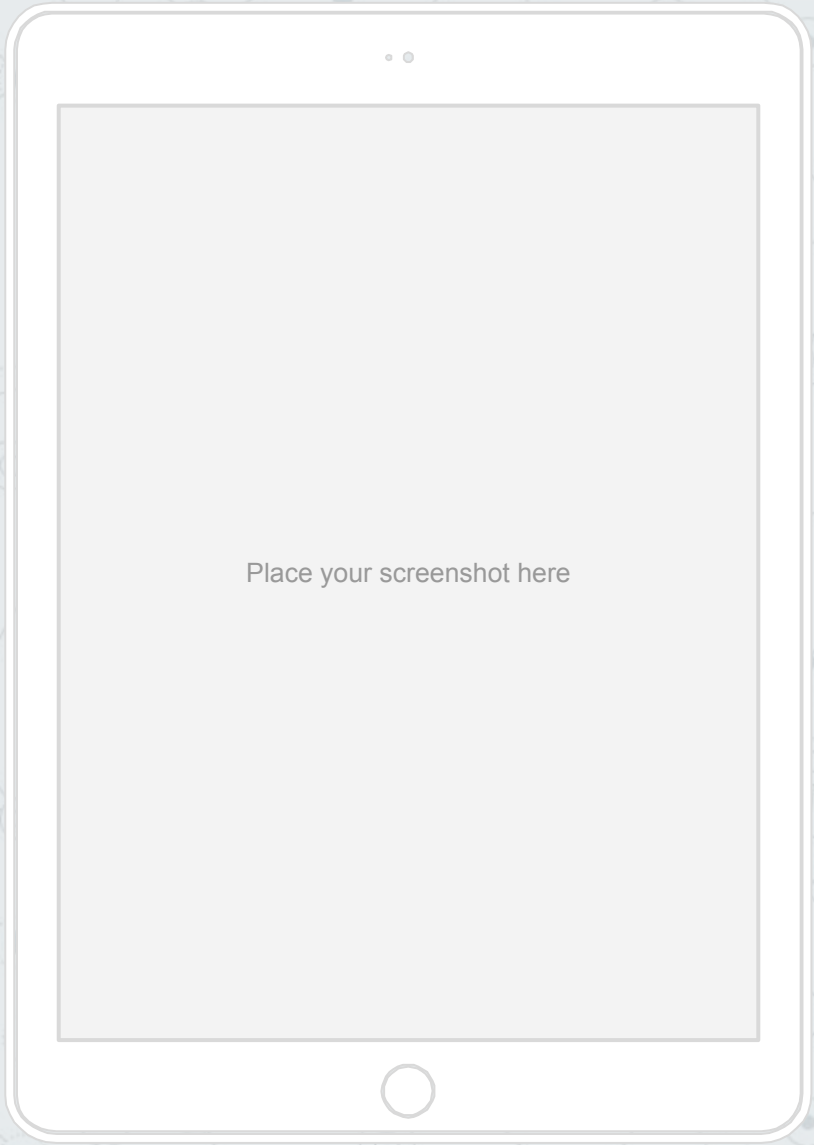
iPhone project

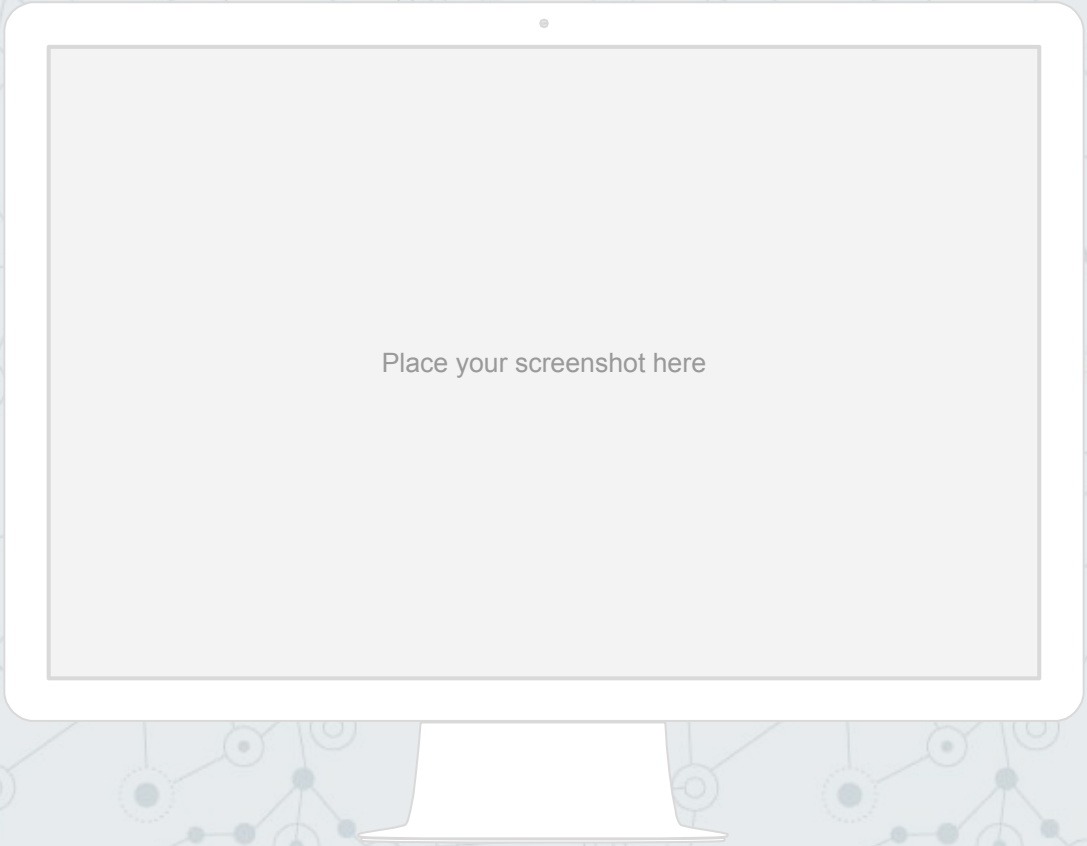
Show and explain your web, app or software projects using these gadget templates.



Tablet project

Show and explain your web, app or software projects using these gadget templates.





Desktop project

Show and explain your web, app or software projects using these gadget templates.

Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash & Death to the Stock Photo (license)

SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change line color, width and style.

Isn't that nice? :)

Examples: