

Engenharia de Software II

Introdução ES II

Prof. André Hora
DCC/UFMG
2019.1



SWEBOK[®] V3.0

*Guide to the Software
Engineering Body of Knowledge*

Editors

Pierre Bourque
Richard E. (Dick) Fairley



IEEE  computer society

SWEBOK

Table I.1. The 15 SWEBOK KAs
Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

Engenharia de Software I

Table I.1. The 15 SWEBOK KAs
Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

Engenharia de Software II

Table I.1. The 15 SWEBOK KAs	
Software Requirements	
Software Design	
Software Construction	
Software Testing	
Software Maintenance	✓
Software Configuration Management	✓
Software Engineering Management	
Software Engineering Process	
Software Engineering Models and Methods	
Software Quality	✓
Software Engineering Professional Practice	
Software Engineering Economics	
Computing Foundations	
Mathematical Foundations	
Engineering Foundations	

Engenharia de Software II

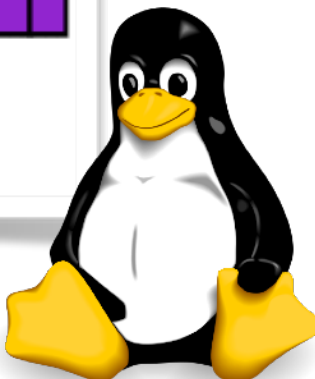
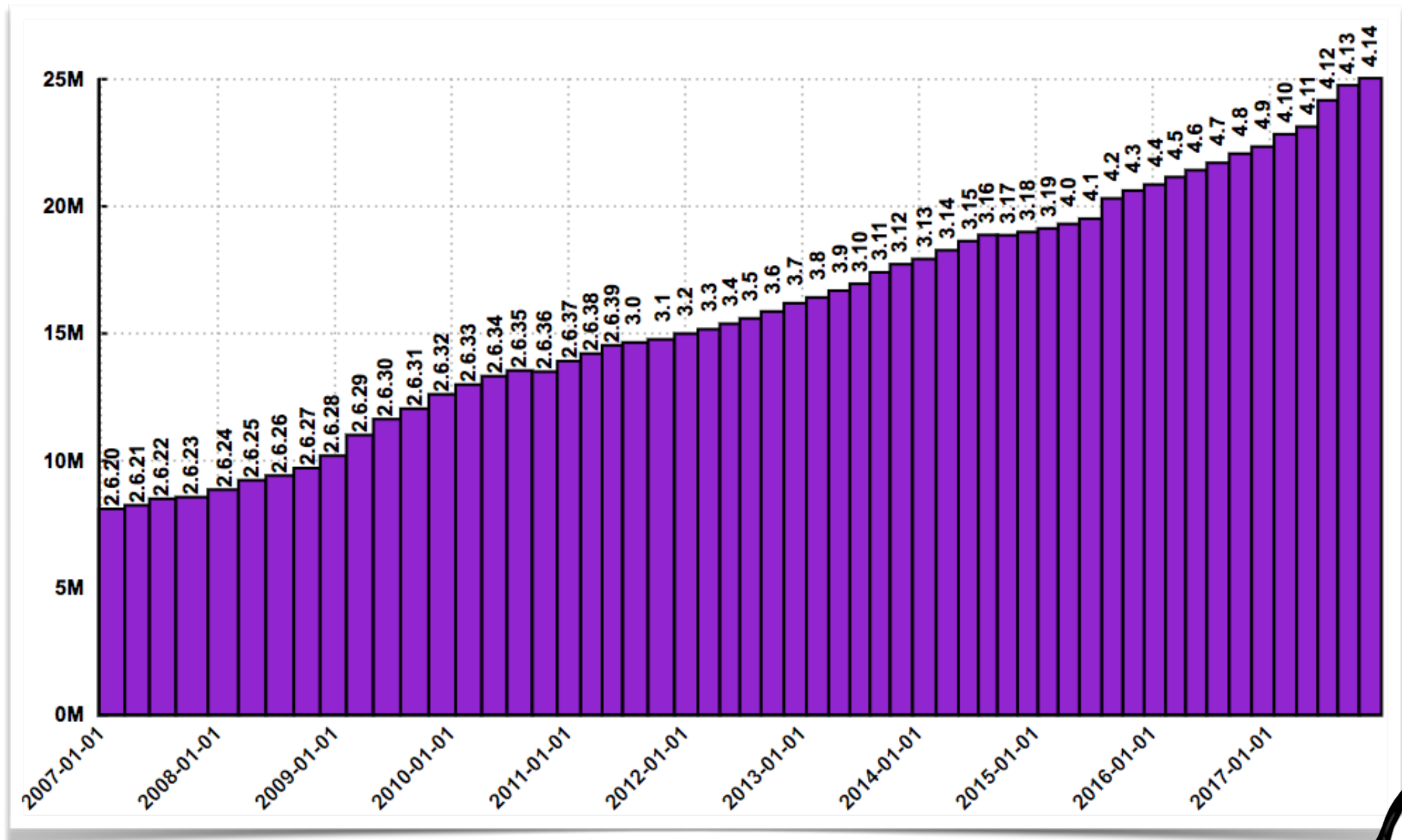
1. Manutenção de Software
2. Qualidade de Software
3. Gerência de Configuração

1 Manutenção de Software

Manutenção de Software

- Sistemas de software estão em constante evolução
- Novas funcionalidades, correção de bugs, refatoração de código
- Representa até **90%** dos custos de desenvolvimento

Linhas de Código do **Linux**



Leis de Lehman

 **Mudança contínua**

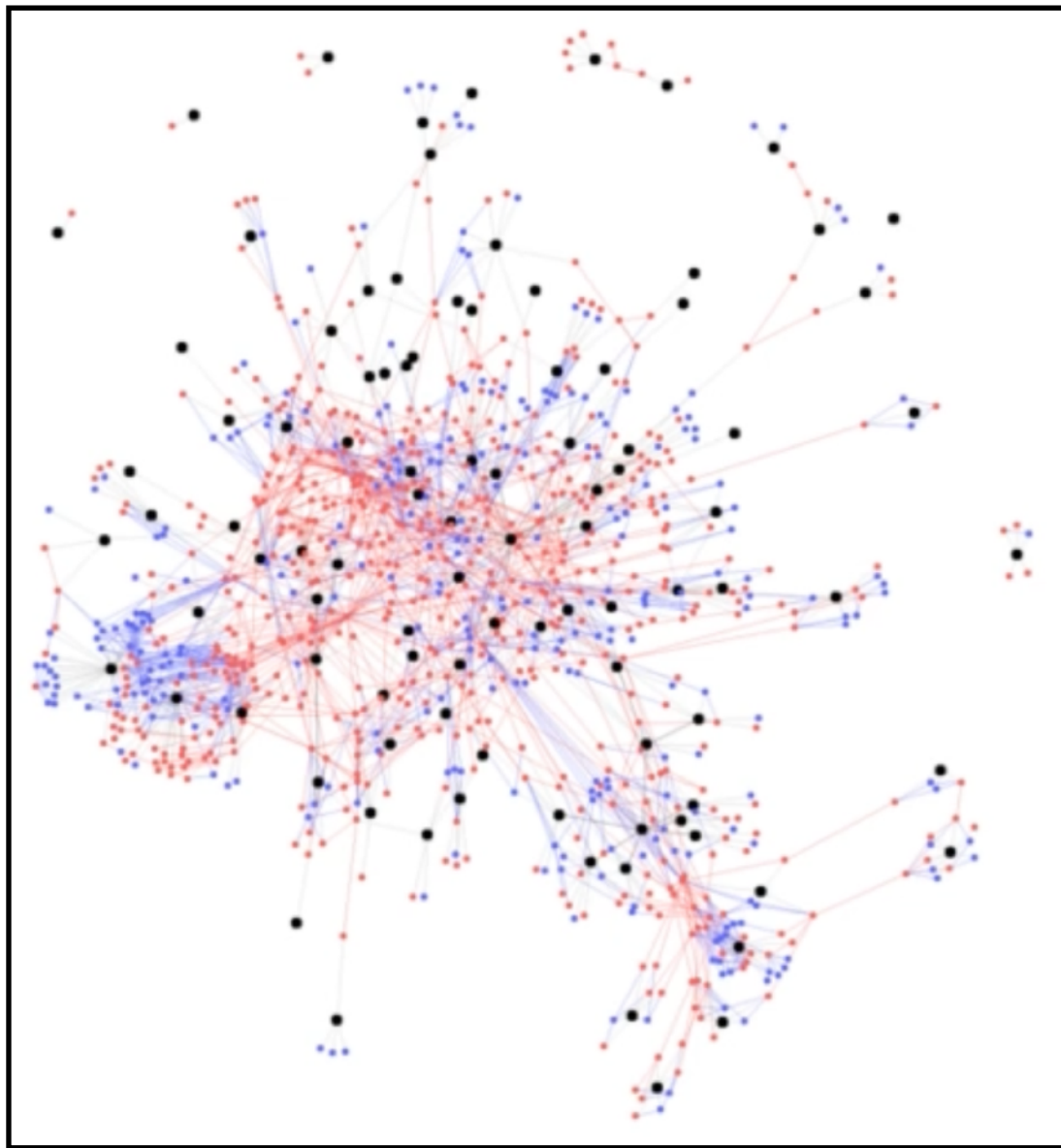
 **Complexidade crescente**

 **Crescimento contínuo**

 **Declínio da qualidade**

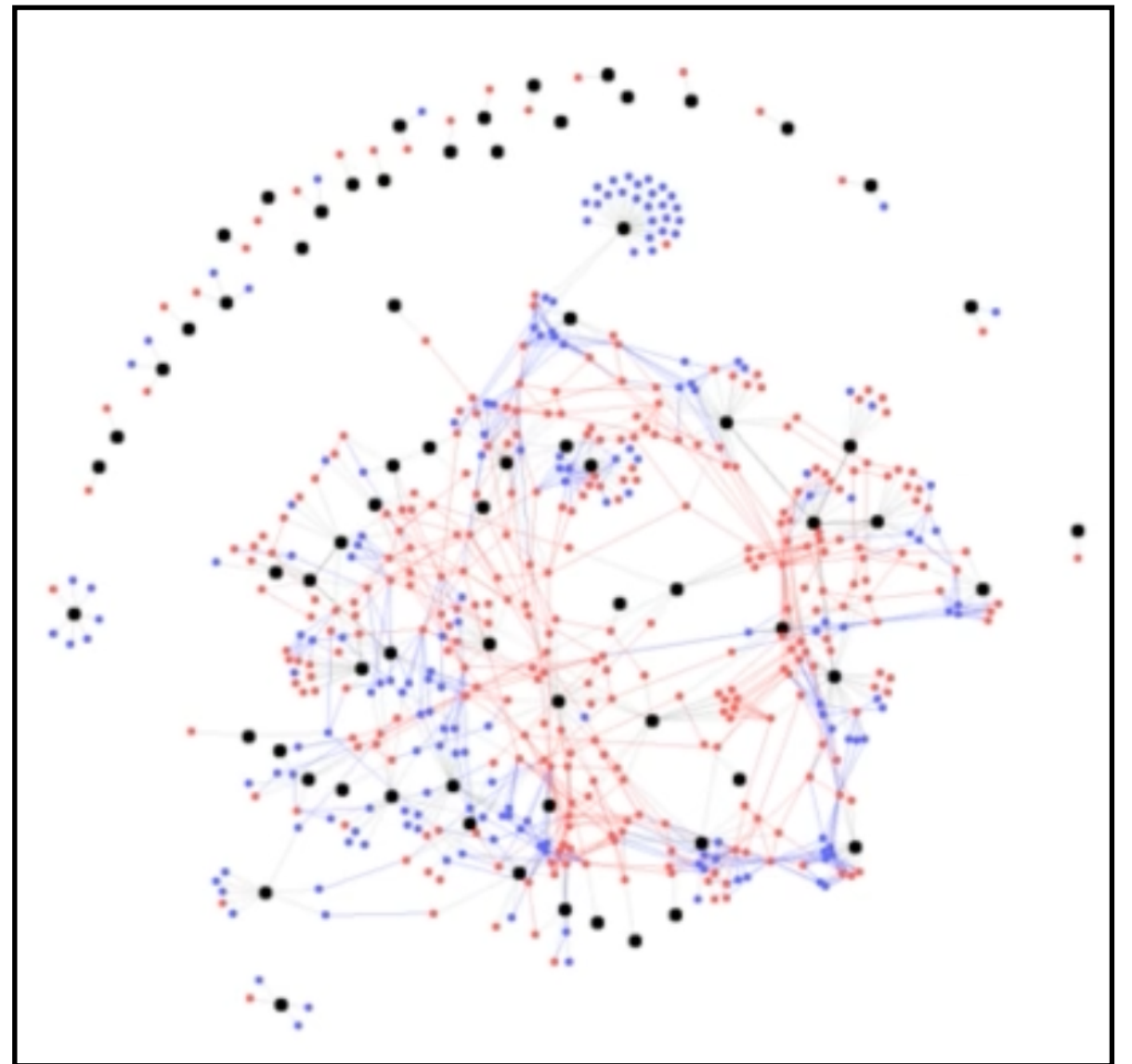
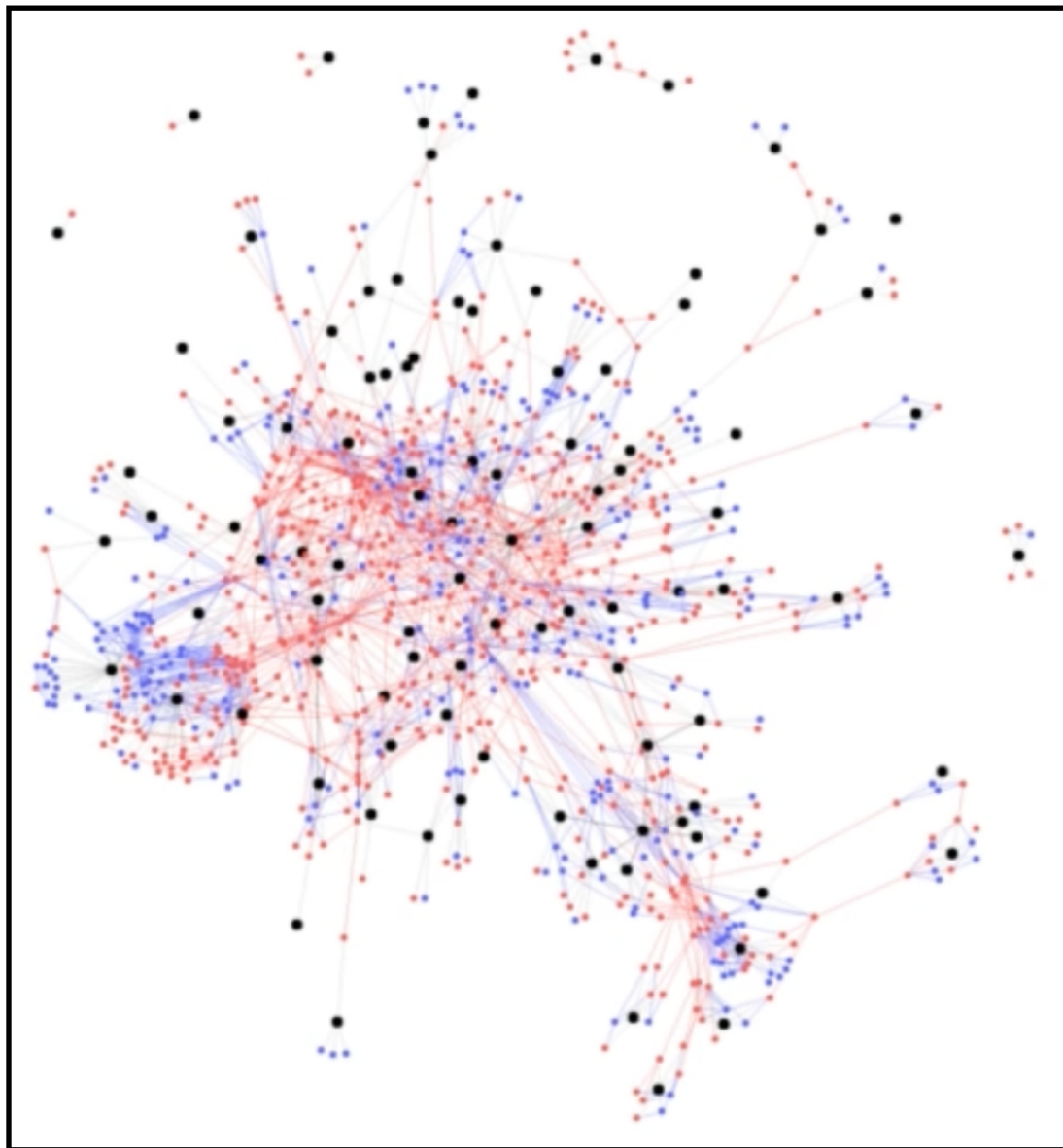
Exemplo

Arquitetura de Software

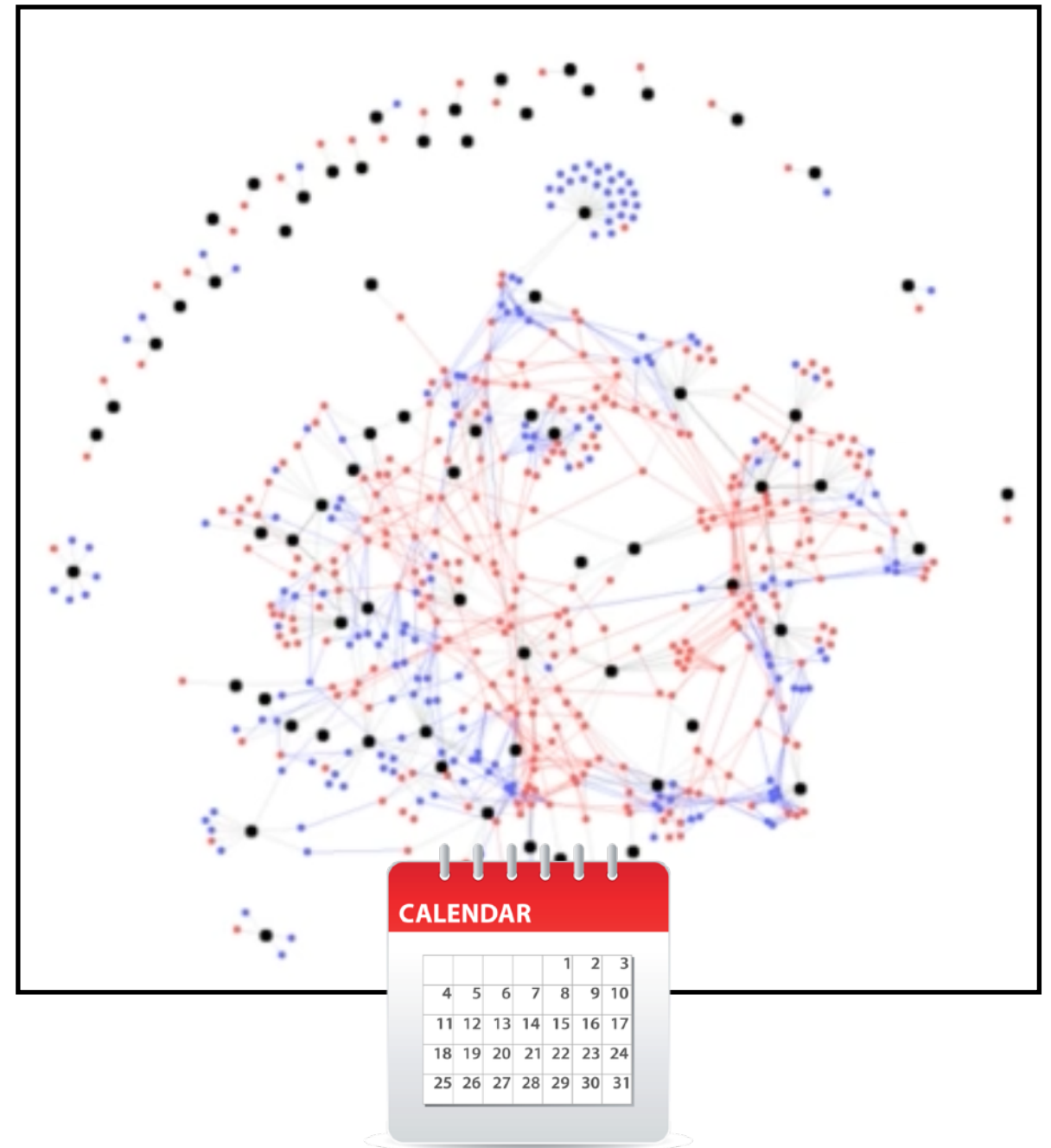
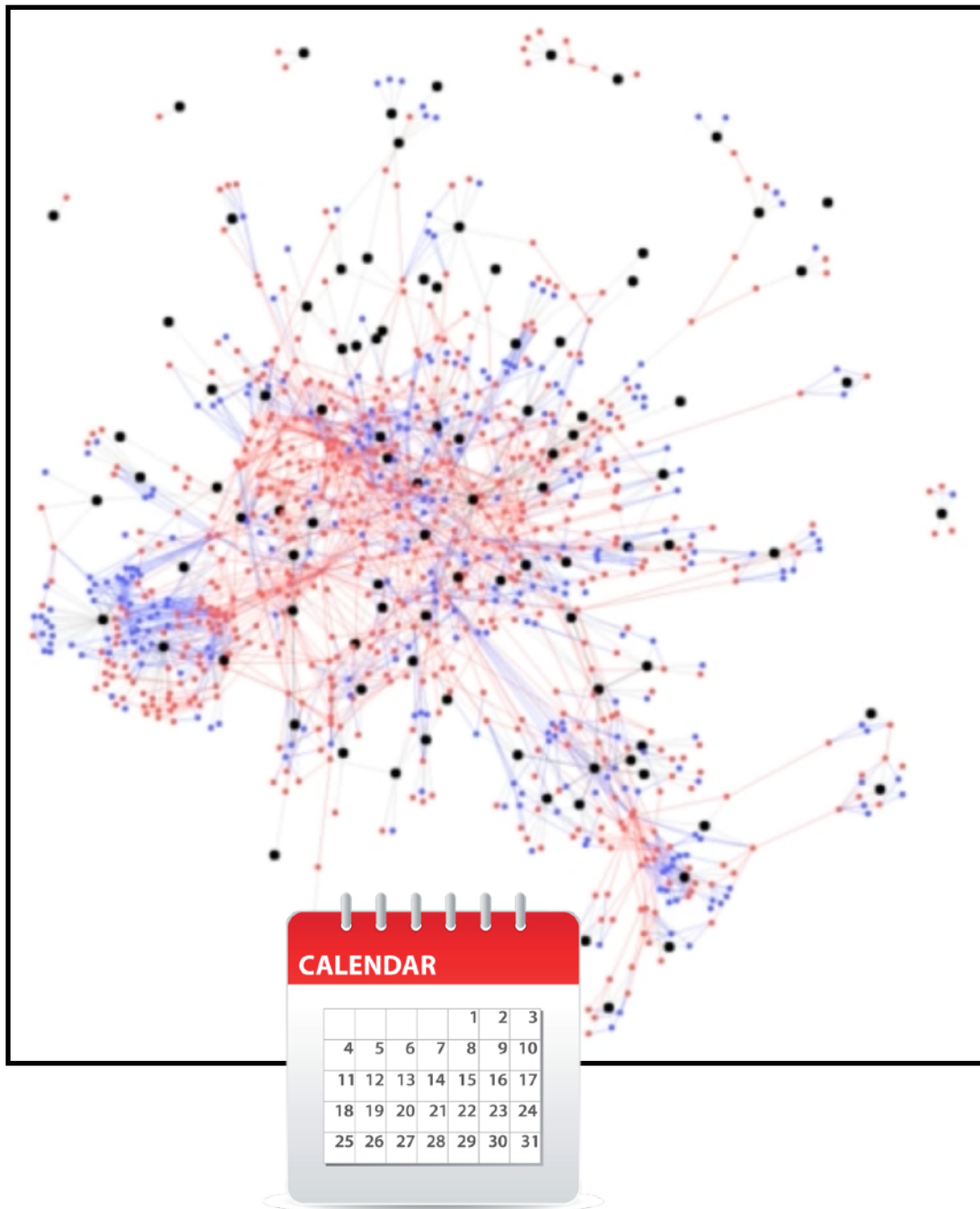


- Classe
- Método
- Atributo
- Relacionamento

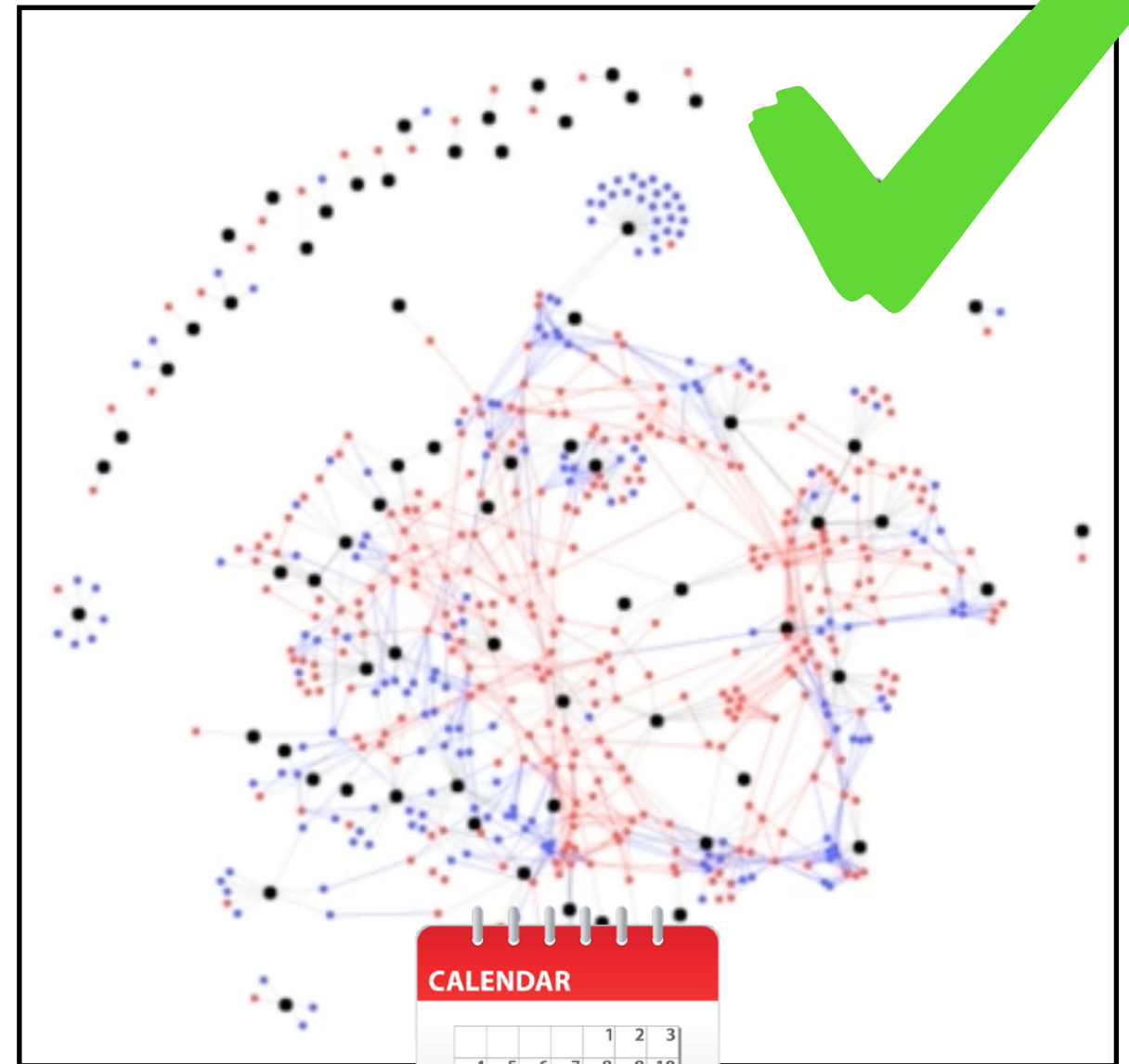
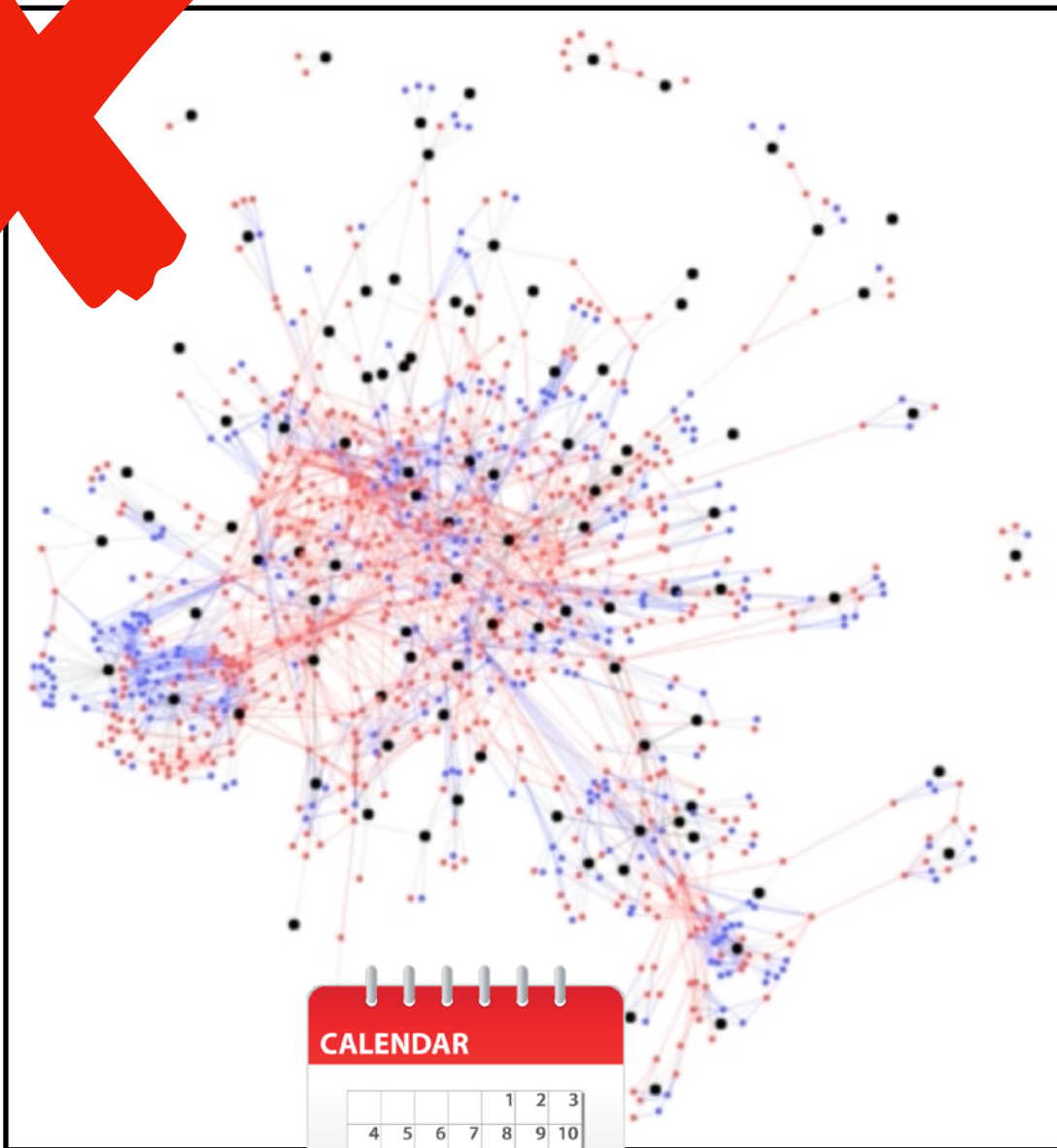
Arquitetura de Software



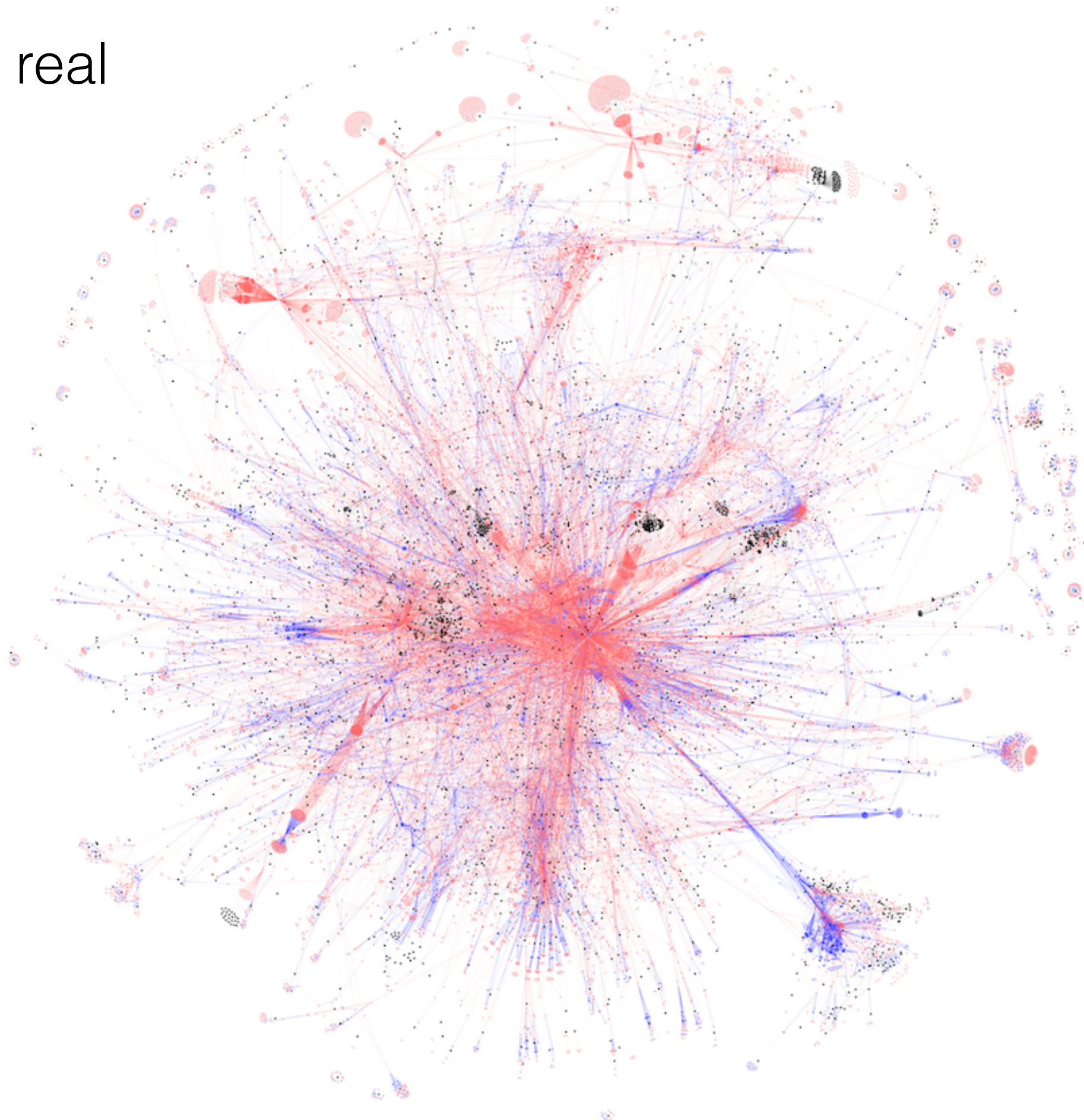
Mesmo comportamento, ≠ estruturas internas



Mesmo comportamento, \neq estruturas internas



Sistema real



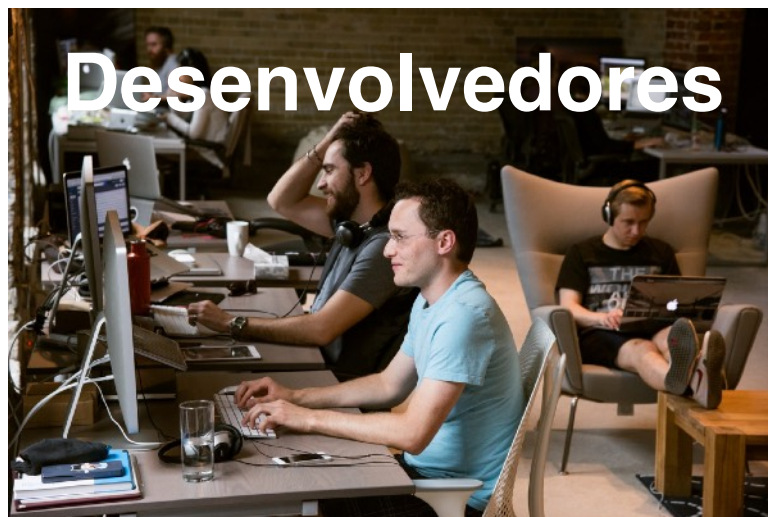


2

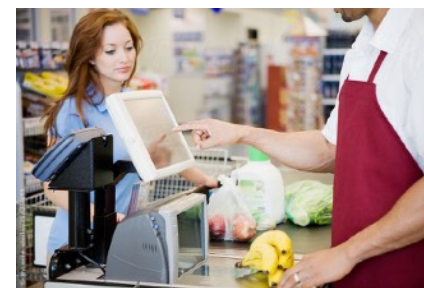
Qualidade de Software

Qualidade de Software

Conjunto de características a serem satisfeitas em um determinado grau, de modo que o software satisfaça às necessidades de seus **usuários**



Usuários finais



Conceito Subjetivo

```
1. public int compute(int n, int m, int o) {  
2.     for(int i = 0; i < n; i++) {  
3.         for(int j = 0; j < m; j++) {  
4.             for(int k = 0; k < o; k++) {  
5.                 this.x := ...  
6.             }  
7.         }  
8.     }  
9.     return (this.x < n) ? 1 : 2;  
10. }
```

Código legível?

Norma ISO 25010

- Atributos de qualidade da ISO 25010:
 - Funcionalidade
 - Usabilidade
 - Confiabilidade
 - Desempenho
 - Portabilidade
 - Manutenibilidade
 - Segurança
 - Compatibilidade

Métricas de Software

- Elemento chave para entender os **atributos de qualidade**
- Quantifica os atributos internos de um sistema
- Métricas de produto, projeto e processo

Compreensão de Programas em Java

- Estudos recentes apontam para dados mais críticos: **56% e 94%** do tempo é gasto com entendimento (dados de 2014)

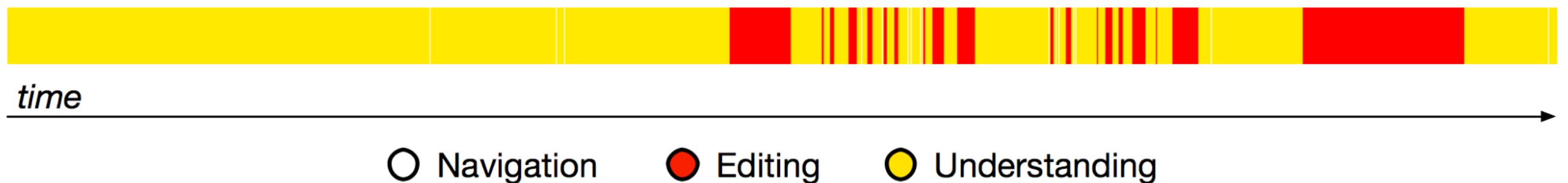


Fig. 2: Visualizing Java Development Activities.

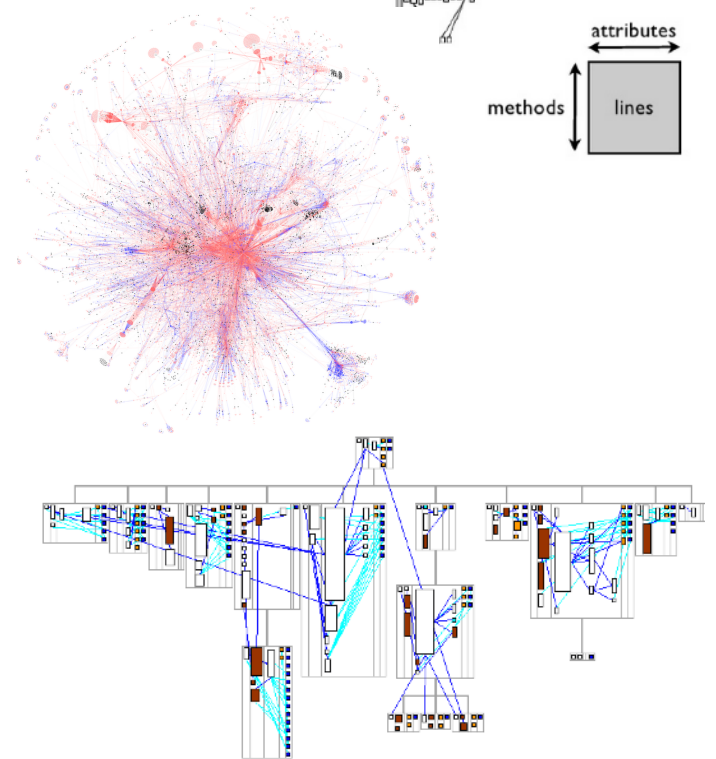
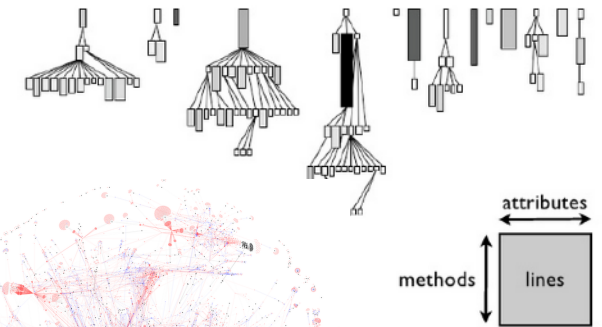


Bugzilla

```
1 (function() {  
2  
3   window.utils = window.utils || {};  
4  
5   /*  
6   Random Number Generator.  
7  
8   Pretty awesome explanation here:  
9   http://stackoverflow.com/a/1527820  
10  */  
11  utils.randomNumber = function(min, max) {  
12    return Math.floor(Math.random() * (max - min + 1)) + min;  
13  };
```



Mineração
de
Software





3

Gerência de Configuração


Evolução de Software

- Sistemas de software estão em constante mudança durante o seu desenvolvimento
 - **Novas funcionalidades** são adicionadas
 - **Defeitos** são corrigidos
 - **Código** é refatorado para melhorar qualidade

Mudanças → Versões

- Quando **mudanças** são realizadas em um sistema, uma nova versão é criada
- Maioria dos sistemas podem ser vistos como conjunto de versões
 - Devem ser mantidas e gerenciadas

 **torvalds** / **linux**

 Watch ▾

6,632


★ Star

69,774

 Fork

24,996

↔ Code

 Pull requests **254**

 Projects **0**

 Insights


Linux kernel source tree

 **812,453** commits

 **1** branch

 **593** releases

 ∞ **contributors**

 View license

Branch: **master** ▾

New pull request


Create new file

Upload files

Find file

Clone or download ▾

 **torvalds** / **linux**

 Watch ▾

6,632


★ Star


69,774

 Fork

24,996

↔ Code

 Pull requests **254**

 Projects **0**

 Insights


Linux kernel source tree

 **812,453** commits

 **1** branch

 **593** releases

 ∞ contributors

 View license

Branch: master ▾

New pull request

Create new file

Upload files

Find file

Clone or download ▾

Michael Ogawa. cc by-n



Gerência de Configuração (SCM)

- Atividade aplicada **ao longo do processo** de desenvolvimento para lidar com alterações
 - Mudanças ocorrem a qualquer momento
- Importante, pois é relativamente **fácil perder controle** das mudanças e adições ao sistema

Tipos de Controle de Versões

- **Centralizado:** repositório principal mantém todas as versões dos componentes desenvolvidos

- Exemplos: SVN, CVS



- **Distribuído:** múltiplas versões do repositório existem ao mesmo tempo

- Exemplo: Git

