



# DR-Tools

---

A tool quality suite to help the developers to maintain health and code evolution

**[drtools.site](https://drtools.site)**



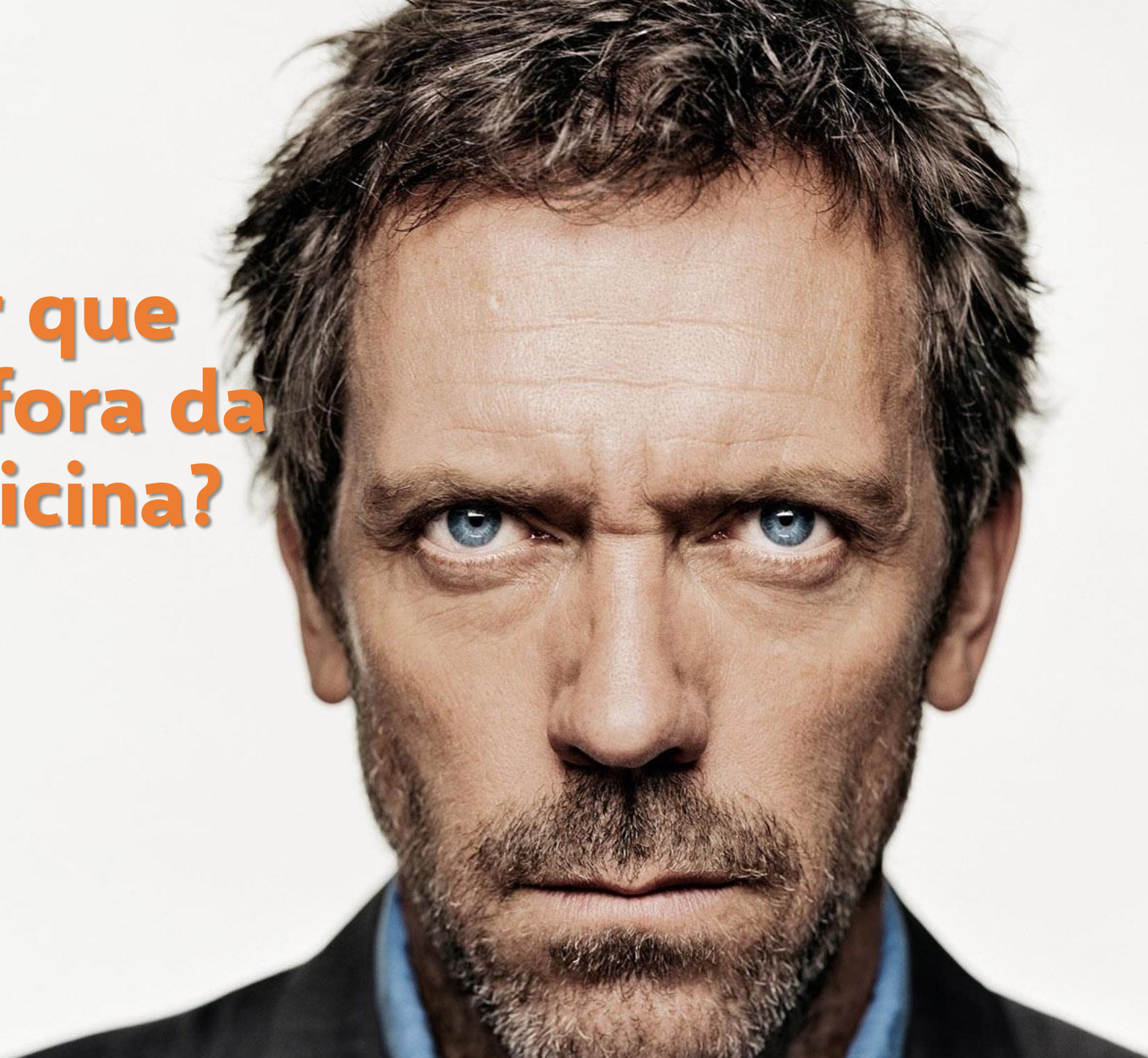
# Quem sou eu?

glacerda@wildtech.com.br  
@guilhermeslac

- ✓ Mestre e Doutorando em Ciência da Computação (UFRGS)
- ✓ Professor de Graduação e Pós-Graduação (Unisinos)
- ✓ Consultor associado da Wildtech
- ✓ Pioneiro em Metodologias Ágeis no Brasil
- ✓ Fundador do XP-RS/GUMA
- ✓ Membro da ScrumAlliance , IASA, SBC e ACM



**Por que  
metáfora da  
medicina?**





**DR-Tools**

---

32 Heurísticas de Análise

**[drtools.site](http://drtools.site)**

# Heurísticas - Summary

- 1. Contexto que fornece informações gerais sobre dimensões do Projeto**  
*Indicativo de filtrar melhor as informações usando a opção '--top X' para ajudar no entendimento (Classificação: SMALL, MEDIUM e LARGE)*
- 2. Considere o número médio de classes por namespace**  
*Indicativo de que as classes não estão distribuídas uniformemente*
- 3. Avalie o número médio de SLOC por classes**  
*Indicativo de classes muito grandes*
- 4. Observe a distribuição média de métodos por classes**  
*Indicativo de muitos comportamentos por classe*
- 5. Considere a complexidade média por classes**  
*Indicativo de como está a complexidade das classes em geral*



# Heurísticas - Namespaces

## 6. Observe a distribuição de classes por namespace

*Se um namespace tem muitas classes (NOC alto), pode ser um indicativo de 'promiscuous package'*

## 7. Avalie a distribuição de tipos abstratos (classes abstratas, interfaces) por namespaces

*Indicativos para extensão e reuso*

## 8. Avalie a relação das métricas NOC e NAC do namespace

*Uma diferença muito grande entre eles pode indicar uma má distribuição entre tipos abstratos e tipos concretos*





# Heurísticas – Types (1)

## 9. Avalie as métricas além do SLOC

*WMC, DEPS (DEP e I-DEP) e NOM/NPM são bons indicativos de como está a classe*

## 10. Classe com NOA alto, mas baixo WMC e NOM alto

*Pode ser um indicativo de POJO (Plain Old Java Object)*

## 11. SLOC alto, mas sem muitos métodos (NOM/NPM baixo)

*Pode ser um indicativo de 'long methods'*

## 12. SLOC e WMC alto, mas sem muitos métodos (NOM/NPM baixo)

*Pode ser um indicativo de 'complex class'*

## 13. NOM/NPM alto pode ser indicativo de classe com muitas responsabilidades

*Indica baixa coesão e possivelmente 'god class'*



# Heurísticas – Types (2)

- 14.** **NOM/NPM alto e NOA baixo pode ser indicativo de classe com muitas responsabilidades**  
*Pode ser um indicativo de uma classe 'controller'*
- 15.** **NOM alto e NPM baixo pode indicar que o comportamento foi dividido**  
*Indicativo de métodos private/protected/default*
- 16.** **NOA alto pode ser indicativo de classe com muitas responsabilidades**  
*Pode ser um indicativo de baixa coesão, dificultando a manutenção*
- 17.** **DEP alto e I-DEP baixo pode indicar uma classe com muitas dependências externas**  
*Dependências de APIs externas (frameworks, libs)*
- 18.** **I-DEP alto (e por consequência DEP alto), pode indicar uma classe com muitas dependências de classe do projeto**  
*Incidência de alto acoplamento*



# Heurísticas – Methods

- 19.** PARAM alto pode ser indicativo de método com baixa coesão  
*Possivelmente é um 'long method'*
- 20.** CYCLO alto e MLOC baixo pode ser um 'complex method'  
*Indicativo de problema de complexidade, legibilidade e entendimento*
- 21.** NBD alto pode ser um 'complex/long method'  
*Indicativo de problema de complexidade, legibilidade e entendimento*
- 22.** CALLS alto pode indicar alto acoplamento  
*Indicativo de problema de várias dependências*
- 23.** MLOC alto, CYCLO alto, CALLS alto e NBD alto é forte indicativo de mais de um problema  
*Pode ser um indicativo de um 'complex/long method'*



# Heurísticas – Coupling (1)

- 24. Evite dependência cíclicas**  
*Tornam as mudanças complexas e gera a 'síndrome da compilação total'*
- 25. CA alto pode indicar que o namespace é estável**  
*Se ele mudar, vai fazer com que quem dependa dele seja alterado*
- 26. CE alto pode indicar que o namespace é instável**  
*A incidência de mudança em outros namespaces que ele depende vai fazer com que ele mude*
- 27. I indica como está a instabilidade do namespace**  
 *$I=0$ , namespace estável ao máximo;  $I=1$ , namespace instável ao máximo*



# Heurísticas – Coupling (2)

- 28.** Se  $I=0$ , indica que  $CA > 0$  e  $CE=0$ , indica uma estabilidade total  
*Ele é responsável e independente. Os dependentes tornam difícil alterá-lo e não tem dependência de outros que pode forçar a mudança*
- 29.** A indica como está o grau de abstração do namespace  
*A=0, namespace não tem tipos abstratos; A=1, namespace somente possui tipos abstratos*

# Heurísticas – Coupling (3)

- 30.** Considere namespaces que estão nas zonas de exclusão

*Zone of Pain (namespaces com I e A próximos a 0) e Zone of Uselessness (namespaces com I e A próximos a 1)*

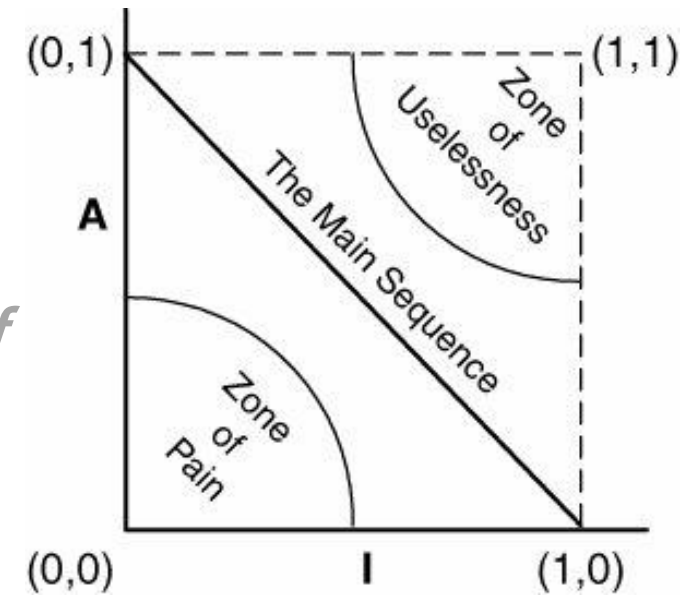
- 31.** Namespace situado próximo a sequência principal indica que não é abstrato nem instável demais

*Valor de D (entre 0 e 1) que vai indicar a posição na sequência principal*

- 32.** D indica o quão longe um namespace está da sequência principal

*D próximo a 0 indica proximidade da sequência principal; D próximo a 1, indica distância da sequência principal*

*Estes valores (mais próximo a 1) podem indicar quando um namespace está passível de manutenção e menos sensível a mudanças*





# DR-Tools

A tool quality suite to help the developers to maintain health and code evolution

**[drtools.site](https://drtools.site)**