



# DR-Tools

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

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



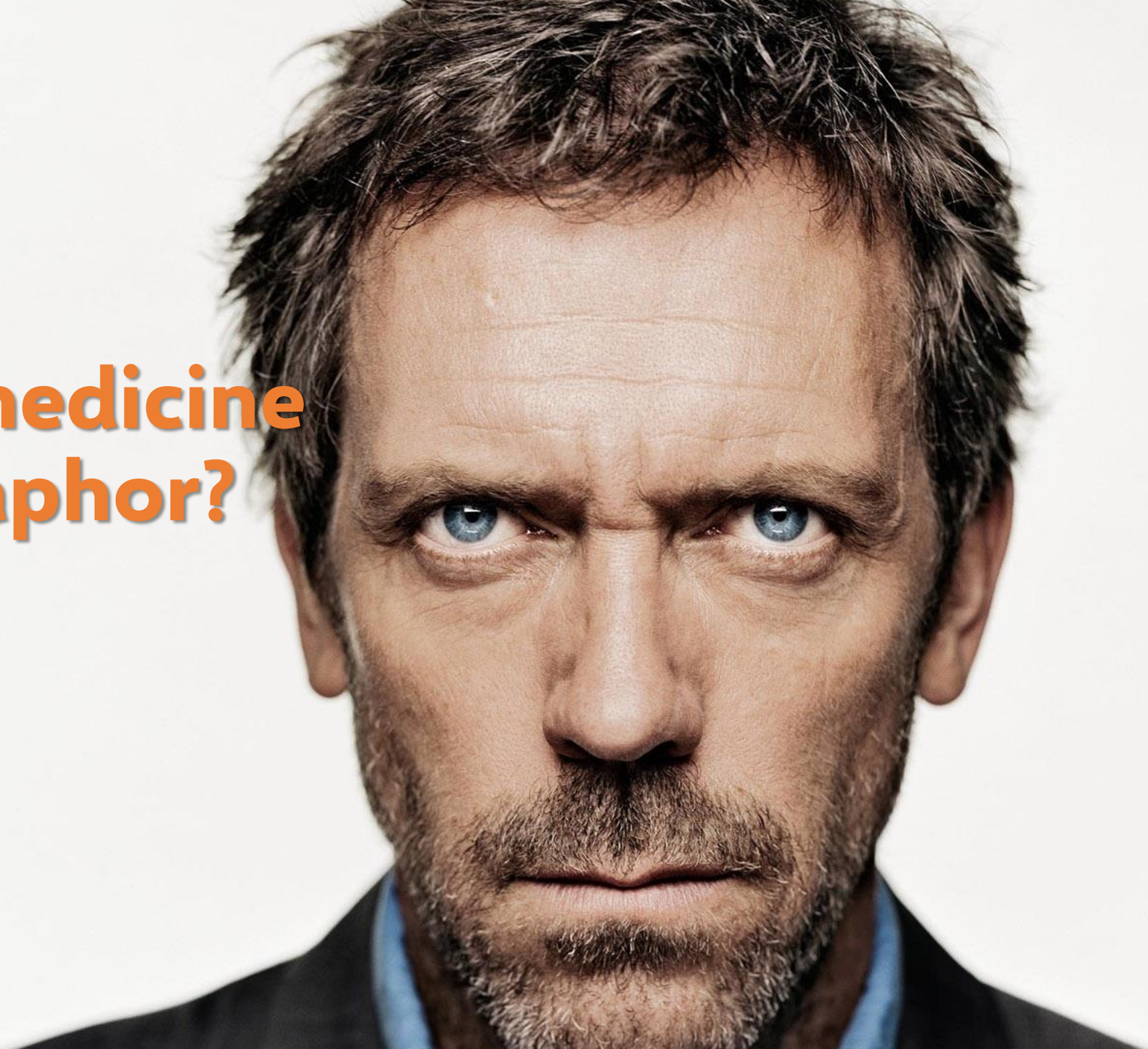
# Who am I?

glacerda@wildtech.com.br  
@guilhermeslac




- ✓ Computer Science MSc and PhD Student (UFRGS)
- ✓ Graduate and Undergraduate Lecturer (Unisinos)
- ✓ Associate Consultant (Wildtech)
- ✓ Agile Methods Pioneer in Brazil
- ✓ XP-RS/GUMA Community Co-founder
- ✓ ScrumAlliance , IASA, SBC, and ACM Member



**Why medicine  
metaphor?**



# DR-Tools Suite

- ✓ **metric**
- ✓ **metric visualization**
- ✓ **smell-detection** 
- ✓ **refactoring-recommender (plugin IDE)** 
- ✓ **smell-refactoring dashboard** 



Metric Definition and Thresholds

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

# Summary



## Small Project (SMALL)

*small project with  $< 50$  KLOC or  $200 < \text{classes}$*



## Medium Project (MEDIUM)

*medium project with  $(50 \text{ KLOC} \leq \text{project} \leq 250 \text{ KLOC})$  or  $(200 \leq \text{classes} \leq 1000)$*



## Large Project (LARGE)

*large project with  $> 250$  KLOC or  $> 1000$  classes*



# Namespaces



## Number of Types/Classes (NOC)

*Good:  $\leq 11$ ; Regular: between 11 and 28; Bad:  $> 28$*



## Number of Abstract Types/Classes (NAC)

*Without references*



# Types (1)

✓ **Type/Class Line of Code (SLOC)**

*Bad: > 500*

✓ **Number of Functions/Methods (NOM)**

*Good:  $\leq 6$ ; Regular: between 6 and 14; Bad: > 14*

✓ **Number of Public Methods (NPM)**

*Good:  $\leq 10$ ; Regular: between 11 and 40; Bad: > 40*

✓ **Weighted Methods per Class (WMC)**

*Good:  $\leq 20$ ; Regular: between 20 and 100; Bad: > 100*

✓ **Number of external (external APIs, frameworks, libs) types/classes dependencies (DEP)**

*Bad: > 20*





# Types (2)

- ✓ **Number of other internal types/classes dependencies (I-DEP)**  
*Bad: > 15*
- ✓ **Number of other types that depend on a given type (FAN-IN)**  
*Bad: > 10*
- ✓ **Number of other types referenced by a type (FAN-OUT)**  
*Bad: > 15*
- ✓ **Number of Attributes/Fields (NOA)**  
*Good:  $\leq 3$ ; Regular: between 3 and 8; Bad: > 8*
- ✓ **Lack of Cohesion in Methods (LCOM3)**  
*Good: = 0; Regular: between 0 and 1; Bad: > 1*



# Methods

✓ **Method Lines of Code (MLOC)**

*Good:  $\leq 10$ ; Regular: between 10 and 30; Bad:  $> 30$*

✓ **Cyclomatic Complexity (CYCLO)**

*Good:  $\leq 2$ ; Regular: between 2 and 4; Bad:  $> 4$*

✓ **Number of Invocations (CALLS)**

*Bad:  $> 5$*

✓ **Nested Block Depth (NBD)**

*Good:  $\leq 1$ ; Regular: between 1 and 3; Bad:  $> 3$*

✓ **Number of Parameters (PARAM)**

*Good:  $\leq 2$ ; Regular: between 2 and 4; Bad:  $> 4$*



# Namespace Coupling



## Afferent Coupling (CA)

*Good:  $\leq 7$ ; Regular: between 7 and 39; Bad:  $> 39$*



## Efferent Coupling (CE)

*Good:  $\leq 6$ ; Regular: between 6 and 16; Bad:  $> 16$*



## Package Instability (I)

*range between 0=Maximally stability and 1=Maximally instability*



## Abstractness Degree (A)

*range between 0=Minimally abstractness and 1=Maximally abstractness*



## Normalized Distance (D)

*range between 0=exactly located in the main sequence and 1=far from the main sequence*



# Type Coupling

- ✓ **Number of external (external APIs, frameworks, libs) types/classes dependencies (DEP)**  
*Bad: > 20*
- ✓ **Number of other internal types/classes dependencies (I-DEP)**  
*Bad: > 15*
- ✓ **Number of other types that depend on a given type (FAN-IN)**  
*Bad: > 10*
- ✓ **Number of other types referenced by a type (FAN-OUT)**  
*Bad: > 15*

# References

- 
- Danijel Radjenović, Marjan Heričko, Richard Torkar, Aleš Živkovič, Software fault prediction metrics: A systematic literature review, Information and Software Technology, Volume 55, Issue 8, 2013
- 
- Mariza A.S. Bigonha, Kecia Ferreira, Priscila Souza, Bruno Sousa, Marcela Januário, Daniele Lima, The usefulness of software metric thresholds for detection of bad smells and fault prediction, Information and Software Technology, Volume 115, 2019
- 
- T. Filo, M. Bigonha, and K. Ferreira, A catalogue of thresholds for object-oriented software metrics, in Advances and Trends in Software Engineering, 2015 The First International Conference on. IARIA, 2015
- 
- Kecia A.M. Ferreira, Mariza A.S. Bigonha, Roberto S. Bigonha, Luiz F.O. Mendes, Heitor C. Almeida, Identifying thresholds for object-oriented software metrics, Journal of Systems and Software, Volume 85, Issue 2, 2012
- 
- Rangasamy, R.Selvarani & Nair, T.R. & Ramachandran, Muthu & Prasad, Kamakshi. Software Metrics Evaluation Based on Entropy. Handbook of Research on Software Engineering and Productivity Technologies: Implications of Globalization, 2010
- 
- Robert C. Martin and Micah Martin. 2006. Agile Principles, Patterns, and Practices in C# (Robert C. Martin). Prentice Hall PTR, Upper Saddle River, NJ, USA
- 
- P. Oliveira, M. T. Valente and F. P. Lima, Extracting relative thresholds for source code metrics, 2014 Software Evolution Week - IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE), Antwerp, 2014
- 
- G. Vale, A benchmark-based method to derive metric thresholds, Master's Dissertation, UFMG, 2015
- 
- di Biase, M., Rastogi, A., Bruntink, M., & van Deursen, A. The Delta Maintainability Model: Measuring Maintainability of Fine-Grained Code Changes. In TechDebt 2019 - International Conference on Technical Debt, 2019
- 
- E. Lima, A. Resende, T. Lethbridge, The Uncomfortable Discrepancies of Software Metric Thresholds and Reference Values in Literature, ICSEA 2016 : The Eleventh International Conference on Software Engineering Advances, 2016
- 
- Checkstyle - Size Violations. [https://checkstyle.sourceforge.io/config\\_sizes.html](https://checkstyle.sourceforge.io/config_sizes.html), 2019



# DR-Tools

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

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