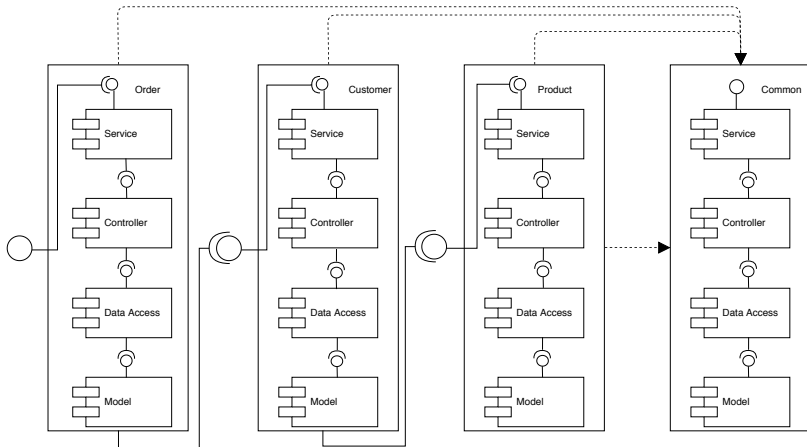# Maintenance Metric

Daniel San Martín
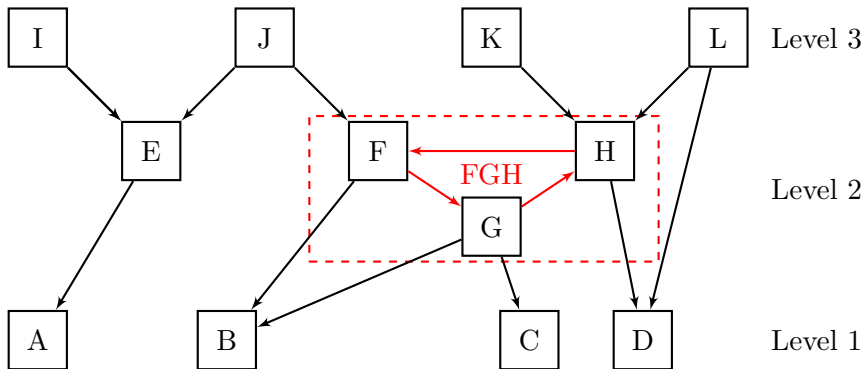
Advanse Group
Department of Computing, UFSCar
São Carlos, SP, Brazil

June 6, 2019

A good vertical design

Dependency graph with a cycle group

Efferent Coupling: The number of classes on which a given class depends.

Afferent Coupling: How many classes depend on a given class.

First approach to define a metric

$$c_i = \frac{size(i) * (1 - \frac{inf(i)}{numberOfComponentsInHigherLevels(i)})}{n} \qquad (1)$$

Where:

- n is the total number of components;
- $size(i)$ is the number of components in the logical node;
- $inf(i)$ is the number of components influenced by c_i.

First approach to define a metric

Example for node A:

Rationales
○○○●○○○○○

Exercises
○○

Resources
○

First approach to define a metric

Advan

Example for node A:

$$c_A = \frac{1 * (1 - \frac{3}{8})}{12} = 0.052 \tag{2}$$

$$ML_1 = 100 * \sum_{i=1}^{k} c_i \tag{3}$$

- k is the total number of logical nodes, which is smaller than n if there are cyclic component dependencies.

Rationales
○○○○○●○○○○

Exercises
○○

Resources
○

Penalty

Advan

Cyclic dependencies have a negative influence on maintainability, especially if the cycle group contains a larger number of nodes.

$$penalty(i) = \begin{cases} \frac{5}{size(i)}, & \text{if } size(i) > 5 \\ 1, & \text{otherwise} \end{cases} \quad (4)$$

$$ML_2 = 100 * \sum_{i=1}^{k} c_i * penalty(i) \quad (5)$$

Some tunnings...

- It did not work very well for small modules with less than 100 components;

$$ML3 = \begin{cases} (100 - n) + \frac{n}{100} * ML_2, & \text{if } n < 100 \\ ML_2, & \text{otherwise} \end{cases} \quad (6)$$

- In some projects, developers said the metric does not fitted well because to them it was difficult to maintain the system although the value of metric showed the opposite.

Rationales
○○○○○○○●○○

Exercises
○○

Resources
○

*Advan*SE

Advanced research group on Software Engineering

Some tunnings...

The **cyclicity of a package cycle group** is the square of the number of packages in the group. A cycle group of 5 elements has a cyclicity of 25.

The **cyclicity of a whole system** is just the sum of the cyclicity of all cycle groups in the system.

The relative cyclicity of a system is defined as follows:

$$relativeCiclicity = 100 * \frac{\sqrt{sumOfCyclicity}}{n} \qquad (7)$$

Where n is the total number of packages.

As an example assume a system with 100 packages. If all these
packages are in a single cycle group the relative cyclicity can be
computed as:

$$100 * \frac{\sqrt{100^2}}{100} = 1 \tag{8}$$

If we have 50 cycle groups of 2 packages we get:

$$100 * \frac{\sqrt{50 * 2^2}}{100} \approx 14,1\% \tag{9}$$

$$ML_{alt} = 100 * (1 - \frac{\sqrt{sumOfPackageCyclicity}}{n_p}) \tag{10}$$

Where n_p is the total number of packages.

The final formula

$$ML_4 = min(ML_3, ML_{alt}) \tag{11}$$

We simply argue that for good maintainability both the component structure and the package/namespace structure must well designed. If one or both suffer from bad design or structural erosion, maintainability will decrease too.
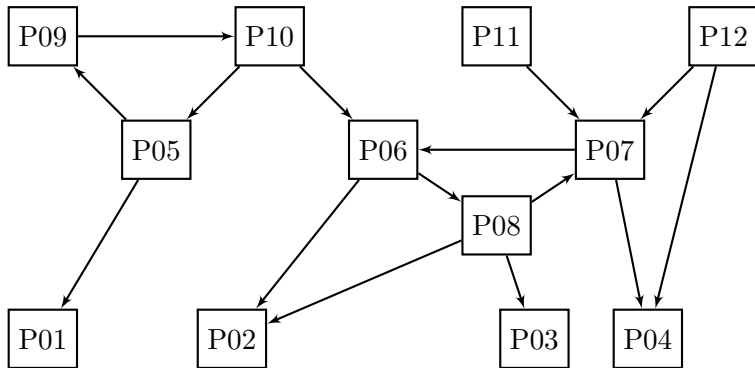
Exercise 1

Compute ML_3 for the following component graph.
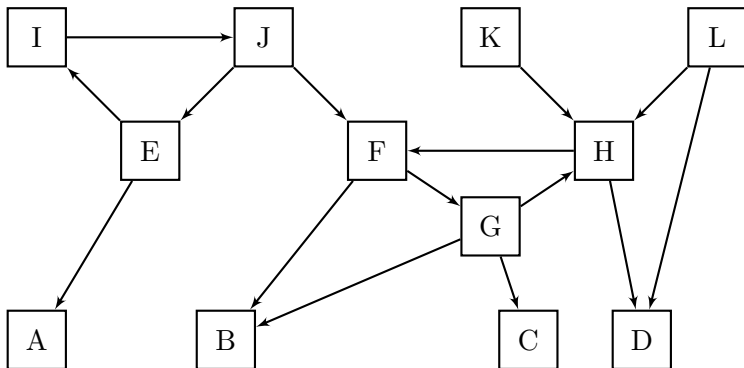
Exercise 2

Compute ML_{alt} for the following package graph. Which is the maintenance level metric if we compare the metrics of both exercises?

Slides & Exercise Solutions

https://github.com/dsanmartins/AulaProfValter

SonarGraph Blog

http://blog.hello2morrow.com/