Le programme *egon* a quelques problèmes comme méthode pour repérer les classes divines. Si le seuil est trop grand, *egon* donnera comme sortie beaucoup des classes, étant divines ou non.

Dans un cas où il n’y a pas de classe divine ou que les « seuil » est trop haut, certaines autres classes vont souvent ressortir. Et si le seuil est très haut ou/et il y a peu de classes dans le dossier pris comme entrée, aucune classe sortira.

Le premier type de classe qu’on remarque est le « main ». Dans notre cas JFreeChart.java est couplé plus souvent, soit 634 fois. On s’attend à ce qu’il soit plus couplé que d’autres méthodes. Cependant, 634 semble être un immense nombre, il serait probablement une bonne idée de tout de même les analyser.

Dans un autre cas, la fonction main aurait aussi pu sortir comme la plus couplé, sans être une classe divine. Certains types de classes devront mentionner beaucoup d’autres classes, par exemple les contrôleurs, et alors l’architecture d’une application Java est telle que le programme *egon* n’est pas toujours un outil précis pour trouver des classes divines. Lors de l’analyse, il faut porter une attention particulière pour être sûr qu’on ne déclare pas une classe comme divine si elle ne l’est pas. Une façon serait probablement de calculer le nombre de lignes moyen/médiane par classe et le nombre de couplage moyen/médiane par classe pour avoir un niveau de comparaison.

Le deuxième type de classe qui sortira lors de l’analyse sont les classes utiles. Une classe utile est créer pour faciliter l’écriture du code et englobé des problèmes qui reviennent souvent. Par exemple, des classes qui travaillent des types de données spéciaux vont souvent revenir. Dans notre cas, nous avons TextUtils, SerialUtils et DatasetUtils qui sont sorties.

Des classes utiles peuvent parfois être grand en quantité de lignes, car elles doivent être bien documentés, pour faciliter l’utilisation et certaines d’entre elles nécessitent plus méthodes pour englober spécifications. Il faudrait donc peut-être changer le calcul du nombre de ligne pour calcul de nombre de ligne de code.

Ceci nous apporte à un autre point important d’*egon*: les commentaires. Si un projet est réalisé par plusieurs développeurs qui ne commentent pas de la même façon (et que certain ne commentent pas), la courbe de répartition du nombre de lignes sera influencée. Certain code avec beaucoup de ligne de code pourrait ne pas sortir dû à des package écris de façon plus concise mais avec plus de commentaire.

En conclusion, *egon* a besoin de quelques améliorations pour être un bon indicateur des classes divines et ne sera probablement jamais parfais. Mais cette méthode peut toujours être utile pour donner une indication de quelles classes sont à risque d’être des classes divines si le développeur l’utilisant le fait d’un œil critique.