Questions :

1/

« Le niveau de documentation des classes est-il approprié par rapport à leur complexité? »

Métriques utilisées :

* CLOC/LOC : le nombre de lignes de commentaires sur la taille physique, nous permet de déterminer le niveau de documentation ou la densité de commentaires contenu nécessaire à un code, la densité doit être proportionnelle à sa taille physique, pour diminuer le niveau de complexité
* Complexité cyclomatique de McCabe : Nécessaire à la mesure de la complexité de notre code

2/

« La conception est-elle bien modulaire? »

Métriques utilisées :

* CSEC qui déterminait une mesure du couplage dans notre tp1, il nous permettait de déterminer le niveau de couplage de chaque classe dans un code, donc intéressant pour une conception modulaire
* LCOM (Lack of cohesion Method): Indicateur d’une mauvaise cohésion du code, il est intéressant de l’utiliser car dans une conception modulaire car une forte cohésion est signe d’une bonne encapsulation

3/

« Le code est-il mature ? »

Métriques utilisées :

* AGE : Qui nous permet de déterminer l’âge d’un fichier depuis sa date de création, ains plus un code est jeune, moins il est plausible que toutes les vérifications nécessaires a son fonctionnement ont été faites donc qu’il n’est pas assez mature et soit encore en sa phase alpha
* On peut aussi déterminer le nombre de commit Nombre de commits effectue dans un code ou une classe comme métrique nécessaire à sa maturité car moins il y a de commits moins il y a de chance que ce code soit en une version déplorable

4/

« Le code peut-il bien être testé automatiquement ? »

Métriques utilisées :

* RFC (Response For Class) : Il permet de déterminer le nombre de méthodes qui répondent a l’execution d’une classe, il est interessant de l’utiliser