Benjamin Bourgeaux
Lucas Hélaine

Rendu le 4 avril 2016

TP3

Question 1

Pour le bon fonctionnement du programme, il faut modifier le chemin vers lequel s'écrit le fichier de log.

Question 2

a) Programmation par composant

La programmation par composant permet d'améliorer la cohésion tout en diminuant le couplage des éléments de l'application. Chaque composant a une fonctionnalité et est plus indépendant des autres composants. La clarté du code s'en fait ressentir en évitant le dispersion de celui-ci et des ses fonctionnalités.
De plus, en ne programmant pas par composant on peut avoir une rupture de l'encapsulation et des problèmes liés à l'héritage. En effet, la plupart de ses problèmes proviennent de deux gros problèmes : la dispersion et l'enchevêtrement du code.
La programmation par composant, reposant notamment sur la programmation par aspect permet de limiter la dispersion du code et supprime l'enchevêtrement. Ainsi, les fonctionnalités de chacun des composants est indépendante du reste du code et évite de sortir du paradigme de la programmation objet.

b) Programmation par contrat

La programmation par contrat permet de définir les règles d'une fonction, d'une méthode ou d'un composant. Ces règles vont être testées avant l'appel des fonctionnalités et sur les retours des méthodes. Ces tests permettent de s'assurer de la cohérence des méthodes.
La programmation par contrat permet de certifier "correct" un programme. En effet, s'assurer qu'un composant est correct permet de traiter ses sorties avec un excellent indice de confiance et de diminuer les bugs.
La programmation par contrat peut être la pierre angulaire de la répartition de tâches dans le développement d'une application, la confiance sur le code tierce pour un développeur et la qualité des composants fournis et assemblés.
Exemple tiré de wikipedia :
Une fonction calculant une racine carrée d'un nombre réel peut être vérifiée de la manière suivante en pseudo-code.

- *Fonction calculant la racine carrée de la valeur x*
    - *Précondition : $x \geq 0$*
    - *Postcondition : résultat ≥ 0 et si $x \neq 1$ et $x \neq 0$ alors résultat $\neq x$*

*La précondition assure que le développeur utilise la fonction correctement, alors que la postcondition permet au développeur de faire confiance à la fonction. La postcondition donnée en exemple est relativement faible, elle ne précise rien sur la valeur numérique attendue. Cette formulation pourrait être complétée par : résultat * résultat = x (définition d'une racine carrée). Il faut néanmoins faire attention à ce que cette expression puisse être effectivement vérifiée. En toute généralité, il faudrait ici faire attention aux erreurs de calculs numériques du carré sur un nombre flottant !*