



Analyse statique et métrique du logiciel

TP5

Mohammed BENAOU

1 Introduction :

ce rapport illustre comment on peut analyser le code des classes java au travers des différentes métriques vues en cours. Dans un premier temps on va télécharger Eclipse puis on lance l'installation du plugin Eclipse Metrics

2 Exercice 3 : Metrics pour Eclipse

Après l'installation avec succès du Plugin Eclipse Metrics on va tout d'abord réaliser un refactoring de code par extraction de méthode tous ça pour atteindre un nombre cyclomatique strictement inférieur à 6 pour la méthode getComponents.

Le nombre cyclomatique de la méthode getComponents était 11.

Après L'extraction de méthode le valeur est devenue 3

On a créé trois méthode test1(), test2(), test3().

Voilà le code après L'extraction :

```
public class Goldbach {
    boolean[] isprime;
    int primes;
    int[] list ;

    public int getPrimes(int n){
        // count primes
        int primes = 0;
        for (int i = 2; i < n; i++)
            if (isprime[i]) primes++;
        return primes;
    }

    public void test1(int N){
        for (int i = 2; i * i < N; i++) {
            if (isprime[i]) {
                for (int j = i; i * j < N; j++)
                    isprime[i*j] = false;
            }
        }
    }

    public void test2(int N){
        primes = getPrimes(N);
        list = new int[primes];

        System.out.println("Done tabulating primes.");
        // store primes in list
        int n = 0;
        for (int i = 0; i < N; i++)
            if (isprime[i])
                list[n++] = i;
    }

    public void test3(int N){
        int left = 0, right = primes-1;
```

```

while (left <= right) {
    if (list[left] + list[right] == N) break;
    else if (list[left] + list[right] < N)
        left++;
    else right--;
}

public String getComponents(int N) {
    isprime = new boolean[N];
    for (int i = 2; i < N; i++)
        isprime[i] = true;

    test1(N);
    test2(N);
    // check if N can be expressed as sum of two primes
    int left = 0, right = primes-1;
    test3(N);
    if (list[left] + list[right] == N)
        return( list[left] + " + " + list[right]);
    else
        return(N + " not expressible as sum of two primes");
}
}

```

Metric	Total	Mean	Std. Dev.	Maximum	Resource
(default package)		3	1	4	/Goldbach/s
Goldbach.java		3,4	0,49	4	/Goldbach/s
Goldbach		3,4	0,49	4	/Goldbach/s
test1	4				
test3	4				
getPrimes	3				
test2	3				
getComponents	3				

3 Exercice 3 : Analyse du code gestion des batiments

Après la vérification du code de chaque classe voilà le tableau des valeurs des métriques

	SLOC	DLOC	Nombre cyclomatique	AF	AC	DMS
Produit	25	0	1	2	0	1
catalogue	7	0	1	1	2	1/2
Boncommande	47	0	1	3	3	1/2
Interface	219	0	1,083	1	3	1/5
Client	15	0	1	2	2	1/2
Valeur	22	0	1	2	2	1/2
Personne	13	0	1	2	0	1

Metric	Total	Mean	Std. Dev.	Maximum	Resource
TP2/src/Interface.java	1,03	0,171		2	/TP2/src/Interface.java
(default package)	1,03	0,171		2	/TP2/src/Interface.java
Interface.java	1,083	0,276		2	/TP2/src/Interface.java
Catalogue.java	1	0		1	/TP2/src/Catalogue.java
Produit.java	1	0		1	/TP2/src/Produit.java
Personne.java	1	0		1	/TP2/src/Personne.java
Client.java	1	0		1	/TP2/src/Client.java
Vendeur.java	1	0		1	/TP2/src/Vendeur.java
BonCommande.java	1	0		1	/TP2/src/BonCommande.java

Finalement on voit clairement que Le nombre cyclomatique de vos méthodes est il inférieur à 10