Method Execution Report

Date: 13/04/17
Type: FN / NF
System: Evolve

Session overall:

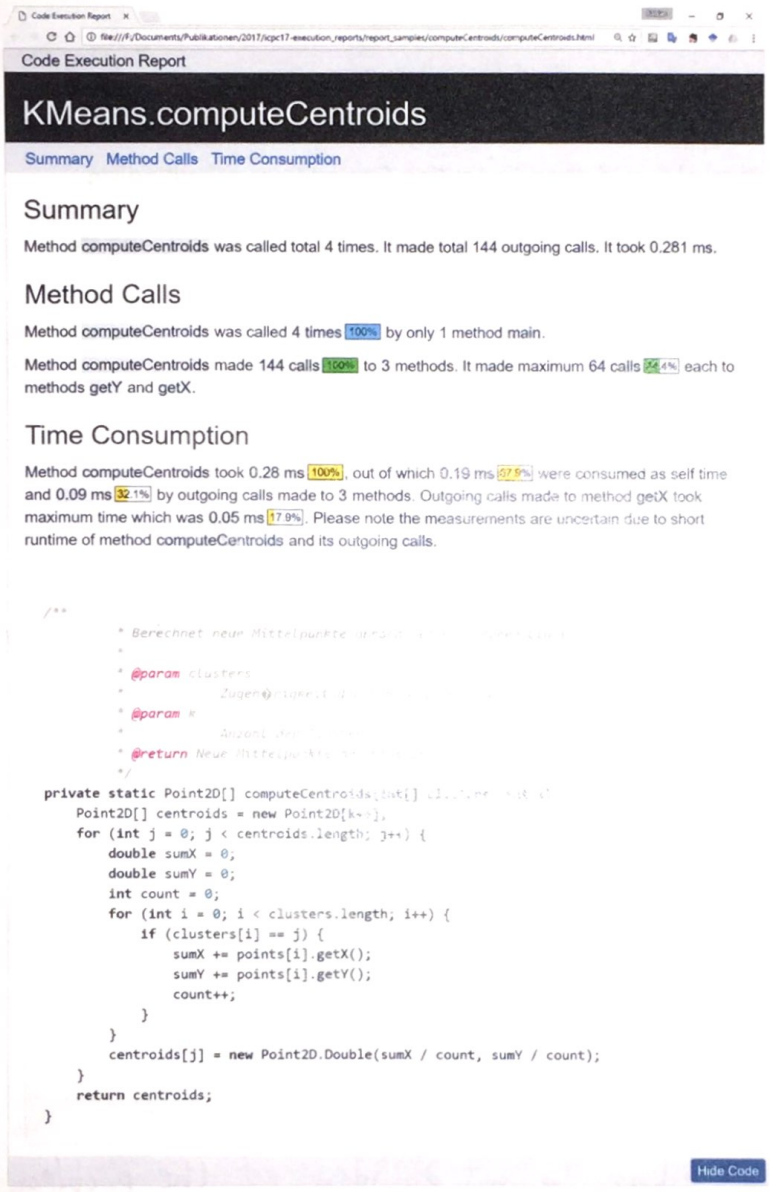
This session aims at gathering impression of the *Method Execution Report* from practitioners
The session is relatively short, only a few minutes long. The session is divided into three phases. The first phase is about yourself. The second phase present Method Execution Report, and the third phase is about the evaluation of the report.
Note that all your answers are treated anonymously.

Phase 1: About yourself

- Are you Female or Male ? Male
- How many years of experience do you have in programming? I learned programming in 2004, but years of strong programming I would say 5 years
- How long have you been programming in Java for? Same as above, but I would say 3 years
- Which Java programming environments (IDE) are you familiar with? Eclipse, Netbeans
- Which other programming languages and programming environments do you use?
  Languages: C, C++, M (MatLab), Mosel (optimisation suite), Python
  IDEs: Edipe, Netbeans, text editors (e.g. Atom), Cloud9
- While programming, if your application does not behave as you expect, what do you usually do? How do you usually debug an application?
  Printing variables here and there to get an idea of the problem.
- How do you usually do to improve the performance of a particular method?
  Analyse the bottleneck (try to find it and then find a way to ~~improve~~ reduce it)

Phase 2: Description of Method Execution Report

Method Execution Report is a textual and interactive report that summarizes the execution of a particular method for a given software execution. The report provides an overview of the dynamic calls and time consumption.



Code Execution Report

KMeans.computeCentroids

Summary Method Calls Time Consumption

Summary

Method computeCentroids was called total 4 times. It made total 144 outgoing calls. It took 0.281 ms.

Method Calls

Method computeCentroids was called 4 times 100% by only 1 method main.

Method computeCentroids made 144 calls 100% to 3 methods. It made maximum 64 calls 44.4% each to methods getY and getX.

Time Consumption

Method computeCentroids took 0.28 ms 100%, out of which 0.19 ms 67.9% were consumed as self time and 0.09 ms 32.1% by outgoing calls made to 3 methods. Outgoing calls made to method getX took maximum time which was 0.05 ms 17.9%. Please note the measurements are uncertain due to short runtime of method computeCentroids and its outgoing calls.

```
/**
         * Berechnet neue Mittelpunkte
         *
         * @param clusters
         *            Zugeordneter
         * @param k
         *            Anzahl
         * @return Neue Mittelpunkte
         */
private static Point2D[] computeCentroids(int[] cl
    Point2D[] centroids = new Point2D[k];
    for (int j = 0; j < centroids.length; j++) {
        double sumX = 0;
        double sumY = 0;
        int count = 0;
        for (int i = 0; i < clusters.length; i++) {
            if (clusters[i] == j) {
                sumX += points[i].getX();
                sumY += points[i].getY();
                count++;
            }
        }
        centroids[j] = new Point2D.Double(sumX / count, sumY / count);
    }
    return centroids;
}
```

Hide Code

Figure on the left gives an example of the report. The report is structured into three sections, and lists the code of the method.

The summary gives first essential data, including the number of incoming and outgoing calls, and how long the summed executions of the method took.

The second section details the description of the method calls, including the most important caller and callees.

The third part provides further information about the timing.

Phase 3:

TIME BEGIN:

QUESTION 1:

What do you think about the content of the textual description?

- I find it easy to understand? *(strongly agree, agree, neutral, disagree, strongly disagree)*

Please, justify

Agree, But it would be much better if instead of plain text the information was presented as a map or diagram.
More visual => easier to understand.

- I find it useful? *(strongly agree, agree, neutral, disagree, strongly disagree)*

Please, justify

Agree. Same justification as above, it is usefut, but I still have to make the map in my head or in a piece of paper.

QUESTION 2:

What do you think about the interaction and the visual elements offered by the report?

- I find them easy to understand? *(strongly agree, agree, neutral, disagree, strongly disagree)*

Please, justify

- I find them useful? *(strongly agree, agree, neutral, disagree, strongly disagree)*

Please, justify

QUESTION 3:

Overall, do you feel that such a report is useful?

(strongly agree, agree, neutral, disagree, strongly disagree)

Please, justify

Agree, although I maintain my first comment.
~~I think it could be~~

QUESTION 4:
In what scenarios and for solving which maintenance tasks would developers use Method Execution Reports?
Please, justify

- Performance analysis and improvement: Time execution info may help to find bottlenecks in the method calls workflow.
- Bug fixes: metho call info can help to find bugs, specially when dealing with recursive methods.

QUESTION 5:
What tools would you use instead of Method Execution Reports to retrieve the same information?
Please, justify

Do not know/remember now. I would need to research a little.

QUESTION 6:
Do you have any suggestion on how to improve the report? Any critic?
Please, justify

My first comment, replace text with maps, such as:

- Method call maps: showing how methods call each other, perhaps displaying the numbers there also.
- Distribution charts: charts showing how method calls and execution time is distributed among methods. Eg. barcharts.
- Recursion maps: showing how methods call each other but pointing out in which parts of the code the calls occur.
(Or something like that)

TIME END: