Aharouni Refaël

Exam Software Metrics

st v2425-25

Short questions:

1. There are several phases in the design:
   - *1.1.* **The architectural design** where we identify the sub-systems
   - *1.2.* **The abstraction specification** where we specify the sub-systems
   - *1.3.* **The interfaces design** where we design the sub-systems interfaces
   - *1.4.* **The component design** where we decompose the sub-systems into components
   - *1.5.* **The Data Structure Algorithm design** where we create the data structures to hold the problem data
   - *1.6.* **The Algorithm design** where we create the functions of the problem

2. There are four external quality attributes:
   - *2.1.* Maintainability
   - *2.2.* Reliability
   - *2.3.* Reusability
   - *2.4.* Usability

3. Code coverage measurements have some flaws.

First, to achieve a coverage score of 100%, we must test all the cases, even those that are not necessarily important and don't bring anything to the code (for example, just test a null case, even if we know that all the variables are not null and will never be). This can take a long time, especially if there are several cases.

Finally, the code coverage just means that the tests have been tested and they act conformal as the expectations. However, the code coverage does not guarantee that we will not have any bugs or exceptions like StackOverFlowException in recursive or OutOfRangeException in iterative. For example, if I have a function that does not return anything (None in Python or void functions in C#) and if in my code, I go from 0 to the length of an array instead of the length of the array – 1, it will generate an OutOfRangeException and still have the test validated because nothing was returned.

4. The requirements traceability is the fact of being able to respect all the requirements of a client during a project and to make sure that everything is complied with by his demands.

5. Software Design is the fact of developing a solution to satisfy the software requirements. This can be done through several design phases as the understanding of a problem by looking from different angles to discover the design requirements. Then, it is necessary to identify one or more solutions and choose the best one depending on the experience of the designer and the available resources. After that, we describe the components of the design using graphical forms for example and we repeat the process until the design is entirely expressed in primitive terms.

6. The GQM paradigm is a process in which a designer uses metrics in a software engineering process to make sure that the project has defined and precise goals. The first part is the goals (G). This is about ensuring that the project's goals are clear and precise.

The second part is the questions (Q). This part has for goal to answer the What/When/Where/How questions about the project and the measurements.

Finally, we have the Metric (M). In this part, we use some metrics to answer the questions and accomplish the goals of the project.

7. Organization measures are tools that ensure that the design of a project is well done and efficient. First of all, some tools like Radon can be used in order to determine the number of Lines Of Code, Comment Lines of Codes and Blank Lines. Then, we have some metrics that have for goals to improve the code, just as Pylint or Flake8 and to cover all the tests with Coverage in Python.

8. Cohesion is a measure of how a component fits together. In theory, a component should implement a single entity or function, so that if changes must be made, they will happen in a single cohesive component. Hence, all the operations of the entity are included with the object. However, in practice, complex tasks cannot be expressed with simple and single elements. Hence, a single-element module has an element that is too complicated or too narrow, which would increase the coupling to the other modules. This is why cohesion is balanced with coupling and unit complexity.

9. A ‘downtime’ is a moment where a part of a system does not work and leads to bugs or unexpected behavior.

10. A Depth in Tree is the distance between the root of the inheritance tree and the class itself.