[Assignment 1] Software Verification & Validation

Author

- **Name**: Steve Djumo Kouekam
- **Email**: djumokouekam.steve@ictuniversity.edu.cm
- **Matricule**: ICTU 2022 27 55

Question 1 : The values checks of static analysis ?

The values checks of static analysis involve examining the values assigned to variables and ensuring that they are within acceptable ranges or meet certain criteria.
This can include checking for *null* or *uninitialized variables*, ensuring that integer values are not *too large or small*, and verifying that *string conform to expected formats*.

Question 2 : The stages of static analysis ?

Static analysis typically occurs during the *development stage of the SDLC*, before the code is compiled and executed. It can also be used during code reviews or as part of a continuous integration process.

Question 3 : The various types of static analysis tools ?

There are a few types of *static analysis*:

- **Control Analysis**: focuses on the control flow in a calling structure
- **Data Analysis**: ensures that data is appropriately used and that they operate accurately. They are 2 mothods : *Data Dependency* and *Data-flow Analysis*
- **Fault Analysis**: helps analyze the failures in different model components. The model design specifications are checked to ensure that the failure are recognized. It uses the output-input description to identify the cause of failure.
- **Interfaces Analysis**: verifies the simulations, allowing developers to check the code and ensure that the interface fits the model. It also focuses on how well the interface is integrated into the system.

On the other hand, there are a few *static analysis tools* in the wild, including:

- **Code Linters**: These tools analyze the source code for potential errors, style violations, and other issues.
- **Security Scanners**: These tools look for vulnerabilities in code that could be exploited by attackers.
- **Complexity Analyzers**: These tools measure the complexity of code and identify areas that may be difficult to maintain or modify.
- **Performance Profilers**: These tools analyze code to identify performance bottlenecks and suggest optimizations.

Question 4 : How to use the static analysis ?

To use static analysis, you need to select a tool that meets your requirement and integrate it to your development process.
This may involve configuring the tools to analyze specific files or directories, setting up rules or thresholds for detecting issues, and reviewing reports generated by the tools.
Static analysis tools are even more useful when used in conjunction with other testing tools/method to ensure comprehensive coverage.

Question 5 : How can you test the completeness, clarity, and correctness of a requirement ?

To test the completeness, clarity, and correctness of a requirement, you can use various techniques such as **reviews, walkthroughs, and inspections**. These techniques involve analyzing the requirement document and cheing if it meets certain criteria such ash being *unambiguous, testable, and vrifiable*. You can also use tools such as *traceability matrices* to ensure that all requirement are coverd by tests.

Question 6 : How do you provide evidence for design verification ?

To provide evidence for design verification, you can use various technique such as **testing, simulation, and prototyping**. These techniques involve *creatting a design that meets the requirements and then verifying it against those requirements*. You can also use tools such as *code coverage analysis* to ensure that all parts of the design have been tested.

Question 7 : How do you prepare the fianl report after a software inspection meeting ?

To prepare the final report after a software inspection metting, you should document all issues identified during the inspection along with their severity level and proposed solutions.
You should also document any decisions made during the meeting regarding changes to the software or requirements.

Question 8 : How can we mathematically prove that requirement are complete and consistent and that the system will meet the functional and non functional requirements ?

To mathematically prove that requirements are complete and consistent and that the syste will meet functional and non-functional requirements, you can use **format methods** such as **model checking or theorem proving**. These methods involve creating a mathematical model of the system and then using logic to prove that it meets certain properties or specifications. However, these methods are often complex and require specialized knowledge in mathematics and computer science.