1. <u>Rework In Software Engineering</u>

The rework phase is where you fix bugs found during the System Testing phase.

Rework in software development is the additional effort of redoing a process or activity that was incorrectly implemented in the first instance or due to changes in requirements from clients.

Basically developers are paid to write code to implement new product features. When the features are done the product as a whole is “released” to a dedicated test team for system testing. They’ll click through it, try and break it etc.

Any bugs or issues found during testing are assigned by the testers to developers for “rework” which is just a fancy term for “essential maintenance” (ie: bug fixing)

When rework is complete the updated product is released to testers again and if the testers find more bugs then the developers are in a big trouble. There might be another rework phase as a result of finding new bugs - management might make the call to start a new phase or not.

2. <u>Verification and Validation In Software Engineering</u>

Verification is the process of checking that a software achieves its goal without any bugs. It is the process to ensure whether the product that is developed is right or not. It verifies whether the developed product fulfils the requirements that we have. Verification is static testing. Verification means Are we building the product right?

Validation is the process of checking whether the software product is up to the mark or in other words product has high level requirements. It is the process of checking the validation of product i.e. it checks what we are developing is the right product, it is validation of actual and expected product. Validation is the dynamic testing.

3. Evolution in Prototype Model

The evolutionary prototyping model combines incremental and extreme models. This model involves a series of prototyping refinements. The first assignment is to design and split the system into several independent modules. Each module is given required functionalities and then presented to the customers. The customer feedback is gathered, and the model is again refined. This process goes on and on until the final satisfactory prototype is created. It will reach a point where you and your customers reach a point that this product is good. That is now the point where you stop the cycles of fine-tuning the project according to the customer's feedback.

4. RTM (Requirement Traceability Matrix)

Requirements Traceability Matrix (RTM) is a document used to ensure that the requirements defined for a system are linked at every point during the verification process. It also ensures that they are duly tested with respect to test parameters and protocols. It is a tabulated document which defines multiple to and fro relationships between use cases (requirements) and test cases. The Requirements Traceability Matrix (or RTM) is usually developed in concurrence with the initial list of requirements and is updated simultaneously with the newly-developed design specifications and test protocols. Ideally, every test step in the testing protocol should be traced with requirements that are specific to that particular step.

RTM proves to be a powerful planning tool when it comes to determining the number of tests that are required, the types of tests that are required, and whether these can be automated, done manually, or whether any existing tests can be re-used.

Using the RTM this way can result in the most effective test execution and provide the overall defect status, focusing majorly on business requirements. More importantly, estimation or analysis of the result on the QA team's work, with respect to re-working on test cases, can be eased via RTM.

5. Coding Standards

Coding standards are collections of coding rules, guidelines, and best practices. Using the right one — such as C coding standards and C++ coding standards — will help you write cleaner code.

Coding rules and guidelines ensure that software is:

- Safe: It can be used without causing harm.
- Secure: It can’t be hacked.
- Reliable: It functions as it should, every time.
- Testable: It can be tested at the code level.
- Maintainable: It can be maintained, even as your codebase grows.
- Portable: It works the same in every environment.

There are four key benefits of using coding standards:

1. Compliance with industry standards (e.g., ISO).

2. Consistency of code — no matter who writes the code.

3. Software security from the start.

4. Reduced development costs and accelerated time to market.