ExamQuestions

woensdag 1 februari 2023 14:07

1. What is a complex system and how it can be developed, tested, and validated?

Complex system is the idea of an integrated system which is not only about "one" variable. It interacts with multiple perspective to achieve what it is intended. Because it is more a little different than a normal system, we would tackle it with three main paradiam,

Communication: tackles with sysml.
Complexity: Model driven engineering. MDSE: concerns with the bigger picture.
Lack of understanding: tackle with creativety

2. What are the scope of System Engineering and its relationship to traditional engineering?

The main difference between system engineering to regular engineering is the holisticness approach. Both involve problem solving in the sense of problem and solution. Yet the view in the system is more about the whole. It includes also technical project manager.

3. What is Automotive Software Engineering and what is the new software operating model for OEMs?

Automotive Software Engineering is a specialized field of software engineering that deals with the development and maintenance of software for the automotive industry, including for in-vehicle systems and related software products.
The new software operating model for Original Equipment Manufacturers (OEMs) is a more agile and flexible approach to software development that allows for faster and more efficient delivery of software updates and new features. This operating model involves the use of DevOps practices, cloud-based development and testing environments, and the adoption of open source software and components. The goal of this new model is to increase collaboration between OEMs and their suppliers, and to promote innovation and cost savings through the reuse of software components and tools.

4. Which are the most important Automotive Standards? Give a short description of the specific standard.
   a. Misra: Coding standard. Standard for c++.
   b. SOTIF: Safety of the intended functionality. The objective is to designed a properly working equipment which was built, and tested to fulfill the equipments given functionality.
   c. Autosar:
      This standard provides a common software architecture for in-vehicle electronic systems. Allow to develop for while not being bounded to the embedded system.
   d. Cyber security guidlines.
5. What is Software Development Process and which are its phases? Describe in short each phase

   a. Requirement gathering: identify constumer, document requirements
   b. High level design: How the components will work. Specific interaction between components
   c. Development: code generation
   d. Testing: programming and fixing bugs.
   e. Testing: requirement , high level design, low level design, and development.
   f. Deployment: check ak the hardware, parallel
   g. Maintainance. Bug tracking and fixing.
   h. Reviewing!

The Software Development Process is a systematic approach to creating software systems. It involves a series of phases that ensure that the software is developed in a controlled, repeatable, and efficient manner. The phases of the software development process typically include:

   a. Requirements Gathering: This phase involves identifying and documenting the requirements for the software

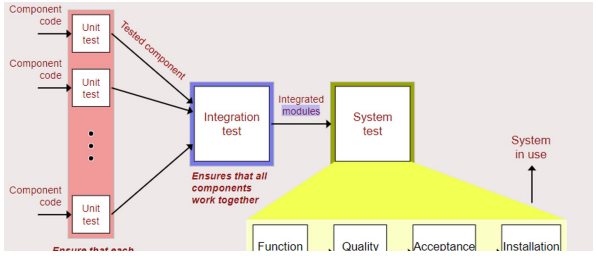
system. This includes defining what the system should do, and what constraints and limitations it must adhere to.
   b. Analysis and Design: In this phase, the requirements are analyzed, and a high-level design of the system is created. This includes defining the architecture, data structures, algorithms, and interfaces.
   c. Implementation: During this phase, the code is written and the system is built. This involves translating the design into code and integrating the various components of the system.
   d. Testing: In this phase, the system is thoroughly tested to ensure that it meets the requirements and functions as expected. This includes both functional testing (to ensure the system does what it's supposed to do) and non-functional testing (to ensure that the system meets performance, reliability, and other requirements).
   e. Deployment: Once the system has been tested and any issues have been resolved, it can be deployed into the production environment. This may involve installation on the target platform, configuring the system, and setting up any required databases or other infrastructure.
   f. Maintenance: This phase involves ongoing support and improvement of the system. This includes fixing bugs, adding new features, and making other changes to improve the system's performance and reliability.

6. What is a requirment and why is it important?
   a. Ensure that the quality of the product is indeed as it should.
   b. It ensures /or at least higher the possibility, that changing the environment would not cause a change in the system. And therefore, make the system more rebust.
   c. Requirment engineering is the concept in enabling the engineering to understand the systme.
   d. Moreover: It increasees the possibility of optimal solution, the decision could be tracked to its source, the quality of the system could be analysed better nd lastly, we essure quality.
7. How do we get good requirements:
   a. Start somewhere: produce an initial step. Maybe from an existing document)
   b. Specification State specific individuals requirement.
   c. Analysis: understanding: the nature of understanding. How and why? Ask yourself always.
   d. Organize: grouping into categories. Embracing more context.
   e. Managments: A complex living entity that evolves over time. model them, and must be properly controlled.
8. How many requirements types are in Automotive Software? Give a short description per type
   a. Busineess requirements, High level requirement, the fundamental nature of the business. Includes strategy, mission and market.
   b. Stackholder requirements/ check if it relates to non functional requirement.
   c. Functional requirements: What the system is supposed to do. The actions of the process. Of course includes also the non functional.
9. Describe in short the properties of a requirement
   a. Requirements often evolve. But the original are also important (the foundation).
   b. Should be clear
   c. Should not be specific.
   d. Who needs this requiements
   e. What needs this requirement.
   f. Should be verified and validated.
   g. Priorotized: what is more important.
10. What is the Context of the system and how to model it?
   a. It defines the boundary of the system. Separate the inside from the outside.
   Context modeling is the process of representing and understanding the situational context in which a system operates. In the context of real-time systems software engineering for the automotive industry, context modeling is used to capture the various aspects of the operating environment, such as vehicle parameters, road conditions, traffic, and user behavior.
   Continue the how to model
11. Architectual Principle: SOLID
   a. Single: responsibility principle: Each system capability should only have one responsibility.
   b. Open closed principle Able to extend the system without modyfing it.
   c. Liskob substitution principle: enables you to replace and sub or parent class without breaking the system.
   d. Interface segragation principle. No code should be dependent on methods it does not use.
   e. Dependency inversion: loosly coupling software modules. High level modules should not depend on low level onces.
12. Describe the architectural views,
   a. Logical View:Concerns with the functionality that the system provides to the end user. SYSMl used to represents the logical view, includes state diagram and system diagram and more.
   b. Process view: deals with the dynamic of the system.explains the system process, its communication and focuses on the run time behaviour of the system.

c. Development view: Illustrates the system from a programmer perspective and its concerns with software managments.
d. Physical view: this takes into account the engineering point of view.It is concern with the physical layer, and the software components attached to it.
e. Scenario!! Dont forget. Make a set of use cases to test the system.

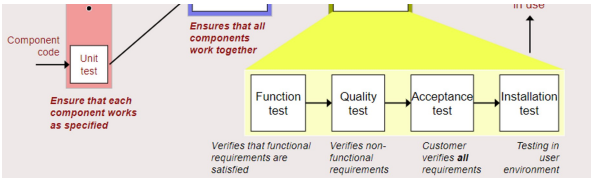
13. Architecture Styles describe 3:
    a. Monolithic: A monolithic kernel is a single large block of code that provides all the essential services for the operating system to function
    b. MicroKernek: has a much small code base and provide only the basic services. The result is a more modular and scalable system. But also increase complexity and overhead.
    c. Event driven application. React to data. This includes listening to data via sensor, driver assistent and more, and react when an activation is made.
14. How can the automotive software architecture stylesbe modeled?Check!!! FOUND THIS ONLINE
    a. Component-Based Architecture: This approach involves breaking down the software system into reusable components that can be easily combined and reused in different systems. The components are designed to interact with each other through well-defined interfaces.
    b. Layered Architecture: This approach involves dividing the software system into layers, each with a specific function or responsibility. The layers are typically stacked on top of each other, with the lower layers providing basic functionality, and the upper layers building on top of that functionality to provide more complex functionality.
    c. Service-Oriented Architecture (SOA): This approach involves breaking down the software system into services that can be reused and combined to create new functionality. Services are designed to interact with each other through well-defined interfaces, and can be located on different systems, allowing for greater flexibility and scalability.
    d. Model-Driven Architecture (MDA): This approach involves modeling the software system at a high level of abstraction, and then generating code from that model. This allows for faster development, and can also help ensure consistency and maintainability of the code.
    e. Microservices Architecture: This approach involves breaking down the software system into small, independent services that can be deployed and managed independently. This allows for greater scalability, as well as the ability to make changes to individual services without affecting the rest of the system.
15. What is Automotive Software Engineering Variant Management, and how it can be implemeted?
    a. Configuration: used when we cofigure parameters of the software without modifying its internal structure. It is used in non safety critical functions. Often refer as runtime. Compiled one, but then used two different configuration when deploting the sofware.
    b. Compliation: When we change the internal structure of the software. Compile it and then deploy on the target ECU. It is used to ensure that the sofware always behaves in the same way. So fclled -design time variability. The designers must decide during design which variant is being developed.
16. Which are the integration stages of Automotive Software Development? Describe in short each phase.
    a. Integration: the stage where softwaer engineer integrate their code or other components with the hardware.
    b. First integration stage: Allow unit and component testing: hardware simulation. DIL, ,SIL,DT
    c. Later integration: source code is integrated together with the target hardware, which is is integrated into a complete electrical system of the car -HIL.
17. Tyoes of integration
    a. Software implementation/ integration. Two or more components are put together, and their joint functionality is tested.
    b. Software-hardware implementation/ integration: the intersection between software and hardware. MIL, SIL.. Deployed to the target ECU.
    c. Hardware implementation/integration: Focus is on the hardware. HIL.
18. Describe the logical organisation of testing:
    a. Unit test/ component code -> Integration test : ensures that all components work together. -> Integrated modules: System testThen we go over (functional test, quality test, acceptance tst and instalation test

Test Driven Development:
Try to break the system. Creating test case and using it to test the pipeline. But its hard: why: you have infinite test to make but needs only one to fail. Therefore we need to run a lot of tests



19. What is Unit test
    a. It is a basic test, which is performed on individual entities of software (class, source code.

19. What is Unit test
    a. It is a basic test, which is performed on individual entities of software (class, source code.
    b. The goal : to find defects related to implementation.
    c. Scheme:
        i. create automated test case(combine Individual method with the data .
        ii. Result is then compared to the expected resutl Assert



20. Component testing, and how it is realized:
    a. Integration testing: we want to link between real components to simulated components. The focus is about interaction between the verify the structure and behaciour of the interface implemented correctly.
    b. Simulation of the environment. MIL
    c. Hardware simulation HIL
    d. Digital wtines.
21. What///

    Contain all the design information of the electrical system. It is controlled differently in different versions of car. Each of the requirement set has a set of requirements linked to it.
    What are the tools? In Rhapsory you would find some.

    a. Contains all elements of the design of the electrical system of the vehicle.
    b. Grow over time and is version controlled and can be controlled from different vehicle.
    c. Each requirement has a set of requirements.
    d. The requirements are also linked to one another to show how they are broken down.

        The role of Construction Database in automotive software engineering is to provide a centralized repository for information about the software.
        By providing a central repository for information about the software construction process, Construction Database can help to improve the efficiency, quality, and reliability of automotive software development and maintenance.

22. What is an Electronic Control Unit (ECU)? Describe in short its architecture.

    ECUs are connected via electronic buses of different types (e.g., Can, FlexRay, Ethernet)
    ECUS are responsible for executing one or several high-level vehicle functionalities defined in the logical architecture - Each logical software component is allocated to at least one ECU
    a. Application sofwatre: number of allocated software components responsible for executing vehicle funcitonalities.
    b. Middlewarew software: responsible for providing services to the application sofware.
    c. Hardware: a number of drivers responsible for controlling different hardware units.

23. What is AUTOSAR? Describe in short its architecture

    Autosar:
    The main objective for it is the break the dependency between the developed software to the used hardware.
    Therefore it is proposed to break it down into two components.
    We don't want that a small change in the hardware would require a total change of the
    Application layer:
    Port and software ports.
    - Reference architecture:
        - Build upon the ECU hardware.
    - Application software: software components that realize a set of vehicle functionalities be exchanging data using interface.
    - RTE: Control the communication between software components.
    - Basic software

    AUTOSAR (Automotive Open System Architecture) is a standard for the development of automotive software systems. It is a collaborative partnership between car manufacturers, suppliers, and tool developers to create a common, open, and standardized architecture for automotive software systems.
    The architecture of AUTOSAR is divided into four layers:

- Application Layer: It contains the high-level functions and services required for the specific application.
- Runtime Environment Layer: It contains the basic software components necessary for the operation of the application software. This layer includes the operating system and middleware components.
- Platform Layer: It defines the hardware-specific interfaces and contains the components required for communication

with the underlying hardware.

- Hardware Layer: It represents the physical components and devices of the electronic control unit (ECU).
- AUTOSAR aims to provide a standardized and reusable software architecture that can be used across different vehicle platforms, reducing the cost and effort of software development while ensuring compatibility and interoperability between ECUs.

.