DO-254 Interview Questions

Simulink Tutorial Series – 1

avionics interviews. Questions

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Table of Contents [hide]

Email *

Website

context of DO-178C / DO178C / DO-178B / DO178B and model answers for an upcoming job interview, you have landed at the right place.

Save my name, email, and website in this browser for the next time I comment.

I'm not a robot

reCAPTCHA
Privacy - Terms

p collection of avionics verification and validation
C / DO-178B / DO178B) that are commonly being asked in any of the aerospace MNCs. I have consulted with a couple of experienced professionals and spent many hours collecting and compiling these Top avionics verification and validation interview questions and preparing these model answers.

Notify me of new posts by email.

All new comments

So, I hope the following list of avionics verification and validation interview questions could be helpful for your upcoming interview and you will learn something new for sure.

Notify me of followup comments via e-mail. You can also subscribe without commenting.

After reading the complete article if you think that I should add some more commonly

Post Comment

fication and validation interview questions, please write in the comment box, I will definitely try to address those in the future so that It could be helpful for others.

All the best for your avionics verification and validation interview.

What is Verification?

Answer:

Verification is the process to evaluate software products to evaluate or determine whether they meet the specified requirement.

The main objective of verification is to ensure that we are building the right software product or the software is correctly implemented as per the specification.

There are various definitions of verification in the software industry. We will see some of them here:

As per **DO-178C** / DO178C / DO-178B / DO178B,

> "**Verification** – The evaluation of the outputs of a process to ensure correctness and consistency with respect to the inputs and standards provided to that process."
>
> **DO-178C**

As per the **IEEE-STD-610**, verification is:

> “A test of a system to prove that it meets all its specified requirements at a particular stage of its development.”
>
> **IEEE-STD-610**

As per **ISTQB**, the verification definition is:

> “Confirmation by examination and through provision of objective evidence that specified requirements have been fulfilled.”
>
> **ISTQB**

As per **PMBOK**, the verification definition is:

> "The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. Contrast with validation."
>
> **PMBOK**

As we can clearly see above, even though there are several definitions, they all mean the same objectives of the verification and provide an answer to the question – ***"Are we building the product right?"***

What is Validation?

Answer:

Validation is the process to evaluate if the software product meets the customer's requirements / meets the user's needs. In other words, It consists of the activity to provide the answer: ***"Are we building the right product?"***

In most of cases, the validation is normally done at the end of the software development. However, some organizations practice executing the validation process during software development. No wonder, people do debate in the software industry as to when the validation needs to be done – at the beginning or at the end.

Technically speaking, it needs to be done at the beginning to ensure that the low-level requirement specification meets the customer's requirements and then keep doing the process during the development. Finally, at the end of the software development phase, there should be a final validation to be done. Please do comment in the comment box below, if you think something different.

Similar to verification, validation has several definitions in the software industry. We will see some of them here:

As per DO-178C / DO178C / DO-178B / DO178B,

> "validation – The process of determining that the requirements are the correct requirements and that they are complete. The system life cycle processes may use software requirements and derived requirements in system validation."

DO-178C

In the **IEEE-STD-610**, validation is defined as:

> "An activity that ensures that an end product stakeholder's true needs and expectations are met."
>
> **IEEE_STD-610**

As per ISTQB:

> “Confirmation by examination and through provision of objective evidence that the requirements for a specific intended use or application have been fulfilled.”
>
> **ISTQB**

As per PMBOK, validation is:

> The assurance that a product, service, or system meets the needs of the customer and other

> identified stakeholders. It often involves acceptance and suitability with external customers. Contrast with verification.
>
> **PMBOK**

Can you explain the difference between verification and validation?

Answer:

As we have seen both the definition of verification and validation, now we can conclude the differences between them.

1. Verification is the process of evaluating the software product to determine if it meets the software requirement specification, whereas validation is to ensure that our software meets the user's needs.

2. Verification addresses the question: ***"Are we building the product right?"*** whereas,

Validation address the following: ***"Are we building the right product?"***

3. The verification process will evaluate the following items – software plan document, Software requirement specification (SRS), Software Design Specification, Software Code, Software Test Cases etc. On the other hand, validation evaluates the actual product as a whole.

4. The verification activities include – Review, Walkthrough, Analysis, Inspection, and Testing.

In the case of validation, only Black Box testing and User Acceptance Testing (UAT) are done in the context of DO-178C / DO-178B.

Can you explain the Verification Process?

Answer:

Some people think verification is only about testing. But, in theory, the verification process consists of the following activities: Review, Walkthrough, Analysis, Inspection, and testing.

So, clearly, the verification is not only about testing or running the test cases. Verification tries to achieve error-free software products. The purpose of the software verification process is to identify the errors that were injected/introduced during the software development.

Can you give an example of both the verification and validation process?

Answer:

Let's take a very simple hypothetical example to explain the verification and validation process.

Before we jump into the example, **Let's understand the context**:

Company X is building a brand-new aircraft and wants to outsource one of the LRU – Cockpit Display to another company-Y. Company Y is mature and has previous experience in building the cockpit of the airplane. So, Company X has given a contract to build the Cockpit and deliver it to the Company-Y.

Now, Company X provides the following requirement for Cockpit Display Unit to Company-Y:

```
Req-1 (Company-x):

"The cockpit display unit shall display Air Speed and Altitude."
```

Now, let's say, the Company Y receive this requirement and misinterpreted and wrote their requirement as:

```
Req-1 (Company-Y):

"The cockpit display unit shall display Air Speed."
```

As you can see, the Altitude part of the requirement is missing here. Now, the software engineer of the Company-Y will look into the above requirement and develop the software to display only Air Speed on the cockpit.

Verification Example:

Now, that we have understood the context, let's understand what verification is. The verification process would verify if the software has been implemented correctly as per the requirement [Req-1 (Company-Y)] to display only the Air Speed in the cockpit.

Validation Example:

The validation process will answer the question: "Are we building the right Cockpit Display Unit which will meet end user's needs?" The answer is definitely No, in this case.

So, Company-X wanted to display Altitude as well along with Air Speed on their aircraft cockpit. But Company-Y built the Cockpit Display Unit which only displays Air Speed (not Altitude).

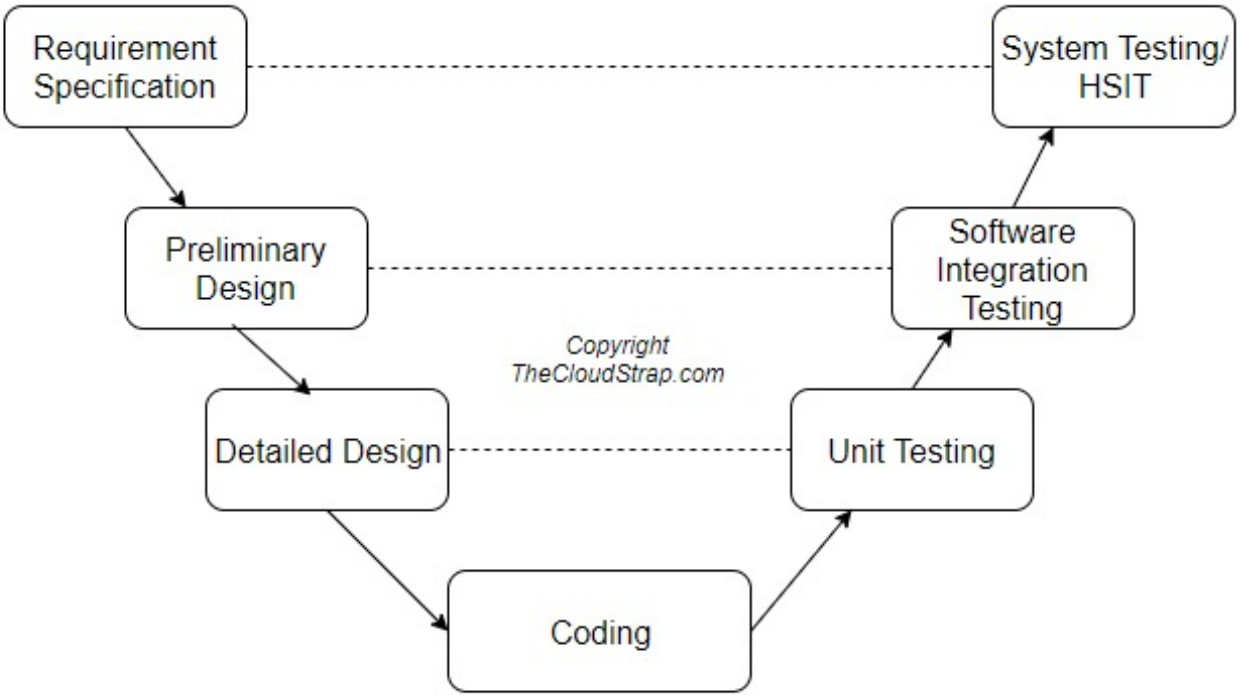
The validation process will uncover such kind of issues/problems.

Note: Hopefully, now it is clear and you understand the difference between verification and validation. If not, please comment in the comment box below. This is one of the popular Avionics Verification and Validation interview questions.

What is V-Model?

Answer:

The V-Model is a very popular software development and verification life cycle model. It has been widely followed by many organizations.



V-Model

What is Unit Testing?

Answer:

Unit Testing also popularly known as Component testing is a process to test individual units, modules or components. The unit can be considered as atomic software code. Sometimes, we the unit as a module, function, or component. The definition of the unit may vary based on the project and industry.

Here, in this article, I am talking about Unit Testing with respect to DO-178C. The unit testing objective is:

1. The unit/module/component has been implemented correctly as per the requirements

2. The unit/module/component implements the algorithm correctly as per the requirements
3. Eliminate the error/bug at the earlier stage of the software life cycle which reduces the overall cost of the project

What is Software Integration Testing?

Answer:

The software integration testing typically needs to be performed after the execution of Unit level testing. The software integration testing is done to ensure that the software components/modules/units are interacting with each other in the correct manner and meeting the software requirement specification and software architecture.

For example:

We have 100 units or functions in our software – U1, U2, U3,.., U99, U100. During the Unit Level testing, we will focus on one unit at a time (U1 or U2 etc) and test the particular unit to ensure it meets the requirement.

During the software integration testing, we will focus on functionally connected units at a time. Maybe, U1, U18, and U87 are functionally connected and interact with each other to perform a function. So, we will focus on U1, U18, and U87 in the software integration testing and verify if they interact with each other correctly and perform the operations in the right sequence to deliver the functionality.

We can find the following issues/errors during software integration testing:

1. Errors during parameter passing between functions.
2. Incorrect initialization of variables, pointers, and constants.
3. Incorrect updates of Global Data.
4. Incorrect sequencing of operations.

These errors can not be found during Unit Level testing.

Note: This is one of the popular Avionics Verification and Validation interview questions.

What is Hardware-Software Integration Testing (HSIT)?

Answer:

The Hardware-Software Integration Testing is also known as HSIT. Hardware-Software Integration Testing (HSIT) is mainly done to identify the errors or error sources associated with the software executing within the target computer environment. Another objective of HSIT is to ensure that, when the software executes on the target hardware platform, it delivers high-level system functionality.

So, typically, in any organization, first, the unit-level testing is done and then Software Integration testing is performed and then the Hardware-Software Integration Testing is executed.

Hardware/Software Integration testing can identify the following errors:

1. Incorrect software response to hardware failures
2. Errors in Hardware/Software interfaces
3. Stack Overflow
4. Incorrect startup sequence
5. Failure to execute the function within a certain time

Note: This is one of the popular Avionics Verification and Validation interview questions.

What is Hardware-In-Loop Testing?

Answer:

Hardware-Software Integration Testing is often known as Hardware-In-Loop testing. Hardware-Software Integration testing is already explained in previous questions.

What is Black-Box testing?

Answer:

Black-Box testing is a very popular software testing methodology in which the software product or software applications are tested without the knowledge of internal software code implementation and structure.

The name of this testing methodology is given as “Black-Box” since the tester does not have any internal knowledge of the system. So, in the eyes of the tester, the software application is a black box.

Black Box Testing

In this testing methodology, the testers are focused on the input and the output.

Sometimes, the Black-Box testing is also called as Functional Testing, Non-Functional Testing, and Specification Based Testing.

What is White-Box testing?

Answer:

Unlike Black-Box testing, White-Box testing inspects the program behavior by exercising the internal software code structure.

So, in the case of White-Box testing, the tester will have knowledge of internal software implementation, code, and structure. The name “White Box” indicates the capability to see through the internal software code from the tester’s perspective.

White-Box testing is also known as Glass Box Testing, Clear Box testing, Open Box Testing, Structural Testing, or Code-Based testing.

White Box Testing

White-Box testing involves writing test cases based on the software's internal code structure to verify the flow of input-output data and to improve the software design or architecture.

The White-Box testing includes – Data-Flow testing, Path Testing, Coverage Testing, Test minimization, etc.

What is Boundary Value Analysis?

Answer:

There are various Black-Box testing techniques:

1. Boundary Value Analysis
2. Equivalence Partitioning
3. Cause-Effect Graph
4. Adhoc Testing
5. Random Testing

So, Boundary value testing is one of the most popular test case design methodologies for Black-Box testing.

Based on the research in the field of software testing, it was found that 30-35% of software

errors occur near the extreme values or boundary values of the test input data. Therefore, the boundary value testing technique is mainly used to identify the software error at the boundaries of the input values.

Note: This is one of the popular Avionics Verification and Validation interview questions.

Normal Boundary Value Testing Technique:

If, A<= x1 <=B, where x1 is the input and A is the minimum value for x1 and B is the maximum value for x1. We can consider the following test cases for **Normal Boundary Value Testing**:

1. x1 = Minimum value of input domain (x1 = A)
2. x1 = Minimum+1 (x1 = A+1)
3. x1 = Normal value or Mid value from the input domain (x1 = (A+B)/2)
4. x1 = Maximum-1 (x1 = B-1)
5. x1 = Maximum value of input domain (x1 = B)

Robust Boundary Value Testing:

Let's consider the same example, A<= x1 <=B, where x1 is the input and A is the minimum value for x1 and B is the maximum value for x1. We can consider the following test cases for **Robust Boundary Value Testing**:

1. x1 = Minimum-1 (x1 = A-1)
2. x1 = Minimum value of input domain (x1 = A)
3. x1 = Minimum+1 (x1 = A+1)
4. x1 = Normal value or Mid value from the input domain (x1 = (A+B)/2)
5. x1 = Maximum-1 (x1 = B-1)
6. x1 = Maximum value of input domain (x1 = B)
7. x1 = Maximum+1 (x1 = B+1)

Let's look at an example for a Boundary Value Analysis technique:

For example, Air_Speed parameter for an avionics application can hold the following functional range values 0 Knot to 1500 Knot.

As we can see clearly in the above image, the valid Air_Speed range is 0 Knot to 1500 knot. Any value less than 0 or any value is greater than 1500 are considered Invalid Air_Speed value.

Therefore in our case, the Minimum value is = 0 and Maximum value is = 1500. Here are the Normal Boundary value test cases for Air_Speed:

Input Air_Speed	**Test Case Description**
0	Minimum value
1	Minimum+1
750	Nominal value or Mid value
1499	Maximum-1
1500	Maximum value

Here are the Robust Boundary value test cases for Air_Speed:

Input Air_Speed	Test Case Description
-1	Minimum-1
0	Minimum
1	Minimum+1
750	Nominal value or Mid value
1499	Maximum-1
1500	Maximum
1501	Maximum+1

What is Equivalence Class Partitioning?

Answer:

The Equivalence Class Partitioning is another popular Black-Box testing technique used for writing functional test cases. The Equivalence Class Partitioning technique can be applied to Unit Level, Integration or System testing.

In the Equivalence Class Partitioning technique, the whole input domain is divided into multiple equivalent partitions. The equivalence partitioning is done in such a way that the software behaves the same/similar ways for each input value belonging to a particular equivalence Class.

Therefore, we can pick one representative value from each equivalence class. This technique is mainly used to avoid a huge set of test cases from the same equivalence class. So, overall, this technique is used to reduce the number of test cases and make the testing more effective.

From the above, image, we can clearly see three equivalence classes:

1. Valid Air_Speed input domain
2. Invalid Positive Air_Speed input domain
3. Invalid Negative Air-Speed input domain

How can you determine Equivalence Classes?

The following steps can be followed to determine the equivalence classes:

1. Examine the Input data domain
2. Identify the Valid Positive Input domain
3. Identify the Valid Negative Input domain
4. Identify the Invalid Positive Input domain
5. Identify the Invalid Negative Input domain

What is Structural Testing?

Answer:

Structural testing is nothing but the White Box testing. We have explained the White-Box testing in the previous question.

Conclusion:

I have tried to cover most of the popular avionics verification and validation interview questions in the context of DO-178C / DO178C / DO-178B / DO178B, that you can face during the interview.

However, if I missed any topic or area in the context of avionics verification and validation interview questions (DO-178C / DO178C / DO-178B / DO178B), please feel free to comment below.

You can also contact me directly: admin {at } TheCloudStrap [dot] com

Hopefully, this article could help you!

Good luck with your interview.

Admin

This post was published by Admin.

Email: admin@TheCloudStrap.Com

Related Posts:

1. Avionics Verification and Validation Interview Questions
2. Top 45 C++ Interview Questions For Freshers and Experienced Professionals
3. 50+ DO178C Interview Questions | DO178C Standards
4. SOI Audit Interview Questions
5. 28 MCDC Interview Questions
6. DO-254 Interview Questions
7. DO-178C PSAC

8. **Demystifying DO-178C: A Comprehensive Guide to Software Considerations in Airborne Systems Certification**
9. **DO-178C Objectives List | Must Read**
10. **Design Assurance Level (DAL) and Software Level in DO-178C: A Deep Dive with Examples**