Os testes de integração são uma prática no desenvolvimento de software que visa verificar se os diferentes componentes ou módulos de um sistema funcionam corretamente quando combinados. Esses testes são realizados para identificar falhas na interação entre os diferentes elementos do software e garantir que eles operem de maneira integrada.Resumidamente, os testes de integração funcionam da seguinte maneira:

- **Identificação dos Componentes:** Os diferentes módulos ou componentes do software são identificados. Isso pode incluir classes, funções, serviços ou qualquer unidade lógica do sistema.
- **Desenvolvimento de Casos de Teste:** São criados casos de teste que exercitam a interação entre os componentes. Esses casos de teste podem incluir chamadas a funções, passagem de dados entre módulos e simulação de cenários de uso real.
- **Execução dos Testes:** Os casos de teste são executados para verificar se os componentes se comunicam e colaboram corretamente. Isso pode envolver a verificação de entradas e saídas, o fluxo de dados e a correta execução de funcionalidades combinadas.
- **Detecção de Problemas de Integração:** Se houver problemas na interação entre os componentes, os testes de integração ajudarão a identificá-los. Isso pode incluir inconsistências nos dados, falhas de comunicação ou comportamentos inesperados.
- **Correção e Reteste:** Após a identificação de problemas, os desenvolvedores corrigem as falhas e os testes de integração são executados novamente para garantir que as correções foram bem-sucedidas e que não surgiram novos problemas.

Os testes de integração são cruciais para garantir que o software funcione como um todo coeso, mesmo quando composto por partes distintas desenvolvidas por diferentes equipes ou em momentos distintos. Eles contribuem para a estabilidade e confiabilidade do sistema, proporcionando uma abordagem eficaz para identificar e resolver problemas de integração.Tipos de abordagens:

Testes Bottom-Up

Nele, integra-se componentes de infraestrutura e depois adiciona-se componentes funcionais. Nesta abordagem, a integração dos sistemas inicia a partir do nível mais baixo do software, ou seja, o módulo. No entanto, para integrar esse conjunto de módulos é preciso criar um driver, programa de controle que coordena a entrada e saída para cada cluster (módulos que executam uma sub-função do sistema).Entre as vantagens de realizar essa

técnica, estão a verificação antecipada do comportamento de baixo nível; stubs não são necessários, é mais simples para formular dados de entrada para algumas sub-árvores e, também, mais fácil de compreender dados de saídas.

Testes Top-Down

Já na abordagem Top-Down, inicia-se a integração pelo primeiro módulo até o último da hierarquia (de cima para baixo). Nele, desenvolve-se o esqueleto do sistema preenchendo com componentes. Esse teste pode ser feito de duas formas: por profundidade ou largura.Dependendo da abordagem escolhida, os pseudocontroladores são substituídos, um por vez, pelos seus componentes reais. Os testes são conduzidos à medida que cada componente é integrado e, ao final de cada conjunto de testes, outro pseucontrolador é substituído por seu componente real. Nesta abordagem, a vantagem é poder testar logo no início as funções principais do software.

Testes Sandwich

Nesta abordagem o sistema é integrado com um mix da técnica Top-down e Bottom-up, dividindo-o em três camadas: lógica – camada formada pelos módulos que são mais frequentemente chamados. Aqui, utiliza-se a técnica Bottom-up; meio (middle) – são os módulos restantes; e o operacional, que é formado pelos módulos principais, do ponto de vista operacional, sendo testado por meio da técnica Top-down.

Testes Big Bang

Ao usar esta técnica, os módulos são testados separadamente e depois integrados de uma só vez. No entanto, para realizar a integração usando o teste Big-bang é preciso stubs e drivers para testar os módulos isoladamente. Muito usados para demonstrar uma operabilidade mínima do sistema, a maior desvantagem dessa abordagem é que caso ocorra algum erro na interface de um módulo com outro, o motivo ficará mais difícil de ser encontrado, já que é uma abordagem não incremental.Na abordagem incremental, os testes são feitos por etapas nas classes. As classes são testadas uma por uma entre si, facilitando encontrar possíveis erros isoladamente, ao contrário do Big Bang.