Teste de Integração, Sistema e Aceitação

Teste de integração

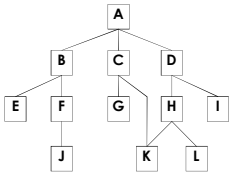
- Unidades ou aplicações que foram testadas em separado são testadas de forma integrada
- A interface entre as unidades integradas é testada
- O teste de integração deve ser feito de forma incremental, ou seja, as unidades devem ser integradas em pequenos segmentos
- Este teste é executado por um testador de integração (geralmente um programador)

Stubs e Drivers

- No contexto de teste de integração, usamos os elementos stubs e drivers
- Stubs são pseudo-implementações de determinadas especificações (Casos básicos/triviais/esperados)
- Drivers são operações que automatizam testes de acordo com casos de teste

Teste de Integração

- Suponha a integração de um grupo de módulos para formar um componente
  - A estrutura de controle forma uma ‘‘hierarquia de chamadas’’ como segue



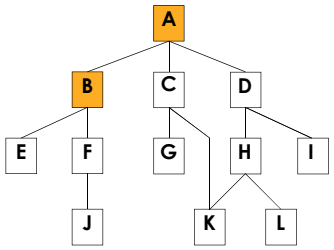
Teste de integração

- Para simplificar a localização de erros, os sistemas devem ser integrados incrementalmente.
- A integração dos módulos pode ser feita através das abordagens
  - Top-down ou
  - Bottom-up

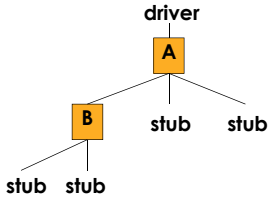
Abordagem Top-down

- Os módulos são integrados de cima para baixo. O teste usa driver e stubs
  - O driver é utilizado como módulo de controle principal, e os módulos reais são substituídos por stubs
  - À medida que os testes vão sendo realizados os stubs são substituídos pelos módulos reais, um de cada vez

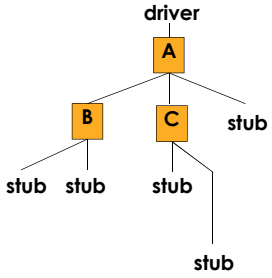
Top-down



Top-down

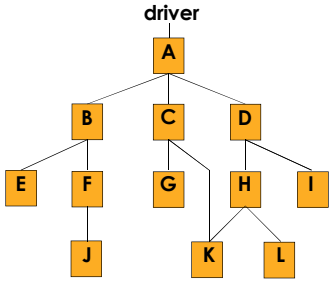


Top-down



E assim por diante. Até que...

Top-down



Top-down: vantagens

- Permite verificação antecipada de comportamento de alto nível
- Um único driver é necessário
- Módulos podem ser adicionados, um por vez, em cada passo, se desejado
- Suporta as abordagens ‘‘breadth first’’ e ‘‘depth first’’

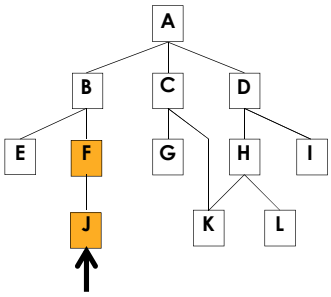
Top-down: desvantagens

- Retarda verificação de comportamento de baixo nível
- Stubs são necessários para suprir elementos ainda inexistentes

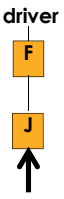
Abordagem Bottom-up

- A integração é feita a partir do nível mais básico da hierarquia. Os stubs nem sempre são necessários
  - Os módulos do nível inferior são combinados.
  - Para cada combinação é criado um driver que coordena a entrada e a saída dos casos de teste.
- O módulo é testado
- O *driver* é substituído pela combinação de módulos correspondentes, que passam a interagir com os módulos do nível superior

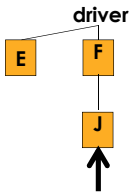
Bottom-up



Bottom-up

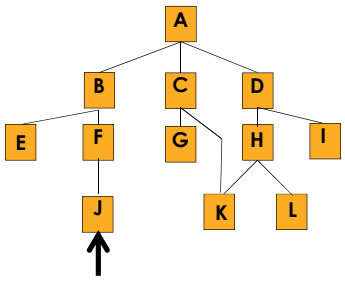


Bottom-up



E assim por diante. Até que...

Bottom-up



Bottom-up: vantagens

- Permite verificação antecipada de comportamento de baixo nível
- Stubs nem sempre são necessários

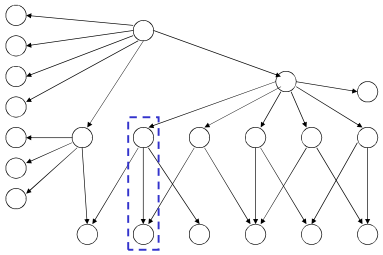
Bottom-up: desvantagens

- Retarda verificação de comportamento de alto nível
- Drivers são necessários para elementos ainda não implementados
- Como sub-árvores são combinadas, um grande número de elementos deve ser integrado de uma só vez

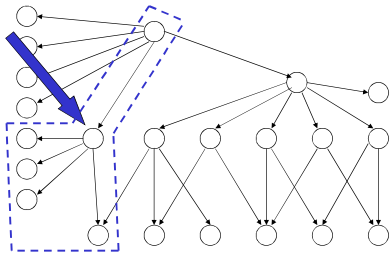
Teste baseado em Chamadas

- Os testes top-down e bottom-up são puramente funcionais
- Usando abordagem estrutural podemos identificar dependências entre unidades
- Duas técnicas:
  - Por pares
  - Por vizinhança
- Obtém-se melhoria ao reduzir stubs/drivers

Teste por Pares



Teste por Vizinhança



Teste de sistema

- Investiga o funcionamento da aplicação como um todo
- Integração dos componentes de software com o ambiente operacional (similar ao de produção) é realizada e testada
- Geralmente emprega teste funcional (Ideal: executado por membro de um grupo independente de testes)
- Pode usar o diagrama de casos de uso como fonte de funcionalidades
- Pode ser guiado pelos fluxos dos casos de uso

Teste de aceitação

- Testes funcionais, realizados pelo usuário, objetivando demonstrar a conformidade com os requisitos do software
- Envolve treinamento, documentação e empacotamento
- Podem ser de duas categorias:
  - Alfa
    - Feitos por usuários, geralmente nas instalações do desenvolvedor. Observam e registram/problemas
  - Beta
    - Feitos por usuários, geralmente em suas próprias instalações, sem supervisão do desenvolvedor. Problemas detectados são então relatados ao desenvolvedor

Exemplos de Casos de Testes

Caso de teste:Tela de Login

	Casos de Teste	
Login do Usuário	Não informar o login	É exibido o alerta "O campo login deve ser preenchido. Verifique, por favor."
	Não informar senha	É exibido o alerta "O campo de senha deve ser preenchido. Verifique por favor."
	Verificar sintaxe	A sintaxe do texto deve estar correta.
	Verificar as atribuições do usuário (teste de segurança)	No momento em que ocorrer o login, apenas as funcionalidades com permissões atribuídas devem ser exibidas ao usuário.
	Logar com o mesmo usuário em dois browsers	É exibido um alertas notificando a duplicidade, e é notificada por e-mail a ocorrência desta redundância ao usuário.

Caso de teste: Pesquisa por Período

	Casos de Teste	
Pesquisa por período	Informar data início e fim em uma seqüência cronológica	O sistema retorna todos os registros cadastrados entre as datas informadas
	Informar apenas a data de início	O sistema pode exigir ou não a data fim como obrigatório, vai depender do critério do analista do sistema (o testador consulta o analista)
	Informar apenas data fim	O sistema pode exigir ou não a data início como obrigatório, vai depender do critério do analista do sistema (o testador consulta o analista)
	Não informar nenhum dos campos	É exibido o alerta para que o usuário preencha o(s) campo(s) obrigatório(s) para a consulta (a obrigatoriedade, mais uma vez, depende do analista).
	Informar a data fim anterior à data início	É exibido o alerta "A data fim informada é anterior à data início. Verifique, por falor."