Testes Automatizados com Banco de Dados

Helves Domingues e Paulo Cheque

12/02/2009 Verão2009

Por quê testar BDs?

- Dados => $
- Lógica
  - Stored Procedures
  - Triggers
- SQL
- Design => Manutenção e Evolução
- Mapeamento ORM
- Segurança

Por quê testar BDs?

- O que é banco de dados evolutivos?
  - Modelagem de dados evolutiva
  - Testes do banco de dados
  - Gerência de configuração (SVN, CVS, GIT, etc)
  - Refatoração do banco de dados

Preocupações

- Compartilhamento da base
- Permissões: Schemas, CRUD
- Execução concorrente do teste
- Muitos sistemas utilizando a mesma base
- Sistemas legados
- Conexão – Rede - Distribuição: Desempenho
- Replicação
- Auditoria

Ambiente de Teste

- Bancos de Dados Compartilhados
- Bancos de Dados Locais
- Bancos de Dados em Memória

Bancos de Dados Compartilhados

- Vantagens:
  - Dados reais de usuários
  - Ambiente mais próximo do real
  - Dados já existentes: Diversas situações prontas
- Desvantagens:
  - Ambiente não controlado para testes
  - Problemas com permissões
  - Preocupação com dados existentes
  - Preocupação com dados criados pelos testes
  - Lentidão

Bancos de Dados Locais

- Vantagens:
  - Ambiente controlado para testes
  - Melhor desempenho
- Desvantagens:
  - Custo de preparação dos dados
  - Querys adicionais
  - Dados criados por desenvolvedores, não usuários
  - Testes de aceitação?
  - Preocupação com dados criados pelos testes
  - Licenças

Bancos de Dados em Memória

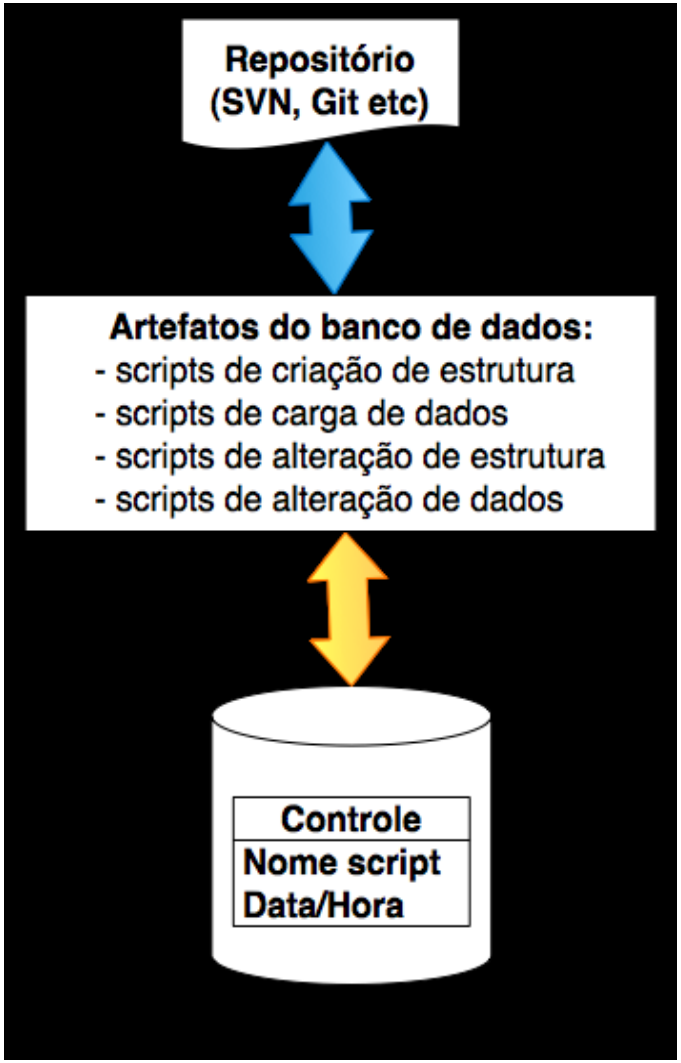
- Vantagens:
  - Desempenho
  - Ambiente controlado para testes
- Desvantagens:
  - Ambiente diferente da realidade
  - Custo de preparação dos dados
  - Querys adicionais
  - Dados criados por desenvolvedores, não usuários
  - Testes de aceitação?
  - Limitações: Stored Procedures, Triggers...

Perguntas

- É possível escolher o ambiente de testes?
- O que é necessário testar?
- Onde estão os erros?
- Qual o custo das desvantagens e o benefício das vantagens?

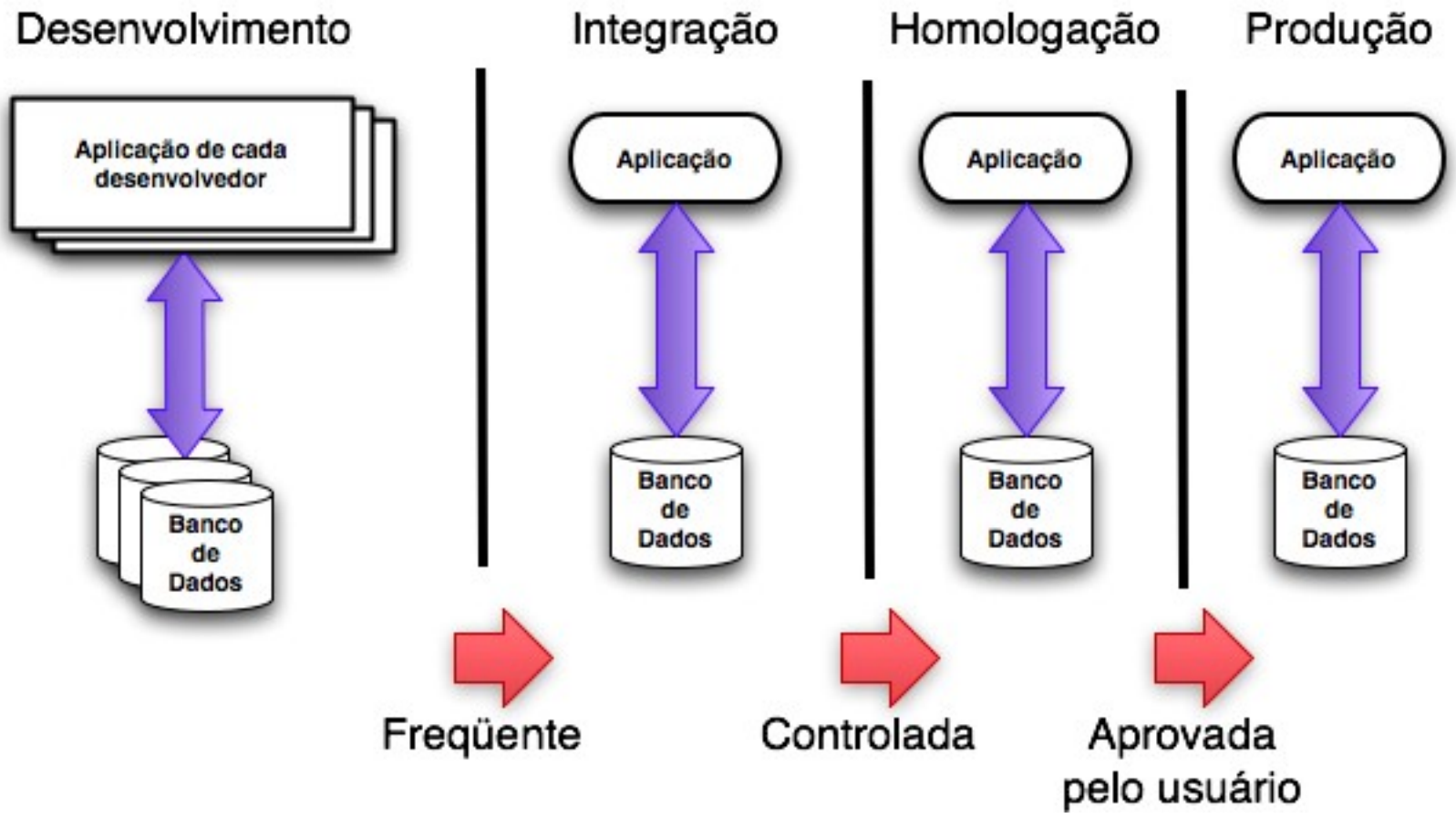
Estratégias

- Ambiente do Desenvolvedor:
  - Controlador de versões
  - Scripts
  - Banco de Dados individual



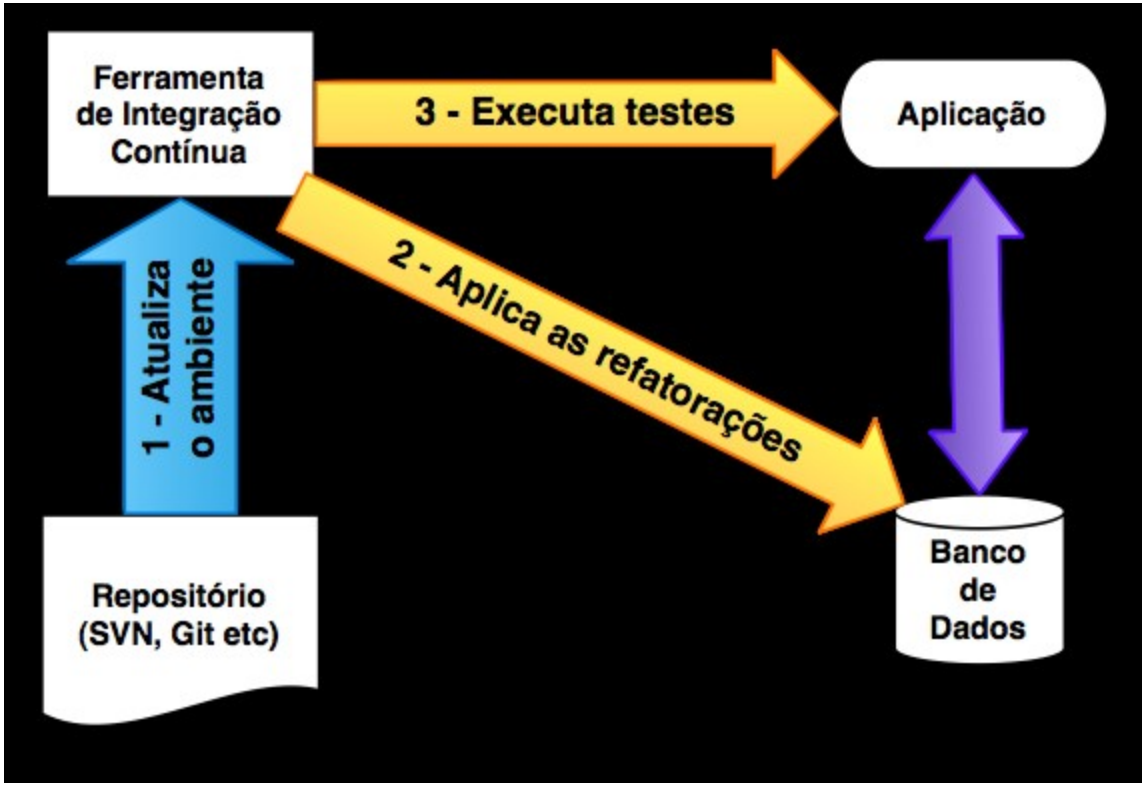
Estratégias

- Ambientes, bancos de dados e transições



Estratégias

- Transição para o ambiente de Integração



Padrões

- Organização:
  - Testcase Superclass
    - Ex: GrailsUnitTestCase
  - Test Utility Method
  - Test Helper

```
@After
public static void removeTodosRegistrosDoBanco() {
    truncateTabelas();
}
```

- Database Sandbox: 1 por desenvolvedor
- Transaction Rollback Teardown
- Table Truncation Teardown

+ Padrões

- Generated Values
  - IDs
  - Timestamp
  - Aleatórios

```
new Pessoa(IDGenerator.uniqueID()).save()

CreditCardGenerator.generate(CreditCardFactory.getMasterCard())
```

Anti-padrões

- Testes frágeis
- Dados/Ambientes não controlados
- IDs fixos
- Lentos
- Sujeira na base de dados

Anti-padrões

```
3  class VolumeIntegrationTests extends grails.test.GrailsUnitTestCase {
4
5      void testToString() {
6          def volume = new Volume(largura: 1.44, comprimento: 2.45, altura: 3.46)
7          assertEquals("1.44m x 2.45m x 3.46m", volume.toString())
8          volume.save(flush: true)
9          assertEquals("1.44m x 2.45m x 3.46m", Volume.get(1).toString())
10         assertEquals("1.44m x 2.45m x 3.46m", Volume.findAll().get(0).toString())
11     }
12
13     void testAceitaNoMaximo2CasasDecimal() {
14         def volume = new Volume(largura: 1.441, comprimento: 2.445, altura: 3.449)
15         volume.save(flush: true)
16         assertEquals("1.44m x 2.45m x 3.45m", Volume.get(2).toString())
17         assertEquals("1.44m x 2.45m x 3.45m", Volume.findAll().get(0).toString())
18     }
```

Sistema Sob Teste

- ... Database Connectivity (DBC)
  - ODBC, JDBC ...
- Frameworks para queries
  - Spring-JDBC ...
- Object-Relational Mapping
  - Hibernate
  - ActiveRecord
- Stored Procedures - Triggers

DBC

- Erros de strings
  - Conversão
  - Valores nulos
- Pouco encapsulamento
  - Código repetido
- Conexão
- Transações

```java
public class UsuarioDAO {
  JdbcTemplate jdbc;

  public UsuarioDAO(JdbcTemplate jdbc) {
    this.jdbc = jdbc;
  }

  public void insereUsuario(String nome, String senha) {
    int id = jdbc.queryForInt("select max(id) from usuario");
    id += 1;
    Object[] args = new Object[] {id, nome, senha};
    jdbc.update("insert into usuario (id, nome, senha) values (?, ?, ?)", args);
  }

  public void atualizaUsuario(Integer id, String nome, String senha) {
    jdbc.update("update usuario set nome = ?, senha = ? where id = ?", new Object[] {nome, senha, id});
  }

  public void removeUsuario(String nome) {
    jdbc.update("delete from usuario where nome = ?", new Object[]{nome});
  }

  @SuppressWarnings("unchecked")
  public List<Map<String, Object>> buscaPorNome(String nome) {
    List<Map<String, Object>> map;
    map = jdbc.queryForList("select * from usuario where nome like ?", new Object[]{nome.replaceAll("[*]", "%")});
    return map;
  }
}
```

```java
@Test
public void testaCRUD() {
  // INSERT
  dao.insereUsuario("Fulano", "123456");
  // VERIFICO
  assertEquals(1, dao.buscaPorNome("Fulano").size());
  assertEquals("123456", dao.buscaPorNome("Fulano").get(0).get("SENHA"));

  // UPDATE
  dao.atualizaUsuario(1, "Ciclano", "123457");
  // VERIFICO
  assertEquals(1, dao.buscaPorNome("Ciclano").size());
  assertEquals("123457", dao.buscaPorNome("Ciclano").get(0).get("SENHA"));

   // DELETE
  dao.removeUsuario("Fulano");
  assertEquals(0, dao.buscaPorNome("Fulano").size());
}

@Test
public void testaBuscaPorNome() {
  dao.insereUsuario("Fulano", "123456");
  dao.insereUsuario("Fulano Tal", "123456");
  dao.insereUsuario("Ciclano Tal", "123456");

  // Ambiente controlado!
  assertEquals(1, dao.buscaPorNome("Fulano").size());
  assertEquals(2, dao.buscaPorNome("Fulano*").size());
  assertEquals(2, dao.buscaPorNome("Fulano%").size());
  assertEquals(0, dao.buscaPorNome(" Fulano").size());
  assertEquals(3, dao.buscaPorNome("*lano*").size());
}
```

Object-Relational Mapping (ORM)

- Tipos
- Validações
  - Min, max, range, null, vazios, padrão, precisão
- Conversões: Tipos primivos ou objetos
- Metadados: XML, Anotações, Convenções ...

```groovy
class Endereco {
    String cep

    static constraints = {
        cep(size: 9..9, blank: false, unique: false, matches: "[0-9]{5}-[0-9]{3}")
    }
```

```groovy
class EnderecoIntegrationTests extends grails.test.GrailsUnitTestCase {

    private Endereco criaEnderecoComCEP(String cep) {
        def endereco = new Endereco(
            cep: cep,
            rua: "Av. Paulista",
            numero: 1,
            complemento: "Bloco B / Sala 3",
            bairro: "Vergueiro",
            cidade: "São Paulo",
            estado: "São Paulo",
            pais: "Brasil"
        )
        endereco
    }

    void testCEPValido() {
        assertTrue(criaEnderecoComCEP("12345-123").validate())
    }

    void testCEPInvalido() {
        assertFalse(criaEnderecoComCEP(null).validate())
        assertFalse(criaEnderecoComCEP("").validate())
        assertFalse(criaEnderecoComCEP("12345123").validate())
        assertFalse(criaEnderecoComCEP("1-2345123").validate())
```

```groovy
class Volume { // em metros
    Float largura
    Float comprimento
    Float altura

    static belongsTo = [ Imovel ]

    static constraints = {
        largura(min: 0f, max: 9999f, scale: 2, nullable: false)
        comprimento(min: 0f, max: 9999f, scale: 2, nullable: false)
        altura(min: 1f, max: 9999f, scale: 2, nullable: true)
    }

    String toString() {
        if(altura)
            largura + "m x " + comprimento + "m x " + altura + "m"
        else
            largura + "m x " + comprimento + "m"
    }

    Float area() {
        largura * comprimento
    }

    Float total() {
        if(altura)
            largura * comprimento * altura
        else
            area()
    }
}
```

```groovy
class VolumeIntegrationTests extends grails.test.GrailsUnitTestCase {

    void testToString() {
        def volume = new Volume(largura: 1.44, comprimento: 2.45, altura: 3.46)
        assertEquals("1.44m x 2.45m x 3.46m", volume.toString())
        volume.save(flush: true)

        assertEquals("1.44m x 2.45m x 3.46m", Volume.findAll().get(0).toString())
    }

    void testAceitaNoMaximo2CasasDecimal() {
        def volume = new Volume(largura: 1.441, comprimento: 2.445, altura: 3.449)
        volume.save(flush: true)

        assertEquals("1.44m x 2.45m x 3.45m", Volume.findAll().get(0).toString())
    }
```

ORM: Mapeamentos

- Mapeamentos:
  - 1-1, 1-N, N-N
- Herança:
  - Tabela única
  - Tabela por tipo
- Listas, mapas ...

ORM: Facilidades

- Cascata de Insert, Update e Delete
- Carregamento de objetos:
  - Preguiçosa, ansiosa
- Cache
- Eventos
  - Before/After: insert / update / delete / load

```groovy
void testCascataDeCaracteristicasDeUmImovel() {
    // Prepara
    def imovel = criaImovelValido()
    imovel.addToCaracteristicas(new CaracteristicaDoImovel(caracteristica: mobiliado, quantidade:1))
    assertTrue("Ops, imóvel inválido!", imovel.validate())

    // Executa INSERT
    imovel.save(flush: true)

    // Verifica
    def imovelConsultado = Imovel.findAll().get(0)
    assertNotNull(imovelConsultado)
    assertEquals(1, imovelConsultado.caracteristicas.size())

    // Verifica cascatas
    assertEquals(1, Localizacao.findAll().size())
    assertEquals(1, Endereco.findAll().size())
    assertEquals(1, CaracteristicaDoImovel.findAll().size())

    // Executa DELETE
    imovelConsultado.removeFromCaracteristicas(imovelConsultado.caracteristicas.iterator().next())
    imovel.save(flush: true)

    // Verifica cascatas
    assertEquals(1, Localizacao.findAll().size())
    assertEquals(1, Endereco.findAll().size())
    assertEquals(0, CaracteristicaDoImovel.findAll().size())
    assertEquals(0, imovelConsultado.caracteristicas.size())
}
```

ORM: Frameworks

- Erros / Limitações
- Gramática SQL específica para um BD
- Particularidades dos bancos

- Exemplo Hibernate:

```
@GeneratedValue(strategy = GenerationType.AUTO)
@GeneratedValue(strategy = GenerationType.SEQUENCE)
// Funciona no MySQL?
```

ORM: Problemas

- Reflexão (entidades com construtor básico...)
- Entidades serializáveis
- Metadados corretos?
- Mapeamento bi-direcional sem dono?
- Representa o design do BD?
  - TablePerClass, SingleTable...

Stored Procedures

- Testes internos nos bancos
- Testes no nível da aplicação
  - Stored Procedure Test
  - Facilidade
  - Menor controle
  - Entrada e Saída

Triggers

- Delay
- Replicação
- BD em comum? Múltiplos usuários?
- Múltiplos desenvolvedores?
- Validação
- Valores gerados

ALTER TABLE ENABLE/DISABLE TRIGGER

Desempenho

- Querys:
  - Junções (joins) lentas
  - Consultas pesadas
  - Dezenas de milhões de registros
- Conexão: Pool de conexões, # de threads e Timeout

- Para resolver: Coletar o plano de acesso e realizar melhorias de desempenho.

Segurança

- SQLInjection I e II

```
....'; DROP TABLE Tabela; --
```

- Estourar memória:

```
select * from Tabela where x = ' ' or 1 = 1
```

- Erros nos drivers, bases...

Contato

http://www.agilcoop.org.br

agilcoop@agilcoop.org.br

helves@ime.usp.br

paulocheque@agilcoop.org.br