



Reflexão em Java, Componentes, Frameworks e Além

Prof. Dr. Frank J. Affonso

Outubro de 2017

Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Reflexão em
Java

Componentes
e frameworks

Além

Principais Tópicos

- ▶ **Apresentação Docente**
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Apresentação Docente

- ▶ **Formação Acadêmica:**
 - Bacharel em Ciências da Computação
 - Mestre em Ciências da Computação
 - Doutor em Engenharia Elétrica
 - Pós-doutorado em Ciências da Computação
- ▶ **Experiência Profissional:**
 - Desenvolvimento de Software
 - Docente
- ▶ **UNESP**
 - Desde fevereiro de 2011

Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Engenharia de Software

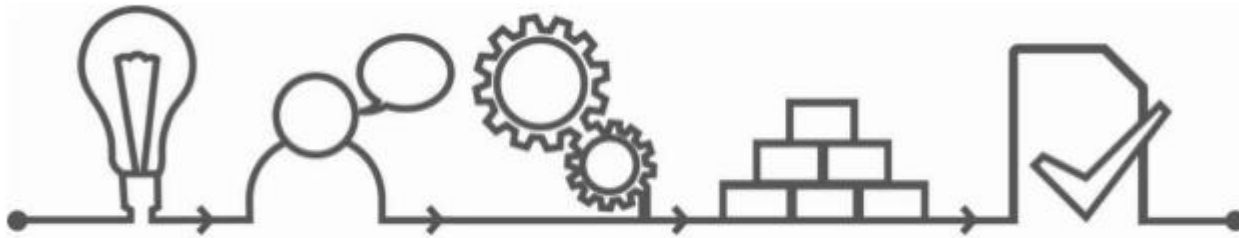
- ▶ O que é uma atividade de engenharia?



Engenharia de Software

- ▶ O que é uma atividade de engenharia?

Engenharia de Software

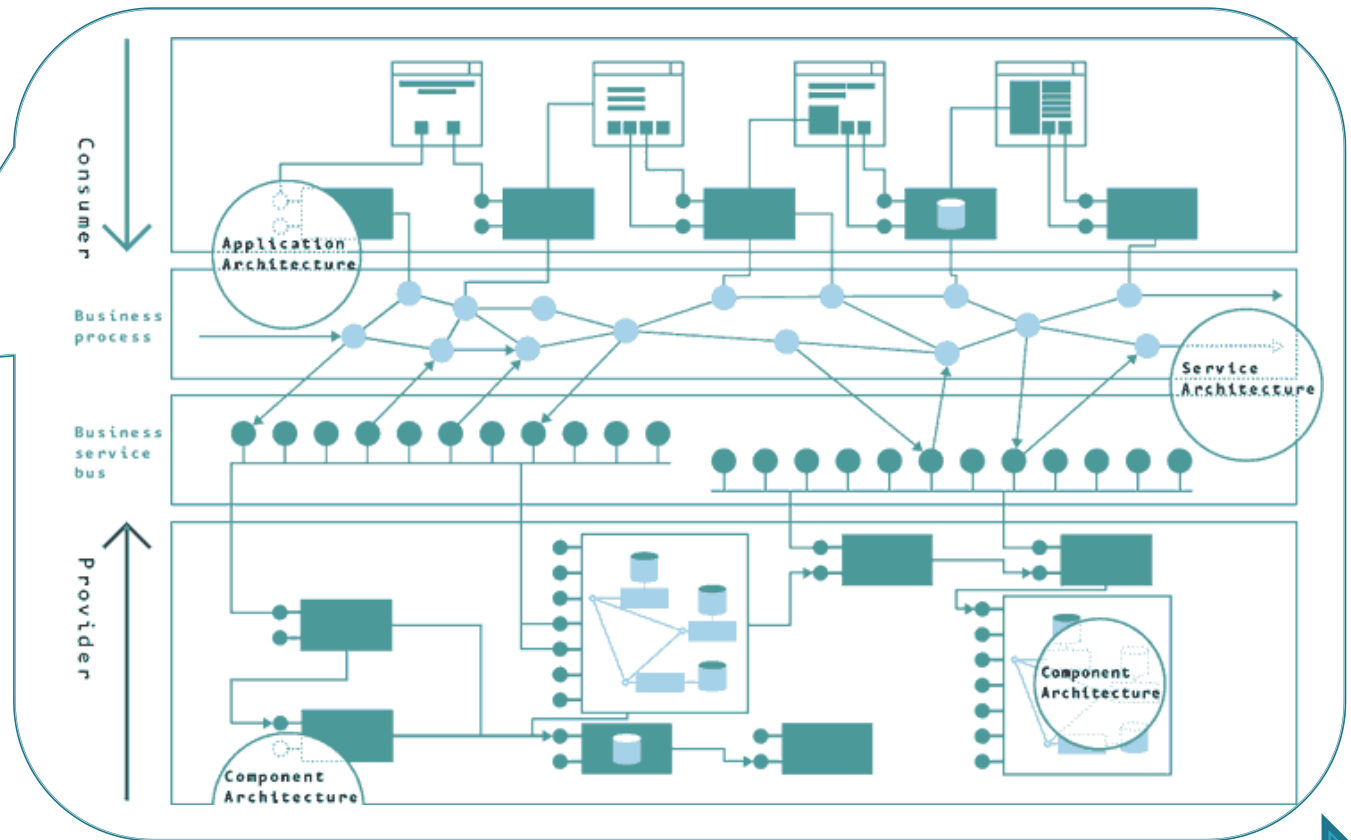
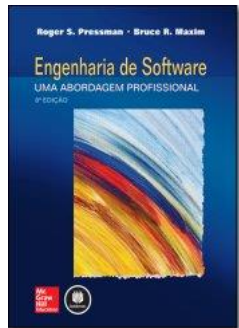


"Engenharia de *Software* é a criação e a utilização de sólidos princípios de engenharia a fim de obter software de maneira econômica, que seja confiável e que trabalhe em máquinas reais"

BAUER, 1960

Engenharia de Software

- O que é uma atividade de engenharia?



Software

Engenharia de Software

► Onde podemos encontrar software?



Engenharia de Software

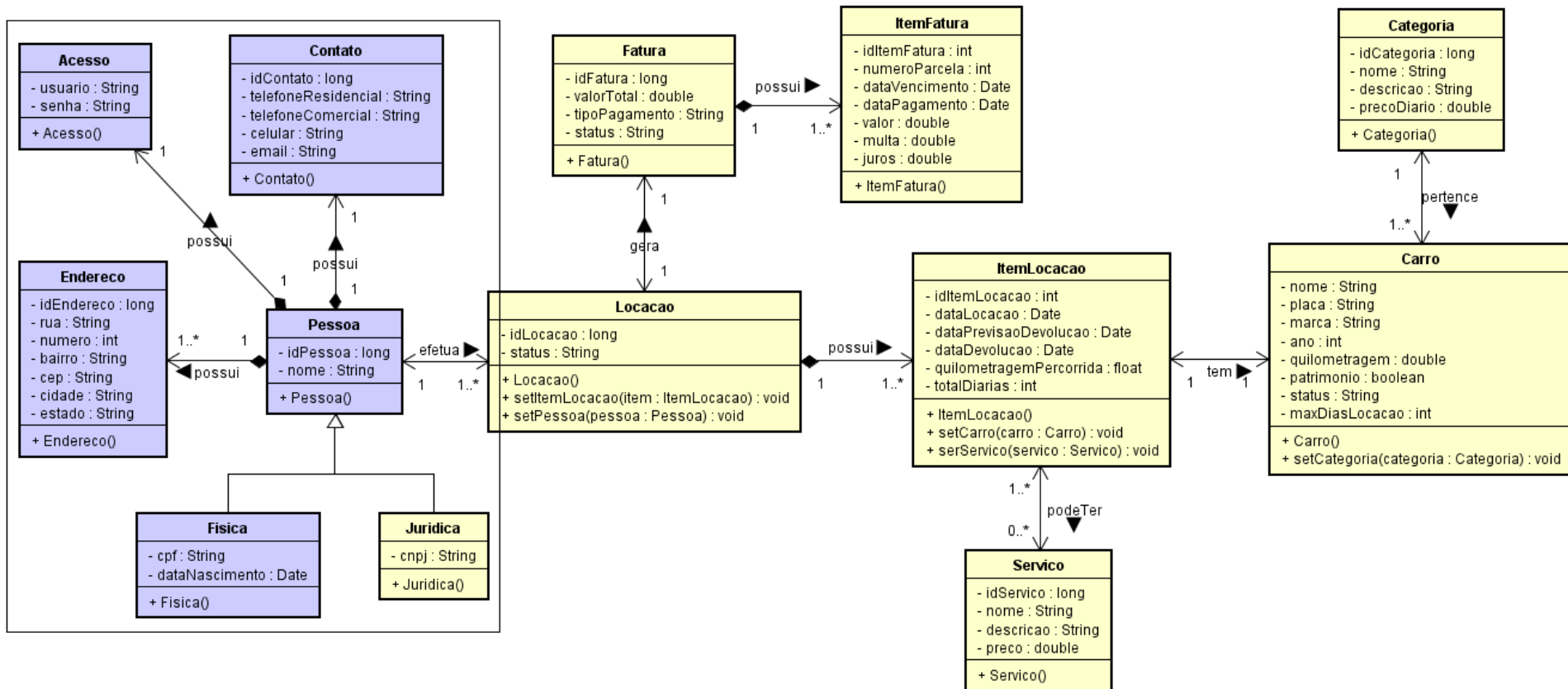
► Locadora de Carros:

- Implementação utilizando a **API JDBC**; e
- Implementação utilizando um **framework de persistência**;



Engenharia de Software

► Modelo UML:



Engenharia de Software

► Locadora de Carros:

- Implementação utilizando a **API JDBC**:
 - Diagramas:
 - UML;
 - MER.
 - Projeto Netbeans.



Engenharia de Software

► Locadora de Carros:

- Implementação utilizando um **framework de persistência**;
- Diagramas:
 - UML.
- Projeto Netbeans.



Engenharia de Software

► Breve comparativo:



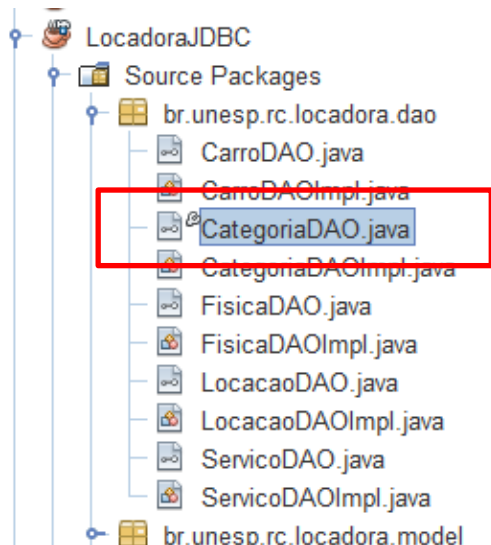
Engenharia de Software

- ▶ **Breve comparativo:**



Engenharia de Software

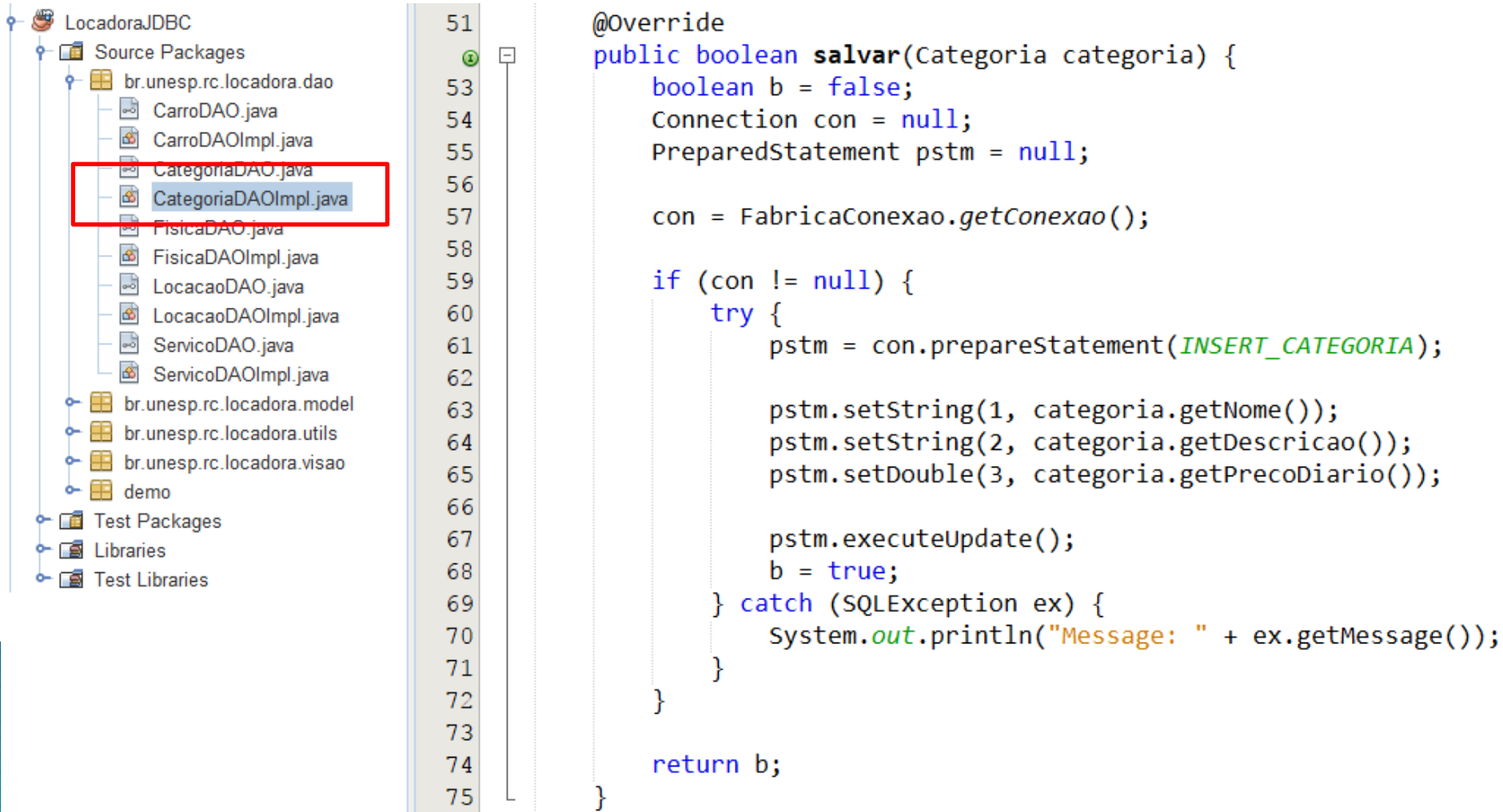
► Breve comparativo:



```
1 package br.unesp.rc.locadora.dao;
2
3 import br.unesp.rc.locadora.model.Categoria;
4
5
6 public interface CategoriaDAO {
7
8     final String INSERT_CATEGORIA = "INSERT INTO Categoria("
9         + "nome, descricao, precoDiario) VALUES(?, ?, ?)";
10
11     public boolean salvar(Categoria categoria);
12
13 }
```

Engenharia de Software

► Breve comparativo:



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure is organized into Source Packages, Test Packages, Libraries, and Test Libraries. The Source Packages section includes the package `br.unesp.rc.locadora.dao`, which contains several DAO classes. The class `CategoriaDAOImpl.java` is highlighted with a red rectangle. The code editor displays the implementation of the `salvar` method in `CategoriaDAOImpl.java`, which is annotated with `@Override`. The method takes a `Categoria` object as a parameter and returns a `boolean`. It initializes a `boolean` variable `b` to `false`, obtains a `Connection` from `FabricaConexao`, and prepares a `PreparedStatement` with the SQL statement `INSERT_CATEGORIA`. It then sets the category's name, description, and daily price to the prepared statement, executes the update, and sets `b` to `true`. If an `SQLException` occurs, it prints an error message. Finally, it returns the value of `b`.

```
LocadoraJDBC
├── Source Packages
│   ├── br.unesp.rc.locadora.dao
│   │   ├── CarroDAO.java
│   │   ├── CarroDAOImpl.java
│   │   ├── CategoriaDAO.java
│   │   └── CategoriaDAOImpl.java
│   ├── FisicaDAO.java
│   ├── FisicaDAOImpl.java
│   ├── LocacaoDAO.java
│   ├── LocacaoDAOImpl.java
│   ├── ServicoDAO.java
│   └── ServicoDAOImpl.java
│   ├── br.unesp.rc.locadora.model
│   ├── br.unesp.rc.locadora.utils
│   ├── br.unesp.rc.locadora.visao
│   └── demo
├── Test Packages
├── Libraries
└── Test Libraries
```

```
51 @Override
52 public boolean salvar(Categoria categoria) {
53     boolean b = false;
54     Connection con = null;
55     PreparedStatement pstmt = null;
56
57     con = FabricaConexao.getConexao();
58
59     if (con != null) {
60         try {
61             pstmt = con.prepareStatement(INSERT_CATEGORIA);
62
63             pstmt.setString(1, categoria.getNome());
64             pstmt.setString(2, categoria.getDescricao());
65             pstmt.setDouble(3, categoria.getPrecoDiario());
66
67             pstmt.executeUpdate();
68             b = true;
69         } catch (SQLException ex) {
70             System.out.println("Message: " + ex.getMessage());
71         }
72     }
73
74     return b;
75 }
```

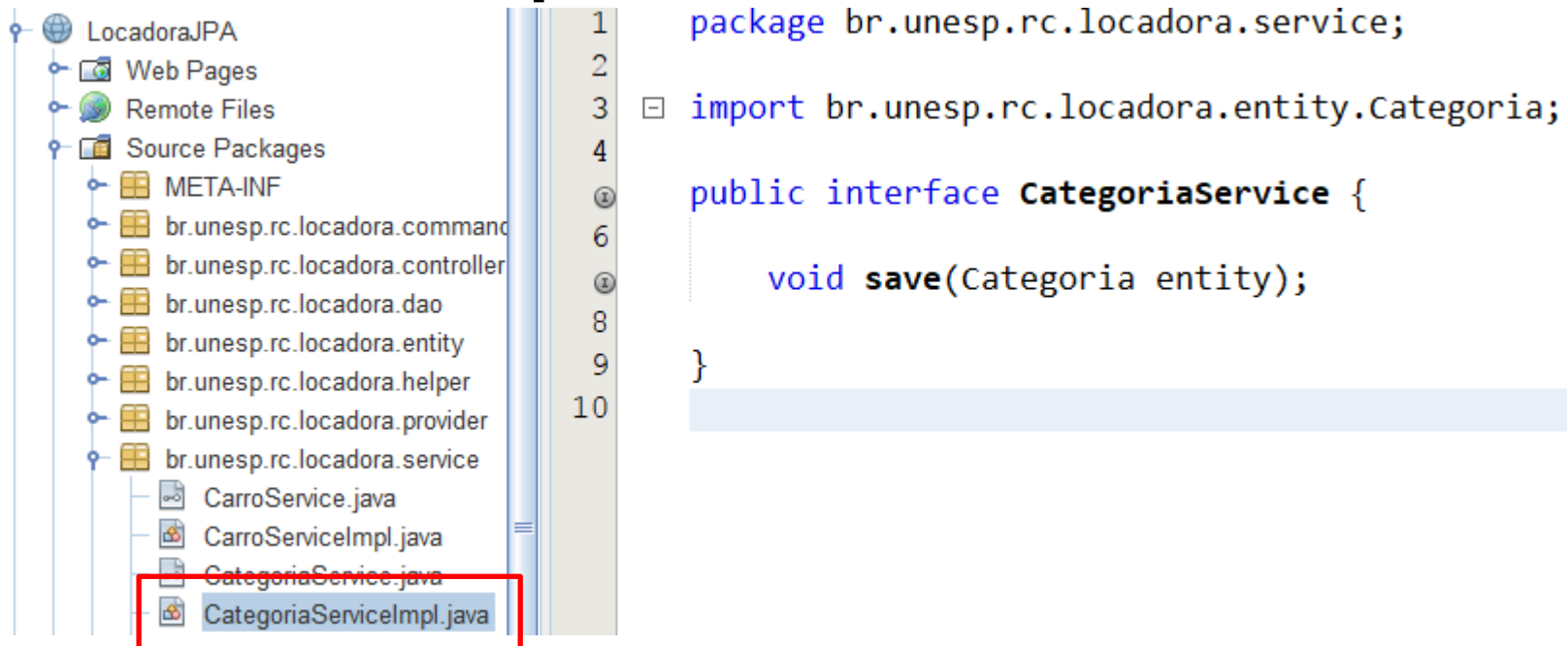

Engenharia de Software

► Breve comparativo:



Engenharia de Software

► Breve comparativo:

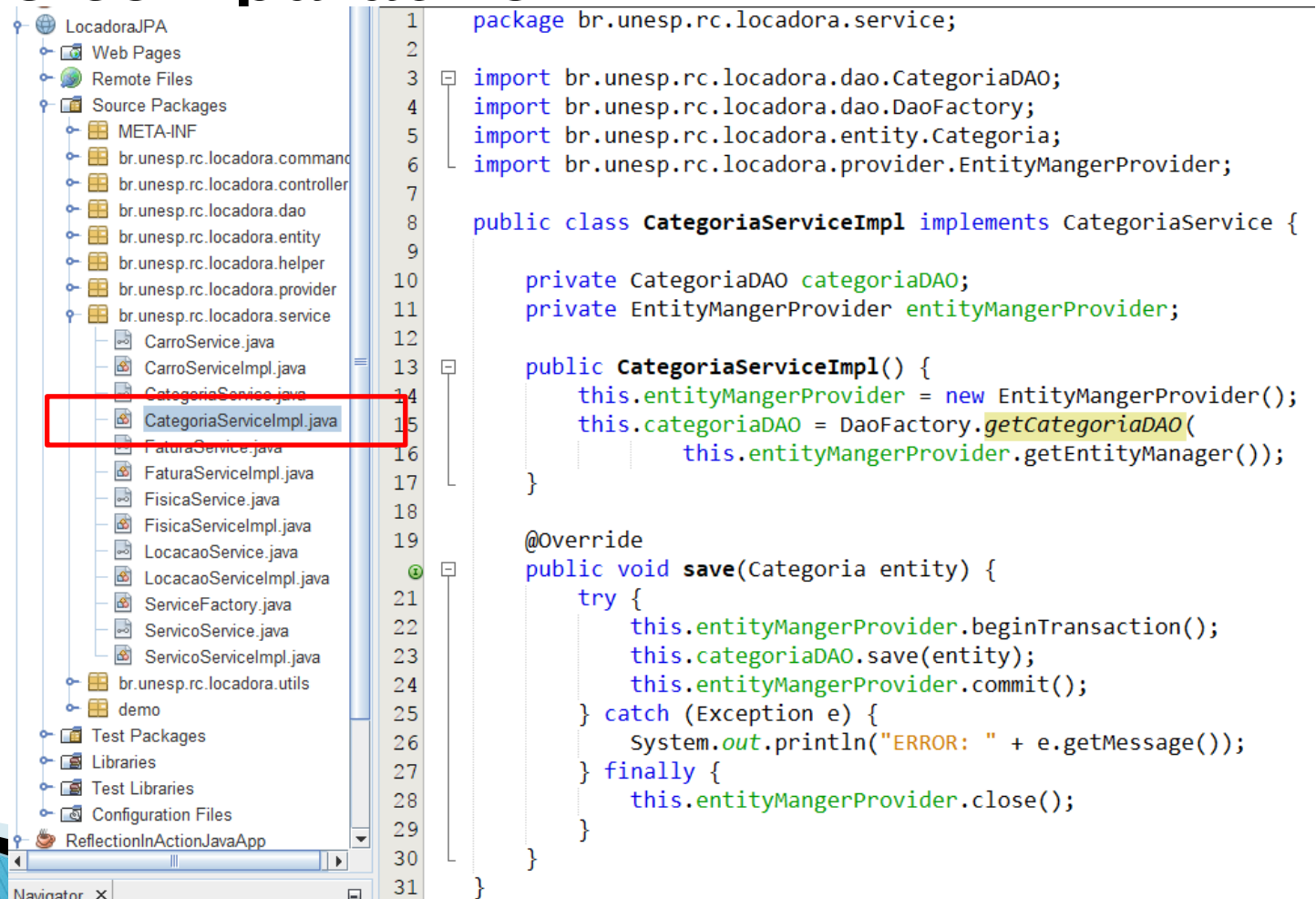


The image shows a screenshot of an IDE. On the left, a project tree for 'LocadoraJPA' is visible. It contains folders for 'Web Pages', 'Remote Files', and 'Source Packages'. Under 'Source Packages', there are several sub-packages: 'META-INF', 'br.unesp.rc.locadora.command', 'br.unesp.rc.locadora.controller', 'br.unesp.rc.locadora.dao', 'br.unesp.rc.locadora.entity', 'br.unesp.rc.locadora.helper', 'br.unesp.rc.locadora.provider', and 'br.unesp.rc.locadora.service'. The 'br.unesp.rc.locadora.service' package is expanded, showing files 'CarroService.java', 'CarroServiceImpl.java', 'CategoriaService.java', and 'CategoriaServiceImpl.java'. The 'CategoriaServiceImpl.java' file is highlighted with a red rectangle. On the right, a code editor shows the content of 'CategoriaServiceImpl.java'. The code is as follows:

```
1 package br.unesp.rc.locadora.service;
2
3 import br.unesp.rc.locadora.entity.Categoria;
4
5 public interface CategoriaService {
6     void save(Categoria entity);
7 }
8
9
10
```

Engenharia de Software

► Breve comparativo:



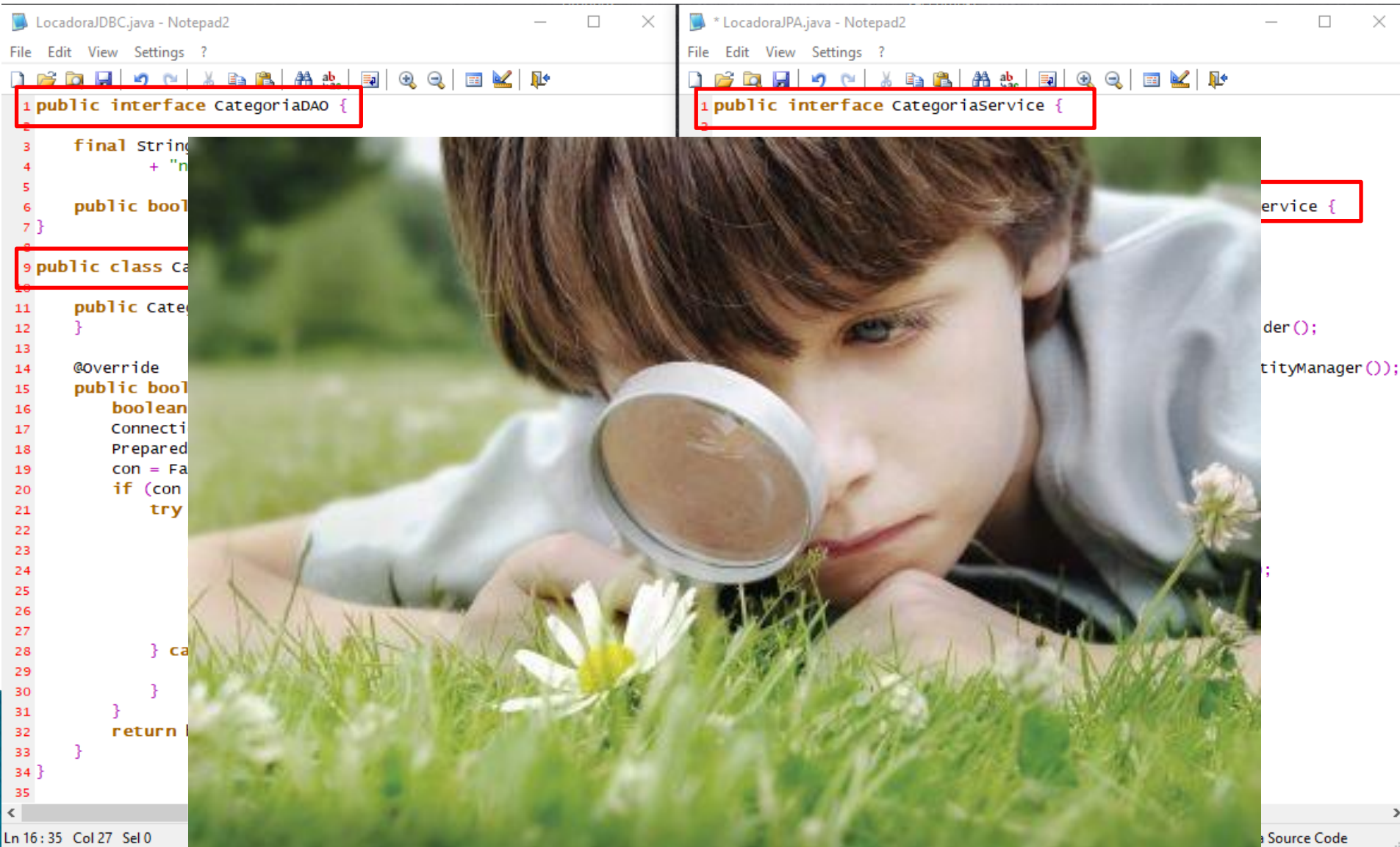
```
1 package br.unesp.rc.locadora.service;
2
3 import br.unesp.rc.locadora.dao.CategoriaDAO;
4 import br.unesp.rc.locadora.dao.DaoFactory;
5 import br.unesp.rc.locadora.entity.Categoria;
6 import br.unesp.rc.locadora.provider.EntityMangerProvider;
7
8 public class CategoriaServiceImpl implements CategoriaService {
9
10     private CategoriaDAO categoriaDAO;
11     private EntityMangerProvider entityMangerProvider;
12
13     public CategoriaServiceImpl() {
14         this.entityMangerProvider = new EntityMangerProvider();
15         this.categoriaDAO = DaoFactory.getCategoriaDAO(
16             this.entityMangerProvider.getEntityManager());
17     }
18
19     @Override
20     public void save(Categoria entity) {
21         try {
22             this.entityMangerProvider.beginTransaction();
23             this.categoriaDAO.save(entity);
24             this.entityMangerProvider.commit();
25         } catch (Exception e) {
26             System.out.println("ERROR: " + e.getMessage());
27         } finally {
28             this.entityMangerProvider.close();
29         }
30     }
31 }
```

Engenharia de Software

```
LocadoraJDBC.java - Notepad2
File Edit View Settings ?
1 public interface CategoriaDAO {
2
3     final String INSERT_CATEGORIA = "INSERT INTO Categoria("
4         + "nome, descricao, precoDiario) VALUES(?, ?, ?)";
5
6     public boolean salvar(Categoria categoria);
7 }
8
9 public class CategoriaDAOImpl implements CategoriaDAO {
10
11     public CategoriaDAOImpl() {
12     }
13
14     @Override
15     public boolean salvar(Categoria categoria) {
16         boolean b = false;
17         Connection con = null;
18         PreparedStatement pstmt = null;
19         con = FabricaConexao.getConexao();
20         if (con != null) {
21             try {
22                 pstmt = con.prepareStatement(INSERT_CATEGORIA);
23                 pstmt.setString(1, categoria.getNome());
24                 pstmt.setString(2, categoria.getDescricao());
25                 pstmt.setDouble(3, categoria.getPrecoDiario());
26                 pstmt.executeUpdate();
27                 b = true;
28             } catch (SQLException ex) {
29                 System.out.println("Message: " + ex.getMessage());
30             }
31         }
32         return b;
33     }
34 }
35
```

```
* LocadoraJPA.java - Notepad2
File Edit View Settings ?
1 public interface CategoriaService {
2
3     void save(Categoria entity);
4 }
5
6 public class CategoriaServiceImpl implements CategoriaService {
7
8     private CategoriaDAO categoriaDAO;
9     private EntityManagerProvider entityManagerProvider;
10
11     public CategoriaServiceImpl() {
12         this.entityManagerProvider = new EntityManagerProvider();
13         this.categoriaDAO = DaoFactory.getCategoriaDAO(
14             this.entityManagerProvider.getEntityManager());
15     }
16
17     @Override
18     public void save(Categoria entity) {
19         try {
20             this.entityManagerProvider.beginTransaction();
21             this.categoriaDAO.save(entity);
22             this.entityManagerProvider.commit();
23         } catch (Exception e) {
24             System.out.println("ERROR: " + e.getMessage());
25         } finally {
26             this.entityManagerProvider.close();
27         }
28     }
29 }
30
```

Engenharia de Software



Engenharia de Software

```
LocadoraJDBC.java - Notepad2
File Edit View Settings ?

1 public interface CategoriaDAO {
2
3     final String INSERT_CATEGORIA = "INSERT INTO Categoria("
4         + "nome, descricao, precoDiario) VALUES(?, ?, ?)";
5
6     public boolean salvar(Categoria categoria);
7 }
8
9 public class CategoriaDAOImpl implements CategoriaDAO {
10
11     public CategoriaDAOImpl() {
12     }
13
14     @Override
15     public boolean salvar(Categoria categoria) {
16         boolean b = false;
17         Connection con = null;
18         PreparedStatement pstmt = null;
19         con = FabricaConexao.getConexao();
20         if (con != null) {
21             try {
22                 pstmt = con.prepareStatement(INSERT_CATEGORIA);
23                 pstmt.setString(1, categoria.getNome());
24                 pstmt.setString(2, categoria.getDescricao());
25                 pstmt.setDouble(3, categoria.getPrecoDiario());
26                 pstmt.executeUpdate();
27                 b = true;
28             } catch (SQLException ex) {
29                 System.out.println("Message: " + ex.getMessage());
30             }
31         }
32         return b;
33     }
34 }
35
```

```
* LocadoraJPA.java - Notepad2
File Edit View Settings ?

1 public interface CategoriaService {
2
3     void save(Categoria entity);
4 }
5
6 public class CategoriaServiceImpl implements CategoriaService {
7
8     private CategoriaDAO categoriaDAO;
9     private EntityManagerProvider entityManagerProvider;
10
11     public CategoriaServiceImpl() {
12         this.entityManagerProvider = new EntityManagerProvider();
13         this.categoriaDAO = DaoFactory.getCategoriaDAO(
14             this.entityManagerProvider.getEntityManager());
15     }
16
17     @Override
18     public void save(Categoria entity) {
19         try {
20             this.entityManagerProvider.beginTransaction();
21             this.categoriaDAO.save(entity);
22             this.entityManagerProvider.commit();
23         } catch (Exception e) {
24             System.out.println("ERROR: " + e.getMessage());
25         } finally {
26             this.entityManagerProvider.close();
27         }
28     }
29 }
30
```

Engenharia de Software

► Breve comparativo:



```

public class FisicaDAOImpl implements FisicaDAO {
    public FisicaDAOImpl() {
    }
    @Override
    public boolean salvar(Fisica fisica) {
        boolean b = false;
        Connection con = null;
        PreparedStatement pstmt = null;
        ResultSet res = null;
        long idPessoa = -1;
        con = FabricaConexao.getConexao();
        if (con != null) {
            try {
                con.setAutoCommit(false);
                // Insert Pessoa
                pstmt = con.prepareStatement(INSERT_PESSOA, PreparedStatement.RETURN_GENERATED_KEYS);
                pstmt.setString(1, fisica.getNome());
                pstmt.executeUpdate();
                res = pstmt.getGeneratedKeys();
                while (res.next()) {
                    idPessoa = res.getLong(1);
                }
                // Insert Fisica
                pstmt = con.prepareStatement(INSERT_FISICA);
                pstmt.setString(1, fisica.getCpf());
                pstmt.setDate(2, new java.sql.Date(fisica.getDataNascimento().getTime()));
                pstmt.setLong(3, idPessoa);
                pstmt.executeUpdate();
                // Insert Contato
                pstmt = con.prepareStatement(INSERT_CONTATO);
                pstmt.setString(1, fisica.getContato().getTelefoneResidencial());
                pstmt.setString(2, fisica.getContato().getTelefoneComercial());
                pstmt.setString(3, fisica.getContato().getCelular());
                pstmt.setString(4, fisica.getContato().getEmail());
                pstmt.setLong(5, idPessoa);
                pstmt.executeUpdate();
                // Insert Acesso
                pstmt = con.prepareStatement(INSERT_ACESSO);
                pstmt.setString(1, fisica.getAcesso().getUsuario());
                pstmt.setString(2, fisica.getAcesso().getSenha());
                pstmt.setLong(3, idPessoa);
                pstmt.executeUpdate();
                // Insert Endereco
                pstmt = con.prepareStatement(INSERT_ENDERECO);
                for (Endereco e : fisica.getEndereco()) {
                    pstmt.setString(1, e.getRua());
                    pstmt.setInt(2, e.getNumero());
                    pstmt.setString(3, e.getBairro());
                    pstmt.setString(4, e.getCidade());
                    pstmt.setString(5, e.getEstado());
                    pstmt.setString(6, e.getCep());
                    pstmt.setLong(7, idPessoa);
                    pstmt.executeUpdate();
                }
                con.commit();
                b = true;
            } catch (SQLException ex) {
                System.out.println("Message: " + ex);
            }
        }
        return b;
    }
}

```

JDBC
Java Database Connectivity

Quanto trabalho !!!



```

public class FisicaServiceImpl implements FisicaService {

    private FisicaDAO fisicaDAO;
    private EntityManagerProvider entityManagerProvider;

    public FisicaServiceImpl() {
        this.entityManagerProvider = new EntityManagerProvider();
        this.fisicaDAO = DaoFactory.getFisicaDAO(this.entityManagerProvider.getEntityManager());
    }

    @Override
    public void save(Fisica entity) {
        try {
            this.entityManagerProvider.beginTransaction();
            this.fisicaDAO.save(entity);
            this.entityManagerProvider.commit();
        } catch (Exception e) {
            System.out.println("ERROR: " + e.getMessage());
        } finally {
            this.entityManagerProvider.close();
        }
    }
}

```

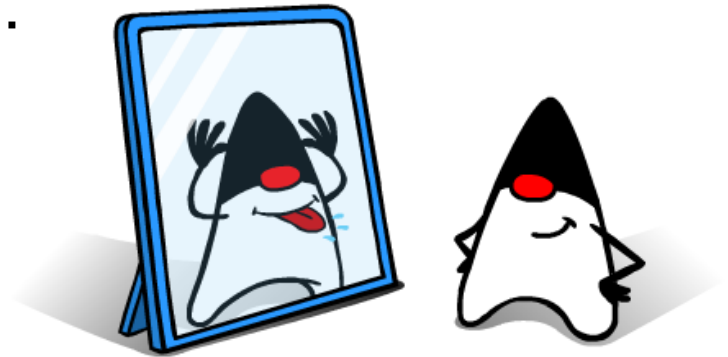
JPA
Java Persistence API
HIBERNATE

Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ **Reflexão em Java**
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Reflexão em Java

- ▶ Assuntos Relacionados:
 - Engenharia de Software
- ▶ Palestra:
 - Reflexão em Java
 - Componentes
 - Frameworks
 - Além



JAVA REFLECTION

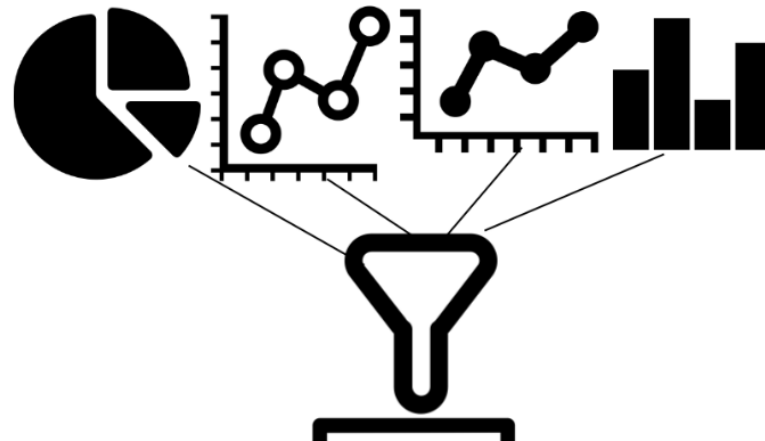
Reflexão em Java

- O que é reflexão?



Reflexão em Java

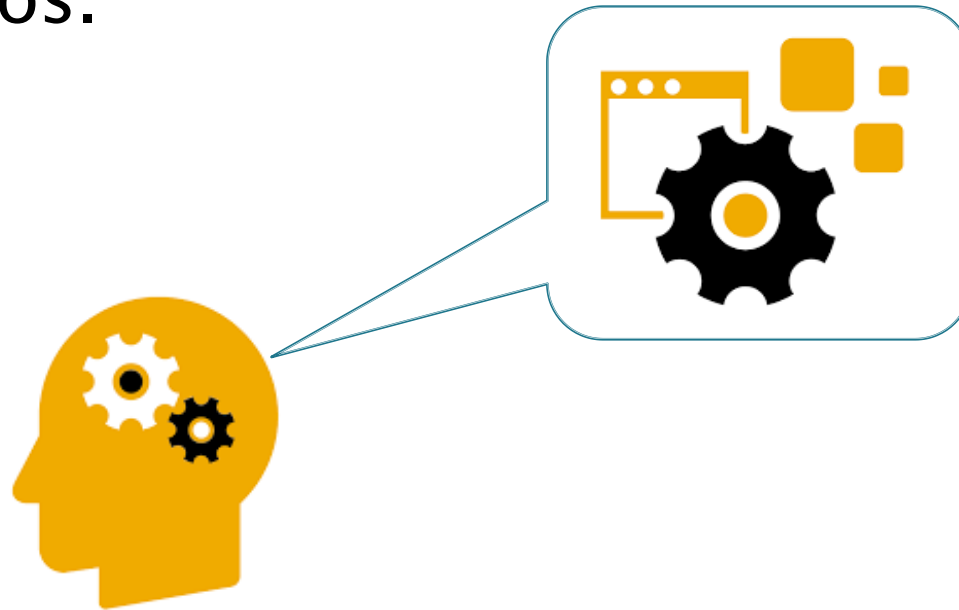
- ▶ A reflexão é o recurso comumente utilizado por programas que requerem **examinar ou modificar a estrutura e/ou comportamento** dos aplicativos que se estão executando na máquina virtual Java em tempo de execução.



<https://docs.oracle.com/javase/tutorial/reflect/>

Reflexão em Java

- ▶ Este é um **recurso relativamente avançado** e deve ser usado apenas por desenvolvedores que tenham uma forte compreensão de seus fundamentos.



<https://docs.oracle.com/javase/tutorial/reflect/>

Reflexão em Java

- ▶ Com essa ressalva em mente, a **reflexão é uma técnica poderosa** e pode permitir que as aplicações executem operações que de outro modo seriam impossíveis.



<https://docs.oracle.com/javase/tutorial/reflect/>

Reflexão em Java

► Examinar o software:

► API Java Reflect:

- Classes
- Atributos
- Métodos
- ...

hands
On



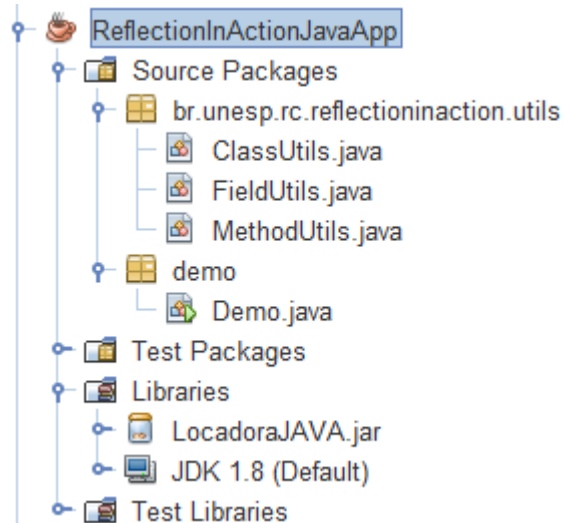
<https://docs.oracle.com/javase/tutorial/reflect/>

Prof. Dr. Frank J. Affonso



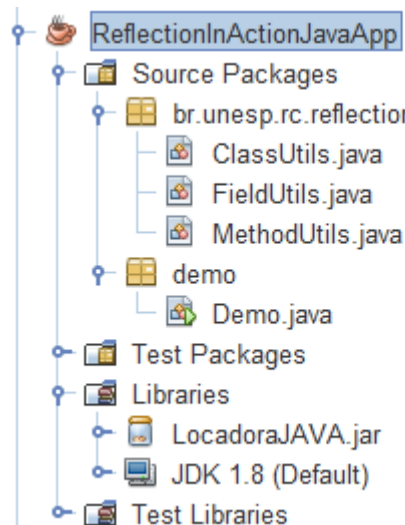
Reflexão em Java

► Projeto:



Reflexão em Java

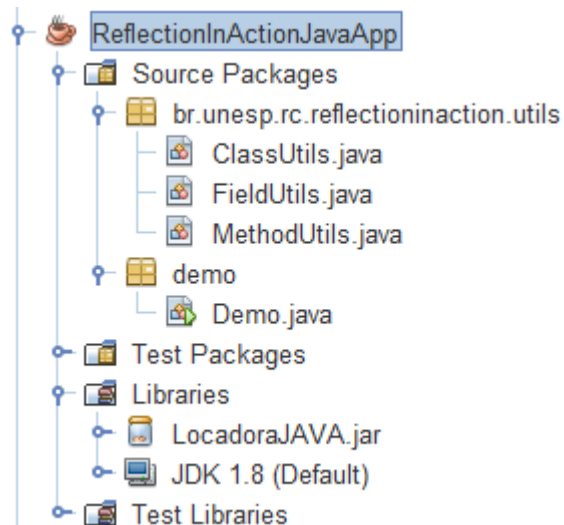
► Projeto:



```
12 public class ClassUtils {
13
14     private ClassUtils() {
15     }
16
17     public static String getName(Object object) {
18         return object.getClass().getName();
19     }
20
21     public static String getPackageName(Object object) {
22         return object.getClass().getPackage().getName();
23     }
24
25     public static String getSuperClassName(Object object) {
26         return object.getClass().getSuperclass().getName();
27     }
28
29 }
30
31 }
```

Reflexão em Java

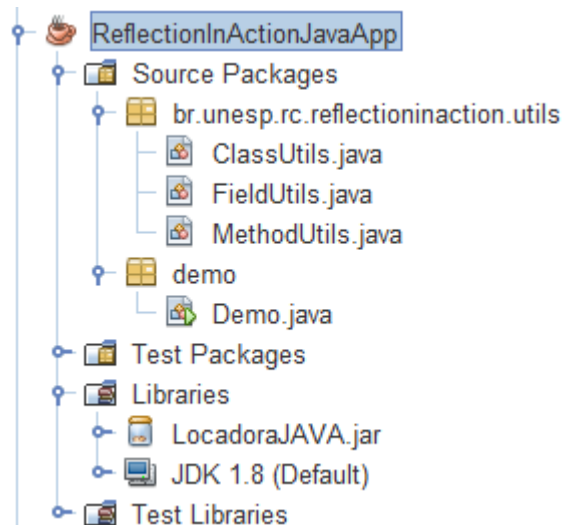
► Projeto:



```
16 public class FieldUtils {
17
18     private FieldUtils() {
19     }
20
21     public static List<String> getFieldNames(Object object) {
22         List<String> names = new ArrayList<>();
23
24         Field[] fields = object.getClass().getDeclaredFields();
25
26         for (int i = 0; i < fields.length; i++){
27             names.add(fields[i].getModifiers() + " - " +
28                     fields[i].getType().getSimpleName() + " - " +
29                     fields[i].getName());
30         }
31
32         return names;
33     }
34 }
```

Reflexão em Java

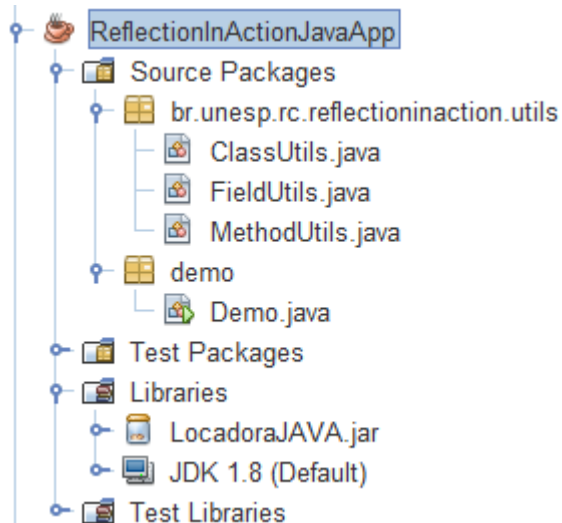
► Projeto:



```
17 public class MethodUtils {
18
19     private MethodUtils() {
20     }
21
22     public static List<String> getMethodNames(Object object){
23         List<String> names = new ArrayList<>();
24         Method[] methods = object.getClass().getDeclaredMethods();
25         for (int i = 0; i < methods.length; i++){
26             names.add(methods[i].getName());
27         }
28         return names;
29     }
30
31     public static List<String> getConstructorNames(Object object){
32         List<String> names = new ArrayList<>();
33         Constructor[] constructors = object.getClass().getConstructors();
34         for (int i = 0; i < constructors.length; i++){
35             names.add(constructors[i].getName());
36         }
37         return names;
38     }
39 }
```

Reflexão em Java

► Projeto:



```
19 public class Demo {
20
21     public static void main(String[] args) {
22         Carro car = new Carro();
23
24         String className = ClassUtils.getName(car);
25         System.out.println("Class name: " + className);
26         System.out.println("-----");
27         String packageName = ClassUtils.getPackageName(car);
28         System.out.println("Package name: " + packageName);
29         System.out.println("-----");
30         List<String> fieldNames = FieldUtils.getFieldNames(car);
31         for (String name : fieldNames){
32             System.out.println("Field name: " + name);
33         }
34         System.out.println("-----");
35         List<String> constructorNames = MethodUtils.getConstructorNames(car);
36         for (String name : constructorNames){
37             System.out.println("Constructors name: " + name);
38         }
39         System.out.println("-----");
40         List<String> methodNames = MethodUtils.getMethodNames(car);
41         for (String name : methodNames){
42             System.out.println("Method name: " + name);
43         }
44         System.out.println("-----");
45         Fisica fisica = new Fisica();
46         System.out.println("Super class name: " + ClassUtils.getSuperClassName(fisica));
47     }
48
49 }
```

Reflexão em Java

Importante!

► Aplicabilidade:

- Analisadores de código
- Engenharia reversa
- Função dotComplete das IDEs
- Serialização e deserialização de objetos Java em XML/JSON. Ex: XStream
- Criação e invocação de objetos;
- Frameworks diversos. Ex: Apache BeanUtils
- **Frameworks ORMs. Ex: Hibernate/JPA**
- Unidades de Testes. Ex: Junit.



Reflexão em Java

- ▶ Nem tudo são flores...
 - Desempenho (tempo x custo)
 - Segurança
 - Exposição da estrutura interna.



Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ **Engenharia de Software para SaS**
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Engenharia para SaS

- ▶ Sistemas autoadaptativos (do inglês, *Self-adaptive Systems SaS*) são capazes de modificar sua estrutura e comportamento em tempo de execução
- ▶ SaS são capazes de reconhecer as mudanças de contexto e propor soluções para continuar em funcionamento

Engenharia para SaS

- ▶ Adaptação:
 - Mudanças de Contexto
 - Novas necessidades de seus usuários
 - Requisitos não previstos na fase de projeto



Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ **Arquitetura de Referência e RA4SaS**
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

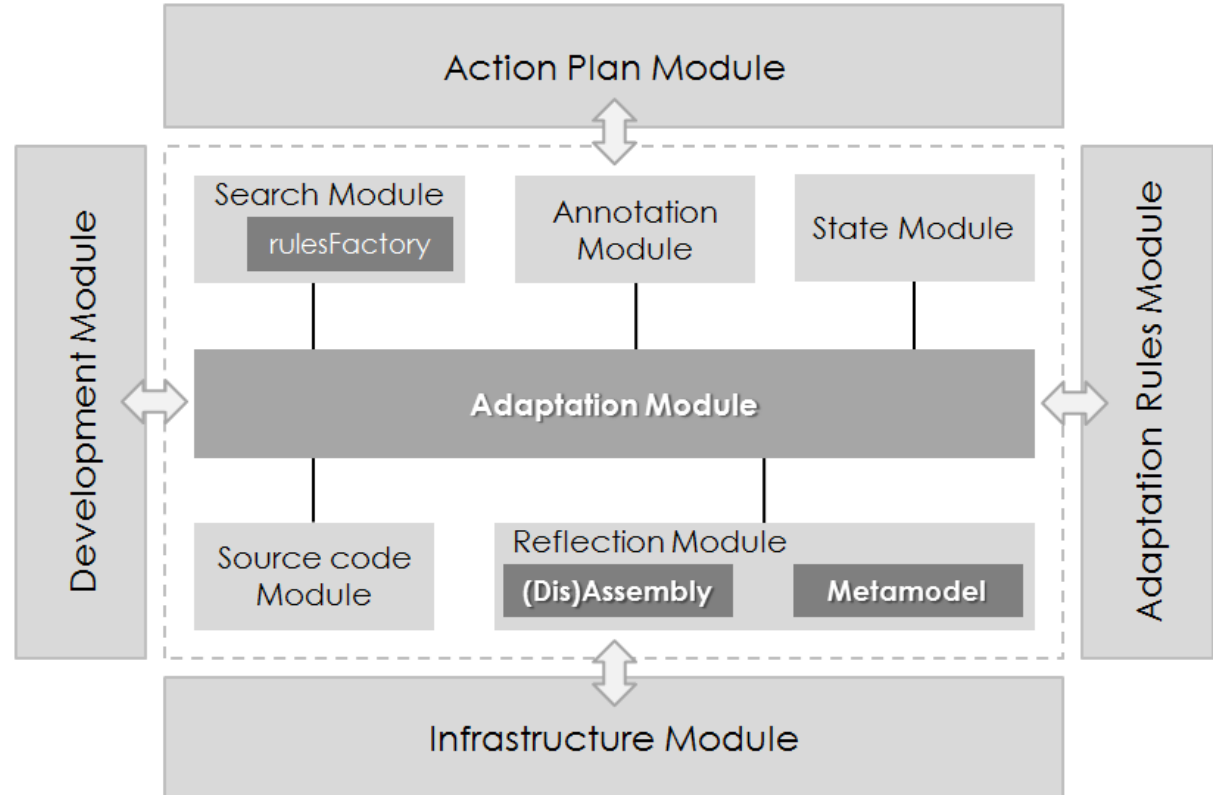
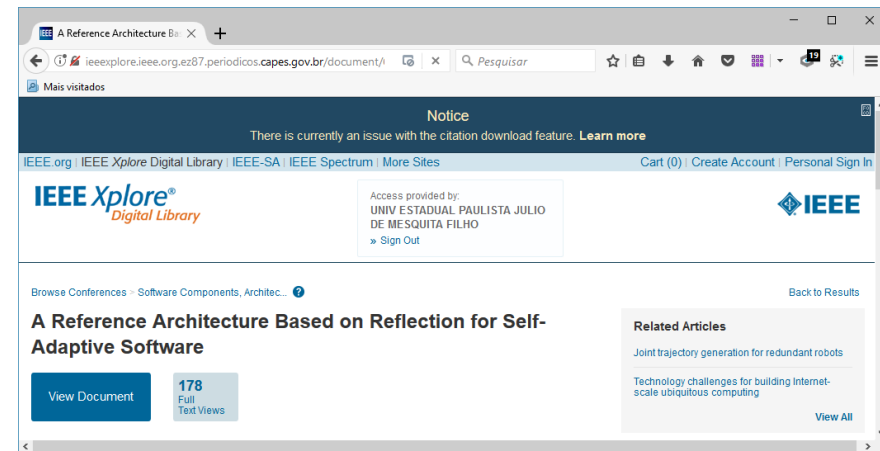
RA4SaS

▶ Reference Architecture for Self-adaptive Software (RA4SaS):

- Baseada em Reflexão
- Abordagem de adaptação externa
- Loop de controle MAPE-K
- Modalidade controlada de adaptação
- Diretrizes de desenvolvimento

RA4SaS

► Representação:



► Processo:



► Processo:



Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ **Grupo de Pesquisa**
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Grupo de Pesquisa

- ▶ Líder: Prof. Dr. Frank J. Affonso



🏠 ▶ Consultas ▶ Consulta parametrizada ▶ Consulta parametrizada

Consulta parametrizada

Grupo de pesquisa: Engenharia de Software para Sistemas Autoadaptativos

Instituição: UNESP

Líder(es): Frank José Affonso

Área: Ciência da Computação

◀◀ 1 ▶▶ 15 ▼

Total de registros: 1

🖨 Imprimir

📄 Exportar

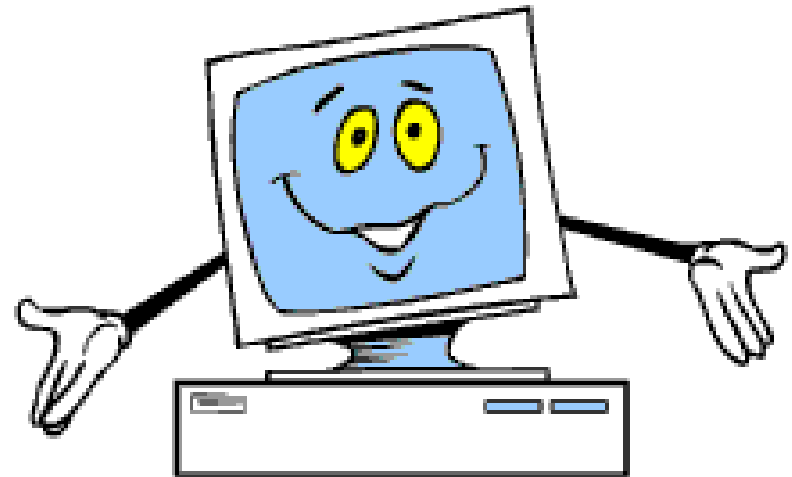
Voltar

Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ **Laboratório de Pesquisa**
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Laboratório de Pesquisa

- ▶ ReLaSE – Research Laboratory in Software Engineering – Coordenação Prof. Dr. Frank José Affonso



Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ **Perspectivas de Trabalho**
- ▶ Mão na massa

Perspectivas de Trabalho

▶ Trabalhos Futuros:

- Iniciação Científica
- TCC
- Grupo de Estudo
- Colaborações



Principais Tópicos

- ▶ Apresentação Docente
- ▶ Engenharia de Software
- ▶ Reflexão em Java
- ▶ Engenharia de Software para SaS
- ▶ Arquitetura de Referência e RA4SaS
- ▶ Grupo de Pesquisa
- ▶ Laboratório de Pesquisa
- ▶ Perspectivas de Trabalho
- ▶ Mão na massa

Mão na massa

► Atividade que combina reflexão com geração de código.

- SourceCodeGenerator
- ReflectionInActionJavaApp
- Outros



Mão na massa

- ▶ Faça um programa em Java que permitir gerar a String SQL de persistência para qualquer objeto de uma classe.

```
public class Demo {  
  
    public static void main(String[] args) {  
        Categoria categoria = new Categoria();  
  
        System.out.println("Source Code:");  
        System.out.println("-----");  
        System.out.print(GeneratorUtils.sourceCode(categoria));  
        System.out.println("-----");  
    }  
}
```



```
final String INSERT_CATEGORIA = "INSERT INTO Categoria(nome, descricao, precoDiario) VALUES(?, ?, ?)";
```



Reflexão em Java, Componentes, Frameworks e Além

Prof. Dr. Frank J. Affonso

Outubro de 2017