```java
import java.io.File;
import java.util.ArrayList;

public interface CompressionAlgorithm {
  public void compressFiles(ArrayList<File> files);
}

class ZipCompressionAlgorithm implements CompressionAlgorithm {
  public void compressFiles(ArrayList<File> files) {
    System.out.println("using ZIP algorithm");
  }
}

class RarCompressionAlgorithm implements CompressionAlgorithm {
  public void compressFiles(ArrayList<File> files) {
    System.out.println("using RAR algorithm");
  }
}

class CompressionContext {
  private CompressionAlgorithm algorithm;

  // This can be set at runtime by application preferences
  public void setCompressionAlgorithm(CompressionAlgorithm algorithm) {
    this.algorithm = algorithm;
  }

  public void createArchive(ArrayList<File> files) {
    algorithm.compressFiles(files);
  }
}

class Client {
  public static void main(String[] args) {
    CompressionContext ctx = new CompressionContext();
    // We could assume context is already set by preferences
    ArrayList<File> list = new ArrayList<>();
    // Run compression algorithm due to setted algorithm
    ctx.setCompressionAlgorithm(new RarCompressionAlgorithm());
    ctx.createArchive(list);
    ctx.setCompressionAlgorithm(new ZipCompressionAlgorithm());
    ctx.createArchive(list);
  }
}
```

Pattern **STRUTTURALI**	Pattern **CREAZIONALI**	Pattern **COMPORTAMENTALI**
□ Adapter □ Decorator □ Composite	□ Abstract Factory □ Singleton	□ Observer □ State □ Visitor □ Strategy