Následující struktura ošetřuje mnoho výjimek

```
try {
        madMethod();
} catch (AuthorizationException e) {
        System.err.println(e.getErrorName() + " " + e.getType());
        e.printStackTrace();
} catch (BusinessRuleViolation e) {
        System.err.println(e.getErrorName() + " " + e.getBusinessRuleViolated() + " " +
        e.getCauser().getClass().getSimpleName());
        e.printStackTrace();
 } catch (DuplicateEntityViolation e) {
        System.err.println(e.getErrorName() + " " + e.getDuplicatedEntity().getName());
         e.printStackTrace();
} catch (SearchException e) {
        System.err.println(e.getErrorName() + " " + e.getMessage()); e.printStackTrace();
} catch (SystemError e) {
        System.err.println("Unexpected error" +e.getErrorName() + " " + e.getType());
        e.printStackTrace();
}
```

Zkusme to vylepšit dle visitor na něco jako

```
public static void main(String[] args) {

        try {
          madMethod();
        } catch (ServiceException e) {
            ExceptionHandler.handle(e);
            // or to apply Visitor, but then it is e.accept(handler)
        }
}
```

Poznámka

ElementA.accept(Visitor v) {v.visit(this)}

a

```
class *Visitor* {

      visit(ElementA) {}

      visit(ElementB) {}

      visit(ElementC) {}

}
```