# Improving Exception Handling with Recommendations

Eiji Adachi Barbosa

Advisor: Alessandro Garcia

LES | DI | PUC-Rio – Brazil | ICSE'14 Doctoral Symposium

**OPUS Research Group**

# Exceptions occurrence

System that manages photo albums

View

```
PhotoScreen.handleEvent( event ){
   Controller.performAction( event );
}
```

Controller

```
Controller.performAction( event ){
   if(event == REMOVE_PHOTO){
      Photo.remove(event.getObject());
      PhotoScreen.update(SUCCESSFUL);
   }
}
```

Model

```
Photo.remove( object ){
   PhotoAccessor.delete(object);
}
```

```
PhotoAccessor.delete( object ){
   RecordStore.deleteRecord(object);

}
```

# Ignoring exceptions

## Implementing an empty and generic catch block

View

```
PhotoScreen.handleEvent( event ){
    Controller.performAction( event );
}
```

Controller

```
Controller.performAction( event ){
    if(event == REMOVE_PHOTO){
        Photo.remove(event.getObject());
        PhotoScreen.update(SUCCESSFUL);
    }
}
```

Model

```
Photo.remove( object ){
    PhotoAccessor.delete(object);
}
```

```
PhotoAccessor.delete( object ){
    try { RecordStore.deleteRecord(object); }
    catch( Exception e ){//ignore }
}
```

# Poor quality of exception handling

- Excessive use of *generic* exception types and *empty* catch blocks [1]
- High number of *uncaught* exceptions [2]
- *Defect* density of exception handling code is three times higher than normal code [3]


[1] Cabral, B., and Marques, P. Exception Handling: A Field Study in Java and .NET, *ECOOP 2007*.
[2] Cacho, N. et al. Trading robustness for maintainability: An empirical study of evolving c# programs. *ICSE 2014*.
[3] Sawadpong, P., Allen, E. B., and Williams, B. J. Exception Handling Defects: An Empirical Study. *HASE 2012*.

# Limitations of state-of-art

- Current solutions focus on computing *propagation paths of exceptions* [4, 5, 6]
  - No support to decide where and how to handle exceptions

[4] Robillard, M.P. and Murphy, G.C. Analyzing Exception Flow in Java Programs. *FSE 1999.*
[5] Fu, C. and Ryder, B.G. Exception-Chain Analysis: Revealing Exception Handling Architecture in Java Server Applications. *ICSE 2007.*
[6] Garcia, I. and Cacho, N. eFlowMining: An Exception- Flow Analysis Tool for .NET Applications. *LADC Workshops 2011*.

# Detecting poor exception handling

View

```
PhotoScreen.handleEvent( event ){
    Controller.performAction( event );
}
```

Controller

```
Controller.performAction( event ){
    if(event == REMOVE_PHOTO){
        Photo.remove(event.getObject());
        PhotoScreen.update(SUCCESSFUL);
    }
}
```

Model

```
Photo.remove( object ){
    PhotoAccessor.delete(object);
}
```

```
PhotoAccessor.delete( object ){
    try { RecordStore.deleteRecord(object); }
    catch( Exception e ){//ignore }
}
```

# Limitations of state-of-art

- Lack of explicit exception handling policies
  - An *exception handling policy* regards to designer's original intent regarding the use of exceptions
    - What exception types
    - Where exceptions are thrown, handled, propagated and remapped
  - There are no means to specify exception handling policies

# Handling exceptions

Exceptions must be handled by the controller

View

```
PhotoScreen.handleEvent( event ){
   Controller.performAction( event );
}
```

Controller

```
Controller.performAction( event ){
   if(event == REMOVE_PHOTO){
      try{Photo.remove(event.getObject());
          PhotoScreen.update(SUCCESSFUL);
       }catch(Exception e){update(ERROR);}
   }
}
```

Model

```
Photo.remove( object ){
   PhotoAccessor.delete(object);
}
```

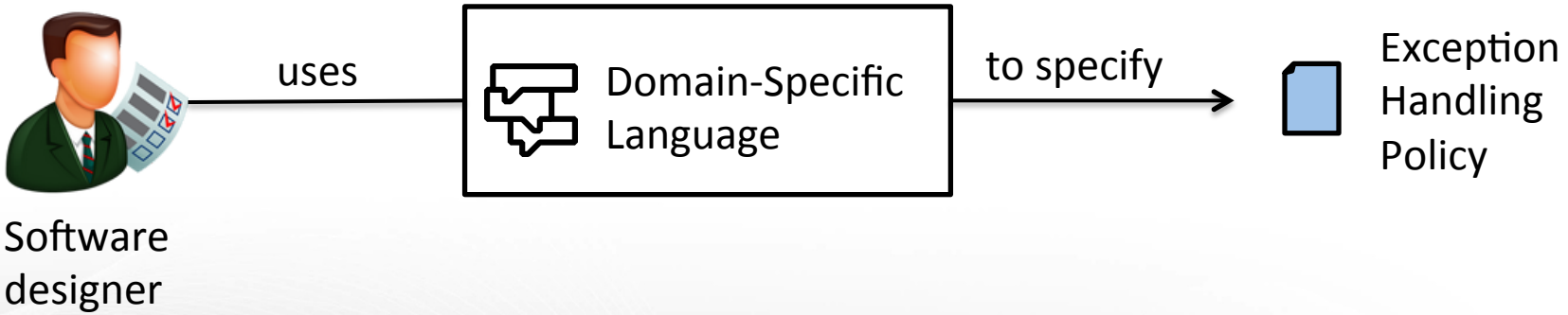
```
PhotoAccessor.delete( object ){
   RecordStore.deleteRecord(object);
}
```

# Objective and Proposed solution

- Objective:
  - Aid developers in producing better exception handling code
- Key elements of proposed solution:
  - **Domain-specific language** to specify exception handing policies
  - **Verification tool** to check source code adherence
  - **Recommender system** to assist developers in implementing adhering code

# Usefulness and Applicability

Designers use the DSL to specify the exception handling policy



# Usefulness and Applicability

Developers read the specification to comprehend expected use of exceptions

Software designer

Domain-Specific Language

Exception Handling Policy

Developer

Developer

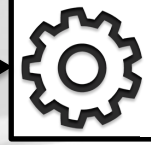
# Usefulness and Applicability

Recommender aids developers to implement code adherent to the specification

Software designer

Domain-Specific Language

Exception Handling Policy



Recommender System

Developer

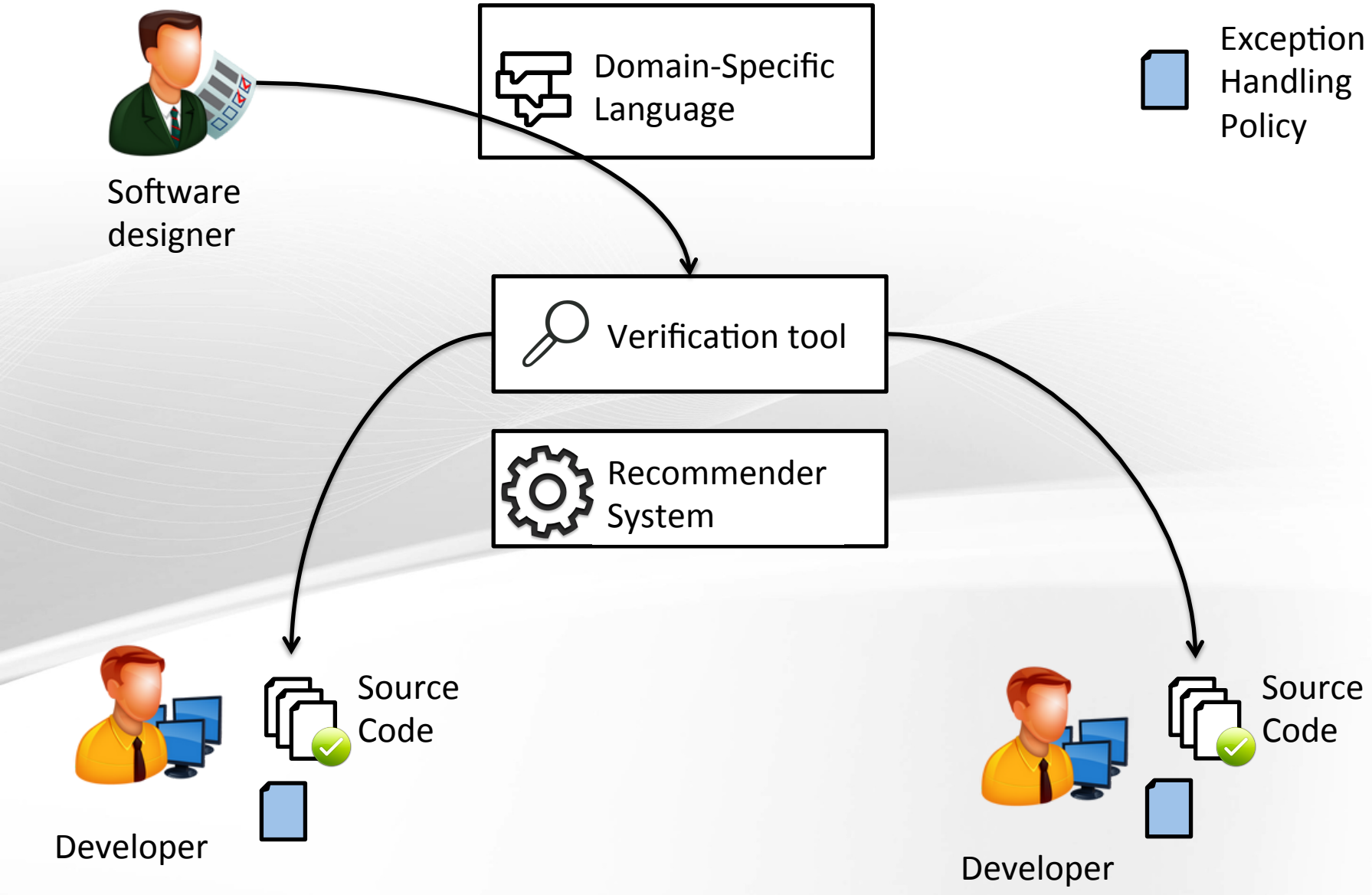
Source Code

Developer

Source Code

# Usefulness and Applicability

Designers use verification tool to check source code



# Technical challenges

- Design and implement an expressive domain-specific language
- Improve current recommending techniques to incorporate information from exception handling policies

# Evaluation

- Domain-specific language:
  - Case study
  - Qualitative study
- Recommender system:
  - Controlled experiments
  - Qualitative study

# Progress and Planning

- Done:
  - DSL design, implementation and evaluation
- To-Do (next 2 years):
  - Expand recommending heuristics
  - Implement recommender system

# Improving Exception Handling with Recommendations

Eiji Adachi Barbosa

Thank you

LES | DI | PUC-Rio – Brazil | ICSE'14 Doctoral Symposium

**OPUS Research Group**