



TECHNISCHE
UNIVERSITÄT
DRESDEN



ResUbic



Technische Universität Dresden, Software Technology Group

Tool Supported OCL Refactoring Catalogue

Jan Reimann, Claas Wilke, Birgit Demuth, Michael Muck, Uwe Aßmann

OCL Workshop 2012, Innsbruck, 30/09/2012



DRESDEN
concept
Exzellenz aus
Wissenschaft
und Kultur



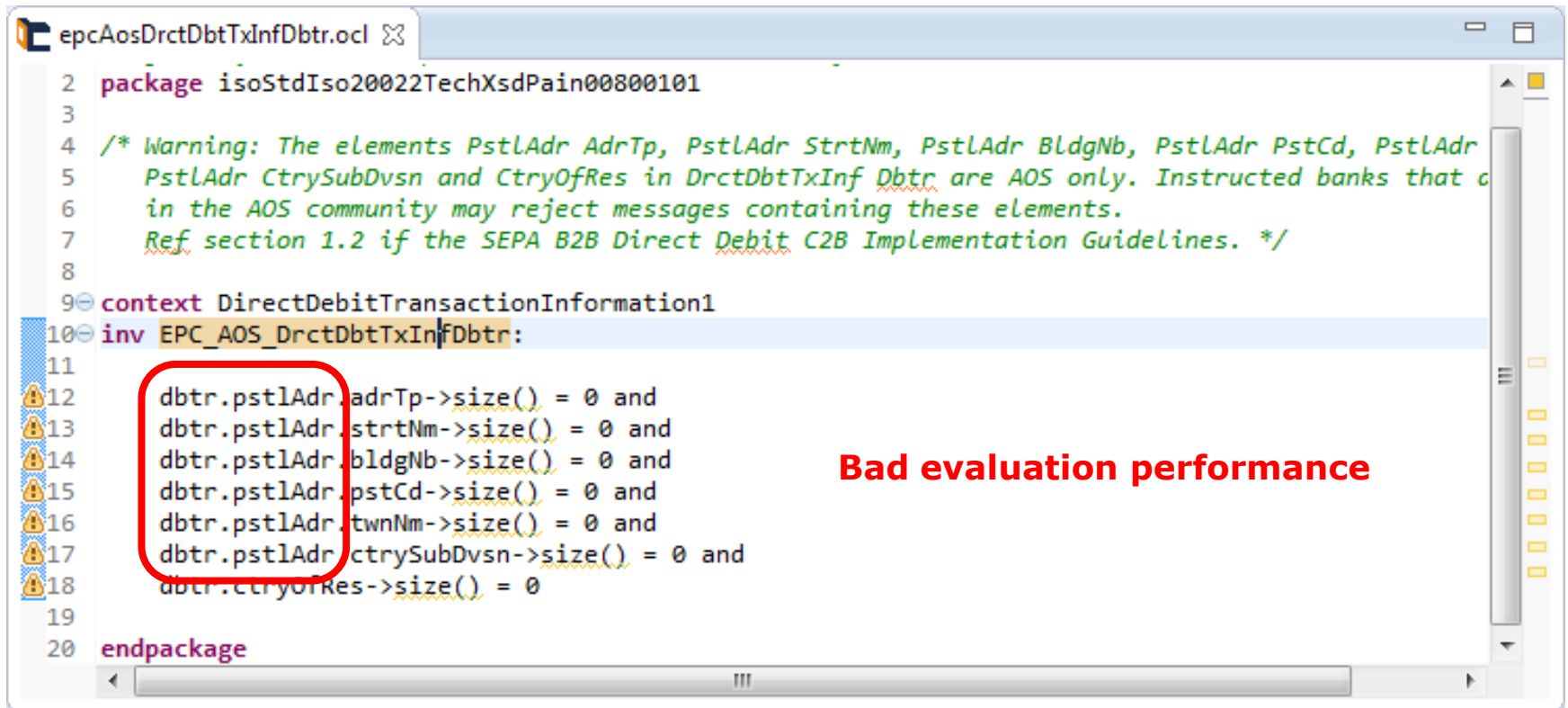
Europa fördert Sachsen.
ESF
Europäischer Sozialfonds



Agenda

- Motivating example from finance sector
- Existing OCL refactorings and their limitations
- Extended catalogue and improvements
- Demo
- Conclusion and Future Work

Motivating Example

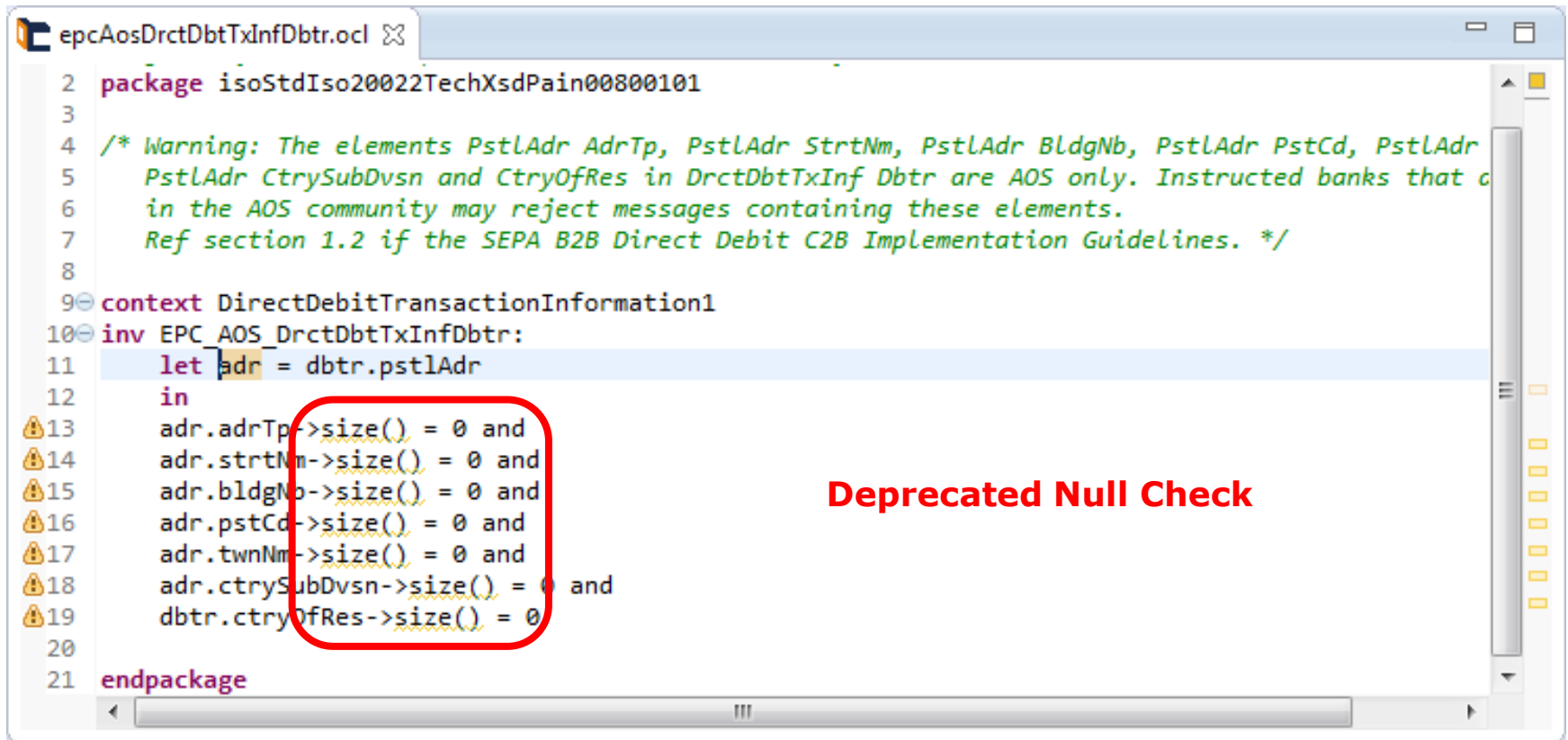


```
epcAosDrctDbtTxInfDbtr.ocl
2 package isoStdIso20022TechXsdPain00800101
3
4 /* Warning: The elements PstlAdr AdrTp, PstlAdr StrtNm, PstlAdr BldgNb, PstlAdr PstCd, PstlAdr
5 PstlAdr CtrySubDvsn and CtryOfRes in DrctDbtTxInf Dbtr are AOS only. Instructed banks that c
6 in the AOS community may reject messages containing these elements.
7 Ref section 1.2 if the SEPA B2B Direct Debit C2B Implementation Guidelines. */
8
9 context DirectDebitTransactionInformation1
10 inv EPC_AOS_DrctDbtTxInfDbtr:
11
12 dbtr.pstlAdr.adrTp->size() = 0 and
13 dbtr.pstlAdr.strtNm->size() = 0 and
14 dbtr.pstlAdr.bldgNb->size() = 0 and
15 dbtr.pstlAdr.pstCd->size() = 0 and
16 dbtr.pstlAdr.twnNm->size() = 0 and
17 dbtr.pstlAdr.ctrySubDvsn->size() = 0 and
18 dbtr.ctryOfRes->size() = 0
19
20 endpackage
```

Bad evaluation performance

<http://nomos-software.com/> Nomos XML validation service demo

Motivating Example



```
epcAosDrctDbtTxInfDbtr.ocl
2 package isoStdIso20022TechXsdPain00800101
3
4 /* Warning: The elements PstlAdr AdrTp, PstlAdr StrtNm, PstlAdr BldgNb, PstlAdr PstCd, PstlAdr
5 PstlAdr CtrySubDvsn and CtryOfRes in DrctDbtTxInf Dbtr are AOS only. Instructed banks that c
6 in the AOS community may reject messages containing these elements.
7 Ref section 1.2 if the SEPA B2B Direct Debit C2B Implementation Guidelines. */
8
9 context DirectDebitTransactionInformation1
10 inv EPC_AOS_DrctDbtTxInfDbtr:
11 let adr = dbtr.pstlAdr
12 in
13 adr.adrTp->size() = 0 and
14 adr.strtNm->size() = 0 and
15 adr.bldgNb->size() = 0 and
16 adr.pstCd->size() = 0 and
17 adr.twnNm->size() = 0 and
18 adr.ctrySubDvsn->size() = 0 and
19 dbtr.ctryOfRes->size() = 0
20
21 endpackage
```

Deprecated Null Check

<http://nomos-software.com/> Nomos XML validation service demo

Motivating Example

```
epcAosDrctDbtTxInfDbtr.ocl
2 package isoStdIso20022TechXsdPain00800101
3
4 /* Warning: The elements PstlAdr AdrTp, PstlAdr StrtNm, PstlAdr BldgNb, PstlAdr PstCd, PstlAdr
5 PstlAdr CtrySubDvsn and CtryOfRes in DrctDbtTxInf Dbtr are AOS only. Instructed banks that c
6 in the AOS community may reject messages containing these elements.
7 Ref section 1.2 if the SEPA B2B Direct Debit C2B Implementation Guidelines. */
8
9 context DirectDebitTransactionInformation1
10 inv EPC_AOS_DrctDbtTxInfDbtr:
11     let adr = dbtr.pstlAdr
12     in
13     adr.adrTp = null and
14     adr.strtNm = null and
15     adr.bldgNb = null and
16     adr.pstCd = null and
17     adr.twnNm = null and
18     adr.ctrySubDvsn = null and
19     dbtr.ctryOfRes = null
20
21 endpackage
```

<http://nomos-software.com/> Nomos XML validation service demo

OCL Refactoring

- ***OCL-exclusive refactorings*** and *OCL co-refactorings*
- Aim at improving qualities such as [Fowler 99]:
 - Reusability
 - Readability
 - Understandability
 - Comprehensibility
 - Maintainability
 - Evaluation performance
- For removing bad smells such as [Correa, Werner 07]:
 - Bad Evaluation performance
 - Deprecated null checks
 - Magic literal
 - Long journey
- One of the most demanded features in OCL IDEs [Chimiak-Opoka et al. 11]

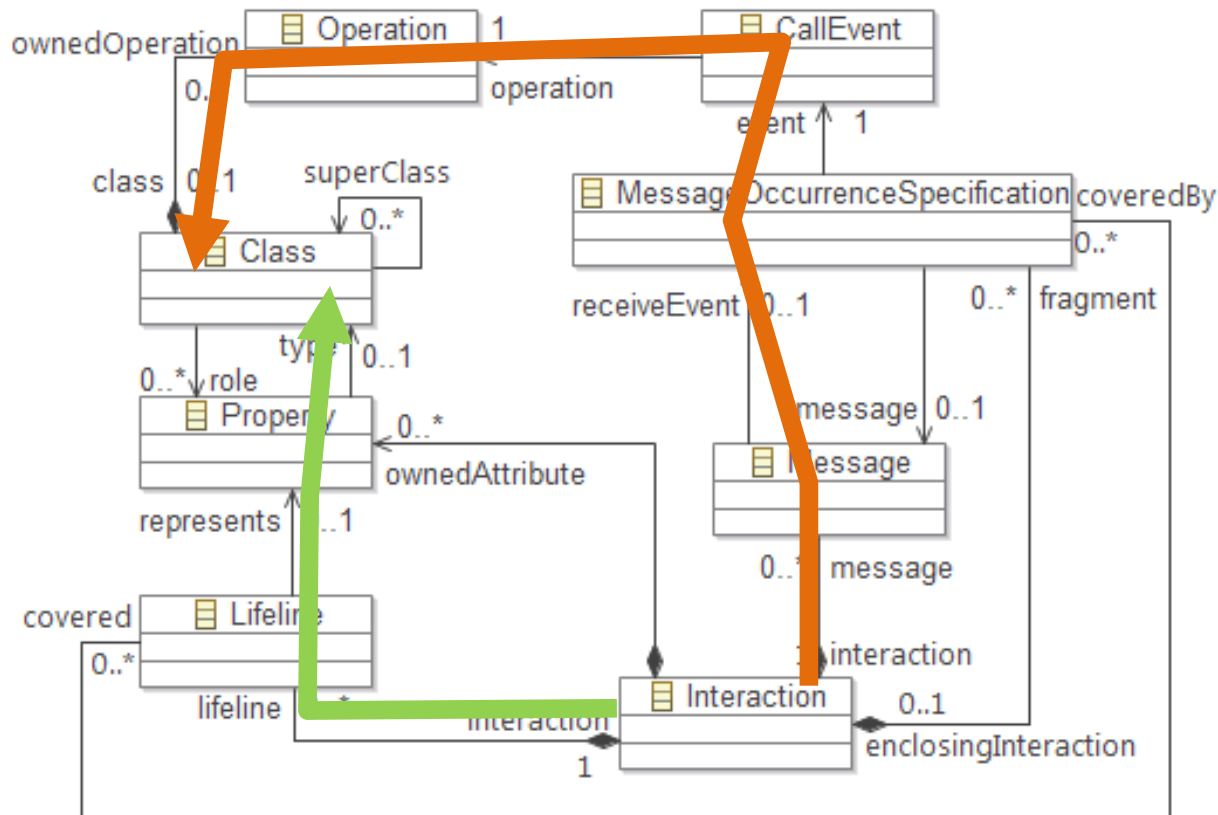
Existing OCL-Exclusive Refactorings

- Automatic simplification of generated OCL constraints [Giese, Larsson 05]
 - Logical expressions are simplified
 - Example: `(false and exp) → (false)`
- First OCL-exclusive refactoring catalogue [Correa, Werner 07]
 - Defined OCL smells...
 - ...upon which they specified according refactorings

Analysis Of Existing OCL-Exclusive Refactorings

- **Change Initial Navigation:**

- Remove *Verbose Expressions* or *Long Journeys*
- Lengthy navigation paths should be replaced by alternative shorter paths
- Just propose to find alternative paths and pick the shortest one
- → might not be semantics preserving!



[Correa, Werner 07], [Correa, Werner, Barros 09]

Analysis Of Existing OCL-Exclusive Refactorings

- **Add Variable Definition/Replace Expression By Variable:**
 - Two separate refactorings
 - Remove *Magic Literal*
 - A variable is initialised with a literal
 - Afterwards all occurrences are replaced by the variable
 - Not combined to one refactoring
 - Not mentioned that variable name uniqueness has to be verified
 - Not only literals but navigation paths can be replaced (like in the example)
 - We combined this refactoring to *Extract Variable*

Extended Catalogue

- Corrected specifications of all refactorings from [Correa, Werner 07]
- Identified 4 categories:
 1. Renamings
 2. Removals/Materialisations
 3. Extractions/Inlinings
 4. Separations/Merges
- Identified further OCL-exclusive refactorings

New OCL-Exclusive Refactorings

Removals/Materialisation	Extractions/Inlinings	Separations/Merges
Remove deprecated <code>null</code> check	Extract/Inline variable	Merge chained <code>let</code> expressions
Remove unused Elements	Extract Property/Operation	Split/Merge context declarations
Remove/Add redundant brackets	Inline Property/Operation	
Remove/Materialise <code>self</code>		
Remove/Materialise type declarations		
Remove implicit <code>asSet</code>		
Remove implicit <code>collect</code>		

Catalogue can be seen here in the following days:

<http://www.dresden-ocl.org/refactoring/>

Demo



Conclusion And Future Work

Conclusion

- Analysed existing OCL-exclusive refactorings
- Identified limitations and improved them
- Specified a catalogue of 28 refactorings
- Implemented OCL-exclusive refactoring support for Dresden OCL [Wilke, Demuth 11]
- Used the generic refactoring framework Refactory [Reimann, Seifert, Aßmann 12]

Future Work

- Finish implementation
- OCL co-refactoring

The End

Thank You for Your Attention

Bibliography

[Chimiak-Opoka et al. 11] *OCL Tools Report based on the IDE4OCL Feature Model*. OCL and Textual Modelling, 2011

[Correa, Werner 07] *Refactoring object constraint language specifications*. Software and Systems Modeling, 2007

[Correa, Werner, Barros 09] *Refactoring to improve the understandability of specifications written in object constraint language*. Software, IET, 3(2):69–90, 2009

[Fowler 99] *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, 1999

[Giese, Larsson 05] *Simplifying Transformations of OCL Constraints*. Model Driven Engineering Languages and Systems. Springer, 2005

[Reimann, Seifert, Aßmann 12] *On the Reuse and Recommendation of Refactoring Specifications*. Software and Systems Modeling, 2012

[Wilke, Demuth 11] *UML is still inconsistent! How to improve OCL Constraints in the UML 2.3 Superstructure*. EASST, 2011

Contact



Jan Reimann
Technische Universität Dresden
Software Technology Group

jan.reimann@tu-dresden.de

<http://st.inf.tu-dresden.de/>



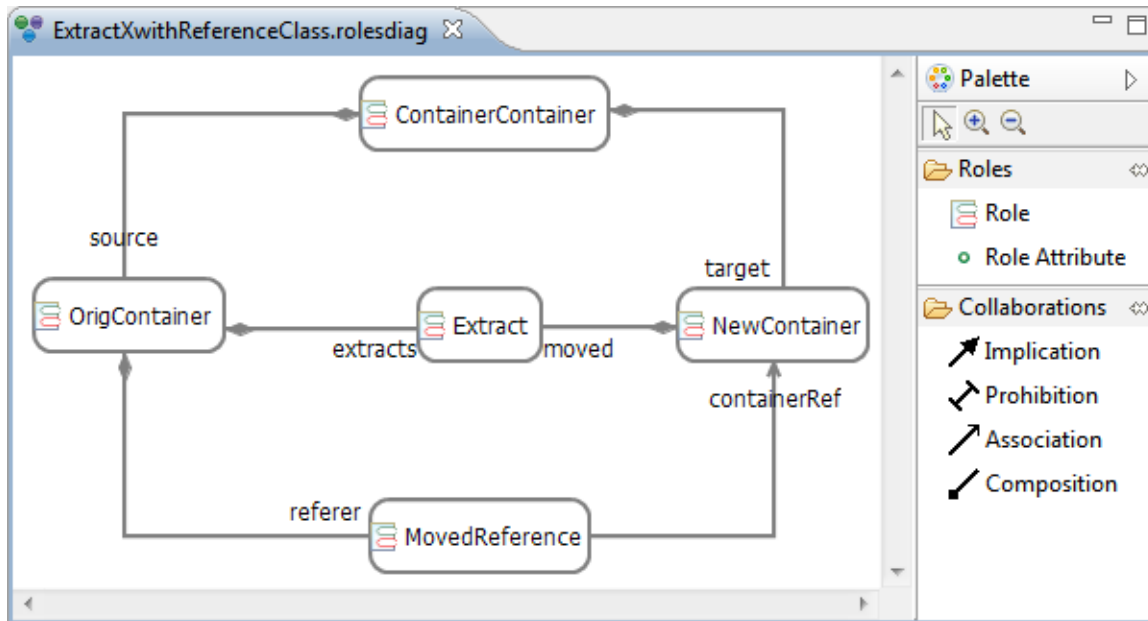
<http://dresden-ocl.org/refactoring/>



<http://modelrefactoring.org/>

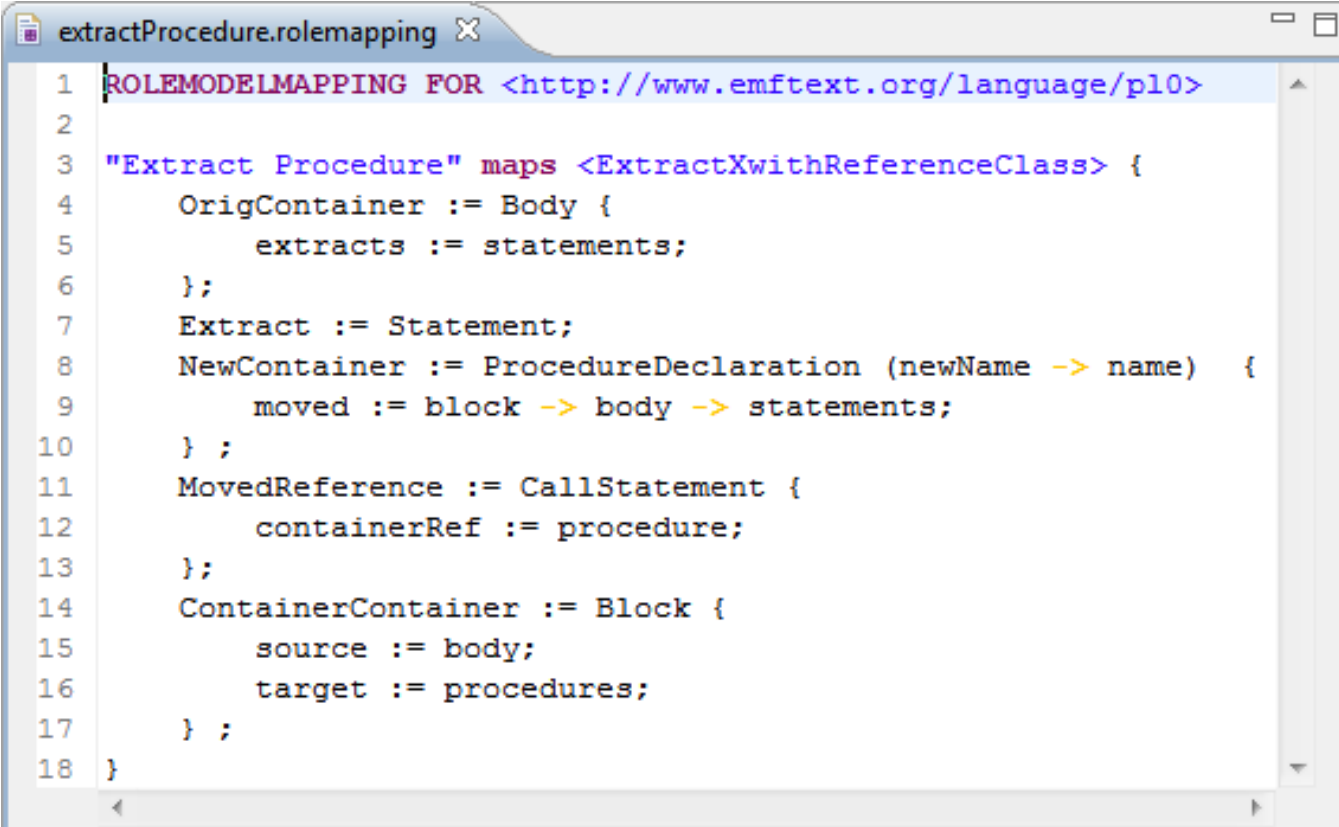


Defining Generic Refactorings With Roles



```
1 REFACTORING FOR <ExtractXwithReferenceClass>
2
3 STEPS {
4   object containerContainerObject := ContainerContainer from uptree(INPUT);
5   object origContainerObject := OrigContainer as trace(INPUT);
6   index extractsIndex := first(INPUT);
7
8   create new nc:NewContainer in containerContainerObject;
9   assign nc.newName;
10  move OrigContainer.extracts to nc;
11  create new mr:MovedReference in origContainerObject at extractsIndex;
12  set use of nc in mr;
13 }
```

Instantiation By Mapping The Roles



```
extractProcedure.rolemapping x
1 ROLEMODEL MAPPING FOR <http://www.emftext.org/language/pl0>
2
3 "Extract Procedure" maps <ExtractXwithReferenceClass> {
4   OrigContainer := Body {
5     extracts := statements;
6   };
7   Extract := Statement;
8   NewContainer := ProcedureDeclaration (newName -> name) {
9     moved := block -> body -> statements;
10  };
11  MovedReference := CallStatement {
12    containerRef := procedure;
13  };
14  ContainerContainer := Block {
15    source := body;
16    target := procedures;
17  };
18 }
```

Mapping To Paths

