



Grupo de Investigación en
Reutilización y Orientación a
Objeto

Caracterización de Refactorizaciones para la Implementación en Herramientas



Autores:

Carlos López Nozal

clopezno@ubu.es

Raúl Marticorena Sánchez

rmartico@ubu.es

Yania Crespo González-Carvajal

yania@infor.uva.es



Índice



- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de Estudio
- Conclusiones y Líneas Futuras



Introducción

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

3/18

■ Actividades del proceso de Refactoring

1. Detectar oportunidades de refactoring
2. Determinación de un conjunto de refactorizaciones
3. Aplicación del conjunto elegido
4. Computar los efectos de la refactorización
consistencia entre los distintos artefactos software
cia entre los distintos artefactos software
ware
mientas no incorporan todas las actividades
ogos son muy extensos y no están orientados a su



Introducción

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

4/18

■ Objetivo



Proporcionar un modelo de caracterización para definir criterios que asistan el proceso de implementación de una herramienta de refactorización.

- "Seleccionar un conjunto de refactorizaciones de diseño que asistan a la solución de un *bad smell*, indicando el orden de implementación de las mismas"



Dotar al proceso de desarrollo

- Organización: agrupar y ordenar refactorizaciones
- Comprensión: conocer relaciones existentes
- Reutilización: compartir elementos
- Estimación de tareas: cuantificación



Antecedentes

- Introducción
- **Antecedentes**
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

5/18

- Clasificación basada en criterios de composición
Opdyke [Opdyke, W. F., 1992]
 - Low Level Refactoring (26)
 - High Level Refactoring
- Clasificación funcional de Fowler (68) [Fowler, M., 2000]
 - composing methods
 - moving features between objects
 - organizing data
 - simplifying conditional expressions
 - dealing with generalizations
- Relación Bad Smell y Refactoring [Fowler, M., 2000], [Wake W. C., 2003]





Antecedentes

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

6/18

- Otras clasificaciones basadas en la estructuración del dominio del conocimiento
 - Caracterización de trabajos de refactorización [Crespo Y., Marqués, J.M., 2001]
 - Ámbito: Análisis, Diseño
 - Automáticas, SemiAutomáticas
 - ...
 - Relaciona el conocimiento declarativo con el conocimiento operativo [Garzas, J., Piattini, M., 2004]
 - Conocimiento declarativo: principios, heurísticas, mejores prácticas, bad smells, patrones
 - Conocimiento operativo: transformaciones, refactorizaciones
- Ninguna clasificación esta orientada a asistir la implementación en herramientas



Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

7/18

- Criterios de caracterización agrupados
 - Relaciones entre las refactorizaciones
 - Ámbito de la refactorización
 - Identifica y cuantifica el conjunto de entidades sobre las que actúa una refactorización
 - Tipo de conocimiento para su comprensión
- Los criterios deben ser generales para ser extendidos a otros múltiples catálogos
 - Refactorizaciones de código
 - **Entidades:** sistema, clases, atributos, métodos, parámetros, instrucciones, variables locales
 - Refactorizaciones sobre diagramas de estados
 - **Entidades:** acción, estado, transición



Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

8/18

■ Relaciones entre las refactorizaciones

■ Relación de similitud

- Mecanismos de definición muy similares
- Facilita comprensión
- Ej: *AddParameter*, *RenameParameter*

■ Relación de uso

- Indica que en la definición de una refactorización puede ser necesaria la aplicación de otras
- Favorece la organización proporcionando un orden de implementación
- Ej: *ExtractClass* usa *ExtractField* y *ExtractMethod*

Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

9/18

■ Relaciones entre las refactorizaciones

■ Relaciones de composición

- Una refactorización esta compuesta por operaciones atómicas aplicadas sobre las entidades del sistema
 - Crear entidad
 - Actualizar entidad
 - Borrar entidad
- *Ej: Extract Class esta compuesta: Crear clase, Crear atributo, Eliminar atributo, Crear método, Borrar método, Crear instrucción, Borrar instrucción, Crear parámetro, Borrar parámetro y Crear Variable Local y Borrar variable local.*
- Permite cuantificar la complejidad de la refactorización



Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

10/18

■ Ámbito de las refactorizaciones

- Identifica y cuantifica el conjunto de entidades sobre las que actúa una refactorización
- Se especifica como un conjunto de pares (multiplicidad, entidad)
- Puede ser analizado desde diferentes perspectivas
 - Entrada
 - Precondición
 - Acción (operación)

Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

11/18

■ Ámbito de las refactorizaciones

■ Ámbito de entrada

- Identifica y cuantifica las entidades necesarias como entrada para realizar una refactorización
- *Ej: $AE = \{(1, \text{método}), (n, \text{parámetro})\}$*
 - *Add Parameter, Remove Parameter*
- Utilidad
 - Reutilización interfaces gráficos
 - Clasificación
 - Diseño -> *Move Method*
 - Implementación -> *Extract Method*

Caracterización de Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

12/18

■ Ámbito de las refactorizaciones

■ Ámbito de las precondiciones

- Identifica y cuantifica las consultas sobre las entidades de un sistema previas a la aplicación
- *Ej: Conjunto de refactorizaciones con $AP = \{(n, \text{clases}) (n, \text{métodos}) (n, \text{instrucciones})\}$*
 - *Add Parameter, Form Template Method, Introduce Parameter Object, Remove Parameter, Replace Parameter with Method*

■ Ámbito de acciones

- Identifica y cuantifica las entidades transformadas
- Puede asistir la reutilización de las acciones de transformación asociadas



Criterios para Analizar Refactorizaciones

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

13/18

■ Tipos de conocimiento

■ Conocimiento declarativo

- Inicia el proceso de refactoring
- Relación entre conocimiento declarativo y un conjunto de refactorizaciones
- *Ej: Herramienta que elimine código duplicado:*
 - *Extract Method, Extract Class, Pull Up Method, Form Template Method*

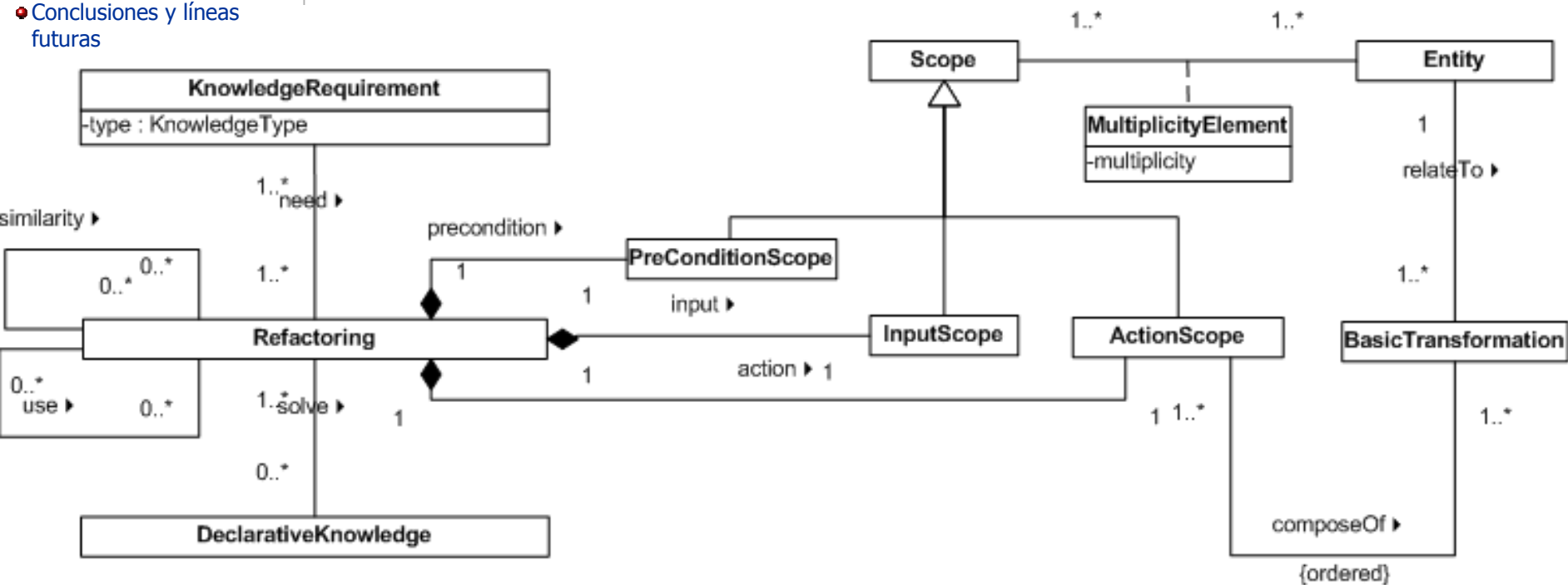
■ Requisitos de conocimiento

- Conceptos básicos y avanzados asociados a la aplicación de la refactorización
 - Conocimientos avanzados en OO: reflectividad, patrones de diseño, aserciones
- *Ej: Agrupación por complejidad*
 - *Form Template Method necesita PD Template Method*
 - *Rename Method se basa en conocimientos básicos*



Modelado de la Caracterización

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras



Caso de Estudio

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- **Caso de estudio**
- Conclusiones y líneas futuras

15/18

■ Carga del modelo

- Caracterización de 44 refactorizaciones del catálogo de Fowler.
 - *Making Methods Calls Simpler*(15), *Dealing with Generalization*(12), *Moving features between Objects* (8) y *Composing Methods* (9)
- Entidades consideradas
 - *System, Class, Method, Instruction, Parameter y Local Variable*
- Conocimiento declarativo considerado
 - *Bad Smell* o defecto de código

Caso de Estudio

- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

16/18

- Seleccionar un conjunto de refactorizaciones de diseño que asistan la solución del Bad Smell *Large Class* indicando el orden de implementación de las mismas
 - Seleccionar el subconjunto de *refactorizaciones* asociados al *bad smell Large Class*
 - Seleccionar aquellas que no trabajan con *instrucciones* en el *ámbito de entrada*
 - Incluir en el conjunto, las refactorizaciones de diseño asociadas con la *relación de uso*
 - Cuantificar el número de *operaciones atómicas* basado en las instancias de la asociación `composeOf` y ordenar las refactorizaciones respecto a la cuantificación
 - Resultado
 - *Rename Method (1), Move Field(3), Push Down Field(3), Extract Interface (3), Parameterize Method(3), Move Method(6), Push Down Method(6), Extract Class (10), Extract Subclass(10)*



Conclusiones y Líneas Futuras



- Introducción
- Antecedentes
- Caracterización de Refactorizaciones
- Modelado de Caracterización
- Caso de estudio
- Conclusiones y líneas futuras

17/18

- Modelo abierto que permite caracterizar refactorizaciones de distintos catálogos
- Se añade un coste de caracterización
- Puede avanzar en la mejora de calidad en el desarrollo de la herramienta dotando al proceso
 - Organización, comprensión, reutilización y estimación de tareas
- Validar el modelo respecto a catálogos de refactorizaciones



Gracias por su atención



Caso de Estudio

● Introducción	
● Antecedentes	
● Caracterización de Refactorizaciones	de
● Modelado de Caracterización	de
● Caso de estudio	
● Conclusiones y líneas futuras	

19/18

- Dado un conjunto de refactorizaciones ya implementado por la una herramienta asistir el proceso de reutilización en la incorporación de una nueva refactorización
 - Conjunto Inicial: *Making Method Calls Simpler*
 - Añadir *Move Method*
 - *Búsqueda de refactorizaciones con el mismo ámbito*
 - Reutilización de Interfaces -> Ámbito de entrada
 - Replace Constructor with Factory Method
 - Reutilización de consultas -> Ámbito de precondiciones
 - Hide Method y Remove Setting Method
 - Reutilización de transformaciones -> Ámbito de acciones
 - Parameterize Method

