



Grupo de Investigación en  
Reutilización y Orientación a  
Objeto

# Un Framework para la Reutilización de la Definición de Refactorizaciones



Autores:

Yania Crespo González-Carvajal

Carlos López Nozal

Raúl Marticorena Sánchez

[yania@infor.uva.es](mailto:yania@infor.uva.es)

[clopezno@ubu.es](mailto:clopezno@ubu.es)

[rmartico@ubu.es](mailto:rmartico@ubu.es)



# Índice

- Introducción y Objetivos
- Contexto Inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras



# Introducción y Objetivos

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

3

## ■ Refactorizar

- "Proceso de cambiar un sistema software para mejorar su estructura interna, preservando el comportamiento externo"

## ■ Líneas de investigación abiertas [Mens 2004]

- Definición de nuevas refactorizaciones
- Identificación *Bad Smell*
- Ejecución de refactorizaciones
- Definición de refactorizaciones con reutilización
- Búsqueda de cierta independencia del lenguaje
- etc...



# Introducción y Objetivos

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

4

## ■ Múltiples IDE's incluyen soporte para un único lenguaje

- Eclipse -> Java
- Visual Studio 2005 Beta -> C#
- ...

## ■ Objetivo

*"Diseñar un soporte software para la definición y ejecución de refactorizaciones con cierta independencia del lenguaje"*

# Contexto Inicial

- Introducción y Objetivos
- Contexto inicial
  - ▶ Trabajos Previos
  - ▶ Problemas
  - ▶ Arquitectura
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

5

## ■ Independencia del lenguaje

- MOON [Crespo 2000]
  - Lenguaje modelo  $\cong$  metamodelo
  - Construcciones comunes de los lenguajes O.O.
    - Descripción de clases, propiedades, métodos...
- Extensiones de MOON [López 2003]
  - Representan características variables

## ■ Preservación del comportamiento

- Basado en Aserciones sobre MOON  
[Marticorena 2003]
- Refactorización  $\cong$   
(Precondición, Acción, Postcondición)



# Contexto Inicial

- Introducción y Objetivos
- Contexto inicial
  - ▶ Trabajos Previos
  - ▶ Problemas
  - ▶ Arquitectura
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

6

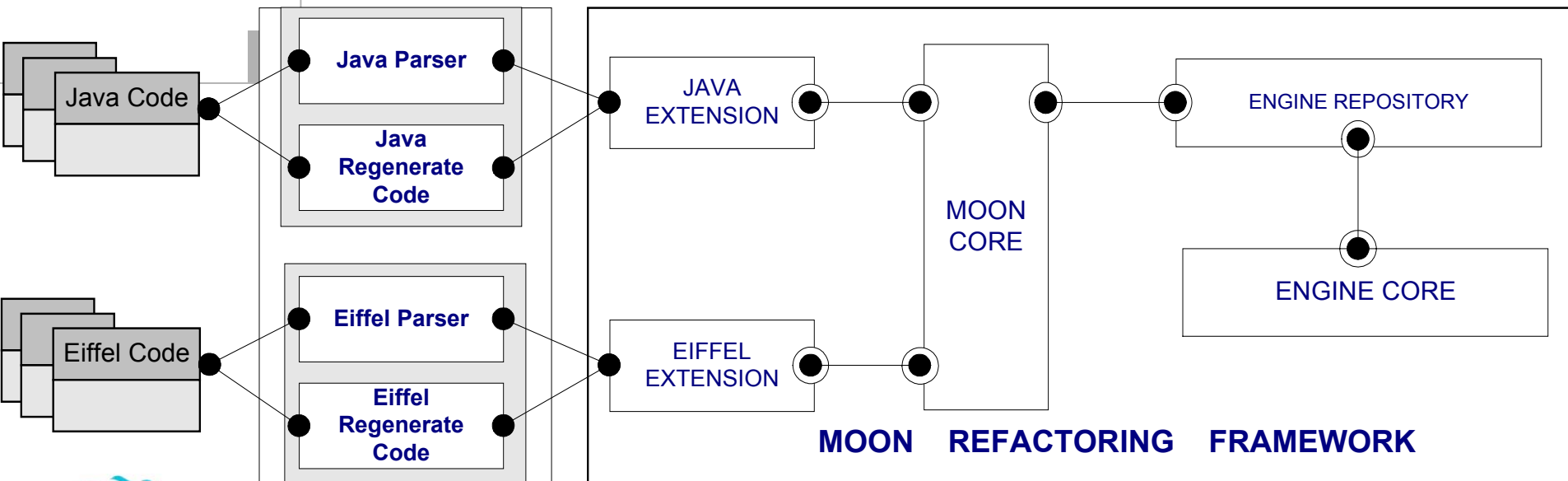
## ■ Problemas por resolver

- 1 Diseño de un soporte de ejecución
- 2 Diseño de un soporte de definición
- 3 Diseño del problema de la regeneración de código

## Contexto Inicial

### ■ Arquitectura propuesta

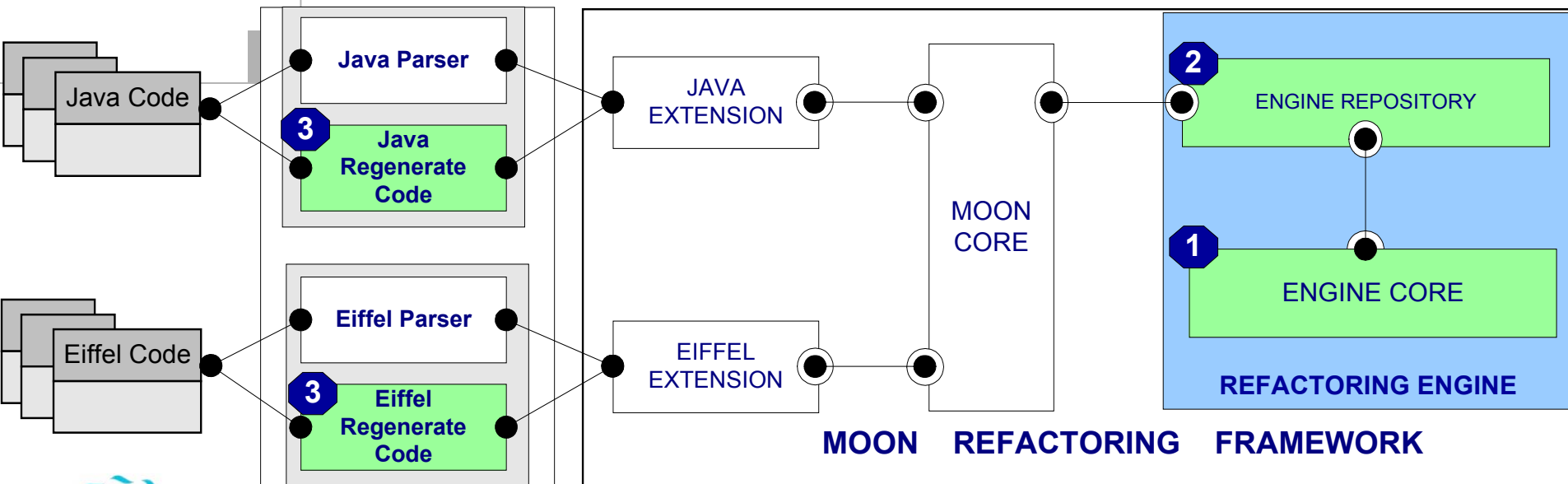
- Introducción y Objetivos
- Contexto inicial
  - ▶ Trabajos Previos
  - ▶ Problemas
  - ▶ Arquitectura
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras



## Contexto Inicial

- Introducción y Objetivos
- Contexto inicial
  - ▶ Trabajos Previos
  - ▶ Problemas
  - ▶ Arquitectura
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

### ■ Arquitectura propuesta





# Motor de Refactorizaciones

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
  - ▶ Núcleo
  - ▶ Extensiones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

9

## ■ Núcleo del motor

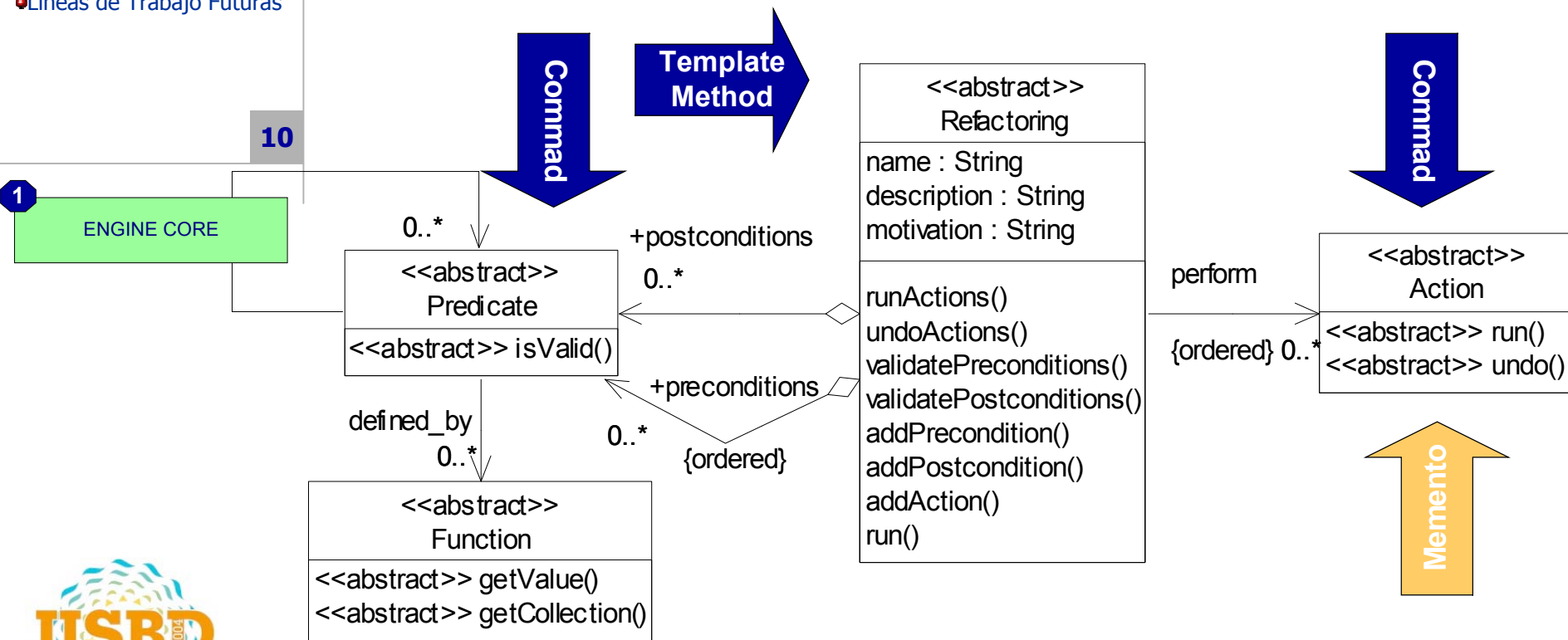
- Soporte de ejecución de refactorizaciones
- Define las abstracciones necesarios
  - Refactorización, Predicados, Funciones y Acciones
- PD Template Method

```
class Refactoring
...
    public void run() {
        validatePreconditions();
        runActions();
        validatePostconditions();
    } ...
```

# Motor de Refactorizaciones

## ■ Diseño del núcleo del motor

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
  - ▶ Núcleo
  - ▶ Extensiones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras



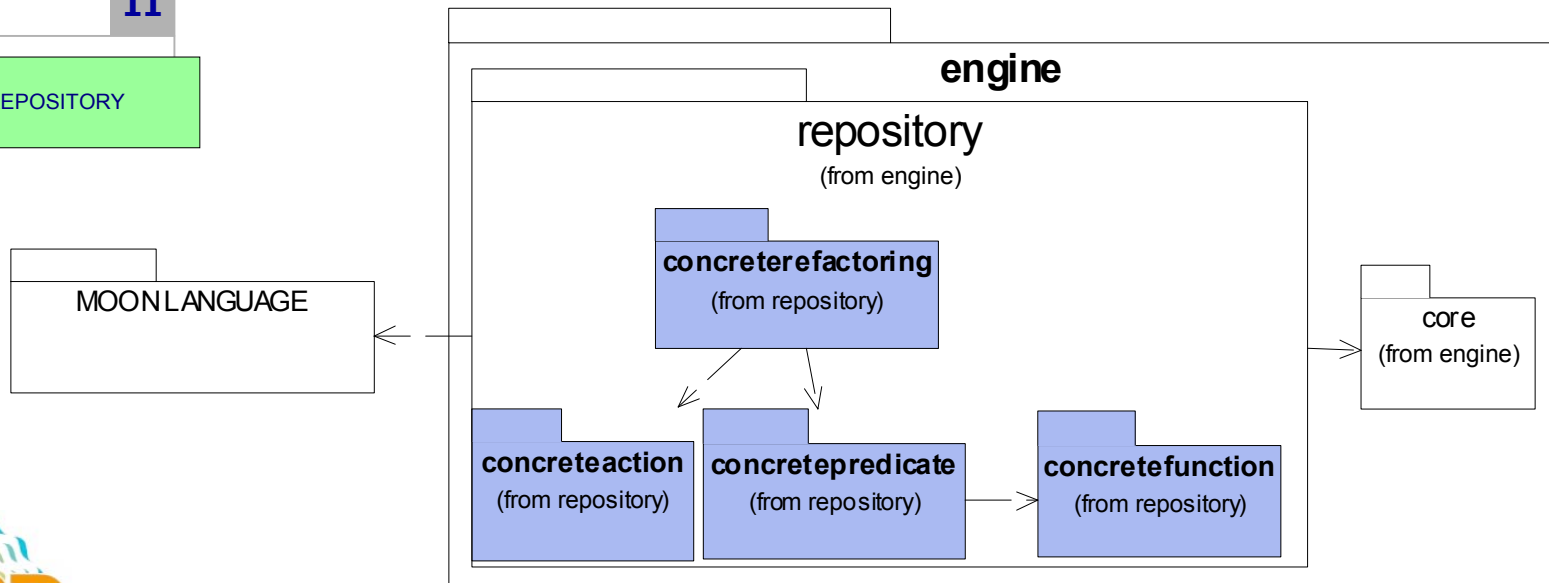
# Motor de Refactorizaciones

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
  - ▶ Núcleo
  - ▶ Extensiones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

11

## ■ Extensiones del motor

- Soporte de definición de refactorizaciones con reutilización
- Repositorios concretos de los elementos del motor



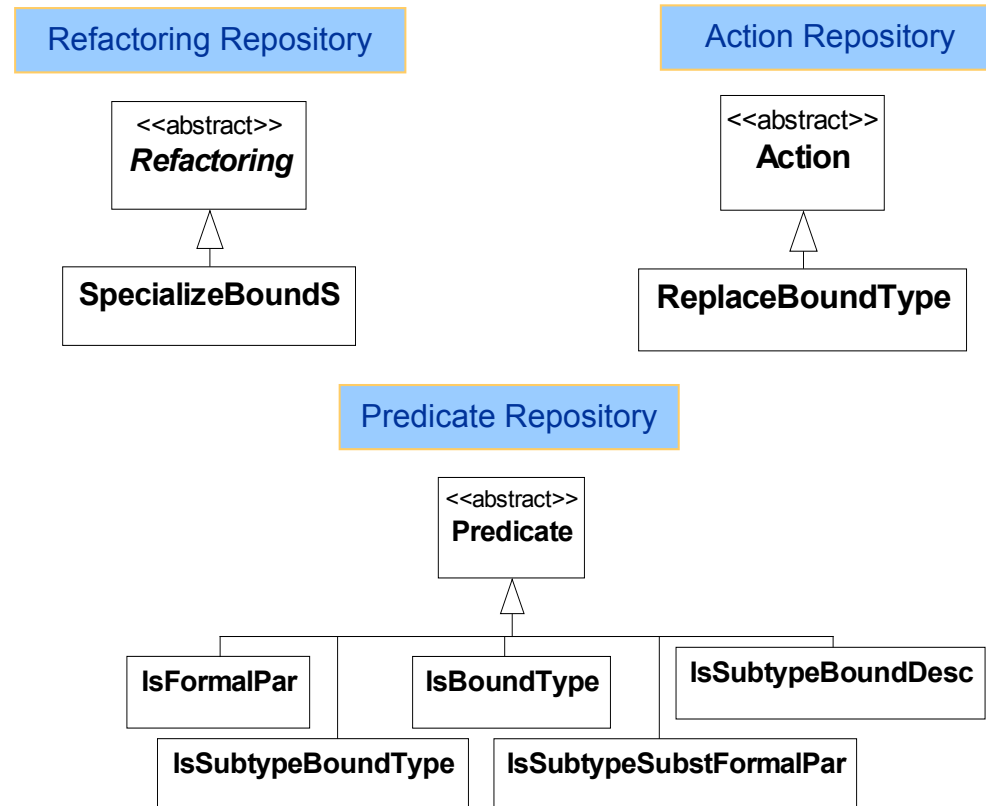
# Reutilización en la Definición de Refactorizaciones

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

12

2  
ENGINE REPOSITORY

## ■ Estado de los repositorios con la definición de la refactorización *SpecializeBounds* [Marticorena 2003]



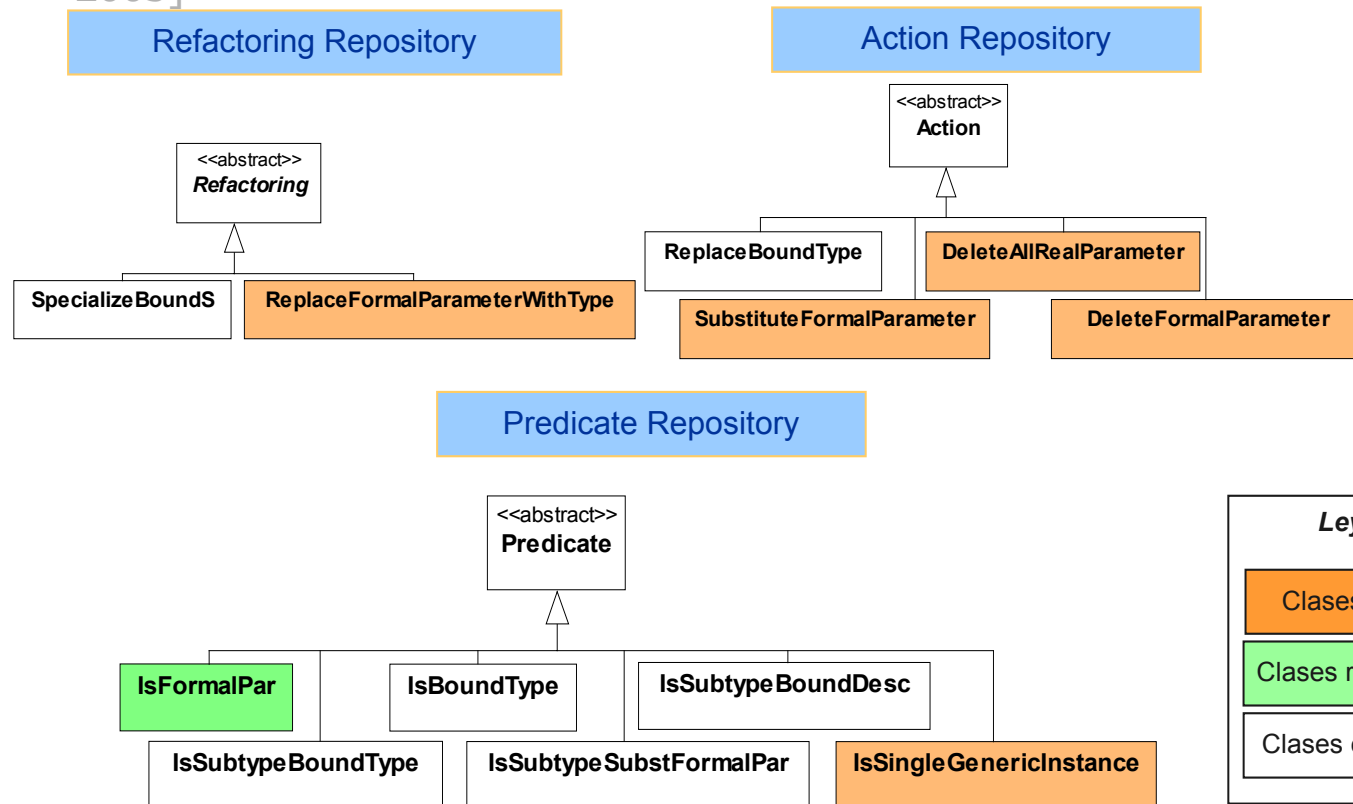
# Reutilización en la Definición de Refactorizaciones

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
- Conclusiones
- Líneas de Trabajo Futuras

13

2  
ENGINE REPOSITORY

## Estado de los repositorios al añadir la definición de *ReplaceFormalParameterWithType* [Marticorena 2003]



### Leyenda

Clases nuevas

Clases reutilizadas

Clases existentes



Noviembre 2004

No se muestra el repositorio de funciones para mejorar enfatizar el aspecto de reutilización en la definición de refactorizaciones

# Regeneración de Código

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración de Código
  - ▶ Problema
  - ▶ Diseño
- Conclusiones
- Líneas de Trabajo Futuras

14

## ■ Definición del problema

### ■ Premisas

- Transformación de código fuente a MOON
- MOON no es completo
- Extensiones de MOON, particularidades del lenguaje

### ■ Clasificación de información

- Común LOO
- Específica de un LOO
- No relevante para refactorizar
  - Localizada en el cuerpo de los métodos
  - Opciones
    - Almacenamiento textual
    - Almacenamiento de AST

3 Language  
Regenerate  
Code



# Regeneración de Código

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración del Código
  - ▶ Problema
  - ▶ Diseño
- Conclusiones
- Líneas de Trabajo Futuras

15

## ■ Diseño

### ■ Trabaja sobre las extensiones de MOON

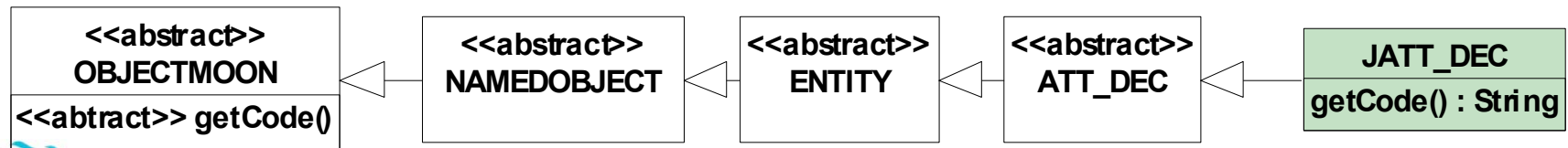
### ■ Alternativa 1

- Distribución de una operación `getCode()` en cada uno de los elementos de la gramática

#### ■ `getCode()`

- encapsula el conocimiento de la gramática del lenguaje concreto
- recorrido de elementos relacionados

#### ■ Ejemplo:



# Regeneración de Código

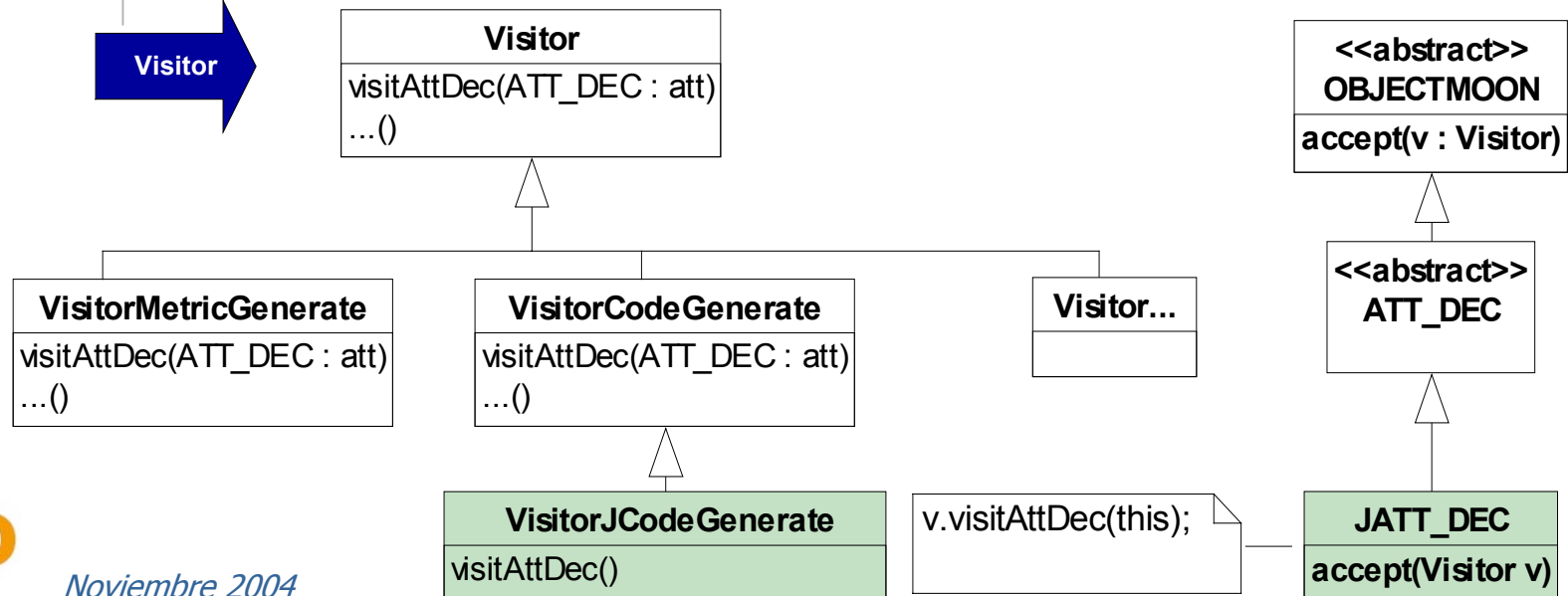
- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración del Código
  - ▶ Problema
  - ▶ Diseño
- Conclusiones
- Líneas de Trabajo Futuras

16

## ■ Diseño

### ■ Alternativa 2

- Encapsular las operaciones de recorrido del árbol sintáctico en la operación `accept()`
- Tratamiento de la operación en clases Visitantes
- PD Visitor





## Conclusiones

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración del Código
- Conclusiones
- Líneas de Trabajo Futuras

17

### ■ Definición de un framework

- Ejecución de refactorizaciones
- Definición de refactorizaciones
  - Con reutilización
  - Para reutilización

Cierta  
independencia  
del Lenguaje

### ■ Análisis y diseño del problema de la regeneración de código



# Líneas de Trabajo Futuras

- Introducción y Objetivos
- Contexto inicial
- Motor de Refactorizaciones
- Reutilización en la Definición
- Regeneración del Código
- Conclusiones
- Líneas de Trabajo Futuras

18

- Estudiar la disminución de tiempo en la definición de nuevas refactorizaciones
- Aumentar el número de extensiones de lenguajes
  - Intermediate Language (IL) .NET
- Refactorizaciones orientadas a tratar la genericidad en los lenguajes
- Objetivo final:

**Herramienta de Refactorización  
Multilenguaje**

