
XVII Jornadas de Ingeniería del Software y Bases de Datos (JISBD)

# Evaluación de la eficiencia de métodos de identificación del defecto de diseño *God Class*

Carlos López, Esperanza Manso, Yania Crespo

**GIRO** Grupo de Investigación en Reutilización y Orientación a Objeto

Universidad de Valladolid y Burgos

clopezno@ubu.es

{manso,yania}@infor.uva.es

# Índice

- Contexto del problema
- Problema
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras

# Contexto del problema


- **Contexto del problema**
- Problema
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras




- Entidades de código
  - Elementos estructurales que componen el código
    - `paquetes, clases, métodos`
- Diseño de entidades de código
  - Se categorizan las entidades
    - Por tipo de responsabilidad, según su naturaleza
  - Existen estereotipos UML estándar con este objetivo
    - `<<Utility>>, <<Entity>>, <<Interfaz>>…`
  - Los diseñadores y programadores utilizan esta clasificación en sus diseños

# Contexto del problema


- **Contexto del problema**
- Problema
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras




- Defecto de diseño
  - ***Describe una situación que sugiere un problema potencial en la estructura del software.***
  - Para decidir si el problema es real o relevante
    - la situación tiene que ser examinada con más detalle en su contexto particular
  - En la literatura se referencia con distintos términos
    - *Code smells*, *Bad smells*, *Disharminies*, *Antipatterns* ...

# Contexto del problema


- **Contexto del problema**
- Problema
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras




- Defecto de diseño - God Class
  - ***En un sistema software orientado a objetos (OO) una God Class es un objeto que controla a demasiados objetos en el sistema y ha crecido más allá de toda lógica para convertirse en la clase que lo hace todo.***
- Excepciones
  - Patrón de diseño mediador
    - *Diseño de interfaces gráficas*
- Herramientas que la identifican

inCode


GIRO



# Problema


- Contexto del problema
- **Problema**
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras




## ¿Cómo identificar entidades con defecto?

- Definición de heurísticas o reglas de detección
  - Basadas en información sobre las entidades
    - Métricas de código conocidas
      - LOC, DIT, NOC, LCOM, **WMC** ...
    - Consultas específicas
      - **ATFD** Access To Foreign Data
      - **TCC** Tight Class Cohesion
  - Heurísticas *God Class* [Marinescu 2006]

```
God Class := ((ATFD > FEW) and
(WMC >= VERY HIGH) and (TCC  < ONE THIRD))
```

# Problema

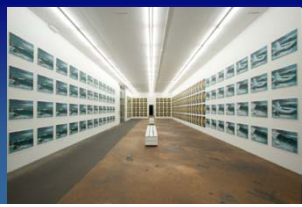

- Contexto del problema
- **Problema**
- Solución propuesta
- Estudio empírico
- Conclusiones y líneas futuras




- Críticas sobre las reglas de detección
  - No son aplicables a todas las entidades del contexto

La información relativa a la naturaleza de la entidad, modelada como estereotipos UML, no se utiliza en la actividad de identificación de defectos de diseño en entidades

# Solución propuesta




- Contexto del problema
- Problema
- **Solución propuesta**
- Estudio empírico
- Conclusiones y líneas futuras




- Realizar un caso de estudio empírico para validar la nueva propuesta
  - Pregunta

    **¿Influye la naturaleza de la entidad, modelada como estereotipo UML, en la predicción de defectos de diseño tipo *GodClass*?**
  - Objetivo GQM
    - **Analizar** entidades de código, **con el propósito** de estudiar cómo impacta el conocimiento de la naturaleza de diseño de la entidad en la detección de defectos de diseño basada en métricas de código, **con respecto a** la eficiencia de la detección, en particular del defecto *God Class*, **desde el punto de vista** de los investigadores, **en el contexto académico** de la Universidad de Burgos (UBU), la Universidad de Valladolid (UVa) y de dos aplicaciones de código abierto JFreeChart 1.0.14 y Eclemma 2.1.0.

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Diseño y variables
  - Variables independientes
    - Medidas de las entidades de código orientado a objeto
      - Chidamber y Kemerer, Lorenz y Kid y otras
    - Naturaleza de la entidad, escala nominal:
      - $e_1$ exception, $e_2$ interface, $e_3$ entity, $e_4$ control, $e_5$ test, $e_6$ utility
    - Coste de falsos negativos, escala nominal:
      - "sin costes", "con costes"
      - Solución para la generación de clasificadores mediante aprendizaje supervisado con validación cruzada en conjuntos de datos desequilibrados
  - Variable dependiente
    - La eficiencia de la predicción binaria del defecto God Class
      - Precisión, Recuperación y F-Measure

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




## Diseño y variables

- Diseño cruzado evalúa la eficiencia de los clasificadores al cruzar las variables naturaleza (n) y costes (n)

| | Con costes | Sin costes |
|---|---|---|
| Con naturaleza | $MedidasRendimiento_{n,c}$ | $MedidasRendimiento_{n,\overline{c}}$ |
| Sin naturaleza | $MedidasRendimiento_{\overline{n},c}$ | $MedidasRendimiento_{\overline{n},\overline{c}}$ |

JFreeChart 1.0.14
y EclEmma 2.1.0

Métricas obtenidas con RefactorIT
Naturaleza de la entidad
Predicción de entidades con defecto
(inicialmente obtenidas InCode JDeodorant)

Considerando costes y naturaleza calcular:
-Heurística obtenidas algoritmo clasificador J48
-Medidas de rendimiento del clasificador:
Precisión, Recuperación y F-Measure

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Objetos y sujetos
  - Sujetos
    - 1 Estudiante de doctorado especializado en el área de mantenimiento del software
  - Objetos
    - 2 Bibliotecas Java: JFreeChart 1.0.14, Eclemma 2.1.0
- Instrumentación
  - Herramienta medición : RefactorIt
  - Clasificador de entidades en estereotipos UML...
  - Predicción de *God Class*: Incode 2.07 y JDeodorant 4.0.12
  - Generación de clasificadores y medidas Weka 3.7.5

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Instrumentación – Clasificador entidades en estereotipos UML
  - **Tabla de Convención de Nombres**

| Categorías | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ |
|---|---|---|---|---|---|---|
| Convención de nombres | exception | interface<br>gui<br>forms<br>ui<br>report<br>swing<br>visual<br>view<br>awt | core<br>model<br>entity | control<br>facade<br>manager<br>handler<br>action<br>callback<br>maker<br>provider | test<br>debug<br>dummy | util<br>properties<br>log<br>preference<br>template<br>options |

  - Entradas y salidas esperadas del algoritmo

```
com.mountains.eclemma.core.test      e5 (conflicto)
ui.actions.MergeSessionAction        e4 (conflicto)
core.analysis.JavaElementCoverage    e3
core.SessionManager                  e4 (conflicto)
ui.coverageview.CoverageView         e2
```

# Estudio empírico Análisis


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras


- Análisis de resultados
  - Con o Sin Costes $(c, \overline{c})$
  - Con o Sin Naturaleza $(n, \overline{n})$



| | Precisión | Recuperación | F-Measure |
|---|---|---|---|
| $\overline{c}$: $n$ vs. $\overline{n}$ | Mejora en ambos casos | Mejora en ambos casos | Mejora en ambos casos |
| $\overline{n}$: $c$ vs $\overline{c}$ | Mejoran en ambos casos | Mejora en ambos casos | Mejora en ambos casos |
| $n$: $c$ vs $\overline{c}$ | JFreeChart mejora | JFreeChart mejora | JFreeChart mejora |
| $c$: $n$ vs. $\overline{n}$ | Mejora en ambos casos | Mejora en ambos casos | Mejora en ambos casos |

# Estudio empírico Análisis


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Análisis de comparación de dos métodos de identificación *God Class*
  - No concordancia
  - Subjetividad en la clasificación

inCode

JDeodorant when quality matters the most...

| | Nº total de entidades | Nºentidades con defectos identificadas InCode | Nºentidades con defectos identificadas JDeodorant | Nºentidades con defectos identificadas JDeodorant e InCode |
|---|---|---|---|---|
| JFreechart | 975 | 67 | 0 | 67 |
| eclemma | 152 | 0 | 6 | 6 |

# Estudio empírico Validez


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Validez de construcción
  - La clasificación de entidades en estereotipos UML **no es ortogonal**
- Validez interna
  - **Subjetividad** de la clasificación de entidades en estereotipos UML
- Validez externa
  - Proyecto elegidos y lenguaje Java
  - Matriz de costes elegida
  - Medidas de rendimiento dependen del algoritmo (J48)

# Conclusiones y líneas futuras


- Contexto del problema
- Problema
- Solución propuesta
- Estudio empírico
- **Conclusiones y líneas futuras**




- En el estudio de la eficiencia de los clasificadores para identificar el defecto *God Class*, los resultados observados indican que la naturaleza de diseño de la entidad mejora su eficiencia
- Además, la matriz de costes, cuando se penaliza el coste de los falsos negativos, también mejora dichas medidas
- En el estudio de comparación de los dos métodos de identificación del defecto *God Class*, los resultados indican que no existe concordancia

# Conclusiones y líneas futuras


- Contexto del problema
- Problema
- Solución propuesta
- Estudio empírico
- **Conclusiones y líneas futuras**




- Añadir la supervisión humana para validar los resultados de los métodos eliminando los falsos positivos en función de la excepciones documentadas
- Más réplicas
  - Nuevos objetos ...
  - Nuevos defectos: *Data Class*, *Feature Envy*
- Estudiar posibles refinamientos del algoritmo de clasificación en estereotipos
  - Clasificar por uso de biblioteca

# Gracias por su atención

Carlos López, Esperanza Manso, Yania Crespo

GIRO Grupo de Investigación en Reutilización y Orientación a Objeto

Universidad de Valladolid y Burgos

clopezno@ubu.es

{manso,yania}@infor.uva.es

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Instrumentación
  - *RefactorIt*
    - Herramienta de cálculo de medidas
  - Clasificación de medidas por característica
    - tamaño (**TAM**)
    - documentación (**DOC**)
    - acoplamiento (**ACO**)
    - herencia (**HER**)
    - complejidad estructural (**COM**)
    - Abstracción (**ABS**)
    - Cohesión (**COH**)
    - Principios de diseño (**PDA**)
  - Clasificación por ámbito
    - **P** paquete
    - **C** Clase
    - **M** Método

| Descripción | Identificador | MinValor | MaxValor | Ámbito | Característica |
|---|---|---|---|---|---|
| Comment Lines of Code | CLOC | | | T | TAM |
| Cyclomatic Complexity | V(G) | 1 | 10 | M | COM |
| Density of Comments | DC | 0.2 | 0.4 | T | DOC |
| Executable Statements | EXEC | 0 | 20 | T | TAM |
| Non-Comment Lines of Code | NLOC | | | T | TAM |
| Number of Parameters | NP | 0 | 4 | M | TAM |
| Total Lines of Code | LOC | 5 | 1000 | T | TAM |
| Abstractness | A | 0.0 | 0.5 | P | ABS |
| Afferent Coupling | Ca | 0 | 500 | P | ACO |
| Depth in Tree | DIT | 0 | 5 | C | HER |
| Efferent Coupling | Ce | 0 | 20 | P | ACO |
| Instability | I | 0.7 | 1.0 | P | ACO |
| Number of Abstract Types | NOTa | 0 | 20 | P | ABS |
| Number of Children | NOC | 0 | 10 | C | HER |
| Number of Concrete Types | NOTc | 0 | 80 | P | ABS |
| Number of Exported Types | NOTe | 3 | 50 | P | ACO |
| Number of Fields | NOF | 0 | 1 | C | TAM |
| Number of Types | NOT | 0 | 80 | P | TAM |
| Response for Class | RFC | 0 | 50 | C | COM |
| Weighted Methods per Class | WMC | 1 | 50 | C | COM |
| Number of Attributes | NOA | 0 | 5 | C | TAM |
| Cyclic Dependencies | CYC | 0 | 1 | P | PDA |
| Dependency Inversion Principle | DIP | 0.3 | 1.0 | C | PDA |
| Direct Cyclic Dependencies | DCYC | 0 | 1 | P | PDA |
| Distance from the Main Sequence | D | 0.0 | 0.1 | P | PDA |
| Encapsulation Principle | EP | 0 | 0.6 | P | PDA |
| Lack of Cohesion of Methods | LCOM | 0.0 | 0.2 | C | COH |
| Limited Size Principle | LSP | 0 | 10 | P | PDA |
| Modularization Quality | MQ | 0 | 1000 | P | PDA |
| Number of Tramps | NT | 0 | 1 | M | O |

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y líneas futuras




- Instrumentación
  - Algoritmo de clasificación que se basa en criterios de convención de nombres de las entidades
    - Precondición: Se dispone de una **Tabla de Convención de Nombres**
  1. Obtener el *nombre simple* y *nombre cualificado* de la entidad de código
  2. Buscar si el *nombre* esta contenido en la **Tabla de Convención de Nombres**
     - Asocia cadenas de caracteres a categorías de entidades
     - Ejemplo: “Action” => e4 Entidad de control
  3. En el caso de **conflicto**, porque el nombre contenga cadenas que correspondan a más de una categoría
     - Prevalece el criterio del nombre simple de la entidad

# Estudio empírico Planificación


- Contexto del problema
- Problema
- Solución propuesta
- **Estudio empírico**
- Conclusiones y línea futuras


## Medidas de rendimiento

| | | Clase predicha | |
|---|---|---|---|
| | | si | no |
| **Clase real** | si | Positivos ciertos (TP) | Falsos negativos (FN) |
| | no | Falsos positivos (FP) | Negativos ciertos (TN) |



Las medidas de eficiencia propuestas se definen así:

$$Precision = \frac{TP}{TP+FP} \qquad Recuperacion = RatiodeTP = \frac{TP}{TP+FN}$$

A partir de ellas se puede calcular una única medida conocida como F-measure (media harmónica o media de ratios), y se define como:

$$F - measure = \frac{2*Recuperacion*Precision}{Recuperacion+Precision} = \frac{2*TP}{2*TP+FP+FN}$$