



# Laboratory of Software Analysis

## *Exam*

Filippo Ricca

ITC-Irst

Istituto per la ricerca  
Scientifica e Tecnologica

ricca@itc.it



# Project Steps: JConsole



- 1. Reverse engineering. Collect some views (Class Diagram)**
- 2. Instrumentation of the JAVA code using TXL and writing test cases with statement/branch coverage.**
- 3. Change requirement - add a new crosscutting feature to Jconsole**
- 4. Extend the JAVA grammar for the ASPECTJ language**
- 5. Instrumentation of the ASPECTJ code using TXL and writing test cases with statement/branch coverage.**
- 6. Regression testing (no side effects)**
- 7. Testcases JUNIT (*no mandatory!*)**



---

# Esame

---

→ *Occorre mandare via mail, almeno due giorni prima dell'esame, una directory che contiene 7 directory: **Step1, Step2, Step3, Step4, Step5, Step6, Step7** che a loro volta contengono:*



---

# Step 1

---

## **2) Fase di reverse engineering. Ricavare il diagramma delle classi**

- a) README
- b) Diagramma delle classi.
- c) Trasformazioni TXL che lo calcolano.



---

## Step 2

---

- 1) **Instrumentare la JConsole usando TXL. Scrivere Testcases t.c. vi sia la copertura (100%?) dei branch.**
  - a) README
  - b) JConsole instrumentato.
  - c) Trasformazioni TXL per instrumentare il codice.  
(commentate ...)
  - d) Testcases scelti e percentuale di copertura raggiunta  
(spiegare perché non è possibile arrivare al 100% ...).
  - e) Infrastruttura software utile ad eseguire i testcases (spiegare).



---

# Step 3

---

## **3) Change requirement - add a new crosscutting feature to Jconsole**

- a) README
- b) codice aspectJ che implementa il comando persistent\_history (commentato ...)



---

# Step 4

---

## **4) Extend the JAVA grammar for the ASPECTJ language**

- a) README: spiegare quali produzioni sono state inserite e quali modificate.



---

# Step 5

---

## **5) Instrumentation of the ASPECTJ code using TXL and writing test cases with statement/branch coverage.**

- a) README
- b) JConsole “new” instrumentato.
- c) Trasformazioni TXL per instrumentare il codice.
- d) Testcases scelti e percentuale di copertura raggiunta (spiegare perché non è possibile arrivare al 100% ...).
- e) Infrastruttura software utile ad eseguire i testcases.



---

# Step 6

---

- 6) Regression testing con i Testcases generati al punto 2.**
- a) README
  - b) Prove di esecuzione che dimostrano che i due programmi (Java e AspectJ) si comportano nello stesso modo con i testcases generati al punto 2 (output dei due programmi, risultato del diff. ed eventuali spiegazioni).



---

# Step 7

---

## 7) Testcases con Junit (facoltativo).

- a) README
- b) L'idea e' quella di riscrivere i testcases usando il tool JUNIT e mandarli in esecuzione. Scopo: facilitare il regression testing.