ROADBLOCK 1

Dato che sicuramente non lo avete fatto, prendetevi il vostro tempo per rifattorizzare il vostro codice. Il passaggio allo step successivo, questa volta, richiede che la qualità del vostro codice migliori, per rendere “sostenibile” la prosecuzione del lavoro.

IN PARTICOLARE TENETE D’OCCHIO QUELLA CHE VIENE CHIAMATA COMPLESSITA’ CICLOMATICA, GENERATA DA TROPPE DIRAMAZIONI (IF o SWITCH) O CICLI ANNIDATI: IN QUESTO CONTESTO SIGNIFICA:

**NON POTETE PROSEGUIRE SE AVETE PIU’ DI 5 IF (O CASI DI UNO SWITCH O ASSEGNAZIONI SIMILI RIPETUTE)**

Altre a questo, vi elenchiamo una breve “Checklist” delle cose da controllare, in ordine di importanza. Quando avete finito, presentate al facilitatore TUTTE E DUE le versioni e mostrate in che modo avete migliorato il codice. Sarà il facilitatore a stabilire se avete fatto un refactoring efficace oppure si è trattato SOLO di tempo perso.

IMPORTANTE: Tutti i test finora effettuati devono restare VERDI dopo ogni refactoring.

* **Intimità inappropriata**: dati indipendenti sono trattati in modo unitario. RICERCA: variabili che gestiscono troppe informazioni: SOLUZIONE: separare i dati in variabili e/o classi e/o metodi indipendenti.
* **Complessità ciclomatica:** Troppe diramazioni (if/switch) o cicli annidati RICERCA: In questo contesto, “troppi” significa “più di 4”. SOLUZIONE: la funzione deve essere spezzata in funzioni più piccole, oppure deve essere semplificata o generalizzata.
* **Metodo troppo lungo**: RICERCA: Tutti i metodi che abbiano più di 10 istruzioni. SOLUZIONE: metodo privato.
* **Codice duplicato**, uguale o pressoché uguale, in diverse sezioni di codice (viola il principio Don't Repeat Yourself). RICERCA: I vostri occhi. SOLUZIONE: metodo privato (ev. statico) . TOOL: Code → Refactor → Introduce Method:
* **Commenti inadeguati**: commenti che spiegano cosa fa una certa porzione di codice (sintomo che il codice non è sufficientemente chiaro di per sé); si noti la differenza tra “commenti” e “documentazione “ (eg. Javadoc). RICERCA: i vostri occhi. SOLUZIONE: modificare il codice in modo da rendere inutile il commento.
* **Nomi oscuri**: nomi e identificatori (di variabili, attributi, classi, ...) che non chiariscono il significato inteso dell'entità corrispondente. ESEMPIO: variabili a, b, c, x, kappa, xxx4; TOOL: Code → Refactor → Rename.