Name: Code Counter™ Enhancements EMS

Presenter(s): Yue Chen

Objectives: Use directed research students to enhance the code counter family of the tools and in the process develop and use an experience management system so that the future students can make more progress faster. The pool of student can also be used for the software engineering experiments like pair programming and pair testing.

Rationale: Code Counter™ is a set of automated tools to count Source Lines Of Code (SLOC). The counters reduce the time and effort needed to effectively estimate future implementations by means of providing accurate statistics of the current source programs. The counting results provide concrete statistical basis for project effort estimating and thus greatly contribute to the project scheduling and cost estimation.

Target Users: Researchers and students doing the enhancement activity

Scope: An Experience Management System (EMS) for a family of software code counting tools.

Project Type: Directed research project.

Runs On: Hyperwave

IPR Status: Code Counter™ and enhancement are copy-left and administrated by the Center of Software Engineering at USC. The Hyperwave based experiments will be available to CeBASE collaborators.

Technical Approach:

1. The enhancement follow the “Learn by doing” approach documented by Winsor Brown. (For more detail about “Learn by doing” please refer to next page.) As part of the agile method research at CSE, our 14-person-directed-research team will be subject for the “pair testing” experiment approach and aim at contributing a set of project process, experience and effort data.

2. For the development of the EMS itself, it uses Hyperwave for information management, adopts a bottom up process to gather the desired information and applies/modifies existing CSE and other tools for gathering effort, defects, progress status, and lessons learned data.

3. On-going activities that can be used for other researches. E.g. pair programing/testing, etc.

Developer: Yue Chen


Future Directions: Introduce more agile methods into the project process. Generate a set of counters for new languages (e.g. C# or VB).
Code Counter Enhancement

Objectives: 1) Improved packages (adding acceptance tests and/or regression tests); 2) add Object Lines of Code counting to the improved package; 3) generate a counter for a new language.

Also the PSP with learning data gathering instruments for effort and defects, including gross estimates and results tracking.

I. Learn to use: Use the C/C++ counter to count
   A. C/C++ counter
   B. the language counter that is to be enhanced

II. Learn by testing (and fixing if needed)
   A. Generate a counting standard to PSP’s "LOC Counting Standard Template" (page 76 of a Discipline for Software Engineering).
   B. Acceptance Tests
   C. Regression Test set
   D. If mistakes are found while doing B or C immediately above, record defect, fix it, record effort
   E. Package up the tests and results

III. Learn by enhancing an existing product: Add (or verify) the object lines of code counting capability via instrumented comments.
    A. Analyze and document the changes in the OCD, SSRD, and SSAD; Update II.A
    B. Enhance the acceptance and regression tests (II.B and II.C)
    C. Code the changes
    D. Run the enhanced acceptance and regression tests, and debug as needed
    E. Repeat II.E

IV. Learn by developing a new one
    A. Determine the language type characteristics, and pick the closest match.
    B. Compile and run the regression tests for that closest match.
    C. Identify the language specific changes that must be made and change the OCD, SSRD and SSAD correspondingly.
    D. Develop and document the acceptance tests (II.B, above)
    E. Change the code
    F. Run the enhanced acceptance, and fix (debug) code as necessary
    G. Design and document regression tests (II.C, above)
    H. Run enhanced acceptance and regression tests, and debug as needed
    I. Repeat II.E