Essential Metrics Project Manager (EPM)

Changed, Added and Deleted SLOC

A Simple Example

Contents

Document

Purpose

This document describes the way that Essential Metrics PM calculates Changed, Added and Deleted Source Lines of Code metrics.

Readership

This User Guide is intended for management, technical and non-technical users.

Version History

1	28-Jan-2006	Document created.

Introduction

An important consideration for Software Project Managers is whether the requirements are being altered in a given project without their knowledge or consent. The impact of unauthorised or unscheduled departure from requirements can lead to considerable problems in maintaining and managing large source projects.

Having received requests from several major projects to measure ‘changes' in source code projects, Power Software was commissioned to manufacture an automated tool to measure changed lines of source code.

The first question we asked was 'are there any existing tools that can already accomplish this task?' The obvious starting place was 'Change Management' tools like CMVC (Configuration Management Version Control) systems. These tools indicate which files have changed between projects but do not gather and present the information in the format our customers’ required. One problem with CMVC systems was that a file shown to be ‘changed’ might have changed only by time and date. This is misleading if we are looking for actual changes in source code.

The next place we looked was at existing Software Metrics tools. These tools measure source code parameters including SLOC (source lines of code). However it is possible that a Software Metrics tool could report that the compared files or projects have the same total number of source lines but these could have changed intrinsically. The actual lines of code could have changed but the total number of lines will be unchanged so the 'changed line' is not detected.

For our customers the particular information required is 'changed source lines of code', 'added source lines of code' and 'deleted source lines of code'. This needs to be gathered across two releases of the same project to show exactly what has 'changed'. The information needs to be presented as 'changed lines' at the file and project level to assist management in their job of maintaining highest quality through proper control mechanisms.

The following example shows how Essential Project Manager meets these requirements.

Example

Figure 1 – Old and New File

Old File

```
void functionA(int k) {
  cout << "param: " << k << endl;
}

void functionB() {
  //Unused function
  int j=0;
}

int main(int argc, char **argv) {
  int i;
  int j=100;
  int count=0;

  for (i=0; i<10; i++) {
    // Calling functionA()
    functionA(i);
    while (j>0) {
      j-= 5;
    }
  }
}
```

New File

```
void functionA(int k) {
  cout << "param: " << k << endl;
}

int main(int argc, char **argv) {
  int i;
  int j=100;
  int count=0;

  for (i=0; i<10; i++) {
    // Calling functionA()
    functionA(i);
    while (j>0) {
      j-= 10;
    }

    while(count<10) {
      count++;
    }
  }
}
```

Color Key
Added Source Line
Changed Source Line
Deleted Source Line

In the above example, since the absolute or total value for Lines of Code and Source Lines of Code has not changed many tools would not show changes for this file. However, EPM compares the two files (using a standard[i] comparison algorithm), and gives values for changes/additions and deletions with respect to the actual Source Lines of Code. *This indicates the reality of the activity within the source file or project.*

File Level

At the file level the metrics produced from the file comparison are as follows:

ADD_SLOC

From the example, in the new file these lines have been added:

```
while(count<10) {
  count++;
}
```

for which Krakatau PM would give **ADD_SLOC = 3**.

DEL_SLOC

From the example, the following lines existed in the old file but were no longer present in the new file:

```
void functionB() {
  //Unused function
  int j=0;
}
```

This would give **DEL_SLOC = 3**. Since comments are not treated as Source lines of code they are not included in the values.

CHG_SLOC

From the example only one line changed:

Figure 2 – Changed Line

Old File	New File

`while (j>0) {`
`  j-= 5;`
`}`
	`while (j>0) {`
	`  j-= 10;`
	`}`

This would be counted as **CHG_SLOC = 1**.

The following HTML report shows the changes between the files produced from the source code used in the example above:

Figure 3 - File Changes HTML Report

test.cpp (C++)

Metric	Old	New	Diff
LOC	23	22	-1
SLOC	17	17	
NSC	9	9	
N1	25	25	
N2	27	27	
n1	12	12	
n2	15	13	-2
N	52	52	
n	27	25	-2
V	247	241	-6
D	10	12	+2
E	2470	2892	+422
B	0	0	
J_COM	0	0	
C_COM	0	0	
CPP_COM	2	1	-1
COM_LOC	2	1	-1
BYTES	321	312	-9

Changed Metrics

Metric	Value
CHG_SLOC	1
DEL_SLOC	3
ADD_SLOC	3

The values for CHG_SLOC (1), ADD_SLOC (3), and DEL_SLOC (3) as described above can be seen in the bottom “Changed Metrics” section.

The top section of this report shows the old and new metric values, then the differences between the metrics between the two files. In this example it can be seen that the value for CPP_COM (C++ style comments) has decreased by 1. This is explained because (as previously shown) the comment was removed in functionB.

As explained above the comment is not included in the DEL_SLOC value as it is not a source line of code, however, the Difference section of the report highlights that the number of comments has gone down.

In other words the Changed Metrics section shows the pertinent real value changes to the code, but the detail can still be determined from the top section.

Project Level

Essential Project Manager provides a summary of these metrics, including changes, at the project level.

Figure 4 - Project Changes HTML Report

It can be seen (fig. 4) that at the project level again both Difference (top section) and Change (bottom section) metrics are produced.

If we firstly consider the "Changed metrics", Changed, Added and Deleted **SLOC** are present – these values are obtained from the sum of their corresponding metrics at the file level.

As well as these, the project level "Changed Metrics" shows how many **files** have been Changed, Added and Deleted.

This gives a quick, high-level view of what has changed between two versions of a project.

The top section shows the metrics for the old and the new projects, and the differences. This again gives an indication at the project level of what is changing.

Project Metrics for Test_New (28/01/2006)

Metric	Old	New	Diff
LOC	23	22	-1
SLOC	17	17	
NSC	9	9	
N1	25	25	
N2	27	27	
n1	12	12	
n2	15	13	-2
N	52	52	
n	27	25	-2
V	247	241	-6
D	10	12	+2
E (k)	2	2	
B	0	0	+0
J_COM	0	0	
C_COM	0	0	
CPP_COM	2	1	-1
COM_LOC	2	1	-1
BYTES	321	312	-9
NFILE	1	1	

Changed Metrics

Metric	SLOC	Metric	FILE
CHG_SLOC	1	CHG_FILE	1
DEL_SLOC	3	DEL_FILE	0
ADD_SLOC	3	ADD_FILE	0

[i] "An O(ND) Difference Algorithm and its Variations", Eugene W. Myers, Algorithmica Vol. 1 No. 2, 1986, p 251.