Visual Basic CodeCount™

Counting Standard

University of Southern California

Center for Systems and Software Engineering

October , 2007

Revision Sheet

Date	Version	Revision Description	Author
10/17/2007	1.0	Original Release	CSSE
1/2/2013	1.1	Updated document template	CSSE
1/14/2013	1.2	Added cyclomatic complexity	CSSE

Table of Contents

1. Definitions

1.1. **SLOC –** Source Lines of Code is a unit used to measure the size of software program. SLOC counts the program source code based on a certain set of rules. SLOC is a key input for estimating project effort and is also used to calculate productivity and other measurements.

1.2. **Physical SLOC –** One physical SLOC is corresponding to one line starting with the first character and ending by a carriage return or an end-of-file marker of the same line, and which excludes the blank and comment line.

1.3. **Logical SLOC –** Lines of code intended to measure "statements", which normally terminate by a semicolon (C/C++, Java, C#) or a carriage return (VB, Assembly), etc. Logical SLOC are not sensitive to format and style conventions, but they are language-dependent.

1.4. **Data declaration line or data line –** A line that contains declaration of data and used by an assembler or compiler to interpret other elements of the program.

The following table lists the Visual Basic keywords that denote data declaration lines:

Class	Const	Declare	Delegate
Dim	Enum	Event	Function
Interface	Module	Operator	Property
Structure	Sub		

Table 1 Data Declaration Types

1.5. **Compiler Directives –** A statement that tells the compiler how to compile a program, but not what to compile.

The following table lists the Visual Basic keywords that denote compiler directive lines:

Define a compiler constant #Const
Compile selected blocks of code: #If…Then…#Else
Collapse and hide sections of code: #Region
Indicate a mapping between source lines and external text: #ExternalSource

Table 2 Compiler Directives

1.6. **Blank Line –** A physical line of code, which contains any number of white space characters (spaces, tabs, form feed, carriage return, line feed, or their derivatives).

1.7. **Comment Line –** A comment is defined as a string of zero or more characters that follow language-specific comment delimiter.

Visual Basic comment delimiter is "'". A whole comment line may span one line and does not contain any compilable source code. An embedded comment can co-exist with compilable source code on the same physical line. Banners and empty comments are treated as types of comments.

1.8. **Executable Line of code –** A line that contains software instruction executed during runtime and on which a breakpoint can be set in a debugging tool. An instruction can be stated in a simple or compound form.

- An executable line of code may contain the following program control statements:
  - Selection statements (If, Select)
  - Iteration statements (For, While, Until)
  - Jump statements (Break, Exit)
  - Expression statements (procedure/function calls, assignment statements, operations, etc.)
- An executable line of code may not contain the following statements:
  - Compiler directives
  - Data declaration (data) lines
  - Whole line comments, including empty comments and banners
  - Blank lines

2. Checklist for source statement counts

PHYSICAL SLOC COUNTING RULES			
MEASUREMENT UNIT	ORDER OF PRECEDENCE	PHYSICAL SLOC	COMMENTS
Executable Lines	1	One Per line	Defined in 1.8
Non-executable Lines			
Declaration (Data) lines	2	One per line	Defined in 1.4
Compiler Directives	3	One per line	Defined in 1.5
Comments			Defined in 1.7
On their own lines	4	Not Included (NI)	
Embedded	5	NI	
Banners	6	NI	
Empty Comments	7	NI	
Blank Lines	8	NI	Defined in 1.6

LOGICAL SLOC COUNTING RULES				
NO.	STRUCTURE	ORDER OF PRECEDENCE	LOGICAL SLOC RULES	COMMENTS
R01	If/Elseif condition Then statement Else statement Endif Select var Case cond:statement . . Case Else :statement End Select	1	Count Once	

R02	*do while* (…) statement
	loop
	for (….)
	statement
	next
	do
	statements
	until(…..)
	while(…..)
	statements
	wend
		2	Count Once	

R03	Block delimiters Private Sub End Sub	3	Count once per pair of Private Subband End Sub	
R04	Compiler directive	5	Count once per directive	

3. Examples

EXECUTABLE LINES

SELECTION Statement

ESS1 – If, Else If, Else and nested If statements

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT

If *condition*Then
code to be executed if condition is true
End If
	If (total = firstnum + secondnum And
	Val(sum.Text) <> 0) Then
	correct.Visible = True
	wrong.Visible = False
	End If
		1
		
		1
		1
		0

If *condition*Then
code to be executed if condition is true
Else
code to be executed if condition is false
End If
	If (total = firstnum + secondnum And
	Val(sum.Text) <> 0) Then
	correct.Visible = True
	wrong.Visible = False
	Else
	correct.Visible = False
	wrong.Visible = True
	End If
		1
		
		1
		1
		0
		1
		1
		0

If *condition1*Then
code to be executed if condition1 is true
Else If *condition2* Then
code to be executed if condition2 is true
Else
code to be executed if condition is false
End If
 	
NOTE: complexity is not considered, i.e. multiple “And” or “Or” as part of the expression.
	If (total = firstnum + secondnum And
	Val(sum.Text) <> 0) Then
	correct.Visible = True
	wrong.Visible = False
	Else If (total = firstnum – secondnum) Then
	correct.Visible = False
	wrong.Visible = True
	Else
	correct.Visible = False
	wrong.Visible = True
	End If
		1
		
		1
		1
		1
		1
		1
		0
		1
		1
		0

ESS2 – Select Case

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Select Case Case <constant 1> : <statements> Case Else <statements> End Select	Select Case Err.Num Case 53 'File not found answer=MsgBox("File not found. Try again?", _ vbYesNo) Case 76 'Path not found answer=MsgBox("Path not found. Try again?", _vbYesNo) Case Else 'unknown error MsgBox "Unknown error. Quitting now." 'SHOULD LOG ERROR! Unload Me End Select	1 0 1 0 1 0 1 1 0

ESS3 – error handler

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
OnError Goto	Private Sub CodeWithErrorHandler() On Error GoTo ErrHandler '...Procedure code ... '...	1 1 1 0

ITERATION Statement

EIS1 – For

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
For num = 1 To 10 STATEMENTS Next	For num = 1 To 10 studentName(num)= 999 Next	1 1 0

EIS2 – While

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
While condition Statements Wend	While Not IsEmpty(ActiveCell) MsgBox ActiveCell.Value ActiveCell.Offset(1, 0).Select Wend	1 1 1 1 0

EIS3 – Do-Until

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Do Statements Until condition	Do MsgBox ActiveCell.Value ActiveCell.Offset(1, 0).Select Until IsEmpty(ActiveCell)	0 1 1 1

EIS4 – Do-While

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Do While condition Statements Loop	Do While counter <=1000 num.Text = counter counter = counter+1 Loop	1 1 1 0

JUMP Statement

EJS1 – Exit

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Exit Sub/Function	Exit Sub	1

EXPRESSION Statement

EES1 – function call

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<function_name> (<parameters>)	read_file (name)	1

EES2 – assignment statement

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
<name> = <value>	x = y	1

DECLARATION OR DATA LINES

DDL1 – subroutine/function declaration, variable declaration

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
Private Sub Name(var_list) Statements End Sub	Private Sub Start_Click() Form1.Cls addName End Sub	1 1 1 0
Dim <var> As <type>	Dim var As String	1

COMPILER DIRECTIVES

CDL1 – directive types

GENERAL EXAMPLE	SPECIFIC EXAMPLE	SLOC COUNT
#Const Directive	#Const Directive	1

4. Cyclomatic Complexity

Cyclomatic complexity measures the number of linearly independent paths through a program. It is measured for each function, procedure, or method according to each specific program language. This metric indicates the risk of program complexity and also determines the number of independent test required to verify program coverage.

The cyclomatic complexity is computed by counting the number of decisions plus one for the linear path. Decisions are determined by the number of conditional statements in a function. A function without any decisions would have a cyclomatic complexity of one. Each decision such as an If condition or a For loop adds one to the cyclomatic complexity.

The cyclomatic complexity metric v(G) was defined by Thomas McCabe. Several variations are commonly used but are not included in the UCC. The modified cyclomatic complexity counts select blocks as a single decision rather than counting each case. The strict or extended cyclomatic complexity includes boolean operators within conditional statements as additional decisions.

Cyclomatic Complexity	Risk Evaluation
1-10	A simple program, without much risk
11-20	More complex, moderate risk
21-50	Complex, high risk program
> 50	Untestable program, very high risk

For Visual Basic, the following table lists the conditional keywords used to compute cyclomatic complexity.

Statement	CC Count	Rationale
If, IIf	+1	If adds a decision
ElseIf	+1	ElseIf adds a decision
Else	0	Decision is at the If statement
Select Case	+1 per case	Each Case adds a decision – not the Select
Select Else	0	Decision is at the Case statements
For	+1	For adds a decision at loop start
While	+1	While adds a decision at loop start
Until	+1	Until adds a decision at loop start
Do	0	Decision is at While/Until statement – no decision at unconditional loop
Catch	+1	Catch adds a decision
When	+1	When adds a decision