Software Metrics
using EiffelStudio

With material by
Yi Wei & Marco Piccioni

June 2007
What is a software metric, and why

A software metric is a measure of some property of a piece of software or its specifications.

- quality must be clearly defined
- quantitative methods have proved powerful in other sciences

“You can’t control what you can’t measure”

-- Tom DeMarco
What to measure

Product properties
- Lines of Code
- Number of classes
- Cohesion & Coupling
- Conformance of code to OO principles

Process properties
- Man-month spent on software
- Number of bugs introduced per hour
- Ratio of debugging/developing time
- CMM, PSP
Traditional Metrics

- McCabe Cyclomatic Complexity (CC)
- Source Lines of Code (SLOC)
- Comment Percentage (CP)
McCabe Cyclomatic Complexity

A measure based on a connected graph of the module (shows the topology of control flow within the program)

Definition

\[ M = E - N + P \]

where

- \( M \) = cyclomatic complexity
- \( E \) = the number of edges of the graph
- \( N \) = the number of nodes of the graph
- \( P \) = the number of connected components.
Example of Cyclomatic Complexity

```
if condition then
    code 1
else
    code 2
end
```

\[ E = 4, \quad N = 4, \quad P = 2, \quad M = 4 - 4 + 2 = 2 \]
Source Lines of Code

A measure of the number of physical lines of code

Different counting strategies:

- Blank lines
- Comment lines
- Automatically generated lines

EiffelBase has 63,474 lines, Vision2 has 153,933 lines, EiffelStudio (Windows GUI) has 1,881,480 lines in all compiled classes.

Code used in examples given here and below are got from revision 68868 in Origo subversion server.
Comment Percentage

Ratio of the number of commented lines of code divided by the number of non-blank lines of code.

Critique:
If you need to comment your code, you better refactor it.
OO metrics

- Weighted Methods Per Class (WMC)
- Depth of Inheritance Tree of a Class (DIT)
- Number of Children (NOC)
- Coupling Between Objects (CBO)
- Response for a Class (RFC)
Weighted Methods Per Class

Sum of the complexity of each feature contained in the class.

Feature complexity: (e.g. cyclomatic complexity)

When feature complexity assumed to be 1,

\[ WMC = \text{number of features in class} \]

In Eiffel base, there are 5,341 features,

In Vision2 (Windows), there are 10,315 features,

In EiffelStudio (Windows GUI), there are 89,630 features.
Depth of Inheritance Tree of a Class

Length of the longest path of inheritance ending at the current module

for **CHAIN**, DIT=7
Number of Children

Number of immediate subclasses of a class.

In Eiffel base, there are 3 classes which have more than 10 immediate subclasses:

- ANY
- COMPARABLE
- HASHABLE

And of course, ANY has most children.
Coupling Between Objects

Number of other classes to which a class is coupled, i.e., suppliers of a class.

In Eiffel base, there are 3 classes which directly depend on more than 20 other classes, they are:

- STRING_8
- STRING_32
- TUPLE

Class SED_STORABLE_FACILITIES indirectly depends on 91 other classes.
Number of features that can potentially be executed in a feature, i.e., transitive closure of feature calls.

```plaintext
foo is do
  bar
end

bar is
  f1
  f2
end
```

Response for a Class

```
RFC=3
```

Diagram:

- `foo` to `bar`
- `f1` from `bar`
- `f2` from `bar`
Metrics tool in EiffelStudio

A code quality checking tool with seamlessly working style:
  Coding - Metricing - Problem solving - Coding

Highly customizable:
  Define your own metrics to match particular requires

Metric archive comparison:
  Compare measurement of your software to others

Automatic metric quality checking:
  Get warned when some quality criterion are not met
Metrics tool – Evaluate metric

![Image of the Metrics tool interface with a selected metric and its value set to 10]
# Metrics tool – Investigate result

<table>
<thead>
<tr>
<th>Metric Evaluation</th>
<th>Detailed Result</th>
<th>Metric Definition</th>
<th>Metric History</th>
<th>Metric Archive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric name:</td>
<td>Type: Basic</td>
<td>Unit: Class</td>
<td>Value: 377</td>
<td></td>
</tr>
<tr>
<td>Input domain:</td>
<td></td>
<td>root_cluster</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Results:

<table>
<thead>
<tr>
<th>Class</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML_TABLE</td>
<td>web.table</td>
</tr>
<tr>
<td>HTML_TABLE_CONSTANTS</td>
<td>web.table</td>
</tr>
<tr>
<td>STDIN</td>
<td>web.stdio</td>
</tr>
<tr>
<td>STDOUT</td>
<td>web.stdio</td>
</tr>
<tr>
<td>SHARED_STDOUT</td>
<td>web.stdio</td>
</tr>
<tr>
<td>SHARED_STDIN</td>
<td>web.stdio</td>
</tr>
<tr>
<td>HTML_PAGE</td>
<td>web.html</td>
</tr>
<tr>
<td>HTML_TEXT</td>
<td>web.html</td>
</tr>
<tr>
<td>HTML_GENERATOR</td>
<td>web.html</td>
</tr>
<tr>
<td>HTML_CONSTANTS</td>
<td>web.html</td>
</tr>
<tr>
<td>HTML</td>
<td>web.html</td>
</tr>
</tbody>
</table>
Metrics tool – Define new metric
### Metrics tool – Metric History

<table>
<thead>
<tr>
<th>Metric name</th>
<th>Current value</th>
<th>Previous value</th>
<th>Difference</th>
<th>Filter</th>
<th>Result</th>
<th>Calculated time</th>
<th>Input domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncommented features</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td>05/26/2007 8:24:52.311 PM</td>
<td>sample</td>
</tr>
<tr>
<td>Features</td>
<td>62</td>
<td>–</td>
<td>–</td>
<td></td>
<td></td>
<td>05/26/2007 7:10:30.859 AM</td>
<td>sample</td>
</tr>
<tr>
<td>Classes</td>
<td>2</td>
<td>–</td>
<td>–</td>
<td></td>
<td></td>
<td>05/26/2007 7:19:40.375 AM</td>
<td>sample</td>
</tr>
<tr>
<td>Classes</td>
<td>242</td>
<td>–</td>
<td>–</td>
<td></td>
<td></td>
<td>05/26/2007 7:40:30.734 AM</td>
<td>base</td>
</tr>
</tbody>
</table>
Metrics tool - Archive

[Image of the Metrics tool interface with highlighted options for Metric Evaluation, Detailed Result, Metric Definition, Metric History, and Metric Archive.]