DCC / ICEx / UFMG

Measuring Size for Reuse

Eduardo Figueiredo

http://www.dcc.ufmg.br/~figueiredo

Measuring Size

- Size is the most obvious software attribute which can be measured statically
- Aspects of size
  - Length: physical size
  - Functionality: what users get
  - Complexity: problem solved

How hard is measuring size?

- Each software artifact is a physical entity
  - It can be described in terms of size
- Measuring size should be simple and straightforward
  - And consistent with measurement theory
- In practice, measuring size (software) presents great challenges

Example: LOC is not Simple

- Measuring Lines of Code (LOC) is not fully consistent
  - Some lines are more difficult to code than others
  - One solution could be give weight to lines that have more “stuff”
- However, this problem also occurs with most metrics

Analysing Metrics Together

- Let’s say human size is measured in terms of only height
  - Based only on height, we cannot determine whether a person is obese
- We need two size metrics, height and weight (+ empirical understanding)

Size and Reuse

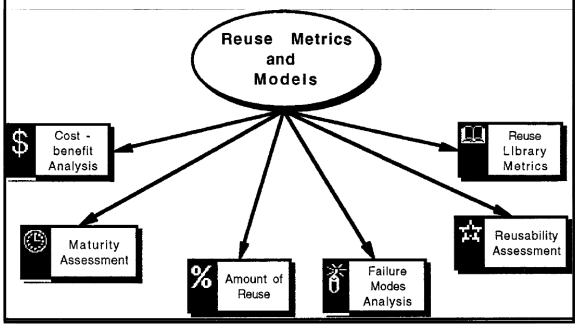
- Reuse is the extent to which the software is not new
  - Size of the reused product
- Extent of Reuse
  - Reused Verbatim (reused as is)
  - Slightly modified (< 25% LOC)
  - Extensively Modified (> 25% LOC)
  - New (nothing comes from a previous code)

Software Reuse: Metrics and Models

Metric and Models

- Organizations implementing systematic reuse must be able to measure
  - Quantify their progress
  - Identify the most effective reuse strategies
- A metric is a quantitative indicator of an attribute of a thing (software, process,...)
- A model specifies relationships among metrics (and external attributes)

Reuse Metrics and Models



Reuse Metrics and Models

1. Reuse cost-benefits models
2. Maturity assessment
3. Amount of reuse
4. Failure model
5. Reusability
6. Reuse library metrics

Cost-benefits and Maturity

- Reuse cost-benefits models
  - Analysis of quality and productivity payoffs
  - Goal is justify the cost and time invested in systematic reuse
- Maturity assessment models
  - Categorize reuse process by maturity level
  - They are inspired by CMM / CMMI

Amount of Reuse and Failure

- Amount of reuse
  - Metrics are used to assess and monitor the reuse improvement
  - Track percentages of reuse for life cycle objects
- Failure model
  - Used to identify impediments to reuse
  - Help to understand why reuse is not taking place in the organization

Reusability and Reuse Library

- Reusability
  - Metrics that indicate the likelihood that a artifact is reusable
- Reuse library metrics
  - Metrics used to manage and track usage of a repository

Amount of Reuse

- In general

$$\frac{\text{amount of life cycle object reused}}{\text{total size of life cycle object}}$$

- LOC Example

$$\frac{\text{lines of reused code in a system}}{\text{total lines of code in a system}}$$

Reuse Level

- A system is composed of parts at different levels of abstraction
  - The metric level of abstraction must be defined to measure reuse
- Level of abstractions for a Java program
  - System, Package, Class, Method, Lines of Code, etc.

Reuse Level Definitions

- Given that a higher level item is composed of lower level items
  - **L** = the total number of lower level items in the higher level item
  - **E** = the number of lower level items from an external repository in the higher level item
  - **M** = the number of items not from an external repository that are used more than once

Reuse Level Metrics

- External Reuse Level (ERL)

  ERL = E / L
- Internal Reuse Level (IRL)

  IRL = M / L
- Total Reuse Level (TRL)

  TRL = ERL + IRL

Bibliography

- N. Fenton, and S. Pfleeger, **Software Metrics: A Rigorous and Practical Approach**, 2nd ed. Thomson, 1996.
- W. Frakes and C. Terry. **Software Reuse: Metrics and Models**. ACM Computing Surveys, 1996.