FiSMA Reuse Measurement Method, FiSMA RMM version 2002

Introduction

Reuse is said to be the most effective way to improve productivity and time-to-market at software development industry. Numbers of organisations have announced their targets to increase reuse rate, at least inside the organisation. On the other hand, there are not any commonly used good methods for software reuse measurement. That's why a work group of FiSMA, the Finnish Software Measurement Association, has developed this method. It is implemented in the current version of Experience Pro, the estimating and measurement software, used largely by FiSMA members.

Basic Concepts

FiSMA RMM is based on ideas of functional size measurement and a simple model of software project deliverables. Third fundamental concept in RMM is the externality or internality of the reuse. If the reuse doesn't cross the border of the project, it's called internal reuse. It is as useful as external reuse, in the projects point of view, but organisations, which want to increase their reuse rate, should concentrate more on external use. It means using reusable components developed somewhere else, before the project measured. It is possible to have both external and internal reuse in a project.

Model of software project deliverables concentrates on those concrete project deliverables, which are directly related to each functional user requirement. These deliverables are:

- Program code
- Test cases
- Software documentation (for maintainers)
- User documentation (for users of the software)

Each deliverable represents certain percentage of the total effort needed in implementation of the function. This is called the weight of deliverable type. Sum of these weights must be 100. Weights may be calibrated for each organisation and domain, based on the common and measured knowledge of its special quality.

Impact of reuse may either increase or decrease the effort needed to implement the piece of software under discussion. If the functions and deliverables of the new software are required to be reusable, the needed effort is bigger. If there are good reusable components available, the needed effort is smaller.

The result of FiSMA RMM is a reuse multiplier R. The value of R is 1, if there isn't any impact of reuse, or if the opposite impacts are exactly as big. If there are more benefits from reusable components, than additional burden caused by reusability requirements, the value of R is between 0 and 1. If the impact of reusability requirements is bigger, the value of multiplier is between 1 and 2.

Assumptions

The functional size measurement of the piece of software must be made, and a complete list of different types of base functional components (functions) shall exist before starting a reuse measurement. The complexity rating and size of each function shall be known.

If there is not any better information available, the default weights of deliverable types are:

- Program code 40 %
- Test cases 30 %
- Sw documentation 20 %
- User documentation 10 %

If a reusable component for any deliverable of any function is available, the impact to effort will be the weight of the deliverable multiplied by the size of the function. The sign is minus, because in this case the impact to total effort is decreasing.

If a deliverable of any function must be developed to be reusable, the impact to effort will be the weight of the deliverable multiplied by the size of the function. The sign is plus, because in this case the impact to total effort is increasing.

Reuse Measurement Process

1. Go through the list of functions on the functional size measurement report. Pick out all those functions where you recognise any impact of reuse.
2. Count the reuse impact r_i for each of those functions.
3. Count the value of reuse multiplier R with a formula:

$$R = (S + ? r_i) / S$$