Chapter VIII

Software Maintenance Cost Estimation

Harry M. Sneed
Institut für Wirtschaftsinformatik, University of Regensburg, Bavaria

This chapter deals with the subject of estimating the costs of software maintenance. It reviews the existing literature on the subject and summarises the various approaches taken to estimate maintenance costs starting with the original COCOMO approach in 1981. It then deals with the subject of impact analysis and why it is essential to estimate the scope of maintenance projects. Examples are given to illustrate this. It then goes on to describe some of the tools the author has developed in the past ten years to support his practice of maintenance project estimation including the tools SoftCalc and MainCost. For both of these tools empirical studies of industrial experiments are presented as proof of the need to automate the estimation process.

THE RATIONALE FOR MAINTENANCE COST ESTIMATION

Software maintenance encompasses the tasks of correcting, adapting, enhancing, and perfecting a software product which is already in operation. (ANSI-IEEE, 1998) Except in the case of critical errors and unavoidable adaptations, the user has an option of whether to correct, adapt, enhance, or perfect the software, or not. The maintenance task is not imperative. Even in the case of non-critical errors, it is often possible to go on using the software as it is. The decision whether to carry through the maintenance task or not depends on two factors – time and cost.

The situation is similar to a person with a defective automobile which still functions. The owner of the automobile has the option of whether to fix it or not, as...
long as it is not a safety hazard. Whether he has it repaired or not depends on the time and costs. In any case, he will want to know what it will cost before he does anything. If he is prudent, he will visit several repair shops to get different estimates. If he happens to be in a foreign country and the repair will take a week, he may decide to wait until he returns home. Or, if the cost is too high, he may decide to wait until he has the money to pay for it.

The time and cost plays an even greater role in the case of enhancements and perfection. It is similar to the situation of a home owner who wants to either add a bathroom on to the house – enhancement – or renovate an existing bathroom – perfection. In neither case is he compelled to act now. He can postpone this task indefinitely depending on his budget. Everything is a question of costs and benefits and priorities in the light of the current economic situation.

Software system owners, like unfinished home owners, have a number of options of what they could do next. They could upgrade the user interface, add new system interfaces, add new functionality, or restructure the code. What actions they take depend on their current priorities and the costs of implementing the actions. There is always more that could be done, than what the owner has the capacity or money to have done. If the owner is rational, he will weigh the costs and benefits of each alternative action and select them on the basis of their cost/benefit relationship.

Therefore, except for critical errors and unavoidable adaptations, knowledge of the costs is essential. The owner of the system must know what a given maintenance task will cost in order to make a rational decision whether to pay for the task or not. It is up to the maintainers of the software to give him a cost and time offer, just as the car repair shop is obligated to make an offer for repair to the automobile owner or the plumber is obligated to make an offer for a new bath to the home owner. This is such a natural phenomenon that it is astounding to note how uncommon it occurs in the software business. One can only attribute the lack of a proper customer service relationship to the immaturity of the software field.

**RESEARCH ON MAINTENANCE COST ESTIMATION**

The lack of attention to a proper cost/benefit analysis of maintenance tasks is reflected in the pertinent literature. There has been very little research published on the subject, and what research there has been, has had little impact on industrial practice. One of the first attempts to estimate the costs of software maintenance is described by Barry Boehm in his book *Software Engineering Economics* (1981). Boehm maintained that annual maintenance costs can be derived from the initial development effort – DevEffort – and the annual change traffic – ACT, adjusted by a multiplication factor for the system type – Type.
Related Content

Factors Affecting Team Motivation: A Survey of Finnish Software Engineers
[www.igi-global.com/article/factors-affecting-team-motivation/126954?camid=4v1a](www.igi-global.com/article/factors-affecting-team-motivation/126954?camid=4v1a)

Design and Evaluation of Automated Scoring: Java Programming Assignments
[www.igi-global.com/article/design-and-evaluation-of-automated-scoring/133112?camid=4v1a](www.igi-global.com/article/design-and-evaluation-of-automated-scoring/133112?camid=4v1a)

Integrating Security into Agile Models: Scrum, Feature-Driven Development (FDD), and eXtreme Programming (XP)
[www.igi-global.com/chapter/integrating-security-into-agile-models/108622?camid=4v1a](www.igi-global.com/chapter/integrating-security-into-agile-models/108622?camid=4v1a)
Using Description Logics for the Provision of Context-Driven Content Adaptation Services
Theoretical and Analytical Service-Focused Systems Design and Development (pp. 176-209).
www.igi-global.com/chapter/using-description-logics-provision-context/66799?camid=4v1a