A Methodology for Software Maintenance

Macario Polo, Mario Piattini, Francisco Ruiz
Escuela Superior de Informatica
Universidad de Castilla - La Mancha, Spain

Software maintenance is the most expensive stage of the software life cycle. However, most software organizations do not use any methodology for maintenance, although they do use it for new developments. In this article, a methodology for managing the software maintenance process is presented.

The methodology defines clearly and rigorously all the activities and tasks to be executed during the process and provides different sets of activities for five different types of maintenance (urgent and non-urgent corrective, perfective, preventive, and adaptive). In order to help in the execution of tasks, some techniques have been defined in the methodology. Also, several activities and tasks for establishing and ending outsourcing relationships are proposed, as well as several metrics to assess the maintainability of databases and their influence on the rest of the Information System.

This methodology is being applied by Atos ODS, a multinational organization among whose primary business activities is the outsourcing of software maintenance.

INTRODUCTION

Software Maintenance has been traditionally the most expensive stage of the software Life Cycle (see Table 1), and it will continue to grow and become the main work of the software industry (Jones, 1994). In fact, new products and technologies need to increase maintenance efforts in corrective and perfective (for hypertext maintenance, for example, as reported in Brereton, Budgen, & Hamilton, 1999), as in adaptive (for adapting old applications to new environments, as client/server as discussed in Jahnke and Wadsack, 1999). With this in mind, it is natural that software evolution laws announced by Lehman (1980) have been recently confirmed (Lehman, Perry, & Romil, 1998).
In spite of this, one study conducted by Atos ODS in Europe has shown that most software organizations do not use any methodology for software maintenance, although they do use it for new developments. This usage is really surprising, above all if we take into account that 61% of the professional life of programmers is devoted to maintenance work, and only 39% to new developments (Singer, 1998). So, we agree with Basili et al. (1996) the statement sense that “we need to define and validate methodologies that take into account the specific characteristics of a software maintenance organization and its processes.” Furthermore, as these same authors express, the improvement of the maintenance process is very interesting due to the large number of legacy systems currently used.

Usual solutions for software maintenance can be divided into two groups: technical, which, among others, encompasses reengineering, reverse engineering and restructuration; and management solutions, characterized by having quality assurance procedures, structured management, use of human resources specialized in maintenance, change documentation, etc. However, whereas every year new technical solutions are proposed, very little has been researched about management solutions. The consequences of this have been denounced by some authors: Pressman (1993), for example, affirms that there are rarely formal maintenance organizations, which implies that maintenance is done “willy nilly.” Baxter and Pigdeon (1997) show that incomplete or out of date documentation is one of the main four problems of software maintenance. For Griswold and Notkin (1993), the successive software modifications make maintenance each time more expensive. For Pigoski, there is a lack of definition of the maintenance process which, furthermore, hampers the development and use of CASE tools for helping in its management. Also, excepting some works (Brooks, 1995; McConnell, 1997), the modeling of organizations is a neglected area of research in Software Engineering in general. Fugetta (1999) states that most of techniques and methods for

<table>
<thead>
<tr>
<th>Reference</th>
<th>Date</th>
<th>% Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressman (1993)</td>
<td>1970s</td>
<td>35%-40%</td>
</tr>
<tr>
<td>Lientz and Swanson (1980)</td>
<td>1976</td>
<td>60%</td>
</tr>
<tr>
<td>Pressman (1993)</td>
<td>1980s</td>
<td>60%</td>
</tr>
<tr>
<td>Schach (1990)</td>
<td>1987</td>
<td>67%</td>
</tr>
<tr>
<td>Pigoski (1997)</td>
<td>1985-1989</td>
<td>75%</td>
</tr>
<tr>
<td>Frazer (1992)</td>
<td>1990</td>
<td>80%</td>
</tr>
<tr>
<td>Pigoski (1997)</td>
<td>1990s</td>
<td>90%</td>
</tr>
</tbody>
</table>

Table 1. Evolution of maintenance costs.
Related Content

A Comparative Analysis of Reliability Assessment Methods for Web-Based Software
[www.igi-global.com/article/a-comparative-analysis-of-reliability-assessment-methods-for-web-based-software/105630?camid=4v1a](www.igi-global.com/article/a-comparative-analysis-of-reliability-assessment-methods-for-web-based-software/105630?camid=4v1a)

2-SQUARE: A Web-Based Enhancement of SQUARE Privacy and Security Requirements Engineering
[www.igi-global.com/article/square-web-based-enhancement-square/77617?camid=4v1a](www.igi-global.com/article/square-web-based-enhancement-square/77617?camid=4v1a)

Big Data Analysis with Hadoop on Personalized Incentive Model with Statistical Hotel Customer Data
Modeling of Linguistic Reference Schemes
www.igi-global.com/article/modeling-of-linguistic-reference-schemes/142513?camid=4v1a