

# Consideration of Psychological Aspects in Efforts to Preserve Code Quality on Maintenance Projects

(abstract)

Wolfgang Ulbrich  
IT & More Gesellschaft für Projekt- und Organisationsberatung mbH  
Geschäftsbereich Qualitätsmanagement  
Schatzbogen 58  
D-81829 München

In software maintenance the following problem situation can frequently be found: Management is dissatisfied with maintenance of legacy applications because maintenance is too expensive, maintenance is error-prone, maintenance employees are not well motivated and using tools has not improved the situation substantially.

What can be done by management and employees to improve the situation?

The costs of possible measures have to be as low as possible and must be realised without stressing the budget. We want to show now that in consideration of psychological aspects during software maintenance a lot can be achieved.

## **1. Maintenance from psychological view**

If the 'soft' factors of the software maintenance are to be regarded, then the following is noticeable.

- Software maintenance is not a continuous process. In general, the process is not finished at a specific date. Usually there is an unspecific goal, like 'error-free working under production circumstances'.
- Software maintenance is rarely done in a team or only in smaller teams respectively. Intensive communication, common team memory and the complete informal model of the project/product are missing.
- One employee has the responsibility for larger system components or even the whole system.
- Bugs apparently always have large consequences (disaster) and must usually be removed under time pressure.
- Changes and extensions are often more difficult than new developments but maintenance receives little appreciation from many IT managers. How can a developer find satisfaction with maintenance?
- The goals of the maintenance team and the user are no longer identical after a while. Conflicts are frequent, e.g. because of suggestions for improvement.

## **2. Selected quality goals for the software maintenance**

Only selected quality goals for software maintenance are looked at. They have in common that they are strongly subjective and/or psychologically influenced.

- code quality (faultlessness, efficiency of the solution, reliability / robustness, maintainability / comprehensibility, reusability)
- process quality (meeting the deadlines, staying within budget, securing full and correct functionality)

## **3. Psychological traps and their background in connection with the preservation of the code quality**

At the time of execution of changes and extensions at existing code, psychological traps are set up for the employee. Traps work against a preservation and/or an improvement of the code quality.

These traps are based on characteristics of our thinking, which we inherit or acquire.

In maintenance process we find

- innate patterns of perception, thought and behaviour;
- acquired abilities and skills,

- hidden work principles (e.g. headlight model, economical principle/simplicity principle, tendency of conciseness / building for categories, linear causal thinking, overestimation of confirming information).

These have different effects e.g. exceptions and borderline cases are ignored, corrections and revisions remain fragmentary, failure in unusual and deviating situations are made and side effects are missed.

#### ***4. 'Immunisation' of the employees against psychological traps***

A complete immunisation cannot succeed because of our human nature (*errare humanum est*). Relevant improvements are however possible with relatively small costs. Sufficient motivation of the employees, change will of the employee and information and support by the management is necessary.

You have to help the employee .

- to recognize working psychological mechanisms,
- to have a self-knowledge (q.v. a self-assessment-questionnaire of a maintenance employee),
- to get an assistance by management and team, support by tools and check lists (e.g. check list 'Preservation of code quality')
- to get an appreciation of its reached progress.

#### **Bibliography**

Grams, Timm: Denkfallen und Programmierfehler. Berlin Heidelberg. Springer 1990