Ten obstacles inspections face in an organisation

Karol Frühauf
INFOGEM AG, Postfach, CH-5401 Baden
Phone: 0041 56 222 65 32, Fax: 0041 56 222 00 38
E-Mail: karol_fruehauf@infogem.ch

ABSTRACT

There is no doubt: inspection is a powerful technique for defect detection in software, in documents as well as in code, during development as well as during maintenance. Additionally, inspections have a number of positive side-effects like more reliable project progress assessment, knowledge dissemination, development of common values, and team spirit fostering. The use of this easy-to-learn technique is beneficial from an economical as well as a cultural and product quality point of view.

Based on our experience in teaching and practising the inspection technique in many different companies we identified ten common obstacles to its successful exploitation. Before these are presented, the embedding of the inspections in the overall software development process is discussed.

1. INTRODUCTION

Inspections are now at the age of twenty [1, 2, 3]. They share the destiny of all other software engineering techniques: It is obvious that they are effective and efficient yet they are not exploited to the full extent possible. Inspections have additionally the advantage of being a simple technique not requiring any investment; only resources already available need to be used a bit differently. This calls for flexibility on both the management and the software staff level, the lack of which is most likely the crucial point why applying inspections fails.

To produce things is considered much more important than to investigate what has been produced. The current fashion is to focus on development process improvement and by that "guarantee" the product quality. However, there is no escape: Defect detection in products during development is not only economically beneficial but also one of the prerequisites for a promising process improvement effort.

2. WHERE DO INSPECTIONS FIT IN

Three entities are of major interest in software engineering: the project, the product and the process. We consider inspections from the project management point of view and assume that somebody wants to use the project result, the software product, and that the organisation carrying out the project has a defined software development process. As shown in Figure 1, the project "executes" a process and delivers a product.

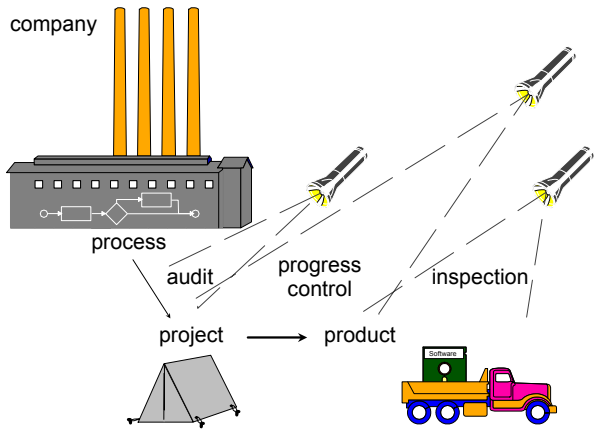

Fig. 1: The three "pros": process, project, product

The quality of these three "pros" is interrelated. According to our experience it is vital that the determination of the quality of these three entities is done independently of each other, with different techniques, and by different groups of people. This separation of concerns is a feasible way to a strong project control. Figure 1 illustrates also the focus of the three examination techniques inspection, audit, and progress control employed to investigate the product, the application of the process in the project, and the project state. In [4] the definitions of these techniques distinguish their aim as follows.

Inspection is the examination of a software item (document, piece of code) by a group of competent human beings in order to detect and identify defects. The software item is checked against the product requirements and later in the project against (design) specifications.

Audit is an independent examination to determine whether the actions and the related results comply with the documented procedures and whether these procedures are suitable to achieve the objectives and are effective. The basis for the investigation is a quality manual or a quality plan.

Progress control (meeting) is a comparison of the actual project state with the planned one with respect to schedule, costs, project goals, and requirements on the product to be delivered. The project plan is the reference for this investigation.

The simple development process model in Figure 2, resembling the "Vorgehensmodell" [5], defines the role of the inspection in the overall development process. Project management issues a task which is carried out by the development process. Data on progress are reported back to the management process and eventually a software item is delivered. The software item is put under configuration management control before the examination starts; only this way can we ensure that the investigated item is uniquely identified and can be retrieved again at any time.

The inspection process, like the test process, delivers the state of the affairs concerning the maturity of the product. The inspection report is used by the management process for taking the decision on the rework recommended by the inspection team. The rework is done again by the development process, the recheck either by the management or inspection process, depending on the criticality of the defects found.

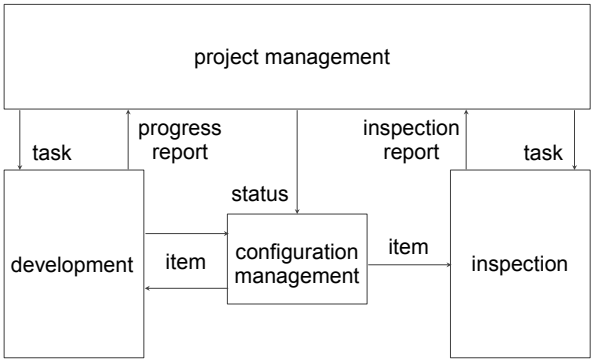

Fig. 2: Embedding of inspections in the development process

3. THE TEN OBSTACLES

Implementing inspections in an organisation according to the above principles faces a number of problems. Several roles are involved in the inspection process and most of them contribute to the ten most often encountered obstacles listed below.

Not surprisingly, many obstacles are put in the way of inspections by management.

1. Managers are blind on one eye
   Managers consider inspection to be "only a quality control" tool (and quality control is only "waste of money") and not what it also is, a project control tool. What is the value of the software engineer's statement "I completed the work item"? Either the manager finds out or he / she lets an inspection team find out.

2. Managers favour action
   To produce things (to design, to code) is held in higher esteem by managers than questioning the value of what has been produced (to inspect, to test). After all, inspections only cause trouble in showing defects which have to be removed, "this slows down the project, doesn't it?". Where are the glasses for the short-sighted ones?

3. Managers abuse inspection
   The results of inspections are taken into account in the performance evaluation of the software engineers. It is enough if this violation of the purpose of inspections happens once. The engineers will act in solidarity, all subsequent inspections will deliver "no defects detected".

The inspectors are the origin of the next couple of obstacles.

4. Inspectors do what they are used to doing
   The mission of inspectors is to spot defects and provide an exact explanation why it is a defect, and nothing else. This is an unusual task for software engineers who most of the time are paid for providing solutions to technical problems.
   Additionally, the inspectors implicitly assume: "Because I've got the task of inspecting what my colleague produced I am considered to be more knowledgeable". The consequence of this attitude is that the inspectors tend to tell the author what to do. Adults like this patronising even less than children do.
   On top of it, management appreciates solutions ("constructive criticism") more than problem raising. However, an exact characterisation of the defect is a prerequisite for finding the right solution and for enabling everybody involved to learn from the mistake.
5. Inspectors don't put their findings into words in advance
   The efficiency of the inspection meeting is highly dependent on the thoroughness of the preparation by the inspectors. Vague defect reporting by an inspector leads to lengthy discussions in the meeting.
   Recording the finding before discussing it focuses the discussion (not only the "audio" but also the "video" channel of the inspectors is used). This requires the inspectors to record "ready to print" findings during the preparation for the meeting.

Authors of the inspected software items also contribute some obstacles.

6. Authors don't recognise their limitations
   The authors are not aware of the fact that the inspector does what an author can't: to read what has been written down and not what one thinks is written down. This is a tremendous help which every author should be very happy about.
7. Authors mistake help for blame
   The issues raised by the inspectors are received by the authors as criticism of the person and not as help; a desire for justification is stirred up. Inevitably, every justification by the author "begs" for blame by inspectors.

Apart from lack of personality necessary for moderation, one major obstacle is originated by moderators of inspections.

8. Moderators do not moderate, they lead
   The moderators assume that to chair an inspection meeting means to lead the decision process. To moderate, however, means to help the others to come to a conclusion. A moderator can only succeed in that if he / she is able to withhold his / her personal opinion concerning the subject of discussion.

The last two obstacles come from the definition of the overall development process and how the inspection is embedded in it.

9. The corrective action process is not defined
   The process of deciding what to do with the inspection results is not defined and nobody in the organisation cares whether the raised issues are resolved or not.
10. They call inspection what they do
    The most drastic example are the inspections carried out at the end of a project phase, before the milestone (see Figure 3). More than a dozen management and technical people are involved and a vast amount of paper is „inspected“. It is carried out late, it is costly, and inefficient concerning defect detection. The mixture of management and technical aspects inhibits a deeply focused investigation and involves the danger of wrong technical decisions (caused by manager’s incompetence). No, don't call this inspection (better yet, don't do that at all).

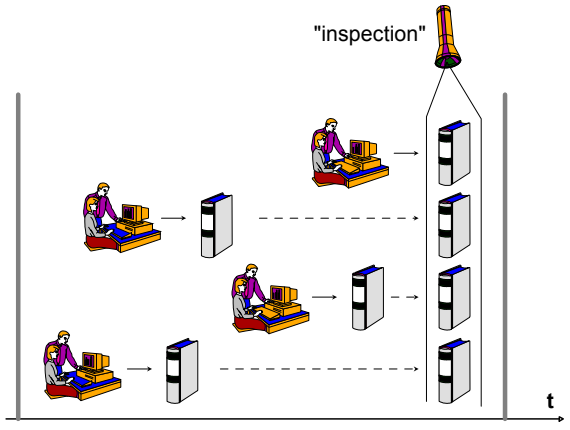

Fig. 3: The "inspection" (phase review)

Compare it with the use of inspections as depicted in Figure 4. The work result is inspected at the earliest point in time, immediately after it has been completed. The amount of paper to be inspected is appropriate for thorough investigation (and for reasonable production). Only persons possessing the necessary knowledge take the role of inspectors and the number of inspectors is restricted (to five); the absence of managers makes it easier to deal only with technical matters in the meeting. The human resources, the available expertise in an organisation can be used much more flexibly: After completion of the rework recommended by the inspection team the project manager knows that the software engineer is free for other tasks; if there is none for him / her in the current phase of the project the person can be „borrowed“ to a project which requires its expertise.

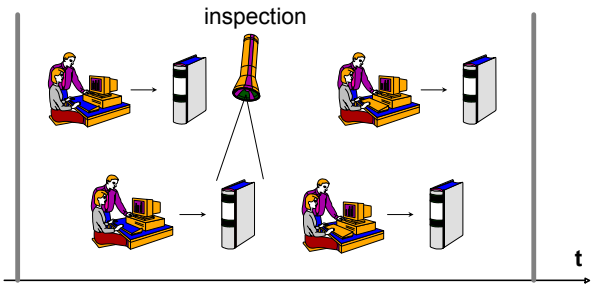

Fig. 4: The inspection

4. CONCLUSIONS

Inspections are a simple technique. The procedure is straight forward, the roles and their duties are well defined.

Inspections are difficult to implement, because

- technical people are not used to play roles
- management staff is not playing its role well
- the full value of the inspections is not understood well enough and
- for a magic reason there is little interest to understand it

Inspections have a lot of tiny ingredients which need to be employed correctly. A single one not applied correctly can already endanger the whole success of an inspection. Look out for the presented ten obstacles for doing it right; remove them before they kill your benefit from inspections.

A successful implementation of the inspection process leads to a certain enthusiasm. If the inspection process is not managed then with this enthusiasm also the benefit decreases.

The most successful implementation can lead to the desired effect: Nearly zero defects are detected in inspections because the delivered software items don’t have much of them. In this case the temptation is very big to stop „save“ inspections. Management need to be persevering and keep inspections going.

5. REFERENCES

[1] M.E. Fagan: Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal, Vol. 15, No. 3, pp. 182-211, 1976.

[2] D.P. Freedmann, G.M. Weinberg: Handbook of Walkhtroughs, Inspections, and Technical Reviews - Evaluating Programs, Projects, and Products. Little, Brown and Company, Boston and Toronto, 1982.

[3] T. Gilb, D. Graham: Software Inspection. Addison-Wesley, London, 1993.

[4] K. Frühauf, J. Ludewig, H. Sandmayr: Software-Prüfung - Eine Anleitung zum Test und zur Inspektion. vdf Hochschulverlag AG, Zürich und B.G. Teubner, Stuttgart, 1995.

{5} Allgemeiner Umdruck 250: Software-Entwicklungsstandard der Bundeswehr. Februar 1991.