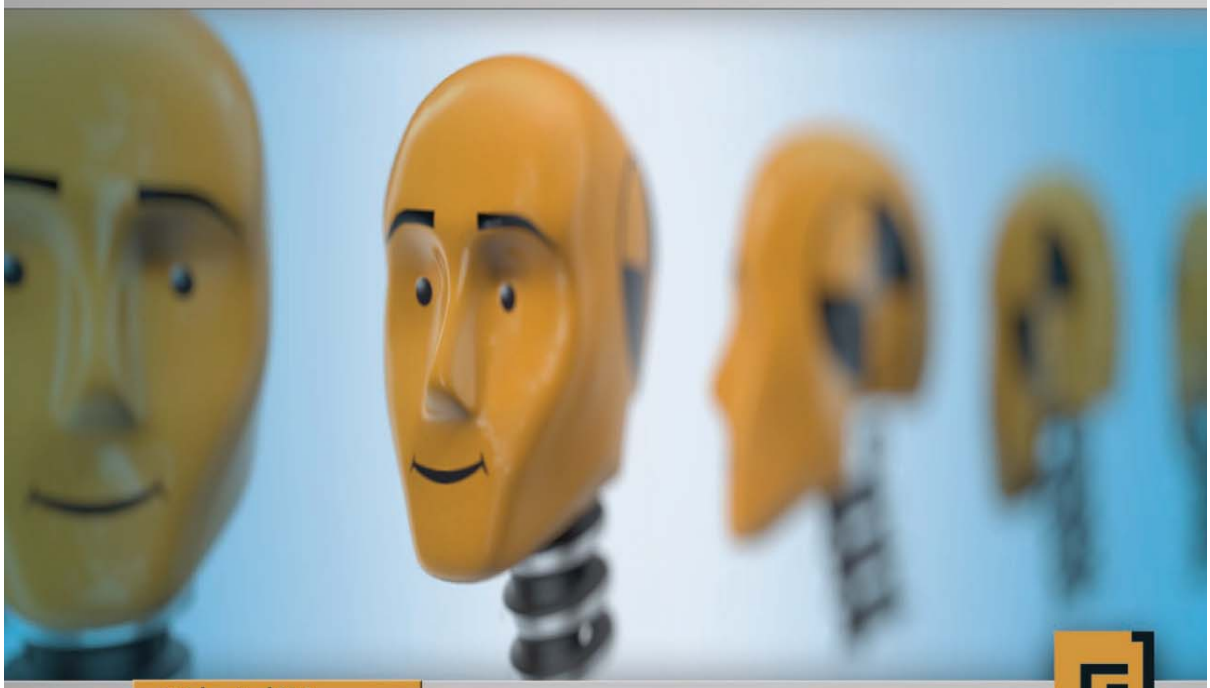
dpunkt.verlag



5th Edition

Andreas Spillner · Tilo Linz

Software Testing Foundations

A Study Guide for the Certified Tester Exam

- Foundation Level
- ISTQB® Compliant

Inhalt

4 Static Testing

Static testing and analysis of work products (documentation and code) contributes measurably to increased product quality. This chapter describes static testing in general as well as the specific process involved, with its activities and the roles that have to be filled. We describe four proven techniques and their specific advantages, as well as the factors that ensure success when applying them. We conclude by comparing static and dynamic testing techniques.

An underestimated technique

Static testing (or "static analysis") can be performed in a tool-based environment or manually, and is a testing technique that is often neglected. While the test object for a dynamic test (see Chapter 5) is an executable program that is run using test data, static testing can be performed on any kind of work product that is relevant to the development of a product. Static testing can take the form of close examination by one or more persons (see section 4.3) or can be performed using appropriate analysis tools (see section 7.1.3).

It's all about prevention

The underlying concept of static testing is prevention. Errors[1] and other deviations from plan are identified before they can have a negative effect during further development. All relevant work products are quality assured before anyone does any further work on them. This prevents faulty interim work products from entering the development flow and causing costly faults later on.

Static tests can be performed in a variety of situations (see below), and are often an integral part of the development process (i.e., testing resources are made available at the planning stage). For example, in safety-critical industries such as aviation and medicine, static analysis is an extremely important factor in ensuring the required product quality.

4.1 What Can We Analyze and Test?

A versatile technique

As well as the code itself, software development produces a lot of other work products too. Most documents play some kind of role in the further development of a product, making the quality of each individual document an important factor in the overall quality of the results.

Work products that can be checked using static analysis (especially reviews, see below) include specifications, business requirements, functional and non-functional requirements, and security requirements. Specification errors need to be found before they are converted into code. In agile projects, epics [URL: Epic] and user stories are subject to static testing. The kinds of defects that static analysis reveals include inconsistencies, ambiguities, contradictions, gaps, inaccuracies, and redundancies.

During the software design process, architecture and design specifications (such as class diagrams) are created that can be tested statically just like program code. The code that makes up a website can be tested this way too (for example, automatic verification of links and their corresponding target sites). Code can be checked for consistency with the project's programming guidelines[2].

Generally speaking, static testing verifies that any predefined standards have been adhered to in the documents being analyzed.

It is also advisable to examine and verify all documents, definitions, and tools that are created during testing (such as the test plan, test cases, test procedure and scripts, acceptance criteria). Other documents and work products that can be verified by static testing include contracts, project plans, schedules, budgets, and user manuals.

Furthermore, the results of static testing can also be used to improve the overall development process. If certain kinds of defects occur repeatedly during specific development steps, this is a sign that this step requires investigation and, if necessary, optimization. This kind of process is usually accompanied by additional staff training.

4.2 Static Test Techniques

Human mental and analytical skills

Human mental and analytical skills can be used to analyze and evaluate complex situations through close investigation of the documents in question. It is essential to the success of such analysis that the people involved understand the documents they are reading and comprehend the statements and definitions they contain. Static tests are often the only effective way to check the semantics of documentation and other work products.

They are various static testing techniques that differ in their thoroughness, the resources they require (i.e., people and time), and the objectives they pursue. Some common static testing techniques are detailed below. The terminology used is based on the ISTQB® syllabus and the ISO 20246 [ISO 20246] standard[3].

"Review" has multiple meanings

The commonest and most important static testing technique is the review[4]. In this context, the term "review" has multiple meanings. It is used to describe the static analysis of work products but is also used a general term for all static analysis techniques performed by humans. The term "inspection" is also used to describe a similar process, although it is often used to mean the formal execution of a static test (and the collection of metrics and data) according to predefined rules.

Different review processes

Reviews can be conducted informally or formally. Informal reviews don't adhere to a predefined process and there is no clear definition of the intended results or what is logged. In contrast, the formal review process (see section 4.3.1) is predefined and the participants (see section 4.3.3) document a planned set of results.

The type of review process you choose will depend on the following factors:

- The development lifecycle model: Which model are you using? Which places in the model and interim results are suitable for conducting a review?
- The maturity of the development process: How mature is the process and how high is the quality of the documents that are to be checked?
- The complexity of the documents to be tested: Which output shows a high degree of complexity and is therefore suitable for a (formal) review?
- Legal or regulatory requirements and/or the necessity for a traceable audit trail: Which regulations have to be adhered to and which proofs of quality assurance (and therefore reviews) are required?

Different objectives

The desired objectives of a review also determine which type of review process you choose (see section 4.4). Are you focusing on defect detection or on the general comprehensibility of the tested work products? Do new team members have to become familiar with a specific document by reviewing it? Does the review help the team reach a consensus decision? The type of review you conduct will depend on your answers to these questions.

4.3 The Review Process

Review activities

The process of reviewing work products comprises planning, commencement, preparation by the participants (specialists, reviewers, inspectors), communicating and analyzing the results, fault correction, and reporting (see figure 4-1).

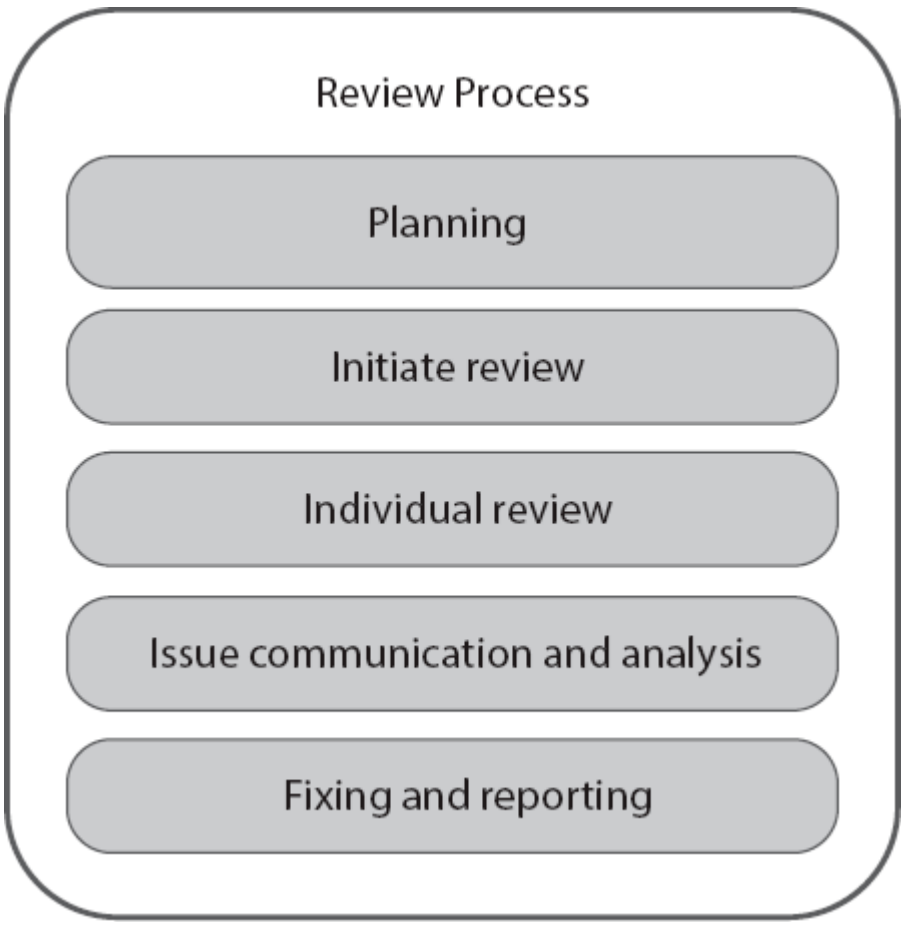


Fig. 4-1 *Review process activities*

4.3.1 Review Process Activities

The following sections go into detail on the individual activities mentioned above.

Planning

Define your objectives and choose a type of review

At the planning stage, management—or, more accurately, project management—has to decide which documents (or parts of documents) are to be subject to which type of review (see section 4.4). This decision affects the roles that have to be filled (see section 4.3.3), the necessary activities and, if required, the use of checklists. You also need to decide which quality characteristics to evaluate. The estimated effort and timespan for each review have to be planned too. The project manager selects a team of appropriately skilled participants and assigns them their roles. The project manager also has to check with the author(s) of the review object(s) that they are in a reviewable state—i.e., they have reached at least a partial conclusion and are as complete as possible.

Formal reviews require predefined entry and exit criteria. If the plan includes entry criteria, you need to check that these have been fulfilled before proceeding with the review.

Different viewpoints increase effectiveness

Checking a document from different viewpoints or having individuals check on specific aspects of a document increases the effectiveness of the review process. These viewpoints and/or aspects have to be defined at the planning stage.

You don’t have to review a document in its entirety. It often makes sense to select the parts that contain high-risk defects, or sample parts that enable you to draw conclusions about the quality of the document as a whole.

If you want to hold a pre-review meeting, you need to decide when and where it takes place.

Initiating a Review

The start (or “kick-off”) of the review process is the point at which the participants are provided with all the necessary physical and electronic materials. This can be a simple written invitation, or a pre-review meeting at which the importance, the purpose, and the objectives of the review are discussed. If participants are not already familiar with the review object’s

setting, the kick-off can also be used to briefly describe how the review object fits into its field of application.

Baseline documents

Alongside the work products that are to be reviewed, review participants require other materials. These include any documents that help to decide whether what they are looking at is a deviation, a fault/defect, or a correct statement. These are the basis against which the review object is checked (for example, use cases, design documents, guidelines, and standards). These documents are often referred to as the "baseline". Additional test criteria (perhaps in the form of checklists) help to structure the procedure. If you are using forms to log your findings, these need to be distributed at the start of the process.

If you are conducting a formal review, you need to check that the entry criteria have been fulfilled. If they are not, the review should be canceled. This saves time that would otherwise be wasted reviewing "immature" work products.

Individual Review Preparation

Studying the review object

A review can only be successful if all participants are well prepared, so each member of the review team has to prepare individually for the review.

The reviewers (or "inspectors") subject the review object to intense scrutiny using the provided documentation as their baseline. Any potential defects, recommendations, questions, or comments are noted.

Individual preparation can take a number of forms. For more detail on these, see section 4.3.2.

Issue Communication and Analysis

Collating your findings

Following individual preparation, the findings are collated and discussed. This can take place at a review meeting or, for example, in a company-internal online forum. The potential deviations and defects found by the team members are discussed and analyzed. You also have to define who is responsible for correction of the identified faults, how to monitor correction progress (see section 6.4.4), and determine whether a follow-up review of the fixed document is required or necessary.

The quality characteristics under investigation are defined during planning. Each characteristic is evaluated and the results of the analysis documented.

The review team then has to provide a recommendation regarding acceptance of the review object:

- Accepted without changes or with slight changes
- Revision necessary due to extensive changes
- Not accepted

Side Note: Recommendations for review meetings

If a review meeting takes place, try to observe the following recommendations:

- Limit the duration of the meeting to two hours. If further discussion is necessary, hold a follow-up meeting at the earliest on the following day.
- The moderator (see below) has the right to cancel or suspend a meeting if one or more reviewers are absent or are present but insufficiently prepared
- The object of the discussion is the work product, not its author:
  - Reviewers need to mind their phrasing.
  - The author mustn't be required to defend himself or his work. However, justification of his decisions can be useful.
- The moderator cannot be a reviewer too
- General questions of style that are not covered by the review criteria are not to be discussed
- The development and discussion of potential solutions is not the review team's job (see below for exceptions to this rule)
- Every reviewer must be given sufficient opportunity to present her findings
- The reviewers should try to reach and record a consensus
- Findings should not be formulated as fault correction instructions for the author. Suggestions for correction or improvement of the review object can, however, help to improve product quality.
- Individual findings are to be classed[5] as:

- Critical (the review object is not suitable for its intended purpose, defect has to be corrected prior to release)
- Major defect (usability of the object is limited, fault has to be corrected prior to release)
- Minor defect (slight deviation from plan—for example, grammatical error in print, doesn't affect usage)
- Good (free of defects, don't alter during revision)

Fixing and Reporting

Correcting defects

The final activities in the review process are reporting your findings and correcting any defects or inconsistencies the review has revealed. The minutes of a review meeting often contain all the required information, so that individual reports aren't necessary for defects that require the review object to be modified. As a rule, the author will rectify any defects revealed by the review.

As well as writing reports, any defects you find can also be communicated directly to the responsible person or team. However, this involves good interpersonal skills, as nobody really likes to talk about their own mistakes (see section 2.4).

Formal reviews involve more work

In a formal review, you need to record the current status of a defect or its report (see section 6.4.4). Changing the status of a defect is only possible with the agreement of the responsible reviewer. You also need to check that the specified exit criteria have been fulfilled.

Evaluating the results of a formal review meeting will help you to improve the review process, and to keep the corresponding guidelines and checklists up to date. This requires the gathering and evaluation of appropriate metrics.

The results of a review vary considerably depending on the type of review and the degree of formality involved (see section 4.4). The following sections detail different ways to approach the review process, and go on to discuss the roles and responsibilities involved in a formal review.

4.3.2 Different Individual Review Techniques

Collating your findings

The basic review process usually requires each participant to prepare for the team review. There are various individual review techniques[6] that help to reveal defects. These techniques can be applied to all types of review (see section 4.4), although their effectiveness can vary depending on the type of review you are conducting. The following sections describe various individual review techniques.

Ad hoc

No rules

As the name suggests, this technique involves no predefined rules regarding individual preparation. The reviewer usually reads the work products sequentially and notes any issues, findings, or defects that are found. The type of documentation is also arbitrary. Because they require very little (or no) preparation, ad hoc reviews take place quite frequently and, because there are no rules involved, the results depend heavily on the skills of the individual reviewer. If a review object is reviewed by more than one person, some issues are sure to be flagged multiple times.

Checklist-Based

Using checklists

Using checklists is a great way to structure your reading and thus increase the effectiveness of an individual review. You can define which checklists are used by which reviewers. A review checklist contains a list of questions based on potential defects that have been discovered in the course of earlier projects. The questions can also be based on formal guidelines or standards that need to be observed, as illustrated by the following example.

Case Study: Using checklists

Checklist for reviewing a requirements document:

- Is the document's structure standardized?
- Is there an explanation of how the document is to be used (i.e., an explanatory introduction)?
- Is a project summary included in the requirements document?
- Is a glossary available to aid consistent comprehension?

- …

Our example shows a checklist for reviewing a requirements document, but checklists can be formulated to aid reviewing any type of work product (for example, a code review). Checklists have to be updated regularly to include questions on newly recognized issues and to remove questions related to outdated or resolved issues. Checklists should generally be kept short and should only include key questions.

The main benefit of the checklist-based technique is that it systematically covers all typical types of defects. One potential downside of this technique is that it tends to "focus" the review on specific questions, thus reducing the likelihood of discovering defects that are not related to the listed questions. Always try to keep an open mind when using this technique.

Using Scenarios and Dry Runs[7]

Use case run-throughs

A scenario-based review is based on predefined, structured guidelines that dictate how to run through the review object. The availability of specific use cases simplifies this type of review, as all you have to do is perform "dry runs" of all the use cases. For example, in our *VSR-II* case study, we can check whether all the "configure vehicle" and "select special edition" cases are correctly implemented (see also section 5.1.6).

In the case of a code review, such scenarios can also be based on classes of defects. For example, one reviewer can concentrate on error handling in the code, while another concentrates on checking that boundaries are adhered to. Scenarios help a reviewer to find specific types of defects and are often more effective than simply working through lists of questions.

However, exclusive use of scenarios makes it more difficult to identify other types of defects (such as missing attributes), and should be avoided if possible.

Role- and Viewpoint-Based

Different roles and perspectives

During a role-based review, the reviewer is tasked with checking the review object from a specialist's point of view. To do this, the reviewer has to either already work in the specialist role, or at least empathize strongly with it. The basic idea of a role-based review is to utilize the skills each role provides. Of course, the person playing a role has to have the appropriate skills. For example,

it is always useful to have testers review requirements, as they can check early on in the development process whether the requirements are sufficiently precisely defined to effectively derive test conditions and test cases from them.

Utilizing specialist skills

Other typical review roles are those of stakeholders such as end-users or the operator of the system within the customer's organization. If end-users or the system's operator cannot take part in review sessions (which is usually the case), these roles have to be played by others. The "end-user" role can be further refined depending on whether the person is "experienced" or "inexperienced". If the system is to be used within an organization, a reviewer can play the role of system or user administrator.

Case Study: A role-based review

A requirements document is due for role-based review. During the review, Joe Smith is assigned the role of designer. Ideally, Joe will know about software system design and will fulfill this role within the real-world project. Joe analyzes the requirements to see what changes are necessary to the system's architecture in order for the requirements to be adequately implemented. Jane Brown is a tester and adopts this role for the review. During the review, she will check whether it is possible to unambiguously decide (following implementation and testing) that each requirement has been fulfilled. Jane flags the requirement "The system will enable rapid operation' as insufficiently quantified. There are no numbers that can be measured, and there are no preconditions or constraints that determine specific situations in which timing can be considered adequate. Other roles need to be filled, too, in order to further scrutinize the requirements.

Different viewpoints

All the available roles point to different defects, inconsistencies, and gaps. Role-based reviewers take on the viewpoints of various stakeholders, thus examining different aspects of the system. The (often subjective) viewpoints of the stakeholders themselves need to be included in the review. The following example illustrates the differing interests that exist in a scheduling review for the next iteration in our case study project.

Case Study: A perspective-based review

The sales department view says that a particular requirement requires high priority in the requirements document, as this feature has been requested by an important customer and needs to be implemented as soon as possible. From the

tester's viewpoint, this feature is very similar to other features that have already been implemented, so there shouldn't be any serious issues involved in implementing and testing it. She therefore recommends timely implementation of the new feature.

However, another new feature is more critical from a testing point of view, as it is the first in this project to implement artificial intelligence (AI). So far, the team has no experience in this field. The tester therefore assumes that this feature will require a lot of implementation and testing effort, so more resources need to be assigned to it.

The basic principle that underlies a perspective-based review is that each reviewer looks at the review object from a different viewpoint and runs through different scenarios as a result. The findings can then be used not only to evaluate the document in question but also to estimate the urgency and scope of any required changes. To do this, each reviewer runs through a different scenario[8] and uses the results to evaluate the document under investigation. Utilizing different stakeholder perspectives increases the depth of each individual review, and the distribution of roles reduces duplicate naming of defects and other issues. Using checklists is also a great way to increase the effectiveness of this kind of review.

Empirical studies (see [Sauer 00] and [Shull 00]) view perspective-based reviews as the most effective way to check requirements and technical descriptions. One of the keys to its success is the inclusion of many stakeholder viewpoints when analyzing and evaluating risk.

4.3.3 Roles and Responsibilities within the Review Process

The roles and responsibilities[9] involved in the review process have already been roughly described in the course of discussing the general principles of reviewing. The following sections go into more detail on roles and their responsibilities within a typical formal review.

Not every role has to be filled by one person, and there are overlaps between roles that justify having one person fill multiple roles (for example, moderator and review leader). However, which roles can be combined depends on the nature of the project and the quality criteria you need to achieve. The individual activities associated with individual roles also vary according to the type of review you are conducting (see section 4.4).

Management[10]

Management

Management selects which work products and supporting materials are to be included in the review and which type of review is to take place. Management is also responsible for planning reviews and allocating the required resources (people, time, money), and needs to keep an eye on cost-effectiveness. If the results of a review aren't conclusive, management needs to step in and take controlling decisions.

Side Note

A member of the management team who is also the review object's author's boss shouldn't take part in review meetings, as this involves an implicit evaluation of the author rather than his work, and can limit the freedom of the review discourse. It may also be the case that management doesn't have appropriate IT skills to take part effectively in technical review meetings. This is, of course, not true for project planning meetings and other management-led tasks, where project and general management skills are of the essence.

Review leader

Review leader

The review leader holds the overall responsibility and is therefore needs to make sure planning, preparation, execution, revision, and any follow-up work all contribute to achieving the review objectives. The review leader decides who participates in a review, as well as when and where it takes place.

Facilitator/Moderator

Facilitator

The job of a facilitator (also often referred to as a moderator) is to ensure the smooth running of review meetings. The success of a review meeting often depends on the moderator's chairing and diplomatic skills. He or she has to be good at bringing together opposing viewpoints and keeping discussions brief without hurting anyone's feelings. Also required is a good degree of assertiveness and the ability to detect undertones within the conversation. A facilitator mustn't have preconceived ideas or a personal opinion of the review object, is responsible for collecting any resulting metrics, and ensures that a report of the meeting is written.

Author

Author

The author in this case is the author of the review object. If there are multiple authors, one of them takes on the role of sole author for the duration of the review meeting. The author ensures that the entry criteria are fulfilled—in other words, that the review object is in a "reviewable" state. Usually, the author is responsible for rectifying any defects found during the review and therefore also for the fulfillment of the exit criteria. The author must never take criticism of the work personally. The review process serves solely to improve the quality of the review object.

Reviewer

Reviewer

Reviewers (sometime also referred to as "inspectors") are a group of up to five experts who take part in a review following appropriate preparation. Reviewers are usually part of the project team whose work is being reviewed, but can also be other stakeholders who have an interest in the results or people with specialist and/or technical skills.

A reviewer's task is to identify and describe problematic places in the review object. Having reviewers with different perspectives (testers, developers, end-users, system operators, business analysts, usability experts, and so on) aids the effectiveness of a review, although you heed to make sure that only specialists take part whose views are relevant to the review object.

Having individual reviewers concentrate on specific aspects of the review increases the efficiency of the review process. For example, one reviewer can concentrate on adherence to a particular standard while another checks the program syntax. This distribution of roles takes place at the review planning stage.

Reviewers must differentiate clearly between parts of the review object that pass the review and those that require improvement. Defects must be clearly documented so that they are easy to identify and resolve.

Scribe

Scribe

The scribe (or "recorder") documents the unresolved issues, open questions, and any other related tasks generated by the review. The documentation has to be brief and precise, and has to include undistorted summaries of all the discussions that took place.

As dedicated tools become more common, the job of review scribe is slowly becoming obsolete. If such tools are in use, every reviewer can record defects, open questions, and decisions directly in the tool interface without the need for separate documentation.

Side Note: Author and scribe

For practical reasons, the author is often also the scribe. The author usually knows exactly what needs documenting in order to perform the changes requested by the reviewers.

4.4 Types of Review

Side Note: Management reviews

Two types of review can be identified when looking at review objects:

- Reviews of documents that are created as work products during the development process
- Reviews that analyze the project or the development process itself

Reviews in the second group are referred to as “management”, “project”, or “process” reviews. Their aims include investigating whether regulations and plans are adhered to, analyzing the implementation of the required tasks, and the effectiveness of changes made to the process.

- The review object is the entire project and the main review objectives are to establish its current technical and economic state, as well as checking whether it is on schedule and how project management is doing
- Management reviews are often conducted when the project reaches a particular planning milestone, a development phase is completed, or as a “post mortem” analysis that supports the learning process for future projects
- In agile projects, such reviews often take the form of “retrospective” meetings. These are usually held following every sprint, and enable the team to compare notes and collect ideas on how to improve things in the next sprint.

The main objective is always to identify defects

The following sections go into detail on the first type of review, whose main objective is to identify defects by investigating documents. The type of review you perform will depend on the nature of the project, the available resources,

the type of review object, potential risks, the business area, company culture, and other criteria.

The review can be an informal review, a walkthrough, a technical review, or an inspection. All of these can be conducted as a “peer review”— in other words, with the participation of colleagues on the same an identical or similar level within the company hierarchy.

A single review object can be reviewed using more than one type of review. For example, an informal review can be conducted to verify that the review object is in a fit state for a subsequent technical review and that the effort involved is justified.

Informal Review

No strict guidelines

An informal review is a kind of “soft” review that nevertheless follows the standard review process (see section 4.3) but without a strict, formal structure. The main aim of an informal review is to identify defects and provide the author of the review object with short-term feedback. It can also be used to develop new ideas and suggest solutions to existing issues. Minor issues can also be resolved during an informal review.

Author/reader feedback cycle

An informal review is usually initiated by the author. The planning stage involves selecting the participants and setting a date for delivering the results. The success of an informal review is highly dependent on the skills and motivation of the reviewer. Checklists can be used, but an informal review usually does without a separate session for discussing the results. In this case, an informal review is really just a simple author/reader feedback cycle.

An informal review is therefore a simple double-check of the review object by one or more colleagues—a “buddy check”. The learning effect and interaction between team members are welcome side effects. A list of the issues found or a corrected/commented copy of the review object usually suffices as far as results are concerned. Techniques such as “pair programming”, “buddy testing”, and “code swapping” can also be seen as kinds of informal review. Because they are easy to organize and involve relatively little effort, informal reviews are widely used in all kinds of projects, and not only in an agile context.

Walkthrough

Running through the review object

As well as identifying defects, a walkthrough can be used to check for conformity with required standards and project specifications. This is also a forum for discussing alternative implementations, and an exchange of ideas about procedures or variations in style can also result and contribute to the participants' learning curve. The results of a walkthrough should be a consensus opinion.

The focus of a walkthrough is a meeting that is usually led by the author. The results need to be recorded, but there is no real need to prepare a formal transcript or summary report. Use of checklists is also optional. There is little individual preparation involved compared with a technical review or an inspection, and a walkthrough can be conducted with no preparation at all.

During the meeting, typical usage scenarios are named and walked through in a process-oriented fashion. Simulations or test runs of program parts (so called "dry runs") are also possible, and individual test cases can also be simulated. The reviewers use the comments and questions raised by the participants as a basis for identifying potential defects.

The author is responsible for making any changes that are required, and there is usually no further supervision involved.

Side Note

This technique is suitable for small teams of up to five people, and little or no preparation and follow-up are required. It is great for checking non-critical objects. In practice, walkthroughs range from extremely informal to quite formal.

Because the author usually leads a walkthrough, you have to take care that the review object is nonetheless critically and impartially scrutinized. Otherwise, the author's potential bias may lead to insufficient discussion of critical issues, thus distorting the results.

Technical Review

Alternative suggestions welcome

Alongside identifying defects, the main focus of a technical review[11] is forming a consensus. Other objectives include evaluating product quality and building confidence in the review object.

New ideas and suggestions for alternative implementations are welcome in a technical review, and technical issues can be solved by specialists. To make the most of this kind of discussion, colleagues of the author who work in the same or closely- related technical domain should participate as reviewers. Additionally, involving experts from other fields can help prevent the team from becoming blind to their own habits.

Individual preparation is critical in technical reviews. Checklists can be used too. If possible, the review meeting should be led by a trained review moderator, but not by the author. Discussion amongst the reviewers mustn't get out of control and should stick to finding a consensus about what the author can do to improve his work in the future. A meeting isn't mandatory, and the discussion can take place in other forums—for example, on the company intranet.

The results of the review need to be recorded, but not by the author. Usually, a list of descriptions of potential defects and a summary review report are prepared.

Technical reviews, too, can take many forms, from completely informal to strictly organized with predefined entry and exit criteria for each step of the process, and mandatory use of reporting templates.

Side Note

It helps to focus the review session on the most important points if the reviewers submit the findings from their individual reviews to the session moderator in advance. The moderator can then prioritize this input and use the meeting to discuss the most important points and the most obviously divergent opinions.

The results of a technical review are the responsibility of all participants. If you cannot reach a consensus during the meeting, you can hold votes to settle discussions and log the results in the review report.

It is not the responsibility of participants in a technical review to consider the consequences of the suggestions they make. This is down to management.

Inspection

Formal predefined flow

An inspection is the most formal type of review and follows a strict predefined flow[12]. All participants are project specialists or specialists in other aspects of the review object, and each adopts a specified role during the review. The review process is governed by rules that define check criteria for each aspect of the process. Each testing step has its own entry and exit criteria.

The objectives of an inspection are identifying defects and inconsistencies, determining the quality of the inspection object, and the building of confidence in the work products. Here too, reaching a consensus is an important part of the process. The findings of an inspection should help the author to avoid similar mistakes in the future and—like a technical review—help to improve the quality of future work.

An additional objective is the improvement of the software development process (see below). The objectives of an inspection are defined at the planning

stage and the number of issues that the reviewers need to address is limited from the start.

The inspection object is formally checked for “reviewability” and the fulfillment of the entry criteria is verified before the inspection begins. The reviewers’ individual preparation takes place according to predefined rules or standards using checklists.

A sample inspection meeting

A sample inspection meeting could take place as follows: The meeting is led by a trained moderator (not the author) who introduces the participants and their roles, and also provides a brief summary of the subject matter due for inspection. The moderator asks all the reviewers if they are sufficiently well prepared (for instance, by making sure that all the checklist questions have been answered). The moderator may also ask how much time the reviewers spent and how many defects they have identified.

General inconsistencies that affect the entire object are discussed and logged first.

A reviewer then makes a concise and logical presentation of the contents of the inspection object. If necessary, parts of the material can be read out (but not by the author). This is where other reviewers can ask questions and where the selected aspects of the object can be discussed in detail. The author (who mustn’t be the review leader, the moderator, or the scribe) answers any direct questions. If the reviewer and the author cannot agree on how to deal with an issue, this is put to a vote at the end of the session.

If the discussion wanders off the point it is up to the moderator to intervene. The moderator also has to make sure that all selected aspects of the inspection object (and the object in general) are covered, and that all defects and inconsistences are clearly recorded.

At the end of the session, all defects are presented and checked for completeness by all participants. Any unresolved issues are briefly discussed, but no suggestions for possible solutions are discussed. If there is no consensus as to whether an open issue represents a defect, this discrepancy is recorded in the report. The final report summarizes all results of the inspection and provides a list of descriptions of all potential defects.

To conclude, the inspection is evaluated and a decision is made as to whether the inspection object needs more work. During an inspection, any changes that are made and follow-up work that is done are formally regulated.

Additional evaluation of the development and review processes

Some of the data collected during an inspection can also be used to identify the causes of weaknesses in the development process and thus to improve its quality. The data can also be used to improve the inspection process itself. Any improvement in the quality of both processes can be verified by comparing data collected before and after any changes are made.

Side Note

This type of review is often referred to as a design, code, or software inspection. The name is based on the type of documents that are being inspected. However, if formal review criteria exist, all types of documents can be inspected.

Selection Criteria

Selecting the type of review

When and what type of review you use depends on the objectives you are pursuing, the required quality, and the effort this will cost. The project environment is critical to these decisions too, and it is impossible to make specific recommendations. You have to decide from project to project which type of review is best suited to the situation at hand. The following questions will help you decide which type of review to conduct:

- The form of the review's results can be a factor. Do you need a comprehensive report, or is an undocumented implementation of the results sufficient?
- Is scheduling easy or difficult? Finding a date when five, six, or seven specialists all have time can be hard work.
- Does the review require experts from multiple disciplines?
- How much specific knowledge of the review object do the reviewers need?
- How motivated are the reviewers to spend time and effort concentrating on the planned review?
- Is the planning effort commensurate with the expected results?
- How formal is the review object? Can tool-based analysis take place in advance of the review?
- How much management support do you have? Are reviews likely to be limited or even axed if time is running out?

4.5 Critical Factors, Benefits, and Limits

Improving quality and lowering costs

Reviews are an efficient tool for assuring the quality of the work products under investigation. Ideally, reviews will take place immediately a work product is finalized. This way, any inconsistencies or defects are identified in a timely manner and the author receives feedback as soon as possible. Resolving any issues leads to improved quality in the documents in question and thus has a beneficial effect on the development process, which is then pursued with fewer defects in tow. Furthermore, static tests often reveal defects that are difficult (or impossible) to find using dynamic tests (see section 4.6).

Early identification and correction of defects through reviews is usually cheaper than resolving defects later when the application has reached an executable stage of development. This is especially true when earlier versions of an application have been delivered or are already in use.

Static tests are generally cheaper than dynamic tests (see Chapter 5) because defects can be corrected directly in the document and further checks are no longer necessary[13]. Correcting failures that appear during dynamic testing generally requires additional confirmation or regression testing. Reviews often save significant amounts of development time.

However, it is not always accurate to say that a review will reduce costs. For example, if a static test finds a memory leak, remedying the issue can be extremely time-consuming. Here, you need to differentiate between the cost of conducting a review and the cost of correcting the defect(s) the review identifies. The cost of correcting text in a document is negligible. In other words, the effort involved in making a correction depends on the nature of the issue that requires attention.

Dynamic testing can involve less effort if a code review has already removed faults in the test object (i.e., the code) and the risk of discovering further faults has been reduced thanks to the review.

A reduced number of defects and inconsistencies in the reviewed and corrected document will also reduce costs during the system's lifetime. For example, reviews can reveal inaccuracies in customer requirements that can be resolved in advance of implementation. Reviews also lead to a reduction in the number of failures that occur when the system is running. Additionally, productivity during development increases, as the team will be working with improved designs and code that is much easier to maintain (i.e., to understand and alter).

Improved team communication

As well as increasing quality and reducing costs, reviews have positive effects on the collaboration within the development team too:

- Because reviews always take place in a group environment, they encourage the exchange of know-how among the participants. This leads to improvements in the working methods of individuals and thus to a general improvement in quality of the work they produce.
- The group environment makes it essential to present material in a way that is clear and comprehensible for all involved. This need for clarity often generates insights that the author wouldn't otherwise have had.
- The entire team feels responsible for the quality of the review object, and the result is a document that everyone understands.

In order to increase review quality, or to have a successful review in the first place, various organizational and people-related factors need to be considered. The following sections list the most important of these.

Organizational Success Factors

Organizational factors

- Project management supports the review process by providing adequate resources for reviews throughout the development process.
- Formal reviews are planned well in advance and are organized with clear, verifiable objectives.
- Participants are given sufficient time to prepare for a review.
- In principle, all types of review (for example, checklist-based or role-based) are suitable for identifying defects in the review object. The most appropriate type can be selected based on the agreed objectives, the type and level of the investigated object, and the skills of the participants.
- Checklists cover the most important risks and are always up to date.
- Large-scale documents are reviewed in parts rather than as a whole so that defect feedback can be provided as quickly as possible to the authors.

People-Related Success Factors

People-related factors

- You need to select the “right” participants to reach your planned review objectives. It is most productive to select people whose skills and viewpoints mean that they need to understand the review object anyway in order to continue their work. Because the review object will usually be used as the basis for designing test cases, it makes sense to have testers take part in a review. The testers are then familiar with the project documents from the start of the development process and can begin specifying test cases right away. Such a test-based viewpoint also aids other aspects of quality assurance, such as verifying an object’s testability.
- Reviews serve to assure the quality of the investigated object(s), so the identification of defects, inaccuracies, and deviations from plan is intended and should be encouraged. However, the reviewers must communicate their findings in an objective, unbiased manner. Reviews must be held in a trusting, confidential atmosphere, and participants should avoid using gestures that indicate boredom, frustration, or hostility to other participants. The author has to be sure that the results of a review don’t influence his standing in the company. The author should see a review as a positive experience and all participants should end up feeling that taking part in the review was time well spent.
- Participants take the time to concentrate on details. Reviews are held so that participants don’t lose their concentration. Don’t attempt to review large documents in one sitting, and limit the length of the review session in advance.
- If necessary, offer participants additional training in advance of the review, especially for a formal review such as an inspection.
- Try to encourage a learning atmosphere so that everyone benefits from each review. This also improves the quality of the review process and individual review techniques.

Side Note: Why reviews sometimes fail

- If a review fails because of insufficient preparation, this is usually due to poor scheduling
- If the reviewers don’t understand the significance of the review or the effectiveness of the review process in quality improvement, you can

use figures from the current project (or from earlier projects) to underscore the importance of the review process

- Reviews can fail due to insufficient or incomplete documentation. Alongside the review object, you have to make sure that all supporting documentation is available with the necessary depth of detail. A review should only go ahead if this is the case.

4.6 The Differences Between Static and Dynamic Testing

Identifying different types of defects

Static and dynamic tests can be used to achieve the same objectives (see section 2.1.2), but complement each other by identifying different types of defects. Static tests identify faults directly in documents and other work products, whereas dynamic tests usually identify failures in the source code rather than other types of documents (with the exception of executable models). Any failures you discover have to be traced back to the faults that cause them if they are to be successfully resolved. Failures often remain undiscovered for a long time—for example, if the corresponding code is only rarely executed. It therefore requires a lot of effort to design appropriate test cases. These kinds of faults in the source code are usually easier to identify using static tests.

Checking "inner" quality

Dynamic testing verifies the visible, external behavior of the test object, whereas static tests focus on improving the internal quality of the object under investigation.

Typical defects found by reviews

The following sections detail typical defects and inaccuracies that can be identified quickly and cheaply, and therefore quickly resolved, using reviews:

- **Requirements defects**

  Typical defects found in requirements documents are: inconsistencies, ambiguities, contradictions, gaps, inaccuracies, or redundancies.

- **Software design defects**

  Where defects are identified in the architecture documents that specify the system's internal interfaces. When the degree of dependency (or "coupling") between individual components or modules is high, it can be

difficult to fully understand and test them, not least because the effort involved in creating a corresponding test environment is disproportionately to the importance of the results. The cohesion between components/modules can also be analyzed. A strongly coherent component is responsible for a single, clearly defined task.

A component that performs a loose selection of tasks indicates low cohesion, which is just as disadvantageous as a high degree of coupling. Algorithms and database structures that have been inefficiently designed can be identified too.

- **Coding defects**

  If you have sufficient time and appropriately skilled personnel, all programming faults can be identified using code reviews. This enables you to detect variables with no value or redundant variables that are never called. You can also use a review to detect unreachable or duplicate code. These types of defects can also be detected by a compiler or by using automated static analysis tools, which is often quicker and cheaper than a human-based technique.

- **Deviations from standards**

  Standards and guidelines help to achieve high quality products, while insufficient adherence to programming guidelines leads to the opposite, namely: poor quality. If it is clear that adherence to certain guidelines will be checked, the motivation to stick to them from the start is much greater.

- **Faulty interface specifications**

  The names of the interfaces between components and their parameters (number, data type, and sequence) must be checked individually, as not all discrepancies will be found during integration.

Example: ***Incorrect interface definitions***

The NASA Mars probe *Mars Climate Orbiter* is a famous and well-documented example of incorrectly specified interfaces [URL: Mars Climate Orbiter]. The system programmers used multiple measuring systems (metric and imperial), and this oversight led to the loss of the probe. A formal code review would probably have identified this inconsistency in the measurement system's specifications.

- **Security vulnerabilities**[14]

  Alongside dynamic security tests, static tests can also help uncover these kinds of vulnerability. For example, when buffer overflows occur because constraints weren't explicitly checked, or when input data or SQL queries are deliberately manipulated.

- **Gaps or inaccuracies in test basis traceability or coverage**

  We discussed the importance of traceability and degree of coverage in section 2.3.8. Gaps in traceability make traceability itself useless, as you can no longer make a connection between the requirements and the corresponding test cases. Inaccuracies in the exit criteria and the required degree of coverage make these aspects unusable too, and can lead to incorrect estimates or distorted results. A review report can, for example, point out missing tests for fulfilled exit criteria.

Maintainability

Software systems often have surprisingly long lifecycles, making maintainability an important aspect of any system. Most shortcomings in maintainability can be identified using static tests. These include incorrect modularization (see coupling and cohesion above), bad reusability of individual components, and complex code that is difficult to modify (i.e., not "clean code" as defined in [Martin 08]) and therefore increases the risk of creating new faults when changes are made.

4.7 Summary

- Every work product (i.e., every document) can be reviewed, provided that all participants in the review are aware of how to read and understand the document in question.
- Reviews are static tests that—unlike dynamic tests—don't require code to be executed. This means that reviews can be applied at any time to all forms of work products that are used in or created by the software development process.
- Multiple pairs of eyes see more than a single pair, and the same applies to software development. Reviews used to monitor and increase quality are based on this principle. Documents are inspected by more than one

person, and the results are discussed and recorded in a dedicated meeting.

- The activities involved in the review process are planning, initiation, individual preparation (i.e., individual review), discussion of the findings (issue communication and analysis, usually in a dedicated meeting), fixing and reporting.
- In order to better identify defects at the individual review stage, there are various different review techniques you can use:
    - **Ad hoc**

      No rules
    - **Checklist-based**

      Pre-formulated questions support the review process
    - **Scenarios and dry runs**

      Scenarios are run through
    - **Role-based**

      Participants adopt specific roles for the review
    - **Perspective-based**

      Different views on the review object
- The roles that need to be assigned for a review are manager, review leader, facilitator (moderator), author, reviewer (expert), and scribe.
- There are many types of review and, because the related standards don't use a unified terminology, the names used to describe them can vary. In this chapter we have discussed the following four review types:
    - An *informal review* doesn't follow any formal rules, and the form of the resulting documents is not regulated. Because it involves relatively little effort, the informal review process has become a widely accepted and well-used technique.
    - A *walkthrough* is an informal technique that sees the author presenting a document to reviewers in a meeting. Here too, preparation is minimal. Walkthroughs are ideal for small development teams who need to discuss alternatives and distribute know-how within the team.

    - A *technical review* follows a more strictly regulated flow in which individual reviewer preparation is essential. Reviewers should be professional colleagues of the author so that alternative realizations can be discussed.
    - An *inspection* is the most formal of these types of review. Preparation is based on checklists, and each review steps has its own entry and exit criteria. The review session is led by a trained facilitator. Alongside checking the quality of the work product, the objective of an inspection is to improve the development and review processes.
- To make them as effective as possible, reviews are generally tailored to the specific requirements of the company. One of the most important factors is establishing a cooperative atmosphere among all participants in the software development process.
- If you keep appropriate organizational and people-related success factors in mind, and as long as you take care to avoid known pitfalls, a review can be a seriously effective tool for increasing quality in software development projects.
- Reviews typically identify different defects than dynamic testing.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W