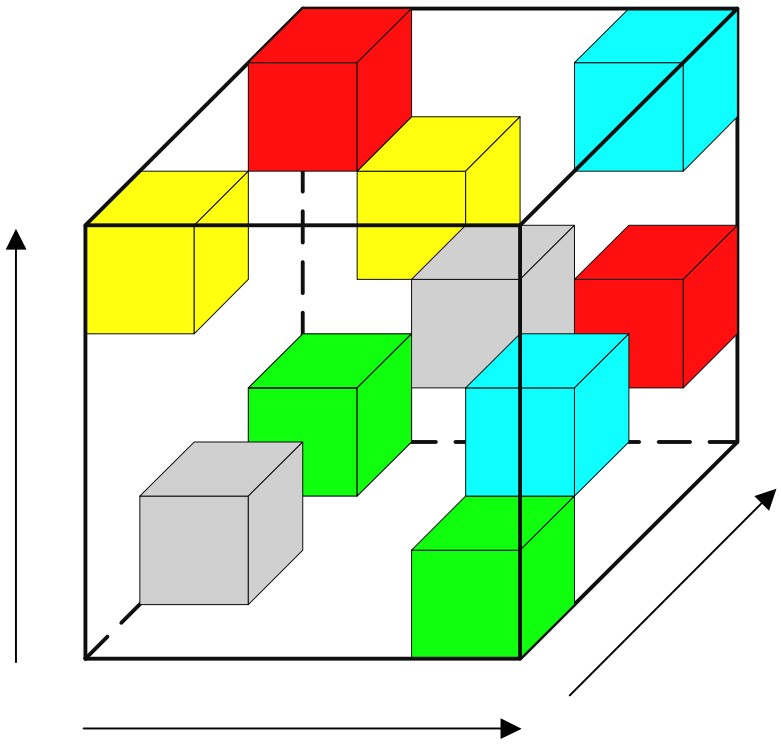
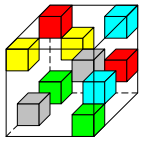
THE TEST CUBE

Approach to structure and manage test sets



Author	Specialist group Test Cube
Version	1.0 (October 25, 2006)
City	Diemen
Status	Definite



PREFACE

The Test Cube is an approach to manage test sets which has been applied for the first time at a customer of Sogeti by Wout Komduur (account manager) and Paul von Winckelmann (test manager).

To make years of acquired knowledge and experience available for Software Control employees and other interested people, a group of specialists has been organized that is comprised of experienced Test Cube users. The objective of this effort is to further develop the Test Cube to a generic treatment which can be applied successfully for all of our customers.

This group of specialists has used Arno Verweij and Jack van de Corput, who were unfamiliar with the Test Cube-approach and acted as a sounding board. Arno has also edited this document at his expense.

This document and described approach is the joint result of the Test Cube specialist group. The following people have contributed to the realisation:

Hans Delprat
Robert Jelgerhuis
Michel Suiveer
Arno Verweij
Guido Wester
Paul von Winckelmann

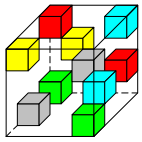
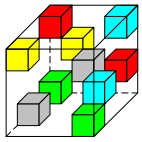
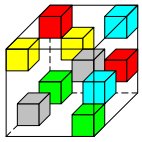


TABLE OF CONTENTS





1. INTRODUCTION

A good test set (= collection of test cases) is of priceless value. During the realization of an application it is the basis for the reporting concerning test coverage and progress. In a maintenance situation it is a source of knowledge concerning the application and a source of ready-made test cases for regression tests.

In practice, test sets are frequently poor. At the beginning of new projects it turns out better than expected, however, as the project comes to an end the time pressure increases. The project focus shifts entirely to delivering the application and the maintenance of the test set slips. By the time the application and the test set reach the maintenance stage, the discrepancy between both is so great that the test set for the maintenance organization is only moderately useful.

Setting up a well thought out test set is easier said than done. Generally, testers themselves consider the test set only as a product which arises automatically when specifying test cases. They are not used to consider the test set entirely as a project deliverable and, for example, to formulate acceptance criteria for the set in advance.

In this document an approach is described to develop, use and manage test sets: the Test Cube. The Test Cube is a coherent collection of principles which makes it possible to:

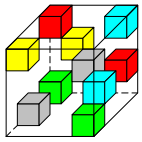
- specify test cases and execute test cases on the basis of prioritizing;
- report progress of test specification and/or test execution quickly and adequately;
- plan and estimate test projects accurately;
- compose regression test sets quickly and flexibly;
- incorporate changes of the test object easily in the test set.

The Test Cube is an application of TMap in which one test product is singled out: the test set. The Test Cube replaces the accepted, but inaccurate idea of the test set as an occasional collection of test cases by means of a concrete product: a project deliverable that meets the initially established acceptance criteria.

With this approach the (future) maintenance organization gets more of an understanding of this important project deliverable. The Test Cube helps to prevent the destruction of capital which appears when a test set, which many hands have worked on for months, appears at the transfer to maintenance no longer useful.

The principles used in the Test Cube make it very suitable for use in combination with Business Driven Test Management (BDTM).

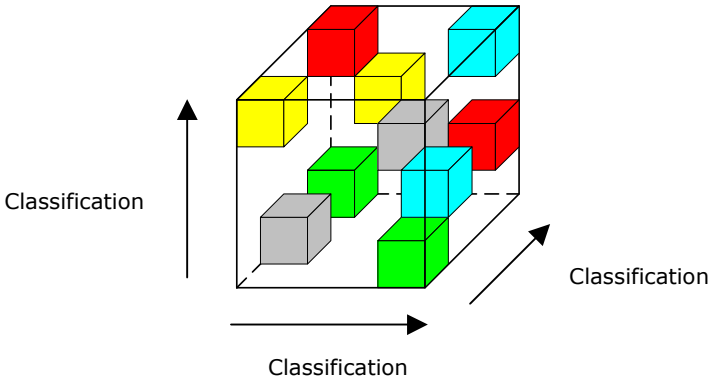
This document begins with a short description of the mechanism of the Test Cube; the principles behind it, how it works and what you can do with it. Then the relation between the Test Cube and BDTM is considered. The setting up and using of the Test Cube is elaborated by TMap phase. A separate chapter is devoted to reporting and estimating with the Test Cube. This document ends with chapters featuring practical information about useful tools and results of experiences.



2. THE CONCEPT OF THE TEST CUBE

The Test Cube is in concept nothing more than a set of additional data that is established per test case: the test cases in the test set are classified. With the help of these classifications and different cross-sections, subsets of test cases can be selected from the test set.

The below figure represents the Test Cube. Every block represents a test case with its classifications.

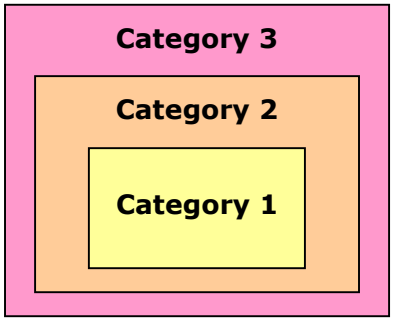


The Test Cube is a translation of a selection mechanism to a three-dimensional figure, a cube. However, where a cube only has three dimensions, the Test Cube has in principle an unlimited number of selection possibilities (classifications) and with that infinite dimensions. For example, these classifications can be:

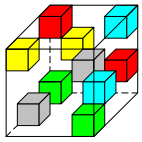
- application
- object part
- function
- product risk
- risk class
- process (part)
- release
- requirement
- transaction
- weight category

A good selection of classifications and correct classification of the test cases is indicative of the usefulness of the Test Cube.

Essential in the Test Cube is the classification to weight category. This classification indicates the 'weight' of the test case in the test set. This classification in particular makes it possible to (regression) test risk-based with a variable depth. This works as follows, in this example with three weight categories:



As the figure indicates, the different weight categories are subsets of each other (category 3 exists of all test cases of category 1, 2 and 3; category 2 exists of test



cases of category 1 and 2). Category 1 contains the most elementary test cases, because these are a part of all other categories.

Every object part has test cases within every category. The numerical relationships between different categories lies between the in practice determined margins (for example category 1 = 10-15% of the total number of test cases, category 2 = 60-70% and category 3 = 100%, see section 4.8).

The application of these weight categories, for example to compose regression test sets, is as follows:

- By only selecting the test cases of category 1 of an object part a small regression test set arises. This subset is used for an object part where no adaptations have been done (or for a pre-test on a new or radical modified object part).
- The test cases of category 2 (= including category 1) produce a normal regression test set, for example for an object part that has been changed.
- The test cases of category 3 (= including category 1 and 2) cover the total object part and is applicable for new or radically modified object parts.

The weight categories can be also used for other objectives. See for that the chapters for set up and use of the Test Cube.

The option to segment the test set with the help of these weight categories is essential for the Test Cube. It is conditional for the scaling of the (regression) test and cannot, under any circumstances, be left undone. See paragraph 4.6 for instructions on how to divide the test cases over weight categories.

The Test Cube does not prescribe fixed test design techniques. All existing techniques can be used to specify the test cases in a test set. The choice for a specific technique is determined in a conventional manner by means of the test strategy.

Also to the degree of detail in which the test cases are specified, makes the Test Cube no demands. When it is foreseen that test cases will be executed by testers without knowledge of the subject matter, the test cases can be specified in more detail. This choice is mainly dependent on the organization of the test project.

In only one matter the Test Cube makes a specific demand to the test cases: they should be independent of each other. This is the so-called independent principle of the Test Cube:

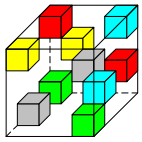
- There are no dependencies between the test cases.

  If the execution of one test case is conditional for the execution of another test case, problems arise when composing test sets from the Test Cube: it can just be that all sort of test cases should be executed first that do not belong to the original test set.

- Test cases can be executed in parallel.

  Test cases that require exclusive use of the test environment for a specific period obstruct the execution of other test cases. This complicates the planning of the lead time of the process.

By using the Test Cube the scope of the test set and with that the coherent activities in the test process become concretely measurable. By keeping up some statuses per test case, the Test Cube offers a current picture of the progress. By keeping up metrics of finished or current tests, the Test Cube can help provide precise planning for future tests or for the continuation of current activities.



3. THE TEST CUBE AND BUSINESS DRIVEN TESTMANAGEMENT

3.1 Risk Based Testing

Using the Test Cube is the ideal way to give an interpretation of Risk Based Testing. The test set in the Test Cube is the product of a test strategy on the basis of which some object parts are tested in more depth than others. In the risk analysis which underpins that test strategy, elementary risk factors *fault chance*, *frequency of use* and *damage*: Risk Based Testing is assumed.

With an application in maintenance the risk factors *frequency of use* and *damage* generally change little. With modifications, especially the *fault chance* increases. In the test strategy for a maintenance release it is possible to determine whether a standard or increased fault chance applies for specific object parts. With the help of the Test Cube it is then easy to compose a more or less comprehensive regression test set: Risk Based Regression Testing.

An important precondition for Risk Based (Regression) Testing is that the test set in the Test Cube *is a good reflection of the current application*. If this precondition is not met, then the regression test set is of limited value: an extract from a test set motivated by risks of which it is not exactly clear which risks are being covered.

Should it be that as a result of a maintenance releases the risk factors *frequency of use* and/or *damage* modify significantly, then the consequence could be that the test set should be adjusted structurally.

3.2 Business Driven Test Management

Business Driven Test Management (BDTM) is a test approach that enables the customer to control a test process based on costs, time versus results and risks. To achieve this, information is needed from the test process. The Test Cube offers this information. This will be explained by means of the three for BDTM essential elements.

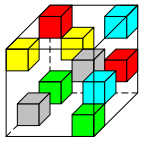
Test strategy

The Test Cube makes it possible (particularly in maintenance situations) to provide concrete insight in the costs to cover the risk, while determining the test strategy. With a well-organized Test Cube, including reliable figures, it can be easily determined which test cases cover a risk or object part with a certain weight category and what the costs of executing these test cases are. These costs can be set side by side with the costs which would be the consequence of the risk manifesting itself. The result of this is a test strategy in which the customer has made more conscious choices for taking or covering certain risks.

Estimation and planning

On basis of the test strategy an estimation is formulated. In case the Test Cube has been set up to support the application and reliable figures are available (in a maintenance situation), the estimation can be very accurate. This advantage is less significant if it concerns a new project. However, since the test set size will be considered in an early stage, the Test Cube supports estimating and planning here as well (at least in adjusting these during the test process).

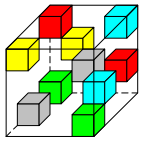
In practice, the test estimation often is perceived by the customer as being too high. This is solved by reducing the lead time of each object part (based on the original test



strategy). As a result, the entire test process will be thinner, without having insight into the additional risks that may occur. Applying the Test Cube will provide an understanding of the (costs to cover the) risks and offer the customer the opportunity to identify the risks that should be covered and which are actually worth taking. Based on this, both test strategy and test set will be adapted to reduce the number of hours needed.

Reports and adjustments

Whether or not the test process is conducted according to plan (in terms of both time and money) it can provide significant information for the business. By using the Test Cube, the business can discover (during the entire test process), both periodically and on-demand insights into the quality of the test objects, pending risks and the progress of the test process. Depending on the chosen tools, it is possible to report and adjust quickly. In this situation as well, savings that are the result of taking additional risks can be expressed in terms of time and money immediately. This enables the business to take timely measures and, if necessary, adjust the project and/or test process. And again, savings due to taking additional risks or the costs of using additional resources can be expressed in terms of time and money immediately. Reporting to the business stakeholders in terms of risks and risk coverage will be very easy if product risks and risk classes are included as classification within the Test Cube. With the same Test Cube, it is also possible to report to other stakeholders across different classifications.



4. SETTING UP THE TEST CUBE

This chapter will discuss the setting up of the Test Cube where the situation without test set (e.g. at the start of the project) acts as point of view. The described phases are related to TMAP phasing. The activities aimed at setting up the Test Cube are additional to those identified by TMAP for each different phase.

Finally, section 4.9 will describe how an existing test set can be implemented into the Test Cube.

4.1 Planning phase

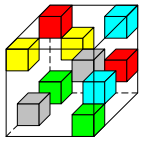
It is important to first determine whether it is feasible to implement the test set of the project within the Test Cube, and if so, which requirements apply to the test cases and the Test Cube.

The Test Cube is an optional supplement to TMAP. Against the advantages of using the Test Cube, investments need to be made. In addition, using the Test Cube implies requirements to the structuring of the test process as well. Several arguments may influence the choice of using a Test Cube positively (+) or negatively (-):

(+) The customer makes specific demands upon the test set, for example based on identified risks (see chapter 8 for a practice example).
(+) Periodic releases and incremental development methods require test set maintenance and regression tests on a regular basis, which is well-supported by the Test Cube.
(+) When under time pressure, it is necessary to have an understanding of and control over the test process, as well as clear demarcation of the test set.
(–) The output of the Test Cube is limited during a one-time test, for example testing a conversion. Returns will be maximized by testing several releases. However, the Test Cube offers the advantage of a well-grounded understanding of the test set for a one-time test process as well.
(–) Immaturity of the test or project organization leads to high investments in controlling and structuring the test process and, moreover, increases the risk of low returns due to inexperience of the test organization or project management.
(–) The test organization is lacking dedicated team members.

The most significant requirements for the test cases in the Test Cube and for the Test Cube itself can be inferred from the test strategy. Supplementary requirements are related to the way the test cases are specified. Some of the requirements are specifically connected to the use of the Test Cube; others apply to each test process:

- Which object parts are recognized?
- Which product risks are covered by the test process (if BDTM is used)?
- How many weight categories are needed and what is the percentage distribution of the test cases across these categories (see chapter 2 for figures)?
- Which other classifications are needed? Each classification adds a dimension to the Test Cube, but this may not be at the cost of the transparency and usability.
- At which level should the test cases be described: either logically, physically or both (and will the tests be executed by testers with or without knowledge of the subject matter)?



- Are there requirements to apply for the connection between logical and physical test cases?
- How can the independent principle (test cases are mutually independent, see chapter 2) be guaranteed and how is dealt with exceptions?
- How important is maintenance of the test set?
- Which tools are available to register the Test Cube?
- How accurately should the test process be estimated?
- Should the test set be automated easily (this requires an unambiguous description of the input and output of the test case)?
- How large will the test set be?

In case the customer makes specific demands to the test set, these requirements will be included in the Mastertestplan (MTP). It is not a necessity to describe the use of the Test Cube and other possible requirements (which are the result of using the Test Cube) in the MTP: it is an approach to realize the test set. However, it is essential to fine tune these requirements with the customer, who should be comfortable with the final product as well.

4.2 Control phase

An important activity during the specification and execution phase concerns maintaining figures that are related to the effort (in hours) for the specification and the execution per test case respectively. These figures are essential for the progress control, the estimation of the continuation of the process and the estimation of future releases.

To gain additional insight into the progress of the specification phase, it is possible to phase the specification process of a test case and indicate the relative effort (figures) per phase. See section 6.2 for an example of such a phasing.

For both the specification and execution phase, it is necessary that all members of the test team keep up their progress in the Test Cube, by registering the status and/or test result of each specified or executed test case.

4.3 Setting up and maintaining infrastructure phase

Not applicable.

4.4 Preparation phase

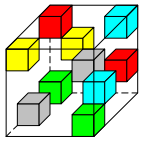
The data collected for the Test Cube are aggregated in the form of, for example, spreadsheets or setting up a test specification tool. Data structures are established and lists with allowed values for, among others, the recognized classifications, releases, object parts and test levels are filled in.

The Test Cube is being made accessible for all members of the test team.

4.5 Specification phase

The test cases are specified according to the techniques as determined in the test strategy. For each test case, a reference is incorporated in the Test Cube and all classifications are entered (which is - in case a test specification tool is used - part of entering the test case and not a separate action).

The size of the test set can be estimated based on a testability review test basis. By quickly specifying the test cases on a logical level, the total size of the test set will be



made clear and the remaining specification process can be estimated and controlled accurately. Consequently, during the remaining specification phase it will be possible to make use of the weight categories of the Test Cube by physically elaborating on the test cases out of category 1 in the first place.

If the size of the test set is estimated and the members of the test team keep up their progress accurately, the Test Cube provides immediate insights into the current status, which implies up-to-date information related to both finished and unfinished processes.

Keeping up the progress adequately enables the test manager to create metrics, thereby getting more insight into the test process.

4.6 Execution phase

For the execution phase, the Test Cube offers the total set of executable test cases. In the planning and control phase, an estimation has been made of the required execution time per test case.

As well as in the preceding phase, it is necessary that the members of the test team register the progress and results of the test execution in the Test Cube. The Test Cube provides immediate insights into the progress, test coverage and/or risks and offers the possibility to create the desired metrics.

4.7 Completion phase

In the completion phase, the quality of the test object and test process is reported. The Test Cube provides both reports with the most important factual information.

In addition, these reports should describe figures related to several metrics as formulated during the specification and execution phase. The better the test manager is able to prove his metrics, the more transparent (for the organization) planning and estimation are for the next test process(es).

When transferring the testware to the management organization, additional attention is required to transmit the applied approach of the testware management: the Test Cube. The Test Cube will only render adequate results in the management situation, and it is important to prevent not using a well set up tool or using one inappropriately.

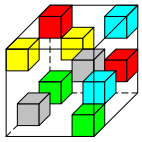
By entering all classifications of each test case correctly and completely during the specification phase, a structured test set is created in the Test Cube step-by-step. As a result, conserving testware takes less time and, moreover, leads to an increased quality level compared to usual processes.

4.8 Guidelines for the use of weight categories

Neither the classification of test cases nor the percentage distribution across the weight categories is an exact science. However, there are some guidelines to fulfill this task adequately. The following steps are hereby recognized:

1. Determining weight categories;
2. Determining allocation formula per weight category;
3. Dividing test cases across weight categories.

For each step, the situation during management releases acts as frame of reference: to assess a test object which has been modified after it was tested.



The purpose of the Test Cube is not to determine for each test case in a new project if the application performs adequately. If in a new project it is not feasible (for any reason whatsoever) to test all functionalities, the Test Cube may assist by reducing the amount of test cases in a controllable way. The nature of the risks taken here differs (i.e. much larger) compared to assessing a test object (with more or less depth) that already has been tested.

All steps are based on controlling risks (Risk Based (Regression) Testing): which risks are allowed and how to cover the risks that are not allowed? It speaks for itself that these steps should be executed by the owner of the system. Like no other (at least better than the tester himself), the owner has an understanding concerning the levels at which the application should be assessed, the amount of test cases that may be needed and which ones.

4.8.1 Determining weight categories

To start, it is necessary to determine at what levels (increasing in depth) the test object in management releases should be evaluated. The recognized gradations of change (with corresponding risk classes) may serve here as guideline, for example *unaltered, modified, small, average, large*. Each level on which the test object should be evaluated becomes a weight category, such as: *pre-test, quick scan, small/average/large regression test, complete re-test*.

Deducing weight categories from the recognized gradations of change makes the composition of regression test sets for a management release very transparent. For the test strategy is determined to what extent an object part has been changed, which automatically leads to the regression weight corresponding to the gradation of change.

The weight categories apply to all object parts in the test set. The practice example of chapter 2 demonstrated three of them: quick scan, regression test and complete re-test. In practice, a number of three or four weight categories are workable.

4.8.2 Determining allocation formula per weight category

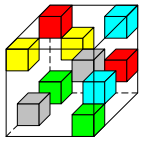
For each weight category should be wondered which percentage of test cases in the test set should be sufficient to evaluate the application at that level. Examples of these questions are:

- Which part of the test set should be executed to be able to evaluate whether the application has sufficient quality in order to continue testing? (*pre-test*)
- Which part of the test set should be executed to be able to determine whether unaltered object parts function as of old? (*quick scan)*
- Which part of the test set should be executed to be able to determine whether (slightly) modified object parts have been subject to regression? (*average regression test*)
- Which part of the test set should be executed for largely modified object parts? (*complete re-test*)

The answers to these questions produce an allocation formula, often in the form of a margin per category. These margins are applicable to all object parts in the test set. The practice example of chapter 2 demonstrated the following margins: *quick scan* 10-15%, *regression test* 60-70%, *complete re-test* 100%.

4.8.3 Dividing test cases across weight categories

Dividing test cases across weight categories is always being done per object part. For each object part (starting with the lowest depth) is determined which test cases



should be executed to be able to evaluate the object part at the desired level (each weight category contains the test cases of all preceding categories as well), for example:

- For a *pre-test* or *quick scan*: some elementary correct paths.
- For a *small regression test*: test cases that touches the core functionality (for example: verifying screens, running batches, printing data, guarantying referential integrity, carrying out modifications) which are not already classified into earlier categories.

  When using these criteria leads to a selection of test cases that is larger than the desired percentage, it is necessary to reduce the number of test cases in order to prevent the lead time from being too long. This can safely be done by examining whether several test cases in the test set cover the same test path. These duplications move to the following weight category.

  In case the number of test cases is less than the desired percentage, it is possible to leave it this way. However, the weight category may also be completed to reach the desired percentage with additional test cases.
- For a *large regression test*: test cases that touch the core functionality as well, but with an extended test coverage (for example: verifying help screens, the remaining correct and error paths, running one-off batches, printing data for internal use, testing user-friendliness, testing performance). And again, only those test cases which are not already classified into earlier categories are selected.

  As well as for the small regression test, the number of test cases may be reduced or completed to reach the desired percentage.
- For a *complete re-test*: all not yet classified test cases (for example: theoretical or exotic test cases).

This way, for each object part a balanced distribution of test cases across the weight categories is obtained.

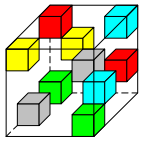
In case an object part is characterized by such a large risk that it will always be tested with the largest possible depth, segmenting the test set for that object part seems to be an administrative task without added value. However, segmenting is useful in this situation as well, since it serves more aims than only composing a (regression) test set. Examples are prioritizing during specification, execution or automation, and preventing random reduction (of test cases) due to lack of time. Moreover, it is not possible to indicate accurately for an unsegmented test set how much testing of the object part with a lower depth would save. The strength of the Test Cube is particularly hidden in such negotiations.

4.9 Test Cube and an existing test set

The preceding sections described the building from the beginning of the test set and setting up the Test Cube simultaneously. The need for a structured test set frequently just arises in the management situation, when the test set for the application already exists. Also in this situation it is possible to apply the Test Cube.

The following 4-steps scenario can be used thereby:

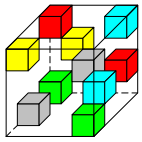
1. The feasibility of the Test Cube should be determined. This depends on several factors, see section **Fout! Verwijzingsbron niet gevonden.**, but it is moreover affected by the finding of the following point.



2. The existing test set should be evaluated:
    - Is the test set still an adequate reflection of the application? If this is not (anymore) the case, effort should be performed to repair it. This is an important precondition, see section 3.1.
    - Is the test set contradictory to the independent principle of the Test Cube? If so, test cases should be rewritten. Starting point is having no dependencies between the test cases.
3. Requirements of the Test Cube should be formulated.
4. The data collections for the Test Cube are constructed and for each test case a classification is included.

This final step can be rather lengthy if too many classifications are recognized. It is recommended to start with two classifications: weight category and object part.

In case only an unstructured or incomplete test set is available, the situation is not hopeless. Setting up the Test Cube helps to develop the missing structure. Later on, after the test set is included in the Test Cube, it will quickly be clear for which object parts too little or too many test cases are selected. This may be a good reason to correspond the test set with the test strategy, or, if this is lacking, to create a test strategy.



5. USING THE TEST CUBE IN MAINTENANCE

This chapter will describe the use of the Test Cube in maintenance. As in the previous chapter, phases are related to TMAP test phasing.

5.1 Planning and control phase

For each maintenance release, the size and impact of the modifications on the existing application should be determined. The test strategy for the maintenance release should be created based upon that.

Furthermore, it should be verified whether the Test set in the Test Cube is still up-to-date and is connected to the chosen test strategy (see also section 3.1). For new and modified functionalities additional test cases should be specified.

The Test Cube offers the possibility to react on an increase in the chance error of an object part, by determining the test weight category for each object part, for example:

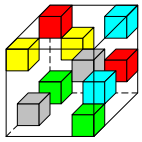
- Unaltered object parts are tested with weight 1 (only test cases with weight category 1 are executed).
- Slightly modified object parts are tested with weight 2 (only test cases with weight category 1 and 2 are executed).
- Radically modified object parts are tested with weight 3 (all test cases are executed).

Based on the weight of the separate object parts, the Test Cube can determine the adequate size of the test set for each given strategy. This, combined with an estimation concerning the number of new test cases for the new or modified functionality, is the size of the new release. By using metrics related to test specification and test execution, an accurate estimation for the test process can be made. This estimation is just as reliable as the metrics determined during earlier releases.

In case the customer limits the quantity of resources, the Test Cube can be used to determine easily the savings related to the decrease in weight of the object part. Consequently, the customer obtains already in the planning phase a concrete overview of the risks taken and the costs needed for covering these. If necessary, the weight can be reduced *optionally*.

The eventual interpretation of the test set (which object parts with which weights) is laid down in a test plan.

If metrics are available with regard to specifying, adapting and executing test cases, it is possible to plan the test process accurately. Applying the independent principle (see chapter 2) simplifies the planning of the execution phase, since little or no interrelationships between test cases need to be taken into account.



5.2 Control phase

When the number of findings identified during the test execution is significant higher compared to other subsystems, still a well-founded request will be initiated for additional resources, to test the object part with a higher weight.

5.3 Setting up and maintaining infrastructure phase

Not applicable.

5.4 Preparation phase

The data collections for the Test Cube are prepared to the upcoming maintenance release.

5.5 Specification phase

The specification of test cases for a new or modified functionality does not differ from the specification during setting up the Test Cube, thus in accordance with the chosen test strategy (see section 4.5).

Test cases for an expired functionality are cleaned (e.g. by awarding these test cases the status *expired*, by which the test execution history is preserved).

The test set for the maintenance release is composed by selecting a subset of existing test cases for each object part, in accordance with the agreed weight.

5.6 Execution phase

In the first place, test cases for new and modified functionalities are executed.

Other than that, the execution of test cases does not differ from the execution during setting up the Test Cube, see section 4.4.

5.7 Completion phase

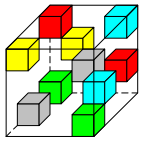
When testing the maintenance release has been completed, the test cases for new and modified functionalities automatically become part of the consolidated test set for the application. For each object part that has been adapted during the maintenance release, the test set should be validated again:

- All test cases for an object part have been stored in the same directory (with correct references to these in the Test Cube) or repository-node.
- The division of test cases across the weight categories is in conformity with the defined standards.

It is verified whether the used metrics for test specification and test execution need adjustments.

Other than that, see the completion during setting up the Test Cube in section 4.5.

By applying the Test Cube, the test set in the management situation automatically evolves with the test object.



6. REPORTS & METRICS

Outsiders often experience the test process as rather unclear and intangible. Thus, the test manager has many questions to answer. To be able to answer those questions satisfactorily, facts are needed. Here, updated and well-grounded figures (metrics) provide the solution.

Metrics do not only serve planning and budget aims, but reflect the compliance of test processes to the standards of the organization as well.

In new projects metrics are created during the duration of the process. Since metrics involve a large amount of numbers, the availability of reliable figures may take a while. Thus, during the introduction phase, the added value of metrics remains limited.

When testing several releases, metrics can be developed based on figures from previous releases. As a result, it is possible to create metrics for the quantity of the test object as well as for the planning of the test process (see TMAP, section 23.5-6). Important metrics in the Test Cube are:

- Required time for creating a new test case;
- Required time for modifying a test case;
- Required time for executing a test case;
- Error handling time.

6.1 Estimation and planning

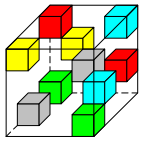
In case a maintenance release is defined, the impact on test object and test set should be determined. Determining the size of the test release is most difficult here. Seeing the structured test set, by using figures maintained by the Test Cube it can be easily determined what the amount of test effort for a certain maintenance release will be.

As the size of the work that needs to be executed is known, the lead time and related costs can be specified by using the collected metrics. The accurateness of the planning stands or falls with a reliable estimation of the activities that need to be executed.

6.2 Reports

During the test process, the test manager should be well informed about the progress. Not only to be able to report to the customer, but even more to be involved in the test process and to be able to, if necessary, take measures in a timely manner.

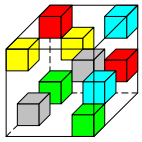
It is possible to report the final results of the different activities (the number of test cases specified, the number of test cases executed with result OK). Extensive activities, such as test specification, can be subdivided into smaller ones by means of a Work Breakdown Structure (WBS). The degree of detail of the report can be refined to the degree of detail of this WBS. A WBS can be included in the Test Cube and be maintained by the testers during the registration of their progress.



A practice example of a WBS related to the specification phase:

10%	identify test case
10%	specify logical test case
20%	specify test data
50%	define test script
10%	update Test Cube

By using such a WBS, more detailed progress reports can be created.



7. TOOLS

In draft form, the Test Cube is a collection of additional data for test cases (see chapter 2). This data can be stored in several ways with several supporting tools.

7.1 Spreadsheets

The cheapest and most practical tool is a spreadsheet (for example MS Excel). Spreadsheets are standard application at most of the workplaces, are easy to use and offer powerful calculation functions that make quick and accurate reports possible.

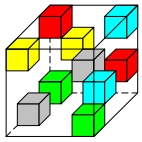
A simple Test Cube may look as follows (more extensive examples are available at the members of the Test Cube expertise group):

release 1.1	release 1.2	release 1.3	Deleted	Identification	TG Name	Weight category 1	Weight category 2	Weight category 3	Object part	Test date	Tester	Result	Defects
x	x	x		1	Enter name	x	x	x	Entry	23-01-05	PWN	OK	
	x			2	Enter name with punctuations			x	Entry				
	x	x		3	Empty name		x	x	Entry	23-01-05	PWN	NOK	437
x	x	x		4	Text file input	x	x	x	Interfaces	05-02-05	HDT	OK	
	x	x		5	Spreadsheet input		x	x	Interfaces	05-02-05	HDT	OK	
			x	6	etc.								

- In the example above, weight categories are included in the Test Cube as separate columns, and not as separate values in one column. This has been done to simplify selecting the test cases.
- The example clarifies the object part *Entry* is tested in release 1.1 with weight 1, in release 1.2 with weight 3 and in release 1.3 with weight 2. The data concerning the test execution are related to the current release. This data will be archived during the transition to a following release.

When using spreadsheets it is important to monitor that the administration of the test cases in the Test Cube (the spreadsheet) continues to run synchronously with the actual specifications in, for example, text documents. Another risk is that the workability of the spreadsheets decreases with the size of the test set and associated calculations.

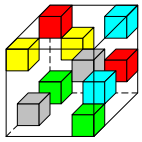
On this last point, database systems (for example MS Access) are in the advantage, but the disadvantage of a database system is that a considerable amount of programming is needed in advance and its data-entry is more difficult compared to data-entry for spreadsheets. Also, such tools are frequently not installed on a standard basis and many organizations discourage uncontrolled use of it.



7.2 Test specification tools

When an organization uses a test specification tool (for example TestDirector), the Test Cube can be classified in the same tool. Condition is that the tool offers the possibility to extend test cases with self-defined fields (necessary for the classifications of the Test Cube). These tools frequently have standard report functions which can also be used for the Test Cube.

The advantage of a Test Cube incorporated in a test specification tool is that registering the additional characteristics and the progress goes in many cases automatically.



8. THE TEST CUBE IN PRACTICE

The Test Cube has been developed and applied for the first time at one of the customers of Sogeti Nederland B.V. The following sections contain a report of the use of the Test Cube in this practice case.

8.1 Starting situation

Customer organization

The concerning organization is the largest player in its industry in the Netherlands. However, competition became tougher. In order to stay one step ahead of the competition, it was necessary to be able to quickly implement new functionalities. Beside this 'time-to-market' challenge, the customer experienced problems with the quality of the produced systems.

The system to test

The system that needed to be tested was the core application of the customer. The information in the system was the capital of the organization. It was a complex system with interfaces to several external parties with different requirements and preferences.

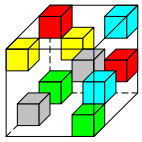
It concerned a package application that was developed by a foreign software supplier. The customer needed new solutions quickly that were not included in the standard package functionality, which led to more and more tailor-made solutions. The supplier experienced some difficulties in providing these tailor-made solutions quickly and with sufficient quality.

The system development process of the customer is deficient as well. The requirement process was not under control. Functional modifications were executed late in the development process, and the documentation was poor. Knowledge of the functionalities of the system resided mainly in the heads of different people.

The test process was characterized by long lead times (five weeks for a regression test), insufficient control and a badly maintained and obsolete test set.

For the new releases, first of all a new test set was needed. By approaching this test set as deliverable, the customer was motivated to specify requirements which the new test set should meet:

1. The test set should be able to be used more quickly with a justified coverage.
2. The test set should be able to be used by testers without knowledge of the subject matter.
3. The test set should be flexible.
4. Testing should be possible based on the risks recognized in a release.
5. The test set should be able to be easily converted to an automated test set in the long run.
6. The test set should be transmissible and maintainable.



8.2 Solutions

To satisfy the described requirements, the following solutions were invented:

1. The test set became modular at test case level. The principle of modularity implies that test cases can be executed independently of each other (requirements 1 and 5).
2. The test cases were specified at such a detailed level, that a person without knowledge of the subject matter could execute them and they could be automated relatively easily (requirements 2 and 5).
3. The number of test actions per test case was minimalized (requirements 1 and 3).
4. The test cases were classified (requirements 1, 3 and 4).
5. An administrative system which should maintain the test set was set up. This administrative system contained a repository as well, with which it was possible to infer which test case covered which functionality (requirements 4 and 6).

By implementing these solutions and further refinements of it, an entire new approach for managing test sets arose: the Test Cube.

8.3 Results

By applying the Test Cube, this customer gained the following advantages:

- The lead time of test specification and execution is significantly shorter (80% shorter than before) by:
  - modular test sets (parallel test execution)
  - reuse of test scripts
  - composition of regression test sets based on risks (justified coverage with much less test cases)
  - efficient use of additional test resources (the Test Cube makes transparent to what extent an additional test resource contributes to the shortening of the lead time).

  Testing was no longer an obstacle to quick time-to-market.
- Test execution has become cheaper by using testers without knowledge of the subject matter. There are fewer problems at filling in the resources.
- The impact on costs and quality of interim modifications of the test strategy is immediately transparent.
- The supervision of the test execution has become easier by lacking dependencies between test cases.
- The progress of test specification and execution is transparent during the entire test process.
- Adequate reports based on business risks have become available for the customer and other stakeholders.
- Metrics (figures) have become available with which test processes can be estimated and planned accurately (to within half a day).

The biggest compliment for the Test Cube was made by a manager of the customer. For him, testing was not longer the most intangible activity in the IT-organization: thanks to the Test Cube, testing had become for him the only *really* transparent project activity.