Frank Witte

Metrics for Test Reporting

Analysis and Reporting for Effective Test Management

Springer

Metrics for Test Reporting

Frank Witte

Metrics for Test Reporting

Analysis and Reporting for Effective Test Management

Springer

Frank Witte
Landshut, Germany

ISBN 978-3-658-44005-3 ISBN 978-3-658-44006-0 (eBook)
https://doi.org/10.1007/978-3-658-44006-0

This book is a translation of the original German edition "Metriken für das Testreporting" by Frank Witte, published by Springer Fachmedien Wiesbaden GmbH in 2018. The translation was done with the help of an artificial intelligence machine translation tool. A subsequent human revision was done primarily in terms of content, so that the book will read stylistically differently from a conventional translation. Springer Nature works continuously to further the development of tools for the production of books and on the related technologies to support the authors.

This Springer imprint is published by the registered company Springer Fachmedien Wiesbaden GmbH, part of Springer Nature.
The registered company address is: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

Paper in this product is recyclable.

Preface

In software projects, one of the central questions that arises is how to effectively present one's own results to management. The project management, superordinate positions, and responsible decision-makers need to recognize the state of the project and where there is a need for action. The test management should show whether it can achieve the given goals or whether special measures need to be taken.

Unfortunately, reporting is too extensive on the one hand, but often devoid of content on the other, and unfortunately, management is often not willing to delve further into detail; they even try to avoid critical questions. Reporting can no longer keep up with the technical and content complexity required by new systems and current technologies in many cases.

It is a great challenge to process the essential data in such a way that one can effectively present a multidimensional and difficult message to its recipients. This has led to metrics, which graphically represent certain facts based on selected data, indices, and curves, gaining increasing importance lately. The central importance of software testing is increasingly recognized as project-critical and leads to the demand for metrics on the project status from the test manager. Metrics are like a thermometer: they are supposed to show how well or how badly the test project is doing. Through the correct interpretation and evaluation of metrics, one can document much better why certain problems have occurred in the project.

Measurements determine our lives, whether in construction, crafts, medicine, or industrial manufacturing. Early in the history of mankind, the measure was important to create orientation and to classify knowledge, to establish comparability, and to create a common understanding, a common compass for all observers. Measurements allow comparability of data and a classification of knowledge into a coordinate system. Measuring distances and distances is a basis for shipping, logistics, and traffic. In medicine, numerous values of the body are measured to determine the patient's state of health. To measure these values meaningfully, one must know which parameters must be used: the size, the weight, the body temperature, the blood pressure, the cholesterol level, or the body fat percentage are different sizes that can indicate problems at one point or another.

When working with metrics, index values are formed: The BMI relates height and weight to each other, thereby allowing the comparability of facts that consist of two sizes. This is also the case in a software project, and analogous to the Body Mass Index, there are also index sizes here that meaningfully relate different key figures to each other. These index values can be expanded to relate several values to each other at the same time. Weighting with factors allows for individual evaluation: The stock index says something about the overall economic situation. The DAX includes the share prices of the 30 largest German joint-stock companies, which in turn are weighted differently: On 2016-12-07, the share price of Deutsche Lufthansa AG was weighted at 0.58%, the value of Siemens AG, on the other hand, was weighted at 9.44%. This weighting changes from year to year. By attributing greater importance to certain values, the reporting of results can be better adapted to reality or to the message that one wants to convey in the reporting. Different weightings can make a result look better or worse, more reassuring or more dramatic.

Project cycles are tending to become shorter, new developments are becoming faster due to the advances in hardware development (processors, memory) and dynamic processes (e.g., in agile projects). This speed creates high expectation pressure on new software applications and extensions of existing features. This deadline pressure also affects testing activities. Projects have to be completed by certain deadlines, milestones and budgets are (unfortunately) subject to quarterly thinking and sometimes unrealistic return expectations. The planning is often not sufficiently detailed and does not adequately consider certain parameters and conditions. As a result, much quality and depth is lost, cost estimates and time estimates are too ambitious and stand in the way of sustainable, high-quality products. While globalization has often increased the pace of development cycles, this usually comes at the expense of product maturity ("banana software matures at the customer"). This problem is often recognized in development projects, but ultimately does not have enough solid and transparent arguments to argue accordingly with management or controlling.

I repeatedly notice that the project effort is underestimated because not enough attention is paid to problems in the processes and the organization. Technically, the problems are mostly solvable, but technology only accounts for about 20% of the total problems, 80% are hidden in communication, work organization, and the framework conditions. This fact is not sufficiently appreciated in the technology-heavy environment. In addition, there is a phenomenon that is inherent in human estimates: We underestimate the goals that we can achieve in 5 or 10 years, but overestimate what can be achieved in 1 or 2 years. However, intervals of 1 or 2 years are precisely the typical duration for projects in software creation.

Statements aimed at increasing software quality and minimizing risk, however, must be substantiated with solid data to indicate dependencies and impacts. Comparative values and sound numbers help in this regard. The more different indices are collected, the more accurate the picture one gets about a project, a product, or software. Numbers help to move from a rough, emotional impression to a defined, reliable, and robust measurement. Only with numbers can one create a secure basis for argumentation and obtain comprehensible statements. Many problems depend on various influencing factors that affect each other. Numbers, metrics, and analyses provide a better basis for making the right decisions and setting the crucial course early on.

A report of 20 to 50 pages, as is already required for a medium-sized software project, overwhelms the recipient with its volume of information. In a project review before a project director or manager, which should ideally take place once a week, one might have an hour to present the current project status: Here, one should get by with a few charts that can be explained in a few words. Unfortunately, there is an increasing trend to provide the metrics online only for "time reasons". This is also a trend where essential information is lost rather than saved due to perceived time savings. With individual activities, for example in reporting, review measures or in the creation of system requirements, the successes are not immediately visible. Therefore, some necessary activities are omitted or saved due to a misunderstood setting of priorities, which, however, makes the overall project much more expensive and lengthy in the end. Furthermore, it must be considered that the reporting of projects is increasingly condensed upwards. If there are 100 different larger development projects in a corporation, then the top management only wants to be informed in detail about those where penalty payments or cancellations are threatened, i.e. the projects where "the hut is burning".

In any case, a particular challenge is to highlight the essential statements with the help of metrics and to focus on the essentials. "Project traffic lights" and "fever curves" help with this because these representations are already known from everyday life. Many projects certainly do not run optimally, but just enough so that with a few weeks delay and a few percent cost overrun and a slightly lower quality than intended or a slimmed-down functionality, it is still possible to avoid being in the focus of top management. Because once you belong to the few disaster projects that are in the spotlight, you gain constant reportings and additional panic and, due to the constant updating of the error lists, you get even less done than you already do. In this case, you often experience activism: Many additional resources are pumped into the project in a short time, measures such as Saturday work and ordered overtime are taken, which often only acutely prevent the worst and ultimately are cosmetic because the actual problems usually lie much deeper and cannot be solved in the short term. But if you do not address the deep causes, you exhaust yourself in appeasing symptoms. The sooner projects are put on a healthy basis, the better the project goals can be achieved.

During the presentation, analysis, and evaluation of metrics, management generally likes to see linear or progressively increasing curves. However, in practice, there are usually far too many influencing factors to be able to present these optimal curve progressions permanently. To use the right metrics, interpret individual facts sensibly, and correctly assess progress, suitable parameters must be specified in a clear presentation.

Different values flow into metrics. First and foremost, it is important to consistently collect as many data as possible. Metrics can also compare different projects with each other. It is of little use to blame the project members if a project is not running properly. A metric clearly and neutrally highlights the real status. Therefore, it is also recognized that perhaps certain prerequisites were not given from the start and the project could not achieve higher goals.

In order for metrics from different projects to be compared at all, it is important to make common agreements within the company Otherwise, you are comparing apples and oranges. Therefore, each individual project should select those metrics from a company-wide set defined by a process group that are meaningful and necessary for the respective project. The required data for these metrics are regularly collected for the duration of the project. I see this aspect fulfilled in the fewest cases in operational practice so far, most often the wheel is reinvented in each project. The required company-wide defined set of metrics hardly ever exists or is so well hidden that the project manager and the test manager cannot find it and then have to come up with their own metrics again. In order not to have to start from scratch in such situations, some suitable metrics are described in more detail in this book.

In the projects, most data is not collected or archived. And if usable results were collected in a project, these data rarely flow into the planning and calculation of subsequent projects, the acquired data base is not further used. In doing so, valuable potential is wasted. Usually, the subsequent project is calculated as optimistically as the past project, which was barely managed and was only completed at the last minute with delays and cost overruns, with compromises in quality.

The "Lessons Learned Meetings" often only consist of 30 minutes of superficial considerations, only a few stakeholders (or those least familiar with the operational implementation) participate in the meetings, or a skeptical review is completely omitted. If necessary activities for the future are recorded at all, they are long forgotten by the next project.

It is particularly important to always ask about the goal of each metric and to have a common understanding of what a particular metric can indicate, what conclusions can be drawn from it and which cannot, and where the limit of each metric is. Individual metrics answer very specific questions, and if simply hundreds of numbers are collected, i.e., measurement without a goal at the end, if the benefit is not transparent to the developers and testers, a defensive attitude arises and only additional effort is seen. Metrics can always be interpreted and under these conditions, they provide a distorted image of reality.

Work in production has been tracked very precisely and examined with the help of metrics for decades. Nowadays, office work is broken down into numerous sub-steps, like assembly line work. The tailoring of processes, the cause-appropriate assignment of activities, and the increasing monitoring and control of individual work steps leads to an increased need for reporting, and this also affects testing activities. In contrast, a defensive attitude arises, and only additional effort is seen. Metrics are useful when all participants in the project understand their meaning and targeted optimizations are initiated and implemented based on the statements.

Over the course of projects, it can happen that individual questions change and different statements come into focus. The individual chapters of this book refer to specific questions related to testing activities. The most common and widespread metrics are mentioned. However, in individual projects, it may be necessary to delve deeper into questions or to extend metrics, collect special index values, and expand metrics. Therefore, in my view, it is necessary to provide test management with support in analyzing and interpreting collected data and to look very closely at where the problems in the project are and how they can be presented in a meaningful way to comprehensively and yet succinctly represent the current state and ask the right questions so that the right measures can be initiated, and proper decisions can be made. Used correctly, metrics can be very efficiently used for project controlling, used incorrectly they quickly lead a shadowy existence or are abolished as unnecessary effort. It is therefore of particular importance that management demands metrics and actively influences their use.

Metrics are intended to map properties of functions to numerical values. They should condense information as objectively as possible, reduce complexity, be prepared in a way that is appropriate for the recipient, and quickly and early identify problems and present these findings transparently. In a car, essential information (speed, rpm, time, warnings) is displayed to the driver on the dashboard. In an airplane, considerably more information is required for the pilot, but here too, the focus is on the meaningful condensation of data. Similarly, a test metric must succeed in showing essential data at a glance. The condensation of information is always a balancing act: there is a tendency to lose too many nuances, the desire for the "One Pager" is understandable, on the one hand, but on the other hand, complex processes and difficult issues also require detailed backgrounds. Often, far too little time is taken to investigate the causes. The abundance of information, on the one hand, and the desire for simple, clear statements on the other hand are among the main causes of communication problems in projects. It is important to provide consistent data, and sometimes statements have to be simplified and reduced because side conditions and special features can only be explained sensibly in a footnote, but would overload the metric and make the statements incomprehensible in the end. It is not always sensible to thoroughly examine and comprehensively explain every detail.

A significant cause of this information abundance is Moore's Law: Moore's Law states that the complexity of integrated circuits with minimal component costs regularly doubles; depending on the source, a period of 12 to 24 months is mentioned. By complexity, Gordon Moore, who formulated the law in 1965, understood the number of circuit components on an integrated circuit. Occasionally, there is also talk of a doubling of integration density, i.e., the number of transistors per unit area. This technological advancement forms a fundamental basis of the "digital revolution". Gordon Moore expressed his observation in an article published in 1965, just a few years after the invention of the integrated circuit. The term "Moore's Law" was coined around 1970 by Carver Mead. Originally, Moore predicted an annual doubling, but corrected this statement in 1975 to a doubling every two years. The rapid development of semiconductor technology in the early years had somewhat slowed down. In addition to the miniaturization of elements and the enlargement of wafers, what Moore called "cleverness" played a role in the early years, namely the art of intelligently integrating components on the chip. The limits of this cleverness were largely exhausted in the 1970s. Moore's then Intel colleague David House introduced an estimate of 18 months for the doubling of chip computing power, which today represents the most common variant of Moore's Law and forms the framework on which the semiconductor industry bases its development plans for several years ahead. In reality, the performance of new computer chips doubles on average about every 20 months, with certain fluctuations.

Moore's Law is not a scientific law of nature, but a rule of thumb based on an empirical observation. At the same time, one can speak of a "self-fulfilling prophecy" as various industries are involved in the development of better microchips. They must agree on common milestones (e.g., optical industry with improved lithographic methods) to work economically. The formulation of Moore's Law has changed significantly over time. While Moore originally spoke of the number of components on an integrated circuit, today the focus is on the number of transistors on an integrated circuit, and sometimes even on the number of transistors per unit area. Although Moore's Law is occasionally seen reaching its limits, the 18 months is an average over several years, but generally, progress in process and memory technology continues: The current processors of the major chip manufacturers are manufactured using 10-nm technology. 14 nm are currently the smallest structures in microprocessors. According to the "International Technology Roadmap for Semiconductors", the structures (transistors and interconnects) are expected to shrink to 5 nm by 2021. These orders of magnitude far exceed human comprehension. A nanometer is one billionth of a meter, which means a number with eight zeros and a one after the decimal point. Even the comparison to a human hair offers little more help. An average hair is more than 4000 times as thick.

Information literacy is a key qualification for dealing with problems in the modern, highly dynamic information society. It belongs to the area of social competence (soft skills) and generally includes a range of skills that enable the individual to handle information competently, efficiently— considering conditions such as time or programs—and responsibly. These skills relate to all aspects of problem-related recognition of a need for information, its localization, its organization, its targeted selection through analysis and evaluation, and its purpose-optimized design and presentation. From the requirements for information literacy, there arise needs for metrics.

Metrics should require little interpretation effort. Metrics should be stable, insignificant changes should have little effect on the result. The individual values must be comparable and analyzable, and the measures must be reflected in a timely manner, so that one can still intervene. The criterion of reproducibility states that the result of the measurement should not depend on the measurer and should not be manipulable.

This book is intended to help gain an overview of the various parameters, describe the purpose of use, and thereby determine the best evaluation parameters for your own, individual project, as these can vary depending on the type of question, project scope, or project progress. In my book "Software Testing and Test Management," I have comprehensively considered the entire topic of software testing. It is in the nature of things that I could only touch on some aspects and could not treat them in their full breadth in depth. With this book, I would like to focus particularly on a significant sub-aspect of test management, as its importance has greatly increased.

Frank Witte

Contents

1 Definition, History, and Benefits of Metrics

Abstract

Metrics are used to facilitate the reporting of project progress. They are used for control and forecasting in development projects and facilitate the understanding of complex relationships.

Measurement is one of the earliest tools that humanity has invented. Everyday life is unimaginable without such units, which form a common system for understanding and a basis for all sciences. The first measures developed with the beginning of trade, the production of everyday goods, or the emergence of construction activities. The more that was explored in natural science, the more the need arose for a clear and useful representation of new information. This information had to be not only correctly interpreted by others, but also comparable. The requirement for comparability has proven to be very important in the development of science. This requirement led to the emergence of the unit system based on the meter in Europe during the 18th century. The higher goal for the validity of the metric system is: “For all the world, for all peoples.” Any country that wanted to support international trade as well as scientific and technical exchange had to accept the system [Alyo2017].

Measurement is one of the essential prerequisites for standardization and globalization. Metrics establish comparability.

A metric (counting, measurement) generally refers to a method for measuring a quantifiable size. The “quantifiable size” in software testing can, for example, represent the testability of the software or the complexity or quality of the test cases. Metrics transform software testing into a measurable technology.

F. Witte, *Metrics for Test Reporting*, https://doi.org/10.1007/978-3-658-44006-0_1

The goals of creating metrics include:

- Evaluation of the software
- Uncovering errors and faulty processes
- Creating a basis of trust in the software
- Weighting and justification of test costs compared to the entire software production process

Metrics enable objectivity through the quantification of observations [Soge2011]. A metric is a measurement for quantitative evaluations or improvements of a process or product. **Key Performance Indicators** (KPI) evaluate quantifiable metrics, which assess the performance of the organization in relation to a goal

Metrics are used to control the test process. They underpin test consulting and are used when comparing systems and processes. A test metric is the measurable property of a test case or test run with the specification of the associated measurement rule, such as the test coverage degree.

Metrics should meet certain requirements so that their collection is regularly updated and not neglected:

1.1 Criteria for Metrics

A metric should meet the following criteria:

- Simplicity of collection
- Compact and objective description
- Predictions for orderly planning and forecasting of improvements
- Representativeness (the representation as a number can be meaningfully implemented)
- Unambiguity (many mappings are possible)
- Traceability
- Scalability

1.2 Quality Criteria for Metrics

Metrics should meet the following quality criteria:

- Objectivity: no subjective influence by the measurer possible
- Validity and consistency: the metric really measures what it claims to measure
- Reliability (reproducibility): repetition yields the same results
- Usefulness: the metric has practical significance

- Standardization: a scale for the measurement results exists
- Comparability: the metric is comparable with other measures
- Economy: the metric can be measured at reasonable costs [Kosc2011]

1.3 Success Criteria for Metrics

There are some important success factors for working with metrics:

- Metrics must be proactive and can be used in daily work. Metrics that have to be recorded retrospectively at the end of the month and whose relation to daily work is not apparent, or error numbers without analysis of the causes of errors, are reluctantly collected and appear only as satisfaction of a requirement, but are not taken seriously by the affected employees.
- Metrics must be simple and understandable in terms of their purpose and effects. Complex and elaborate calculation models are not suitable for the introduction of metrics, but only in the further course and for questions derived from the models.
- Metrics must be collected and gathered in the project and not by external planners or own "metric groups".
- The object of measurement must be clearly defined; it must be clearly representable what matter is being investigated. It is often at this point, which initially sounds relatively trivial, that reporting with metrics fails. This includes that the terms used in the organizational unit are clearly defined, preferably in a glossary, and that all stakeholders have the same ideas about a certain term. Successful communication is the essential element for this, and it is often neglected.
- Programs for metrics require training and precise instructions on how they should be determined. A high manual effort in the collection of metrics should be avoided, the determination, recording and consolidation of data should be as automated as possible.
- Working with metrics requires a detailed analysis and individual interpretation of the data without personal blame.
- Metrics must be demanded and actively supported by management. Successful projects and new insights based on metrics lead to greater acceptance of their use [Eber1996].

1.4 Validation of Measures

In the **validation** of measures, a distinction is made between internal and external validation. Internal validation is the proof that a measure is a valid numerical characterization of an attribute, by

- proof of fulfillment of the representativeness condition and
- checking the scale type.

An external validation serves as a prediction model:

- Hypothesis about the relationship between two measures
- Recording of the measured values of both measures on the same test set
- statistical analysis of results by determining parameters and testing universality
- Investigation of processes, resources and products
- isolated (internally) or externally (with environment)
- at different phases of the process
- objectively or subjectively, directly or derived

In metrics, a distinction is made between internal and external metrics. An internal metric is defined by the fact that it only measures properties within the object under investigation, whereas external metrics take into account interactions and interdependencies of the object with its environment.

Metrics are used both for retrospection and for forecasts. Test metrics serve test management to document the aspect of sustainable quality assurance in the software project.

A metric always only provides statements regarding a certain aspect under investigation. The determined figures are always meaningful only in comparison to the figures from other investigated program parts or values to other criteria.

1.5 Problems in the Software Development Process

In connection with the creation of new applications, problems typically occur in the following areas. Metrics should be suitable for investigating these problem areas and expressing them with the help of index values:

- Exceeding the planned project duration
- Incomplete realization of the desired quality
- Incomplete realization of functions
- Exceeding the planned project costs

1.6 History of Test Metrics

Activities in software testing developed around 1970, and soon test coverages were measured. As program reliability became increasingly important at that time, it was also necessary to measure found errors. Bill Hetzel described possible approaches to test measurement in 1993, and Stephan Kan also dealt with different measurement models in a study at the end of the 1990s. In 2006, Sneed and Jungmayr published the first German-language contribution on the topic of test metrics [Snee2010]. Meanwhile, the use of test metrics in software projects is generally widespread.

1.7 Steps to Set Up Test Metrics

It should be defined as early as possible in the project which metrics make sense and should be collected during the course of the project. Unfortunately, at the beginning of the project (as for many other activities), in practice, far too little time is invested in this necessary preliminary work and necessary definitions are underestimated. Especially things that seem trivial at first glance are often not given enough attention and lead to permanent problems in the project.

The metrics should already be defined in the test concept (Table 1.1). A subsequent introduction of test metrics is usually associated with additional effort, especially when you have to reconstruct values from history. If you only define metrics during the course of the project and start using them at a certain point, you do not fully show the development in the course of the project. However, it is still better to introduce a metric later than not to measure values at all.

Table 1.1 Steps to define test metrics

Serial no.	Steps to test metric	Example
1	Determine key processes and procedures to be measured in the test	Test progress tracking process
2	Use data to define the basics of the metric	Number of test cases planned for testing per day
3	Determination of the data to be tracked, frequency of data collection, and employees responsible for data collection	The number of completed test cases is collected at the end of the day by the test manager
4	Calculation, management, and interpretation of the defined metrics	Number of test cases conducted per day
5	Identification of potential improvements based on the interpretation of the results	The number of test cases conducted is lower than expected, the causes must be analyzed and countermeasures proposed

1.8 Lifecycle of Test Metrics

In general, all test metrics have the following process:

1. Analysis:
   Identification and definition of the test metrics to be used
2. Communication:
   - Explanation of the necessities to the stakeholders and the test team
   - Explanation of the data collection procedures
3. Evaluation:
   - Collecting and verifying the data
   - Calculation of the index values
4. Report:
   - Creating the test report using the metrics
   - Informing project management and collecting feedback

References

[Eber1996] Ebert C, Dumke R (1996) Software-Metriken in der Praxis. Springer, Heidelberg

[Soge2011] TPI Next (2011) Geschäftsbasierte Verbesserung der Testprozesse. dpunkt-Verlag, Sogeti

[Snee2010] Sneed B (2010) Seidl (2010) Software in Zahlen. Hanser-Verlag, München

[Kosc2011] Koschke. Softwaretechnik, Fachbereich Mathematik und Informatik, Universität Bremen, WS 2011/12. https://www.informatik.uni-bremen.de/st/lehre/swt11/metriken.pdf. Accessed: 05.Okt. 2017

[Botr2006] Projektübergreifendes Testmanagement am Beispiel des Produkts „Quality Center" in einem Finanzunternehmen, Bachelorarbeit Sinan Botros im Studiengang Informatik. Hannover (2006) http://www.se.uni-hannover.de/pub/File/pdfpapers/Botros2006.pdf. Accessed: 07. Okt. 2017

[Alyo2017] Alyokhin V. Management von Softwaresystemen, Systembewertung: Metriken und Prozesse. http://www4.in.tum.de/lehre/seminare/hs/WS0506/mvs/files/Ausarbeitung_Alyokhin.pdf. Accessed: 12. Okt. 2017

2 Classification of Metrics

Abstract

Metrics can be divided into different categories according to several classifications.

2.1 Key Figures

Key figures are numbers that summarize measurable, economically relevant data. Thus, they bundle several data into a meaningful size. With the help of key figures, companies or organizations can be evaluated and results, for example from previous years, can be compared. At the same time, companies and administrations can also measure themselves against other organizations using these key figures. Key figures are usually based on aggregated and concise information. Roughly, key figures can be divided into

- absolute key figures: e.g. travel time, total costs, personnel capacity, average weekly turnover
- relative key figures (ratio key figures, index values), including
  - dimensional relative key figures: e.g. unit costs, expenses per day, turnover per customer
  - dimensionless relative key figures: e.g. price index, stock index, degree of completion, percentage share, return on sales
- Inventory key figures: e.g. sickness rate, market price, temperature (validity at a specified point in time)
- Progress key figures: e.g. trends and average values (validity for a specified period)

F. Witte, *Metrics for Test Reporting*, https://doi.org/10.1007/978-3-658-44006-0_2

Key figures summarize measurable, relevant data and put them in a larger context. They are quantitative information that is prepared for the needs of analysis and control providing

- the measurement of operational processes,
- the assessment of company-relevant facts,
- the brief and concise presentation of complex facts,
- the determination of future standards as well as
- the determination of critical success factors.

A key figure value is the value of the key figure at a certain point in time (for example, number of employees on December 31, 2016) or over a specified period (for example, profit in a fiscal year). For key figures, names such as -share, -factor, -degree, -index, -coefficient, -ratio, -number and similar additions are typically found, which are partly reserved for special types of key figures according to the metrological standards. Strictly speaking, a key figure is not meaningful in itself; what is always meant is a key size, i.e. the product of characters (e.g. letters, also in combination with mathematical characters or special characters in ratings [e.g. AA+] or a number in numerical information and its unit (e.g. Celsius, Euro, Meter).

In mathematical statistics, there are various key figures. With these key figures, for example in descriptive statistics, it is possible to get a good overview of distributions, averages, etc. with few quantitative data. Following some examples of statistical key figures.

Arithmetic mean: Mean value, which is calculated as the quotient of the sum of the considered numbers and their number. The two numbers 1 and 2, for example, have the arithmetic mean 1.5 (= (1 + 2)/2). In statistics, the arithmetic mean of a sample is also called empirical mean.

Kurtosis is a measure of the steepness or “sharpness” of a probability function with a peak, statistical density function or frequency distribution. The curvature is the central moment of fourth order. Distributions with low curvature scatter relatively evenly; in distributions with high curvature, the scattering results more from extreme, but rare events.

Median: The median or central value is a mean value in statistics and a location parameter. The median of a list of numerical values is the value that is in the middle (central) position when the values are sorted by size. For example, for the values 4, 1, 37, 2, 1 the number 2 is the median, namely the middle number in 1, 1, 2, 4, 37.

In general, a median divides a data set, a sample or a distribution into two (equally large) halves, such that the values in one half are not larger than the median value, and in the other not smaller. The median is, for example, often used for the distribution of household wealth in a country.

Mode: In stochastics, a branch of mathematics, the mode or modal value is a measure of the distribution of a random variable or a probability measure. The mode belongs to the location measures and thus has to characterize the position of a distribution, like the expected value and the median.

Variance: In statistics, variance is an important measure of dispersion for the probability distribution of a real random variable. It describes the expected square deviation of the random variable from its expected value. Thus, the variance represents the second central moment of the random variable. The square root of the variance is called the standard deviation of the random variable.

2.2 Indicators

Indicators are quantitative information that allow conclusions to be drawn about the expression or change of other important variables as a substitute size. Indicators are often not directly measurable and are also referred to as “soft factors”.

Key figures and indicators are often used for decision-making, strategic planning, and hedging. A basic distinction is made between absolute and relative key figures, which are usually taken from balance sheets or other documents. Key figures and indicators represent and map values and processes that serve as a basis for decisions. In addition to its application for management or controlling, this method can be transferred to development projects.

An important question that project leaders and test managers ask themselves before the test is what characteristics a project has, and which parameters influence the project to what extent. This complexity is often expressed through experience values and intuition. However, the complexity of a project significantly determines the scope of testing and the selection of test methods. To document this objectively and comprehensibly, key figures and so-called expert audits can be used to make the experience of project staff visible and measurable. It will become clear that one cannot speak exclusively of the term “key figures”, as not all factors for the complexity of a project are based on measurable sizes, but often estimates, assessments, and opinions are also included. This means that indicators are also used. However, it turns out that it makes sense to evaluate projects not only technically measurable with key figures, but also on a socio-technical level through indicators to depict the real influences.

2.3 Classification of Test Metrics

A rough classification divides metrics into process metrics and product metrics:

- **Process metrics** evaluate the test as a service and express index values such as test coverage or the error detection rate.
- **Product metrics** evaluate all elements of the test (concept and design documents, source files, test cases, etc.) and their relationship to each other.

Some metrics are both process and product metrics:

- **Project metrics** are used to measure the efficiency of a project team or a tool used in the project.

Another form of subdivision of metrics leads to the following groups:

- Requirement-based metrics (assessment of test scope, testability)
- Metrics based on test cases (test specification, test execution, development of test coverage, quality of test cases)
- Metrics for test automation
- Test productivity metrics (cost-based metrics, test effectiveness)
- Error-based test metrics (number of errors, error density, development over time)

Content-based metrics check the information content of individual requirements. These are metrics such as completeness, redundancy-free, and necessity of requirements. Combined metrics are composed of several metrics. Examples of combined metrics are metrics for modifiability or testability.

Depending on the **test classes**, the following metrics emerge:

a) **Module test**:
   - Size and coupling to other modules
   - Control flow complexity (path complexity)
   - Interface complexity (number and type of parameter per function call)
   - Data usage complexity (artificial parameters per combination)

b) **Integration test**
   - **Interface density** (generation of interfaces between system components)
     The test cases used here can be derived from the interfaces between the individual components. These must be generated during test execution and create higher complexity with increasing number. Therefore, care should be taken during system design to use only as many interfaces as are absolutely necessary.
     Interface density = Number of components/(Number of components + Number of interfaces)
   - **Interface width** (component interface versus Message interface)
     Here, two types of interfaces are distinguished. In a component interface (e.g., CORBA, RMI, RPC), parameter lists are passed. In the case of a message-oriented interface, a closed message is passed (e.g., XML, SOAP). In this case, a metric from communication technology (bandwidth) is used and the number of characters that are transmitted at once is measured. Furthermore, the number of (logical) attributes is determined. The interface width can thus be determined as follows:
     Interface width = Number of interfaces/(number of interfaces + number of attributes)