intetics

Where software concepts come alive™

The ultimate guide to measuring software quality

Experience-based advice for defining the measurement program, accurately analyzing source code quality, and writing effective SLAs.

Why quality?

1 Why is source code quality important?

In 1998, $328 million was lost when Mars Climate Orbiter's faulty software miscalculated the landing and the Orbiter disintegrated. The project that took two years to complete was gone because of one software glitch.

Similarly, in 2005 the Michigan Department of Corrections granted release to 23 prisoners and an undisclosed amount were kept in jail past their release date because of a computer bug.

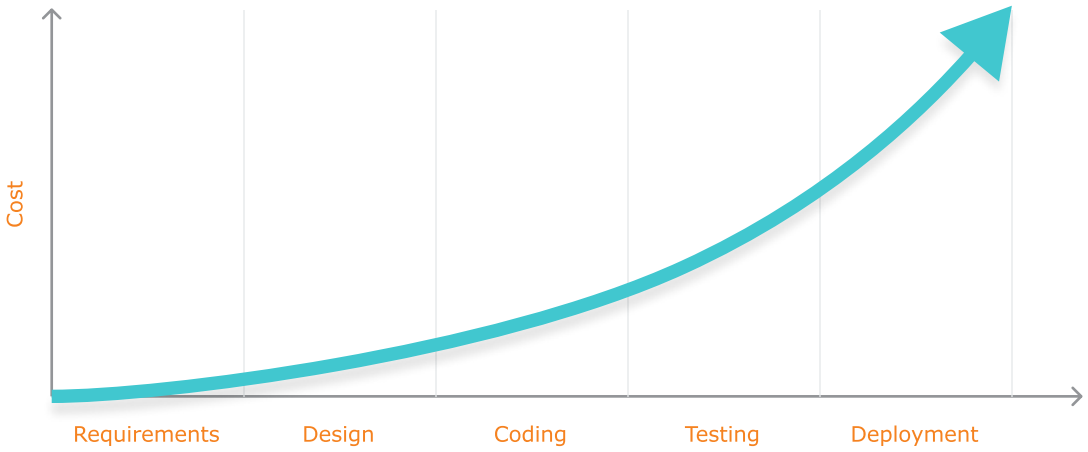
More recently, Apple's 2012 blunder with its new, inaccurate map app that came with its Apple iOS 6 upgrade reminds us that low quality software can have a negative impact on the business value of even the most successful brands.

These are just a few examples of what happens when software doesn't work as intended. Software quality can affect many things and people – from loss of knowledge and time, to loss of money, and sometimes even to loss of lives.

2 Costs of software quality

Faulty software has a huge impact. The cost of failed IT projects globally is estimated to be an astounding $3 trillion USD. The cost of software quality is all the money spent trying to prevent projects from failing. Philip Crosby famously claimed that "software quality is free", but anyone working in software knows that extra resources are a required for a quality product.

So despite quality being theoretically free, it comes at a cost. The cost of software quality is actually the cost of not creating a quality product or service. In other words, not having quality software can get very expensive, very quickly. The best way to keep cost of bad quality down is to address the problems during development, since costs of undetected defects rise exponentially after deployment (illustrated in graph below). The costs of bad quality include internal and external failure costs, losing business, customer reparations and complaints, lost customers, lawsuits, and even lower brand equity value.



In order to avoid these costs, companies invest a lot of money into getting software right the first time. The quality costs they incur can be considered prevention costs: activities such as new product review, quality planning, process evaluation, and appraisal costs that include rigorous in-development evaluation and quality audits. These costs are all parts of what it takes to create quality software.

Since software quality affects all facets of business, it is not the sole concern of software developers, but it is also an issue for software sponsors and users. For that reason, companies also demand SLAs (Service Level Agreements) that detail the quality process. If a client receives bad software and there is no comprehensive SLA, they may be stuck paying more.

3 Business value of quality management

Effective management of software quality can lead to lower costs and more efficiency. Yet, quality in software does not come easy. It requires constant monitoring of software engineering processes and methods. Universal and generic in nature, these QA (Quality Assurance) processes and methods can be applied across many domains. Software code quality is one of them.

Quality management can be achieved through different venues, for example by following the ISO 9001 certification guidelines. It can lead to significant improvements in almost all aspects of business. These benefits include:

INTERNAL ORGANIZATIONAL BENEFITS

- Make more informed, effective decisions.
- Attain continual improvement, with more flexibility to respond to opportunities and better organization and alignment.
- Create better processes that lead to lower costs and consistent results.
- Achieve desired results and focus on key processes.

BETTER BUSINESS RELATIONSHIPS

- Improve products and customer focus, increasing customer satisfaction.
- Better leadership, less miscommunication, and more motivation due to established practices.
- More commitment and accountability from employees due to clear objectives.
- Effective cooperation with suppliers, as the QMS helps set clear goals, respond to market needs, and optimize costs and resources.

In short, an established process for measuring quality will help avoid costs of bad software, create better products, and helps businesses gain significant organizational advantages.

SLA: Quality as an obligation, not an abstract promise

Measuring quality is critical to developing valuable, long-lasting software. It is especially important when working with third parties or remote teams. Working with external teams necessitates writing of legal contracts, where quality has to be rigorously defined. To ensure legal compliance, companies write service level agreements (SLAs). SLA is a formal document outlining a service commitment, for example by a software development provider to their customers. Among other things it is important for these documents to outline attainable and measurable quality goals, so each party can easily assess whether the negotiated value was delivered or received.

SLA FOR GUARANTEED QUALITY

SLAs were in part created to ensure legal obligation towards software quality. Without rigorous measurements of code metrics and other important quality metrics, SLAs rely on very abstract language. With expressly defined quality standards it is much easier to assess whether the promised level of service has or hasn't been received (such as time, quality, complexity etc.). It also makes it easier to decide what corrective actions should be taken.

When one party has a mature comprehensive quality management system in place, it makes it easier to determine which standards are detrimental to the project (essentially, each party should pick and choose the metrics that are most important to track for the project). It also makes it clear what quality metrics will be measured and how the analysis will be conducted. The clarity of the process, in turn, makes it much easier to write enforceable SLAs, giving clients more control over projects. The more the SLA specifies the quality process, the more it is likely that a successful result will be obtained. Having a comprehensive way to measure quality, or working with a company that has one, is thus a very useful tool to guarantee product quality, and assure completion of project objectives.

THE IMPORTANCE OF QUALITY ASSURANCE TO COMPREHENSIVE SLA

Establishing a comprehensive quality management system that can help deliver quality is not an easy task. Nevertheless, many providers will claim to deliver software quality. The majority of these providers are talking about quality control: they will guarantee that they will deliver an app that works.

While having a working application is a big step, it tells the client nothing about the longevity and the real value of their application. Companies who only provide quality control are often unable to provide measurable and guaranteed SLAs. In fact, it is very hard to define "quality" for companies that offer only quality control. As a result it is harder for clients to assess their project's longevity and establish fail-safe clauses in their contracts.

Far fewer providers guarantee quality assurance (QA), which is what really helps prevent majority of quality costs. Quality assurance is a systematic approach to software development that looks at the entire development process in order to prevent future mistakes in addition to delivering working software.

Quality assurance concentrates on quality of the project and quality of the processes. A typical QA audit includes 3 levels of control: checking the quality of the product and corresponding processes (development, testing, etc), checking knowledge and ability of the programmers, and ensuring the suitability of infrastructure. Few providers have a comprehensive QA system in place that can measure all three aspects of a project accurately. Quality control may be enough when a simple application is being developed, but larger projects with complicated code may suffer in the long run without quality assurance processes.

Essentially, a QA process helps outline the standard procedures that will be taken to achieve quality. It establishes a reproducible way to track quality. The reproducible tracking can be used to objectively assess project goals. This in turn gives more power to the client and gives them a way to assess their project. The expected results of these analyses can be input into SLAs to objectively assess goal completion and hold the other party legally accountable if something goes wrong. In the end, the client is not stuck paying the cost of 'bad' quality if the provider doesn't deliver.

FROM SLA TO PARTNERSHIP

Since quality benchmarking depends on independent entities, each company and team is likely to have their own set of quality standards. That is why regardless of whether you're working with an in-house team or outsourcing a project, every team, company and partner working together must negotiate clear quality standards before project start. Once standards are determined, SLAs can be used as a way to entrench these expectations.

While accurate measurements are vital, it is also important to remember that working with a third party should be a partnership. Although hitting all the number goals is ideal, it is not always enough since there is always room to find something lacking – for example lack of depth in test case documentation. The results will be much more fruitful if the relationship becomes outcome-based rather than numbers-based.

What to measure

Benefits of quality metrics

Reliable and consistent metrics help objectively assess the state of software and the development team, as well as efficiency of the processes. There are at least 5 major benefits of clearly identifying quality metrics:

- **Predict speed and schedule**

The speed and schedule are more predictable when the development process is measured.

- **Better team productivity**

Metrics help break down the process and give the right tasks to the right skill level, resulting in better team productivity.

- **Objective progress evaluation**

Clear goals allow the team to assess progress and goal completion.

- **Anyone can work with anyone**

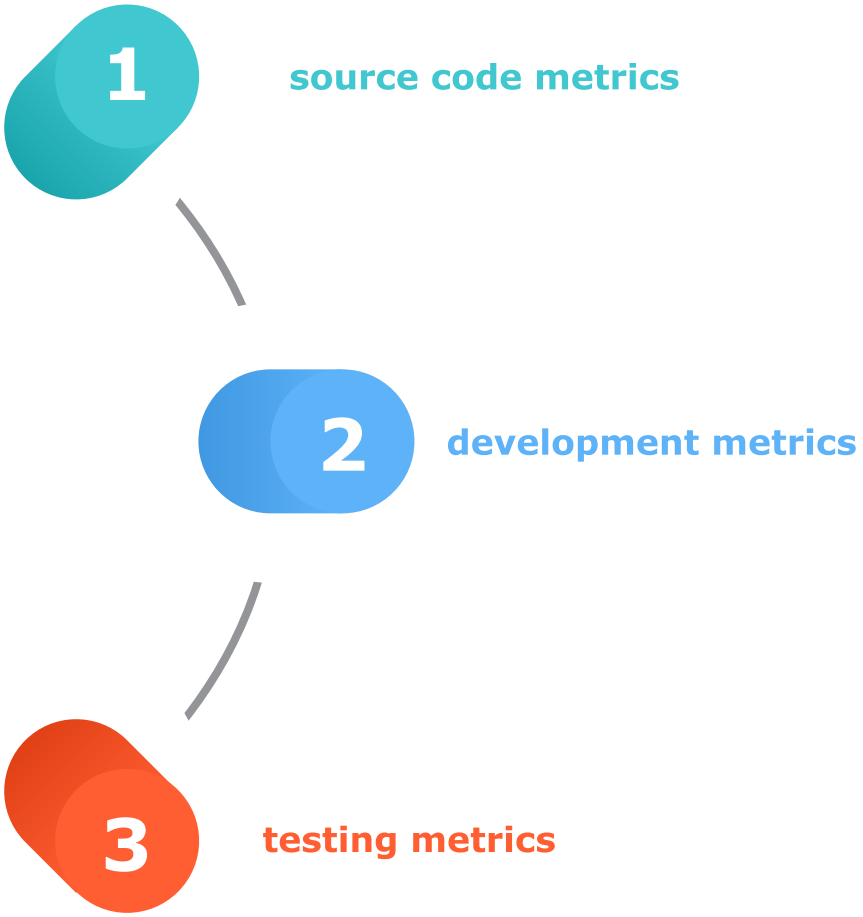
Clearly identified quality parameters make working with colleagues or remote teams smoother. Clear standards set the goals and expectations of the project, so everyone (regardless of location) knows what is expected of them.

- **Easy project planning and process improvements**

Quality metrics make it easy to recognize and plan enhancements.

Defining what to measure

ISO and most industry best practices do not explicitly tell companies what to measure and what levels to achieve, plus each project might have their own requirements. In general however there are three main types of metrics that are beneficial to measure. They are



Each category breaks down to more specific measurements. We explain each below, noting our own standards (though these standards tend to be higher than industry average).

1 SOURCE CODE METRICS

The most basic component of software quality boils down to receiving quality code. By measuring certain aspects of software code, it becomes possible to quantify software quality, assess its effectiveness, and discover potential improvements and problems.

Source code metrics measure the quality of the code itself to establish the quality of the product. There are "seven axes of code quality": code complexity, unit test coverage, duplications, coding standards, comments, potential bugs, and overall design & architecture. When these are accurately measured we get an accurate representation of product quality.

- **Lines of Code**
  Number of physical lines that contain at least one character, which is neither a whitespace, a tabulation, nor part of a comment. Number of lines should be appropriate for the software product being developed. It tells us about the size of the program.

- **Percent of Comments**
  Percent of comments inside source code. It is good practice to leave comments in the source code to allow easy maintenance. This metric suggests how easy it will be to maintain the source code in the future.

- **Cyclomatic Complexity**
  Complexity of the program. Cyclomatic complexity is a measure of the number of linearly independent paths through a program module. This determines the minimum number of inputs needed to test all the ways of executing a program. Overly complex modules are more prone to error and are harder to understand, test, and modify. The goal is to achieve least complexity. Obviously, the bigger the application, the more complex the code, but it is important to keep complexity under control.

- **Rules compliance**

  Accuracy of source code. It measures how accurate the source code is in relation to the standard usage of each programming language. Intetics best practices suggest that more than 80% of code should be compliant with standard usage rules.

- **Duplications**

  Number of physical lines touched by a duplication. Intetics in-house standards strive to keep duplications at less than 4%. This reduces complexity and improves system performance.

- **Code Coverage**

  Percent of source code covered by unit tests. Unit tests help test individual units of the source code to evaluate whether they are fit for use. This helps break down the testing process and test code in smaller pieces throughout development. While some consider 50% coverage optimal, the Intetics standard is that 70% of code should be covered by unit tests.

- **Test Results**

  The source code has to pass basic testing to ensure the final product is functional. This is most effectively done with unit tests throughout development. Testing can be done in a variety of ways. At Intetics there are generally two ways the testing is conducted: the developer writes a unit test and then writes the code (test-driven development), or a testing team checks the functionality, security, and performance of the application. This metric shows how many unit tests passed or failed. Our goal is for 100% unit tests to pass.

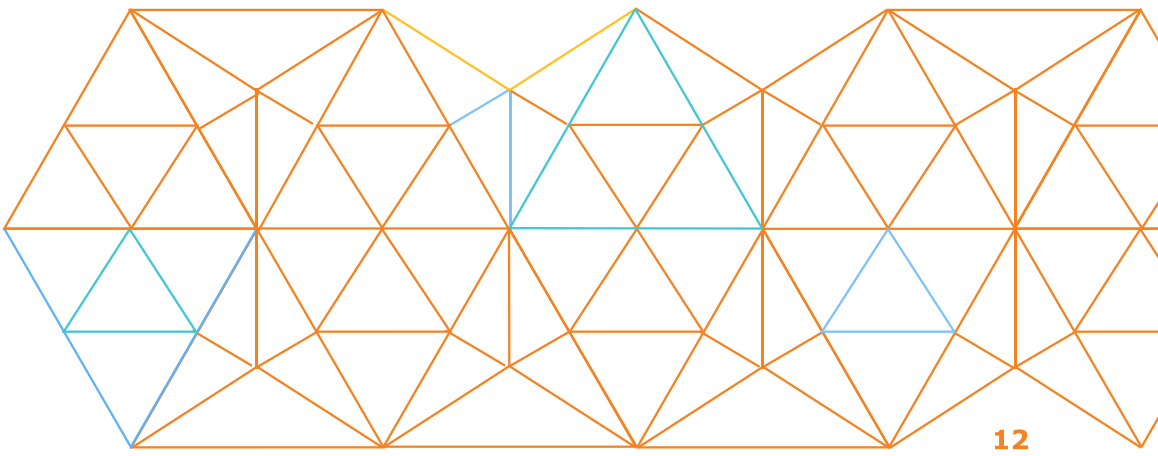
2 DEVELOPMENT METRICS

Development metrics provide information about the quality of the development process. They give great insight into where there is an issue in the process and where it can be improved. Three metrics can reflect the quality of the development process:

- **Defect Density**
  Number of bugs that appear in code. This number should be relatively low, but gives a quick picture of the functionality of code and helps estimate deadlines.

- **Defect Life-time**
  Amount of time it takes to fix bugs. Defects should be fixed as soon as possible. If there's a delay in fixing defects, that suggests that the problem is more complex or that a developer needs help.

- **Reopened Defect Density**
  Number of reopened defect notices. The goal is to have a small reopen density. If the reopened density is high, it means the team is working on the same defects without completely fixing them. A high reopen density may point to bigger problems, such as problems within the team, deadlines, understanding the goals or mastering specific skills. If it's high it may be necessary to check the team's skills and investigate whether they understand the goals of the project.



3 TESTING METRICS

Testing metrics measure the quality of the entire product. They are the final step in assessing software quality and certify that the product works. The two metrics, defect removal efficiency and test coverage, can help identify the success of testing. Testing should take place throughout the development process, until the required goals and acceptance criteria are achieved.

- **Defect Removal Efficiency (DRE)**
  It is calculated as a percentage of the defects identified and corrected internally with respect to the total defects in the complete project life cycle. Defect removal efficiency shows how useful or effective quality management activities are on phase-by-phase basis and it can be used as a measure to set process and product improvement goals.

- **Test Coverage**
  Proportion of the program covered by a test suite, usually expressed as a percentage. This involves collecting information about which parts of a program are actually executed when running the test suite in order to identify which branches of conditional statements have been taken.

How to measure: real-world quality analysis

You've selected the metrics, now how do you actually measure them? Defining the right metrics is only onestep in assuring software quality. The second step is to create an automated process that can track and analyze the collected metrics. This process needs to be perfected and adopted, and includes considerations such as choosing the right tools, process automation, accessibility and customization. This will help deliver not only working code, but also assess the progress of development and longevity of the product.

Intetics saw great value in developing its own quality measurement process and as a result created its own proprietary Quality Management Platform (QMP). The QMP effectively deals with code metrics analysis, accurately measures quality of the product and helps Intetics developers conduct test-driven development with an Agile approach on every project.

Creating a successful Quality Management Platform (QMP)

The most important part of measuring quality is to use convenient tools that allow flexibility, reproducibility, automation. To that end Intetics created a platform that consists of three main parts. First, there is a version control system (SVN, Git, etc), which stores the source code. This local repository is connected and integrated with a continuous integration tool (Hudson) and fed into Sonar, a program that conducts the analysis of the source code. Second, the code is checked against the Sonar database for things such as lines of code, percent comments, cyclomatic complexity, rules compliance, duplications, code coverage, and test results. Finally, the QMP shows the analysis results in a comprehensive dashboard. In addition, the QMP has the following features:

1

Combines analysis of multiple programming languages into one QA platform

In a single project there may be multiple source code languages, and multiple tools may be needed to assess the quality of the code across all languages. This can significantly slow the process and increase costs, unless there is a comprehensive platform that can be used for every type of project.

Instead of using a different collection of tools and measurements, Intetics QMP is based on open source software (Sonar) and is able to work with multiple programming languages at once. Tools such as CheckStyle, PMD, and FindBugs, etc. are built in, eliminating the need for multiple tools.

2

Measures code quality consistently

Measuring quality is not a one-time thing. Rather it is a process that will only help improve quality of code if it accurately reflects the situation throughout development. Intetics QMP collects and measures the 7 main types of source code metrics discussed in detail above. The consistent and accurate measurement across all projects helps conduct regular analysis, and ensures consistency as all measured values are comparable, repeatable and reproducible.

3

Uses tools that are accessible to everyone

Allows accessibility to code analytics for programming and testing teams. Many code quality testing tools are tailored for use only by developers, and testers often cannot use the same set of tools. These tools also replace the myriad other tools available to developers. The integration of tools makes the quality management process more effective and understandable to the entire team.

4 Analyzes source code in real-time

The QMP integrates with continuous integration servers for real-time analysis of source code quality with effective visualization and reporting capabilities, which helps automate and conduct regular quality measurements. This unifies the information provided to the entire team, so everyone knows the state of the code. The dashboard is customizable, provides easy access to understanding all risk areas, and conveniently displays where source code errors occurred and what parts need to be improved.

5 Customizable depending on project needs

The main tool used by the QMP, Sonar, is highly customizable. Intetics quality assurance specialists customize the Sonar interface to reflect the needs of every project. If Sonar is properly configured, it can measure the 7 source code metrics very accurately. The system measures the general trends of source code quality, and it also sub-divides the metrics based on specific parts of the project to show problematic areas. This makes it easier to fix and analyze problems.

Developing a comprehensive set of tools for quality management may be one of the best ways to ensure the process stays accurate and consistent. For many organizations, it takes time and an understanding of how to measure source code quality before a similar tool is developed and adapted to accurately reflect the actual state of software source code. At Intetics, the QMP has become an integral part of the development process, used on every one of our client's projects.

Results of Intetics QMP

In general, the QMP has at least three notable advantages:

1

Tracks, analyzes and improves code quality

First and foremost, the QMP helps assess the state of code quality on any given project and provides the necessary improvements.

- It allows automated analysis and source code reviews.
- It enforces coding standards and serves as a useful centralized quality metrics repository.
- It allows tracing the root cause of poor code quality and identifies areas that need to be fixed in the coding stage often before production release, which significantly reduces the cost of development of the whole project.

As a result, it dramatically improves code quality and decreases probability of system defects.

2

The Code Quality Management Platform is used by all team members

- The main target group is developers. QMP helps them assess the condition of the source code and identifies parts that need to be improved.
- Testers benefit from the QMP as well. It assists them in determining which parts of the system lack unit testing and where, accordingly, they need to do more regression testing.
- Software architects use it to assess the initial design and complexity of the application.

- It aids managers' understanding of whether the system is improving (with metrics and graphics).
- Customers can also view the project dashboard so they can see if we have fulfilled the code quality requirements.

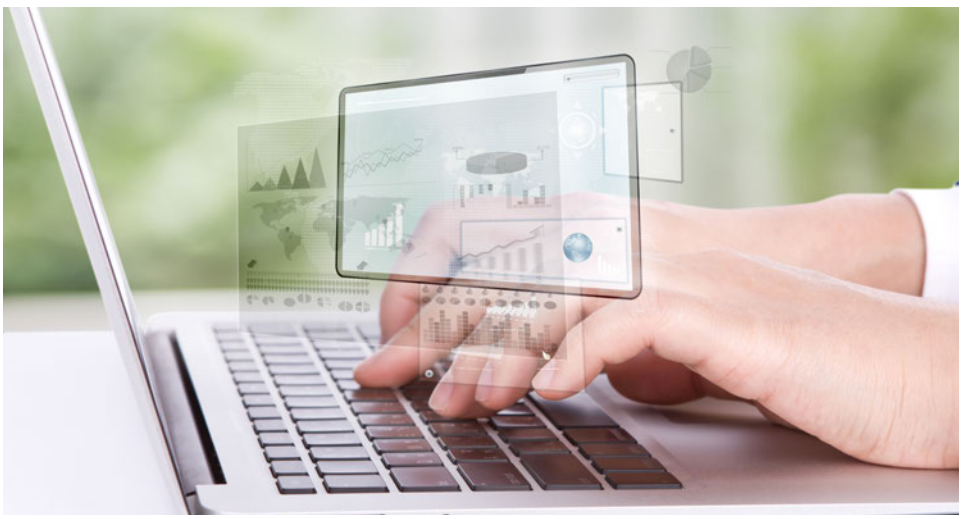
Since it integrates a number of tools into one platform, the QMP reduces the amount of time and effort spent on quality assessment for everybody involved.

3

Improves team performance and proficiency

Finally, the QMP keeps the coding process transparent and organized.

- Since the platform can be used by everyone on the team, it is easier to track the progress of the project.
- It creates a natural way of performing individual corrective actions and helps discover underperforming team members.
- It facilitates the sharing of best practices and allows developers to retain familiarity with the source code base.
- It enhances team proficiency and performance, as members and managers use the common webbased tool to track their own work and share best practices.



> ***If you can't control it, you can't improve it.***

The ultimate goal of software metrics and quality measurements is the ability to quantitatively evaluate the condition of a project and to arrive at a better result. Quality expert H. James Harrington wrote that "If you can't measure something, you can't understand it. If you can't understand it, you can't control it. If you can't control it, you can't improve it." Without a comprehensive quality measurement, there is no way to understand quality or to control it, but most of all – there is no way to improve your systems.

Measurement provides a quantitative indication of the extent, dimension, amount, capacity or size of some attribute of a product or process. Having selected the most important attributes for your product or process, quality measurement helps determine the progress toward meeting those goals. It quantitatively answers the question of whether or not the requirements were met.

Every person with goals has used a measuring system to determine if their goal was met and what steps were needed to reach that goal. SLAs are such measuring systems for business objectives. SLAs can use software quality metrics to assess which business and organizational goals are appropriate for the organization (given its particular mission) and which goals have been achieved. Similarly, development teams can use quality measurements to assess their progress and code quality.

It is up to each organization to define and measure quality. Yet, at the beginning of any project it is imperative to ensure that everyone participating understands the quality expectations – from writing SLAs, to creating a quality assurance process, and to training teams to use the right tools on every project. At Intetics we developed a whole platform to determine what goals our teams should achieve. While perhaps a whole proprietary platform is not what your organization needs, if your goal is to create quality software consistently, you need to determine and communicate the quality standards you expect. These standards should be quantifiable and replicable. Only then will you be able to create products that truly reflect (and often exceed) your expectations.

About Intetics

Intetics creates and operates effective distributed technology teams focused on software product development, IT support, quality assurance and data processing. Intetics enables IT rich, innovative organizations to capitalize on available global talent, based on a proprietary business model of Remote In-Sourcing®, advanced Quality Management Platform and measurable SLAs, and Intetics' in-depth engineering expertise. Our core know-how lays in design of software products in conditions of incomplete specifications.

Intetics is ISO 9001 (quality) and ISO 27001 (security) certified and Microsoft Gold Partner. The company's innovation and growth achievements are reflected in winning prestigious Deloitte Technology Fast 50, Inc 5000, CRN 100, Software 500 and European IT Excellence awards and inclusion into the Top 100 Global Emerging Service Providers and Top 100 Global Outsourcing Companies lists.

You can find more information at www.intetics.com.

intetics
Where software concepts come alive™

INTETICS MEANS YOUR SUCCESS

Toll Free: +1 (877) SOFTDEV
US: +1 (239) 217-4907
DE: +49 (211) 3878-9350
UK: +44 (20) 3514-1416
Email: intetics@intetics

www.intetics.com