B829844

Talysurf Surface Texture Instrument Software (TSTIS) Technical Report

This document is created to provide our client an assessment of the Talysurf software and to determine if this software is suitable for commercial use. We will be discussing the recommended quality requirements for this software, how they could be assessed, suitable set of tests, result analysis of tests, safety and reliability requirements. A software of substandard quality might cause financial lost from lawsuits or even lost of life.

The definition of software quality by IEEE [1] is "The degree to which a system meets specified requirements and customer's need or expectations.". [2] [3] Software developers often think that software quality is just how well the system operates e.g. code is written according to standards, software fits hardware specs, this could be assessed based on the CISQ model which we will explore later or others such as the ISO/IEC/IEEE 29119. However, we must also consider the customer requirements, such as customer satisfaction, conformance to requirements and variation with respect to specification.

Quality could be assessed based on the [4] CISQ model, which consists of 5 main factors. Reliability is the risk of failure when exposed to both expected and unexpected conditions. [5] This is also sometimes known as the stability of the software which describes how likely the software is to fail if changes to the code are made. A reliable piece of software could also be described as resilience, if the software is able to automatically fix the problem after it fails. This could be assessed based on rate of major bugs per 1000 lines of code with a value typically between 15-50 (See Table 1), load testing by creating a simulating use under high loads. The data could then be represented by MTBF which describes the average amount of time before the next software failure.

The second factor in the CISQ model is performance efficiency. This is the software's use of resources in the hardware, the software architecture and the ability for the software to be scaled up or down depending on the user requirements. Performance also directly correlates to customer satisfaction as a slow system will not be tolerated in a lot of applications. Load testing could also be used to measure performance by looking at how the system operates at its maximum specified capacity; Soak testing could be conducted to look at how the software would perform under long periods of time.

Security is how well the software responds against malicious attacks such as gaining access to confidential files, overloading the system causing it to crash and how easy is it for developers to patch vulnerabilities. We can measure the quality of security by checking the system logs if there are any breaches or the average time it takes for a patch to be implemented, the severity of pass security breach should also be considered.

Another important aspect in the CISQ model is maintainability, how easy its it to modify the software if required, either to upgrade or convert to use in other similar purposes, run in different operating system. Maintainability could be assessed by recording how many lines of code are written and checking if the code is writing to a standard such as the [6] Indian Hill C Style Guide. Static code analysis could also be performed which checks the code against industry standards without executing the software.

Finally, the rate of delivery [7] and usability is also a crucial aspect in the CISQ model. Rate of delivery is the number of updates provided by software developers, either to improve the quality of the software or patch security vulnerabilities, this is quite common in agile project management methods. Usability is the quality of the Graphical User Interface (GUI), how easy its it for a user to learn how to use the GUI and how fast is the response time. [8] To assess the rate of delivery we can just count the number of updates for a specific period, the quality of the GUI could be assessed using market research.

TSTIS calculate parameters and frequency spectral data obtained from a surface texture instrument, it allows the user to load the data into the program, calculate the frequency data, saves the spectral data, computes and display the parameters. The processed information could be used to measure the performance mechanical components, as this would allow engineers to predict interactions between two surfaces and probability of cracks/corrosion. This component could potentially be used at a critical system such as a nuclear plant or a chemical manufacturing plant in which any miscalculations may cause a catastrophic failure, therefor TSTIS

should be tested for reliability. This could be implemented by [9] importing a large set of data into the system and observe how the software responds to it, a stress test could also be conducted by calculating values using the same data and checking if the results differ. The accuracy of the result could be checked by importing a set of predetermined values and calculating the results using another method, then comparing the results. Performance could also be a major aspect in TSTIS, as it could be used in the manufacturing industry to test parts if they are up to standard before performing the next stage, major manufacturing plant could be processing up to 1000 parts per second. This could be tested by finding the typical data profile and measuring the processing time to determine if TSTIS is suitable for your application. Security, maintainability and rate of delivery are less important aspects for TSTIS as it will be mostly running on an isolated embedded system which is specialised for this application. As TSTIS runs on a command line user interface, usability should be evaluated. This could be conducted by asking technicians or industry professionals to trial the software and getting feedback from them.

To conclude, software quality could be assessed by reliability, performance, security, maintainability, rate of delivery and usability. I recommend our client to conduct tests to TSTIS outlined in this report and evaluate findings before selling it commercially.

Source	Language	Errors / KLOC	Life-cycle
Siemens - operating systems	Assemblers	6-15	Post-del.
IPL - language parser	C	20-100	**PRE**-delivery, therefore higher
NAG - scientific libraries	Fortran	3	Post-del.
Air-traffic control	C	0.7	Post-del.
Lloyds - language parser	C	1.4	Post-del.
IBM cleanroom	Various	3.4	Post-del.
IBM normal	Various	30	Post-del.
Loral - IBM MVS,	Various	0.5	Projected, not actual !
NASA Goddard satellite planning studies	Fortran	6-16	Post-del.
Unisys communications system	Ada	2-9	Post-del.
Ericsson telephone switching	?	1	Post-del.
Average for applications	all	25	Post-del.
Average for US and European applications	all	5-10	Post-del.
Average for Japan	all	4	Post-del.
Motorola	?	1-6	Post-del.
Operating systems, Akiyama, 1975	Assembly	20	Post-del.

Table 1: Number of defects in different languages [10]

Sources:

[1] Software Testing Fundamentals. (2019). Software Quality - Software Testing Fundamentals. [online] Available at: http://softwaretestingfundamentals.com/software-quality/ [Accessed 17 Nov. 2019].

[2] AltexSoft. (2019). What Software Quality (Really) Is and the Metrics You Can Use to Measure It. [online] Available at: https://www.altexsoft.com/blog/engineering/what-software-quality-really-is-and-the-metrics-you-can-use-to-measure-it/ [Accessed 17 Nov. 2019].

[3] SearchSoftwareQuality. (2019). An expert suggests how to measure software quality. [online] Available at: https://searchsoftwarequality.techtarget.com/opinion/An-expert-suggests-how-to-measure-software-quality [Accessed 17 Nov. 2019].

[4] It-cisq.org. (2019). Code Quality Standards | CISQ - Consortium for Information & Software Quality. [online] Available at: https://www.it-cisq.org/standards/code-quality-standards/ [Accessed 17 Nov. 2019].

[5] Sealights. (2019). A Comprehensive Guide to Measuring Software Quality. [online] Available at: https://www.sealights.io/software-quality/measuring-software-quality-a-practical-guide/ [Accessed 17 Nov. 2019].

[6] Maultech.com. (2019). [online] Available at: https://www.maultech.com/chrislott/resources/cstyle/indhill-cstyle.pdf [Accessed 17 Nov. 2019].

[7] Techopedia.com. (2019). What is Software Security? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/24866/software-security [Accessed 17 Nov. 2019].

[8] Guru99.com. (2019). Reliability Testing Tutorial: What is, Methods, Tools, Example. [online] Available at: https://www.guru99.com/reliability-testing.html [Accessed 17 Nov. 2019].

[9] Keyence.com. (2019). Measuring Procedure For Stylus-Type Surface Roughness Instruments | Introduction To Roughness | KEYENCE America. [online] Available at: https://www.keyence.com/ss/products/microscope/roughness/line/measurement_procedure.jsp [Accessed 17 Nov. 2019].

[10] Leshatton.org. (2019). [online] Available at: https://www.leshatton.org/Documents/FFF_IEE397.pdf [Accessed 17 Nov. 2019].