**1. Abstract**

With rapid development of software technologies, more and more development tools are invented to help people create applications while software applications are involved in every aspect of human society such as business, industry, entertainment, government, etc. Creating applications is easier and faster than before, their importance and influence is larger as well. Therefore, People have more concerns and focus about the quality of applications:“The quality control for the software applications have become the most important issue...It has been demonstrated by various researchers that, the poor quality of the software or the extreme negligence of software quality may lead to extreme high maintenance for the software applications”(1). The quality of a software not only represents its commitments to stakeholders, but also decides follow up costs for individuals or organizations who created it. Therefore, scientific methods should be used in the software development process in order to assure quality performance.

**2. Introduction**

The cleanroom software engineering process is a software development process allowing softwares to be engineered under statistical quality control. The statistical quality control can produce certified reliability data to the delivered application in order to evaluate and control quality performance. The first priority of the cleanroom software engineering process is to prevent defects instead of removing defects, while removing not prevented defects. Due to the significant difference between human’s mathematical and debugging fallibility, this priority is implemented by using mathematical verification instead of program debugging(2). The provision of additional tests and process changes improve the measurement of quality. The second priority of the cleanroom software engineering process is to provide valid statistical certification of software’s quality, based on system level’s representative-user testing results. The increase of the statistical certification is the reflection of growth of reliability during the testing process(2). In order to achieve the benefits of statistical quality control, in the cleanroom software engineering process, concurrent construction and incremental certification development cycles are required to accumulate the application system. In management perspective, “Cleanroom software engineering is a practical process to place software development under statistical quality control”(2). The cleanroom software engineering process is not only a theoretical process model, but also a practical guide that provides detailed procedures and instructions to produce realistic high quality applications.

**3. Development**

The theories of cleanroom were founded in the late 1970s and early 1980s by Harlan Mills, an mathematician and IBM Fellow. The scientific foundations of this engineering approach were related to mathematics, statistics and engineering fields in software. There are two fundamental facts that inspired Mills’s work, the first one is “programs are rules for mathematical functions”(3). It is the start of function theories in software development and results into technologies such as box structure specification and design, function-theoretic correctness verification and incremental development(3). The second one is potential program executions are infinite populations, to achieve quality certification, it requires statistical sampling. This fundamental insight is the start of statistical theories in software testing, as well as the reason for using statistical usage testing and quality certification technologies(3).Mill’s ideas were refined and demonstrated with the help and collaborations with colleagues and then being introduced in several publications. In 1987, the masthead cleanroom was proposed and these ideas were integrated. The word cleanroom is a term in the semiconductor industry that shows the reflection of an emphasis on problem prevention rather than defect removal. The research paper “Cleanroom Software Engineering” was published in the May 1987 issue of IEEE Software(3). The first cleanroom software project was developed in the mid 1980s. It is a commercial reengineering software project managed by Richard Linger of IBM and developed by the COBOL Structuring Facility project. The result shows it has remarkable quality and reliability in customer use which provided an initial validation of cleanroom engineering process(3).

**4. Goal**

According to an empirical evaluation of the cleanroom engineering process:“The Cleanroom software development approach is intended to produce highly reliable software by integrating formal methods for specification and design, non execution based program development, and statistically-based independent testing”(4). With the requirement of high reliability, a more practical and instructive definition of the goal of cleanroom engineering process is:”Cleanroom software engineering is designed to achieve two critical goals: a manageable development process and no failures in use”(3). The manageable development process is the goal for the development stage, it allows managers and teams to control the development process intellectually, which means complexity management, risk reduction, avoid rework, meet objectives schedule and budget control. The no failure in use is the goal for the use phase, it is against common attitude about software defects which is to correct failures and problems after exposure, but it allows softwares avoid failures and related consequences such as money, resource and time costs in tracking and fixing problems. Therefore, the final goal of the cleanroom engineering process is to create softwares with high reliability, by setting different minor goals in different phases throughout the whole stage.

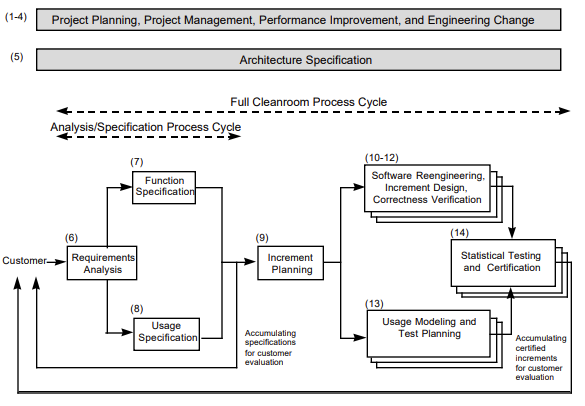
**5. Defining element**

Cleanroom software engineering is special because it uses three characterized principal technologies, separately or together: incremental development under statistical process control, function-based specification, design and verification and statistical testing and software certification(3).

***Incremental development under statistical process control***: The incremental development establishes a series of small development passes, cumulatively adding new works to previous work until final product delivery. It allows the team to focus on a portion of the project instead of working on the whole thing simultaneously, while customers can evaluate or confirm results during the development phase in order to minimize deviation.

***Function-based specification, design and verification***: The specification in cleanroom development method has three phases, the start is an external view called the black box, the next stage is a transformation process into a state machine view called the state box, the last one is called the clear box which is a fully developed procedure. The whole specification model is called box structures. The black box defines all required external behavior of a system or system component based on input history and its correct output, the state box defines all elements that need to be stored as state data by the black box, the clear box implements corresponding state box based on procedures and execute mapping rule, or maybe introduce new black boxes to describe major operations. The black box is used to record required behavior, the state box is used to define required state data, the clear is used to define required processing(3). For each box structure step, the team verifies the correctness by looking at the previous box structure and using the thesis in function theory.

***Statistical testing and software certification***: The testing method of the cleanroom engineering process is based on the usage model. It represents the status of all possible system uses and generates test cases. The validation statistics can be applied and predict expected operational performance because the test cases are random examples of system usus population. The usage model is a reusable practice that helps testing systems and various probability distributions, while it can be combined with various other forms of high-consequence testing functions such as safety-critical usage, hazardous usage, malicious usage, etc(3).



**Figure 1. Cleanroom Process Flow(5)**

Figure 1 is the cleanroom reference model, it defines all integrated processes and work products of cleanroom projects. There are 14 different processes for 4 major aspects: management, specification, development and certification(3).

***Project Planning, Project Management, Performance Improvement, and Engineering Change***: This process affects all other processes by creating fundamentals of the project. In the project planning process, the team creates and maintains development plans, which are used for managing and controlling incremental development and certification. The performance improvement process evaluates the performance constantly, identifies potential improvements and implements them. The engineering change process monitors and manages all change activities based on configurations and engineering disciplines.

***Architecture Specification***: This process describes the life cycle of the architecture, structures and strategies. It is the key in software design and it may affect low-level requirements design(3).

***Requirements Analysis***: The requirement analysis process’s goal is to create initial definitions of requirements based on customer proposals. It describes function specification process and usage specification process in detailed and precise terms.

***Increment Planning***: The increment planning reassigns specific functions to a set of smaller units, which are increments, then make plans about their development and certification under the overall schedule’s framework.

***Development and Certification***: This process represents continuous increments.

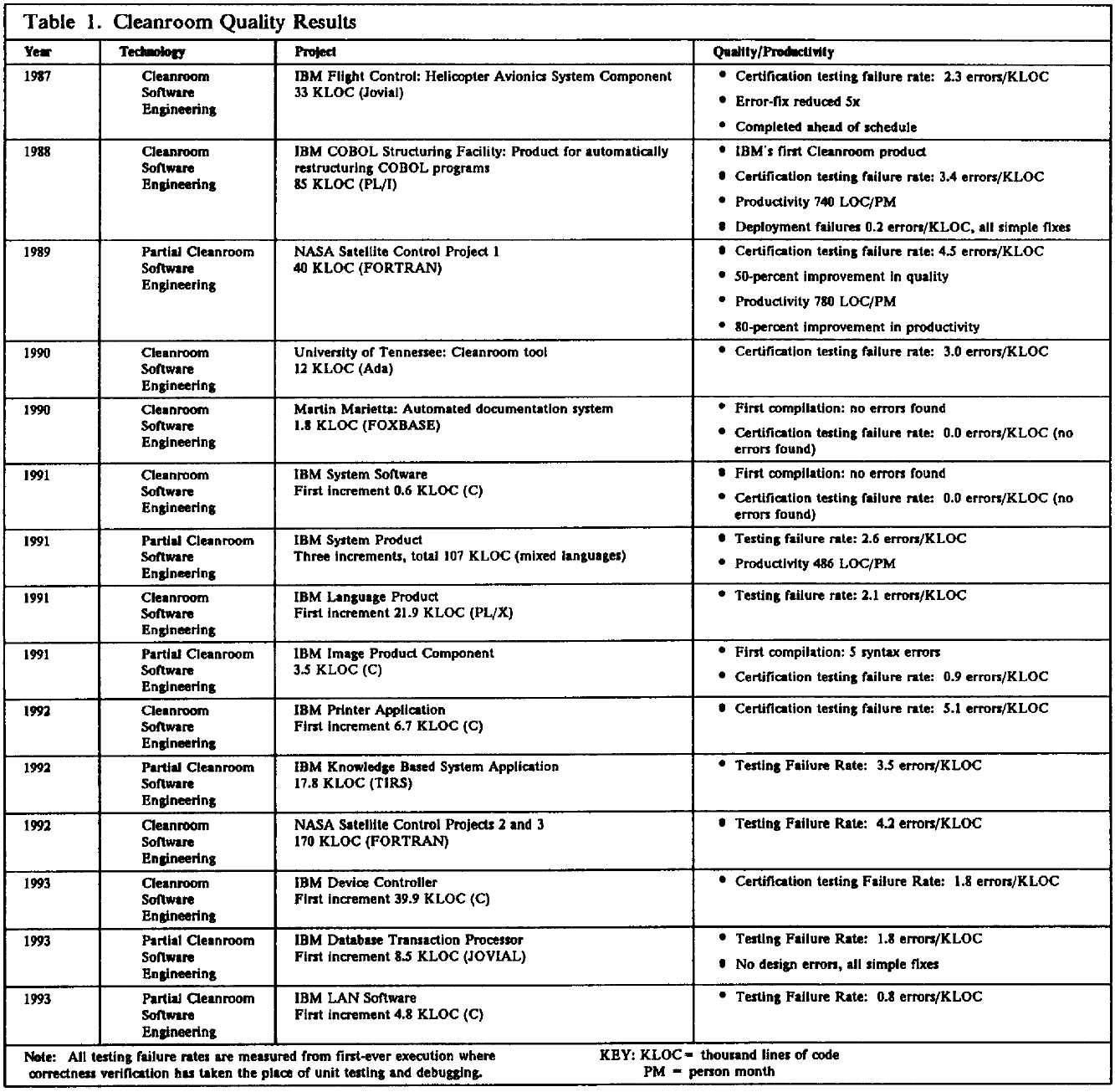
***Software Reengineering***: This process prepares existing software for use in an increment(3).

***Increment Design and Correctness Verification***: This process implements increment design and code and verify the correctness.

***Usage Modeling and Test Planning***: This process generates test cases from usage models and it is parallel with the incremental development activities.

***Statistical Testing and Certification***: This process evaluates an increment’s fitness for use(3). For a completed increment, customers give evaluation and feedback about requirements validation and it may result in revisitation of high-level specification processes.

**6. Practical result**



**Figure 2. Quality results from Cleanroom projects(6)**

Figure 2 is a table that summarizes quality results from Cleanroom projects. The “Certification testing failure rate” is the indicator for the performance which shows how many errors per thousand lines of code. These numbers represent all errors in all project development phases including testing, execution and completion, as well as the remaining error rates after the correctness verification work done by the teams. These projects are led by four different industrial companies and organizations: IBM, NASA, University of Tennesses and Martin Marietta, including fifteen industrial-strength software systems: IBM Flight Control, IBM COBOL Structuring Facility (COBOL/SF), NASA Satellite Control Project 1, Martin Marietta Automated Documentation System, IBM System Software, IBM System Product, IBM Language Product, IBM Image Product Component, IBM Printer Application, IBM Knowledge Based System Application, NASA Satellite Control Projects 2 and 3, IBM Device Controller, IBM Database Transaction Processor and IBM LAN Software(6). The performance results are excellent, according to the Cleanroom quality results research:”The projects in...produced over a half a million lines of Cleanroom code with a range of 0 to 5.1 errors per KLOC for an average of 3.3 errors per KLOC found in all testing, a remarkable quality achievement indeed”(6). Therefore, the cleanroom software engineering was used in various industrial-strength software systems and the excellent results confirms it is an efficient and reliable development process to produce high quality software applications.

**7. Assessment**

**Pros**

*Efficiency:* According to an empirical evaluation of cleanroom software development:”Most of the developers were able to apply the techniques of Cleanroom effectively (six of the ten Cleanroom teams delivered at least 91% of the required system functions). The Cleanroom teams' products met system requirements more completely and had a higher percentage of successful operationally generated test cases. The source code developed using Clean room had more comments and less dense control-flow complexity...All ten Cleanroom teams made all of their scheduled intermediate product deliveries...Eighty-one percent of the Cleanroom developers said that they would use the approach again”(4). Therefore, the cleanroom engineering process can provide effective, complete, more likely to succeed and on time software development procedure which lets developers willing to use this approach again.

*Predictability*: The biggest and most important difference between the cleanroom engineering process and traditional or other development methods is the attitude toward defects. The prevention of defects instead of tracking and solving afterwards allows application creators save efforts after the deployment and use. This mechanism also makes the product’s performance more predictable.

*Feasibility*: The cleanroom engineering process is not not only a theoretical or conceptual process model that needs more elaboration, but a well-developed and validated development method. It has enough technical details, procedures and instructions that allow individuals or organizations to produce actual high quality applications, as well as many research results on this topic. Furthermore, the feasibility of this process is verified by many practical cases including many industrial-strength software systems, including technology companies such as IBM and government organizations such as NASA.

**Cons**

*Cost*: The number phases and processes in the cleanroom engineering model is more than traditional or other development methods. Although the defect prevention trait can reduce or eliminate the cost in maintenance, the overall development cost may be the same or even higher than traditional or other development processes.

*Skills*: There are many technologies used in the cleanroom engineering method, especially the involvement of human mathematical verification, which makes the higher requirements of the team's skill sets. If the development team lacks certain skills required, the cleanroom engineering process cannot proceed. Training or searching talents will be another consideration, which may take extra time, effort and other resources that increase the overall cost.

**8. References**

1. Sheetal, A. & Kongara, Ravindranath. (2018). *Software metric evaluation on cloud based applications.* International Journal of Engineering and Technology(UAE). 7. 13-18. 10.14419/ijet.v7i1.5.9071.

2. Mills, H.D. & Dyer, M. & Linger, Rick. (1987). *Cleanroom Software Engineering*. Software, IEEE. 4. 19-25. 10.1109/MS.1987.231413.

3. Prowell, Stacy & Trammell, Carmen & Linger, Rick & Poore, Jesse. (2021). *Cleanroom software engineering: technology and process.*

4. Selby, Richard & Basili, Victor & Baker, F.. (1987). *Cleanroom Software Development: An Empirical Evaluation.* Software Engineering, IEEE Transactions on. SE-13. 1027 - 1037. 10.1109/TSE.1987.233525.

5. Linger, Rick & Trammell, Carmen. (1997). *Cleanroom Software Engineering - Reference Model Version 1.0.*

6. Linger, Rick. (1993). *Cleanroom software engineering for zero-defect software*. 2-13. 10.1109/ICSE.1993.346060.