Cleanroom software engineering

The **cleanroom software engineering** process is a software development process intended to produce software with a certifiable level of reliability. The cleanroom process was originally developed by Harlan Mills and several of his colleagues including Alan Hevner at IBM.[1] The focus of the cleanroom process is on defect prevention, rather than defect removal. The name "cleanroom" was chosen to invoke the cleanrooms used in the electronics industry to prevent the introduction of defects during the fabrication of semiconductors. The cleanroom process first saw use in the mid to late 1980s. Demonstration projects within the military began in the early 1990s.[2] Recent work on the cleanroom process has examined fusing cleanroom with the automated verification capabilities provided by specifications expressed in CSP.[3]

Contents

Central principles

The basic principles of the cleanroom process are

Software development based on formal methods

Software tool support based on some mathematical formalism includes model checking, process algebras, and Petri nets. The Box Structure Method might be one such means of specifying and designing a software product.[4] Verification that the design correctly implements the specification is performed through team review, often with software tool support.

Incremental implementation under statistical quality control

Cleanroom development uses an iterative approach, in which the product is developed in increments that gradually increase the implemented functionality. The quality of each increment is measured against pre-established standards to verify that the development process is proceeding acceptably. A failure to meet quality standards results in the cessation of testing for the current increment, and a return to the design phase.

Statistically sound testing

Software testing in the cleanroom process is carried out as a statistical experiment. Based on the formal specification, a representative subset of software input/output trajectories is selected and tested. This sample is then statistically analyzed to produce an estimate of the reliability of the software, and a level of confidence in that estimate.

References

1. Mills, H.; M. Dyer; R. Linger (September 1987). "Cleanroom Software Engineering". *IEEE Software*. **4** (5): 19–25. doi:10.1109/MS.1987.231413 (https://doi.org/10.1109%2FMS.1987.231413).
2. Foreman, John (2005). "Cleanroom Software Engineering Reference" (http://www.sei.cmu.edu/library/abstracts/reports/96tr022.cfm). *Software Technology Roadmap*. Software Engineering Institute (SEI). Retrieved 2006-04-27.
3. Guy H. Broadfoot and P. J. Hopcroft (2005). "Introducing formal methods into industry using Cleanroom and CSP" (http://www.omimo.be/magazine/emagazine/fulltext/2005Q1_1.pdf) (PDF). Dedicated Systems e-Magazine. Retrieved 2011-08-02.
4. Linger, R. (April 1994). "Cleanroom Process Model". *IEEE Software*. **11** (2): 50–58. doi:10.1109/52.268956 (https://doi.org/10.1109%2F52.268956).

Further reading

- Stavely, Allan (1999). *Toward Zero-Defect Programming*. Addison-Wesley.
- Stacy J. Prowell and Carmen J. Trammell and Richard C. Linger and Jesse H. Poore (1999). *Cleanroom Software Engineering: Technology and Process*. Addison-Wesley.
- Jesse H. Poore and Carmen J. Trammell (1996). *Cleanroom Software Engineering: A Reader*. NCC Blackwell.

External links

- An introduction (http://www.uta.edu/cse/levine/fall99/cse5324/cr/clean/page1.html)
