# Classification of Software Requirement Errors: A Critical Review

Pawan Kumar Chaurasia
Department of Information Technology
Babasaheb Bhimrao Ambedkar University,
Lucknow-226025, India

R.A. Khan
Department of Information Technology
Babasaheb Bhimrao Ambedkar University,
Lucknow-226025, India

## ABSTRACT
From last three decades various tools, techniques and methods are developed by researchers. The objective of research is to optimize the error and improve the quality of the software. During development of software, various errors are introduced by the developer at various phases of software development life cycle (SDLC). It is difficult to identify all the errors of the software by the developer of different phases. Various methodologies are proposed and implemented by the researcher to identify the errors. The objective of this paper is to review and develop taxonomy of requirement errors, prepare a list of requirement errors for the analysis. Conclusions are listed on software requirement errors at last. The list of requirement errors may support the researchers to improve their work in a systematic way and classify all requirement errors to increase the software quality.

## Keywords
Software Quality, Systematic Review, Error, Fault, Failure, Reliability.

## 1. INTRODUCTION
There is a lot of change in the size, complexity and software code of a program. Software development is beginning with various components and stages. Various errors and faults are encountered during the development. Main objectives of the developer are to optimize the error of the software and deliver the quality software to the customer. The major problem with the developer is to detect the hidden errors. It is time consuming and 50% cost of the whole project invested in it. These software errors are introduced from the requirement level. If all the errors are identified and removed at the initial level, then the quality of the software and reliability will be improved. Many organizations are introduced to accumulate and maintain the software error data. From last two-three decades; software error tools, techniques and methodologies are tested by the developer or software engineer. The purpose of these tools and techniques is to improve software reliability and reduce its cost by optimize the no of errors.

The purpose of software process is used to transform the information from one form to another. This process is executed by the human and probably errors are also occurred at the time of transformation. To improve software quality, tools and mechanism are required to prevent from these errors and identify when they occur. A good software quality is to identify the errors, fix the problem at early stage, minimize the expenses and rework. To improve the software quality; error, fault and failure causes are clearly defined to classify.

**ERROR**
It is defined as the difference between the actual output and the calculated output. Errors are defects in the human thought and occur while trying to understand given information, solve problems, or to understand the methodology and tools.

**FAULT**
Fault also referred as a bug. Fault is caused due to internal error or failure of the program. One error may cause many faults and many errors may cause identical faults.

**FAILURE**
It is defined in terms of incorrect results with respect to specification of user requirements or the system behavior is changed on the boundary of software systems. A particular may be caused by several faults and some faults may never cause a failure.

**According to Thomas Muller**

*"A human being can make an error (mistake), which produces a defect (fault, bug) in the code, in software or a system, or in a document. If a defect in code is executed, the system will fail to do what it should do (or do something it shouldn't), causing a failure."*

Errors have multiple definitions. IEEE 610 define errors in four different ways, it is an incorrect program, condition to a mistake by the human behavior. It is a human error rather than program error. Quality, always focus on the identification of error and removed faults at the early stage of the life cycle. This paper prepares taxonomy for classification of faults which are used in improvement of quality approaches. In spite of these, quality is still a problem for developer to miss the source of error.

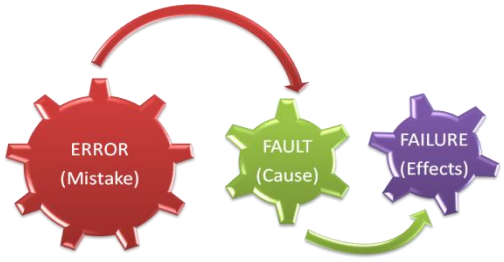


**Figure 1: Failure Life Cycle**

They have not any resources to learn from error; do not have anability to learn from mistakes and do not have any tool for verification of error. Therefore it is required to more focus on the faults, research required to strengthen to identify the source of error and cause of the error.

In this paper a systematic literature review is prepared for identification of error by other quality researcher. A systematic literature is a continuous process of improvement of knowledge on particular subject. The advantage of continuous knowledge is to improve and defended to implement more ad hoc approach. It improves the confidence of the developer to gain more relevant information as much as possible. It is also used in medicine, aircraft, automobile etc.

Paper is organized in various phases and explained in a following manner. Section I, introduce about the errors and discuss its background. Section II, prepare a review literature of various research paper proposed or implemented by the researcher concept on errors. Section III, describes various existing quality improvements approaches by researcher. Section IV describes about systematic review process and section V describes about the findings from the reviews.

## 2. LITERATURE

In this section, quality improvement approaches are described which are focused on various types of faults. How these faults are occurred and the source of error is identified.

### *Quality improvement approaches*

The error detection can be affected by various factors such as testing environment, testing strategy and resources. Error detection rate may not be change, when the testing environment, strategy and resources are changed [1]. Quality improvement classify faults from different phases into a review to support risk, cost and cycle time reduction. By using measurement framework, Software Engineering Institute (SEI) increased the quality by understanding the process and product. Software Engineering Institute (SEI) and Software Engineering Laboratory (SEL) approaches for fault classification to prepare a list of faults that facilitate to understand the faults and improves quality of the software. However faults are exist with these approaches. Without identifying the source of the faults, it is impossible to remove all the faults, or some faults are neglected.

Root cause analysis approach is used to classify the errors on orthogonal defect classification (ODC), due to complex methods; it has not found popularity [2]. It is also used to understand the relationship between cause and effect of errors. From the experiment, it is found that the Perspective-Based Reading (PBR) is effective in some context for defect detection at detecting faults in requirement documents. Requirements faults are difficult to define classify and quantify [3].

Software inspection techniques are used to classify software classification on the base of check list, fault based and perspective. Using these fault techniques inspectors cannot identify all the present faults. For undetected faults, various techniques are introduced like re-inspection or defect estimation techniques. Error information is only the prediction to identify the effectiveness of the errors [4].

Some of the errors are occurred in a group which are defined in error abstraction as described in figure 2. When one error is discovered, then locate the other error related to that faults. Disadvantage of this approach is that this approach is depends on the seriousness of inspectors for identifying the errors which comes from another error called abstraction error [3]

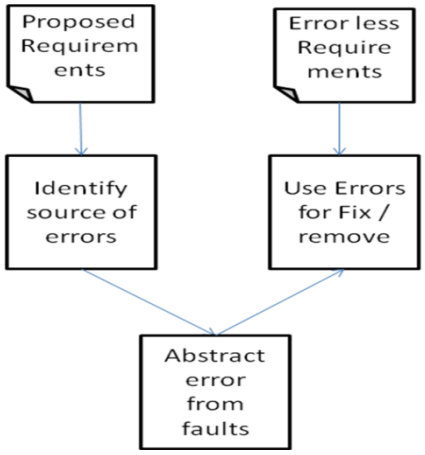


**Fig 2: Error Abstraction**

However, software development processed by human beings. Human nature and activity may also cause the requirement errors. Two projects are evaluated by two human beings and found that both the observation have different results. So, psychological effects also cause the effect on requirement errors is defined in [16, 17, 18].

## 3. METHODOLOGY

The objective of this review paper is to identify and classify the requirement errors. Various types of questions are reviewed to achieve the objective.

What are the various methodologies for identifying the requirement errors?

What are the attributes to classify the requirement errors?

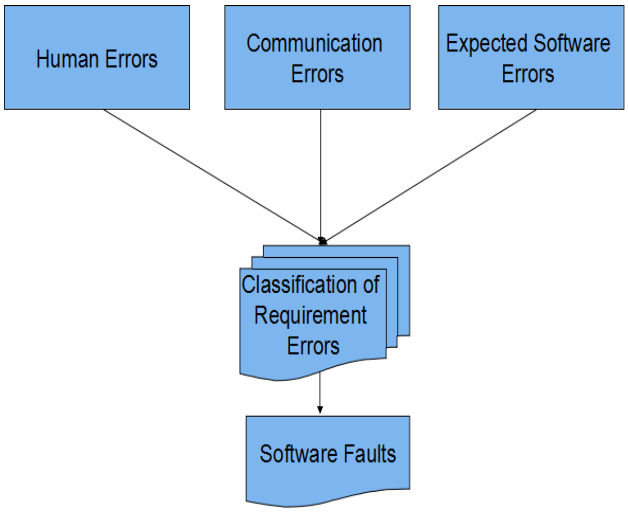


**Fig 3: Review for requirement errors**

Figure 3 shows that literature collected from software errors about faults and their efforts are combined with human errors and communication errors literature to develop the classification of requirement errors. All the collected problems are break into three small segments and sub-segments.

- How to improve quality improvements approaches which focus on errors.

These approaches helped in identifying the defects and information about errors.

Identify the explicit requirement errors and errors generated in

different life cycle phases.

Analysis of faults helped to understand the fundamental errors.

Identify the human errors from human information and psychological effects to identify the various types of faults.

It is helpful for analyzing the planning and problem solving skills of human at the requirement level.

# 4. POINT'S TO PONDERS

**Problem 1:** Is there any manifest that error information can improve the software quality?

**Problem 2 :** Is there any methodology or process that error information can improve the software quality?

**Problem 3:** What type of requirement errors have been identified from the previous researcher in requirement level?

**Problem 4:** What are the other errors occur in other phases of software life cycle which can affect at the requirement phase?

**Problem 5:** What are the actual faults identified at requirement phase?

**Problem 6:** What are the human errors information available and its classification?

**Problem 7:** Which of the following human errors behaves as a communication errors in software requirement errors?

Author identified above problems, based on requirement error, discuss all its aspects in a systematic approach:

**Discussion 1**

Software quality improved, by identifying the source of faults. Various researchers used this approach in different ways and improved the result on various parameters. A review of various methods indicate that the source of faults process improve the result and help in prevention of errors from previous experience and detection of defects during review as described in fig 4.

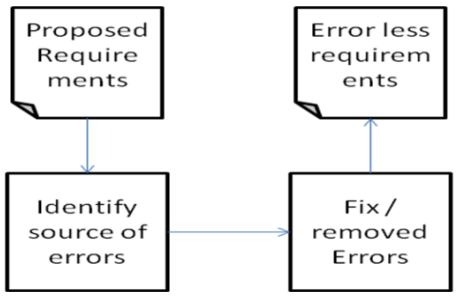


**Fig 4: Error detection/removal process**

Disadvantage of these approaches is that there is no methodology or process to assist developers in finding and fixing of errors. Laundible et. al. discussed in his research that the error information can improve the quality of a requirement document. There is no framework to identify the source of error. It is only used as a tested data but some of them are hidden and missed during the requirement testing errors.

## Discussion 2

From the various research and reviewed, few of them used software error information as to improve the software quality for software development. The usage of types of errors, help in identifying the source of error which are used by various researchers are as follows:

Analysis of defects approach is a unit based quality improvement procedure to prepare a sample of error to give help in prevention from that error from future faults. By using this approach Computer Science Corporation (CSC) optimize 50 % of the defect error [5]. Most of the software faults are found in one of the categories which is classified as follows:

Methods (Incomplete, Ambiguous, Wrong and Enforced)

Tools and Environment (Unreliable or Defective)

People (Lack of training or understanding)

Input and Requirements (Incomplete, Ambiguous or defective)

The goal of this approach, improve the software development and process. Developers try to prepare a prototype for defect data to find the particular module of defect error. Hawlett-Packyard prepared a guideline to interface design to optimize the interface error. Using these guidelines various researchers used and found 50% of the interface error decreased [6].

Defect cause of analysis help in expert knowledge and find the depth of the cause of errors. The cause of error found during the development of the software, help in identifying the errors during development and after delivery of the software. From the review it is found that most of the defects were found from the lack of communication [7].

Defect prevention analysis approach determine the source of a fault and suggest about the common errors occurred, reduce in defect rates, optimize the test time effort and customer satisfaction is increased. Mays et. al. worked on the analysis approach [8] and classified defect causes into various parts:

Supervision problem (like developer overlooked something or something avoid).

Education (developer cannot understand some features).

Communication (lack of communication).

Transcription (developer understand but miss at the time of development).

Software bug analysis approach help in identifying the source of bugs. To assess this approach, 28 sample of bugs were found during the debug and test phases of software. These bugs are occurred from group leaders, designers and external designers. Total 23 causes were found, half of them related to designers carelessness [9].

## Discussion 3

Reviewer tried to cover all the aspects of requirement errors, but a number of requirement errors are uncovered. Various errors and cause methods are identified and described: Some of the requirement errors [10] occurred like communication between developers, incomplete training, consider all the aspects, confusion in understanding the development aspects.

Second source of errors classify the requirement errors problems from developers which classify difficulties for influencing the requirement errors faced during the development process [11, 12]. The outcomes from these reviews are as follows; lack of user participation, poor skills and knowledge, complexity of problem, undefined requirement process.

Third source of errors occurred from root causes inconvenience projects like; wrong business plan, use of tremendous technology, user are not available at all levels, lack of experience, proper management, lack of guidance and work environment.

From requirement error it is also found that influence of

requirement traceability, inconsistency, domain knowledge, requirement management errors, time management and commitment for execution of projects during development [13, 14, 15]. These studies of errors represent poor planning, informal change request, lack of proper analysis and misunderstanding between team members.

**Discussion 4**

Various paper are reviewed by the researcher but some of the portions are uncovered during the design and coding phases. These errors are also occur during the requirement phase like missing information, insufficient specification, inappropriate action, wrong plan, incomplete details in specification, lack of attention, technical problems and organizational problems etc [ 16, 17, 18].

**Discussion 5**

In addition to these errors represent in the list, this review also identified some fundamental errors [19, 20, 21] like documents are prepared by various team members, a functionality will be missing because of misunderstanding. These problems occur, because of lack of communication between team members, two requirements are incomplete; lack of information, developer could not incorporate the particular function in multiple places.

**Discussion 6**

The various types of human errors which are occurred and it is classified in two parts human knowledge and human psychology.

Classification of mistakes (wrong plan, forgot execution of write statement, incorrect execution) [22, 23].

Knowledge based human errors (inattention, rule based errors because of in intentions, unfamiliar situation)

Individual discrete errors (commission errors, sequence errors, timing errors and omission errors)

Control Error (substitution, adjustment, forgetting the control position and reach control in time)

- Influence factors for occurrence of errors (Environmental conditions, organizational factors, task factors and user qualities)

**Discussion 7**

Errors which are included in the initial list of errors that served as input value. Errors which are translated into requirement errors are listed below:

1. Wrong interpretation due to complex nature of the problems.
2. Misunderstanding the execution of the problems.
3. Errors in applying the requirement engineering process.
4. Poor methods of achieving goals and objectives.
5. Incorrect translating requirements written in natural languages.

Mistakes caused by unfortunate conditions, loss of situation knowledge, lack of motivation.

## 5. REQUIREMENT ERROR TAXONOMY

| SN | TYPES OF ERRORS | DESCRIPTION | REF. |
|---|---|---|---|
| 1 | Communication errors | Insufficient project communication | 6 |
| | | Requirement editing is not communicated | 7 |
| | | Lack of communication between developers and users | 24 |
| | | Poor communication between developers team | 11 |
| | | Poor communication between development process | 25 |
| | | Lack of communication information not reach between peoples | 26 |
| 2 | Participation errors | Involvement of users at requirement level | 5 |
| | | Participate only selected users | 27 |
| | | Do not involve all the neutrals | 7 |
| 3 | Domain knowledge errors | Lack of domain knowledge | 10, 28 |
| | | Complexity of problem | 11, 12 |
| | | Lack of appropriate proper knowledge and information | 29 |
| | | Lack of proper training | 6 |
| | | Misunderstanding due to complexity | 28 |
| 4 | Specific application errors | Knowledge of hardware and software specification | 31 |
| | | Knowledge of input, output and process mappings | 32 |
| | | Errors in expected output | 15 |
| | | Requirements are interpreted or predict while solving conflict problems | 9 |
| | | Knowledge of software interface module | 30 |
| 5 | Process execution errors | Errors in sequence of execution or requirement process | 33, 34 |
| | | Storage problem, sequence order of stages and missing stages | 34, 28 |
| 6 | Human knowledge errors | Lack of situation awareness problem | 14, 26 |
| | | Environmental conditions | 25 |
| 7 | Inadequate method | Incomplete knowledge for achieving goals | 28 |
| | | Errors in achieving goals | 28 |
| | | Selection of wrong method | 36 |

| | | | |
|---|---|---|---|
| | | Transcription error | 8 |
| 8 | Management errors | Poor management of people & resources | 29 |
| | | Lack of leadership | 13 |
| 9 | Specification errors | Missing conditions | 10 |
| | | Errors while documenting requirements | 36 |
| 10 | Organizational requirement errors | Poor organization of requirement | 6 |
| | | Errors in organizing requirements | 22 |
| 11 | Requirement analysis errors | Selection of incorrect model | 35 |
| | | Misuse of error solution process | 24 |
| | | Unsolved issues and problems | 10 |
| | | Errors while analyzing requirement use cases | 24, 37 38 |
| 12 | Requirement simulation errors | Inadequate requirement gathering process | 10 |
| | | Lack of information for source of resources | 29 |

## 6. FINDINGS

The objective of this review paper is to find and classify the errors in requirement phase of software development life cycle. A critical review was carried out for covering various research domains. From these information, requirement error review was formulated. The main findings of this review are as follows:

- Software quality improvement approaches that cover error information, limitations and causes of requirement errors.
- Software methodologies have been explained and described the method to find software requirement errors.
- A description of requirement errors is described in review and revision.
- At last all the uncovered requirement errors during review and revision are classified in requirement errors.

## 7. CONCLUSIONS

Software development process starts from the transformation of information from one stage to another. Quality research focused on the detection and removal of the errors. To make assure to develop high quality software, good mechanism are selected and applied on the software. Based on the above review and the information collected from various reviews and research papers the critical requirement error is predicted. The main objective of developer is to optimize the errors and apply relevant test cases before delivering the software to the customer. Quality will be improved these identify errors, faults and failures of the software early. List of requirement errors help the seekers who are searching methods and tools to improve quality and usage of tools in more effective manner. List of error points reflect the review of the paper in a vital manner. More systematic reviews are required to identify errors in the other software life cycle phases.

## 8. REFERENCES

[1] Nikora, A. P.; Lyu, M. R. Software reliability measurement experience. Handbook of Software Reliability Engineering, Lyu, M. R. Ed.;McGraw-Hill: New York, 1996; 255–301.

[2] P. Fusaro, F. Lanubile, G. Visaggio, A replicated experiment to assess requirements inspection techniques, Journal of Empirical Software Engineering 2(1) (1997) 39–57.

[3] F. Lanubile, F. Shull, V.R. Basili, Experimenting with error abstraction in requirements documents, in: Proceedings of Fifth International Software Metrics Symposium, METRICS'98, IEEE Computer Society, Bethesda, MD, 1998, pp. 114–121.

[4] S. Basu, N. Ebrahimi, Estimating the number of undetected errors: Bayesian model selection, in: Proceedings of the Ninth International Symposium on Software Reliability Engineering, IEEE Computer Society, Paderborn, Germany,1998, pp. 22–31.

[5] D.N. Card, Learning from our mistakes with defect causal analysis, IEEE Software 15 (1) (1998) 56–63.

[6] R.B. Grady, Software failure analysis for high-return process improvement, Hewlett–Packard Journal 47 (4) (1996) 15–24.

[7] J. Jacobs, J.V. Moll, P. Krause, R. Kusters, J. Trienekens, A. Brombacher, Exploring defect causes in products developed by virtual teams, Journal of Information and Software Technology 47 (6) (2005) 399–410.

[8] R.G. Mays, C.L. Jones, G.J. Holloway, D.P. Studinski, Experiences with defect prevention, IBM Systems Journal 29 (1) (1990) 4–32.

[9] T. Nakashima, M. Oyama, H. Hisada, N. Ishii, Analysis of software bug causes and its prevention, Journal of Information and Software Technology 41 (15) (1999) 1059–1068.

[10] Bhandari, M. Halliday, E. Tarver, D. Brown, J. Chaar, R. Chillarege, A case study of software process improvement during development, IEEE Transactions on Software Engineering 19 (12) (1993) 1157–1170.

[11] S. Beecham, T. Hall, C. Britton, M. Cottee, A. Rainer, Using an expert panel to validate a requirements process improvement model, The Journal of Systems and Software 76 (3) (2005) 251–275.

[12] G.J. Browne, V. Ramesh, Improving information requirements determination: a cognitive perspective, Journal of Information and Management 39 (8) (2002) 625–645.

[13] C. Debou, A.K. Combelles, Linking software process improvement to business strategies: experiences from industry, Journal of Software Process: Improvement and Practice 5 (1) (2000) 55–64.

[14] M.R. Endsley, Situation awareness and human error: designing to support human performance, in: Proceedings of the High Consequence Systems Surety Conference, SA Technologies, Albuquerque, NM, 1999, pp. 2–9.

[15] D.A. Norman, Design rules based on analyses of human error, Communications of the ACM 26 (4) (1983) 254–258.

[16] D.A. Norman, Steps towards a cognitive engineering: design rules based on analyses of human error, Communications of the ACM 26 (4) (1981) 254–258.

[17] C. Trevor, S. Jim, C. Judith, K. Brain, Human Error in Software generation Process, University of Technology, Loughborough, England, 1994.

[18] Endres, An analysis of errors and their causes in system programs, IEEE Transactions on Software Engineering 1 (2) (1975) 140–149.

[19] T.E. Bell, T.A. Thayer, Software requirements: are they really a problem?, in: Proceedings of Second International Conference on Software Engineering, IEEE Computer Society Press, Los Alamitos, CA, 1976, pp 61–68.

[20] T. Berling, T. Thelin, A case study of reading techniques in a software company, in: Proceedings of the 2004 International Symposium on Empirical Software Engineering (ISESE'04), IEEE Computer Society, 2004, pp. 229–238.

[21] B. Freimut, C. Denger, M. Ketterer, An industrial case study of implementing and validating defect classification for process improvement and quality management, in: Proceedings of the 11th IEEE International Software Metrics Symposium, IEEE Press, 2005.

[22] P.C. Cacciabue, A methodology of human factors analysis for systems engineering: theory and applications, IEEE Transactions on System, Man and Cybernetics – Part A: Systems and Humans 27 (3) (1997) 325–329.

[23] Endres, An analysis of errors and their causes in system programs, IEEE Transactions on Software Engineering 1 (2) (1975) 140–149.

[24] S.H. Kan, V.R. Basili, L.N. Shapiro, Software quality: an overview from the perspective of total quality management, IBM Systems Journal 33 (1) (1994) 4–19.

[25] D. Batra, Cognitive complexity in Data modelling: causes and recommendations, Requirements Engineering Journal 12 (4) (2007) 231–244.

[26] K. Sasao, J. Reason, Team errors: definition and taxonomy, Journal of Reliability Engineering and System Safety 65 (1) (1999) 1–9.

[27] D.A. Gaitros, Common errors in large software development projects, The Journal of Defense Software Engineering 12 (6) (2004) 21–25.

[28] J. Galliers, S. Minocha, A. Sutcliffe, A causal model of human error for safety critical user interface design, ACM Transactions on Computer–Human Interaction 5 (3) (1998) 756–769.

[29] J. Coughlan, D.R. Macredie, Effective communication in requirements elicitation: a comparison of methodologies, Requirements Engineering Journal 7 (2) (2002) 47–60.

[30] P. K Chaurasia, Software Reliability Chain Model, International Journal of Software and Web Services (IJSWS), Vol 1, Issue 8, pp 46-50.

[31] R.R. Lutz, Analyzing software requirements errors in safety-critical, embedded systems, in: Proceedings of the IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, San Diego, CA, USA, 1993, pp. 126–133.

[32] S. Sakthivel, A survey of requirement verification techniques, Journal of Information Technology 6 (2) (1991) 68–79.

[33] B. Cheng, R. Jeffrey, Comparing inspection strategies for software requirement inspections, in: Proceedings of the 1996 Australian Software Engineering Conference, IEEE Computer Society, Melbourne, Australia, 1996, pp. 203–211.

[34] P.M. Fitts, R.E. Jones, Analysis of factors contributing to 460 'pilot error' experiences in operating aircrafts control, in: Proceedings of Selected Papers on Human Factors in the Design and Use of Control Systems, Dover Publications Inc., New York, 1961, pp. 332–358.

[35] A.J. Ko, B.A. Myers, Development and evaluation of a model of programming errors, in: Proceedings of IEEE Symposium on Human Centric Computing Languages and Environments, IEEE Computer Society, 2003, pp. 7–14.

[36] Swain, H. Guttman, Handbook of Human Reliability Analysis with Emphasis on Nuclear Power Plant Applications, Nuclear Regulatory Commission, Washington, DC, 1983.

[37] P. K Chaurasia, How Accountability Improves Software Reliability?, International Journal of Computer Science and Technology (IJCSET), Vol 5, Issue 9, pp 868-871.

[38] S.T. Shorrock, B. Kirwan, Development and application of a human error identification tool for air traffic control, Journal of Applied Ergonomics 33 (4) (2002) 319–336.