Inserting Safety into Software – Safety Critical Systems – Management Perspective

P. V. Srinivas Acharyulu and P. Seetharamaiah

***Abstract*—Organizations are more interested in delivering the software product within a limited time frame with more profitability. Hierarchical positions in the organization, from top to bottom define constraints for non-disclosure of information in view of the safety of the entire organization, resulting in poor flow of information, causing incomplete understanding of the desired safety measures, at the lowermost level. Mere understanding the problem by the peer does not ensure understanding the degree of severity and importance of the software care, needs to be flown to the bottom level of software development personnel with magnificent visualization and illustration. The developing team would be under pressure for delivering the ultimate product under compelling reasons, causing negligence or ignorance to critical issues involved, thereby exposing the problems, during software implementation. Software development team has incomplete knowledge of safety and organization has incomplete knowledge of software behavior and its life cycle. This paper proposes integrated typical structure of information flow and reviews in a cyclical way for continuous evaluation and validation through specialized team for developing safety critical software. Software Safety greatly depend on adequate salient information flows, with detailed consequences of failures and their effects, down to developmental team, and their characteristic importance of assignment. Organization behavior and software safety though both are independent but are of intricate nature having effects on both sides, such as software behavior and organization safety on the other hand. Thus neither of these should be ignored. In this paper we critically analyze various organizational factors and software safety factors having interaction between them. An amalgamated design is suggested in between these independent domains. This model is applied to laboratory prototype of RTCS (Road Traffic Control System). The results have inferred that all safety critical operations were safe and risk free, adequate to handle contingent situations arising out of practices.**

***Index Terms*—Safety-Critical Systems, Software Safety, Software Quality, Road Traffic Control System (RTCS).**

I. Introduction

Management practices under different compelling situations, lacks is eliminating appropriate information flow flaw downstream causing errors while execution of software with desired safety. Similarly, various constraints and strict boundaries administered at lower cost, with more reliability and expectations. Exhaustive information is collected for analyzing the various factors necessary for making the fail-safe software, particularly, safety critical systems, arising out of managerial practices, practical situations, software design professionals, safety professionals, ecologists, and quality personnel in addition to public opinion, traffic management personnel and from web. This information is kept in mind while designing the safety critical software, Road Traffic Control System (RTCS). An approach of involving professionals with specific domains and applied to software, could result in improving the safety, and applied in an integrated way. Vigorous study is being done, to derive at plausible solution to form Integrated Software Safety Group (ISSG). The responsibility of the group is to eliminate various errors, missing specification complete knowledge of other domain. The group is having complete documented information, as discussed in this paper, with specialized in software safety, and can deliver a software product with desired safety.

Manuscript received August 31, 2011; revised September 14, 2011. This work is being done as part of partial fulfillment of doctoral degree course leading to award of Ph.D. "*Inserting Safety into Software – Safety Critical Systems – Management Perspective*".

P.V.Srinivas Acharyulu is with IT Department, as Assistant Engineer (IT), NTPC Limited (A Government of India Enterprise). He is at present pursuing part-time doctoral degree in Computer Science & Systems Engineering from GITAM University, Visakhapatnam, Andhra Pradesh, India. (email: pvschari@yahoo.com)

Dr. P.Seetharamaiah is with Professor, Dept. of Computer Science and Systems Engineering, Andhra University College of Engineering, Visakhapatnam- 530 003, Andhra Pradesh–India. (email :psrama@gmail. com).

II. Software Safety and Its Need

Software Safety is considered as most important and discussed in various software standards, specifying the needs for well being of the users, applications, equipment etc. to avoid software failures leading to hazards by involvement of computer systems in real life. Specifically in applications of safety-critical systems, the contributions or attributions of software failures made significant danger to human life, substantial economic loss and extensive damage to environment. A study and proper remedy and requirement of software quality and standards triggered the need for review of the various standards and models in safety-critical computing systems. A safety-critical computer system is such a system which has the potential to cause hazards or allow hazards to occur. A software is said to be safe if it is quite not possible or a seldom instance to produce an output that could cause a catastrophic incident to the system which it controls. Most of the systems that do not have adequate safety design aspects caused loss to the physical property, harm and loss of life[9]. Software Engineering of Safety-critical computer systems needs a clarified/classified understanding of exact role of software and its interactions with the system. According to W.R. Dunn [7], dependable, seemingly safe concepts and structures fail while in practice due to three primary reasons. Their originators or users.

- having an incomplete understanding of what makes a system "Safe"

- Fail to consider the larger systems into which the implemented concept is to be embedded or
- Ignore single points of failure that makes the safe concept unsafe when put into practice

There are many well known examples of safety-critical systems' application areas such as automotive, defense, air traffic, air craft controlling, transportation, communications, medical diagnostics, nuclear, thermal and atomic power, instrumentation etc. Since, the safety is dependant on the correct and perfect desired performance of the software, this paper primarily emphasizes on the software component of safety-critical computer system, while considering the organizational constraints and factors specific to particular system. Since scope of safety does not confine to software element only, but also to consider the safety of whole equipment, software, operators or users, and environment, organizational factors that contribute to software safety are analyzed. Most of the systems keep their reliability and confidence on software to achieve their ultimate goals. Thus safety-critical computer systems are real-time control systems and require most attention and care in their specification, planning, design, implementation, validation, evaluation and operational maintenance. A thorough understanding is very much required to eliminate errors in software which may lead to or allow hazardous condition that could potentially result in catastrophic accident pertaining to life, un-sustainable injury, and damage to equipment and/or environment. In this paper, it is considered that such type of safety-critical computer system for application to make fail-safe. Some of the examples of hazards induced by software failures are given for reference as under. The following are some of the concepts and terms relating to safety found in the literature on the web relevant to the safety-critical computer system.

III. Terminology

Failure: An event where a system or subsystem component does not exhibit the expected external behavior and environmental conditions under which it must be exhibited should be documented in the requirements specification [9]. **Error:** An incorrect internal system state [9]. **Mishap:** Mishap is an unplanned event or series of events resulting in death, injury, occupational illness, or damage to or loss of equipment or property or damage to the environment [2]. **Hazard:** A system state that might, under certain environmental conditions, lead to a mishap [1]. Hence hazard is a potentially dangerous situation. **Risk:** Risk is the combination of the possibility of an abnormal event or failure, and the consequence(s) of that event or failure to a system's components, operators, users or environment [2]. **Safe:** Safe is having acceptable risk of the occurrence of a hazard [2]. **Safety:** Safety is the freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property [3]. **Safety-critical:** Those software operations, that if not performed, performed out of sequence, or performed incorrectly could result in improper control functions (or lack of control functions required for proper system operation) that could directly or indirectly cause or allow hazardous condition to exist [4]. A real-time system is safety critical when its incorrect behavior can directly or indirectly lead to a state hazardous to human life [5]. Decisions which shape the software architecture for safety-critical, real-time systems are driven in part by three qualities namely availability, reliability and robustness [5] [6]. **Software Safety:** The application of the disciplines of system safety engineering techniques throughout the life cycle to ensure that the software takes positive measures to enhance system safety and that errors that could reduce system safety have been eliminated or controlled to an acceptable level of risk [5]. **System Safety:** Application of engineering and management principles, criteria and techniques to optimize safety and reduce risks within the constraints of operational effectiveness, time and cost throughout all phases of the system life cycle [5].

IV. Software Failures In The Past

Computers are introduced in safety-critical systems, as a result, software failure found to have contributions to accidents. The ariane-5 explosion [9], Therac-25 Accidents [11] are some of the most referred software related accidents. An unmanned Ariane-5 rocket was launched by the European Space Agency, had exploded in short time after its take-off from Kourou, French Guiana in 1996. The rocket was developed after a decade of exercise and cost of $7 Billion dollars, the board of which investigated and found that software error caused failure and the loss was valued at $500 million. Therac - 25, a computerized radiation therapy machine lead to six known accidents, involving excess massive doses resulting in deaths and serious injuries. This was the ever worst series of incidents of radiation accidents in the medical history [11].

Section V describes various organizational functions typically. Software developmental issues are dealt in section VI. Section VII addresses the objectives and aims in making organization run with safety. Application of the model for integrating organizational and software factors to Road Traffic Control System (RTCS) in a cyclical way is described in section VIII. Concluding comments are given in section IX.

V. The Organization And Its Functions

Organization and management are more attracted and interested in reducing the costs and time of delivery while increasing the safety, looking for more profits and adaptation of austerity measures. To the best of their set goals, various levels of individuals are deployed conferring different administrative and financial delegation of powers. Sometimes their position becomes constraint for non-disclosure of actual information, data in view of the safety of the whole industry. Here it forms constriction of non-flow of proper information downstream, leading to the flaws and broken communication channels affecting the actual degree of expected performance, undesired behavior thus reflects. Different levels are liable to hold certain responsibilities which they need to be executed. Policies of adapting austerity measures, to lower the cost factor, and

target time frame enforce in fulfilling these objectives rather less concentration on software related issues. This results in inadequate time frame, budget to educate and train people working in the development and in the field. Hence no proper attention or care, understanding can be taken up by the stake holder available in the location or development.

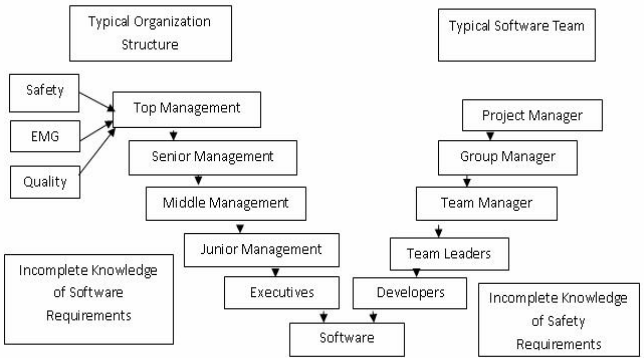

Fig. 1. Organization and Software Teams in V Model towards Software Safety

Any industrial organization mainly has three independent mandatory departments, mandatory, such as Safety, Environment Management Group (EMG), and Quality. These three are independent and functions as technical advisory to the management which takes actions upon their recommendations. These three groups now-a-days mixed and formed into another group named as Business Excellence, having no knowledge, or minimal awareness of special aspects of a particular system, such as software and its development. A separate wing with software do exist, who are unaware of safety, EMG and Quality principles and guidelines. Normally software domain people are utilized only for maintenance of existing databases, software etc, with the outcome of Process Outsourcing, who are aware of neither management nor maintenance part. Their purview is limited till delivery of the system product only. Thus problems arise while in practice.

A. Components of Organization

Figure 1 shows various levels in the organization and workflow. Organization comprises of humans deployed in various levels to carry out tasks of production. Each and every level is conferred with different delegations of administrative and financial powers. The Safety Division: This division primarily deals with the statutory standards pertaining to the equipment, safety measures, providing protective equipments to avoid or mitigate or sustain accidents, and conducts periodical audit for any of the failures in compliance with the standards and directives. They provide information to the top management for preparing disaster management plans, mitigation measures, suggesting corrective actions. The EMG Division :

These people deals with the hazardous environmental conditions and their management. They conduct periodical audits and recommends accordingly for improvement in the area and suggest suitable measures. The Quality Division: This group of individuals deals with the quality of performance of equipment, and reports for improvement of quality levels to the standards. This division mainly deals with the equipment, machinery and quality of the material. But quality of software is being untouched. Although safety requirements are suggested, there is lacking in understanding how safety be introduced into software. Management groups, such as top, senior, middle and junior management, while in consideration, towards downstream of safety requirements gets diluted stage by stage with lacking in communication of proper information. Top management is concerned with decision makings for productivity, expansion, discipline, manpower turnover, and various administrative controls. Senior management looks after planning implementation of top management's decisions. Middle management deals with the division of planning, providing infrastructure etc., junior management is concerned with execution of instructions from higher-ups. Executives develop various programs for realizing the plans over a period of time. It is very important to note at this point that none of management from organization side has complete knowledge of software safety development life cycle and its requirements, leading to arguments, because their prime importance is to achieve their set goals within the committed time frame by using computer systems. It is to note that no software safety and its standards are neither discussed nor addressed by the management in the organization. This forms least concern on software safety issues.

B. Factors That Influence the Typical Organization

Organizational behavior changes from time to time whenever changes occur in high manpower turnover. These factors are analyzed and influence on the working styles, are not compromising having a different direction of a common objective. Though the objective is not missed out, responsibility being diffused, may lead to uncertainty. Under the context for software safety, in safety-critical systems, insertion of safety into software is of less priority. Completion of the tasks and winning rewards are of prime importance, due to reward structures are for other performance rather than quality performance. This sort of organization behavior causes diversion in the quality software development & thus behavior, impact will be reciprocal. Moreover, administrative controls shall be exercised to bring the situation under control, an indication in unnecessary intervention, can be eradicated with judicious flow of information and transparency with clear, authenticated policy and practice. Feed back is no where having any consideration, and flow of information is downwards only.

C. Personnel Quality & Experience

This characteristic mainly of human resource (HR) related, but greatly contributes towards the quality of performance, the crux of the output, whether Managerial performance or Technical performance, in question, could not be measured. And who decide & determine? How to decide & determine? Thus four combinational abilities arise with personnel quality & experience. A best managerial performer can be technically weak. A low managerial performer can be technically best. On the other hand, best managerial & technical performers, and performers with both weak skill sets. This condition requires to be elucidated, and has serious impacts of governing the policies into the practices,

with respect to software safety. The personnel quality and experience plays a very important role, if subjected in the proper arrangement. Since this area has wider jurisdiction, perhaps enters into controversies.

D. Configuration Management

This characteristic, do refer to both managerial skill and technical skill. Configuring right men in the right place and position, is a crucial task but absolutely necessary for achieving the success. Particularly with safety critical systems, one must be strict technically, while with some susceptibility managerially, due to the condition, that managerially and technically weak men can be utilized for better documentation. Thus configuration management should be strong enough and greatly depend on abilities of the resource person, who may be well versed with the past and current practice, can stand as a barrier for change management in both the sets.

E. Stable & Validated Requirements

Management Stability? Or Technical Stability? Or Stable Management Requirements? or Stable Technical Requirements? Which of these to be considered? Management stability in the sense that it has close relevance between policy and practice, and is unpredictable often, depends on personnel quality & experience. Technical stability is that a technical aspect which had successfully run over years, without failure. This cannot be the situation with technological advances, technological changes inevitable. Are the management requirements, stable for longer periods? Absolutely not, be, as they keep on changing. Thus there should be clear, stable, validated requirements, policy and practice, defined management and technical requirements is, the need of the hour, should be guided by HR policy & configuration management, having stability control.

F. Independent Verification & Validation

Both verification and validation should have an independent charge, and should not have any sort of influences, such as administrative, financial and reporting constraints. Otherwise they suffer from appropriate documentation, there by lacking in safety issues. Verification by independent team with respect to safety issues, environmental issues and quality issues. And a strict adherence to the policy framework should be assured without any compromise. Validation of safety software product must pass through expertise review committee.

G. Formal Life Cycle

A management should have a very formal systemic and systematic life cycle for simplicity and should never attempt for inter mingling of issues. And it should be well defined to enforce responsibility for each and every individual as well for system. Making a complex model seems to be more attractive, voluminous but efficiency & effectiveness becomes reduced. Maintainability particularly, safety -critical systems becomes much difficult.

H. Traceability

Traces of an issue or event or incidence or procedure must be recordable without any ambiguity, should cater the requirement for measuring the performance in a qualitative way. Minute points or parameters cannot be ignored, should be available to represent diagrammatically.

I. Hazards Guide

Unswerving results of risks and hazards analyses, must be readily available, to guide every stage of development phase.

VI. Functions Of Software Development Team

As depicted in the figure 1 software development team too comprises of different levels to cater the requirement of developing a product relevant to the organization by using computer systems and controls. The decisions or commitment shall be taken by the project manager whose target primarily of acquiring the works contract. The contract shall have main constraints of budget and time frame, earnings and expenses on these developmental issues are more concentrated for the survival of entire group. Since safety guidelines are not available with, extending safety into software in real time systems are mostly untouched. Concentration is on the part of successful completion and run of machinery or equipment with least consideration of software safety failures, where a potential hazard can exist while implementation of the software. Once the contract is acquired, the project shall be handed over to the group manager for further actions. The entire software system will be divided into various parts and shall be assigned to teams. Now the part of the team manager to fragment further and handover to next lower groups for necessary developmental activities. In this software development team none of them are from safety side having complete knowledge of safety critical issues, where software can contribute to hazards. In addition, software development team are lacking in the safety requirements to the desired level, but go on developing the software without much safety consideration and check. Dividing the entire software system into different segments and assigning for development of modules to various individuals, forms a lacuna in information flow from top to bottom, with respect to follow the necessary recommended safety precautions. As the level goes down, the pressure increases gradually for earlier completion of the project, rather less concentrating on the safety issues. Moreover, training on detailed operation of the software function is not deliberated as part of development, and absence of trained personnel in the operational area can also lead to hazards. In addition wrong data or results suggest wrong decisions by human operators at the work place can contribute to hazard. Data validation and verification is not part of the software development issue, but is a safety critical issue.

VII. Objectives and Aims

In order to achieve understanding between these two independent domains towards software safety, an integrated software team is proposed which has both safety and software requirements knowledge can provide better safe design pattern for software in real time systems. The proposed integrated software development which shall have complete information from all sides through verified and

validated documentation, of safety requirements, environmental recommendations, quality standards, software requirements, configuration management, and process feedback. Having complete requirements thoroughly documented, the formed group can develop software with integration of safety. Thereafter each execution of the developed software shall provide updated data by verification from safety, environment and quality professionals duly documented. Similarly validation shall also be documented for next run process from software development team. This is a continuous process, every time with updates expected to run successfully.

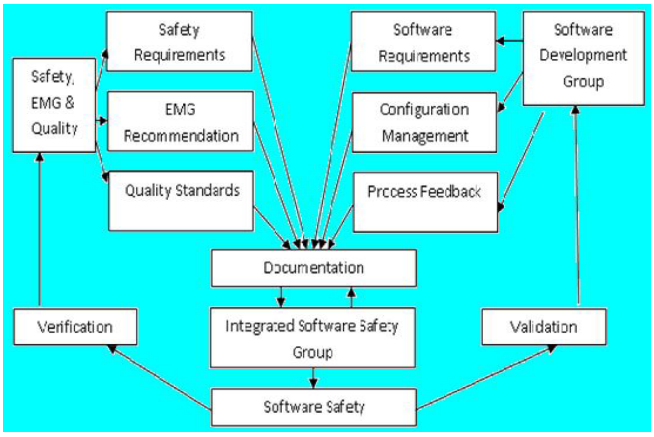

Fig. 2. Integrated Software Safety Life Cycle

A. Inserting Safety & Responsibility into Software

Responsibility and public safety must be the prime objective of team members of software development at all levels irrespective of their own interests. And this responsibility and public safety must have accountability, needs to be enforced, with the help of standards and guidelines. HR management should improve morale, should not adopt machinery concept or commodity concepts of management priniciples. Due to the fact that construction takes more time than destruction, concentration should be made to retain talent, skill and morale. It should be the responsibillity of the management, to make all team members to feel that their contribution has importance. Specific developmental training needs should be accorded with respect to the software and relevant to safety issues.

B. Technical Skill and Quality

Technical skill and quality is not ones own property. It should be identified thoroughly, to be promoted by application of various measures. A rich experienced qualitative skilled member seldom available, thus selection process for deployment is a crucial task, but not impossible. Hence appropriate selection model is one of efficient design factor. Making a fail-safe design, a tough task, but to maintanain the same is much more tougher.

C. Organizing Abilities

Well defined functional knowledge of software behaviour processes are the main areas of software safety to cover, a well understood member should have control of the entire software development process, should be authorised for organising in a systematic approach with indepth systemic process knowledge. Categorising them in an orderly manner, requires highest degree of organising abilities.

D. Efficient & Effective Life Cycle

Selection of efficient & effective development life cycle, more important, as it should update itself with the recent experiences. A cyclical model of development may be the best suitable for updated & documented uptodate data, information, should inrease or decrease the boundaries of execution with safety as main concern. This aspect automatically trace & track the defficiencies & efficencies, causes continuous improvement, acts as a guide for future expansion of the project.

E. Software Hazards Guide

History of software errors, detection methodologies, prevention techniques, failure mechanisms, combating the failure should be available handy, in a categorical manner. Since the history always makes a foundation future developmental issues, the necessity of historical reference is mandatory. However, these may vary from context to text, availability of relevant and appropriate manual is necessary.

F. Management Issues

The prime responsibility is for selection and assigning the right person in right place with desired personnel quality as indicated and required for the desired product delivery. There is need to consider qualities, skills like architectural, coding, system administration, systems management, data analytics, optimum utilization of resources, requirements analysis, hazards analysis, evaluation, testing etc., failing which the desired software safety could not be achieved.

G. Process Controls

A phase where prior definition of process should persist, which guide for on-going development process compliance with process models, if not, need to be eradicated or nip in the bud, cyclically until compliance is met with. Standard, periodical process measurement with statistical data balancing, rolling plan of technological changes, ascertained feedback, are the most essential components for construction of safety critical software.

H. Development Planning

Conducting and scheduling Hazards Analysis, Requirements analysis, Verification and Validation, Identification of Critical Components, Defect Tracking, Test plan are part of development planning.

G. Handling Safety Critical Issues

Analysis of safety critical factors that are potential to contribute to, which may be hardware, software, configuration, design, development, code, others etc., and handled by prevention, elimination, correction, or destroyed.

H. Reliability Assessment

Most of the accidents happen, of non-technological factors, combination irresponsibility and ignorance, being most dangerous, thus ultra-reliability cannot be assured. Reliability assessment for specified range of hazards analysis, can be, but not beyond.

I. Data Management & Documentation

Data vulnerability, being more prone to effect software behavior, better data management, data integrity need to be checked with available history and appropriately documented.

J. Evaluation by Validation & Verification

Evaluation of all processes by validation & verification form quality professionals, ecologists, safety professionals, software experts must be done.

VIII. Application to Road Traffic Control System

Less complex Road Traffic Control System (RTCS), a laboratory prototype, a Safety Critical Computer System Application, a modern traffic signaling system, comprising of many components, is expected to operate functionally to expectations with effectiveness and efficiency. Figure 3 diagrammatically represent the two road crossing forming four roads junction, with installed signaling system. The system consists of microprocessor based controller, driver, signal head status indicators and operator interface. The four roads leading to the junction are named as A, B, C and D with corresponding signal heads as shown on left hand side for each lane, with each signal head having three colored lights as per the standard color indicators, Red, Yellow and Green, as guidance for commuters whether to stop, proceed with caution or proceed.

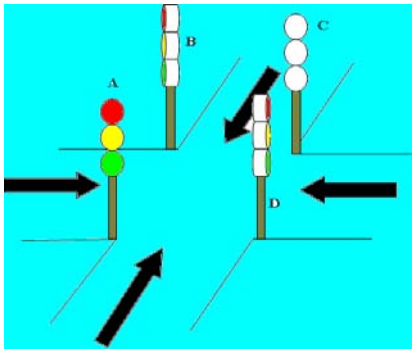

Fig. 3. Intersection of 2 Roads & Signal Indicator

A. Components of RTCS

Functional logic diagram of RTCS, is given in the figure 4, containing Operator Interface, Microprocessor based Controller, Driver, and Signal Heads with Status Indicators. *Operator Interface* communicates for different operations to be carried out, and for selecting a particular operation to be performed by the microcontroller. *Microprocessor Based Controller*, is responsible for operating the signal indications with synchronized timing schedules between signal heads, as per the traffic norms of traffic management, in compliance with pre-defined sequences. Signaling software is embedded. *Driver* acts as a mediator between signal head status indicators and controller, and controls as per pre-defined instructions. *Signal Head Status Indicators*, the current functional status of signal lights, at each of the points A, B, C, and D, as shown in the figure 4.

B. Normal Operations of RTCS

A primary check is being done when the system is switched on, for finding out any functional abnormalities for all systems and sub-systems involved, if not, proceed to execute the software defined for normal operations, continues to executes, with the timing schedules, else, alternative procedures are activated for any abnormal situation arises out of control due to situations beyond control like lightning, storm, etc., signal lights begin to blink indicating failure, implied suggestion to take necessary precautions. All of the points discussed above are thoroughly followed to implement, by collecting information from all concerned.

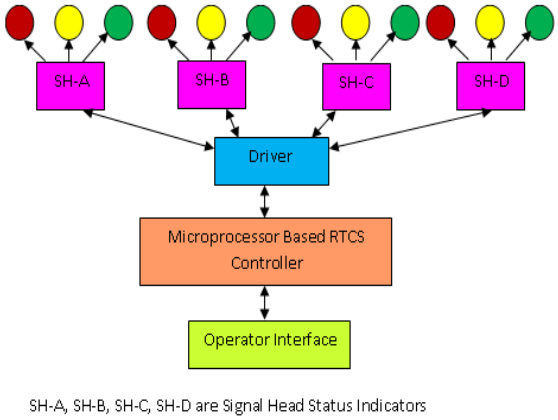

Fig. 4. Logic Diagram of RTCS Operations

IX. Conclusion

This paper discussed about the various organizational factors influencing software safety in detail. A set of integrated software team, for forming the basis of software safety contributing factors was suggested. This proposed method of integrating safety and software professionals is applied to laboratory prototype of four road junction traffic control system, as that includes safety critical operations, and observed meaningful, improved results and found safer. Further work is in progress to apply this integrated approach to software based safety critical systems in other areas of industrial applications like power generating stations. This can be further extended to address the issues in the developmental costs and time in implementation for software safety. Rigorous work is needed to meet the complete set of software safety requirements leading to the standardization of the framework.

References

[1] N.G. Leveson "*Software Safety – Why, What and How* ", ACM Computing Surveys, 18(2), pp 125-163, June 1986

[2] IEEE 100 "*The Authoritative Dictionary of Standard Terms* ", IEEE Press 2000

[3] MIS-STD-882B "*System Safety Program Requirements*", Department of Defense, 1984

[4] Department of Defense. "*System Safety Program Requirements*" MIL-STD-882C.1984

[5] *"Software Safety"*, NASA Technical Standard, 1997 *http://satc.nasa.gov/assure/distasst.pdf*

[6] N. Leveson "*Safeware : System Safety and Computers*" Addison Wesley Publishing Company, Reading, Massachusetts, 1995

[7] William R. Dunn, "Designing Safety Critical Computer Systems " Published by the IEEE Computer Society 0018-9162/03/$17.00 © 2003 IEEE (40-46)

[8] J. Dennis Lawrence G. Gary Preckshot 1994 *Design Factors for Safety-Critical software* http://www.llnl.gov/tid/lof/documents/pdf /228132.p

[9] Ben Swarup Medikonda and Seetha Ramaiah Panchumarthy, "A Framework for Software Safety in Safety-Critical Systems", http://doi.acm.org/10.1145/1507195.1507207 DOI: 10.1145/1507195.1507207

[10] Nancy G. Leveson, "The Role of Software in Spacecraft Accidents"

[11] Leveson, N.G., Turner, C.: "An investigation of the Therac-25 accidents" In IEEE Computer, July 1993, page 18-41

P.V. Srinivas Acharyulu, Assistant Engineer (IT), working in IT Department, since December 1992 (19 years) at NTPC Limited (A Government of India Enterprise) Prior to joining he was working as Programmer at GITAM college of Engineering from February, 1989 to November 1992 (4 Years) in Department of Computer Science & Systems Engineering. He obtained his Bachelor Degree in Science, from Sambalpur University in the year 1985 and Masters Degree in Computer Applications from IGNOU in the year 2003. He has both academic and industrial experience in Information Technology for more than 21 years. He is at present pursuing part-time doctoral degree in Computer Science & Engineering from GITAM University. His research interests are Safety Critical Computer Systems, Software Safety, Software Engineering and Operating Systems.

Dr. P. Seetharamaiah, Professor, Department of Computer Science and Systems Engineering, College of Engineering, Andhra University Visakhapatnam-530003, Andhra Pradesh – INDIA. Dr. Panchumarthy Seetha Ramaiah obtained his PhD in Computer Science from Andhra University in 1990. He is presently working as a professor of Computer Science in the department of Computer Science and Systems Engineering, Andhra University, College of Engineering, Visakhapatnam–INDIA. He is the Principal Investigator for several Defense R&D projects and Department of Science and Technology projects of the Government of India in the areas of Embedded Systems and robotics. He has published seven journal papers, and presented Fifteen International Conference papers in addition to twenty one papers at National Conferences in India.