Ganesh Panatula, Nagabhushan S.V, Dr. T. V. Suresh Kumar / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 5, September- October 2012, pp.1371-1375

Integration Analysis Of Safety Critical Systems In Information Technology

[1] Ganesh Panatula, [2] Nagabhushan S.V, [3]Dr. T. V. Suresh Kumar
1(Associate Professor , Dept. of MCA,BMSIT)
2(Assistant Professor, Dept. of MCA,BMSIT)
3(Professor&HOD , Dept. of MCA, MSRITT,)

Abstract

Safety-critical systems are those systems whose failure could result in loss of life, significant property damage, or damage to the environment. There are many well known examples in application areas such as medical devices, aircraft flight control, weapons, and nuclear systems. The emphasis of this paper is on the software element of safety critical systems, which for convenience is often referred to as safety critical software. Many modern information systems are becoming safety-critical in the sense that their failure results in financial loss and even loss of life. Future safety-critical systems will be more common and more powerful. From a software perspective, developing safety critical systems in adequate numbers and with adequate dependability is going to require significant advances in areas such as specification, architecture, verification, and process. The current project work is regarding fault tolerance client-server system that addresses safety-critical issue in the web application by suing smart server approach.

Keywords: Fault Tolerance System, Safety Critical System

NOMENCLATURE:

ADS- Autonomous Distributed System
ASCM-Adaptive Safety Critical Middleware
DESCS-Distributed and Embedded Safety-Critical Systems
FCS - Flight Control System
FTS - Fault Tolerant System
FMEA- Failure Mode and Effect Analysis
PHL - Preliminary Hazard List
PHA- Preliminary Hazard Analysis
QOS -Quality of Service
SCP -Self-Checking-Pair
SCS -Safety Critical System

I. INTRODUCTION

A safety critical system is a system where human safety is dependent upon the correct operation of the system. The emphasis of this paper is on the software element of safety critical systems, which for convenience is often referred to as safety critical software. However, safety must always be considered with respect to the whole system, including software, computer hardware, other electronic and electrical hardware, mechanical hardware, and operators or users, not just the software element.

Safety critical software has been traditionally associated with embedded control systems. As awareness of how systems can impact safety has developed, the scope of safety critical software has expanded into many other types of systems.

An obvious example of a safety critical system is an aircraft fly by wire control system, where the pilot inputs commands to the control computer using a joystick, and the computer manipulates the actual aircraft controls. The lives of hundreds of passengers are totally dependent upon the continued correct operation of such a system.

Moving down to earth, railway signaling systems must enable controllers to direct trains, while preventing trains from colliding. Like an aircraft fly by wire, lives are dependent upon the correct operation of the system. However, there is always the option of stopping all trains if the integrity of the system becomes suspect. It is not possible to just stop an aircraft while the fly by wire system is in use. [1].

Software in medical systems may be directly responsible for human life, such as metering safe amounts of X-rays. Software may also be involved in providing humans with information, such as information which a doctor uses to decide on medication. Both types of system can impact the safety of the patient.

Big civil engineering structures are designed on computers and tested using mathematical models. An error in the software could conceivably result in a bridge collapsing. Aircraft, trains, ships and cars are also designed and modeled using computers. Even something as simple as traffic lights can be viewed as safety critical. An error giving green lights to both directions at a traffic junction could result in an accident. Within cars, software involved in functions such as engine management, anti-lock brakes, traction control, and a host of other functions, could potentially fail in a way which increases the likelihood of a road accident.

A well conceived and executed safety case is a key element in bringing a safety critical system into use. In areas which have been traditionally concerned with safety critical systems, such as the aviation industry and the nuclear industry, a certification body

will have to be convinced that a system is safe before it is put into use. In some other areas, users have their own safety monitoring groups. Nevertheless, the vast majority of software safety is entirely in the hands and conscience of the software developers and suppliers.

II. LITERATURE REVIEW

By the analysis of the previous research done on safety critical systems, we highlight some of the major research done by the research scholars on safety critical issues.

The work proposed by Qing Sun Lirong et al. [2] states that there are three states in safety critical manufacturing systems namely working, fail-safe and fail-dangerous states. This paper studies two different safety-critical parallel-series models by considering their components' lifetime distribution possessing general forms. The indices of reliability and safety, including the probabilities that the system in these states and mean time for the system under two different failure ways, are derived respectively. Various corresponding indices comparisons between the two different parallel-series system models, and among the series, parallel and parallel-series systems, are conducted. Some illustrative numerical examples are employed to show the procedures. The derived indices formulae are without component lifetime distribution assumptions, which have significant meanings for reliability analysis and safety design of the system.

The work proposed by Zhang Yi et al. [3] states that Distributed and embedded safety-critical systems (DESCS) are those systems whose failure could result in loss of life, significant property damage, or damage to the environment. Because of the nature of DESCS, designing the applications for DESCS is harder than those for distributed real-time embedded systems. In this paper, a multilevel embedded safety-critical middleware called adaptive safety-critical middleware (ASCM) is described that provides related services to ease the development of embedded safety-critical applications. It also presents a multi-layer end-to-end adaptive QOS management technology to satisfy the dynamic and unpredictable mission requirements of DESCS.

The work proposed by By Nisha, G.R. [4] describes that faults in safety critical systems are the important elements to be avoided. To avoid these errors or faults, Fault Tolerant Systems (FTS) are evolved. But still, some more hidden design faults are not weeded out before realization by traditional and life cycle tests and analysis. In this paper, an approach has been described to find out these types of faults by simulating system architecture with modeling and simulation. By this model based methods, one can enable early verification of the system and quickly find errors or faults and deal with it. This is a very attractive approach, since the systems are critical systems. Through this method, one can test and analyze their system even before hardware as well as software realization. Also, nowadays, the cost of developing a critical system increases exponentially when system design errors are found after the implementation or integration of the system. In this paper, a model driven approach is described for design and development of a safety critical system. The stated approach is described along with functional description of electrical system architecture and the concept of modeling. Self-Checking-Pair (SCP) based computer architecture with high speed communication bus interface is modeled and the performance is evaluated before its hardware and software realization

The work proposed by Kumagai S et al. [5] explains that to secure modern complex system activities, rigid and heavily centralized organization is rather obstacle. To facilitate quick decision, mutual collaboration, and to maintain performability in unexpected severe situation, autonomous decision unit should act at best in intelligent way by itself. Autonomous distributed system concept plays a central role for operating complex systems commonly existing in today's networked society of 21st century. This paper explains the effectiveness of ADS system paradigm to secure safety critical systems and clarifies key issues to realize the objective as intended.

The work proposed by Hovakimyan N et al [6] narrates about the development of L_1 adaptive-control theory and its application to safety critical flight control system (FCS) development. Several architectures of the theory and benchmark examples are analyzed. The key feature of L_1 adaptive-control architectures is the decoupling of estimation and control, which enables the use of arbitrarily fast estimation rates without sacrificing robustness. Rohrs's example and the two-cart system are used as benchmark problems for illustration. NASA's flight tests on subscale commercial jet verify the theoretical claims in a set of safety-critical test flights.

III. MOTIVATION FOR THE PAPER

Today, most of the safety critical systems use a combination of the following architectures.

[a] Intrinsic Fail-Safe Design: This is generally used for discrete mechanical or electrical components where the credible failure modes of all components can be directly analyzed to ensure no unsafe conditions are created as a result of failure. Most systems use this technique as some portion of their system design, particularly for I/O or comparison mechanisms identifies a minimum set of credible failure modes to be included in the analysis.

[b] N-version programming: This technique requires at least two software programs, executing together and performing identical functions. An independent team using independent tools must write each software program. They may or may not run on independent hardware platforms.

[c] Numerical Assurance: This technique requires that the state of each safety-critical parameter be represented by a large encoded numerical value. Off-line data structures are defined such that real-time permissive results can only be calculated (by pseudo-randomly combining these values) when all the proper conditions for a permissive result are present.

With the above said architectures, we highlight the importance of safety verification & validation. It is necessary to provide verification to all the identified unacceptable or undesirable hazards so that they have been properly mitigated. In order to do this, all of the safety-critical functions necessary to implement the system (down to a very low level) must be identified. Functions that have to be implemented must be implemented fail-safely. The fail-safe implementation means that we look at all the credible failures that could occur and make sure that occurrence of any one of them (or combination of failures in the event that the first failure is not self-evident) maintains the system in a safe state, either by forcing the system to a stop (or other safe state such as a less permissive signal) or by transferring control to a secondary system (e.g. redundant computer).

As part of verification & validation in SCS, Functional Fault Trees are a widely accepted tool for identifying the safety-critical functions from the top down. These trees start at the upper level hazards previously identified and branch down through the system, subsystem and interfaces to identify all of the functions that, if not performed correctly, could precipitate the upper level hazards. The functional fault trees provide a top-down analysis of the system. A Failure Mode and Effect Analysis (FMEA) is also used to provide a bottom-up analysis of the system. Starting at the component level, all credible failures are analyzed to verify that they do not create any unsafe condition. It provides guidance on credible component failure modes. As with all safety analyses described so far, secondary failures must be considered in combination with the initial failure if the initial failure is not self-revealing.

Based on the above research findings, it is intended to design a safety critical system for web application and its implementation is showcased through a case study.

IV. PROPOSED WORK

It is intended to design and implement safety critical system for web Application based on n-tier client server system. We analyze the safety critical issues that are addressed in client -server system by providing fault tolerance using smart server approach.

Issues in client server system

In the corporate world, more prominence is given to data and securing data is the major concern for them to maintain confidence and long term relationship with the clients. Since they heavily depend on client data, maintaining their data for 24/7 is very crucial for them, but there are internal and external factor which creates negative impact on their business, if they lose the data. Maintain the same data in multiple places lead to data redundancy and also it is expensive. Since processing of data takes place at different levels and in different versions, keeping track of data in the consistent manner, it is very difficult.
So we investigate the above problem and solve it by using smart server approach. We design a framework to architect safety critical system for web application and it is implemented by taking case study as an example and show case how our approach is suited for web application.
Our framework is viewed from two core perspectives.

a) **Server Perspective:** In the server, business critical data is stored and clients are accessing the data for the live business transactions. Suddenly processing of the data stop's by external factors like server down, application crash, unexpected execution of some unknown event, virus intrusion etc and client has to wait until the application in the server is up. From the client perspective, this unexpected event is very expensive and no patience to wait. If the server scalability and performance issue is not handled during design and development, there are every chances that client may lose huge amount (scenario can be understood in online trading). To address the above issue, we introduce a middleware component in terms of proxy server which can create instances of running application in the dynamic environment In our approach, when unexpected event arises which stops the client from dynamic access,, an event is triggered and notification will go to the proxy server , In that server, from where main server has stopped the execution, proxy server takes the responsibility to process the remaining computation and put it in the off-line mode and store that part of information in a file and from that file, information will be passed to client. Hence client will have continuous access to data. At the time when main server notifies about its failure to proxy server, there will be some lag time which is unnoticeable at the client end.

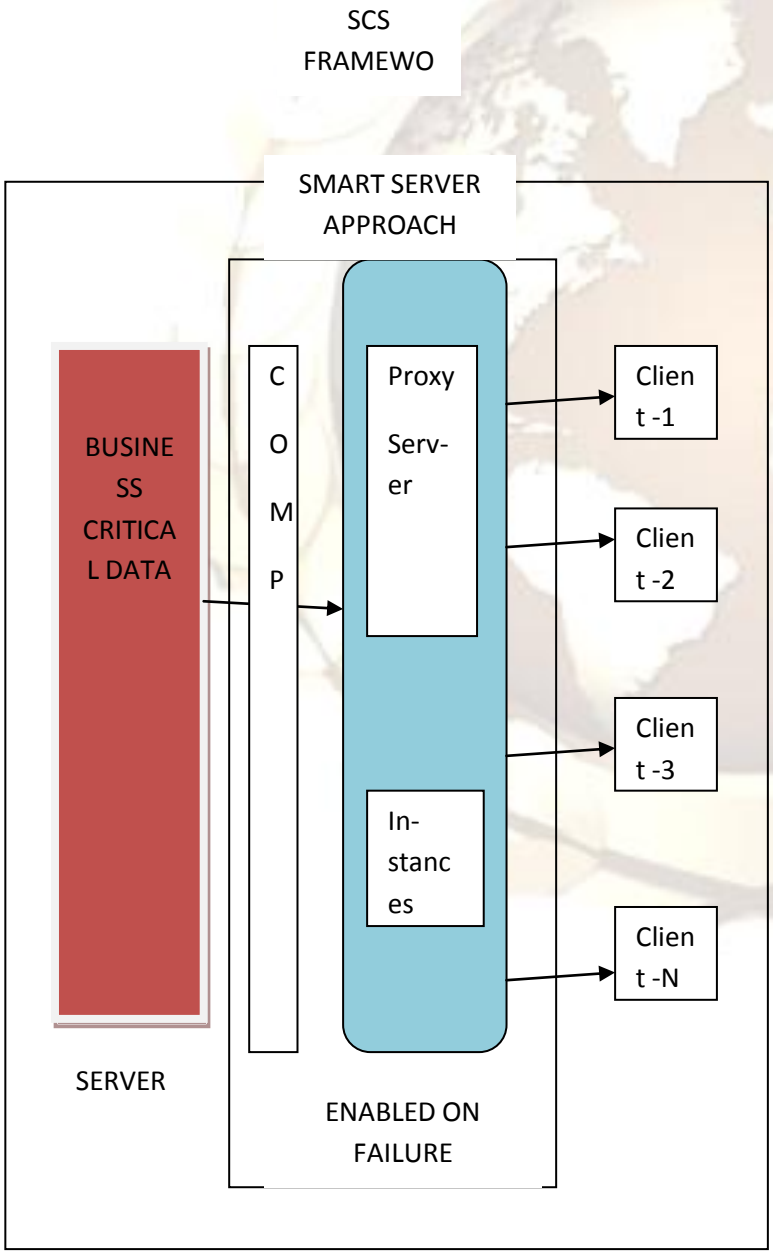
b) **Client Perspective:** Since clients are accessing the data continuously in critical applications stopping the application for a minutes will incur huge loss as the data is tracked every minute for some analysis (scenarios can be understood regarding the data coming from the satellite)[7], hence continuous access for the data is very much required. Client should have authentication to access the application and it can be any general Client Server setup application. At any point of time, the client should be assured to have con-

tinuous connectivity with the server without any disruptions.

By considering the above two perspectives, we study and analyze and implement safety critical system though case study for web application and experiment in our lab.

Case Study:

Assume an application is installed in the server machine and accessed by n clients (n=5, initially). Here clients access the application by knowing server IP address which is run in their browser in the following format http: // ipaddress/application.
We assume all the systems are interconnected through LAN. Each client is provided with an interface to login to the server. Each client accesses the server for different purpose, like downloading, uploading, reading documents etc. Now, if



the server is down abruptly, the client has to wait for the server to be up, in traditional approaches. Now, our approach suggests that inspite of server breakdown the client still can continue with normal operation. The methodology to facilitate the above is below.

At the time of server eventuality, the software component (COMP) detects the server failure and the recent updated information in the server is loaded to the Proxy server. At this juncture, the proxy server acts as a normal server and caters to the needs of client. In this process, care will be taken to avoid data redundancy. Once the main server is up, the COMP notices the same and loads back the recent updated information to the main server from the proxy server. Now main server resumes its normal work.

During the process of switching, as stated above, the time gaps need to be measured and reduced to the optimal extent.

V. CONCLUSION

Safety critical software is a complex subject. The approach discussed in this paper, is to make the system more reliable and to reduce the time gaps of context switching between main and proxy servers, in the web application environment. The custom application discussed in case study will provide a good example for simple approach to address the safety critical issue of fault tolerance in the client server application by providing the user with simple interface to interact with system and application availability in both online and offline condition.

ACKNOWLEDGEMENT:

We thank all the authors for the information support.

VI. FUTURE WORK

The scope of the work can be extended to reduce the time gap between main sever and proxy server. Also, once the main server resumes normalcy, data stored in the proxy server can be deleted thereby avoiding redundancy of data.

REFERENCES

[1] Briere, Dominique, and Traverse, Pascal, "Airbus A320/A330/A340 Electrical Flight Controls A Family of Fault Tolerant Systems", IEEE, Proceedings of 23rd International Conference On Fault Tolerant Computing, 1993.

[2] Qing Sun; Lirong Cui; Rong Pan; , "Modeling and analyzing safety-critical parallel-series system safety," Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on , vol., no., pp.2463-2467, 8-11 Dec. 2009

[3] Zhang Yi; Wandong Cai; Wang Yue; , "Adaptive Safety Critical Middleware for Distributed and Embedded Safety Critical

System," Networked Computing and Advanced Information Management, 2008. NCM '08. Fourth International Conference on , vol.1, no., pp.162-166, 2-4 Sept. 2008

[4] Nisha, G.R.; , "A model driven approach for design and development of a safety critical system," Electronics Computer Technology (ICECT), 2011 3rd International Conference on , vol.4, no., pp.15-18, 8-10 April 2011

[5] Kumagai, S.; Miyamoto, T.; Morihiro, Y.; , "Autonomous Distributed System Paradigm to Secure Safety Critical," SICE-ICASE, 2006. International Joint Conference , vol., no., pp.47-50, 18-21 Oct. 2006

[6] Hovakimyan, N.; Chengyu Cao; Kharisov, E.; Xargay, E.; Gregory, I.M.; , "$mathcal{L}_1$ Adaptive Control for Safety-Critical Systems," Control Systems, IEEE , vol.31, no.5, pp.54-104, Oct. 2011

[7] Spitzer, Cary.R, "Avionics Handbook" New York/CRC press/2001 4S Symposium, Small Satellites Systems And Services Proceedings: (2004)