Chapter – 6

Concurrent Reliability Evaluation and Design of a Software Component

6.1 Overview

This chapter is written in two parts – the first part, from section 6.2 to section 6.8, attempts to describe *Concurrent Failure Modes and Effects Analysis (CFMEA)* of the component based software system using a graph theoretic approach and suggests improvement over the present practiced procedures. Using this approach reliability evaluation of CBSS can be done effectively and extensively at the early design stage. The important feature of the approach is that it takes into account structural and functional interactive complexity of CBSS concurrently. This method works for both kinds of failures – dependent and independent. A digraph model and matrix approach is used to provide an in depth analysis and evaluation of failure modes and effects. The second part, comprising sections 6.9, 6.10 and 6.11, utilizes the graph theoretic approach to calculate *reliability* index of architecture based on heterogeneous architecture styles when the *reliability* of participating components and their interactions are known. Both the approaches are explained with case studies.

The chapter is organized in the following manner: Section 6.2 provides description of software *reliability* and the use of FMEA at an early design stage. In section 6.3 the component failure modes and effects and its digraph representation to model typical component based web application is discussed. In section 6.4 equivalent digraph matrix representation is developed for computer processing. Section 6.5 deals with the analysis of the failure mode and effects. Section 6.6 discusses the development of failure index by concurrent consideration of failure modes and effects. Section 6.7 provides the utility of the methodology by using typical component based web application (Hong, 2005). Section 6.8 presents the usefulness of the proposed *concurrent* FMEA. Section 6.9 covers the second part of the chapter and discusses the development of *reliability* index when *reliabilities* of constituent elements of design are known at the design stage, using approach based on chapter 2. In section 6.10 case studies are considered to calculate *reliability* index of architectural designs. Section 6.11 provides the comparative analysis of the approach with

some of the well established approaches. Finally section 6.12 presents the concluding remarks of the chapter.

6.2 Introduction

Software *reliability* is not a direct function of time although it is defined as a probabilistic function and comes with the notion of time. As compared to the hardware parts, software does not rust or wear-out during its life cycle. Also software does not change over time unless intentionally changed or upgraded. Software *reliability* is an important characteristic of software quality. Customers are placing increased demands on industries for just-in-time, high quality, reliable, distributed and large scale software products. To do so the concept of integrating pre-fabricated components (COTS) in order to achieve the functionality acts as a cornerstone in the software engineering discipline. Achieving *reliability* early in the development cycle is a major challenge and an extensive research area. Identifying component failures and failure mechanisms is essential in understanding, predicting and testing *reliability* of component based software system. Software failures may be the result of errors, ambiguities, oversights or misinterpretation of the specification that the software is supposed to satisfy, carelessness or incompetence in writing code, inadequate testing, incorrect or unexpected usage of the software or other unforeseen problems. Software faults are *design faults*, which are harder to visualize, classify, detect, and correct (Lyu, 1996). It is to be noted that once software is uploaded into the storage and once it starts running its quality does not change. Since design fault can not be masked off by voting thus higher software *reliability* can not be achieved by simply duplicating the same software components. Failure Modes and Effects Analysis (FMEA) is a methodology for analyzing potential reliability problems early in the development cycle where it is easier to take actions to overcome these issues, thereby enhancing *reliability* through design. FMEA is used to identify potential failure modes to determine their effects on the product operation and to identify actions to mitigate the failures. Anticipation of every failure mode is not possible, thus the development team should formulate an extensive list of possible potential failure modes. The general effects of the failure modes are due to the consequence of operations, functions or status of the system etc. The specific effects are decided based on the issues of complexity, cost operation and maintenance action, environmental and economic factors, quality etc. In order to formulate realistic effects, failure modes and effects for CBSS should consider interdependence of failure modes apart from the effect of

components. The benefit of using such approach at an early design stages allows the designer to design out failures and produce reliable, safe, and customer pleasing software products. The current chapter deals with *concurrent* FMEA approach which apart from considering the software component failure modes, their effects and actions to mitigate those effects' also considers the interdependence and the impact of failure modes in totality.

6.3 Concurrent Failure Modes and Effects Digraph Modeling (CFMEA)

In this chapter a failure is defined as the inability of a system or component to perform its intended function as defined by the specification. A failure is a consequence of faults (i.e. a defect with in the system or a component) or a bug that has been executed. When a fault is executed, an error propagates and becomes externally visible for an observer of a system or components and the failure occurs. Components can fail in different ways and the ways in which they fail can be categorized as failure modes. Failure modes are defined through their effects, as perceived by the component/ system user. Some of the potential failure modes of a component are – complete failure, partial failure, incorrect operation, failure to cease functionality at allotted time, intermitted failure, failure over time, premature operation and failure to function at the allotted time and failure to synchronize. Table 6.1 shows specific failure modes to components.

Reference	**Failure modes**
(Clarke and McDermid, 1993)	Control failure, value failure, addressing failure, termination failure, input failure
(Henrik and Anders, 2002)	Timing failure
(Chen and Burns, 1998; Kopetz and Reisinger, 1993)	Ordering failure, synchronization failure, interleaving failure
(Reifer, 1979)	Computational failure, logic failure, I/O data failure, data handling failure, interface failure, data definition failure, database failure, other
(Ristord and Esmenjaud, 2001)	Operating system stops, program stops with clear message, program stops without clear message, program runs, producing wrong results, the program runs producing apparently correct but in fact wrong results
(Lutz and Woodhouse, 1999)	Missing data, incorrect data, timing of data, extra data, halt/abnormal termination, omitted event, incorrect logic, timing/order
(Becker and Flick, 1996)	Software stops, failure message, checkpoint file failure, internal capacity exceeded, loss of service

Table 6.1 Failure modes of component

Failure modes effect can be at the local or at the global level. Each affected component can have some impact on other component or on the complete system. It is to be noted that failure mode in a component influences the other components in terms of interactive damage and may even change their operating conditions. The most common example under this category is processing wrong business logic.

From the above discussion it can be inferred that the process of component failure is often directly or indirectly affected by other component in the system. The state of the art literature does not provide models that explicitly consider influences of failure modes and effects in totality with respect to software domain. In Gandhi and Agrawal (1992) research work, failure modes and effects analysis using digraph approach is dealt for mechanical and hydraulic systems. The same approach is extended in the current chapter to model, analyze and evaluate software (CBSS) failure modes and effects concurrently. The chapter models this approach using digraph model and matrix approach. A CBSS concurrent failure modes and effects digraph (weighted) G_{cfmea} – (V, A), here onwards digraph is considered as weighted digraph, is defined to represent the structural and functional interactive complexity of failure modes and effects of CBSS in terms of nodes and edges. The node set V = (C_i), i = 1…n, represents the failure modes of sub-system/component (e.g. complete failure, increased downtime, reduced reliability, low usability etc.). The edge set A = (a_{ij}), represent the interactions of failure mode and effect of sub-system/component (e.g. strong, medium, weak, none). Designers are free to put specific/required attribute to both nodes and edges depending upon the level of analysis to be done for CBSS. The number of nodes in a digraph is equal to the number of sub-system/component of the CBSS. Sub-node C_{im}, corresponds to the effect of m^{th} *failure* mode of i^{th} sub-system/component and is placed in the node v_i representing the sub-system/component.

A case study presented in chapter 2 is taken to illustrate the methodology. The component based typical web application (an insight to e-business) in Figure 1.2 consists of four components (Hong, 2005) – client side User interface (*UI*) (C_1), *web (application) server* (C_2), *database server*(C_3) and *page generator*(C_4). Each of the four components can fail in various modes. The failure modes of *UI* include: incompatible browser, low memory condition, inappropriate display, network failure, client crash etc. Table 6.2 lists out the failure modes (not exhaustive) of all four components (sub-system) of CBSS.

Each failure mode affects the system and is dependent upon the criticality of the component/ sub-system, component/ sub-system consequences etc. The *UI* failure mode may be a low memory condition. This means that user can view file only as per specification of the available memory. If the memory to hold information is 512 KB and server is sending 10 MB file then the consequence of this will be longer wait cycle to see information or only patches of information will be viewed. This leads to low efficiency and low usability of *CBSS*. Also, failure mode and effect of one component may interact with the failure mode and effect of other component/ sub-system. In general, all failure modes and effects of the component/ sub-system interact with failure modes and effects of other components/ sub-systems. This can be depicted in Figure 6.1.

Tag ID	**Component**	**Failure Mode**	**Modes interactions**
1	User Interface (UI)	11. Incompatible browser	13
		12. Low memory condition	13
		13. Inappropriate display	-----
		14. N/W failure	11; 13
		15. Client crash	11; 13
2	Web Server	21. Overloading	13; 22;
		22. Operating system stops	---
		23. Inappropriate business logic	21; 13; 15;32
		24. Internal N/W failure	23; 21
3	Database Server	31. Missing metadata information	32
		32. Inappropriate stored procedures & triggers	33; 41
		33. Data inconsistency & redundancy	32; 34; 43;13
		34. Poor indexing	33;
		35. Poor data design	31; 32; 34; 13; 33;41
		36. Internal N/W failure	
4	Page generator	41. Processing inappropriate request	13; 43
		42. Overloading	44
		43. Data mismatch	41; 44; 13
		44. Crash	---
		45. Internal N/W failure	41; 42

Table 6.2 Failure modes (partial list) description and interaction for component based web application

The digraph represented in Figure 6.1 is complex as all the possible interactions of failure modes and effects are considered. The level of interaction depends upon the problem domain and scope of CBSS**.** It is to be noted that the number of nodes is equal to the number of component/ sub-system in a system whose effects are to be considered. Also, the

number of sub-nodes is equal to the number of failure modes of a component. Industry is free to identify more or altogether different set of failure modes and effects for CBSS as per the domain requirement. The complexity of the digraph to represent all possible failure modes and effects interaction increases when CBSS consists of more sub-systems/components. Typically, CBSS application does not have to consider all interactions. Figure 6.2 represents failure modes and effects analysis of a special case as per Table 6.1.

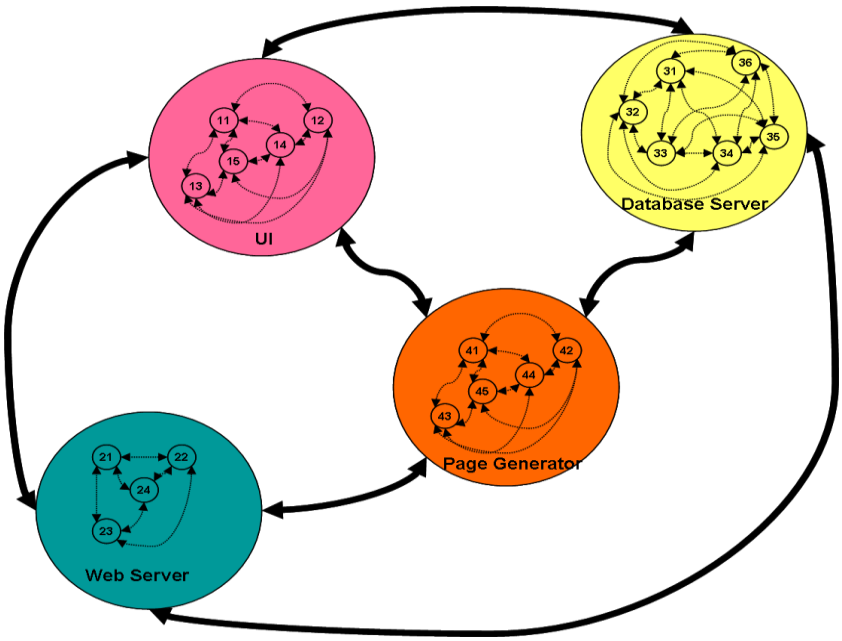


Figure 6.1 Complex failure modes and effects digraph for component based web application

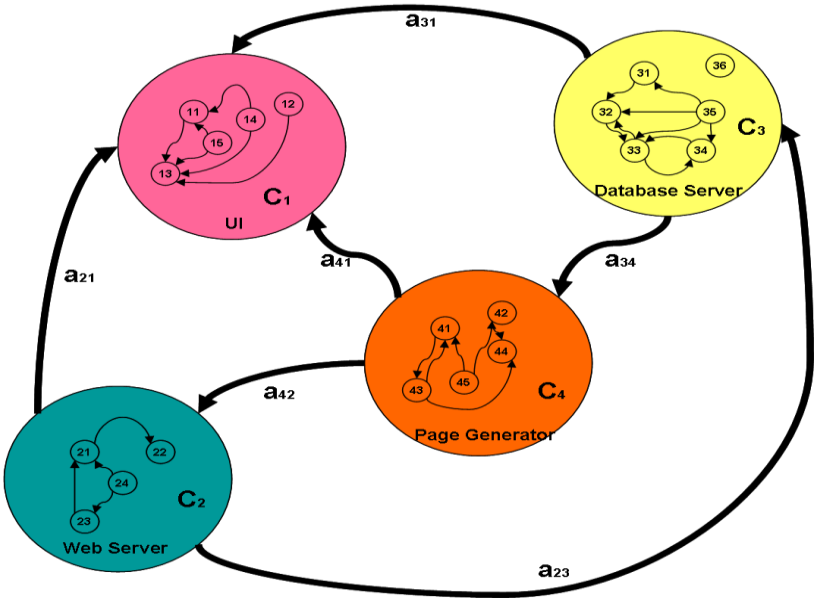


Figure 6.2 Specific case digraph for component based web application

It can be inferred from Figure 6.2 that it represents less complexity as compared to Figure 6.1 as only selected interactions of failure modes and effects of components are considered. The digraph representation gives mainly visual representation and helps in the analysis to a limited extent only. In case if a CBSS's sub-systems/components are large in number then their corresponding CFMEA digraph will be complex and this will complicate the visual understanding. In view of this, matrix (permanent) is developed which is one-to-one representation of the considered CFMEA digraph. This will help in establishing expression characteristics of CBSS failure modes and effects and it will be convenient for computer processing. If a CBSS CFMEA digraph represents/ contains N nodes (sub-system/component), then its matrix representation will be of size $N * N$. In the matrix the *diagonal* elements represent effect of failure modes on component/sub-system while *off-diagonal* elements correspond to the effects of one component/sub-system on another component/sub-system (direct or indirect interactions of failure modes). Permanent of a matrix is called *variable permanent concurrent failure modes and effects function* (*VPF – cfmea*). This expression is the system characteristic of failure modes and effects by considering all structural and interactive complexity. As the CFMEA digraph is crucial for the study, this may be prepared by the experts in the area/domain.

6.4 Matrix Representation

In this section, concurrent failure modes and effects digraph of CBSS is represented using permanent matrix, see chapter 2. Let us consider the digraph shown in Figure 6.2 for defining variable permanent concurrent failure modes and effects matrix for CBSS. Let a *diagonal matrix,* F_D, with diagonal elements C_i, $i = 1, 2,...,4$ is considered. Here, C_i represents the failure modes and effects of the i^{th} component, whose value can be obtained by considering level of effect/consequence of failure modes to the component. These can also be represented using vector entity or all together with the *permanent*. Such type of consideration is not present in other research work, and therefore the diagonal elements are assigned zero value. This results into weak analysis of failure modes and effects for a system. Let us also define another matrix, F_O, with *off-diagonal* elements, a_{ij}*'s,* representing the failure modes and effects between component i and j. It is to be noted that C_i*'s* and a_{ij}*'s* represent nodes and edges respectively in failure modes and effects digraphs of the system

(*CBSS*). The matrix, variable permanent concurrent failure modes and effects matrix (*VPM – cfmea)*, for Figure 6.2 is written as:

$VPM - cfmea = [F_D + F_O]$

$$VPM - cfmea = \begin{matrix} \\ C_1 \\ a_{21} \\ a_{31} \\ a_{41} \end{matrix} \begin{matrix} 1 & 2 & 3 & 4 \\ \hline C_1 & 0 & 0 & 0 \\ a_{21} & C_2 & a_{23} & 0 \\ a_{31} & 0 & C_3 & a_{34} \\ a_{41} & a_{42} & 0 & C_4 \end{matrix} \begin{matrix} \textit{Components} \\ \hline 1 \\ 2 \\ 3 \\ 4 \end{matrix} \tag{6.1}$$

The *permanent* function of the matrix following standard procedures is written as:

$$VPF - cfmea = C_1 * C_2 * C_3 * C_4 + C_1 * a_{23} * a_{34} * a_{42} \tag{6.2}$$

The inclusion of *permanent function* to analyze failure modes and effects in CBSS is completely new and is a significant contribution in *CFMEA* which leads to an effective reliability design and evaluation of CBSS. Expression 6.2 consists of various terms as point of combinatorial mathematics. Each term of the expression (function) yields a test/heuristic. It may be noted that due to the selective failure modes and effects of components, few terms appear in the multinomial (expression 6.2). The maximum failure modes and effects complexity yields to the complete expression consisting of $N!$ terms. This expression considers maximum failure complexity of the system where each component is affected by other components. Expression 6.2 in symbolic form is a useful tool for analysis, and terms can be arranged in a number of groupings. It is to be noted that mathematically each term is a product of four different matrix elements ($N = 4$, in the example). Interpreting same expression it can be noticed that different terms are the set of distinct diagonal elements (C_i, $C_1,\ldots,C_4$) and loops of *off-diagonal* elements of different sizes ($a_{ij}a_{ji}$, $a_{ij}a_{jk}a_{ki}$). This generates new meaning to multinomial from *CBSS* structural and functional interactive point of view considering failure mode and effects. By arranging terms of the same structure in the same grouping and of different structure in different groupings, *VPF – cfmea* may easily be written in $(N + 1)$ groupings.

6.5 Concurrent Failure Modes and Effects Analysis

The *VPF – cfmea* is a useful expression for the failure modes and effects analysis of CBSS as it is an invariant of CBSS failure modes and effects. If there are *M* distinct terms then there will *M* distinct ways for analyzing failure modes and effects of CBSS**.** The overall analysis is done in the following manner:

1. The first group represents failure modes and effects of components (i.e. C_i) and is written as

   $$/C_1/C_2/\ldots/C_n/$$

   A slash represents separation mark between two entities. A designer or a practiced expert in the field/domain needs to consider the failure modes and effects with an aim to analyze the failure mode and effects and suggest ways and means to minimize its contribution in affecting the component and the system. For example, if the analysis is carried out for the failure modes and effects of *CBSS* i.e. typical web application, the first set is:

   /*UI*/*Web Application Server*/*Database Server*/*Page Generator*/

   $$/C_1/C_2/C_3/C_4/$$

   If the entity 2 i.e. C_2 (web application server) failure modes and effects is more critical then an in-depth study may reveal that this attributes to overloading, deadlock, processing of improper business logic, synchronization, system crash etc. By the application of appropriate techniques, the failure modes and effect of the web application server can be reduced. Using similar approach, the failure modes and effects of the other components are also considered.

2. The second group is absent as there is no self loop present in the digraph.

3. The third group consists of two – component failure modes and effects loop (i.e. $a_{ij}a_{ji}$) and failure modes and effects of $(n - 2)$ components. The set is represented as

   $$/a_{ij}a_{ji}\ /C_3/C_4/\ldots/C_n/$$

   or

$$/A_{ij}/C_3/C_4/\ldots/C_n/$$

For convenience $a_{ij}a_{ji}$ is represented as A_{ij}. In the above term the entity to be analyzed first is $a_{ij}a_{ji}$. This is a 2 – component failure modes and effects loop and represent failure modes and effects relation between *i* and *j*. If this value is towards higher side as per the analysis, then an in-depth study is needed to reduce its value. For the present failure modes and effects of CBSS i.e. typical web application, the third group terms contains zero entities

4. The fourth group consists of a three – component failure modes and effects loop i.e. A_{ijk} or its pair A_{ikj} and failure modes and effects of (n - 3) components. The group term is represented as:

$$/A_{ijk} + A_{ikj} /C_3/C_4/\ldots/C_n/$$

The first entity to be analyzed is the 3 – component failure modes and effects loop A_{ijk} or its pair A_{ikj}. If the analysis indicates the entity's value is vital (higher side), then efforts should be made to reduce this value. For the present failure modes and effects of CBSS i.e. typical web application, the fourth set is:

$$/(a_{23}*a_{34}*a_{42})/ C_1/$$

The first entity to be analyzed in the set is A_{234}. This is the resultant interaction relation between components 2, 3 and 4. That is the failure modes and effects between the *web application server* and the *database server* and between the *database server* and the *page generator* and between the *page generator* and the *web application server* have to be studied. Along with this resultant failure modes and effects among these components and the other entity have to be considered and studied.

5. The entities of the fifth grouping are arranged in two sub-groupings. The entities of the first sub-grouping are two – component failure modes and effects loop (i.e., $a_{ij}a_{ji}$).The second sub-grouping consists of a four-component failure modes and effects loop (i.e., $a_{ij}\ a_{jk}\ a_{kl}\ a_{lm}\ a_{mi}$) or their pairs ($a_{im}\ a_{ml}\ a_{lk}\ a_{kj}\ a_{ji}$). For the present

failure modes and effects of *CBSS* i.e. typical web application, the fifth set contains zero entities.

Various means and ways to reduce the value of the entities can be suggested and employed. Finally, this leads to the minimization of failure. Similarly other set of the expression can be analyzed. It is to be noted that from the foregoing discussion and analysis it may be inferred each set of the *VPF – cfmea* derives some useful conclusion. It is preferable to consider the values of C_i and a_{ij} based upon the failure profile or experience of the domain personnel. However, these are to be normalized and placed on a suitable scale. Here C_i is quantified considering all failures modes of the i^{th} component. To compute this, C_i is represented as *VPF-cfmea*(C_i) of the permanent matrix, where diagonal elements, [$C_{i1,}$ $C_{i2,}$ $C_{i3,\ldots,}$ C_{im}] represent the failure modes and effects of the i^{th} component and a_{ij} are the effects of i^{th} failure mode on j^{th} failure mode. For the computation purpose C_{im} is quantified on a scale of 1 – 10, with value of 1 indicating extremely negligible effect of failure mode and 10 indicating a major effect. Table 6.3 aids in assigning the value to C_{im} on the consideration of safety and risk. Based upon the discussion C_i is calculated. The degree of effect of interaction among failure modes and components is shown in Table 6.4.

Description	C_{im}
Extremely low – causing minor inconvenience only	1
Very low– causing major inconvenience only	2
Minor – loss of minor operating mode or capability	3
Low – loss of major operating mode or substantial performance	4
Significant – total loss of operating capability	5
Moderate – minor damage to system	6
High - major damage to system only	7
Very High - major damage to system including fatality of operating/system personnel	8
Exceptionally high - major damage to system including fatality of number of operating/system personnel and affecting surrounding environment and population slightly	9
Catastrophic - fatality of very large number of operating/system personnel and major effect of environment and population around	10

Table 6.3 Effects of failure mode of a component (C_{im})

Qualitative description of effect relation between two elements (failure modes/components)	**Degree of relation**
Strong	3
Medium	2
Weak	1
None	0

Table 6.4 Degree of effect of interaction among component failure modes

It is to be noted that the range value for the consequence of the i^{th} component has to be identified. The maximum range value of C_i is assigned 10 and the minimum range value is 1. This is true, if the value of the failure modes and effects of the component increases it implies increased consequence. If this is not true then the maximum range value of C_i is 1 and the minimum range value will be 10. The intermediate values are to be assigned accordingly. Similarly, a_{ij} is assigned value 0 – 3. It is possible that data pertaining to C_i and a_{ij} are not available. In such cases, these are assigned value based on Table 6.3 and Table 6.4.

6.6 Concurrent Failure Modes and Effects Index (I_{cfmea})

An index/measure will help in evaluating the failure modes and effects of components and/or component based software system quantitatively or qualitatively. The concurrent failure modes and effect index (I_{cfmea}) is a quantitative measure of the CBSS. This means it indicates the extent of the consequence in the event of the possible failure modes on component and/or system. As *VPF – cfmea* considers structural and interactive complexity of failure modes and effects it can be used to generate the measure. Based on the I_{cfmea} the selection and evaluation of the component and/or system can be carried out as per failure modes and effects point of view. To evaluate *VPF – cfmea*, numerical values of C_i and a_{ij} are required.

Using the failure modes and effects index **(I_{cfmea})**, comparison of the two CBSS designs can be carried out on the basis of failures. High value of I_{cfmea} indicates high effects (consequences) of the CBSS failure modes and effects. This high value of the index is due to high values of C_i and a_{ij}*'s*. The lower value of index implies lower effects of the CBSS*'s* failure modes and effects and is a result of low values of C_i and a_{ij}*'s*. Based on this, given CBSS product design or a family of CBSS products designs are compared and may be ranked from increasing or decreasing value of index. This helps in the selection of an optimum CBSS product design based on the reliability (failure mode and effect). Such a tool is useful to a designer especially for the selection of a best structure for the CBSS product, among the possible alternatives, at the initial stage of design based on failure mode and effects. Permanent, expression 6.2 is a function of failure modes and effects of a number of distinct structural components i.e. C_i, $(a_{ij}a_{ji}/a_{ij}^2)$, $(a_{ij}a_{jk}...a_{mi})$ etc. Decision maker may compare these component failure modes and effects as well as single failure

modes and effects index of competitive candidates. This comparison provides complete insight for making right selection of CBSS product design for a given application.

6.7 Case Study - Development of Concurrent Failure Modes and Effects Index

The applicability of the approach is validated with the case study of a typical component based web application. Table 6.5 lists down the failure modes (partial list; not exhaustive) for each of the components. It also lists the failure modes and effects of the components and the system and also interaction among components.
The effect of i^{th} component on j^{th} component, a_{ij} can be computed in following ways:

- if only one failure mode of i^{th} component is affecting only one failure mode of j^{th} component then direct value from Table 6.4 is used in place of a_{ij}.
- if only one failure mode of i^{th} component is affecting many failure modes of j^{th} component then value of a_{ij} is computed by taking root mean square value of all interactions.
- if many failure modes of i^{th} component is affecting only one failure mode of j^{th} component then value of a_{ij} is computed by taking root mean square value of all interactions.

It may be noted that the value of C_i*'s* and a_{ij}*'s* are obtained by the design of components and their expected performance. To obtain the realistic values, Table 6.5 may be prepared by experts. For calculation of the I_{cfmea} the permanent of the matrix is evaluated based on the values of Table 6.5:

$$I_{cfmea} = 7.431782e+09$$

The index is the estimated index of the failure modes and effects of typical web application. This may be compared with the other alternate candidate design indices. It is also to be noted that each component can also be treated as sub-system/composition and same procedure can be applied up to the component level for an in depth analysis.

Tag ID	Component	Failure Mode	Failure modes and effects						
			of the component			among other component	System	Modes interactions	a_{ij}
			Description	C_{im}	C_i Per (C_i)				
1	UI	11. Incompat-ible browser	No support for extra features	2	128	None	Less usable	13	1
		12. Low memory condition	Longer wait cycle to see information -- Patches of information can be viewed	2		None	Low efficiency	13	2
		13. Inappropr-iate display	Dissatisfaction of User	2		None	Less usable & Less understanda ble		0
		14. N/W failure	Services not available	4		None	---	11; 13	1; 1
		15. Client crash	UI not available	4		None	Services cannot be accessed	11; 13	1; 1
2	Web Server	21. Overload-ing	Slow processing of the request	2	90	1, 2	Less efficient in giving services	13; 22;	0; 2
		22. Operating system stops	Server stop working	3		None	Services not available	---	0
		23. Inappropr-iate business logic	Server busy in processing logic for longer time; overloaded with processing	5		1, 3	Inefficient information; Poor usability and poor understanda bility	21; 13; 15;32	2; 2; 1; 2
		24. Internal N/W failure	----	3		None	Services not available	23; 21	0; 1

Continued…

Tag ID	Component	Failure Mode	Failure mode and effects						
			of the component			among other component	System	Modes interactions	a_{ij}
			Description	C_{im}	C_i Per (C_i)				
3	Database Server	31. Missing metadata informati-on	unmanageable data;	4	3360	None	Requested information not available	32	2
		32. Inappropriate stored procedur-es & triggers	Make server overloaded; produces wrong or inappropriate results	4		4	Generates wrong and improper set of information	33; 41	2; 2
		33. Data inconsist-ency & redundanc y	Produces unmanageable data; inappropriate data	3		1, 4	Leads to unmanageab le page generation	32; 34; 43;13	2; 2; 2; 1
		34. Poor indexing	Take longer time for processing and producing output	3		None	Low efficiency in generating results	33;	1
		35. Poor data design	Make server overloaded; take longer processing time; Produces unmanageable data; inappropriate data	5		1, 4	Low quality of results	31; 32; 34; 13; 33;41	3; 3; 3; 1; 3; 2
		36. Internal N/W failure	---	3		None	Service not available		0

Continued…

Tag ID	Component	Failure Mode	Failure mode and effects						
			of the component			among other component	System	Modes interactions	a_{ij}
			Description	C_{im}	C_i Per (C_j)				
4	Page generator	41. Processing inappropriate request	Makes server overloaded	2	192	1, 2	Leads to unmanageable page generation	13; 43	1;2
		42. Overloading	Slow processing of the request	2		None	Low efficiency	44	2
		43. Data mismatch	Produces unmanageable page format	2		1	Leads to inappropriate page generation	41; 44; 13	2;2; 1
		44. Crash	---	4		None	Page cannot be generated	---	0
		45. Internal N/W failure	-----	3		None	Services not available	41; 42	1;1

Table 6.5 Description and value of failure modes and effects for typical web application (partial list)

6.8 Usefulness of the Concurrent FMEA

Concurrent FMEA is developed to assist the designer and the developer to identify the potential component and/or design failures in order to improve the quality and reliability of design. If properly used, the CFMEA provides several benefits to engineers and failure experts. The approach can be used as to:

- Develop component based software design requirements that minimize the likelihood of those failures.

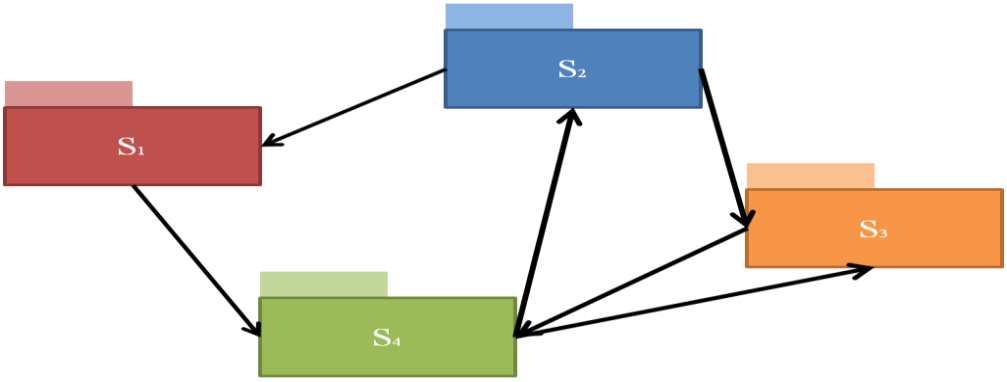
- Evaluate the requirements obtained from the customer or other participants in the design process to ensure that those requirements do not introduce potential failures.
- Identify design characteristics that contribute to failures and design them out of the system or at least minimize the resulting effects.
- Identify interactions and impact among failure modes and components.
- Develop methods and procedures to develop and test the product/process to ensure that the failures have been successfully eliminated.
- Track and manage potential risks in the design. Tracking the risks contributes to the development of corporate memory and the success of future products as well.
- Ensure that any failures that could occur will not injure or seriously impact the customer of the product/process.

A concurrent FMEA, acts as an important artifact during the software development life cycle. If at any time updates and/or changes happen it should be incorporated in the digraph model followed by matrix representation. These changes can and often do introduce new failure modes. Thus proper modeling of digraph considering interactive complexity of failure modes will enhance the capability of improving reliability. It is therefore important to review and/or update the concurrent FMEA when: new CBSS process is being initiated (at the beginning of the cycle); changes are made to the operating conditions the software product or process is expected to function in and a change is made to the software product design.

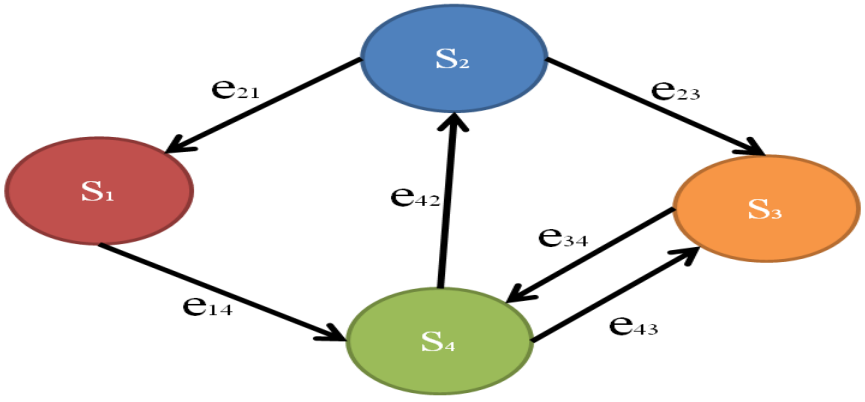
6.9 Design Reliability index (I_r)

Part II of this chapter describes the method for calculating the design *reliability* index (I_r) of software architecture based on heterogeneous architectural styles. When the *reliability* of the architectural elements is known then the approach developed in chapter 2 can be utilized to calculate design *reliability* index of component based software system at the design level. In this section, first a hypothetical example is considered to create a proper base for calculating (I_r) and then a case study is used to demonstrate the applicability and validity of the approach.

The system *reliability* graph *SRG* = (S_i, e_{ij}), consists of a set of nodes or vertices; S_i = (S_1,...,S_2), and a set of edges; e_{ij} =(e_{11}, e_{12},…, e_{ij}). The node (S_i) represents the reliability of the sub-system, and the edge (e_{ij}) the *reliability* of interaction/connection between the sub-systems S_i and S_j. Various types of edges and weights can differentiate the type of interactions, for example, for undirected edges interaction/connection means that – elements communicate with each other, control each other, send data to each other, invoke each other, synchronize with each other, share some information-hiding secret with each other etc. For directed edges the aforementioned interactions become unidirectional. Each node of a system reliability graph of a sub-system consists of a number of components/sub-sub-systems. To show the operational flow of the methodology, hypothetical CBSS architecture based on batch-sequential and call-return style, which comprises of four sub-systems- S_1, S_2, S_3, and S_4 is assumed. The system reliability graph (SRG_1**)** based on UML diagram, Figure 6.3 (a), is shown in Figure 6.3 (b). The four nodes (UML packages) represent respective sub-systems of CBSS and edges corresponding to the connections/interactions between the subsystems. In section 6.12, a real case study is considered to show the applicability and validity of proposed approach.



6.3(a) ***CBSS*** architecture/structure



6.3(b) ***CBSS*** system reliability graph ***SRG₁***

Figure 6.3 Architecture and system reliability graph of CBSS

Let a *diagonal matrix*, R_D, with diagonal elements R_i, $i = 1, 2,\ldots,4$ be considered. Here, R_i represents the reliability of i^{th} sub-system, whose value can be obtained looking at failure profile or historical data. Let us also define another matrix, R_O, with *off-diagonal* elements, r_{ij}*'s,* representing the reliability of interaction/edge between subsystem *i* and *j*. It is to be noted that R_i*'s* and r_{ij}*'s* represent nodes and edges respectively in SRG_1 digraph of the system (*CBSS*). A variable permanent reliability matrix (*VPM- r*) of *SRG* is written as:

VPM- r = $[R_D + R_O]$

$$VPM\text{-}\ r = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{bmatrix} R_1 & 0 & 0 & r_{14} \\ r_{21} & R_2 & r_{23} & 0 \\ 0 & 0 & R_3 & r_{34} \\ 0 & r_{42} & r_{43} & R_4 \end{bmatrix} \end{array} \begin{matrix} \textit{Subsystems} \\ 1 \\ 2 \\ 3 \\ 4 \end{matrix} \qquad (6.9)$$

It is a complete representation of CBSS, as it does not contain any negative sign. This means that it preserves all the structural information about dyads, loops of systems, or system attributes for elements reliability even in the numerical form. The permanent of the matrix, *VPM- r*, is written as *VPF- r*:

$$VPF\text{-}\ r = R_1R_2R_3R_4 + R_1R_2r_{34}r_{43} + r_{23}r_{34}r_{42}R_1 + r_{14}r_{42}r_{21}R_3 \qquad (6.10)$$

It is possible to write these expression (6.10) simply by visual inspection of the CBSS system of Figure 6.3 (b) as every term corresponds to a physical subsystem of the complete system. To achieve this objective, the permanent function of expression (6.10) is written in a standard form as (N + 1) groups. All these distinct combinations of sub-systems and interactions of the macro system are shown graphically in Figure 6.4.

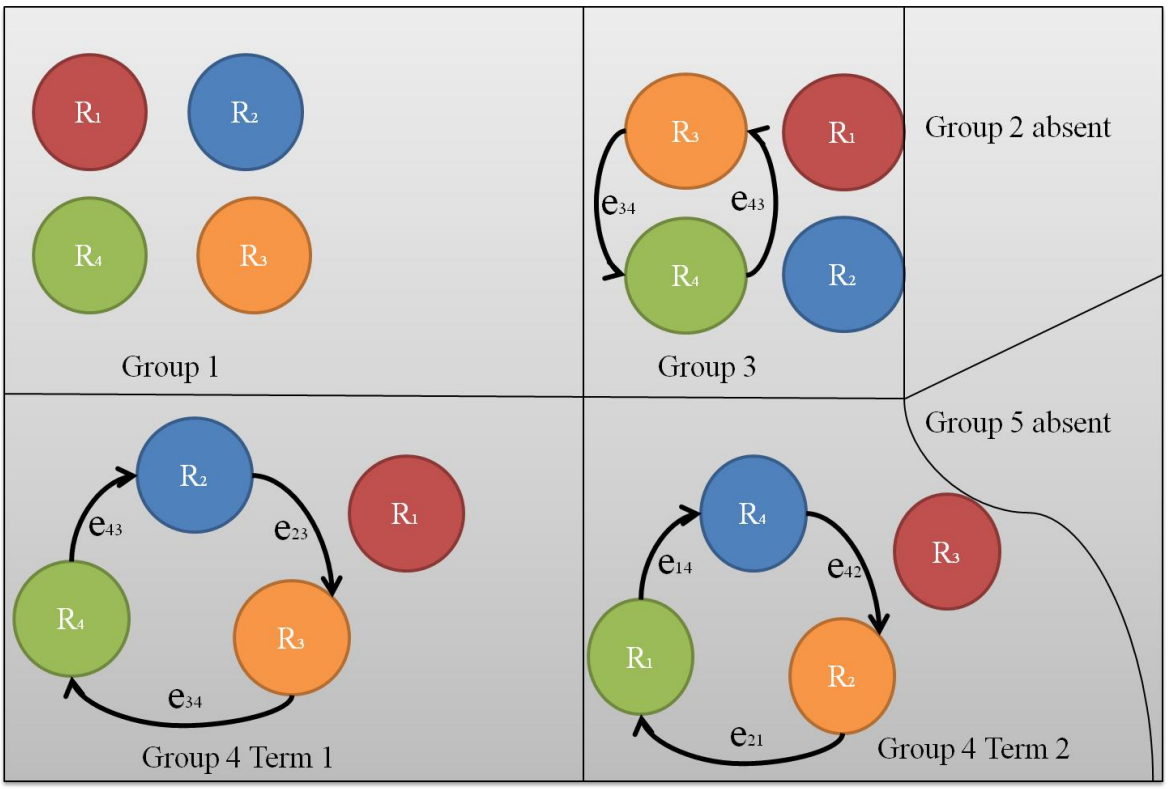
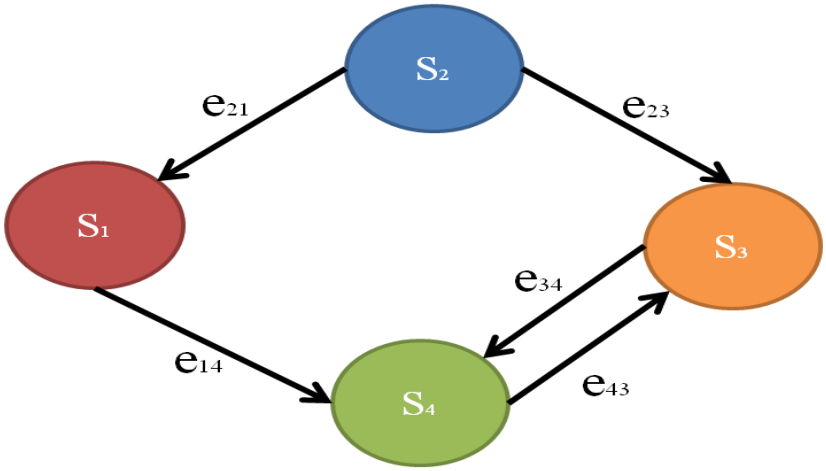
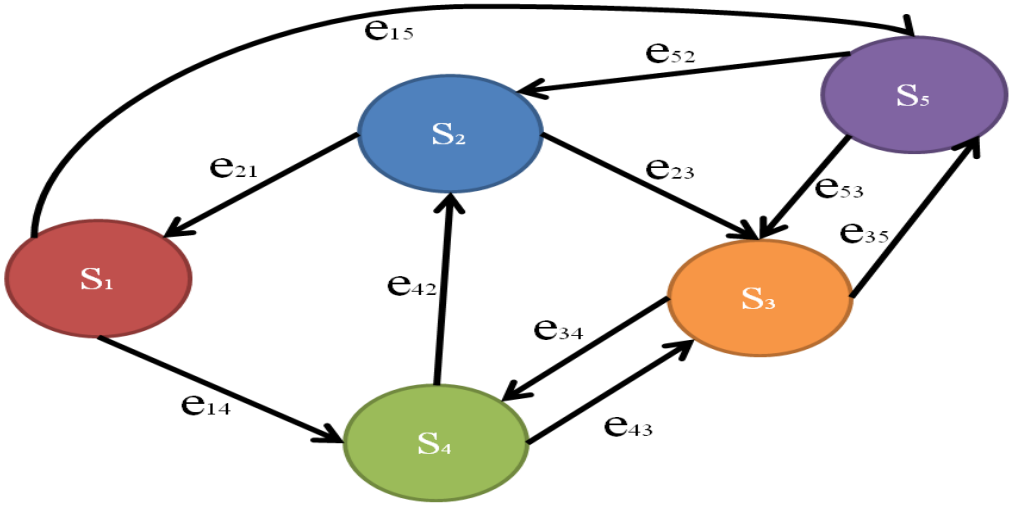


Figure 6.4 Graphical/physical interpretations of group terms

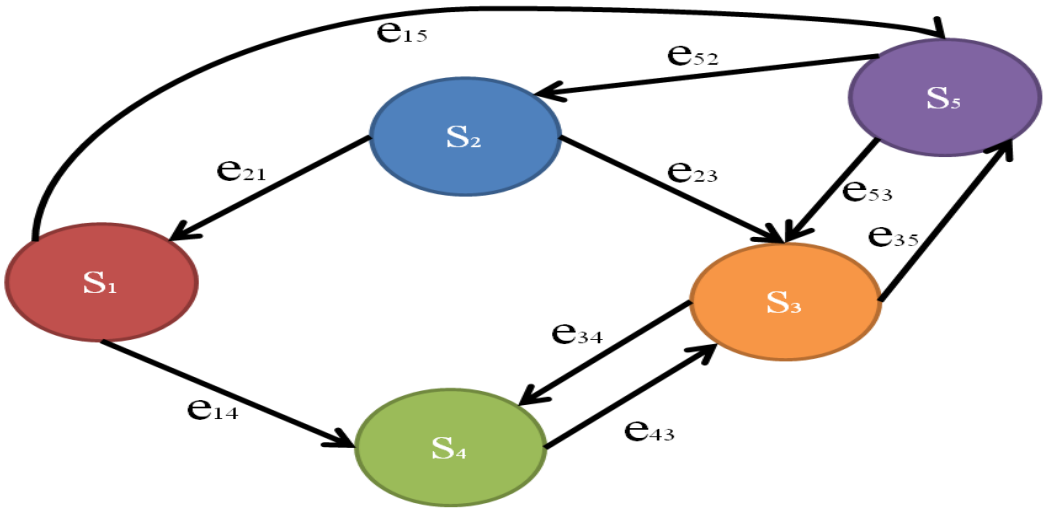
Coefficient of similarity/dissimilarity (see chapter 2) can be used to compare two given CBSSs on the basis of architecture/structure and also on the basis of reliability point of view. To illustrate this, three variants of SRG_1, Figure 6.3, are considered, Figure 6.5(a), Figure 6.5(b) and Figure 6.5(c). The SRG_2, Figure 6.5(a), is identified by deleting an edge e_{42} from SRG_1 of Figure 6.3(b). This shows that there is no interaction between sub-system S_4 and S_2. The third and fourth variants of SRG_1, Figure 6.3(b), are identified by the inclusion of fault-tolerant architecture style. This is shown by the addition of sub-system S_5 (*DBMS*) and edges $\boldsymbol{e_{15}}$, $\boldsymbol{e_{32}}$, $\boldsymbol{e_{35}}$ and $\boldsymbol{e_{53}}$. To simplify the calculation for reliability identification set, Table 6.6 shows the assumed reliabilities for elements and their interactions.



6.5 (a) System reliability graph ***SRG₂***



6.5 (b) System reliability graph ***SRG₃***



6.5 (c) System reliability graph ***SRG₄***

Figure 6.5 Variants of system reliability graph ***SRG₁***

The structure and reliability identification set based on aforementioned criteria are shown in Table 6.7.

Elements-subsystems	*Reliability Symbol*	*Reliability*	*elements-interactions*	*Reliability Symbol*	*Reliability*	*elements-interactions*	*Reliability Symbol*	*Reliability*
S_1	R_1	**0.8**	e_{15}	r_{15}	**0.7**	e_{35}	r_{35}	***0.6***
S_2	R_2	**0.7**	e_{14}	r_{14}	**0.8**	e_{42}	r_{42}	***0.7***
S_3	R_3	**0.9**	e_{21}	r_{21}	**0.5**	e_{43}	r_{43}	***0.3***
S_4	R_4	**0.6**	e_{23}	r_{23}	**0.6**	e_{52}	r_{52}	***0.3***
S_5	R_5	**0.8**	e_{34}	r_{34}	**0.5**	e_{53}	r_{53}	***0.5***

Table 6.6 Reliabilities of elements and their interactions

Based on criteria mentioned in chapter 2, the structure identification set is used to identify coefficient of dissimilarity (similarity) on the basis of structure (architecture styles, elements and interactions) while reliability identification set is used to identify coefficient of dissimilarity (similarity) on the basis of structure (architecture styles, elements and interactions) and reliability point of view. Both the sets are calculated using permanent function.

CBSS	***Total Terms***	*Structure Identification Set*	*Reliability Identification Set*
I (***SRG1***)	4	/R_1 R_2 R_3 R_4/r_{34} r_{43} R_1 R_2/ (r_{23} r_{34} r_{42} R_1+ r_{14} r_{42} r_{21} R_3)/ /1/1/2/	/0.8*0.7*0.9*0.6/0.7*0.5*0.8*0.7/(0.6*0.5*0.3*0.8+0.8*0.3*0.5*0.9) /0.3024/0.196/0.18/
II (***SRG2***)	2	/R_1 R_2 R_3 R_4/r_{34} r_{43} R_1 R_2/ /1/1/	/0.8*0.7*0.9*0.6/0.7*0.5*0.8*0.7/ /0.3024/0.196/
III (***SRG3***)	11	/R_1 R_2 R_3 R_4 R_5/(r_{34} r_{43} R_1 R_2 R_5+ r_{35} r_{53} R_1 R_2 R_4)/(r_{23} r_{34} r_{42} R_1 R_5+ r_{23} r_{35} r_{52} R_1 R_4+ r_{14} r_{42} r_{21} R_3 R_5+ r_{15} r_{52} r_{21} R_3 R_4)/((r_{14} r_{42} r_{21} r_{35} r_{53}+ r_{15} r_{52} r_{21} r_{34} r_{43})+(r_{15} r_{53} r_{34} r_{42} r_{21} + r_{14} r_{43} r_{35} r_{52} r_{21}))/ /1/2/4/(2+2)/	/0.8*0.7*0.9*0.6*0.8/(0.5*0.7*0.8*0.7*0.8+0.6*0.5*0.8*0.7*0.6)/(0.6*0.5*0.7*0.8*0.8+0.6*0.6*0.5*0.8*0.6+0.8*0.3*0.5*0.9*0.8+0.7*0.3*0.5*0.9*0.6)/((0.8*0.3*0.5*0.6*0.5+0.7*0.3*0.5*0.5*0.7)+(0.7*0.5*0.5*0.3*0.5+0.8*0.7*0.6*0.3*0.5))/ /0.24192/0.2576/0.3639/0.1493/

Continued…

CBSS	***Total Terms***	*Structure Identification Set*	*Reliability Identification Set*
IV (***SRG$_4$***)	7	/R_1 R_2 R_3 R_4 R_5/(r_{34} r_{43} R_1 R_2 R_5+ r_{35} r_{53} R_1 R_2 R_4)/(r_{23} r_{35} r_{52} R_1 R_4+ r_{15} r_{52} r_{21} R_3 R_4)/((r_{15} r_{52} r_{21} r_{34} r_{43})+(r_{14} r_{43} r_{35} r_{52} r_{21}))/	/0.8*0.7*0.9*0.6*0.8/(0.5*0.7*0.8*0.7* 0.8+0.6*0.5*0.8*0.7*0.6)/(0.6*0.6*0.5 *0.8*0.6+0.7*0.3*0.5*0.9*0.6)/((0.7 *0.3*0.5*0.5*0.7)+(+0.8*0.7*0.6*0.3* 0.5))/
		/1/2/2/(1+1)/	/0.24192/0.2576/ 0.10854/0.08715/

Table 6.7 Structure and reliability identification set

The coefficients of dissimilarity and similarity are computed based on the criteria mentioned in chapter 2 and results are shown in Table 6.8 and Table 6.9 respectively. If SRG_1 is compared with its variants, it is found that SRG_1 and SRG_2 have the same number of nodes, but the new system has only one edge less. This deleted edge causes a large change in the structural complexity and reliability index, which is directly reflected in the similarity/dissimilarity coefficient as calculated. Similarly if SRG_1 is compared with its fault tolerant variants *SRGs* (SRG_3 *and* SRG_4) the overall impact on reliability and structural complexity can be seen. It may be noted that the coefficient of similarity and dissimilarity lies in the range between 0 and 1.

Coefficient of Dissimilarity	***I***	***II***	***III***	***IV***
I		0.80	0.89	0.67
II			0.97	0.90
III				0.78
IV				

Table 6.8 Coefficient of dissimilarity

Coefficient of Similarity	***I***	***II***	***III***	***IV***
I		0.20	0.11	0.33
II			0.03	0.10
III				0.21
IV				

Table 6.9 Coefficient of similarity

If two CBSS architectural designs are isomorphic or completely similar, their coefficient of similarity is 1 and the coefficient of dissimilarity is 0. Likewise, if two *VPF- r* architectures are completely dissimilar, their coefficient of similarity is 0 and the coefficient of dissimilarity is 1. By putting reliability values of sub-systems and their interactions, Table 6.6, in *VPF-r* an index can be calculated that is named as 'Design Reliability Index (I_r)'. Complete computation is shown in Table 6.7. Finally, all the *CBSS* are arranged in I_r sorted order, Table 6.10, and one with high I_r value is chosen as per reliability point of view.

CBSS Architecture	***Design Reliability Index***
III	1.012
IV	0.695
I	0.678
II	0.498

Table 6.10 Design reliability Index in sorted order

Different elements and their associated parameters/characteristics (reliability) can be substituted in order to develop alternate designs for reliability. For all designs the reliability and structure identification set can be calculated using permanent function and aforementioned criteria. The design having high I_r values will be considered as potential/optimum candidate from reliability point of view. Thus the proposed methodology not only helps in modeling and analyzing CBSS but also in developing, evaluating and selecting candidate design for reliability.

6.10 Case Study – Development of Design Reliability Index

A case study presented in Zaraas and Issarny (2000) is under taken to demonstrate the applicability of the developed approach. A software system consists of CORBA and Legacy system (i.e. IBM CICS) to provide services to users. To get services from legacy system, CORBA server has to interoperate with IBM CICS. It is to be noted that the legacy system does not come under the functionality of CORBA, thus direct interaction/linking of CORBA with legacy system is not possible. To achieve this two different designs are identified– first design in a very simple way provides services to users and second design apart from providing services to users also considers fault tolerant behaviour. The reliability index is calculated by utilizing elements values as mentioned in Table 6.11.

Design 1: In this design it is proposed to use CORBA facade component in order to provide interoperation between CORBA server and legacy system. This CORBA facade exports CORBA interface that matches the specification of legacy system. Based upon this design CORBA server can diffuse requests to CORBA facade component which in turn call, at-most, the corresponding functionality provided by the legacy system. Figure 6.6 depicts the design scenario. The system reliability graph of Figure 6.6 is shown in Figure 6.7.

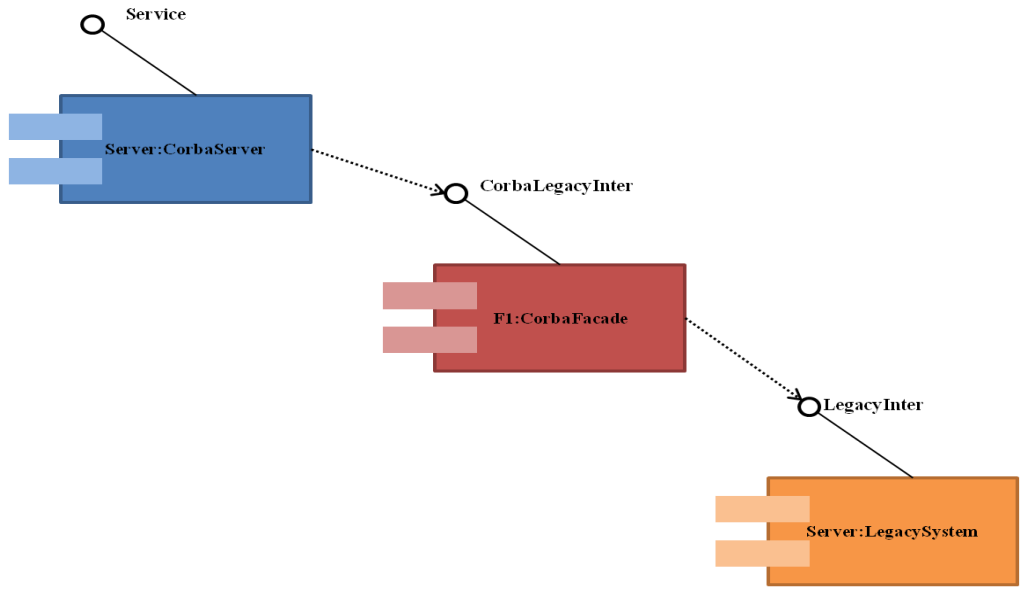


Figure 6.6 UML specification of design 1 concern

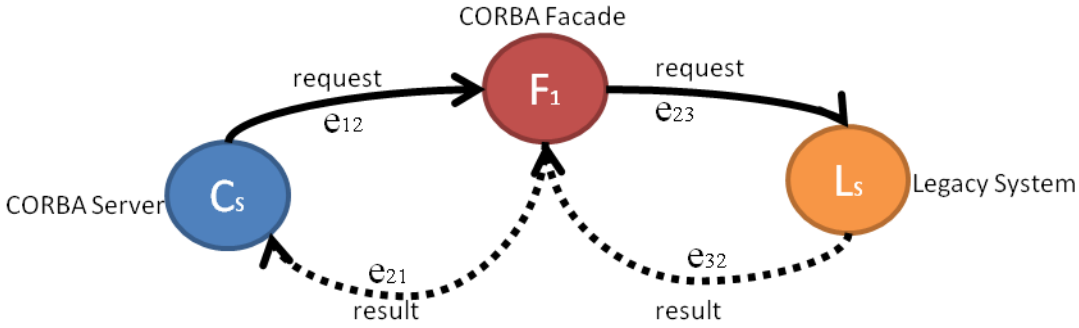


Figure 6.7 Software system design 1 digraph

A permanent matrix representation of Figure 6.7 is mentioned below:

$$VPM\text{-}\ r\ (D_1) = \begin{bmatrix} R_1 & r_{12} & 0 \\ r_{21} & R_2 & r_{23} \\ 0 & r_{32} & R_3 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} \quad (6.11)$$

Below is a unique reliability equation (6.12) for calculating reliability index of design 1 as mentioned in Figure 6.7.

$$Per(D_1) = R_1R_2R_3+R_1r_{23}r_{32}+r_{21}r_{12}R_3 \quad (6.12)$$

Table 6.11 is utilized to calculate the reliability index of design 1:

$I_r(D_1) = 1.732$

Elements-components	*Reliability Symbol*	*Reliability*	*elements-interactions*	*Reliability Symbol*	*Reliability*	*elements-interactions*	*Reliability Symbol*	*Reliability*
CORBA Server	R_1	**0.8**	e_{12}	r_{12}	**0.8**	e_{24}	r_{24}	***0.8***
New CORBA facade F_2	R_2	**0.8**	e_{13}	r_{13}	**0.8**	e_{31}	r_{31}	***0.8***
Old CORBA facade F_1	R_3	**0.8**	e_{21}	r_{21}	**0.8**	e_{34}	r_{34}	***0.8***
Legacy System	R_4	**0.9**	e_{23}	r_{23}	**0.9**	e_{42}	r_{42}	***0.8***
						e_{43}	r_{43}	***0.8***

Table 6.11 Reliabilities of elements and their interactions for design 1 and design 2 concerns

Design 2: A new design is considered to check fault tolerant characteristics of design 1. In this design, only fault tolerant characteristic with regard to CORBA facade component failure is considered. It is to be noted that in design 1 if CORBA facade component fails there is no way to get the requested services. In design 2, replica of CORBA facade component (with little modifications) is shown. Each time a new request comes from the user new CORBA facade component first pings old CORBA facade component to check whether it is available or not. If old CORBA facade component (F_1) is available (by getting the acknowledgement) then by default request is sent to legacy system via old CORBA facade component otherwise new CORBA facade component (F_2) takes the responsibility of sending requests to legacy system and getting results from it. In this way, even if old CORBA facade component fails system is up and user will be served. Figure 6.8 depicts the design scenario. The system reliability graph of Figure 6.8 is shown in Figure 6.9.

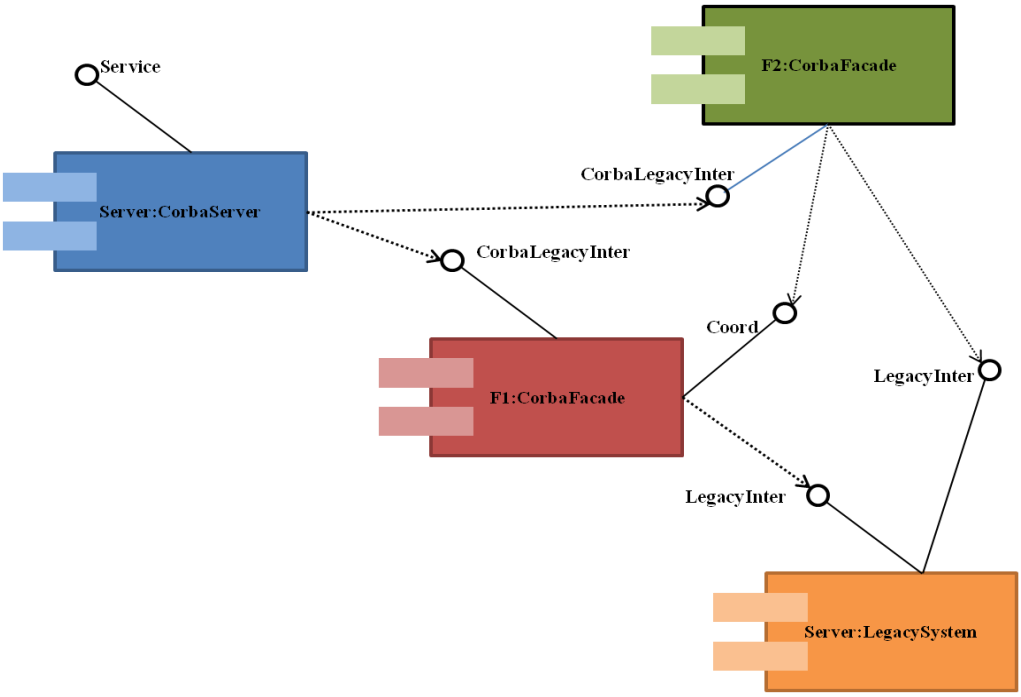


Figure 6.8 UML specification of design 2 concern

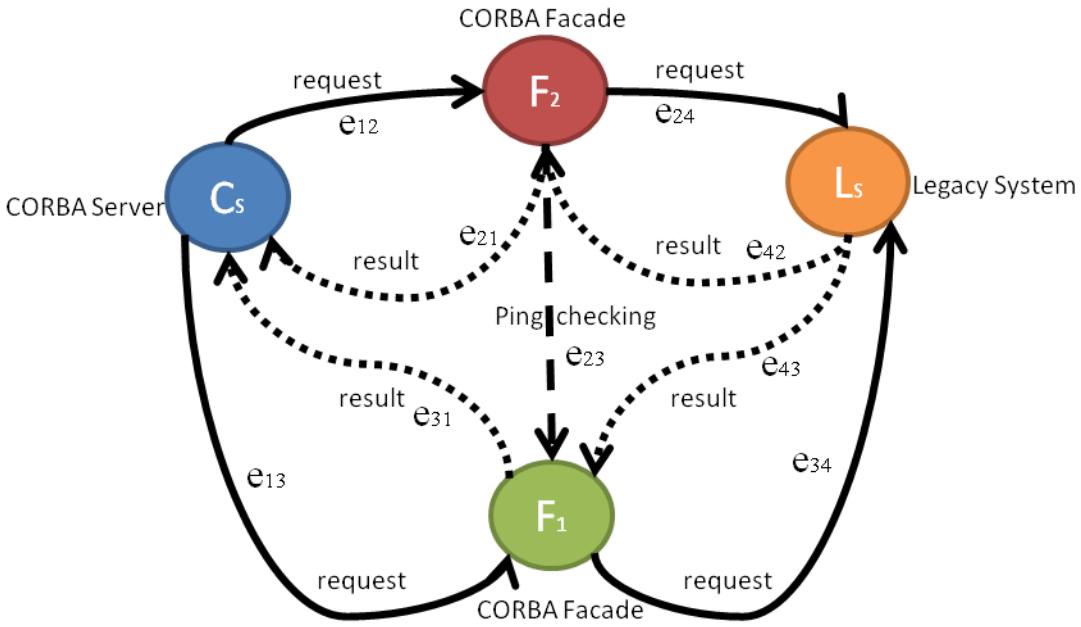


Figure 6.9 Fault tolerant software system design 2 digraph

A permanent matrix representation of Figure 6.9 is mentioned below:

$$VPM\text{-}\ r\ (\mathrm{D}_2) = \begin{bmatrix} R_1 & r_{12} & r_{13} & 0 \\ r_{21} & R_2 & r_{23} & r_{24} \\ r_{31} & 0 & R_3 & r_{34} \\ 0 & r_{42} & r_{43} & R4 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} \quad (6.13)$$

Below is a unique reliability equation (6.12) for calculating reliability index of design 2 as mentioned in Figure 6.9.

$$Per(D_2) = R_1R_2R_3R_4+R_1R_2r_{34}r_{43}+R_1r_{42}r_{23}r_{34}+R_1r_{42}r_{24}R_3+r_{21}r_{12}R_3R_4+r_{21}r_{12}r_{34}r_{43}+r_{21}r_{42}r_{13}r_{34} +r_{31}r_{12}r_{23}R_4+r_{31}r_{12}r_{24}r_{43}r_{31}R_2r_{13}R_4+r_{31}r_{42}r_{13}r_{24} \quad (6.14)$$

Table 6.11 is utilized to calculate the reliability index of design 2:

$I_r(\mathrm{D}_2) = 4.928$

On the lines of the hypothetical example, the structure and the reliability identification set and coefficients of dissimilarity and similarity can be obtained. Finally, both the designs are arranged in I_r sorted order, Table 6.12, and one with high I_r value is chosen as per reliability point of view.

CBSS Design	***Design Reliability Index (I_r)***
Design 2	4.928
Design 1	1.732

Table 6.12 Design reliability index in sorted order (case study)

6.11 Comparative Analysis

In this section the developed approach is compared with the established approaches used for reliability calculation such as Reliability Block Diagram (*RBD*), Failure Mode Effect Analysis (*FMEA*) and Fault Tree (*FT*). Reliability Block Diagram is a graphical representation of the system's components and connectors which can be used to determine the overall system reliability given the reliability of its components. There are several important assumptions that accompany *RBD's* (Bream, 1995):

1. The reliability of each block (component) is known or estimated.

2. The lines have a reliability of one.
3. All lines share the same semantics (type-less).
4. Failures of blocks are statistically dependent
5. xBlocks are bi-modal: they either operate or fail completely (degradation of service is not allowed)
6. All success paths are shown in diagram

A direct mapping of RBD to software architecture for the reliability calculation is not possible. This is so because the particular software architecture style is noncompliance with one or more assumptions of the RBD. The presence of different type of connectors such as pipes and spawns in software architecture style violates the type-less connector assumption 3 of *RBD's*. The spawn gives architecture the ability to be dynamic. *RBD* is basically used to give a point estimate of system's reliability thus is incapable to hold dynamism. It is to be noted that a software architecture style can utilize an implicit global data distributor (e.g., the event manager in the event based style) that violates the assumption 6 of *RBD's*. *RBD* lacks support for concurrency, distribution, dynamism, and an implicit global data distributor. Thus, *RBD* is mainly suitable for styles that comply with features such as – static (no dynamism), shared data variables, single threaded (no concurrency) and single node (no distribution). *RBD* is best suited for main/subroutine type architecture style (Llyod and Lipow, 1962). The developed approach overcomes the drawbacks of *RBD* (assumptions) and can be used to calculate reliability index of any complex system (need not be a sequence of serial or parallel combination). This index will later be used to optimize the design by performing sensitivity analysis on critical parameters or to compare the pool of designs. Since fault tree can be mapped to equivalent *RBDs* the developed approach overcomes the drawbacks of fault tree as well. Fault interdependencies are not considered by fault tree method thus composite analysis is not possible. It may be added that in a fault tree/event tree the structure of the CBSS is not explicitly evident as only the logical relationships among various fault events are depicted rather than the physical interconnections.

The graph theoretic approach is also utilized to overcome the drawbacks of failure modes and effects analysis (Upadhyay et. al., 2010) as mentioned in part I. Interdependencies of failure modes are also considered for calculating the failure index of

component based software system. It is to be noted that high failure index leads to low reliability index of software system which in turn means low reliability.

6.12 Concluding Remarks

In this chapter, the graph theoretic approach is utilized to analyze different failure modes and effects of CBSS which leads to improvement of the CBSS reliability at the design stage. The case study of typical component based web application is used to develop, illustrate, and demonstrate the applicability of the approach. However, this approach is equally applicable for considering failure modes and effects index of other software domain. The numerical value of the permanent function is the CBSS failure modes and effects index. This index is a measure of the consequence of the failure modes and effects. The procedure is extensive and is useful to the designers at the initial stage of design to improve reliability of CBSS. A methodology is also developed to compute the reliability of architectural design when reliabilities of architectural elements are known. The developed reliability index calculation is very critical in those designs which are very complex and cannot be broken into sequence of series or parallel schematic

In the next chapter, software component quality model is developed that overcomes the shortcomings of existing quality models. The technique for concurrent evaluation of quality characteristics is also presented and demonstrated with a real case study.