TESI DI LAUREA MAGISTRALE

Sviluppo e testing di tool per analisi di sicurezza/affidabilità (Fault Tree Analysis)

Relatori:

Prof. Ing. *Paolo* ***Maggiore***

Ing. *Stefano* ***Genta***

Candidato:

Andrea ***Garino***

Davide ***Zollo***

Luglio 2018

A mamma e papà,
semplicemente per tutto.

Indice

Elenco delle figure

Elenco delle tabelle

Ringraziamenti

Sono tante le persone che devono essere ringraziate dal sottoscritto. Grazie al loro aiuto attraverso suggerimenti, critiche ed osservazioni, è stata possibile la stesura del seguente elaborato: a loro va la mia gratitudine, anche se a me spetta la responsabilità per ogni errore contenuto in questa tesi.
Anche se nella maggior parte delle pubblicazioni i ringraziamenti ai propri genitori vengono fatti alla fine, quasi a voler avvalorare la famosa frase pseudo-latina *Dulcis in fundo*, sono dell'idea invece che debbano essere messi al primo posto, così come hanno fatto loro con me in tutti questi anni: per questo motivo desidero ringraziare, dal profondo del cuore, mamma e papà. Siamo partiti per questo viaggio universitario insieme 7 anni fa ed ora, dopo mille difficoltà, delusioni ma anche tante gioie, siamo arrivati alla sua conclusione: dal primo giorno sono sempre stati al mio fianco, dandomi quella spinta in più necessaria per raggiungere tutti gli obiettivi che mi prefissavo nel corso dei vari anni, anche se sembravano impossibili (come dimenticare i 7 esami passati in 16 giorni nella sessione di luglio del terzo anno?) e fornendomi tutto l'aiuto necessario, facendo anche alcuni sacrifici, affinché la mia unica preoccupazione fosse lo studio e nient'altro. Per questo motivo penso che il primo e più grande ringraziamento debba andare a loro.
Il secondo ringraziamento deve essere rivolto al mio relatore di tesi, il Professor Paolo Maggiore il quale per tutti questi anni, a partire dalla realizzazione della tesi triennale, è stato un ottimo insegnante sia dal punto di vista didattico che umano. Lo ringrazio per aver speso ore preziose del suo tempo a seguirmi e per avermi fornito tutto il supporto necessario nella ricerca dell'azienda dove svolgere la tesi. A tal proposito, un sentito ringraziamento deve andare al mio tutor Stefano Genta, che oltre ad avermi permesso attraverso i suoi consigli di realizzare il seguente lavoro di tesi mi ha anche introdotto in un mondo a me fino ad allora sconosciuto ma che ho da subito imparato ad apprezzare come quello ferroviario.
Come poter dimenticare Davide Zollo, il mio braccio destro durante tutto il lungo cammino che ha permesso a questo elaborato di vedere la luce. Lo ringrazio dal punto di vista professionale ma soprattutto umano, perché non è facile trovare persone così semplici e genuine che hanno sempre una parola di conforto anche nei momenti più difficili, che ti spronano a cercare di migliorare sempre. Lo ringrazio perché oltre ad un valido collega, ho trovato il lui un amico sincero.
Un grosso ringraziamento va sicuramente a Paolo Bizzarri, grazie al quale è stato possibile sviluppare i numerosi script in Matlab che costituiscono le colonne portanti del programma sviluppato in questo lavoro di tesi. Lo ringrazio per le numerose ore che ha concesso a me e a Davide nel suo ufficio per risolvere tutti i dubbi ed i problemi che di volta in volta riscontravamo. Infine, un ringraziamento a Marco Albano, che ci ha fornito tutti i mezzi necessari per poter affrontare al meglio questo lavoro di tesi ed ai miei colleghi, Laura Andrea e Fabiano, che tanto mi hanno insegnato in questi mesi di lavoro in azienda.

Introduzione

Lo studio della sicurezza e dell'affidabilità dei sistemi riveste al giorno d'oggi un aspetto fondamentale in tutti i campi dell'ingegneria, da quella civile a quella ferroviaria, fino ad arrivare a quella spaziale. Le analisi legate ad affidabilità, disponibilità, manutenibilità e sicurezza rientrano nel grande campo dell'analisi RAMS (Reliability, Availability, Maintainability and Safety) e hanno il compito di definire con la maggior accuratezza possibile gli eventuali scenari di guasto dei diversi componenti che potrebbero portare alla perdita delle funzionalità di un intero sistema, con rischi legati anche alla possibile perdita di vite umane. Per effettuare questo tipo di analisi sono disponibili diversi programmi commerciali, come ad esempio Reliability Workbench dell'azienda Isograph, che permettono la realizzazione di studi di sicurezza ed affidabilità come l'analisi degli alberi dei guasti (Fault Tree Analysis); le licenze di questi tools, sebbene molto precisi ed affidabili, hanno di norma costi elevati (diverse migliaia di Euro), pertanto le aziende che intendono avvalersene devono tenere in considerazione l'impatto economico che queste possono avere.
Il presente lavoro di tesi nasce con l'idea di dimostrare la possibilità di realizzare *ex novo*, un programma in Matlab e Simulink, software forniti dalla società MathWorks, in grado di emulare le analisi compiute da un software commerciale conosciuto a livello mondiale come FaultTree+. L'obiettivo che ci si è prefissati è stato la realizzazione di un programma dall'aspetto grafico simile a quello del tool commerciale che sia in grado di effettuare, partendo dai dati di input inseriti nei Basic Events, un'analisi sia numerica che simbolica dei diversi alberi di guasto, e che una volta comparata con la stessa analisi effettuata in FaultTree+, fornisca un errore relativo massimo tra i risultati ottenuti non superiore come ordine di grandezza a 10^{-6}.
Per realizzare il programma si è deciso di sfruttare Simulink per quanto riguarda la parte grafica, creando una libreria apposita contenente i blocchi necessari a realizzare un'analisi degli alberi dei guasti, mentre Matlab viene utilizzato per creare tutte le funzioni e gli script contenenti le formule necessarie alla corretta realizzazione dell'analisi dal punto di vista numerico e simbolico.
Il lavoro svolto è documentato in questa tesi ed è strutturato in 6 capitoli, suddivisi nel seguente modo:

- Il capitolo 1 è dedicato all'introduzione al mondo dell'analisi RAMS; in esso viene fornita una breve ma efficace spiegazione di tutti gli aspetti che questo grande insieme contiene
- Il capitolo 2 analizza nel dettaglio la Fault Tree Analysis. Viene fornita una descrizione di alto livello rispetto alla teoria delle probabilità, sulla quale si basa tutto lo studio degli alberi dei guasti. Successivamente viene posta l'attenzione sugli alberi stessi, in particolare sulle regole da seguire per la loro costruzione, i blocchi e le porte logiche da cui sono costituiti, e i metodi e modelli che sono alla base dell'analisi di grandezze come l'indisponibilità Q(t), l'inaffidabilità F(t) e la frequenza di guasto $\omega(t)$
- Il capitolo 3 introduce il programma realizzato attraverso Matlab e Simulink: in esso viene descritta la libreria realizzata e successivamente vengono dettagliati, uno per uno, tutti i blocchi che la compongono, descrivendone le principali funzioni

- Il capitolo 4 si focalizza su un esempio pratico di analisi degli alberi dei guasti, legato al mondo ferroviario. Il capitolo è di fondamentale importanza per tutto il lavoro di tesi, in quanto l'analisi viene effettuata in prima battuta mediante l'utilizzo di FaultTree+, quindi lo stesso esempio viene riprodotto con il programma sviluppato in Simulink nel corso della tesi. Il capitolo si chiude con un primo confronto tra i risultati ottenuti dalle due analisi

- Il capitolo 5 è dedicato alla fase di test del programma realizzato, con l'obiettivo di valutarne l'aderenza alle aspettative. Test unitari e di sistema vengono realizzati e schedati in apposite Test Cases. Viene quindi realizzato il confronto dei risultati di un albero costituito da ben 19 Basic Events analizzato prima in FaultTree+ e successivamente con lo strumento di analisi oggetto del presente lavoro di tesi

- Il sesto e ultimo capitolo traccia un quadro finale dell'attivita' svolta nel corso della tesi, proponendo inoltre alcune possibili espansioni del programma realizzato, con l'intento di ampliare ulteriormente le capacita' di analisi dello stesso.

Capitolo 1

L'analisi RAMS

1.1 Introduzione

La nascita dell'analisi RAMS (Reliability, Availability, Maintainability and Safety) può essere ricondotta all'inizio degli Anni '30, quando una sempre maggiore attenzione veniva posta sulle tematiche di affidabilità e sicurezza dei sistemi nell'industria aerospaziale. Attraverso l'applicazione di tecniche statistiche nell'analisi di guasto dei sistemi, questi due campi diventarono a partire dagli anni '50 discipline ingegneristiche fondamentali nei sistemi aerospaziali, per poi estendersi a tutti gli altri campi [1]: ogni giorno infatti, senza accorgersene, si applicano delle semplici regole che derivano da considerazioni legate ai concetti di affidabilità e sicurezza. Le case, i mezzi con i quali le persone si muovono, i luoghi che vengono frequentati per lavorare, studiare, divertirsi, devono avere tutti un certo livello di sicurezza ed affidabilità. Nonostante questi concetti non siano strani o difficili, di solito vengono sottovalutati o dati per scontati negli studi di ingegneria; in realtà una gran quantità di prodotti è affetta da problemi relativi all'affidabilità ed alla sicurezza, soprattutto per quei sistemi considerati complessi, i quali devono svolgere missioni altrettanto complesse. Il fine ultimo di ogni progetto (e quindi dell'attività ingegneristica) è la realizzazione e la messa in esercizio di un sistema con un livello di sicurezza accettabile ed al tempo stesso caratterizzato da un'efficacia adeguata.
Lo sviluppo dell'analisi RAMS venne completato con l'introduzione dei concetti legati a disponibilità e manutenibilità per la sicurezza e l'affidabilità, che permisero di definire un processo standard di valutazione e controllo di tutti i possibili pericoli a cui un sistema va incontro.
Quando si parla di sistema, si fa riferimento ad un insieme di elementi (hardware, software, servizi, personale) strutturati in modo da svolgere una funzione (ossia la missione) che soddisfi una certa necessità identificata (dal cliente). Ogni elemento avrà caratteristiche proprie peculiari legate ad affidabilità, disponibilità, manutenibilità e sicurezza comuni in ambiti molto diversi tra di loro, come quello aeronautico, energetico e ferroviario. Ogni volta che si vuole mettere in esercizio un sistema, questi componenti devono essere analizzati attentamente, per evitare guasti che potrebbero avere conseguenze catastrofiche per l'uomo e l'ambiente.
L'analisi RAMS si concentra sullo studio dell'affidabilità e della sicurezza di sistemi soggetti a guasto (failure); è un'analisi mirata a definire le caratteristiche proprie di un qualunque prodotto. Il suo studio può essere condotto attraverso l'uso di due diversi approcci:

- Qualitativo: legato ad un'analisi statistica
- Quantitativo: caratterizzato da valori esatti provenienti, ad esempio, da banche dati

Di questi due approcci si parlerà più diffusamente nelle sezioni 1.2.3 e 1.2.4.

1.2 Reliability

Si definisce affidabilità (reliability) la probabilità che un oggetto possa eseguire una funzione richiesta in condizioni stabilite per un dato intervallo di tempo (t_1, t_2) [3]. Una seconda definizione può essere la seguente: l'affidabilità è la probabilità di non avere guasti, di un certo tipo, per una certa missione. I suoi obiettivi sono:

- Definire gli obiettivi di affidabilità allo scopo di soddisfare le prestazioni richieste dell'applicazione specifica
- Definire i modi di guasto del sistema ed il Tempo Medio Tra Guasti (MTBF-Mean Time Between Failure)

Si possono avere diversi tipi di affidabilità a seconda del tipo di guasti da prendere in considerazione: quella Basica (o Logistica) e quella di Missione.

1.2.1 Affidabilità Basica o Logistica

L'affidabilità basica o logistica è la probabilità di non avere alcun guasto, indipendentemente dalla sua natura e dalle conseguenze che questo comporta. L'obiettivo è quello di realizzare un determinato sistema in grado di compiere la propria missione al minor Life Cycle Cost (LCC) possibile (del Life Cycle Cost si parlerà più ampiamente nella sezione 1.4.2); è necessario quindi considerare le caratteristiche logistiche del sistema al pari di quelle prestazionali e già nelle primissime fasi del progetto. E' un parametro legato alla qualità globale del prodotto e al numero di interventi di riparazione che esso richiederà durante la sua vita in esercizio. Quello della Logistica è un concetto recente , nato in seguito alla crisi economica che ha comportato la drastica riduzione di budget e risorse, che ha portato alla necessità di migliorare la produttività globale del prodotto; si è cominciato a considerare il costo globale del ciclo di vita del prodotto che, come si può notare dal grafico seguente, è fortemente condizionato già dalle scelte fatte nelle fasi iniziali del progetto.

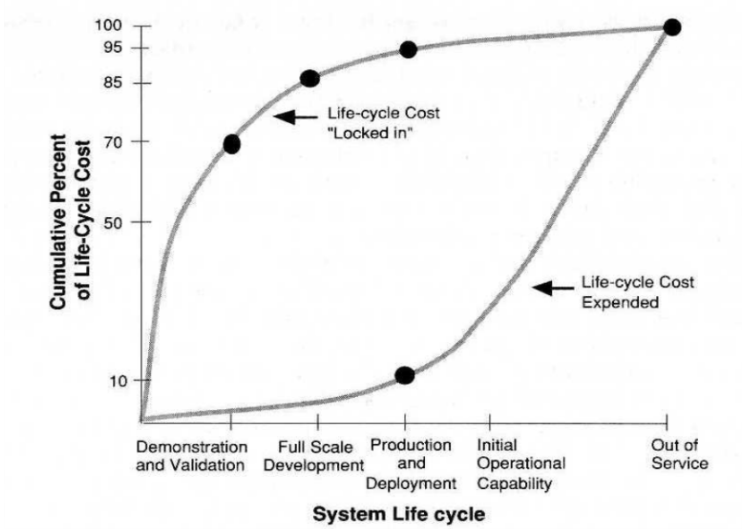

Figura 1.1: Life Cycle Cost

La Logistica si occupa di tutti gli aspetti e persone che ruotano attorno ad un progetto, in particolare della:

- Verifica e supporto di equipaggiamenti, componenti e dispositivi

- Gestione delle strutture
- Reperimento e gestione di informazioni e software
- Gestione dei servizi
- Addestramento del personale

Da questo parametro dipendono la manutenibilità e la disponibilità. Infatti gli aspetti logistici vanno a toccare tutta una serie di concetti relativi al ciclo di vita del sistema, come ad esempio:

- La definizione di concetto di manutenzione del sistema
- La definizione dei requisiti di manutenzione
- L'allocazione dei requisiti quantitativi e qualitativi per gli elementi del supporto
- L'analisi preliminare del supporto logistico
- L'aggiornamento del progetto supporto logistico
- L'identificazione dei requisiti per le risorse logistiche
- L'acquisizione dei materiali di supporto
- La valutazione del sistema in operatività

1.2.2 Affidabilità di Missione

Viene definita affidabilità di Missione la probabilità di non avere guasti che possano impedire il completamento della missione del sistema (guasti con conseguenze maggiori). Con missione si intende un determinato lasso di tempo, nel quale il sistema lavora in condizioni operative. Conoscere le potenziali cause dei guasti è condizione necessaria per poter tentare di prevenirli o ridurli; la loro conoscenza tuttavia è incompleta, per questo deve essere contemplato il concetto di incertezza. Le principali cause di guasto possono essere così classificate:

- Progetto intrinsecamente inadeguato
- Componente sovrasollecitato durante il funzionamento
- Degrado delle prestazioni durante la vita del componente
- Eventi inattesi
- Errori umani
- Altro

I guasti possono essere classificati in tre diverse categorie, in base alle conseguenze legate ad un loro accadimento:

- Guasti pericolosi: possono avere conseguenze catastrofiche, ossia danni alle persone, alle cose ed al sistema stesso
- Guasti con conseguenze maggiori: pregiudicano la funzionalità del sistema, ma senza mettere a repentaglio la sicurezza delle persone e del sistema stesso

- Guasti con conseguenze minori: la loro conseguenza è quella di rendere necessario un intervento di manutenzione per ripararli

In quest'ottica RAMS la sicurezza è un sottoinsieme dell'affidabilità.

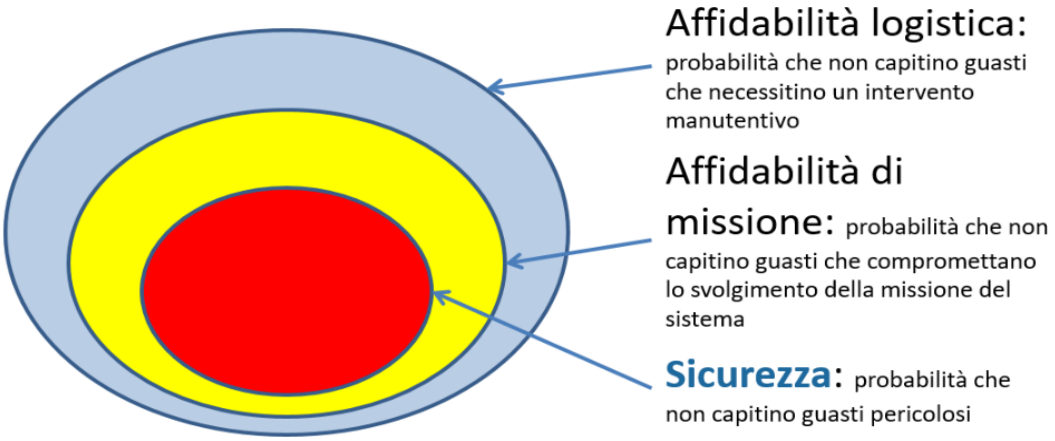

Figura 1.2: Relazione tra affidabilità e sicurezza

1.2.3 Parametri di affidabilità

Facendo riferimento alla normativa IEC 50126 [3], si possono individuare i seguenti parametri legati all'affidabilità:

Parametro	**Simbolo**	**Dimensione**
Tasso di guasto	Z(t), λ	Guasti/tempo, distanza, ciclo
Tempo medio di disponibilità	MUT	Tempo, distanza, ciclo
Tempo medio al guasto (oggetti riparabili)	MTTF	Tempo, distanza, ciclo
Tempo medio tra guasti	MTBF	Tempo, distanza, ciclo
Probabilità di guasto	F(t)	Adimensionale
Affidabilità (Probabilità di successo)	R(t)	Adimensionale

Tabella 1.1: Parametri di affidabilità (IEC 50126)

I concetti di affidabilità e guasto sono strettamente collegati tra di loro; infatti, definita la densità di guasto (fault density) f(t) come il numero di guasti al tempo t, si può calcolare la probabilità di guasto (fault probability) F(t)

$$F(t) = \int_0^T f(t)dt \tag{1.1}$$

definita comunemente unreliability (inaffidabilità). Essendo un numero compreso tra 0 ed 1, attraverso l'inaffidabilità è possibile ottenere il valore di affidabilità R(t) attraverso la formula

$$R(t) = 1 - F(t) = 1 - \int_0^T f(t)dt \tag{1.2}$$

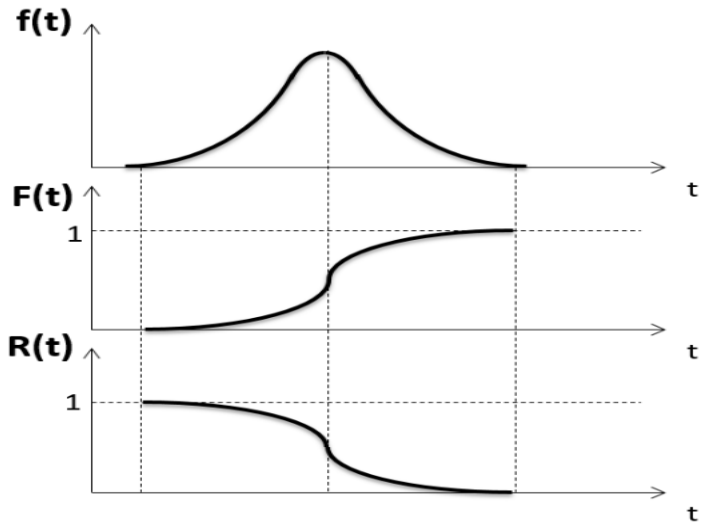

Figura 1.3: Densità di rischio, probabilità di rischio ed affidabilità

Dal seguente grafico è possibile notare come in un generico componente l'inaffidabilità cresca all'aumentare del tempo di missione, con conseguente diminuzione dell'affidabilità.

Tasso di guasto (failure rate)

Definito come il numero di guasti nell'unità di tempo [guasti/ora] ad un certo tempo (probabilità di guasto istantanea) il tasso di guasto (failure rate) è uno dei parametri più rappresentativi per l'analisi di affidabilità di un sistema/componente. E' inversamente proporzionale all'affidabilità e può essere definito come

$$FR = -\frac{1}{R}\frac{dR}{dt} \tag{1.3}$$

I guasti vengono definiti casuali quando la loro probabilità di manifestarsi è indipendente dal tempo di funzionamento accumulato. In questo caso la densità di guasto f(t) ha una distribuzione esponenziale, come si può notare dalla seguente formula

$$f(t) = \lambda e^{-\lambda t} \tag{1.4}$$

dove λ può essere o no costante, in base al modello matematico con il quale si decide di analizzare il tasso di guasto. Tramite una serie di passaggi, si può dimostrare il valore del tasso di guasto in regime di guasti casuali:

$$F(t) = \int_0^t f(t)dt = \lambda \int_0^t e^{-\lambda t}dt = 1 - e^{-\lambda t} \tag{1.5}$$

da cui, sapendo che

$$R(t) = 1 - F(t) = e^{-\lambda t} \tag{1.6}$$

si ottiene che il tasso di guasto è pari a

$$FR = -\frac{1}{R}\frac{dR}{dt} = -\frac{1}{e^{-\lambda t}}\frac{d(e^{-\lambda t})}{dt} = \lambda \tag{1.7}$$

Risulta quindi che il tasso di guasto è pari a λ che, come detto in precedenza, può essere o meno costante; attraverso questo parametro è possibile definire qualitativamente la vita di un componente, utilizzando il cosiddetto "diagramma a vasca da bagno", che tiene conto di

mortalità infantile, regime di guasti casuali durante la vita utile e di usura a fine vita del sistema.

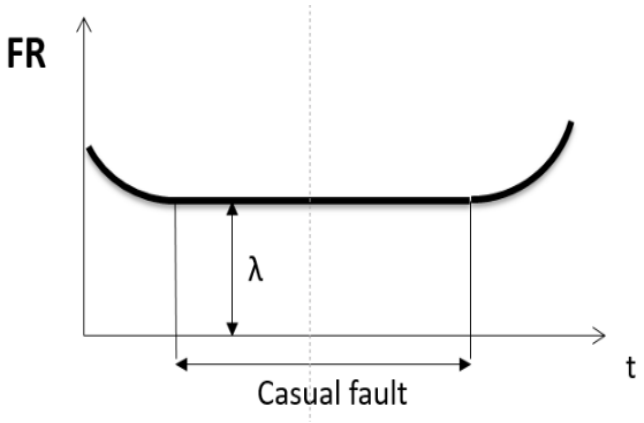

Figura 1.4: Distribuzione di guasto

Come si può notare dallo schema, la probabilità che un componente vada incontro a guasti è molto più alta nelle fasi iniziali e finali della sua vita, mentre durante la vita operativa il tasso di guasto si mantiene pressoché costante.

1.2.4 Analisi di Criticità (risk analysis)

Il concetto di rischio include sia le conseguenze indesiderate, come ad esempio il numero di persone che vengono ferite in un incidente, che la probabilità di accadimento dell'incidente. [2] L'analisi di criticità si basa sull'analisi del rischio. I modi di guasto critici sono quelli caratterizzati da un più alto valore di rischio. La criticità di un guasto è caratterizzata principalmente da due parametri: il primo legato alle conseguenze che questo può generare, il secondo alla frequenza con cui un guasto si può presentare. In particolare, se assegnamo un numero alle conseguenze (C) ed alla frequenza (F), la criticità (R) sarà pari al prodotto delle due:

$$R = F * C \tag{1.8}$$

In base al risultato ottenuto, si può classificare il rischio in diverse classi, come mostrato nella seguente tabella tratta dalla normativa EN 61508-5 [4]:

Frequenza	**Conseguenza**			
	Catastrofica	**Critica**	**Marginale**	**Trascurabile**
Frequente	I	I	I	II
Probabile	I	II	I	III
Occasionale	I	II	III	III
Remota	II	III	III	IV
Improbabile	III	III	IV	IV
Inverosimile	IV	IV	IV	IV

Tabella 1.2: Classificazione del rischio (EN 61508-5: 2010)

Frequenza e conseguenza sono stimate mediante indici qualitativi. Nell'analisi di criticità, effettuata mediante matrici di rischio, vengono analizzati i rischi rispetto alla sicurezza,

all'ambiente, alla produttività e ai danni infrastrutturali o costi di manutenzione. Prendendo come riferimento la normativa EN 61508-5 [4], è possibile definire le diverse classi di rischio, dalla I alla IV, nel seguente modo:

Classe di rischio	Interpretazione
Classe I	Rischio non tollerabile
Classe II	Rischio indesiderato e tollerabile solamente se la riduzione del rischio è impraticabile o se i costi di mitigazione superano i miglioramenti ottenibili
Classe III	Rischio tollerabile se il costo di mitigazione supera i miglioramenti ottenibili
Classe IV	Rischio accettabile

Tabella 1.3: Definizione delle classi di rischio (EN 61508-5: 2010)

La necessità di riduzione del rischio è legata al fatto di dover renderlo tollerabile per una specifica situazione. Lo scopo della determinazione di tollerabilità del rischio per uno specifico evento pericoloso è quello di definire quando questo possa essere ritenuto tollerabile rispetto sia alla frequenza che alla conseguenza del suo accadimento.
Come risultato dell'analisi di criticità ci si aspetta di ottenere l'elenco dei modi di guasto che possono risultare critici per un componente; questi dovranno tener conto delle possibili conseguenze che possono portare a livello di sicurezza, ambiente, produttività e l'analisi dovrà contenere indicazioni sulle strategie per ridurre la criticità C.

1.2.5 L'approccio qualitativo

L'approccio qualitativo all'analisi RAMS si basa sui seguenti passi:

- Definizione del sistema oggetto di analisi, identificandone caratterizzazione, struttura e comportamento
- Individuazione delle funzioni di sottosistemi e componenti, con particolare attenzione sulle fasi operative
- Individuazione dei Modi di Guasto (failure modes) e degli effetti (danni)
- Valutazione del rischio
- Individuazione componenti/modi di guasto critici

Il modo di guasto di un componente è la perdita di una delle funzioni assolte da questo. I modi di guasto dipendono dalle fasi operative del sistema, che descrivono le diverse configurazioni assunte dallo stesso durante la sua vita utile (avviamento, regime, spegnimento, manutenzione...). L'analisi qualitativa richiede l'identificazione di tutti i modi di guasto possibili del sistema e viene effettuata identificando quelli di ciascun componente nelle diverse fasi operative.
Gli obiettivi dell'analisi dei modi di guasto e degli effetti possono essere riassunti nei seguenti punti:

- Identificare per ogni modo di guasto gli effetti sul sistema
- Valutare qualitativamente gli effetti in termini di sicurezza, impatto ambientale, produttività, costi di manutenzione
- Evidenziare le misure di prevenzione e mitigazione già presenti
- Evidenziare le diagnostiche disponibili per rilevare il guasto

- Segnalare le misure nuove da implementare

Storicamente esistono due approcci principali per realizzare un'analisi qualitativa:

- FMECA (Failure Mode, Effects and Criticality Analysis) orientata ai componenti
- HAZOP (HAZard and OPerability Studies) orientata ai processi

Failure Mode, Effects and Criticality Analysis (FMECA)

I concetti di affidabilità e sicurezza ingegneristica furono introdotti per valutare i guasti nei sistemi e gli errori umani. La prima tecnica per studiarli fu la Failure Mode and Effect Analysis (FMEA) introdotta negli anni '40; successivamente la Boeing aggiornò la FMEA, trasformandola in quella che oggi è conosciuta come Failure Mode and Effect (and Criticality) Analysis (FMECA) [1]. La FMECA rientra nell'approccio qualitativo dell'analisi RAMS ed è la tecnica più utilizzata per progettare l'affidabilità. Il suo obiettivo è quello di studiare gli effetti dei guasti dei componenti sulla funzionalità del sistema e classificare ogni potenziale guasto in base alla sua criticità. Deve essere condotta in modo formale e oggettivo, seguendo una procedura ben definita. L'analisi esamina strutture, sistemi e/o componenti per analizzare e valutare modi operativi in funzionamento nominale, fuori condizioni nominali e transitivi, modi di guasto e le rispettive conseguenze. Può essere applicata a qualunque livello. La FMECA classifica i modi di guasto di un dato sistema, sottosistema o processo in base agli effetti sugli altri sottosistemi/componenti e infine sul sistema globale. E' particolarmente adatta allo studio di sistemi analogici, mentre risulta più difficile applicarla a sistemi digitali (causa circuiti integrati e complicazione sistema). La FMECA identifica le rotture, i punti deboli e i rischi che possono portare a oltrepassare i limiti di progetto, facilitando lo sviluppo delle azioni correttive da intraprendere per evitare, eliminare o ridurre le condizioni che possono pregiudicare l'integrità del sistema o la vita delle persone. Di seguito vengono riportati i passi necessari a condurre un'analisi FMECA:

- Identificazione di ogni componente del sistema
- Determinazione di tutti i possibili modi di guasto per il singolo componente
- Determinazione di tutte le possibili cause di ogni singolo modo di guasto
- Determinazione dell'effetto più grave di ogni modo di guasto sul componente stesso e la conseguenza sul sistema globale, in base alla fase di missione
- Valutazione della severità/criticità del singolo modo di guasto

La FMECA è utilizzata per identificare aree critiche e per aiutare lo sviluppo delle azioni correttive da intraprendere per evitare, eliminare o ridurre le condizioni che possono pregiudicare l'integrità del sistema o la vita delle persone. Identifica inoltre i componenti che presentano la maggior influenza sul rischio globale del programma.

HAZOP

L'HAZardous OPerations Analysis (HAZOP) è un metodo sistematico per stimolare l'immaginazione a identificare e valutare i modi in cui un elemento del processo può funzionare male o essere utilizzato impropriamente. Un team interdisciplinare di esperti esamina sistematicamente il processo al fine di individuare come possono verificarsi delle deviazioni rispetto alle condizioni nominali di progetto. Il team valuta poi la gravità di queste deviazioni.

1.2.6 L'approccio quantitativo

Cenni della teoria delle probabilità

A causa del fatto che l'affidabilità e la safety sono legate al concetto di incertezza, nel senso che i valori calcolati relativamente a queste caratteristiche non possono essere certi, il loro studio da un punto di vista quantitativo si avvale della teoria delle probabilità e della statistica. La teoria delle probabilità è la scienza matematica che studia le leggi che sono alla base di fenomeni aleatori, ossia quei fenomeni che, se si ripetono molte volte, forniscono risultati diversi fra loro in modo casuale (come succede ad esempio nel gioco d'azzardo). La probabilità (p) del verificarsi di un dato evento è espressa dal rapporto tra il numero di casi favorevoli (a) all'evento ed il numero di casi possibili (c)

$$\text{p=a/c}$$

La probabilità statistica sarà sempre affetta da un certo errore, in quanto risulta impossibile effettuare infinite prove: tanto più il numero di queste sarà grande, maggiore sarà la fiducia nella stima effettuata.
Una definizione assiomatica della probabilità, fornita dallo studioso russo Andrej Nikolaevic Kolmogorov, è la seguente: la probabilità di un evento è una quantità numerica che soddisfa gli assiomi del calcolo delle probabilità.
Nei capitoli successivi verrà approfondito il discorso legato alla teoria della probabilità applicato ai casi di studio che verranno presi in considerazione.

L'approccio quantitativo

L'approccio quantitativo è legato alla caratterizzazione dei componenti, che può essere reperita attraverso:

- Banche dati di affidabilità commerciali
- Standard per raccolta dati
- Enti

In particolare la raccolta dei dati sul campo permette la creazione di database contenenti informazioni utili per evitare in futuro l'accadimento di un guasto ad un determinato componente. L'approccio quantitativo ruota attorno al calcolo di determinati parametri, utili a definire la vita utile di un componente; in particolare, i principali parametri utilizzati per un'analisi di rischio sono la indisponibilità Q

$$Q = \frac{\lambda}{\lambda + \mu}(1 - e^{-(\lambda+\mu)t}) \tag{1.9}$$

e l'inaffidabilità F(t)

$$F(t) = 1 - e^{-\lambda t} \tag{1.10}$$

dove

- λ = failure rate (tasso di guasto)
- μ = repair rate (tasso di riparazione)

Attraverso questi due parametri, considerando il failure rate costante, è possibile definire tre famiglie di componenti:

- Componenti riparabili, che possono essere riparati senza compromettere la missione e che evidenziano immediatamente il guasto
- Componenti non riparabili, che non possono essere riparati senza compromettere la missione
- Componenti riparabili testati, che possono essere riparati senza compromettere la missione ma che non evidenziano il loro guasto

Analisi quantitative

Per uno studio RAMS di tipo quantitativo, sono due le tecniche principalmente utilizzate: la Fault Tree Analysis (FTA) e il Reliability Block Diagram (RBD).

Fault Tree Analysis (FTA)

La Fault Tree Analysis è un'analisi top-down adatta all'analisi di eventi complessi derivanti dall'accadimento di più eventi singoli (valutazione di efficacia delle ridondanze); per realizzarla vengono utilizzati dei tool specifici, come ad esempio FaultTree++. L'approccio top-down prevede che partendo da un evento di guasto di un componente (TOP event), si scenda nell'analisi fino a trovare l'evento o gli eventi che hanno scatenato il guasto (Basic event). L'FTA è in grado di identificare le combinazioni di guasti/rotture che possono portare ad uno specifico guasto del sistema e quindi può evidenziare debolezze di progetto che tecniche più semplici come la FMECA possono non cogliere, soprattutto in sistemi con molte ridondanze. La Fault Tree Analysis è impiegata per esaminare problemi dalle conseguenze molto gravi, che possono risultare da guasti la cui eliminazione/miglioramento può comportare un grande impegno di risorse finanziarie.
I passi per svolgere una corretta FTA possono essere riassunti nei seguenti punti:

- Viene sviluppato il Fault Tree (albero dei difetti) ragionando deduttivamente a ritroso da un evento (top event) attraverso gli eventi intermedi, fino alle radici che hanno causato il top event
- Il modello viene rivisto con il fine di migliorarlo con azioni correttive che eliminino i possibili errori e/o debolezze
- Il Fault Tree comincia con un guasto di sistema e giunge a determinare quali guasti di quali componenti lo hanno provocato. Può considerare anche la componente umana (human factors) quale causa di errori.
- Procedure software vengono utilizzate per identificare le possibili combinazioni di guasti che hanno causato il top event

Per contro, la Fault Tree Analysis è soggettiva e non sistematica: non esiste un metodo standardizzato per realizzarla e persone diverse possono arrivare a realizzare alberi differenti dello stesso evento; inoltre è un'analisi complessa, in quanto richiede un elevato grado di conoscenza dei sistemi/processi da analizzare e della matematica da applicare (logica booleana).

Reliability Block Diagram (RBD)

L'analisi RBD è concettualmente simile alla FTA, adatta all'analisi di eventi complessi derivanti dall'accadimento di più eventi singoli (valutazione di efficacia delle ridondanze). Rispetto a questa, evidenzia meglio i "single point failure" e le ridondanze. Gli schemi a blocchi illustrano in maniera chiara e oggettiva come i componenti di un sistema si interfacciano dal

punto di vista logico, ai fini dello studio dei vari tipi di affidabilità. Non corrispondono quindi a schemi fisici, tanto che, per lo stesso sistema, possono cambiare a seconda del tipo di affidabilità. Gli RBD si basano sull'analogia con i circuiti elettrici:

- Componente guasto = resistenza interrotta
- Segnale che arriva al termine dello schema = successo

Gli RBD possono essere rappresentati quindi come blocchi in serie o in parallelo: nel primo caso perché il sistema abbia successo occorre che tutti i componenti funzionino, mentre nel secondo occorre che almeno uno dei componenti funzioni.
Per contro, la rappresentazione grafica attraverso RBD è adatta per sistemi semplici: in caso di sistemi complessi, diventa di difficile lettura.

1.3 Availability

Si definisce disponibilità (availability) la capacità di essere in uno stato per eseguire una funzione richiesta sotto date condizioni in un dato istante di tempo o in un dato intervallo di tempo assumendo che siano fornite le risorse esterne richieste [3]. La disponibilità è espressa come la probabilità che il sistema entri in modalità operativa quando viene chiamato in causa in un dato momento casuale. Per i sistemi che lavorano in modo continuo, è la probabilità che il sistema stia operando in un determinato tempo casuale.
I modi per determinare la disponibilità di un sistema sono principalmente tre:

Disponibilità intrinseca

E' la probabilità che uno specifico sistema opererà correttamente quando chiamato in causa in un qualsiasi momento sotto specifiche condizioni operative e in un ambiente ideale, in cui il personale di manutenzione è prontamente disponibile. Non viene considerato nessun ritardo logistico o amministrativo, inoltre vengono esclusi compiti di manutenzione preventiva e programmata. E' dato dalla formula

$$A_I = \frac{MTBF}{MTBF + M_{CT}} \tag{1.11}$$

dove:

- MTBF=Mean Time Between Failure
- M_{CT}= Mean Corrective Maintenance Time

Disponibilità raggiunta

E' la probabilità che uno specifico sistema opererà correttamente quando chiamato in causa in una qualsiasi momento sotto specifiche condizioni operative, in un ambiente ideale, in cui il personale di manutenzione è prontamente disponibile. Non viene considerato nessun ritardo logistico o amministrativo, ma vengono inclusi compiti di manutenzione preventiva. E' dato dalla formula

$$A_A = \frac{MTBM}{MTBM + M} \tag{1.12}$$

dove:

- MTBM = Mean Time Between Maintenance
- M= Mean active Maintenance Time

Disponibilità operativa

E' la probabilità che uno specifico sistema opererà correttamente quando chiamato in causa in un qualsiasi momento, sotto specifiche condizioni operative, in un ambiente reale. Questa misura è la più vicina alla realtà, in quanto vengono considerati non solo i compiti di manutenzione preventiva e correttiva, ma anche i ritardi logistici ed amministrativi. E' data dalla formula

$$A_O = \frac{MTBM}{MTBM + MDT} \tag{1.13}$$

dove:

- MTBM=Mean Time Between Maintenance
- MDT=Mean Down Time

La disponibilità di sistema, A, può essere specificata in parti attribuite a:

- Indisponibilità programmata (Manutenzione): 1-A_M
- Indisponibilità non programmata (Riparazione): 1-$A - R$

A=1-[(1-A_M)+(1-A_R)]=MUT/(MUT+MDT)

Seguendo la normativa IEC 50126 [3], i parametri legati alla disponibilità sono:

Parametro	**Simbolo**	**Dimensione**
Disponibilità intrinseca	A_I	Adimensionale
Disponibilità raggiunta	A_A	Adimensionale
Disponibilità operativa	A_O	Adimensionale
Disponibilità di flotta	FA	Adimensionale
Rispetto dell'orario	SA	Adimensionale

Tabella 1.4: Parametri di disponibilità (IEC 50126)

1.4 Maintainability

Si definisce manutenibilità (maintainability) la probabilità che un'azione di manutenzione attiva, per un dato oggetto, utilizzato in condizioni assegnate, possa essere eseguita durante un intervallo di tempo dato, quando la manutenzione è assicurata nelle condizioni date e mediante l'uso di procedure e mezzi prescritti [3]. Tipicamente si misura in $/km e può essere divisa in due sottocasi:

- Manutenzione correttiva, a sua volta dipendente dall'affidabilità logistica (meno un sistema è affidabile e più spesso dovrà essere riparato)
- Manutenzione preventiva, dipendente in buona parte dalle caratteristiche di sicurezza del sistema analizzato

L'analisi di manutenibilità si pone come obiettivi principali la valutazione e l'analisi del progetto al fine di potenziare la maintainability del sistema.

La manutenibilità è una delle caratteristiche principali di un progetto, ed i legami con gli altri requisiti sono molto numerosi. I requisiti di manutenibilità devono essere:

- Pianificati e inclusi dentro la documentazione del progetto
- Specificati ad alto livello
- Progettati attraverso processi iterativi di analisi funzionali, allocazione di requisiti, trade-off e ottimizzazioni, sintesi e caratteristiche del componente
- Misurati in termini di adeguatezza attraverso test e valutazioni di sistema

I requisiti di manutenibilità devono essere strettamente integrati con tutte le altre caratteristiche di progetto.
La manutenibilità, essendo una caratteristica del progetto, può essere espressa in termini di fattori di frequenza di manutenzione, tempi di manutenzione e costi di manutenzione.

La manutenzione include tutte le azioni necessarie per mantenere un sistema, o ripristinarlo, allo stato operativo desiderato e può essere classificata nei seguenti modi:

- Manutenzione correttiva (corrective maintenance): include tutte le azioni manutentive non pianificate eseguite, dovute ad un guasto al sistema, per ripristinarlo in una condizione specifica. La manutenzione correttiva include l'identificazione e la verifica dei guasti, la loro localizzazione e l'isolamento, il disassemblaggio per accedere ai componenti guasti, rimuoverli e rimpiazzarli con nuovi o ripararli sul posto, riassemblaggio e verifica del corretto funzionamento. Inoltre, azioni di manutenzione non programmata possono risultare necessarie in seguito ad un possibile guasto, anche se successive indagini indicassero che non si sono verificate failure (ad esempio falsi allarmi). Le manutenzioni non programmate possono essere espresse in termini di frequenza ($MTBM_u$), tempo trascorso (Mean Time To Repair MTTR o M_{ct}) ed ore-lavoro per ore operative (MLH/OH).
- Manutenzione preventiva (preventive maintenance): include tutte le azioni di manutenzione programmata avvenute per assicurare che il sistema operi in determinate condizioni. La manutenzione preventiva include ispezioni periodiche, monitoraggio dello stato del sistema, sostituzione di elementi critici e calibrazioni periodiche. Alcune azioni di manutenzione vengono effettuate nei periodi in cui il sistema non è operativo, mentre altre possono essere espletate durante il normale svolgimento delle azioni o in stand-by. La manutenzione preventiva può essere valutata in termini di frequenza ($MTBM_s$), tempo di inattività (Mean Preventive Maintenance Time M_{pt}), ed ore-lavoro per ore operative (MLH/OH).
- Manutenzione predittiva (predictive maintenance): spesso si riferisce ad un programma di monitoraggio delle condizioni del sistema preso in esame e di manutenzione preventiva. Vengono applicati metodi di monitoraggio diretto, utilizzati per determinare l'esatto stato dei componenti al fine di predire possibili degradazioni e per focalizzare l'attenzione sulle aree in cui è necessaria la manutenzione; per stabilire i requisiti di questa attività è necessario conoscere in che modo i diversi componenti possano guastarsi (capire quindi la fisica della failure). L'obiettivo è quello di definire dopo quanto tempo si possa manifestare una failure in un determinato componente e di adottare di conseguenza le necessarie misure preventive.
- Prevenzione della manutenzione (maintenance prevention): con questo termine si indicano quelle azioni che è necessario applicare per ottenere un "sistema senza manutenzione". In particolare, consiste nel progettare e sviluppare la necessaria strumentazione per affidabilità e manutenibilità con l'obiettivo di minimizzare i tempi di fermo legati alla manutenzione, migliorando produttività e Life Cycle Cost.

1.4.1 Misure per la manutenibilità

Riferendosi alla normativa IEC 50126 [3], si definiscono i seguenti parametri di manutenibilità:

Parametro	**Simbolo**	**Dimensione**
Tempo di inattività medio	MDT	Tempo, distanza, ciclo
Tempo/distanza media tra manutenzioni	MTBM/MDBM	Tempo, distanza, cicli
MTBM/MDBM, Correttiva o Preventiva	MTBM (c, p), MDBM (c, p)	Tempo, distanza, cicli
Tempo medio alla manutenzione	MTTM	Tempo
MTTM, Correttiva o Preventiva	MTTM (c, p)	Tempo
Tempo medio al ripristino	MTTR	Tempo
Tempo di falso allarme	FAR	$Tempo^{-1}$
Copertura delle avarie	FC	Adimensionale
Copertura delle riparazioni	RC	Adimensionale

Tabella 1.5: Parametri di disponibilità (IEC 50126)

1.4.2 Life Cycle Cost

Come detto in precedenza, la maintainability di un sistema è utilizzata per valutare l'economicità della manutenzione: in questo caso si parla di Life Cycle Cost (LCC), analisi che consiste nel determinare i costi di manutenzione lungo tutto il ciclo di vita del sistema, monetizzando la manutenzione correttiva e preventiva. I costi da valutare saranno relativi a:

- Manodopera dei manutentori suddivisa per professionalità
- Costo delle parti di ricambio
- Costo delle attrezzature speciali

E' possibile definire il LCC come una lista di tutte le attività di manutenzione correttiva e preventiva che dovranno essere svolte sul sistema con relativi tassi di guasto o scadenze, costi di materiali e tempi di esecuzione.

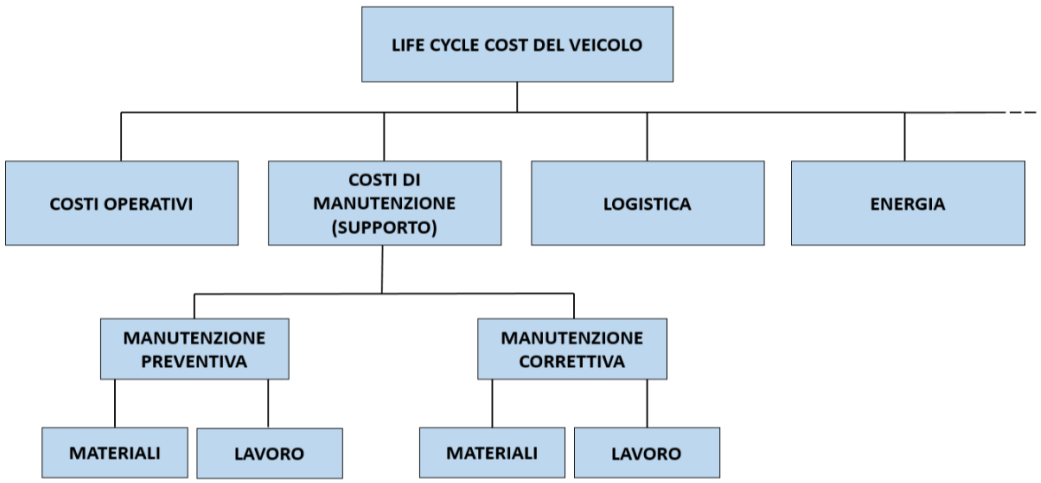

Figura 1.5: Schema composizione LCC

1.5 Safety

Si definisce sicurezza (Safety) l'assenza di un rischio inaccettabile. Questo concetto è applicabile ad ogni settore, ma a seconda dell'ambito varia il numero di persone coinvolte dal funzionamento di un oggetto e quindi variano le conseguenze legate al malfunzionamento. Treni e aerei, che possono trasportare centinaia di persone, sono quindi potenzialmente più pericolosi di un'automobile; saranno quindi sottoposti ad una regolamentazione in materia di sicurezza più severa. Dato che la probabilità che si verifichino guasti pericolosi è bassa, diventa molto difficile realizzare indagini di tipo sperimentale; per lo studio delle conseguenze di guasti critici vengono utilizzate diverse tecniche, che danno informazioni per lo più di tipo qualitativo: una di queste è la FMECA, grazie alla quale è possibile evidenziare guasti critici.

A seconda dei settori tecnici di appartenenza, le norme sulla sicurezza sono differenti:

- Sistemi di controllo in generale: EN 61508
- Aerospazio: DO-160, DO-178B...
- Sistemi di controllo di macchine in generale: ISO 13849
- Sistemi di controllo di macchine agricole: ISO 25119
- Sistemi di controllo di automobili: ISO 26262

1.5.1 Safety Integrity Level

Un parametro fondamentale legato alla sicurezza è il Safety Integrity Level (SIL) il quale, preso in considerazione il livello di pericolosità di un sistema, permette di assegnare di conseguenza un livello di sicurezza con cui deve essere realizzato. Questo parametro è stato introdotto per far fronte alle avarie ed agli errori sistematici (errori di progettazione/produzione). I requisiti di sicurezza specifici per hardware e software (qualitativi e quantitativi) dipendono dal SIL. Il SIL è la probabilità richiesta ad un sistema di sicurezza per effettuare correttamente le sue funzioni in tutte le condizioni previste; è funzione dell'affidabilità dei componenti selezionati e della frequenza di prova stabilita [5]. Una volta valutato, quantitativamente o dimensionalmente, un rischio ci si pone l'obiettivo di ridurlo fino ad un livello accettabile. Per ottenere questo si interpongono dei livelli di sicurezza che sottraggono quantità di rischio fino ad un livello, denominato residuo, che dovrà corrispondere o, meglio ancora, superare gli obiettivi stabiliti inizialmente.
Essenzialmente il SIL è legato ai tassi di guasto dei singoli componenti e la sua valutazione trae origine da metodologie di calcolo di affidabilità della catena specifica adottata. È facile notare, nella determinazione del SIL, che tale analisi numerica va oltre i requisiti cogenti di una Direttiva di prodotto, la quale in genere non regolamenta accuratamente aspetti legati alla manutenzione. L'approccio della normativa IEC 61508 invece impone la scelta di tempi di controllo (Interval Test-proof test) e dell'eventuale ripristino dei requisiti del singolo componente della catena (MTTR). Spesso il livello di SIL determina anche la ridondanza da adottare.
Lo standard di realizzazione deve essere associato ad uno standard elevato di verifica, la norma IEC 61508 introduce la figura dell'Advisor/Assessor (Esperto di supporto/perito): questi è deputato a validare ogni procedura concepita dal fabbricante nella realizzazione del prodotto, presunto conforme, e ad assisterlo nella corretta realizzazione del fascicolo tecnico da allegarsi al prodotto.
La normativa IEC 61508 definisce quattro livelli di Safety Security Level (da SIL 1 a SIL 4), ciascuno dei quali definisce una misura quantitativa della necessaria riduzione del rischio e quindi il grado di affidabilità che il sistema di sicurezza deve raggiungere per poter garantire

tale riduzione. Nelle tabelle sottostanti sono indicati i limiti corrispondenti a ciascun valore di SIL

Safety Integrity Level	**Low Demand Mode of Operation**
4	$>= 10^{-5}$ to 10^{-4}
3	$>= 10^{-4}$ to 10^{-3}
2	$>= 10^{-3}$ to 10^{-2}
1	$>= 10^{-2}$ to 10^{-1}

Tabella 1.6: SIL per operazioni "Low-Demand" [5]

Safety Integrity Level	**Continuous/High Demand Mode of Operation**
4	$>= 10^{-9}$ to 10^{-8}
3	$>= 10^{-8}$ to 10^{-7}
2	$>= 10^{-7}$ to 10^{-6}
1	$>= 10^{-6}$ to 10^{-5}

Tabella 1.7: SIL per operazioni "Continuous/High Demand" [5]

Ogni componente sarà caratterizzato da un determinato Safety Integrity Level. La normativa EN 61508 afferma che si parla di modo "Low Demand" quando un sistema legato alla sicurezza ha un tasso di guasto minore di uno all'anno, di "High Demand" se superiore.

1.5.2 Safety nell'ambito ferroviario

Le funzioni di sicurezza all'interno dei sistemi dovrebbero essere attuate usando architettura, metodi, strumenti e tecniche definiti in altre norme pertinenti di dettaglio. Per esempio, in campo ferroviario, la EN 50128 definisce gli standard da seguire per sviluppare sistemi software, mentre la EN 50129 definisce un processo per l'accettazione e l'approvazione di sistemi elettronici di segnalamento ferroviario.
Se si concentra l'attenzione all'ambito ferroviario italiano, secondo la definizione del legislatore (art. 3 del D.Lgs 162/2007) l'incidente ferroviario è un "evento improvviso indesiderato e non intenzionale avente conseguenze dannose" e si divide nelle seguenti tipologie:

- Collisioni fra treni o fra treno ed ostacolo
- Deragliamento
- Incidente al passaggio a livello
- Incidente a persone causato da materiale rotabile in movimento (esclusi i suicidi)
- Incendio al materiale rotabile
- Altro

L'incidente ferroviario è definito "grave" nel caso di collisione ferroviaria o deragliamento di treni che causa la morte di almeno una persona o il ferimento grave di cinque o più persone o seri danni (almeno 2 milioni di euro).

Capitolo 2

La Fault Tree Analysis

2.1 Introduzione

La Fault Tree Analysis (FTA) è l'analisi che permette di stabilire la probabilità di accadimento di un evento critico per il sistema che viene preso in esame. Consiste nella costruzione e nella successiva analisi dell'albero dei guasti (fault tree), composto all'estremità superiore da un top event (l'evento critico che identifica il guasto dell'intero sistema con la conseguente nascita di problemi dal punto di vista della sicurezza) collegato ad una serie di blocchi rappresentanti eventi che, uniti tramite opportune porte logiche, ne determinano la probabilità di accadimento [6]. Gli eventi che portano ad un top event coincidono con possibili guasti o incidenti, come ad esempio la rottura di componenti hardware, errori umani o del software; ogni albero dei guasti ha uno ed un solo top event specifico. Nella seguente figura è possibile osservarne un tipico schema.

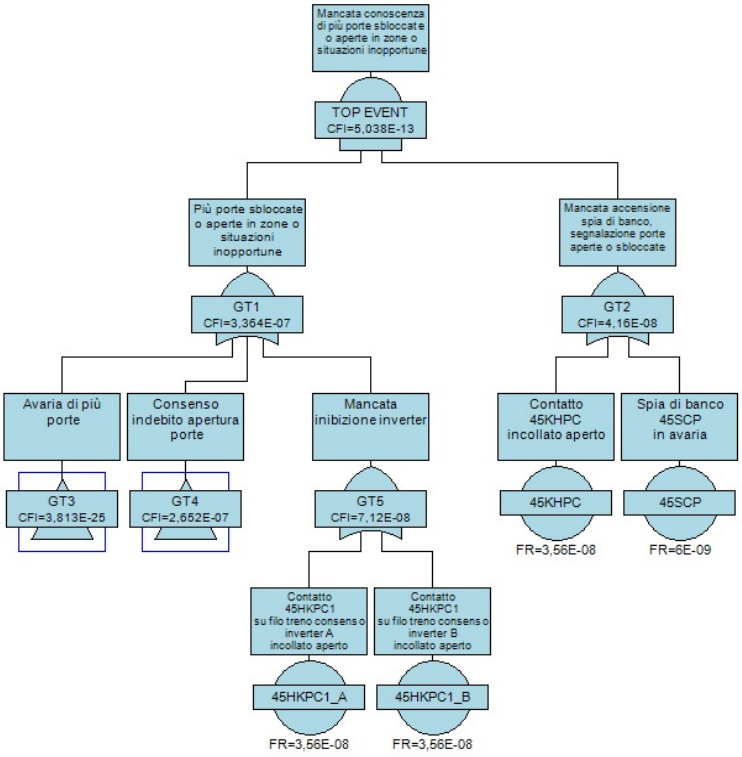

Figura 2.1: Esempio di albero dei guasti

Come già accennato nel primo capitolo, la Fault Tree Analysis è un modello quantitativo di analisi strettamente legato alla teoria delle probabilità, riguardo alla quale risulta conveniente

fornire alcune informazioni generali.

2.2 La teoria delle probabilità

La teoria delle probabilità analizza in modo analitico gli eventi che costituiscono un albero dei guasti. Generalmente la teoria delle probabilità viene descritta attraverso tre classi principali:

- La teoria degli insiemi
- L'algebra delle probabilità
- Il teorema di Bayes

2.2.1 La teoria degli insiemi

La teoria degli insiemi viene vista come un approccio generale nel quale gli eventi, dei quali bisogna determinare la probabilità di accadimento, vengono rappresentati come degli insiemi [6]. Un esempio in grado di definire con più chiarezza le basi di questa teoria riguarda l'esito di eventi casuali, come ad esempio il lancio dei dadi da gioco: i possibili esiti possono essere 1, 2, 3, 4, 5, 6; questi vengono definiti elementi ed un evento appartenente ad un albero dei guasti può essere visto come una collezione di questi. Si considerino, ad esempio, i seguenti possibili eventi associati al lancio dei dadi:

- A: esce il numero 2
- B: il risultato è un numero pari
- C: il risultato è minore di 4
- D: esce un numero qualsiasi
- E: il risultato è divisibile per 7

Ognuno di questi eventi può essere considerato come un particolare insieme di elementi legati al possibile esito dell'esperimento. In questo caso:

- A = 2
- B = 2, 4, 6
- C = 1, 2, 3
- D=1, 2, 3, 4, 5, 6
- E:=insieme vuoto

Si può notare che l'elemento 1 appartiene a C e D ma non ad A e B, mentre gli elementi di A, B, C sono sottoinsiemi di D. Esiste una rappresentazione grafica che permette una facile visualizzazione della teoria degli insiemi, il diagramma di Venn:

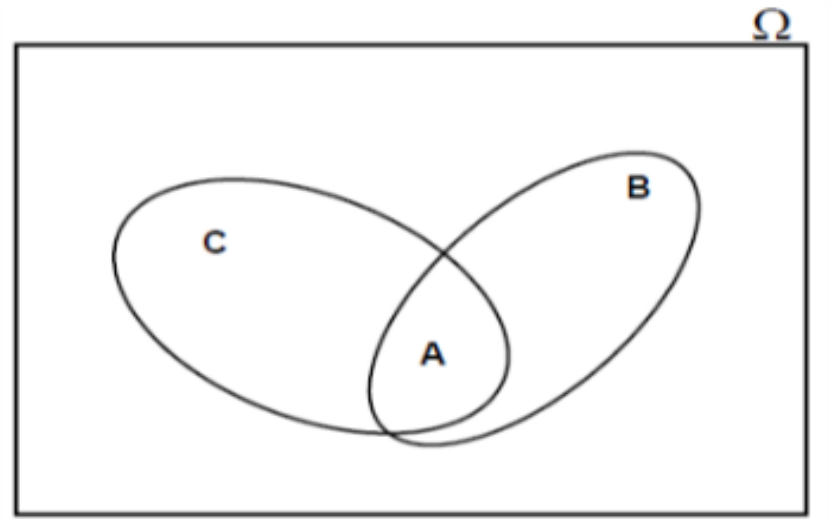

Figura 2.2: Diagramma di Venn

L'insieme universo (Ω) è rappresentato come una figura geometrica (solitamente un rettangolo) mentre i sottoinsiemi (che corrispondono agli eventi) di interesse sono collocati all'interno. L'unione di due insiemi (eventi) X ed Y è un insieme che contiene gli elementi di entrambi, e viene indicato come

$$\mathrm{X} \cup \mathrm{Y}$$

L'intersezione tra due insiemi X ed Y, ad indicare l'insieme di elementi che questi hanno in comune, viene indicato come

$$\mathrm{X} \cap \mathrm{Y}$$

Il complemento di un insieme X, ossia quello che contiene gli elementi non appartenenti ad X viene indicato con $\overline{X}$.
Tenendo a mente questi concetti è possibile capire come la teoria degli insiemi possa essere calata nell'analisi degli alberi dei guasti: se si considerano infatti tre eventi A, B e C e si suppone che il top event avverrà al verificarsi di due o più di questi, si ottiene che gli eventi corrispondenti alla failure di sistema sarebbero:

- S_1=A$\overline{BC}$
- S_2=$\overline{A}$B$\overline{C}$
- S_3=$\overline{AB}$C
- S_4=$\overline{ABC}$

ed il top event, la failure del sistema S, sarebbe pari a S=$S_1 \cup S_2 \cup S_3 \cup S_4$.

2.2.2 L'algebra delle probabilità

Se si considera un evento casuale e si prendono in considerazione due suoi possibili esiti, come ad esempio A e B, questi si dicono mutualmente esclusivi se in un singolo esito dell'esperimento non possono accadere entrambi. In questo caso, l'espressione della probabilità di accadimento è la seguente:

$$P(AorB) = P(A) + P(B) \tag{2.1}$$

Dal punto di vista dell'albero dei guasti, la rappresentazione grafica di eventi mutualmente esclusivi viene identificata dalla porta OR

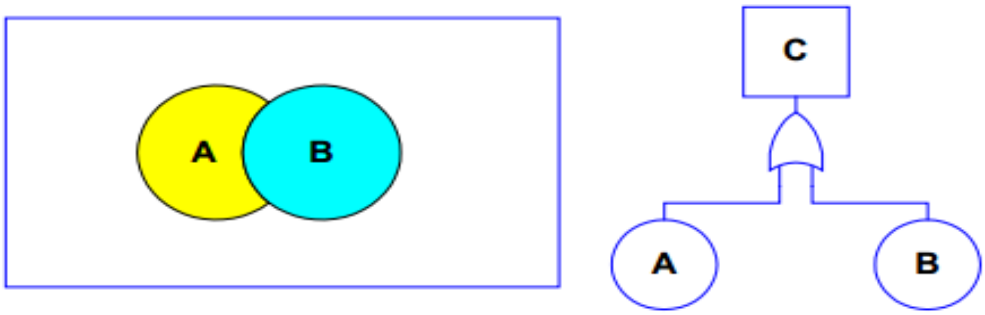

Figura 2.3: Eventi mutualmente esclusivi

Per eventi non mutualmente esclusivi, viene utilizzata la formula più generica

$$P(AorB) = P(A) + P(B) - P(AandB) \tag{2.2}$$

Questa espressione viene chiamata regola dell'addizione per le probabilità.
Si considerino ora due eventi mutualmente indipendenti: questo significa che, per il numero di volte in cui l'esperimento viene ripetuto, l'accadimento (o il non accadimento) dell'evento A non ha influenza sull'accadimento (o non accadimento) di B e viceversa. Dal punto di vista sistemistico, se due sistemi operano in parallelo non influenzandosi a vicenda, il guasto di uno dei due non avrà effetto sull'altro. In questo caso, le failure dei componenti sono eventi indipendenti. Se A e B sono mutualmente indipendenti, allora

$$P(AandB) = P(A) * P(B) \tag{2.3}$$

Questa espressione spesso viene chiamata regola della moltiplicazione; in questo caso, il corrispettivo grafico per l'albero dei guasti è rappresentato dalla porta logica AND

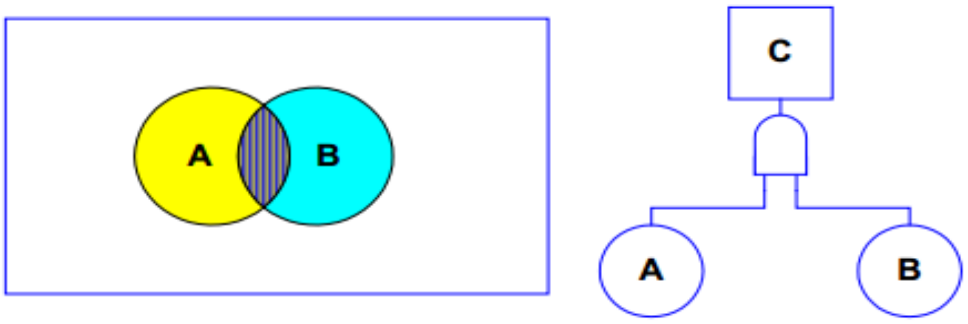

Figura 2.4: Eventi mutualmente indipendenti

Molto spesso gli eventi non sono mutualmente indipendenti, in quanto il verificarsi dell'uno influenza la probabilità di accadimento dell'altro. In questo caso si ha che

$$P(AandB) = P(A)P(B|A) = P(B)P(A|B) \tag{2.4}$$

dove P(B|A) indica la probabilità di accadimento dell'evento B quando A si è già verificato.

2.2.3 Il teorema di Bayes

Le formule sviluppate da Sir Thomas Bayes ricoprono un importante ruolo all'interno della teoria delle probabilità. Per definire il teorema di Bayes è necessario dividere l'insieme universo Ω considerato in precedenza nel seguente modo

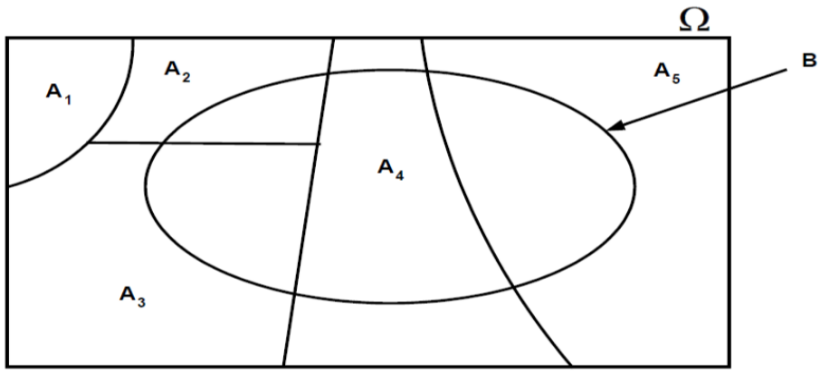

Figura 2.5: Partizione dell'insieme universo

dove ogni insieme di A costituisce una partizione dell'insieme universo ed ha le seguenti caratteristiche:

$$A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5 = \Omega \tag{2.5}$$

Si può notare che l'insieme B ha le seguenti caratteristiche:

$$(B \cap A_1) \cup (B \cap A_2) \cup (B \cap A_3) \cup (B \cap A_4) \cup (B \cap A_5) = B \tag{2.6}$$

Consiederando ora l'equazione di probabilità dell'intersezione si ha che

$$P(A_K \cap B) = P(A_K|B)P(B) = P(B|A_K)P(A_K) \tag{2.7}$$

o

$$P(A_K \cap B) = \frac{P(A_K \cap B)}{P(B)} = \frac{P(B|A_K)P(A_K)}{P(B)} \tag{2.8}$$

ipotizzando che gli eventi $B \cap A_i$ siano mutualmente esclusivi, si ottiene la formula del teorema di Bayes

$$P(A_K|B) = \frac{P(B|A_K)P(A_K)}{\sum P(B|A_i)P(A_i)} \tag{2.9}$$

L'equazione è valida in generale per ogni numero di eventi A_n mutualmente esclusivi.

2.3 Algebra booleana nella Fault Tree Analysis

Un albero dei guasti, come si è potuto intuire, è un diagramma logico basato sul fatto che alcuni eventi devono accadere perché altri di conseguenza si verifichino [6]. Gli eventi vengono definiti guasti se, come effetto del loro accadimento, comportano la rottura dei sistemi ad essi riferiti. Il blocco rappresentante l'output finale (il guasto principale del sistema) è definito top event e rappresenta la punta dell'albero; sotto di esso si diramano gli eventi intermedi e basici, collegati tra di loro tramite porte logiche, che combinati in modo opportuno portano alla

failure di sistema. Per stabilire la probabilità di accadimento di un dato evento, è necessario combinare tra loro secondo algebra booleana i diversi parametri di input degli eventi base; la logica booleana è espressa attraverso le porte logiche.

2.3.1 Porta OR

Come descritto nel paragrafo della teoria delle probabilità, questa porta rappresenta l'unione di due o più eventi: è l'equivalente booleano dell'operazione somma e, se si considerano due eventi A e B, si ha l'espressione

$$Q = A + B \tag{2.10}$$

Affinché l'evento Q si realizzi, basta che si verifichi solamente uno dei due eventi. In termini di probabilità:

$$P(Q) = P(A) + P(B) - P(A \cap B) = P(A) + P(B) - P(A)P(B|A) \tag{2.11}$$

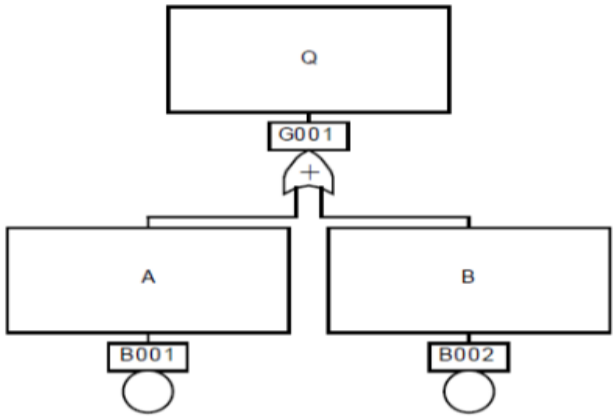

Figura 2.6: Porta OR a due ingressi

Possono essere fatte le seguenti ossevazioni:

- Se A e B sono due eventi mutualmente esclusivi, allora $P(A \cap B)$=0
- Se A e B sono due eventi indipendenti allora P(B|A)=P(B)
- Se B è dipendente da A, ossia che quando accade A si verifica anche B, allora P(B|A)=1
- Se A è dipendente da B, ossia che quando accade B si verifica anche A, allora P(A|B)=1
- Per stime conservative è possibile effettuare l'approssimazione per cui P(Q)$\simeq$P(A)+P(B)
- Se A e B sono eventi indipendenti e con bassa probabilità di accadimento, allora è valida la semplificazione P(Q)=P(A)+P(B)

2.3.2 Porta AND

La porta AND rappresenta l'intersezione tra due eventi. E' l'equivalente booleano della moltiplicazione e, se si considerano due eventi A e B

$$Q = A * B \tag{2.12}$$

Affinché l'evento Q si realizzi, sia A che B devono verificarsi. In termini di probabilità:

$$P(Q) = P(A)P(B|A) = P(B)P(A|B) \tag{2.13}$$

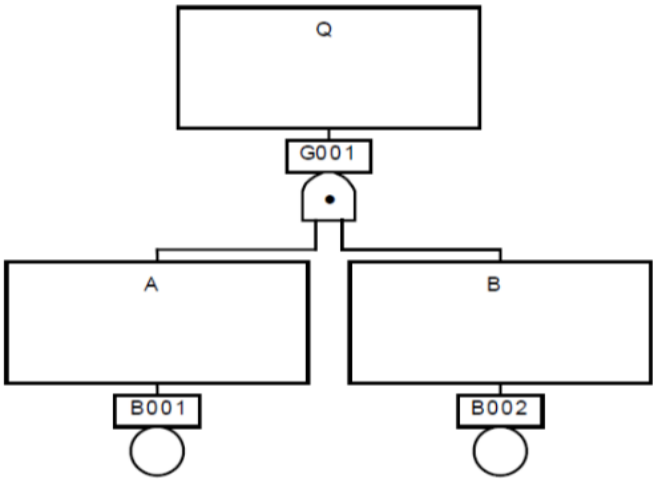

Figura 2.7: Porta AND a due ingressi

Possono essere fatte le seguenti osservazioni:

- Se A e B sono eventi indipendenti, allora P(B|A) = P(B), P(A|B) = P(A), P(Q) = P(A)P(B)
- Se A e B sono dipendenti, ed in particolare B si realizza sicuramente se si verifica A, allora vale l'approssimazione P(Q)=P(A)

2.4 Analisi albero dei guasti

Al contrario di altre analisi basate su metodi induttivi (approccio bottom-up), quali la FMECA e l'RBD, la Fault Tree Analysis è un metodo deduttivo (approccio top-down) [6]. La differenza tra i due metodi è definita dalla direzione dell'analisi: la FTA incomincia da un evento indesiderato e si muove a ritroso in cerca delle cause che lo hanno scatenato, mentre un metodo induttivo parte dagli eventi base per poi risalire fino al guasto finale.
Per essere efficacemente rappresentata, una Fault Tree Analysis deve seguire i seguenti passi:

- Identificazione dell'obiettivo della FTA
- Definizione del top event
- Definizione dello scopo della FTA
- Definizione del metodo di risoluzione
- Definizione delle regole base da seguire
- Costruzione del FT
- Valutazione del FT
- Interpretazione e presentazione dei risultati

Il top event di un fault tree è il punto di partenza da cui partiranno tutte le analisi successive; se questo viene identificato in maniera non corretta, inciderà negativamente in cascata su tutto il processo. E' quindi di estrema importanza definire e capire gli obiettivi dell'analisi ed il problema che si vuole risolvere. Per definire correttamente il top event è quindi necessario:

- Definire i criteri per i quali questo si manifesta
- Assicurarsi che sia coerente con il problema da risolvere e gli obiettivi dell'analisi

E' necessario definire lo scopo della FTA, ossia indicare quali eventi possono incidere sulla possibilità di accadimento del top event e definire i limiti dell'analisi. Le regole base da seguire per la risoluzione della FTA includono le procedure e la nomenclatura da assegnare ad ogni evento e porta logica; queste possono includere la descrizione del modo di guasto di uno specifico componente, di un errore umano e delle cause comuni di guasto (Common Cause Failure, CCF). La valutazione del FT è sia di tipo quantitativo che qualitativo, con quest'ultima che fornisce informazioni sui minimal cut sets (definiti più avanti) per il top event, mentre la valutazione quantitativa definisce non solo la probabilità di accadimento del top event, ma anche i cut sets dominanti che contribuiscono al suo accadimento. Un cut set è un insieme di eventi basici i quali, se dovessero tutti verificarsi, porterebbero all'accadimento del top event. Il più piccolo insieme di basic events viene definito minimal cut sets. Infine, l'interpretazione dei risultati fornisce una visione generale sull'impatto che l'accadimento di un top event può avere sul sistema globale.

Il paradigma base per la costruzione di un albero dei guasti è "pensare in piccolo": per ogni evento che viene analizzato, sono identificati solamente gli immediati eventi necessari e sufficienti affinché questo si realizzi. Nei paragrafi successivi verranno definiti i passi che solitamente vengono seguiti per la costruzione dell'albero dei guasti, partendo dalla definizione delle caratteristiche dei blocchi da cui è costituito.

2.4.1 Blocchi tipici costituenti un Fault Tree

Nella figura seguente viene rappresentato un tipico albero dei guasti

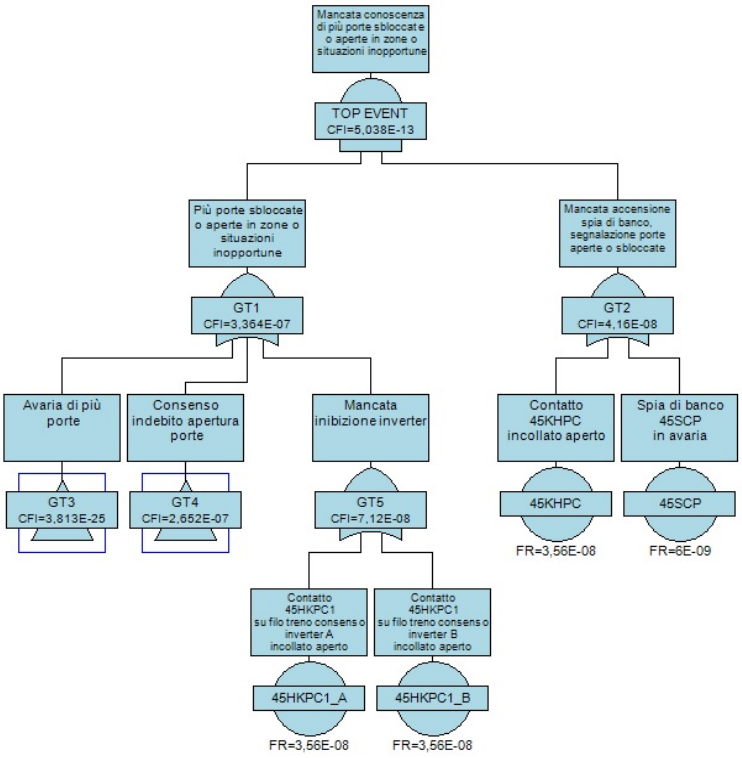

Figura 2.8: Albero dei guasti

Come si può notare, esistono alcune definite tipologie di blocchi che permettono la realizzazione di un albero: eventi, porte logiche e porte di trasferimento.

Eventi base

Gli eventi base (basic events) non hanno ulteriori diramazioni verso il basso. A questi eventi dovranno essere forniti determinati input al fine di ottenere risultati intermedi che, una volta combinati tra di loro a seconda delle porte logiche presenti nei livelli superiori dell'albero, permetteranno di ottenere la probabilità di accadimento del top event. Esistono quattro tipi di basic events:

- Primary events: descrivono il guasto iniziale
- Undeveloped events: indicano eventi che non possono essere descritti in dettaglio per mancanza di ulteriori informazioni
- Conditional events: utilizzati per registrare qualsiasi informazione o restrizione che deve essere applicata a qualche porta logica. Sono utilizzati soprattutto con porte INHIBIT e PRIORITY-AND
- House events: utilizzati per identificare quegli eventi che normalmente ci si aspetta che accadano.

Porte logiche

Le due tipologie di porte logiche più utilizzate per la costruzione di un fault tree sono quelle già presentate nei paragrafi precedenti, ossia quelle AND ed OR. Tutte le altre sono sottocasi di queste due:

- Porta OR: l'evento in output si verifica se uno o più eventi in input si verificano

- Porta AND: l'evento in oputput si verifica se tutti gli eventi in input si verificano
- Porta VOTE: caratterizzata da un numero, è utilizzata per indicare il numero di eventi in input che si devono verificare affinché si realizzi quello in output
- Porta INHIBIT: caso speciale di porta AND, l'output è causato da un singolo input, che però si realizza solo se determinate condizioni si verificano
- Porta EXCLUSIVE OR: caso speciale di porta OR, è caratterizzata da due input e un output, che si verifica se uno dei due si realizza ma non entrambi
- Porta PRIORITY AND: l'output si realizza solo se tutti gli input si verificano in una determinata sequenza

Blocchi di trasferimento (transfer symbols)

Sono utilizzati per semplificare la struttura dell'albero dei guasti, al fine di evitare ripetizioni di rami che potrebbero renderne più complicata l'interpretazione. La porta "transfer in" si collega alla corrispondente "transfer out" che contiene un'ulteriore porzione dell'albero.

2.4.2 Proprietà dei componenti

Generalmente i componenti che caratterizzano un sistema si dividono in attivi e passivi. Un componente passivo contribuisce in modo quasi statico al funzionamento del sistema: può ad esempio agevolare il trasferimento di energia da un posto ad un altro o trasmettere il moto. Una sua rottura comporterebbe la perdita del passaggio dell'informazione da un punto all'altro del sistema. Un componente viene definito attivo quando contribuisce dinamicamente al funzionamento del sistema. Generalmente, questo tipo di componente necessita di un input in ingresso per generare un segnale in uscita; se un componente attivo si guasta, il segnale di output utile al funzionamento del sistema non viene più generato o viene elaborato in modo non corretto.
Un altro modo per suddividere i componenti è in base alla tipologia di guasto che subiscono. In particolare si hanno tre gruppi:

- Guasti primari
- Guasti secondari
- Guasti di controllo

Sono definiti primari quei guasti che occorrono ad un componente durante lo svolgimento della funzione per cui è stato progettato, ad esempio un serbatoio realizzato per resistere ad una pressione p_0 che si rompe ad una pressione p$<p_0$. I guasti secondari si identificano nei componenti che si rompono quando operano in condizioni o in ambienti per cui non sono stati realizzati, come ad esempio la rottura del serbatoio descritto in precedenza a seguito di una pressione p$>p_0$. Un guasto di controllo infine si realizza quando un componente assolve alla funzione per cui è stato realizzato, ma in un momento o in un luogo errati.

2.4.3 Costruzione di un Fault Tree

La definizione delle regole e delle metodologie per lo sviluppo dell'albero dei guasti è un processo che si è evoluto nel corso di più di 50 anni [6]. E' possibile individuare due regole base per la sua costruzione:

- Definire con precisione quale sia il guasto e sotto quali determinate condizioni questo si verifica.

- Se la risposta alla domanda "Il guasto comporta la rottura del componente?" è "Sì", l'evento viene classificato come "guasto dello stato del componente"; se è "No" allora viene classificato come "guasto dello stato del sistema".

Un esempio di guasto legato allo "stato del componente" è la mancata apertura di una valvola, che potrebbe essere legata sia intrinsecamente alla valvola stessa sia al non ricevimento del segnale necessario alla sua apertura da parte dell'operatore. Un guasto relativo allo stato del componente è sempre modellato con una porta OR o come un evento primario.
Uno "stato di sistema" invece può essere rappresentato dal mancato passaggio di un fluido attraverso due pompe ridondanti: il guasto coinvolge due sistemi.
La profondità con cui deve essere realizzata un'analisi dei guasti determina quanto è significativo il risultato. La regola generale da seguire è quella di sviluppare un albero fino all'identificazione delle dipendenze funzionali e che sia legata alle informazioni disponibili ed agli obiettivi dell'analisi.

2.5 Valutazioni quantitative

Per quantificare la probabilità di accadimento di un top event è necessario definire quella dei diversi basic events, che viene poi propagata verso l'alto secondo la logica booleana. Le probabilità legate agli eventi base possono essere ricondotte all'evento principale attraverso l'approccio del minimal cut sets, molto utilizzato dai software commerciali in quanto fornisce delle importanti informazioni aggiuntive a corredo di quelle principali [8].
In quanto il top event è espresso come l'unione dei minimal cut sets, la sua probabilità di accadimento può essere approssimata come la loro somma: questa approssimazione viene definita "rare event approximation" ed è utilizzata dalla maggior parte dei software in commercio. Dato che un minimal cut set è dato dall'intersezione di basic events, la sua probabilità di accadimento è data semplicemente dal loro prodotto, mentre la probabilità di accadimento del top event può essere espressa come la somma dei prodotti tra i diversi eventi basici. Dal punto di vista matematico la probabilità di accadimento del top event è data dalla formula

$$P(Top) = \sum P(M_i) \tag{2.14}$$

con

$$P(M_i) = P(BE_1)P(BE_2)...P(BE_K) \tag{2.15}$$

dove M_i denota un determinato cut set, mentre con BE si identificano i basic events.

I dati in input che solitamente devono essere assegnati agli eventi basici sono di tre tipi:

- La probabilità di guasto (failure probability) del componente in un determinato intervallo di tempo
- La probabilità di accadimento di un evento (event occurrence probability) in un determinato intervallo di tempo
- L'indisponibilità (unavailability, Q) del componente

Per calcolare la probabilità di guasto di un componente in un certo intervallo di tempo è necessario indicare in input il tasso di guasto (failure rate λ [1/h]) ed il tempo di missione del componente. Il tasso di guasto è pari a

$$\lambda = \lambda_0 d + \lambda_N (1 - d) \tag{2.16}$$

dove d = tempo operativo del componente o tempo di missione, λ_0 = tasso di guasto del componente in fase operativa, λ_N = tasso di guasto del componente in fase non operativa. La semplificazione standard che si applica è quella di considerare il tasso di guasto costante. Come già presentato nel capitolo 1, la probabilità di guasto del componente, definita inaffidabilità (unreliability) è data dalla formula

$$F(t) = 1 - e^{-\lambda t} \tag{2.17}$$

Come si può notare, questa formula è caratterizzata da una distribuzione esponenziale della probabilità di guasto. La distribuzione esponenziale è solitamente applicata nei casi in cui non si abbiano sufficienti dati di input da manipolare ed è largamente usata nello studio dell'affidabilità in quanto è l'unica che permette di considerare un failure rate λ costante.
La formula (2.17) viene chiamata cumulative distribution function (CDF) ed assume la generica forma

$$F(x) = \int_{-\infty}^{\infty} f(x)dx \tag{2.18}$$

f(x) viene definita probability density function (pdf) e, per una distribuzione esponenziale, assume la forma

$$f(t) = \lambda e^{-\lambda t} \tag{2.19}$$

La probabilità di accadimento di un evento in un determinato intervallo di tempo è formalmente simile alla probabilità di guasto: in questo caso λ identifica il tasso di accadimento dell'evento.
L'indisponibilità di un componente, Q, viene definita attraverso i parametri tasso di guasto, tempo medio di riparazione (average repair time τ) e tempo di test (test interval T) e l'espressione che ne consegue è la seguente:

$$Q = \lambda_0\tau/(1 + \lambda_0\tau) \simeq \lambda_0\tau \tag{2.20}$$

per componenti operativi e

$$Q = (1/2)\lambda_S T/(1 + 1/2\lambda_S T) + 1 - e_0^{-\lambda T} \simeq (1/2)\lambda_S T + \lambda_0\tau \tag{2.21}$$

per componenti in stand-by.

2.5.1 Dati in input

Per poter effettuare una Fault Tree Analysis è necessario fornire agli eventi base dei determinati input grazie ai quali è possibile determinare la loro probabilità di accadimento. Gli input da assegnare dipendono dal tipo di basic event che deve essere descritto.

Tasso di guasto del componente

E' richiesto per calcolare la probabilità di guasto del componente e la sua indisponibilità. Spesso questo non è un valore esatto ed è quindi necessario definire dei limiti di tolleranza che tengano conto non solo dell'incertezza statistica nella stima del valore fornito, ma anche del campo in cui il componente opera. In aggiunta, è necessario definire se il componente a cui si stanno fornendo gli input sia operativo o in stand-by, per capire il tipo di failure rate necessario; il tasso di guasto operativo è utilizzato per determinare la probabilità di guasto di quei componenti operativi a seguito di una corretta accensione, mentre per quelli in stand-by è necessario tenere in considerazione, per il calcolo dell'indisponibilità, il tempo medio in cui questi componenti rimangono non operativi per definire la probabilità di guasto.

Dati legati ad errori umani

L'analisi dell'affidabilità umana definisce la probabilità dell'accadimento di differenti tipi di azioni umane ed è di fondamentale importanza per la Fault Tree Analysis. Solitamente vengono utilizzate le opinioni di esperti per stimare il tasso di errore umano, in quanto è difficile avere a disposizione database contenenti questo genere di dati. In ogni caso, l'incertezza associata al tasso di errore umano dovrebbe essere tenuta in considerazione per le possibili variazioni nelle performance umane e condizioni in cui queste vengono svolte.

Dati legati ai Common Cause Failure

Prendendo in considerazione la normativa IEC 61508, viene definito un Common Cause Failure (CCF) come il risultato di uno o più eventi, che provoca errori concomitanti di due o più canali separati in un sistema a più canali, con conseguenti errori di sistema [5]. Un canale è definito come un elemento o un gruppo di elementi che implementano in modo indipendente una funzione di un sistema. I Common Cause Failure sono singoli eventi di errore che interessano più componenti o funzioni di un sistema e sono una parte importante di qualsiasi modello di affidabilità o di rischio, in quanto lavorano per annullare i miglioramenti offerti dalla ridondanza. Sono spesso coloro che contribuiscono maggiormente ai livelli di rischio e dovrebbero pertanto essere attentamente considerati. La maggior parte dei sistemi legati all'affidabilità sono caratterizzati da ridondanza, progettata per limitare l'impatto di un guasto di un singolo componente con lo scopo di evitare l'accadimento di un evento catastrofico. E' bene tenere a mente che raddoppiare il numero di componenti (ridondanza geometrica) non dimezza le probabilità di fallimento; in questo scenario, i CCF si oppongono a questa ridondanza riducendo l'indipendenza degli eventi. Esistono molti metodi di analisi che offrono soluzioni per tenere in considerazione i CCF, come ad esempio il metodo chiamato "β-factor" dato dal rapporto

$$\beta = \frac{\lambda_C}{\lambda_T} \tag{2.22}$$

con λ_C = guasti di causa comune e λ_T = guasti totali del componente. Generalmente i CCF sono più significativi per componenti con ridondanza attiva e diventano fattori determinanti all'aumento del numero di ridondanze.
L'importanza del contributo dei Common Cause Failure può essere illustrata dal seguente esempio: si considerino tre componenti ridondati che, affinché l'intero sistema si guasti, devono tutti rompersi. Per ogni componente la probabilità di guasto p è pari a $1 * 10^{-3}$, per cui la probabilità che i tre componenti si rompano in maniera indipendente è pari a

$$P_{independent} = p^3 = 10^{-9} \tag{2.23}$$

Di conseguenza, la probabilità che tutti e tre i componenti si guastino è di una su un miliardo. Si consideri ora la possibilità di una causa comune che porti alla rottura di tutti e tre i componenti: assumendo che tale eventualità si verifichi con un rateo pari all'1% di possibilità, quindi 10^{-2}, la probabilità che di guasto dei tre componenti è pari a

$$P_{CCF} = 10^{-5} \tag{2.24}$$

Di conseguenza, la probabilità di guasto di tutti i componenti è 10000 volte maggiore rispetto al caso $P_{independent}$.
Per integrare un contributo legato ad un CCF in un albero dei guasti, questo deve essere trattato in maniera separata, come mostrato nella seguente figura:

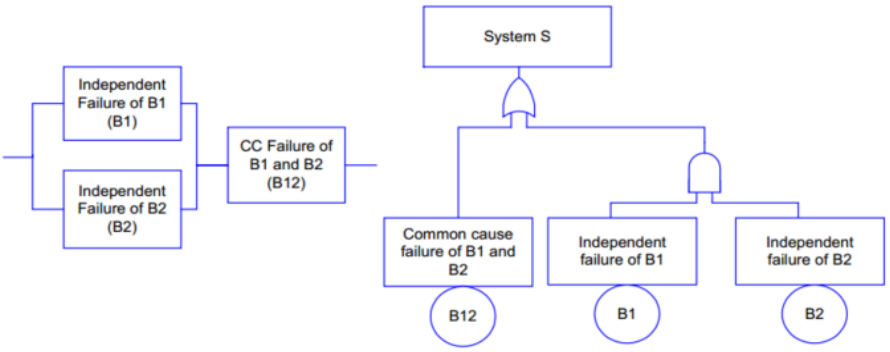

Figura 2.9: Common Cause Failure

Una buona regola da seguire è quella di includere il contributo dei CCF per ogni componente attivo con ridondanze.

2.5.2 Modelli di analisi degli eventi

Affinché venga effettuata un'analisi quantitativa, è necessario specificare i modelli che vengono utilizzati per il calcolo della probabilità di accadimento degli eventi. Esistono nella letteratura molti modelli applicabili al calcolo di questi parametri, utilizzabili a seconda del livello di precisione che si vuole raggiungere; tra tutti meritano un approfondimento i seguenti modelli [7]:

- Fixed unavailability and failure frequency (Fixed)
- Constant failure and repair rate (Rate)
- Dormant failure with periodic inspection (Dormant)
- Weibull failure model

Durante un'analisi, vengono calcolate l'indisponibilità Q e la frequenza di guasto ω di ogni evento secondo le regole matematiche del modello risolutivo scelto.

Fixed unavailability and failure frequency model

Questo modello è caratterizzato dal fatto che i valori di indisponibilità e frequenza di guasto non variano nel tempo e sono già disponibili come input per ogni evento base. Utile per definire gli errori dell'operatore o bug presenti nel software, l'utilizzo di questo modello è solitamente utilizzato da chi è alle prime esperienze nell'analisi degli alberi dei guasti.

Constant failure and repair rate model

Questo modello viene utilizzato per rappresentare quei componenti i cui guasti vengono immediatamente rilevati e riparati. Viene assunta una distribuzione esponenziale sia per il processo di guasto che per quello di riparazione. L'indisponibilità e la frequenza di guasto di un componente vengono rappresentate dalle seguenti formule

$$Q(t) = \frac{\lambda}{\lambda + \mu}(1 - e^{-(\lambda+\mu)t}) \tag{2.25}$$

$$\omega(t) = \lambda(1 - Q(t)) \tag{2.26}$$

dove

- Q(t) = indisponibilità del componente
- ω(t) = frequenza di guasto del componente
- λ = tasso di guasto del componente
- μ = tasso di riparazione del componente

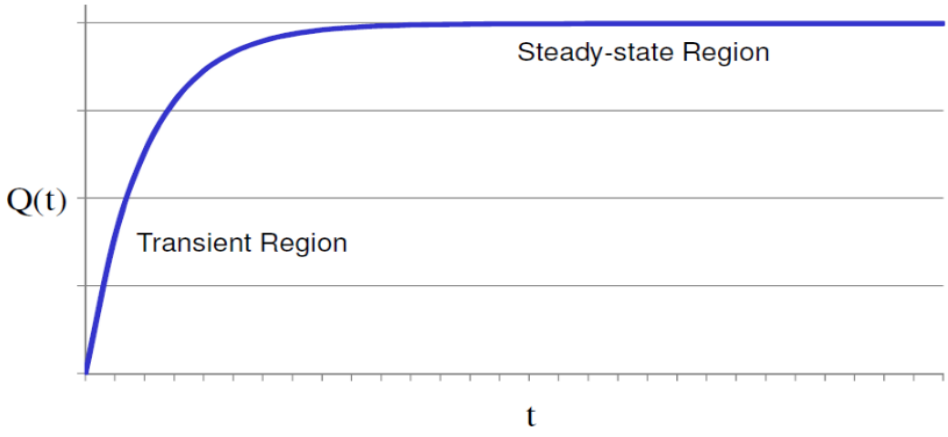

Figura 2.10: Variazione di indisponibilità nel tempo

Il seguente grafico mostra due regioni distinte, una di transitorio ed una stazionaria: la prima è applicabile a quei sistemi che hanno vita breve e che vengono soventemente sostituiti come quelli militari per cui, essendo (λ+μ)t $\ll 1$

$$Q(t) \simeq \lambda t \tag{2.27}$$

mentre la seconda viene applicata a componenti con cicli di vita più lunghi per i quali, essendo (λ+μ)t $\gg 1$

$$Q(t) = \frac{\lambda}{\lambda + \mu} \tag{2.28}$$

Per i componenti non riparabili, il tasso di riparazione μ è posto pari a 0. In questo caso la formula si riduce a

$$Q(t) = 1 - e^{-\lambda t} \tag{2.29}$$

Mean Time to Failure e Repair model

Questo modello è totalmente simile al Constant failure and repair rate, con l'eccezione che, come valori in input, non vengono forniti i tassi di guasto e riparazione bensì il Mean Time To Failure (MTTF) ed il Mean Time To Repair (MTTR) pari a

$$\lambda = \frac{1}{MTTF} \tag{2.30}$$

$$\mu = \frac{1}{MTTR} \tag{2.31}$$

Dormant failure with periodic inspection

I guasti ai componenti che caratterizzano i sistemi di protezione o quelli che spesso sono in stand-by solitamente non vengono immediatamente identificati finché non viene richiesto loro di entrare in funzione o quando viene effettuato un intervento di manutenzione. Tali componenti vengono definiti inattivi ("dormant") e per caratteristiche sono simili a quelli non riparabili durante il periodo che trascorre tra due ispezioni.

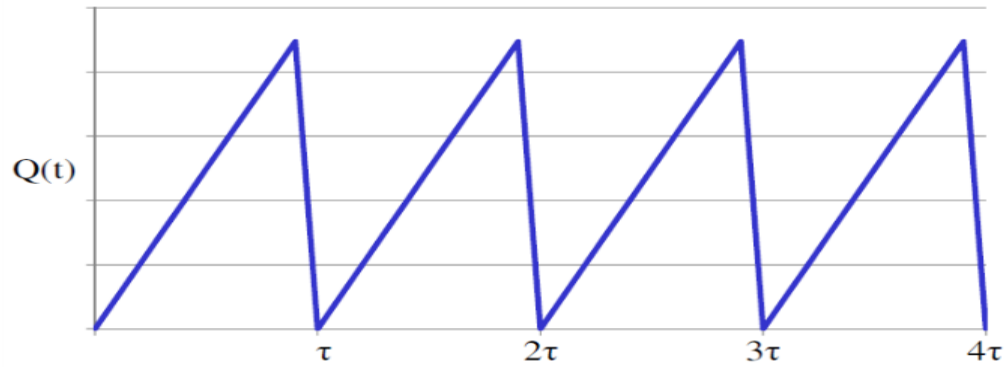

Figura 2.11: Variazione di indisponibilità nel tempo dormant model

Come si può evincere dalla figura, la variazione di indisponibilità nel tempo non assume più un andamento esponenziale, bensì periodico. Il modello presenta quindi un rischio massimo o medio di indisponibilità e frequenza di guasto:

$$Q_{mean} = \frac{\lambda\tau - (1 - e^{-\lambda\tau}) + \lambda MTTR(1 - e^{-\lambda\tau})}{\lambda\tau + \lambda MTTR(1 - e^{-\lambda\tau})} \tag{2.32}$$

dove

- Q_{mean} = indisponibilità media
- λ = tasso di guasto costante
- MTTR = Mean Time To Repair
- τ = intervallo d'ispezione

mentre la frequenza di guasto sarà pari a

$$\omega_{mean} = \lambda(1 - Q_{mean}) \tag{2.33}$$

Se viene utilizzato invece il modello a indisponibilità massima l'equazione avrà la seguente forma

$$Q_{max} = 1 - e^{-\lambda t} \tag{2.34}$$

con Q_{max} = indisponibilità massima per intervallo d'ispezione. La frequenza di guasto in questo caso sarà pari a

$$\omega_{max} = \lambda(1 - Q_{max}) \tag{2.35}$$

Weibull model

Il modello Weibull viene utilizzato per rappresentare i componenti non riparabili i cui tassi di guasto variano nel tempo. Nello specifico vengono identificati tre parametri

- η = parametro di vita caratteristica
- β = parametro di forma
- γ = parametro di locazione

che costituiscono la formula del failure rate r(t)

$$r(t) = \frac{\beta(t-\gamma)^{\beta-1}}{\eta^{\beta}} \tag{2.36}$$

L'inaffidabilità F(t) è data da

$$F(t) = 1 - exp[-(\frac{t-\gamma}{\eta})^{\beta}] \tag{2.37}$$

e, dato che il modello rappresenta i componenti non riparabili, questa sarà uguale all'indisponibilità

$$\mathrm{Q(t)} = \mathrm{F(t)}$$

2.5.3 Metodi di analisi dei sistemi

In questo paragrafo si definiscono i metodi utilizzati per risolvere un albero dei guasti esplicitando il minimal cut set al fine di calcolare i parametri caratteristici quantitativi del sistema:

- Q_{SYS} = indisponibilità del sistema
- ω_{SYS} = frequenza di guasto del sistema
- TDT_{SYS} = Total Down Time del sistema
- λ_{SYS} = Tasso di guasto o Conditional Failure Intensity (CFI) del sistema durante la vita
- F_{SYS} = inaffidabilità del sistema
- $MTTF_{SYS}$ = Mean Time To Failure del sistema
- $MTBF_{SYS}$ = Mean Time Between Failure del sistema
- $MTTR_{SYS}$ = Mean Time To Repair del sistema

I cut sets vengono generati seguendo la logica booleana con una metodologia di tipo bottom-up per arrivare fino alla formula del top event. Le seguenti espressioni sono un esempio in cui la logica booleana viene applicata per produrre i minimal cut sets:

$$\begin{gathered}
\mathrm{A{+}A{*}B = A} \\
\mathrm{A{*}A = A} \\
\mathrm{A{+}A = A} \\
\bar{A}\mathrm{{*}A = 0} \\
\overline{A * B} = \bar{A}{+}\bar{B} \\
\overline{A + B} = \bar{A}{*}\bar{B}
\end{gathered}$$

Durante il processo di generazione, i cut sets aventi una probabilità di accadimento o una frequenza al di sotto di definiti limiti probabilistici vengono eliminati. Probabilità di accadimento e frequenza di guasto di un cut set vengono identificate dalle seguenti formule:

$$Q_{cut} = \prod_{i=1}^{n} Q_i \tag{2.38}$$

dove

- Q_i = indisponibilità del componente i-esimo del cut set
- Q_{cut} = probabilità di occorrenza del cut set
- n = numero di eventi nel cut set

e

$$\omega_{cut} = \sum_{j=1}^{n} \omega_j \prod_{i \neq j}^{n} Q_i \tag{2.39}$$

dove

- ω_j = frequenza di guasto per l'evento j-esimo del cut set
- ω_{cut} = frequenza di occorrenza del cut set

Partendo da queste formule è possibile calcolare gli output sopra elencati del sistema preso in esame con diversi metodi. Di seguito si pone l'attenzione sui tre metodi più utilizzati: Esary-Proschan, Cross-Product e Rare Approximation.

Esary-Proschan

L'indisponibilità e e la frequenza di guasto del sistema sono calcolate con il metodo Esary-Proschan attraverso le seguenti espressioni

$$Q_{SYS} = \prod_{i=1}^{m} Q_i [1 - \prod_{j=1}^{n} (1 - Q_{cutj})] \tag{2.40}$$

dove

- Q_i = indisponibilità dell'evento comune i-esimo
- m = numero di eventi comuni che accadono in tutti i cut sets
- Q_{cutj} = indisponibilità del j-esimo cut set escludendo gli eventi in comune
- n = numero di cut sets
- Q_{SYS} = indisponibilità del sistema

e

$$\omega_{SYS} = \sum_{i=1}^{n} \omega_{cuti} \prod_{j \neq i}^{n} (1 - Q_{cutj}) \tag{2.41}$$

Cross-Product

L'indisponibilità ottenibile dal metodo Cross-Product è data da

$$Q_{SYS}(t) = \sum_{i=1}^{n} Q_{cuti}(t) - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Q_{ij}(t) + ... + (-1)^{n+1} Q_{123...n}(t) \tag{2.42}$$

dove $Q_{ij}(t)$ = prodotto delle indisponibilità degli eventi basici nei cut sets i e j.

Rare Approximation

L'espressione per calcolare l'indisponibilità del sistema utilizzando la Rare Approximation è pari a

$$Q_{SYS} = \sum_{i=1}^{n} Q_{cuti} \tag{2.43}$$

mentre la frequenza di guasto del sistema

$$\omega_{SYS} = \sum_{i=1}^{n} \omega_{cuti} \tag{2.44}$$

2.5.4 Calcolo parametri di sistema

Altri parametri legati ai sistemi vengono calcolati grazie alle seguenti espressioni:

Total system Down Time (TDT_{SYS})

$$TDT_{SYS} = \int_0^T Q_{SYS}(t)dt \tag{2.45}$$

Conditional Failure Intensity (CFI)

$$\lambda_{SYS} = \frac{\omega_{SYS}}{1 - Q_{SYS}} \tag{2.46}$$

Indisponibilità del sistema

$$F_{SYS} = 1 - e^{-\int_0^T \lambda_{SYS}(t)dt} \tag{2.47}$$

System MeanTime To Failure ($MTTF_{SYS}$)

$$MTTF_{SYS} = \int_0^\infty R(t)dt \tag{2.48}$$

System Mean Time Between Failure ($MTBF_{SYS}$)

$$MTBF_{SYS} = \frac{1}{\omega(\infty)} \tag{2.49}$$

System Mean Time To Repair ($MTTR_{SYS}$)

$$MTBF_{SYS} = \frac{Q(\infty)}{\omega(\infty)} \tag{2.50}$$

System Mean Unavailability ($\bar{Q}_{SYS}$)

$$\bar{Q}_{SYS} = \frac{TDT_{SYS}}{T} \tag{2.51}$$

2.5.5 Importance Measures

Uno degli output principali derivante dall'analisi FT è rappresentato da un set di importance measures che sono legate al top event e stabiliscono l'importanza in termini di impatto sul sistema complessivo e di probabilità di accadimento che ha ogni singolo componente. Sia gli eventi intermedi che quelli basici possono essere catalogati in base alla loro priorità legata alla loro importanza. Queste misurazioni sono molto utili per definire la sensibilità del top event alla variazione degli eventi nell'albero dei guasti, oltre ad essere un importante strumento per determinare le aree deboli all'interno del sistema. Storicamente si evince da queste misurazioni che il numero di eventi che pesano maggiormente sulla probabilità di accadimento del top event sono solamente un 20% di quelli presenti in un albero dei guasti, i quali però contribuiscono al 90% alla probabilità di accadimento di guasto dell'intero sistema; esistono quattro tipi base di Importance Measures che vengono descritte di seguito.

Fussell-Vesely (F-V)

Definisce in che quantità gli eventi incidano sulla probabilità di accadimento del top event; questa misurazione è applicabile a tutti gli eventi che costituiscono il sistema, fornendone così un significato numerico e rendendone possibile il loro ordinamento in base alla priorità [7]. La misurazione F-V è legata al parametro di indisponibilità Q ed è realizzata sottraendo a tutte le cause (minimal cut sets) del top event (Q_{SYS}) quelle del top event stesso senza il contributo del componente i-esimo considerato ($Q_{SYS}(q_i = 0)$); la percentuale di guasto a cui contribuisce l'evento i-esimo è quindi data da

$$I_i^{FV} = \frac{Q_{SYS} - Q_{SYS}(q_i = 0)}{Q_{SYS}} \tag{2.52}$$

dove

- I_i^{FV} = misurazione F-V per l'evento i-esimo
- Q_{SYS} = probabilità del sistema o rischio
- $Q_{SYS}(q_i = 0)$ = probabilità di sistema o rischio con la probabilità di accadimento dell'evento i-esimo=0 (il componente si guasta)

Risk Reduction Worth (RRW)

Misura la riduzione della probabilità di accadimento del top event quando si assume che un determinato evento sicuramente non accadrà. E' anche definita come massima riduzione possibile del rischio ed è l'inverso della misurazione Fussell-Vesely

$$I_i^{RRW} = \frac{Q_{SYS}}{Q_{SYS}(q_i = 0)} \tag{2.53}$$

Risk Achievement Worth (RAW)

Misura l'aumento della probabilità di accadimento del top event quando si assume che un determinato evento accadrà. Questa misurazione mostra dove le attività di prevenzione dovrebbero concentrarsi per assicurarsi che non si verifichino guasti. Solitamente la RAW viene calcolata rimodellando l'albero dei guasti in base alla formula del minimal cut sets

$$I_i^{RAW} = \frac{Q_{SYS}(q_i = 1)}{Q_{SYS}} \tag{2.54}$$

Birnbaum's Importance Measure (BM)

Misura la variazione del tasso di probabilità di accadimento del top event in relazione a quella di un dato evento ed è data dalla formula

$$I_i^{BB} = \frac{\delta Q_{SYS}}{\delta q_i} \tag{2.55}$$

dove

- I_i^{BB} = misurazione di Birnbaum per il componente i-esimo
- Q_{SYS} = indisponibilità del sistema
- q_i = indisponibilità del componente i-esimo

Barlow-Proschan

Rappresenta la probabilità che il sistema si guasti in seguito al guasto di un componente critico. La misurazione di Barlow-Proschan è data dalla formula

$$I_i^{BP} = \frac{\sum_{j=1}^{n} \omega_i Q_{cutj}}{\omega_{SYS}} \tag{2.56}$$

dove

- ω_i = frequenza di guasto del componente i-esimo
- Q_{cutj} = somma delle indisponibilità dei cut sets contenenti l'evento
- ω_{SYS} = frequenza di guasto del sistema

Capitolo 3

Realizzazione di una FTA basata su Matlab e Simulink

Nel seguente capitolo verrà posta l'attenzione sull'obiettivo principale di questo lavoro di tesi, ossia la realizzazione di un programma che permetta di eseguire un'analisi degli alberi dei guasti che fornisca dei risultati fedeli il più possibile a quelli che si otterrebbero utilizzando un software commerciale professionale. Il programma (o tool) è stato realizzato grazie all'utilizzo dei software Matlab e Simulink, sui quali verrà fatta una rapida panoramica delle caratteristiche principali nei successivi paragrafi.

3.1 Introduzione a Matlab

Matlab, sviluppato dalla software house MathWorks, è uno tra i tanti programmi commerciali che forniscono sofisticati strumenti di calcolo scientifico, come ad esempio Maple, Mathematica e MathCad [9]. Nonostante le affermazioni dei responsabili commerciali, nessuno di questi programmi può essere considerato "il migliore". Ciascuno di loro presenta punti di forza e debolezze: tutti permettono all'utente l'esecuzione di operazioni matematiche di base, ma differiscono nella gestione del calcolo simbolico e dei processi matematici complessi, come la gestione delle matrici.
Matlab eccelle nella manipolazione delle matrici, come sottolineato dall'acronimo MATLAB che deriva da MATrix LABoratory ("laboratorio matriciale") mentre, ad esempio, Maple eccelle nel calcolo simbolico. A livello base, tutti questi programmi sono considerabili al pari di una sofisticata calcolatrice scientifica: possono infatti svolgerne i compiti, ma rappresentano sicuramente molto di più. Se infatti si ha un PC a portata di mano è probabile che si preferisca usare Matlab, invece della calcolatrice, per eseguire anche i conti più semplici, come ad esempio il bilancio delle spese personali; in molti settori dell'ingegneria, tuttavia, eseguire calcoli con programmi di questo tipo consente di sostituire la programmazione tradizionale. Questo non significa che l'apprendimento di un linguaggio di programmazione ad alto livello come C++ o Fortran non sia necessario, ma piuttosto che programmi come Matlab siano ormai diventati strumenti comuni per ingegneri e scienziati.
La semplicità d'uso di Matlab è tale da consentire lo svolgimento di molti compiti di programmazione, anche se non è sempre il migliore strumento per risolvere problemi di questo tipo. Il programma eccelle nel calcolo numerico, specie nell'uso delle matrici e nella grafica, ma certamente nessuno scriverebbe un programma di elaborazione dei testi in Matlab. Linguaggi come C++ o Fortran sono adatti allo svolgimento di compiti generici e costituiscono la scelta migliore per un impiego nell'ambito di sistemi operativi e programmi complessi (non a caso Matlab, che è un programma complesso, è scritto originariamente in Fortran e solo più tardi convertito in C, il precursore del C++). D'altra parte, i linguaggi di programmazione ad alto livello in genere non offrono strumenti per la grafica, applicazione nella quale, viceversa,

Matlab primeggia. L'area di applicazione principale che accomuna Matlab ai linguaggi ad alto livello è il number crunching ("sgranocchiare numeri"): l'esecuzione di calcoli ripetitivi oppure l'elaborazione di grandi quantità di dati. Sia Matlab sia i linguaggi ad alto livello sono appropriati per il number crunching e, in genere, anche se lo stesso programma potrebbe essere eseguito più velocemente se scritto in C++ o Fortran, è più facile creare programmi di elaborazione numerica in Matlab. Un'eccezione a questa regola è rappresentata dal calcolo con le matrici: poiché Matlab è ottimizzato per il calcolo matriciale, se il problema può essere espresso in tale forma, Matlab lo risolverà più rapidamente rispetto a un programma simile scritto in linguaggio ad alto livello.

In Matlab non è necessario dichiarare esplicitamente all'inizio del lavoro una variabile in termini delle sue dimensioni e del tipo dei suoi coefficienti (interi, reali, complessi). Inoltre come ulteriore comodità è già predefinito un ampio insieme di matrici elementari, come la matrice identità, quella nulla ecc.... Sono previsti vari operatori algebrici di uso comune fra matrici, quali ad esempio somma, prodotto, elevamento a potenza, nonché il calcolo del determinante o del rango di una matrice. Sono inoltre predefinite numerose funzioni di uso generale dette built-in functions. Esse permettono di risolvere problemi complessi, ad esempio il calcolo degli autovettori ed autovalori di una matrice, la risoluzione efficiente di sistemi lineari, oppure la ricerca degli zeri di una funzione.
Le raccolte di funzioni dedicate ad uno specifico argomento vengono dette toolboxes. La finanza, la statistica, l'analisi dei segnali e delle immagini sono alcuni dei campi a cui sono dedicati dei toolboxes di Matlab.

Per lanciare Matlab è sufficiente cliccare col mouse sull'icona corrispondente. All'avvio il desktop Matlab contiene le seguenti finestre:

- Command window (finestra dei comandi)
- Workspace window (finestra dello spazio di lavoro)
- Current directory (finestra della directory corrente)
- Command history (finestra della storia dei comandi)

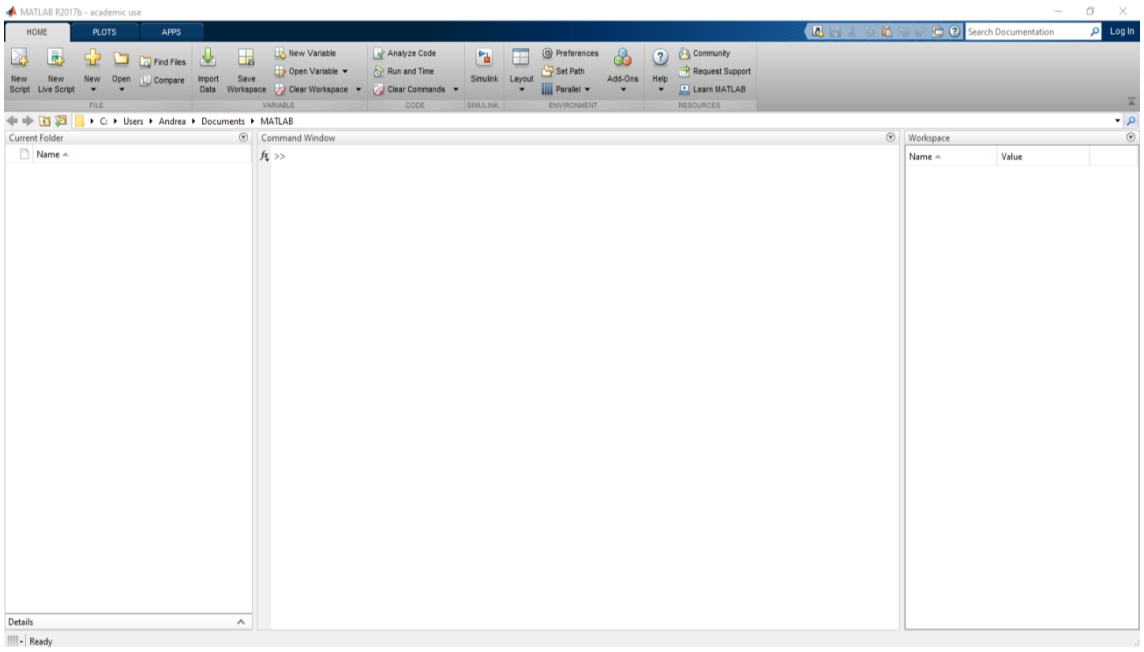

Figura 3.1: Desktop Matlab

Nella Command window vengono digitati i comandi Matlab, come la valutazione di un'espressione, l'assegnazione di un valore ad una variabile e l'esecuzione di una funzione. Le istruzioni vengono immesse al prompt (seguite da invio) ed eseguite una ad una. La finestra Workspace elenca le variabili presenti nello spazio di lavoro insieme ad alcune informazioni su di esse (dimensioni, tipo, memoria occupata...); è possibile modificare il valore di una variabile utilizzando l'apposito workspace array editor. La finestra Command History contiene una lista dei comandi digitati con funzioni di copia e incolla, mentre tramite la Current Directory è possibile spostarsi tra le cartelle come con un qualsiasi File Manager.
Un comando fondamentale, soprattutto per chi è alle prime armi, è il tasto "Help" che, digitato su linea di comando seguito dalla funzione di cui si vogliono conoscere le caratteristiche, fornisce tutte le indicazioni di contenuto e sintattiche che svolge data funzione (ad esempio, *help sin* fornisce tutte le informazioni utili sulla funzione seno).
Tutte le informazioni (dati in forma numerica o alfanumerica) digitate in Matlab vengono memorizzate nella RAM utilizzando quelle che vengono chiamate nei linguaggi di programmazione "variabili". Una variabile ha due caratteristiche: il nome che la identifica e il valore, cioè il dato che essa rappresenta e che viene memorizzato nella RAM.

3.2 Introduzione a Simulink

Il software Simulink è un applicativo che "vive" all'interno del programma di calcolo Matlab e ne costituisce, in buona sostanza, una potente e intuitiva interfaccia grafica che ne semplifica grandemente l'impiego da parte dell'utente [10].
Lo scopo principale di Simulink è quello di modellare, simulare e analizzare i sistemi dinamici. Mediante questo software è possibile programmare l'esecuzione di calcoli in ambiente Matlab in maniera molto più rapida ed error-free rispetto alla scrittura dei lunghi e complessi m-files (estensione con cui vengono creati e salvati gli script di Matlab) che sono necessari, ad esempio, per programmare in Matlab l'integrazione numerica di un sistema di equazioni differenziali di ordine elevato. Mediante gli strumenti "visuali" disponibili in ambiente Simulink è possibile simulare dei sistemi anche molto complessi con uno sforzo da parte dell'utente che si limita al tracciamento, su un foglio di lavoro elettronico, di uno schema a blocchi rappresentativo del sistema in esame.
Simulink si compone di una serie di librerie che contengono dei blocchi elementari i quali, opportunamente interconnessi, andranno a realizzare lo schema a blocchi che rappresenta la funzionalità desiderata. Per accedere alla lista delle librerie è necessario avviare Matlab e selezionare la finestra Simulink, dalla quale si otterrà la seguente schermata

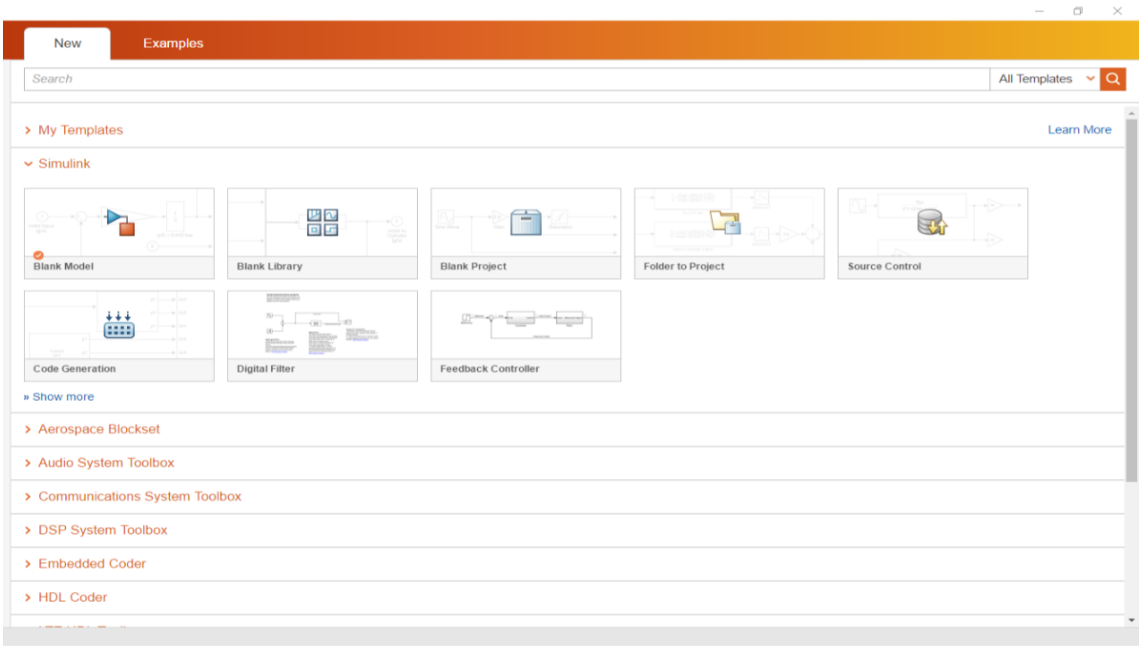

Figura 3.2: Schermata di avvio di Simulink

dalla quale sarà possibile aprire la finestra delle librerie di Simulink

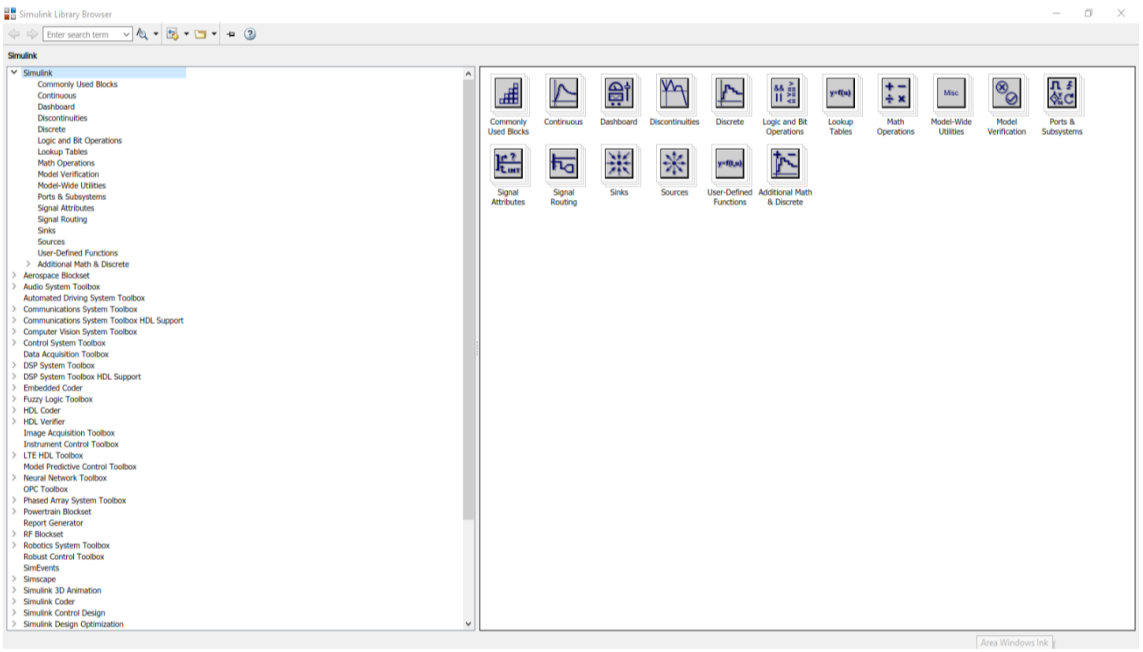

Figura 3.3: Lista delle librerie di Simulink

Si può notare l'elenco delle librerie (Continuous, Discrete, Functions & Tables, Math ecc...) ognuna delle quali contiene una certa tipologia di componenti base, come ad esempio:

Libreria	**Contenuto**
Continuous	Componenti lineari a tempo continuo
Discrete	Componenti lineari a tempo discreto
Functions & Tables	Funzionalità pre-programmabili e look-up tables
Math	Funzioni matematiche
Nonlinear	Componenti non lineari
Signals & Systems	Condizionamento di segnali
Sinks	Visualizzatori di segnale
Sources	Generatori di forme d'onda

Tabella 3.1: Contenuto di alcune librerie di Simulink

Ciascun blocco elementare presente nelle librerie svolge una particolare funzione, ad esempio, nella libreria "Sources" possiamo trovare il blocco "Step" che genera un comando a gradino, o quello "Ramp" che ne genera uno a rampa.

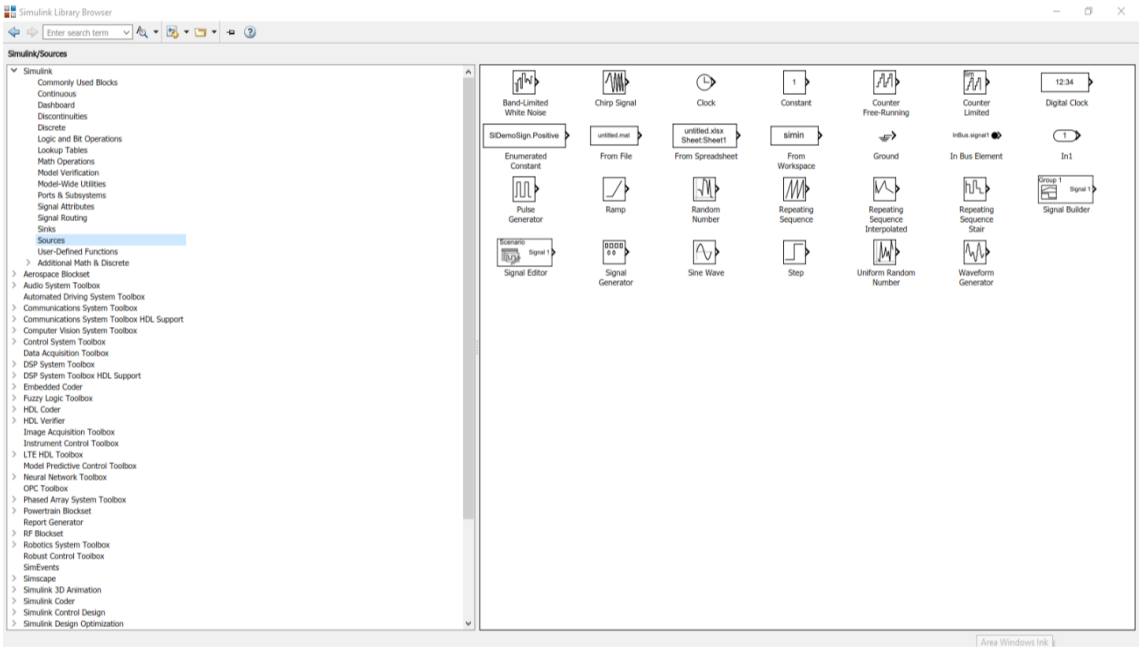

Figura 3.4: Blocchi libreria Sources

Una volta definito il modello, questo può essere immediatamente simulato (senza compilazione). La simulazione dei modelli Simulink richiede l'integrazione numerica di equazioni differenziali ordinarie (ODE) e c'è la possibilità di scegliere solutori a passo variabile (ode45, ode23, ode 113) o fisso.
Anche se il modello che si vuole analizzare può essere definito interamente nell'ambiente Simulink, questa scelta fornisce una soluzione molto limitata e rigida. E' preferibile dividere il modello fra due ambienti: nel codice Matlab si definiscono i valori dei parametri, delle condizioni iniziali e i parametri della simulazione (ad esempio tempo finale, precisione), mentre nello schema Simulink si definisce il modello come schema a blocchi o come S-function (che rappresenta una descrizione del sistema preso in esame mediante linguaggio di programmazione) scritta in codice Matlab o in C

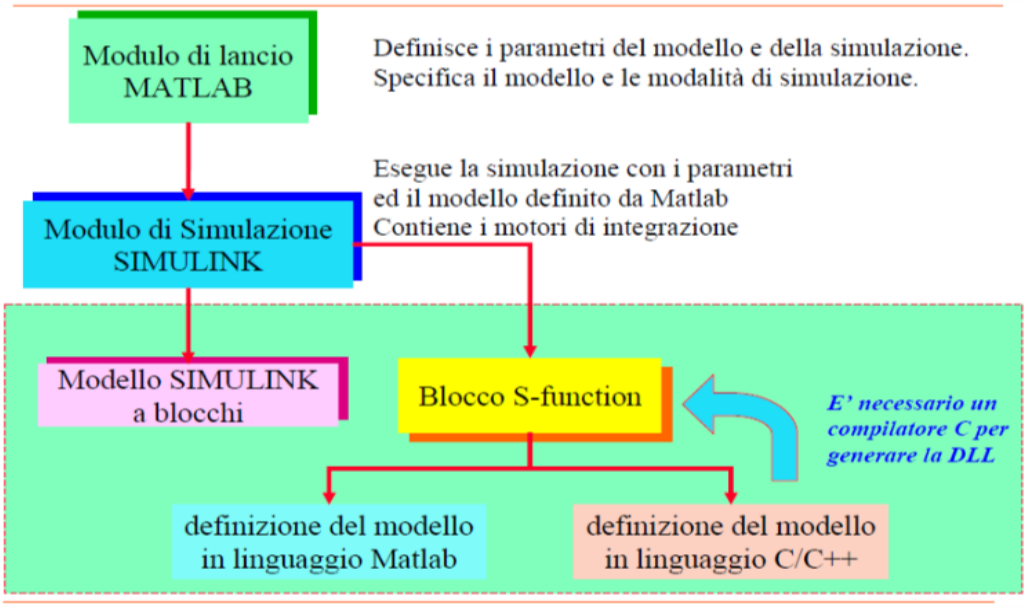

Figura 3.5: Simulazione di un sistema dinamico

3.3 Obiettivo della tesi

Le introduzioni a Matlab e Simulink appena fatte sono di grande importanza, in quanto questi sono i due software utilizzati per la realizzazione della tesi. Il loro utilizzo per il raggiungimento del risultato voluto può essere definito "non convenzionale", soprattutto per quanto riguarda Simulink: infatti il programma non è stato utilizzato con lo scopo di realizzare difficili studi riguardanti sistemi dinamici (sfruttandone quindi le caratteristiche per cui nasce originariamente), ma piuttosto come un potente supporto grafico in grado di rappresentare alberi di guasto attraverso l'inserimento di blocchi di eventi collegati tra di loro da porte logiche. All'interno degli script Matlab, legati ai blocchi della libreria creata in Simulink illustrata ed analizzata in dettaglio nei paragrafi successivi, sono state inserite le funzioni e le equazioni relative ad ogni blocco rappresentante eventi o porte logiche.
L'obiettivo della tesi è stato quindi quello di realizzare un software in grado di effettuare un'analisi degli alberi dei guasti fornendo risultati simili a quelli prodotti dal programma commerciale leader del settore, FaultTree+ della casa Isograph.

3.3.1 FaultTree+

FaultTree+ è un programma appartenente alla suite Reliability Workbench di Isograph in grado di sviluppare analisi legate ad affidabilità e disponibilità sia di sistemi semplici che complessi. Reliability Workbench è un ambiente integrato nel quale è possibile realizzare analisi, tra le altre, degli alberi dei guasti (Fault Tree Analysis), degli alberi degli eventi (Event Tree Analysis) ed analisi Markov. Il programma è ricco di funzionalità e può modellare ed analizzare un grandissimo numero di possibili scenari; la sua facilità di utilizzo lo rendono un software in grado di essere utilizzato anche da utenti inesperti senza difficoltà, e la chiarezza dei comandi unita all'efficace rappresentazione grafica dell'albero dei guasti lo rendono uno dei programmi più utilizzati a livello globale per questo tipo di analisi. Il programma presenta, tra le altre, le seguenti funzioni:

- Interfaccia intuitiva che permette una facile creazione di fault tree, event tree e diagrammi di Markov

- Algoritmi per la generazione di minimal cut sets
- Scelta dei metodi di risoluzione quantitativi, quali Rare Approximation ed Esary-Proschan
- I modelli di Markov possono essere integrati nell'analisi di guasto dei singoli componenti
- Analisi dei CCF, importance measures e fasi di missione
- Modelli di guasto basati sulla normativa IEC 61508
- Risultati personalizzabili e salvabili in formato Word o PDF
- Integrazione con l'ambiente Excel

Modelli di Markov personalizzati possono inoltre essere collegati ai singoli eventi del diagramma; l'analisi Markov fornisce un mezzo per analizzare l'affidabilità e la disponibilità di sistemi i cui componenti presentano forti dipendenze, mentre altri metodi di analisi dei sistemi, come ad esempio quello della Teoria Cinetica degli alberi, generalmente assumono che i componenti siano tra di loro indipendenti, ipotesi che porta ad ottenere risultati ottimistici di affidabilità e disponibilità del sistema preso in esame. I moduli di Markov permettono all'utente di costruire facilmente diagrammi di stati di transizione ed effettuare integrazioni numeriche per problemi su sistemi complessi.
Entrando nel dettaglio di FaultTree+, questo programma ha la capacità di analizzare alberi dei guasti caratterizzati da molte diramazioni, producendo correttamente i minimal cut sets per il top event; il software fornisce inoltre la possibilità di determinare l'effetto sul sistema considerato dei Common Cause Failures (CCF), oltre a fornire l'impatto che ogni singolo componente ha sul sistema. I dati di input solitamente richiesti dal programma sono:

- Nome dell'evento
- Modello che appartiene al singolo evento (Fixed, Constant Rate...)
- Failure rate (λ)
- MTTR
- Test interval (τ)
- Unavailability Q (se il componente è di tipo Fixed)
- Failure frequency ω (se il componente è di tipo Fixed)

Il programma permette all'utente di generare un singolo database del progetto contenente i dati necessari all'analisi, alberi di guasto con più top events, tabelle dei CCF; l'impaginazione dell'albero è automaticamente controllata. L'utente può inoltre copiare intere diramazioni di eventi ed inserirle in altre zone dell'albero dei guasti, in modo da non dover riscrivere eventi già presenti. Un esempio di finestra di lavoro di FaultTree+ è mostrata nella seguente figura.

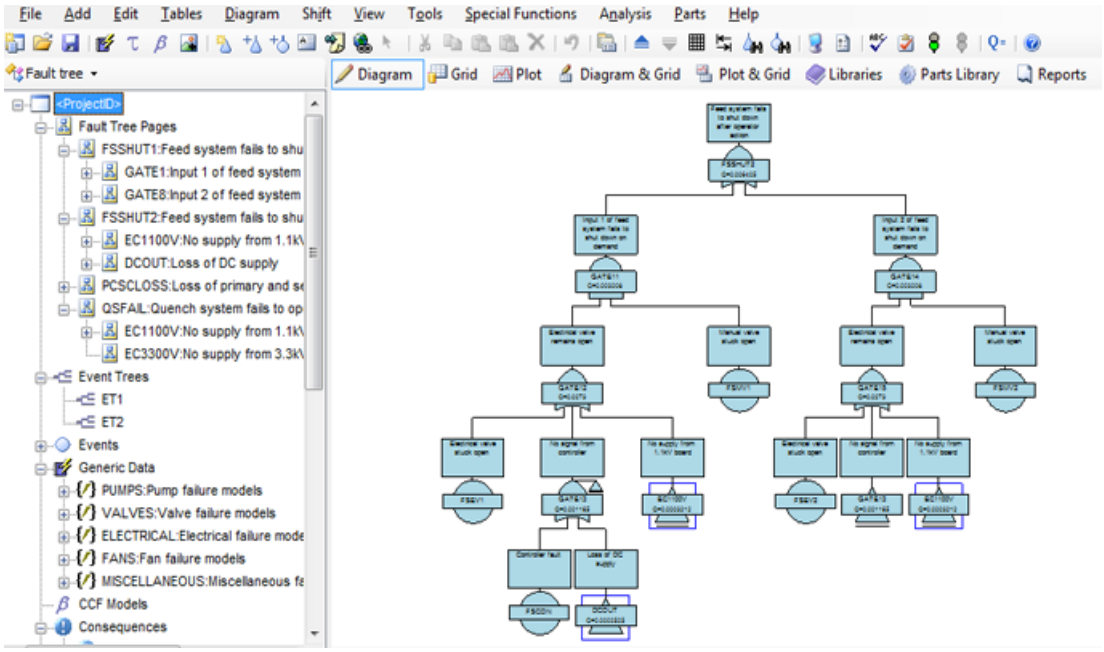

Figura 3.6: Finestra di lavoro FaultTree+

Il comando *Report Generator* permette all'utente di personalizzare come meglio crede ogni albero dei guasti, modificando la struttura dei blocchi rappresentanti i singoli eventi, il font dei caratteri, inserendo figure, filtri per i dati ed altro ancora. I risultati vengono forniti in forma tabellare e sono facilmente esportabili in fogli di lavoro come Microsoft Excel; è inoltre possibile la generazione di grafici al fine di rendere più chiari gli output ottenuti.

3.4 Introduzione al tool realizzato

Tenendo a mente quello che è stato descritto nel paragrafo precedente è ora possibile esaminare approfonditamente le caratteristiche del programma realizzato in Simulink e Matlab nel corso della tesi che permette l'analisi degli alberi di guasto in maniera molto simile a FaultTree+. E' necessario specificare sin dall'inizio che il programma realizzato è un primo prototipo e, benché funzionante correttamente, fornisce i risultati essenziali di una fault tree analysis, comunque sufficienti per testare la bontà del lavoro svolto attraverso una serie di test che verranno descritti nei prossimi capitoli.

3.4.1 Realizzazione della libreria

La prima operazione svolta è stata quella di creare una libreria in Simulink apposita che comprendesse tutti i blocchi necessari alla realizzazione di alberi di guasto, in quanto molti dei tipici blocchi utilizzati per questo tipo di analisi non sono presenti di default nel software di Mathworks.

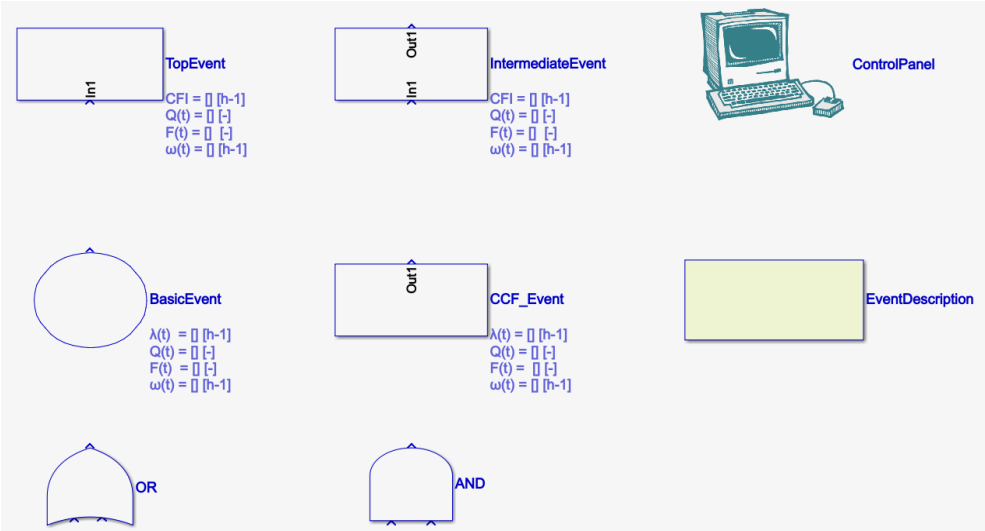

Figura 3.7: Libreria in Simulink tool per Fault Tree Analysis

Al fine di costruire un albero dei guasti, è necessario combinare insieme i diversi blocchi presenti in libreria; in quella creata, sono stati realizzati i seguenti:

- Basic event
- Blocco Common Cause Failure (CCF)
- Blocco Intermediate event
- Blocco Event description
- Blocco Top event
- Porta logica AND
- Porta logica OR
- Control Panel

Nei paragrafi successivi verrà preso in considerazione ed analizzato ogni singolo blocco creato ed inserito nella libreria appositamente creata.

3.4.2 Control Panel

Il pannello di controllo può essere definito come il "cervello" che gestisce ogni albero dei guasti che viene creato in Simulink. Attraverso questo blocco è possibile, oltre a svolgere la maggior parte delle azioni legate al calcolo dei vari parametri del sistema preso in esame, automatizzare molte delle operazioni che normalmente l'utente deve svolgere manualmente.

Figura 3.8: Icona del pannello di controllo

Cliccando sopra l'icona, si accede al menu principale del pannello di controllo, dal quale è possibile svolgere diverse operazioni, quali:

- Il nome del modello Simulink creato compare automaticamente all'interno di *Simulink Model* e permette di avviare l'analisi dell'albero dei guasti
- Scegliere, attraverso *Success Criteria*, quale dei parametri di output, tra Q(t), F(t) ed $\omega(t)$ mostrare come output principale, con la possibilità inoltre di inserire una breve descrizione
- Far partire, attraverso il tasto *Start Propagation*, l'analisi dell'albero dei guasti fino ad ottenere i risultati del Top Event una volta inseriti i dati di input nei Basic Events
- Se si volesse effettuare una nuova analisi con lo stesso albero, attraverso il pulsante *Clean Model* è possibile eliminare automaticamente tutti i dati contenuti nel Top Event, negli Intermediate Events, nelle porte logiche e nei Basic Events, evitando inutili perdite di tempo che si avrebbero svolgendo l'operazione manualmente

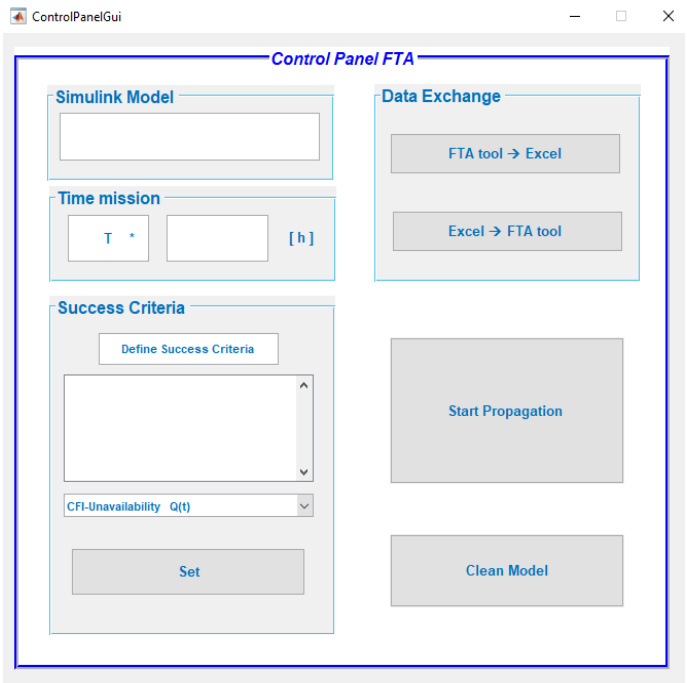

Figura 3.9: Menu pannello di controllo

Come ulteriore opzione è possibile far generare un foglio Excel contenente tutte le informazioni contenute nei Basic Events e negli Intermediate Events semplicemente premendo il tasto *FTA tool–>Spreadsheet*. L'utente potrà così avere a portata di mano in un formato universalmente utilizzato in ambito industriale quale è Excel i valori forniti dal tool, senza dover aprire ogni volta le schermate dei diversi eventi. Un'ulteriore funzione implementata è quella di poter caricare tramite il comando *Spreadsheet–>FTA tool* direttamente in un nuovo modello Simulink i Basic Events immessi in precedenza all'interno di un template Excel, con i rispettivi valori di input. In questo modo il tempo di lavoro legato alla creazione di un albero dei guasti si riduce notevolmente. Verrà quindi generato un nuovo modello contenente i blocchi Basic Events nel numero definito dal template ed insieme a questi anche il blocco pannello di controllo.

BASIC EVENTS										
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]
BasicEvent										
BasicEvent1										
BasicEvent2										

Figura 3.10: Template Excel per Basic Events

Lo user può scegliere direttamente, tramite menu a tendina, il modello di componente (*Fixed, Constant Rate*) ed il suo tipo (*Repairable, UnRepairable, Tested*). La scelta di realizzare un template Excel che si interfacci con il programma Matlab realizzato rientra in quella che oggi viene conosciuta come Model Automation, ossia la tendenza da parte di molte aziende a realizzare strumenti indipendenti dai programmi di calcolo utilizzati normalmente, che permettano l'inserimento in modo automatico di una grande mole di dati, riducendo così di molto i tempi necessari allo svolgimento delle operazioni di caricamento e calcolo dei valori.
Un appunto deve essere fatto per quel che riguarda i cosiddetti Criteri di Successo: solitamente in una Fault Tree Analysis l'utente ha come obiettivo quello di calcolare solamente uno dei parametri di output che normalmente il programma realizzato calcola: per evitare la generazione di errori dovuti ai troppi valori che vengono forniti come risultato, e soprattutto per rendere più chiara l'interfaccia grafica del tool, si è deciso di dare all'utente la possibilità di scegliere quale parametro intende calcolare, scegliendolo direttamente dal pannello di controllo e facendo apparire questa scelta in tutti i blocchi dei diversi eventi nella cella *Success Criteria*.

3.4.3 Basic event

Come facilmente intuibile, questo blocco permette di inserire gli eventi primari che caratterizzano un albero dei guasti. I Basic Events, così come gli altri blocchi, sono già stati analizzati approfonditamente nel capitolo 2, ma è utile ricordare che questi possono essere considerati come le radici di ogni albero: infatti, ognuno di essi non è a sua volta scomponibile in altri eventi.

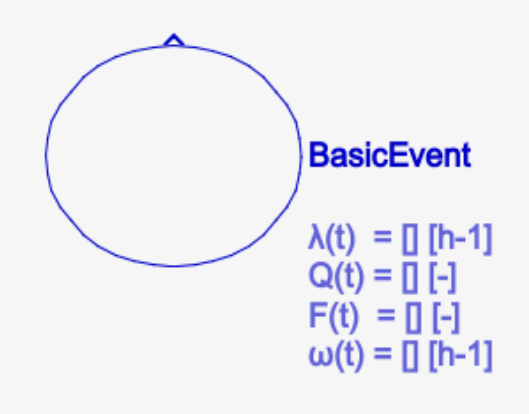

Figura 3.11: Icona blocco Basic Event

Aprendo un nuovo modello in Simulink, è possibile inserire il blocco al suo interno semplicemente trascinandolo dalla libreria, oppure inserire il numero di Basic Events voluti con i rispettivi valori di input direttamente da template Excel. Cliccando sull'icona apparirà una schermata nella quale sarà possibile inserire i dati di input necessari (se non già presenti) per il calcolo dell'indisponibilità Q(t), della inaffidabilità F(t) e della frequenza di guasto $\omega(t)$ dell'evento. In caso si inserisse il blocco direttamente dalla libreria Simulink, è necessario scegliere il modello con cui analizzare l'evento, in base al fatto che questo sia del tipo *Fixed unavailability and failure frequency* o *Constant failure and repair rate*.

Fixed unavailability and failure frequency model

Scegliendo come opzione dal menu a tendina *Model* il modello *Fixed*, la schermata del basic event apparirà con il seguente layout.

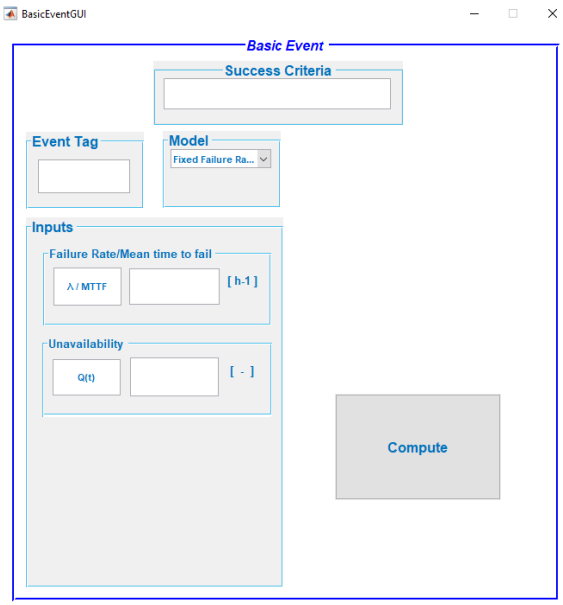

Figura 3.12: Schermata Fixed model

Il fixed model è un modello per il quale si prevede che l'utente conosca già come dati di input il tasso di guasto $\lambda(t)$, l'indisponibilità Q(t), l'inaffidabilità F(t) e la frequenza di guasto ω(t) del componente, che saranno quindi inseriti direttamente negli appositi campi. Prima di chiudere la finestra sarà necessario assegnare il nome con cui identificare l'evento

nell'apposito spazio *Event Tag*: questo sarà l'identificativo di quello specifico evento base, utilizzato successivamente dal programma stesso per definire l'equazione di ogni cut set e del minimal cut set del Top Event. Se l'utente non vi inserisce l'etichetta, il programma ne genererà una automaticamente. A questo punto sarà possibile premere il tasto *Compute* e chiudere la finestra di dialogo del Basic Event: i valori verranno salvati in Simulink, con la possibilità di essere modificati ogni volta che l'utente vorrà riaprendo semplicemente la schermata del blocco.

Constant Failure and Repair Rate model

Se il componente da analizzare presenta un tasso di guasto costante nel tempo, quindi non è affetto da guasti infantili o di vecchiaia, l'inaffidabilità, l'indisponibilità e la frequenza di guasto non saranno più parametri di input ma risultati ottenuti dalle equazioni presentate nel capitolo 2 nel paragrafo riguardante il Constant Failure and Repair Rate model. La finestra di dialogo del Basic Event definito con questo modello è la seguente.

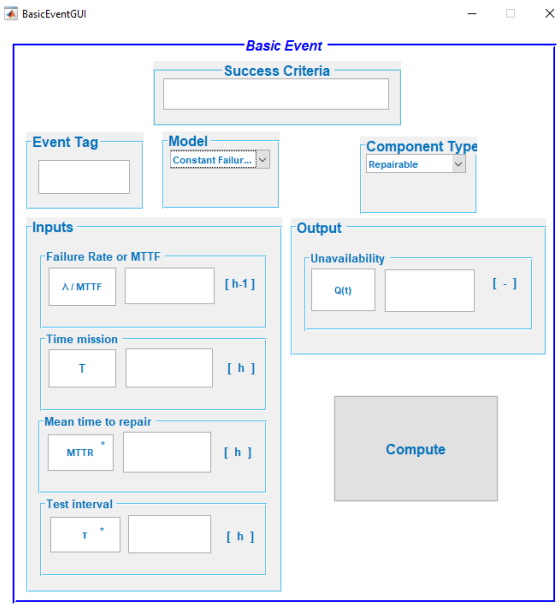

Figura 3.13: SchermataConstant Failure and Repair Rate model

L'utente dovrà scegliere la tipologia del componente che il programma analizzerà attraverso il menu a tendina *Component Type* a seconda che questo sia

- Riparabile (repairable)
- Non riparabile (NonRepairable)
- Testato (Tested/Dormant)

Come nel modello Fixed, è necessario assegnare ad ogni evento un'etichetta che lo identifichi nella casella *Event Tag*; il programma utilizzerà questa proprietà per definire i successivi cut sets dell'albero.
I dati di input necessari sono:

- Failure rate λ o MTTF [h^{-1}]
- Mean Time To Repair MTTR [h]
- Tempo di missione T [h]

- Test interval τ [h]

Deve essere fatto un appunto sul Mean Time To Repair, in quanto il valore di questo parametro non sempre viene fornito; in questi casi il programma, se non specificato diversamente dall'utente, impone come valore di input 1, che secondo letteratura è spesso caratterizzante l'MTTR.
Una volta inseriti i valori di input e premuto il tasto *Compute*, il programma fornirà a schermata i risultati caratterizzanti l'evento, servendosi delle equazioni presenti in appositi script su Matlab. Una volta chiusa la finestra di dialogo è possibile notare che le informazioni principali, ossia la tag dell'evento, λ, Q(t), F(t) ed ω(t) vengono stampati a video di fianco al blocco stesso, così da permettere all'utente un'immediata lettura.

3.4.4 Porte logiche

La libreria presenta al suo interno le due principali porte logiche della fault tree analysis, la porta AND e la porta OR. Nel presente elaborato si è deciso di non sviluppare ulteriori porte secondarie, come quelle exclusive OR o priority AND, in quanto solitamente meno utilizzate rispetto alle prime due, essendo inoltre l'obiettivo principale quello di realizzare una prima soluzione prototipale di cui si voleva effettuare una validazione prima di ulteriori sviluppi.

Figura 3.14: Icone porte AND e OR

Non è stato necessario creare i blocchi, in quanto questi sono già presenti di default nella libreria di Simulink. Le porte logiche possono avere in input due o più eventi basici o intermedi ma sempre un solo output. Il numero di ingressi di una porta logica può essere impostato direttamente dall'utente tramite la schermata che comparirà una volta selezionata col mouse la porta stessa

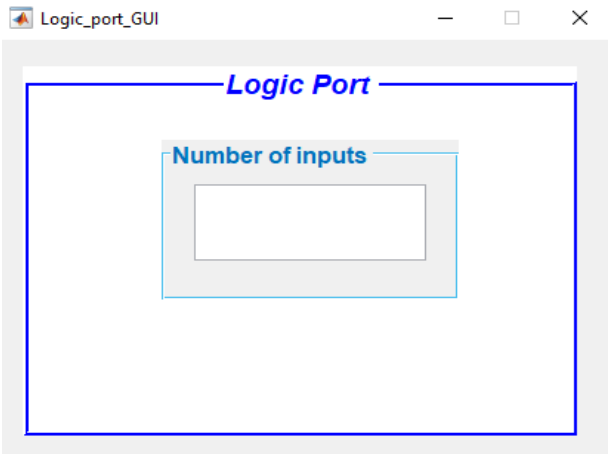

Figura 3.15: Inserimento numero ingressi porta logica

La forma della porta si modificherà graficamente in base al numero di ingressi selezionati. Nel programma le porte logiche svolgono la funzione di contenitori di informazioni: infatti queste si limitano a salvare nel loro campo *Description* tutte le informazioni (nome dell'evento e output dello stesso) dei diversi eventi che ricevono in ingresso, ordinati su diverse righe

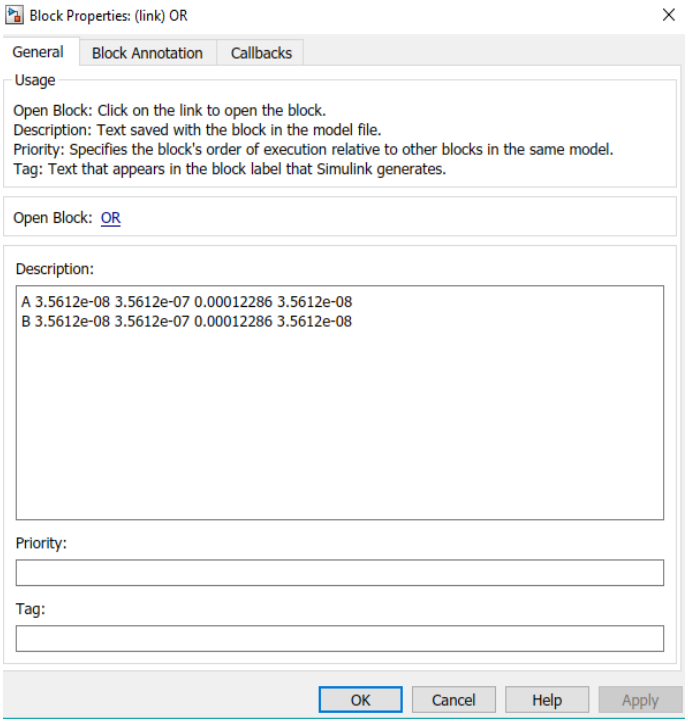

Figura 3.16: Campo Description porta logica

Queste informazioni verranno inviate direttamente all'Intermediate Event del livello superiore collegato alla porta logica (per definizione ogni porta è collegata direttamente ad un evento intermedio o al Top Event).

3.4.5 Common Cause Failure

La definizione di Common Cause Failure è stata già trattata nel capitolo 2, pertanto di seguito verrà descritto ed analizzato solamente il funzionamento del relativo blocco, spiegando la sua importanza nel calcolo dei parametri che caratterizzeranno l'analisi dell'albero dei guasti.

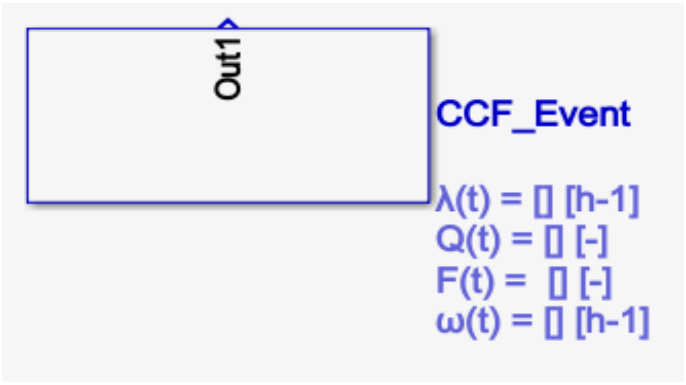

Figura 3.17: Icona blocco CCF

Componenti diversi che costituiscono un sistema possono guastarsi a causa di uno stesso fenomeno: i relativi blocchi che li rappresentano devono quindi tenere conto di questa causa comune, che rende inutile il concetto di ridondanza. La presenza di cause comuni di guasto modifica in concreto valori come tasso di guasto λ e indisponibilità del componente, andando di conseguenza a cambiare i valori finali del sistema analizzato; per calcolarli è stato necessario creare appositamente un blocco nella libreria Simulink. Una volta inserito nel modello contenente gli eventi basici affetti da Common Cause Failure, il blocco presenta la seguente interfaccia:

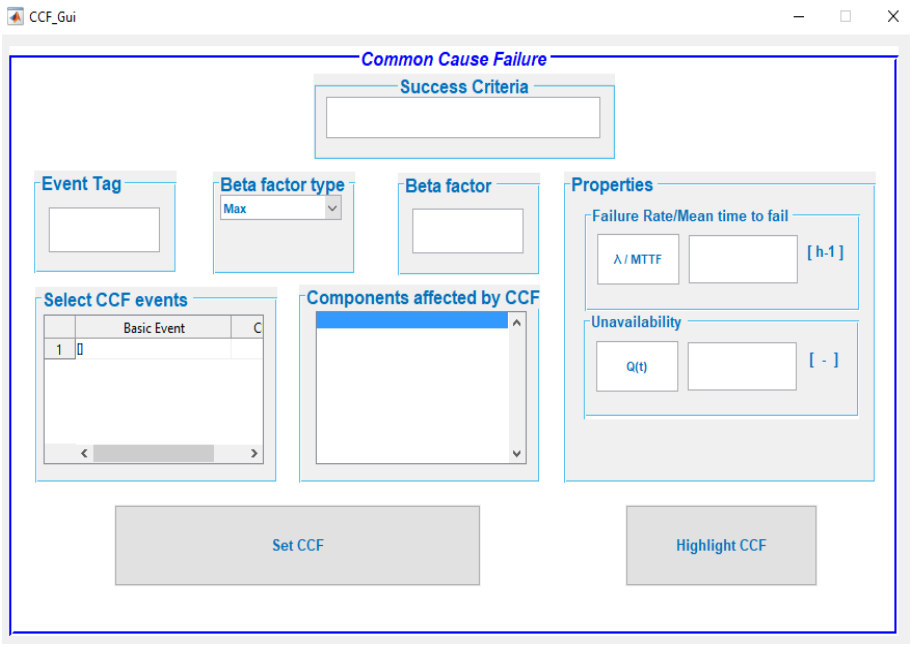

Figura 3.18: Schermata di interfaccia blocco CCF

All'interno della check box *Select CCF Events* saranno presenti tutti i Basic Events che caratterizzano l'albero: è possibile selezionare quelli che sono affetti da CCF i quali, una volta premuto il comando *Set CCF*, saranno riportati nella cella *Components affected by CCF*. Lo user dovrà decidere che tipo di analisi effettuare, a seconda del tipo di β-factor che vuole utilizzare (min, average, max); questa scelta andrà a caratterizzare il risultato dei valori che il blocco andrà a calcolare, ossia λ, Q, F, ω: selezionando l'opzione β-factor max, lo script Matlab facente capo al blocco CCF sceglierà il valore massimo di questi valori tra i Basic Events, con l'average quello medio e con min il minimo tra tutti. Solitamente per le analisi viene utilizzato il β-factor average, il quale risulta essere meno conservativo rispetto a quello max. Una volta inserito il β-factor, le funzioni del blocco andranno a modificare i parametri direttamente dei Basic Events che si vogliono calcolare, di fianco ai quali verranno stampati a schermo: in *Properties* verranno mostrati i valori di output calcolati che saranno utilizzati per il calcolo finale dell'albero. Infine, attraverso il comando *Highlight CCF*, saranno evidenziati di colore diverso, direttamente nel modello Simulink, quei blocchi caratterizzati da Common Cause Failure, per poter fornire all'utente un immediato riscontro visivo.

3.4.6 Intermediate Event

Questi elementi sono dati dall'unione, attraverso le porte logiche, di diversi eventi base e/o altri eventi intermedi dei livelli inferiori dell'albero dei guasti. Sono caratterizzati da un ingresso, ossia la porta logica dalla quale ricevono le informazioni, e da un'uscita, collegata alla porta logica del livello superiore.

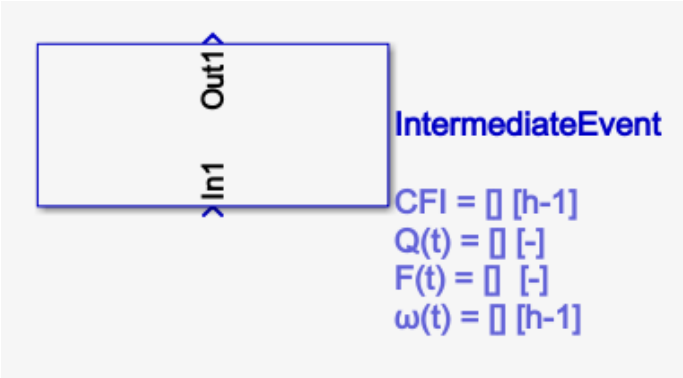

Figura 3.19: Icona blocco evento intermedio

Il blocco, grazie alla stesura di un apposito script in Matlab, interroga la porta logica in ingresso e ne riconosce la natura (se è di tipo AND o OR); in questo modo è possibile applicare le formule matematiche corrette per calcolare i parametri di output del cut set che si sta valutando, ossia CFI_{Cut} (Conditional Failure Intensity), $Q_{Cut}(t)$, $F_{Cut}(t)$ ed $\omega_{Cut}(t)$ che vengono successivamente stampati a schermo come per i Basic Events.

3.4.7 Blocco di testo

Per una più chiara ed efficace interpretazione dell'albero dei guasti è stato inserito nella libreria Simulink un blocco di testo che, una volta posizionato di seguito agli eventi basici e intermedi, permette di inserire la descrizione dell'evento di guasto.

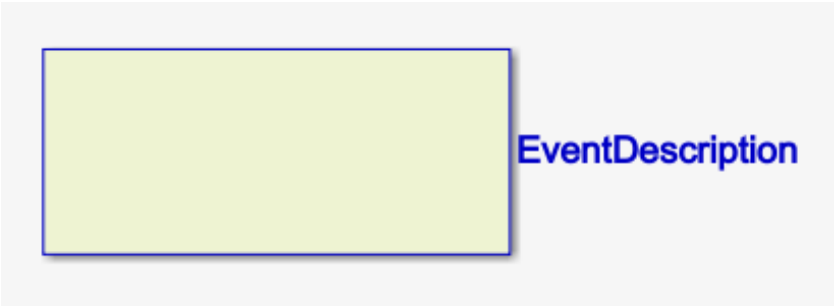

Figura 3.20: Blocco di testo

Una volta trascinato il blocco e collegato all'evento a cui si riferisce, è sufficiente cliccarci sopra due volte per far aprire la finestra di dialogo in cui inserire il testo

Figura 3.21: Schermata blocco testo

Nella cella *Failure mode* è possibile inserire l'identificativo dell'evento a cui il blocco di testo viene assegnato; è possibile inserire una spiegazione più approfondita del tipo di guasto nella casella di testo sottostante. Una volta premuto il tasto *Edit* e chiusa la finestra, il testo inserito in *Failure mode* apparirà direttamente sul blocco; questo permette all'utente di avere sempre sotto controllo i diversi eventi inseriti.

3.4.8 Top event

Ogni albero dei guasti presenta, alla sua sommità, un blocco rappresentante il Top Event, ossia quell'evento che definisce il guasto dell'intero sistema preso in considerazione. Nella libreria Simulink è stato necessario realizzare un blocco apposito, dotato di un ingresso ma ovviamente senza un'uscita verso ulteriori blocchi.

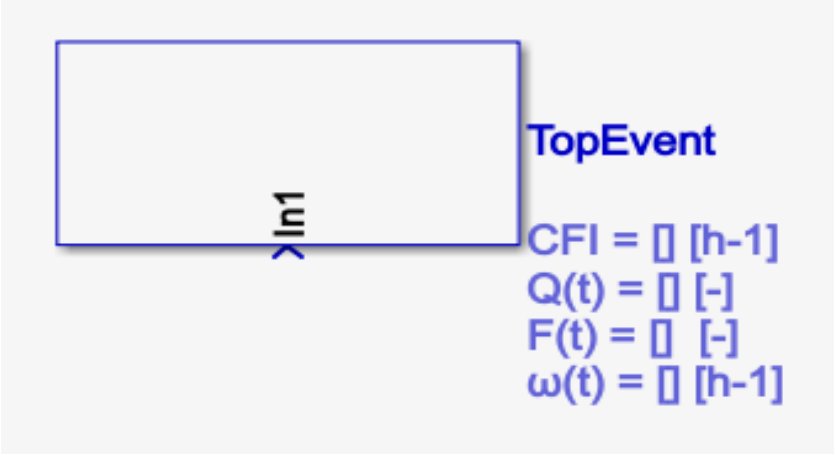

Figura 3.22: Icona blocco Top Event

Il blocco riceve in input le informazioni relative agli eventi sottostanti attraverso la porta logica alla quale è collegato e, utilizzando il metodo Rare Approximation (scelto come default in questo elaborato), fornisce come risultati la Conditional Failure Intensity del sistema CFI_{SYS}, l'indisponibilità complessiva Q_{SYS}, la sua inaffidabilità F_{SYS} e frequenza di guasto ω_{SYS}. Inoltre, appoggiandosi a determinate funzioni elaborate in script di Matlab, il blocco

permette all'utente di ottenere, premendo semplicemente il tasto *Evaluate MCS* dell'apposita schermata, l'equazione del Top Event già nella forma di minimal cut set.

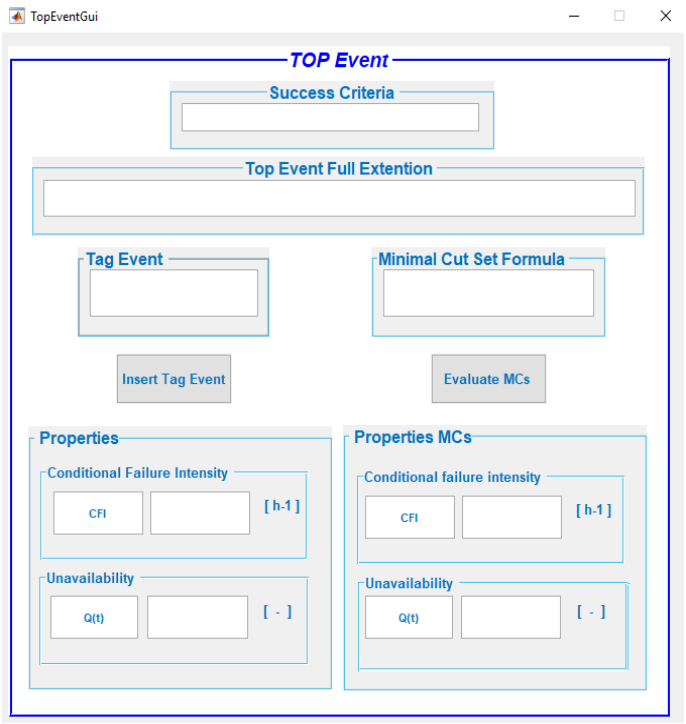

Figura 3.23: Schermata Top Event

Come già per i Basic e gli Intermediate Events, anche per il Top Event le informazioni principali vengono stampate a schermo per una più immediata lettura da parte dell'utente.

Capitolo 4

Esempio pratico

Dopo aver introdotto e descritto le funzionalità dei blocchi che costituiscono la struttura principale del programma, è ora possibile costruire un albero dei guasti vero e proprio. In questo capitolo verrà presentato un esempio pratico di Fault Tree Analysis, realizzata mediante FaultTree+: essendo Teoresi un'azienda che opera principalmente in ambito ferroviario, l'esempio considerato riguarda l'analisi della sicurezza sulla conformità ai requisiti della normativa TSI LOCO & PASS 1302/2014 legata al sistema di porte esterne del treno NTV - ETR675 di ALSTOM (Italo alta velocità).

4.1 Sistema porte esterne: FaultTree+

Il capitolato ALSTOM nel quale viene analizzato il sistema di porte esterne del treno NTV - ETR675 ha come titolo "Analisi di sicurezza sulla conformità ai requisiti della TSI LOCO & PASS 1302/2014 in accordo al Regolamento (UE) 402/2013 modificato dal Regolamento (UE) 2015/1136".

Lo scopo del documento è quello di dimostrare, attraverso l'analisi del rischio, che il sistema porte esterne dell'unità a composizione bloccata ETR675 per l'operatore ferroviario NTV rispetti i relativi requisiti di sicurezza riportati nei paragrafi 4.2.5.5.8 e 4.2.5.5.9 della TSI LOCO & PASS 1302/2014, ai sensi del Regolamento (UE) 402/2013 modificato dal Regolamento (UE) 2015/1136 [11].

4.1.1 Descrizione del sistema oggetto dell'analisi

Nell'unità a composizione bloccata ETR675 il Sistema Porte Esterne è fornito da IFE [12] ed è integrato con componenti del banco di manovra di fornitura SPII ed altri componenti elettrici direttamente definiti da ALSTOM Ferroviaria S.p.A (AF). I veicoli 1, 2, 3, 6 e 7 di ogni unità ETR675 sono dotati di 2 porte esterne ciascuno (una per lato), i veicoli 4 e 5 di 4 porte esterne (2 per lato).

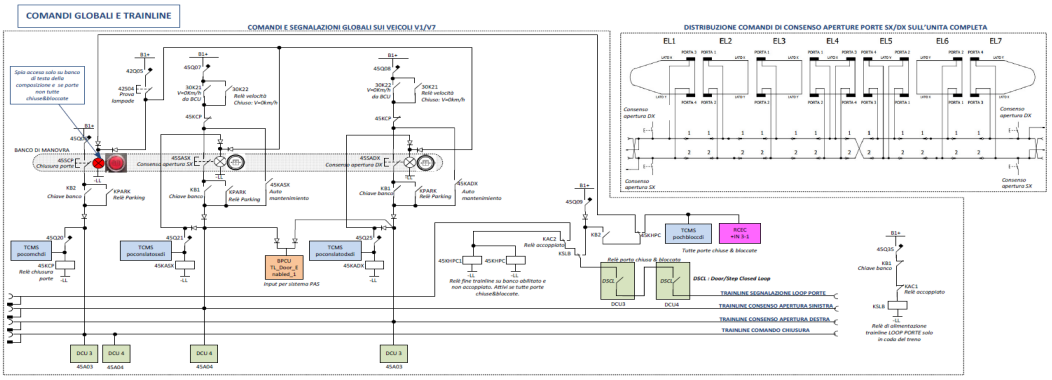

Figura 4.1: Comandi globali e trainline

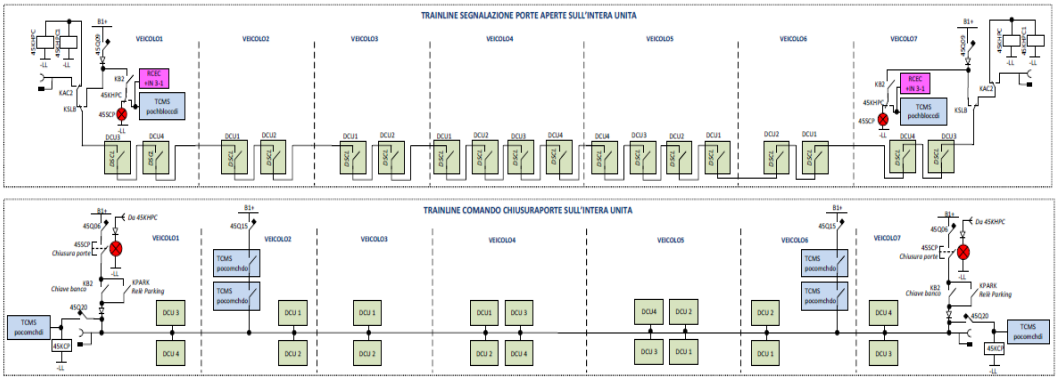

Figura 4.2: Trainline porte sull'intera unità

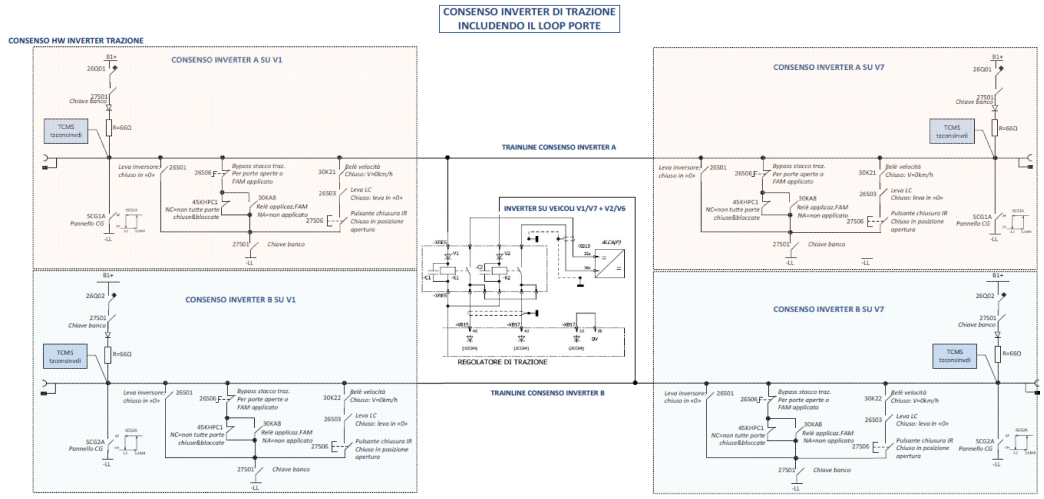

Figura 4.3: Consenso inverter di trazione includendo il loop porte

In condizioni di funzionamento normale, affinché le porte esterne possano essere aperte, è necessario che l'unità sia ferma (segnale V_0=1, ovvero velocità<3 km/h attivo e velocità>10 km/h inattivo) e che uno dei due segnali consenso apertura porte (destro e sinistro) sia attivo. Nonostante ciò, per aprire le singole porte è comunque necessario premere il relativo pulsante di apertura posizionato nelle vicinanze della porta stessa (e che la porta non sia isolata). Per comandare la chiusura di tutte le porte è invece necessario che il personale di condotta (PdC) prema il pulsante chiusura porte posizionato sul banco di manovra (azione

che, oltre a comandare la chiusura delle porte, causa la caduta del segnale consenso apertura porte). Per come è stato realizzato il circuito di consenso inverter, per poter trazionare sarà quindi necessario che tutte le porte risultino chiuse e bloccate (oppure isolate) in modo da alimentare tramite filo treno il relè 45KHPC1 (figure 4.1, 4.2, 4.3) che, attraverso i suoi contatti normalmente chiusi (NC), mantenga attivo il segnale "consenso inverter"; in altre parole, se una porta risulta aperta (o sbloccata), il filo treno "segnalazione porte aperte" viene interrotto, quindi il relè 45KHPC1 viene disalimentato in modo che i suoi contatti sui due fili treno "consenso inverter" si chiudano mettendo a massa i due segnali e togliendo il consenso agli inverter di trazione.
Sul banco di manovra realizzato da SPII sono presenti i seguenti componenti coinvolti nella gestione del sistema porte esterne:

- Pulsante consenso apertura porte DX, 45SADX
- Pulsante consenso apertura porte SX, 45SASX
- Pulsante chiusura porte con relativa spia porte aperte, 45SCP

Gli altri componenti necessari al funzionamento del sistema e direttamente definiti da AF sono:

- Relè 45KCP di chiusura porte: se eccitato tramite pulsante chiusura porte fa cadere il segnale di consenso apertura porte tramite i suoi contatti sulle linee di alimentazione dei segnali consenso apertura porte DX e SX. Si trova nel quadro QCA in cabina di guida e fa parte di una scheda a 5 relè
- Relè 45KASX e 45KADX: vengono eccitati all'attivazione dei relativi pulsanti monostabili 45SASX e 45SADX, mantenendo il segnale attivo anche al rilascio dei medesimi pulsanti. Entrambi si trovano nel quadro QCA in cabina di guida e fanno parte di una scheda a 3 relè
- Relè KSLB: normalmente alimentato in coda al treno, viene disalimentato in seguito all'inserimento della chiave di banco (quindi in testa): se alimentato chiude il contatto che alimenta il filo treno di segnalazione porte aperte, se disalimentato il contatto va ad alimentare i relè 45KHPC e 45KHPC1. Si trova nel quadro QCA in cabina di guida e fa parte di una scheda a 3 relè
- Relè 45KHPC: se alimentato segnala che tutte le porte sono chiuse e bloccate spegnendo la spia rossa 45SCP sul banco di manovra; se disalimentato la spia rossa si accende segnalando che almeno una porta è aperta. Si trova nel quadro QCA in cabina di guida e fa parte di una scheda a 5 relè
- Relè 45KHPC1: se alimentato tramite i suoi contatti mantiene i due fili treno consenso inverter a 1, diversamente li mette a negativo togliendo il consenso alla trazione. Si trova nel quadro QCA in cabina di guida e fa parte di una scheda a 5 relè
- Contatti 30K21 e 30K22: se chiusi (V<3km/h) rendono possibile alimentare i segnali consenso apertura DX e SX; si trovano trova nel quadro QCA in cabina di guida e fanno parte di una scheda a 5 relè

4.1.2 Analisi di Sicurezza

La normativa TSI LOCO & PASS 1302/2014 descrive nel paragrafo 6.2.3.5 [11] come effettuare la valutazione di conformità per i propri requisiti di sicurezza. Nello specifico, è ammissibile l'applicazione di un criterio armonizzato di accettazione del rischio associato alla

gravità specificata nella TSI, da effettuarsi applicando le disposizioni dell'allegato I, punto 3, del metodo comune di sicurezza per la valutazione del rischio. Il regolamento (CE) n. 352/2009 della Commissione è stato però sostituito dal Regolamento (UE) 402/2013, a sua volta modificato dal Regolamento (UE) 2015/1136. Riassumendo quanto contenuto in questi regolamenti, applicare il metodo comune per la sicurezza in essi descritti per gli scenari di rischio previsti dalla TSI significa adottare una delle seguenti metodologie di verifica:

- Similarità con altro sistema o sistemi di riferimento
- Applicazione di codici di buona pratica
- Applicazione di una stima esplicita del rischio, ed in questo caso:
  - Se le conseguenze dello scenario di rischio sono state valutate come catastrofiche (più persone coinvolte nell'incidente, possibile morte di più persone), si dovrà dimostrare che la frequenza di accadimento dell'evento indesiderato sia "altamente improbabile", ovvero minore o uguale a 10^{-9} eventi per ora di funzionamento
  - Se le conseguenze dello scenario di rischio sono state valutate come critiche (poche persone coinvolte nell'incidente e al più un morto), si dovrà dimostrare che la frequenza di accadimento dell'evento indesiderato sia "improbabile", ovvero minore o uguale a 10^{-7} eventi per ora di funzionamento

Requisiti di sicurezza

Per le porte esterne, nei capitoli 4.2.5.5.8. e 4.2.5.5.9. della TSI LOCO & PASS 1302/2014 sono riportati i seguenti requisiti di sicurezza:

- 4.2.5.5.8.(1): per lo scenario "una porta non è bloccata (con personale di bordo non correttamente informato di questo stato della porta) o è sbloccata o aperta in zone (ad esempio, sul lato sbagliato del treno) o situazioni (ad esempio, treno in movimento) inopportune", è necessario dimostrare che il rischio sia tenuto sotto controllo ad un livello accettabile, considerando il fatto che un'avaria funzionale presenta in genere un potenziale notevole di provocare "perdita di una singola vita umana e/o lesioni gravi" nel caso in cui i passeggeri dovessero stare in piedi nella zona antistante la porta (lunga distanza)
- 4.2.5.5.8.(2): per lo scenario "diverse porte non sono bloccate (con personale di bordo non correttamente informato di questo stato delle porte) o sono sbloccate o aperte in zone (ad esempio, sul lato sbagliato del treno) o situazioni (ad esempio, treno in movimento) inopportune", è necessario dimostrare che il rischio sia tenuto sotto controllo ad un livello accettabile, considerando il fatto che un'avaria funzionale presenta in genere un potenziale notevole e diretto di provocare "perdita di una singola vita umana e/o lesioni gravi" nel caso in cui i passeggeri dovessero stare in piedi nella zona antistante la porta (lunga distanza)
- 4.2.5.5.9.(4): per lo scenario "avaria del sistema interno di apertura di emergenza di due porte adiacenti lungo un percorso diretto (quale definito al punto 4.2.10.5 della TSI) restando disponibile il sistema di apertura di emergenza delle altre porte", è necessario dimostrare che il rischio sia tenuto sotto controllo ad un livello accettabile, considerando il fatto che un'avaria funzionale presenta in genere un potenziale notevole di provocare "perdita di una singola vita umana e/o lesioni gravi"

Si è pertanto deciso di procedere alla dimostrazione di conformità eseguendo una stima esplicita dei tre rischi verificando che la loro frequenza di accadimento sia "improbabile", ovvero minore o uguale a 10^{-7} eventi per ora di funzionamento.

Analisi di conformità ai requisiti di sicurezza

La prima fase di analisi di conformità ai due requisiti è stata svolta da IFE che, in base all'architettura del sistema, ha realizzato le seguenti FTA di sicurezza limitatamente ai suo scopo di fornitura:

- FTA 010 "Door opens while train in motion". La porta si apre mentre l'unità è in movimento, evento che corrisponde ad un tasso di guasto di 4.16^{-15} eventi/ora
- FTA 020 "Door opens at standstill while not enabled to open". La porta si apre a unità ferma quando non abilitata ad aprirsi, evento che corrisponde ad un tasso di guasto di 4.16^{-15} eventi/ora
- FTA 410 "Interior manual emergency device fails". Il dispositivo di aperture di emergenza interno non funziona, evento che corrisponde ad un tasso di guasto di 2.72^{-15} eventi/ora

Sulla base di queste analisi si possono quindi analizzare i tre scenari previsti dalla TSI.

Una porta non è bloccata o è sbloccata o aperta in zone o situazioni inopportune con personale di bordo non correttamente informato di questo stato delle porte

Lo scenario previsto dal requisito 4.2.5.5.8.(1) corrisponde al manifestarsi di almeno uno degli eventi analizzati da IFE con le FTA 010 e 020. Il tasso di guasto relativo allo scenario previsto dalla TSI è quindi approssimabile con la somma dei due tassi di guasto degli eventi analizzati da IFE, ovvero:

$$4{,}16*10^{-15} + 4{,}16*10^{-15} = 8{,}32*10^{-15} \text{ eventi/ora}$$

Minore del target di 10^{-7} eventi per ora di funzionamento. È bene sottolineare che il suddetto calcolo è conservativo, infatti il tasso di guasto effettivo dovrebbe essere più piccolo, in quanto ci sono sicuramente dei componenti il cui guasto porta ad entrambi gli scenari previsti da IFE e perciò sommare i due tassi di guasto comporta conteggiare due volte queste avarie.

Diverse porte non sono bloccate o sono sbloccate o aperte in zone o situazioni inopportune con personale di bordo non correttamente informato di questo stato delle porte

Lo scenario previsto dal requisito 4.2.5.5.8.(2) corrisponde al manifestarsi contemporaneo di almeno uno dei seguenti eventi:

1. Manifestarsi contemporaneo delle varie eventualità contemplate dallo scenario "Una porta bloccata o è sbloccata o aperta in zone o situazioni inopportune" in più di una porta
2. Abilitazione indebita di almeno una delle due linee treno di consenso apertura porte. Data l'architettura Hardware descritta dallo schema di principio (fig. 4.3):
    - Perché l'abilitazione indebita del consenso apertura porte si manifesti con l'unità in corsa si devono in contemporanea avere:
        - o persistenza del segnale velocità a 1 (V<3km/h) di uno dei due contatti 30K21 o 30K22 rimasto incollato chiuso
        - o pulsante consenso apertura porte (SX o DX) incollato chiuso o contatto 45KASX (o 45KADX) di auto-mantenimento incollato

- Perché l'abilitazione indebita del consenso apertura porte si manifesti con l'unità ferma basta invece o il pulsante consenso apertura porte (SX o DX) incollato chiuso o il contatto 45KASX (o 45KADX) di auto-mantenimento incollato

3. Mancata inibizione del consenso inverter nonostante una o più porte siano rimaste aperte. Data l'architettura Hardware descritta dallo schema di principio questo corrisponde ad avere uno dei due contatti 45KHPC1 sui due fili treno consenso inverter bloccato aperto

4. Mancata accensione della spia di banco 45SCP che segnala che almeno una porta è sbloccata dovuto o a 45KHPC incollato aperto o a spia di banco 45SCP guasta

La probabilità di accadimento dell'evento "Diverse porte non sono bloccate o sono sbloccate o aperte in zone o situazioni inopportune con personale di bordo non correttamente informato di questo stato delle porte" è valutabile attraverso lo studio del seguente albero dei guasti.
!!Molto probabilmente questi grafici verranno sostituiti con gli stessi ma aventi più cifre significative dopo la virgola per una maggiore precisione di calcolo!!

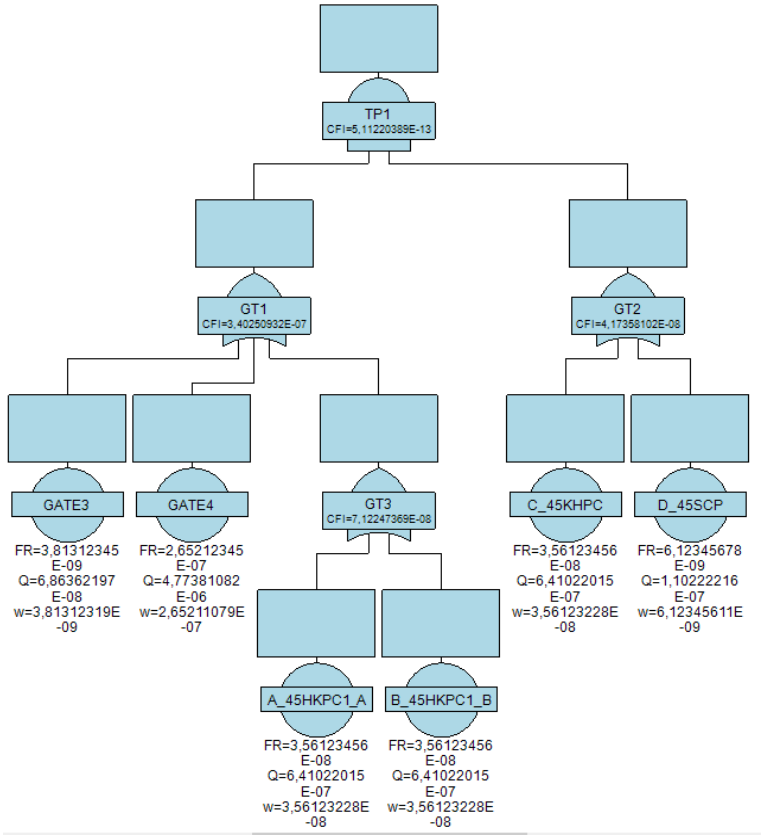

Figura 4.4: Probabilità di accadimento

Come si può notare, si ha che la probabilità di accadimento è pari a $5{,}112*10^{-13}$ eventi per ora di funzionamento.
In seguito sono state realizzate altre simulazioni legate allo stesso albero (figure 4.5, 4.6, 4.7), al fine di valutare l'indisponibiltà, l'inaffidabilità e la frequenza di guasto.

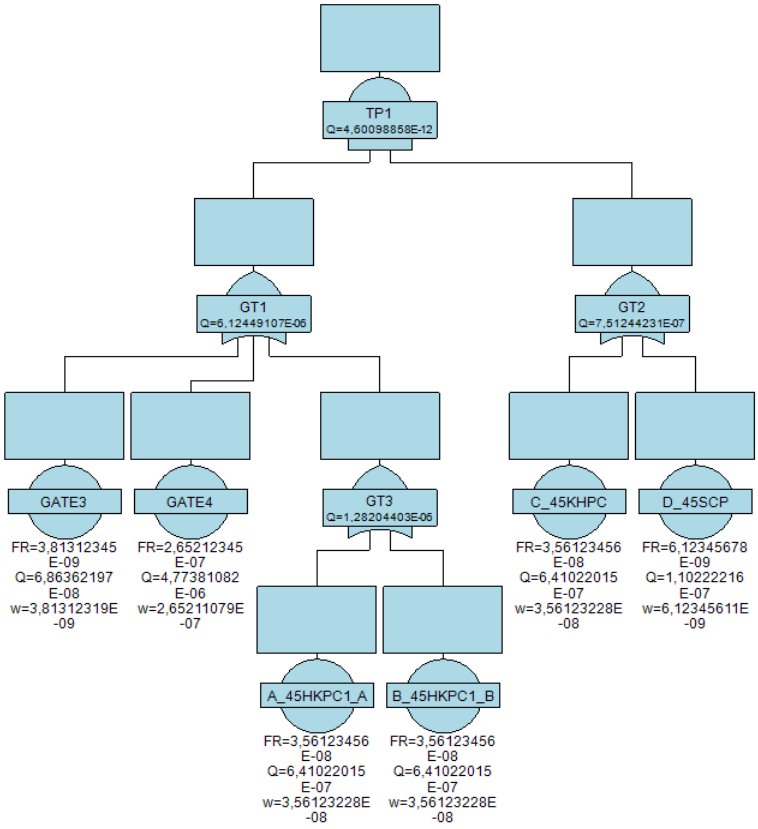

Figura 4.5: Indisponibilità

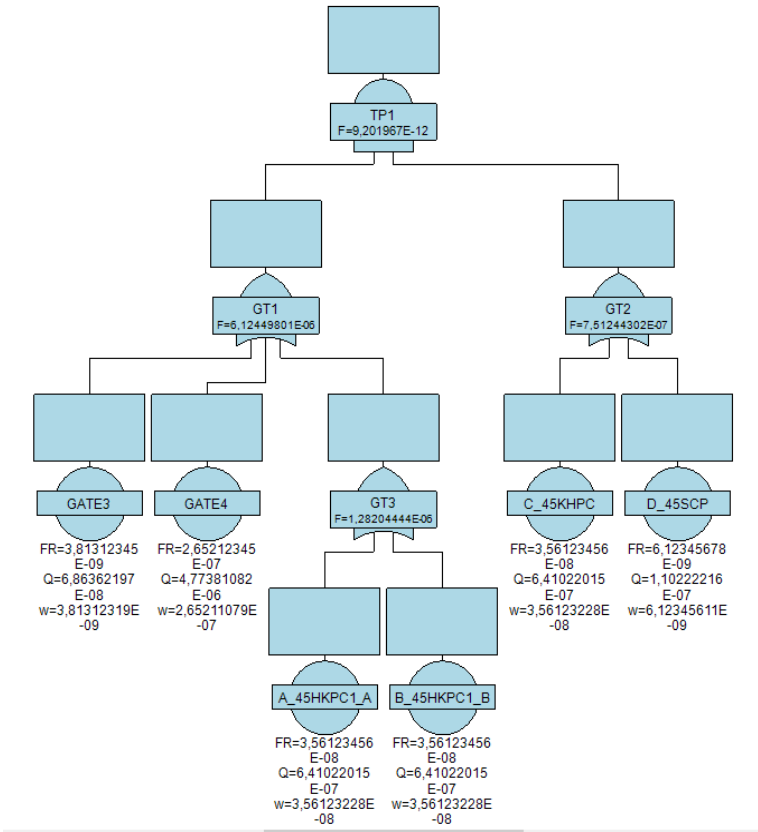

Figura 4.6: Inaffidabilità

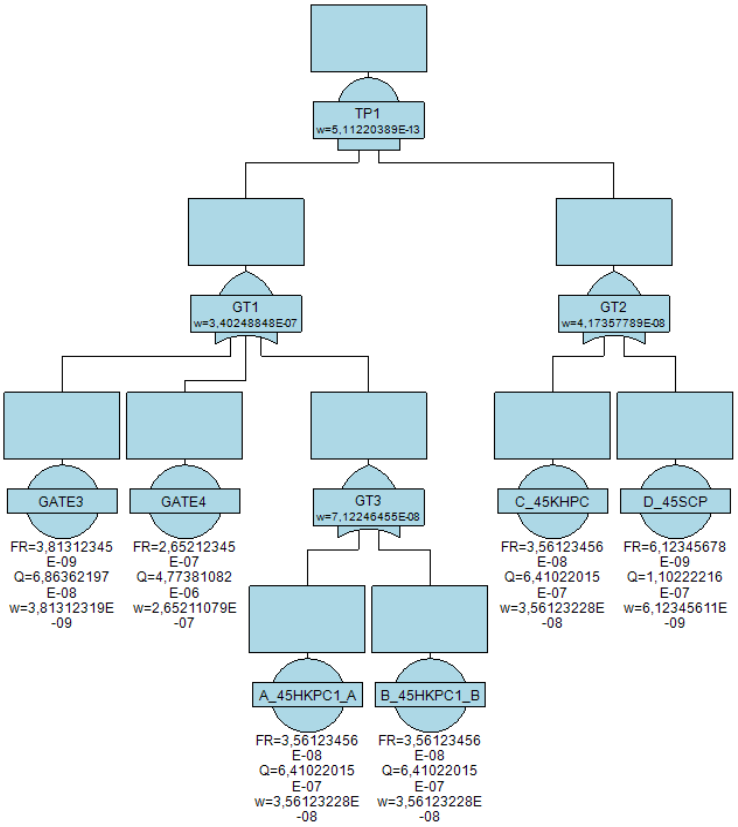

Figura 4.7: Frequenza di guasto

Avaria del sistema interno di apertura di emergenza di due porte adiacenti lungo un percorso diretto

Lo scenario previsto dal requisito 4.2.5.5.9.(4) corrisponde al manifestarsi contemporaneo su due porte adiacenti della scenario analizzato da IFE con la FTA 410 "il dispositivo di aperture di emergenza interno non funziona". La relativa probabilità di accadimento è pari a $2{,}66*10^{-12}$ eventi per ora di esercizio.

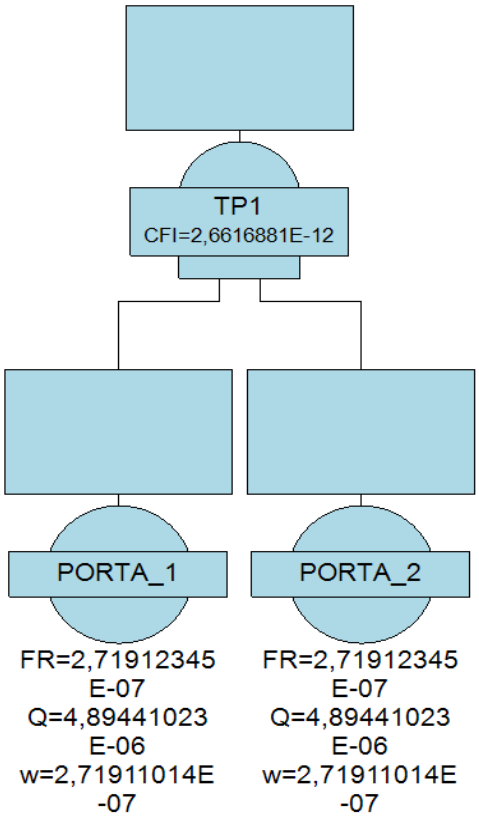

Figura 4.8: Probabilità di accadimento

Come per il caso precedente, si è poi passato all'analisi degli altri principali parametri, i cui risultati sono riportati nei seguenti alberi.

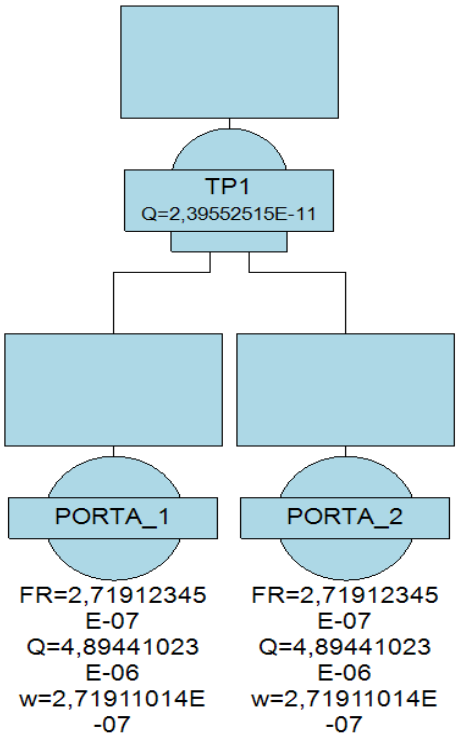

Figura 4.9: Indisponibilità

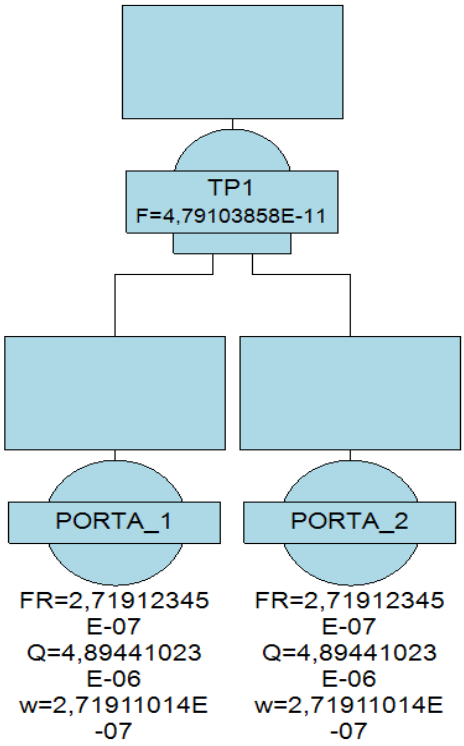

Figura 4.10: Inaffidabilità

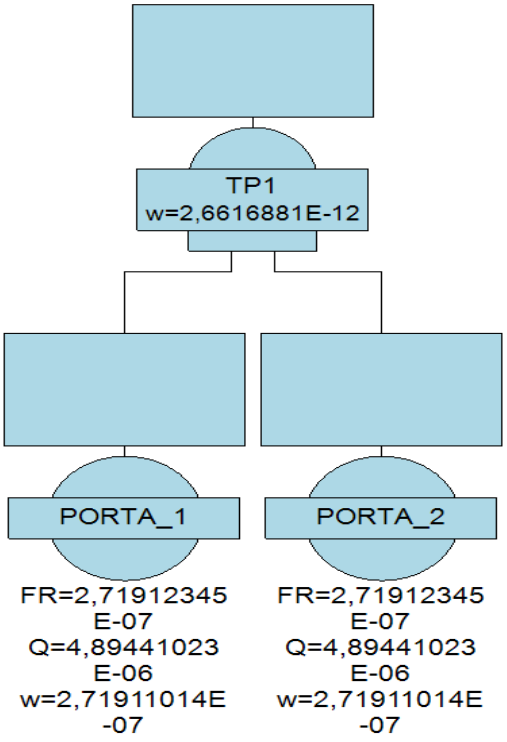

Figura 4.11: Frequenza di guasto

Questi risultati, come quelli del caso di studio presentato in precedenza, verranno analizzati nei paragrafi successivi e confrontati con i risultati ottenuti utilizzando il programma Simulink realizzato nel corso dello svolgimento della tesi.

Sintetizzando quanto detto finora, i risultati delle analisi effettuate con FaultTree+ relativi ai tre scenari di rischio previsti dalla TSI per il sistema porte esterne sono:

- La frequenza di accadimento dell'evento "una porta non è bloccata (con personale di bordo non correttamente informato di questo stato della porta) o è sbloccata o aperta in zone (ad esempio, sul lato sbagliato del treno) o situazioni (ad esempio, treno in movimento) inopportune" è pari a 8,32*10^{-15} eventi per ora di funzionamento
- La frequenza di accadimento dell'evento "diverse porte non sono bloccate (con personale di bordo non correttamente informato di questo stato delle porte) o sono sbloccate o aperte in zone (ad esempio, sul lato sbagliato del treno) o situazioni (ad esempio, treno in movimento) inopportune" è pari a 5,112*10^{-13} eventi per ora di funzionamento
- La frequenza di accadimento dell'evento "avaria del sistema interno di apertura di emergenza di due porte adiacenti lungo un percorso diretto (quale definito al punto 4.2.10.5 della presente TSI) restando disponibile il sistema di apertura di emergenza delle altre porte" è pari a 2,66*10^{-12} eventi per ora di funzionamento

È quindi dimostrato per tutti e tre gli scenari una frequenza di accadimento "improbabile" ovvero minore di 10^{-7} eventi per ora di funzionamento. Risultano pertanto rispettati i requisiti di sicurezza dei capitoli 4.2.5.5.8. e 4.2.5.5.9. della TSI LOCO & PASS 1302/2014.

4.2 Sistema porte esterne: analisi con Simulink

Sulla base di quanto definito nel paragrafo precedente, verranno ora svolte le stesse analisi di alberi dei guasti utilizzando il programma sviluppato in Matlab-Simulink nel corso della tesi. L'obiettivo è quello di dimostrare che l'errore relativo tra le Q(t), F(t), $\omega(t)$ e CFI calcolato con i risultati ottenuti dai due software (considerando come valori esatti quelli di FaultTree+) sia inferiore o uguale, considerando uno **studio a 9 cifre significative**, ad un ordine di grandezza pari a $\mathbf{10^{-6}}$ per ogni singolo evento, da quelli Basic al Top Event. Tutte le analisi verranno svolte basandosi sul metodo Rare Approximation (lo stesso utilizzato per gli esempi in FaultTree+).

4.2.1 Diverse porte non sono bloccate o sono sbloccate o aperte in zone o situazioni inopportune con personale di bordo non correttamente informato di questo stato delle porte

Il primo caso di studio che viene affrontato in Simulink riguarda lo scenario previsto dal requisito 4.2.5.5.8.(2). In quanto il tempo di missione risulta essere molto breve (18 ore), i diversi componenti sono stati catalogati come non riparabili; come dato di input quindi è stato necessario fornire, oltre al tempo di missione stesso, il tasso di guasto $\lambda(t)$ di ogni singolo componente. Una volta creato l'albero, inseriti i dati di input e avviata l'analisi, sono stati ottenuti i seguenti risultati

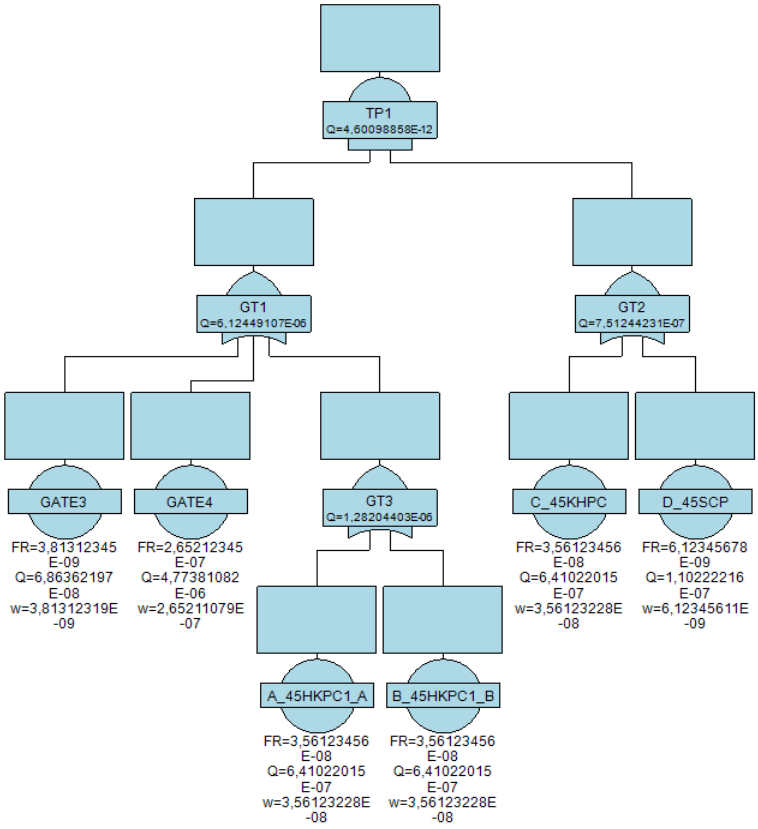

Figura 4.12: Indisponibilità albero FaultTree+

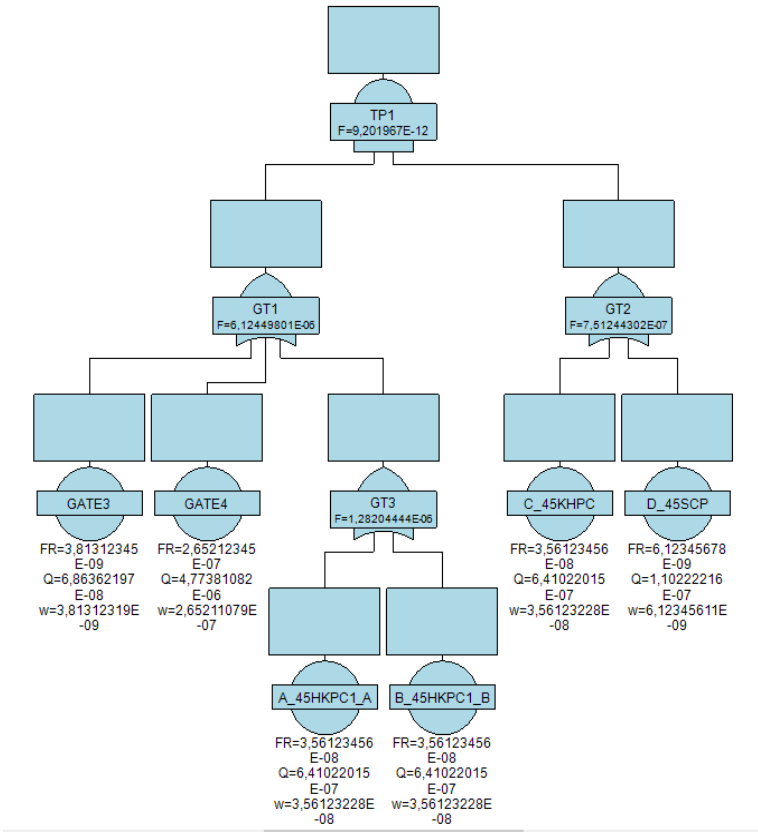

Figura 4.13: Inaffidabilità albero FaultTree+

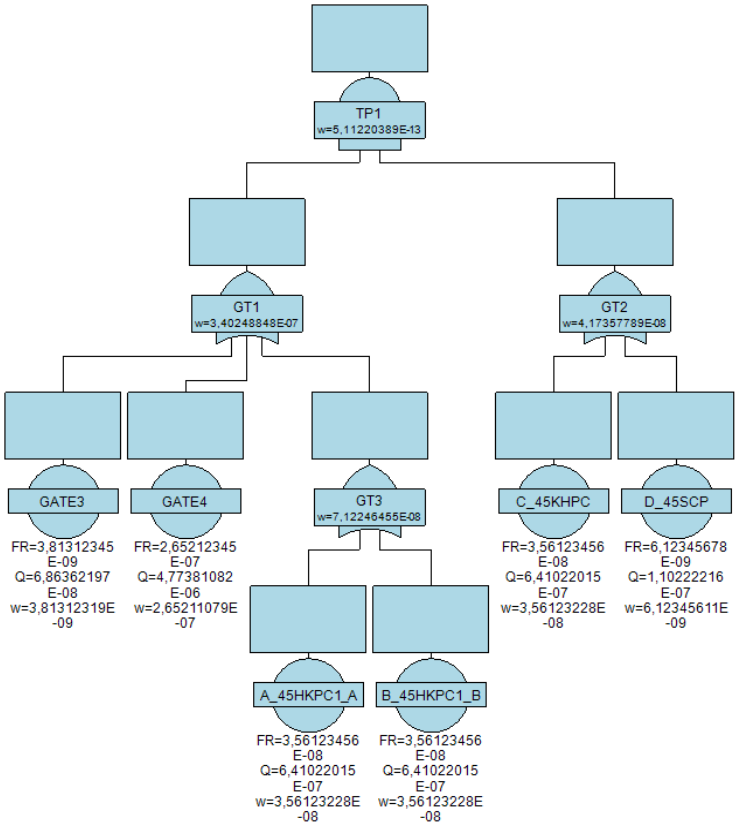

Figura 4.14: Frequenza di guasto albero FaultTree+

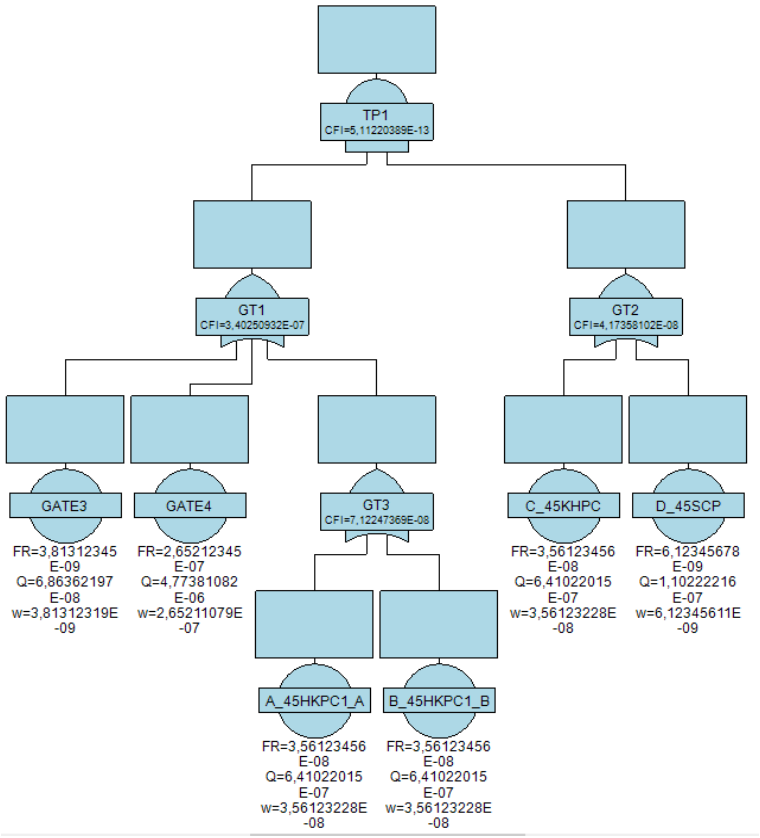

Figura 4.15: CFI albero FaultTree+

Successivamente, le stesse analisi sono state svolte utilizzando il programma sviluppato in Simulink, il quale ha fornito i seguenti risultati.

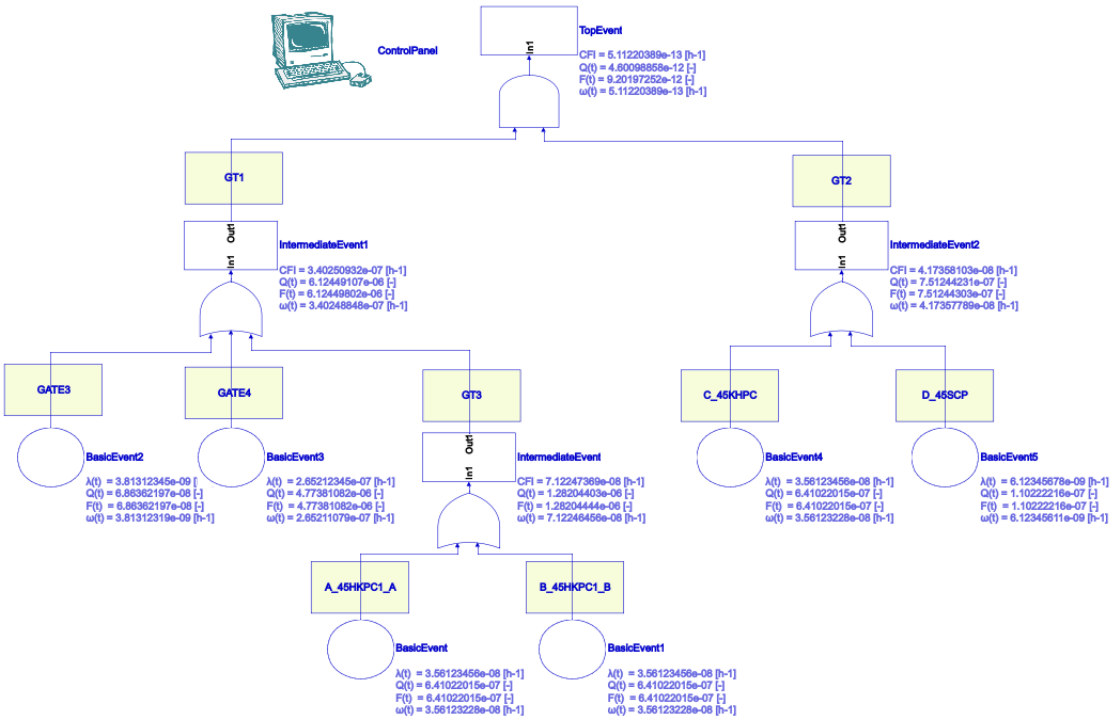

Figura 4.16: Albero Simulink

Come si può notare, rispetto a FaultTree+, il programma ha il vantaggio di mostrare a display in maniera chiara tutti i risultati di interesse per l'utente, non dovendo ripetere l'analisi ogni volta cambiando il parametro di output che si desidera. Per una più facile lettura, i risultati sono stati riportati nella tabella seguente

Basic Events									
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	Mission Time t [h]	Q(t) [-]	ω(t) [h^{-1}]	Q(t) [-] FT+	ω(t) [h^{-1}] FT+
BasicEvent	A_45HKPC1_A	Constant	UnRepairable	3.56123456E-08	18	**6.41022015E-07**	**3.56123228E-08**	**6.41022015E-07**	**3.56123228E-08**
BasicEvent1	B_45HKPC1_B	Constant	UnRepairable	3.56123456E-08	18	**6.41022015E-07**	**3.56123228E-08**	**6.41022015E-07**	**3.56123228E-08**
BasicEvent2	C_45KHPC	Constant	UnRepairable	3.56123456E-08	18	**6.41022015E-07**	**3.56123228E-08**	**6.41022015E-07**	**3.56123228E-08**
BasicEvent3	D_45SCP	Constant	UnRepairable	6.12345678E-09	18	**1.10222216E-07**	**6.12345611E-09**	**1.10222216E-07**	**6.12345611E-09**
BasicEvent4	GATE3	Constant	UnRepairable	3.81312345E-09	18	**6.86362197E-08**	**3.81312319E-09**	**6.86362197E-08**	**3.81312319E-09**
BasicEvent5	GATE4	Constant	UnRepairable	2.65212345E-07	18	**4.77381082E-06**	**2.65211079E-07**	**4.77381082E-06**	**2.65211079E-07**

Figura 4.17: Risultati caso di studio 4.2.5.5.8.(2) (Basic Events)

Intermediate Events									
Intermediate Event ID	Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] FT+	F(t) [-] FT+	ω(t) [h^{-1}] FT+	CFI [h^{-1}] FT+
IntermediateEvent	GT1	6.12449107E-06	6.12449802E-06	3.40248848E-07	3.40250932E-07	6.12449107E-06	6.12449801E-06	3.40248848E-07	3.40250932E-07
IntermediateEvent1	GT2	7.51244231E-07	7.51244303E-07	4.17357789E-08	4.17358103E-08	7.51244231E-07	7.51244302E-07	4.17357789E-08	4.17358102E-08
IntermediateEvent2	GT3	1.28204403E-06	1.28204444E-06	7.12246456E-08	7.12247369E-08	1.28204403E-06	1.28204444E-06	7.12246455E-08	7.12247369E-08

Figura 4.18: Risultati caso di studio 4.2.5.5.8.(2) (Intermediate Events)

Top Event								
Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] FT+	F(t) [-] FT+	ω(t) [h^{-1}] FT+	CFI [h^{-1}] FT+
TOPEVENT	4.60098858E-12	9.20197252E-12	5.11220389E-13	5.11220389E-13	4.60098858E-12	9.20196700E-12	5.11220389E-13	5.11220389E-13

Figura 4.19: Risultati caso di studio 4.2.5.5.8.(2) (Top Event)

Dai dati appena mostrati, è possibile calcolare l'errore relativo (Er) che si ha tra il programma e FaultTree+. Si ricorda che, per calcolare tale errore, si è ipotizzato come valori esatti quelli ottenuti con FaultTree+ (x) ed approssimati quelli di Simulink ($\bar{x}$)

$$Er = |\frac{x - \bar{x}}{x}| \tag{4.1}$$

I risultati sono riportati nelle seguenti tabelle, rappresentanti l'errore relativo di Basic Events, Intermediate Events e Top Event

Basic Events			
Basic Event ID	Event Label	Q(t) error (%)	ω(t) error (%)
BasicEvent	A_45HKPC1_A	0.00000000E+00	0.00000000E+00
BasicEvent1	B_45HKPC1_B	0.00000000E+00	0.00000000E+00
BasicEvent2	45KHPC	0.00000000E+00	0.00000000E+00
BasicEvent3	45SCP	0.00000000E+00	0.00000000E+00
BasicEvent4	GATE3	0.00000000E+00	0.00000000E+00
BasicEvent5	GATE4	0.00000000E+00	0.00000000E+00

Figura 4.20: Risultati caso di studio 4.2.5.5.8.(2) (Basic Events)

Intermediate Events					
Intermediate Event ID	Event Label	Q(t) [-] error (%)	F(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
IntermediateEvent	GT1	0.00000000E+00	1.63278699E-09	0.00000000E+00	0.00000000E+00
IntermediateEvent1	GT2	0.00000000E+00	1.33112491E-09	0.00000000E+00	2.39602397E-09
IntermediateEvent2	GT3	0.00000000E+00	0.00000000E+00	1.40400832E-09	0.00000000E+00

Figura 4.21: Risultati caso di studio 4.2.5.5.8.(2) (Intermediate Events)

Top Event				
Event Label	Q(t) [-] error (%)	F(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
TOPEVENT	0.00000000E+00	5.99871745E-07	0.00000000E+00	0.00000000E+00

Figura 4.22: Risultati caso di studio 4.2.5.5.8.(2) (Top Event)

Come si può notare i risultati risultano essere ben al di sotto del limite imposto di 10^{-6}, ed in moltissimi casi per 9 cifre significative l'errore è addirittura dello 0%! Inoltre questo risultato non vale solo per i Basic Events, ma anche per il Top: ciò significa che la propagazione dell'errore è molto ridotta.

4.2.2 Avaria del sistema interno di apertura di emergenza di due porte adiacenti lungo un percorso diretto

Lo scenario previsto dal requisito 4.2.5.5.9.(4) corrisponde al manifestarsi contemporaneo su due porte adiacenti dello scenario analizzato da IFE con la FTA 410 "il dispositivo di apertura di emergenza interno non funziona". Di seguito vengono riportati i valori di indisponibilità, inaffidabilità, probabilità di accadimento e frequenza di guasto (tempo di missione 18 ore).

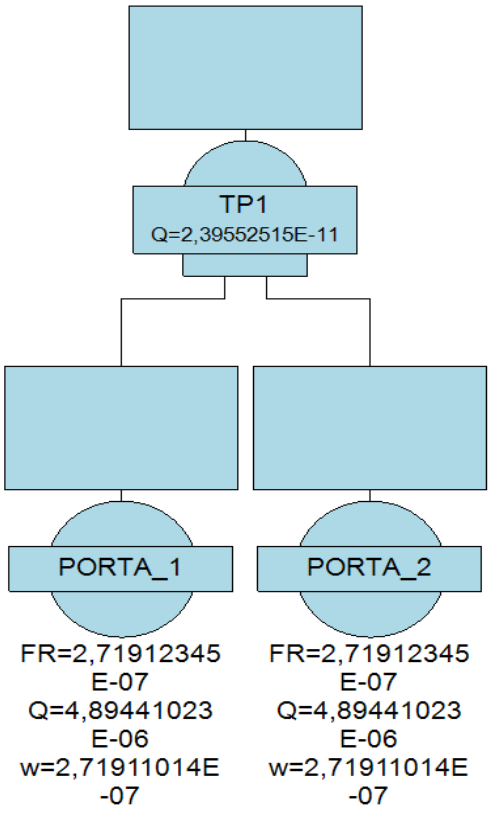

Figura 4.23: Indisponibilità albero FaultTree+

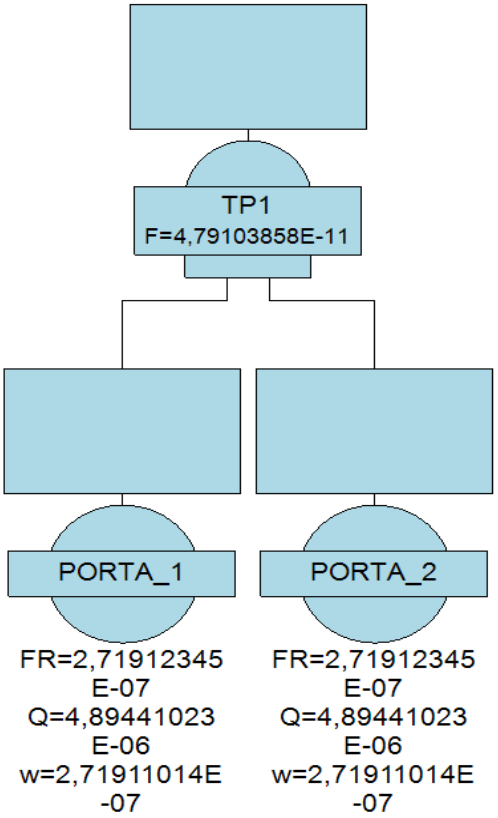

Figura 4.24: Inaffidabilità albero FaultTree+

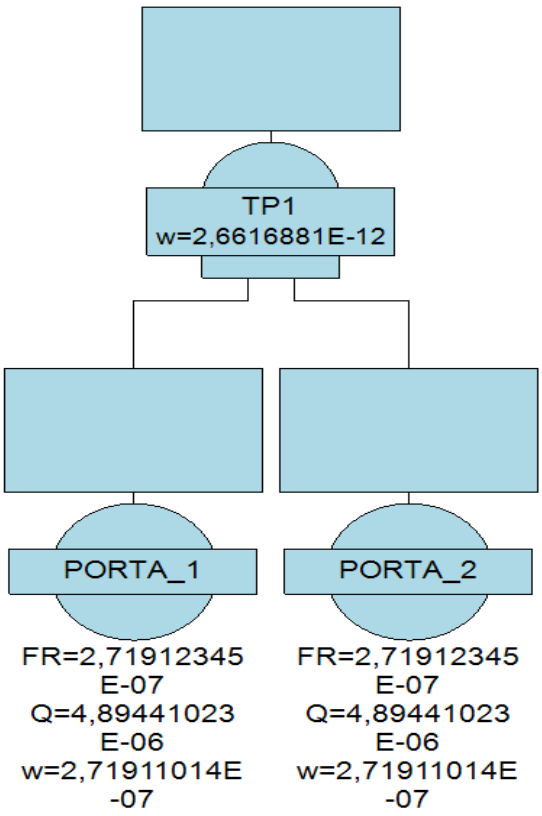

Figura 4.25: Frequenza di guasto albero FaultTree+

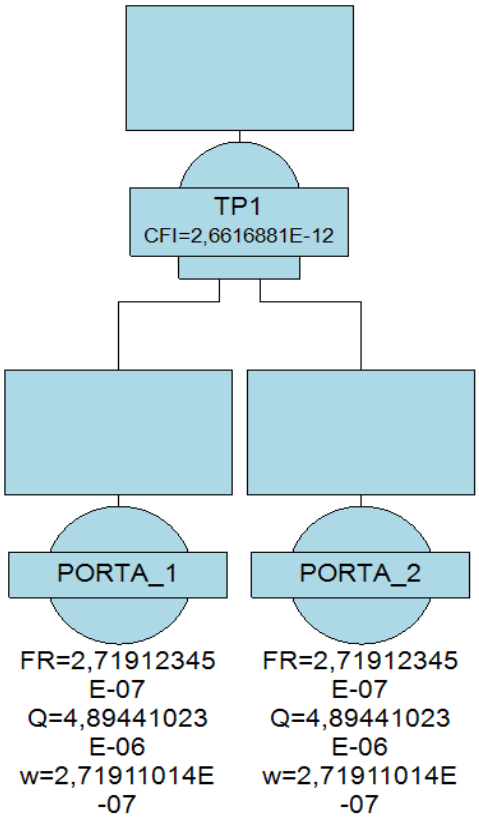

Figura 4.26: CFI albero FaultTree+

Successivamente, le stesse analisi sono state svolte utilizzando il programma sviluppato in Simulink nel corso della tesi, il quale ha fornito i seguenti risultati.

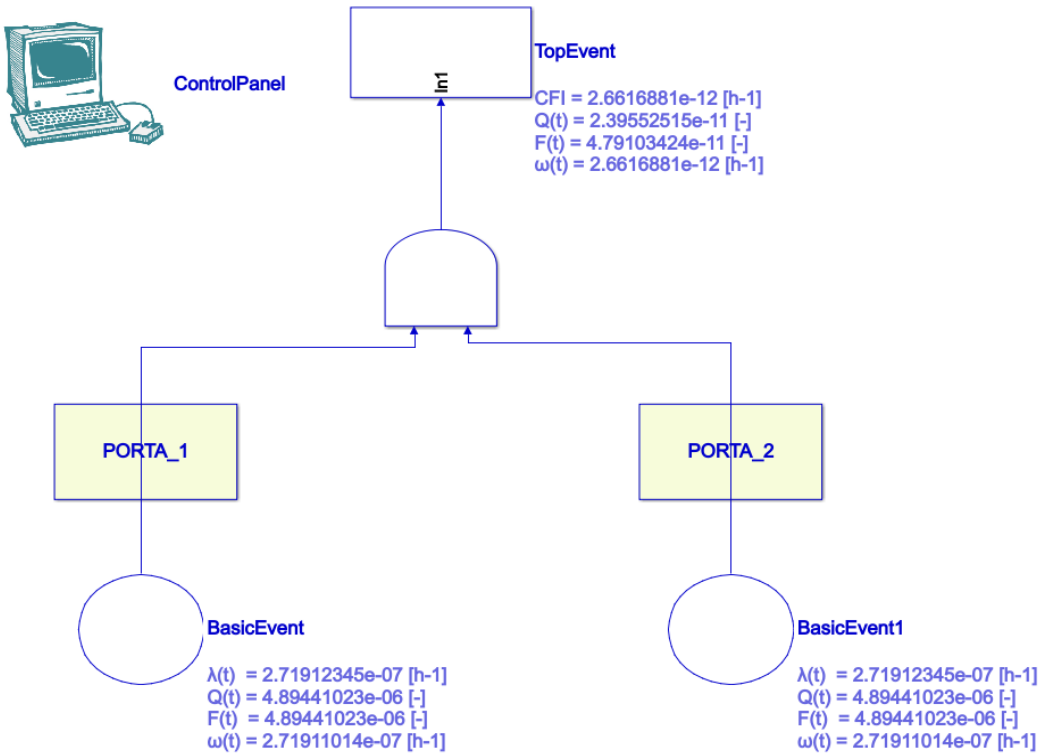

Figura 4.27: Albero Simulink

Per una più facile lettura, i risultati sono stati riportati nella tabella seguente

Basic Events									
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	Mission Time t [h]	Q(t) [-]	ω(t) [h^{-1}]	Q(t) [-] FT+	ω(t) [h^{-1}] FT+
BasicEvent	PORTA_1	Constant	UnRepairable	2.71912345E-07	18	4.89441023E-06	2.71911014E-07	4.89441023E-06	2.71911014E-07
BasicEvent1	PORTA_2	Constant	UnRepairable	2.71912345E-07	18	4.89441023E-06	2.71911014E-07	4.89441023E-06	2.71911014E-07

Figura 4.28: Risultati caso di studio 4.2.5.5.9.(4) (Basic Events)

Top Event								
Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] FT+	F(t) [-] FT+	ω(t) [h^{-1}] FT+	CFI [h^{-1}] FT+
TOPEVENT	2.39552515E-11	4.79103424E-11	2.66168810E-12	2.66168810E-12	2.39552515E-11	4.79103858E-11	2.66168810E-12	2.66168810E-12

Figura 4.29: Risultati caso di studio 4.2.5.5.9.(4) (Top Event)

Viene ora calcolato l'errore relativo legato ai diversi blocchi. I risultati sono riportati nelle seguenti tabelle, rappresentanti l'errore relativo di Basic Events e Top Event

Basic Events			
Basic Event ID	**Event Label**	**Q(t) error (%)**	**ω(t) error (%)**
BasicEvent	PORTA_1	**0.00000000E+00**	**0.00000000E+00**
BasicEvent1	PORTA_2	**0.00000000E+00**	**0.00000000E+00**

Figura 4.30: Risultati caso di studio 4.2.5.5.9.(4) (Basic Events)

Top Event				
Event Label	**Q(t) [-] error (%)**	**F(t) [-] error (%)**	**ω(t) [h-1] error (%)**	**CFI [h-1] error (%)**
TOPEVENT	**0.00000000E+00**	**9.05857869E-07**	**0.00000000E+00**	**0.00000000E+00**

Figura 4.31: Risultati caso di studio 4.2.5.5.9.(4) (Top Event)

Anche in questo caso, come in quello precedente, i risultati ottenuti rispettano i requisiti definiti in precendenza, fornendo un errore relativo massimo dell'ordine di grandezza di 10^{-6}.

Capitolo 5

Fase di test

5.1 Testare un software

La fase di test di un'applicazione è un processo di verifica e validazione, che mira a dimostrare il corretto funzionamento del programma affiché questo soddisfi i requisiti tecnici richiesti dal cliente; è una fase molto importante dell'intero ciclo di vita di un sistema perché permette di identificare possibili errori o comportamenti anomali del software, permettendone così la correzione [13].
Il test di un software ha tre obiettivi principali: la verifica, la scoperta di difetti e la validazione:

- Il processo di verifica assicura che il software rispetti le specifiche tecniche sotto le quali è stato realizzato; con specifica si intende la descrizione di una funzione in termini di un output misurabile dato uno specifico input sotto determinate condizioni iniziali
- Un difetto è la variazione che si ha tra il risultato aspettato e quello ottenuto
- Il processo di validazione assicura che il software rispetti i requisiti tecnici a cui è vincolato

Contrariamente a quello che intuitivamente si può pensare, il fine ultimo del testare un programma non è quello di scovarne i "bachi" (dall'inglese bugs, ossia gli errori tipicamente legati al codice) al suo interno: la fase di test ha come obiettivo principale quello di scovare difetti direttamente all'interno del prodotto finale; la loro mancata segnalazione può infatti portare a conseguenze catastrofiche anche all'intero progetto. Nell'ottobre del 1999 il Mars Climate Orbiter, un satellite dal costo di 125 milioni di dollari della NASA realizzato per analizzare le condizioni climatiche di Marte, andò perso nello spazio a causa di un'errata conversione delle unità di misura. Le indagini successive dimostrarono che un software del satellite aveva realizzato alcuni calcoli utilizzando le unità di misura imperiali, non eseguendo la conversione a quelle internazionali.
Il test di un software quindi permette di rispondere a quelle domande alle quali gli sviluppatori del codice non sanno rispondere:

- Il programma funziona come ci si aspetta?
- Soddisfa le richieste iniziali del cliente?
- Piace al cliente?
- Quali aree vanno migliorate?
- E' pronto per il rilascio?

Rispondendo a queste domande, si ha la possibilità di risparmiare tempo e soldi identificando, nelle fasi precedenti il rilascio, i difetti del programma, riducendo i tempi necessari alla sua correzione e fornendo al cliente finale un prodotto in linea con le sue aspettative nei tempi prestabiliti.

Il test di un programma viene realizzato attraverso una serie di precisi passi. Prima di tutto è necessario testare le sue parti fondamentali (che solitamente sono quelle più critiche), eseguendo prove che rappresentano situazioni normali per cui il programma viene utilizzato, prima di passare ad analizzare quelle particolari o poco probabili; per esempio, se un software è stato realizzato per elaborare una serie di dati, inizialmente verrà testato utilizzando un numero ragionevole di input, per poi solo in seguito fornirgli un numero maggiore di valori iniziali, mettendolo in una situazione di stress. Il valore aggiunto nel test di un programma è che questo va oltre al semplice test del codice, in quanto esamina il comportamento dell'intera applicazione. Non sempre infatti, se un programma non fornisce all'utente i risultati ottenuti, la colpa è automaticamente del codice: è possibile che questo sia corretto e solido, ma che i requisiti che questo deve soddisfare siano stati raccolti e comunicati in maniera non corretta. E' inoltre possibile che l'applicazione svolga correttamente quello per cui è stata realizzata, ma che gli input forniti dall'utente non siano corretti.
Una corretta fase di test analizza tutte le componenti che caratterizzano un programma. In più, questa fornisce l'opportunità di validare e verificare diversi altri aspetti, come le assunzioni fatte nei requisiti e la documentazione che fa da corredo all'applicazione. Il test può coinvolgere alcuni o tutti i seguenti aspetti, come i requisiti funzionali di design, quelli tecnici, il codice del programma, le norme e le restrizioni da applicare, gli standard dell'azienda e la configurazione dell'hardware.

Non c'è una sola persona incaricata a svolgere la fase di test: solitamente questo lavoro richiede un team, dalle dimensioni più o meno grandi a seconda della complessità dell'applicazione su cui devono essere effettuate le prove. I programmatori che hanno sviluppato il codice dovrebbero avere un ruolo marginale in questa fase, in quanto già coinvolti nella realizzazione del prodotto e quindi a conoscenza dei suoi possibili limiti e comportamenti. Coloro impiegati nella fase di test devono effettuare il numero più alto di prove possibili, essere curiosi, critici e buoni comunicatori; una parte del loro lavoro consiste nel porsi domande alle quali gli sviluppatori non hanno pensato, come ad esempio:

- Quanto bene lavora il software?
- Cosa significa che il software "funziona"?
- Come si sa con certezza che funziona? Quali prove si hanno?
- Quale può essere la causa del fatto che il programma non lavori correttamente?

Gli sviluppatori e gli addetti ai test devono condividere tra di loro il maggior numero possibile di informazioni, al fine di ridurre tempi morti che causerebbero ritardi e ulteriori costi all'intero progetto, producendo alla fine come documentazione un Test Report che elenchi tutti i controlli effettuati. Inoltre, le figure professionali che svolgono test devono avere una certa esperienza in questo campo, avendo così una maggiore abilità nello scovare malfunzionamenti del prodotto che potrebbero sfuggire ad un occhio meno esperto.
Solitamente la fase di test prevede la presenza di due figure chiave:

- Il test coordinator(s): crea il piano dei test che devono essere svolti e le specifiche che questi devono soddisfare in base ai requisiti posti dal cliente. Produce come output documenti tecnici volti a giustificare i risultati ottenuti
- Chi svolge i test: coloro i quali eseguono i test e documentano i risultati

5.2 Le fasi principali di un test

La fase di test è così importante che ricopre una buona parte del ciclo di vita di un programma. Come si può notare dal tipico ciclo a V riportato nella figura seguente, le attività procedono verso il basso e poi verso l'alto, da sinistra a destra definendo le sequenze basiche nello sviluppo delle attività di test; il modello evidenzia l'esistenza di diversi livelli di testing e definisce in che modo questi sono legati alle diverse fasi di sviluppo.

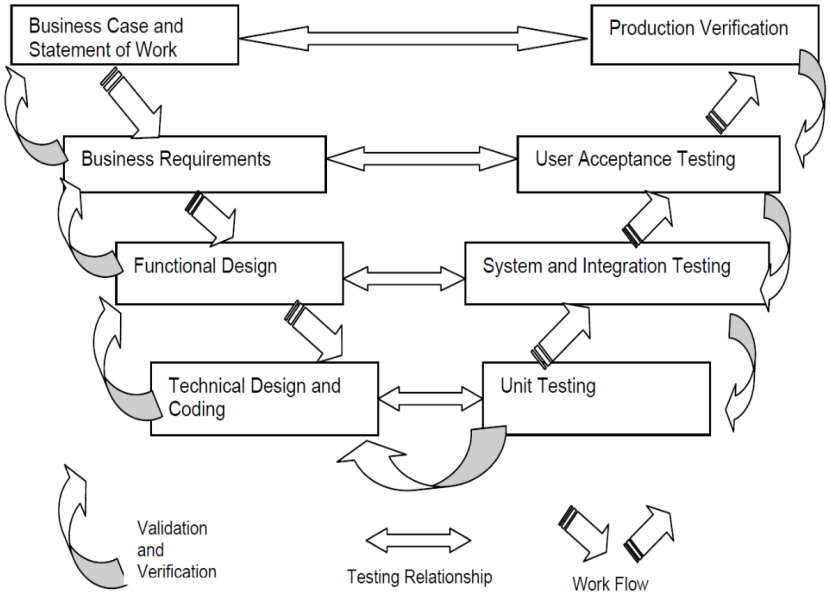

Figura 5.1: Modello a V fase di test

Lo schema del modello a V presenta cinque fasi principali di testing del programma, ad ognuna delle quali è associato una diversa tipologia di prova.

Fase	**Documentazione**	**Tipo di test**
Fase di sviluppo	Design tecnico	Test unitario
Integrazione dei sistemi	Design funzionale	Test di sistema/di integrazione
Approvazione dell'utente	Business requirements	Test di accettazione
Implementazione	Business case	Test di verifica del prodotto

Tabella 5.1: Fasi di test del programma

La quinta fase, che viene applicata alla fine di ognuna delle precedenti, consiste nel test di non regressione, volto a verificare che la risoluzione di un difetto non ne comporti la nascita di altri in parti del programma precendentemente non affette da errori. Ogni fase di test deve essere svolta attraverso l'inserimento di specifici parametri di ingresso, standardizzati e concordati in precedenza, che forniranno determinati risultati in caso di funzionamento corretto del software; questi valori d'ingresso vengono definiti dal Coordinatore dei test e documentati all'interno del Test Plan.

5.3 Il Test Plan

Il Test Plan è un documento obbligatorio che deve essere presentato come allegato alla fase di test di un prodotto. I seguenti campi devono essere presenti in un Test Plan

Campo	**Descrizione**	**Scopo**
Responsabilità	Compiti attribuiti a determinate persone	Affidamento di responsabilità
Assunzioni	Stato e disponibilità di codice e sistema	Evitare incomprensioni su scadenze
Test	Obiettivo del test, scadenze, durata e priorità	Delineare l'intero processo e definire test specifici
Comunicazione	Comunicare il piano (chi, cosa, quando, come)	Tutti conoscono tutto ciò che c'è da sapere
Analisi del rischio	Aspetti critici che devono essere testati	Identificare le aree critiche di successo
Segnalamento dei difetti	Come i difetti vengono registrati e documentati	Indicazione del difetto al fine di sistemarlo
Ambiente	Ambiente tecnico, aree di lavoro	Ridurre/eliminare le incomprensioni ed i possibili ritardi

Tabella 5.2: Struttura di un Test Plan

Un Test Plan redatto in maniera corretta aumenta le possibilità che il software funzioni in modo corretto; risulta infatti possibile concentrare l'attenzione dei test sulle aree che vengono identificate come quelle più a rischio di malfunzionamento. Queste riguardano non soltanto gli aspetti fondamentali del programma che devono sistematicamente funzionare in maniera corretta, ma anche in quali aspetti le conoscenze tecniche degli sviluppatori sono più deboli. Per identificare queste aree è di assoluta importanza raccogliere il maggior numero di informazioni possibili dai vari attori che hanno contribuito alla realizzazione del programma, oltre a consultare gli archivi contenenti dati ed errori riscontrati e catalogati in programmi precedenti al fine di evitare di commetterli nuovamente. Quando un errore viene trovato, è necessario scegliere il metodo più efficace per affrontarlo: questo significa avere una metodologia volta ad assegnare un valore d'importanza al difetto, affinché i maggiori sforzi per le correzioni del programma si concentrino prima di tutto sugli errori ai quali è stato assegnato il massimo peso. Uso comune è quello di identificare con un valore pari a 1 gli errori che impattano maggiormente sul corretto funzionamento del software, mentre con 6 quelli che influiscono marginalmente su di esso:

1. Impedimento: è impossibile continuare il test a causa della gravità del difetto
2. Critico: il test può continuare ma il prodotto non potrà essere rilasciato finché l'errore non verrà corretto
3. Importante: il test può continuare ma il difetto potrebbe non soddisfare i requisiti richiesti
4. Medio: il test può continuare ed il difetto comporta conseguenze minime rispetto al suo corretto funzionamento
5. Minore: il test può continuare ed il prodotto svolge correttamente i compiti richiesti. I difetti possono essere corretti senza difficoltà
6. Cosmetico: difetti minori, come font e colori non corretti, che non incidono sulla bontà dell'output

Normalmente un Test Plan viene condotto in maniera schematica, eseguendo prima quelli che vengono definiti test unitari, ossia su singole parti del programma, e successivamente dei test di sistema per verificarne, ad alto livello, il corretto funzionamento. Questa esecuzione ordinata di istruzioni è poco realistica, in quanto l'utente finale potrebbe non utilizzare le funzionalità del programma nel medesimo ordine, portando nei casi più gravi ad ottenere risultati errati; è necessario quindi realizzare un Test Plan tenendo in considerazione il fatto che il software potrebbe essere utilizzato su diversi dispositivi hardware, sistemi operativi differenti oppure in combinazione con altri programmi. Un test fondamentale consiste nel vedere come si comporta il programma sotto carichi di lavoro normali e sotto stress, ad esempio quando è necessario gestire una grande mole di dati: l'obiettivo è vedere se l'applicazione risponderà comunque in maniera corretta soddisfacendo i requisiti richiesti o se, sotto opportune ipotesi, presenterà un comportamento diverso da quello nominale. I test sotto stress sono solitamente svolti alla fine dell'intero ciclo, quando tutti gli altri problemi legati al normale funzionamento riscontrati in precedenza sono stati risolti; sfortunatamente questo si traduce nell'avere poco tempo a disposizione per realizzare le opportune modifiche nel caso si presentassero ulteriori problemi.

5.3.1 Test unitari

I test unitari vengono effettuati sulle singole unità di un programma ed hanno l'obiettivo di esaminare un componente individuale che è stato modificato oppure introdotto per la prima volta. Ogni test che ha come obiettivo di validare un singolo modulo deve essere presentato con la relativa documentazione tecnica in allegato, la quale conterrà tra le altre informazioni gli output che ci si aspetta che il modulo testato fornisca. I test unitari si focalizzano sulla funzionalità e affidabilità del software e sono svolti in una fase di test precedente all'integrazione del sistema. Se durante un test unitario venisse scoperto un difetto, verrebbe valutata la sua natura e l'impatto che questo può avere sul sistema generale; l'obiettivo è quello di risolverlo prima che il modulo testato venga approvato.

5.3.2 Test di sistema

I test di sistema vengono svolti su tutti i componenti e i moduli nuovi o modificati che caratterizzano un prodotto. L'obiettivo è quello di capire come i diversi blocchi unitari interagiscano tra di loro e se forniscano complessivamente gli output richiesti; viene posto l'accento sulla validazione e verifica dei requisiti di sistema e su come i singoli moduli lavorano tra di loro quando vengono connessi. Solitamente i test sul sistema sono più di uno: particolare menzione merita il primo (definito normalmente "smoke test"), il quale ha l'obiettivo di studiare a grandi linee come il programma si comporta e se le funzioni principali vengono svolte correttamente, senza soffermarsi sui dettagli. I test sull'intero sistema richiedono molto tempo, in quanto è necessario effettuarne un numero elevato per poter analizzare tutti i possibili scenari che possono verificarsi; il Test Plan in questo frangente ricopre un ruolo molto delicato, perché contiene la descrizione delle casistiche dei test, la sequenza in cui questi devono essere effettuati e la documentazione necessaria per elencarne i risultati. Quando viene scoperto e sistemato un errore, il test deve essere nuovamente effettuato, per accertarsi che le correzioni apportate non abbiano avuto influenze negative su componenti che precedentemente non presentavano errori (il test di regressione già citato in precedenza).

5.3.3 Test di integrazione

Dopo aver provveduto ai diversi test di sistema, è necessario accertarsi che il programma sviluppato fornisca i risultati desiderati anche se viene eseguito in ambienti diversi da quello nativo: risulta quindi necessario effettuare dei test di integrazione, nei quali il prodotto viene

testato in concomitanza di altre interfacce e applicazioni. A differenza dei test di sistema, in quelli di integrazione non è necessario effettuare nuovamente la prova se viene scoperto un difetto dopo che questo è stato sistemato. I test di integrazione sono divisi in diversi gruppi e possono essere effettuati o meno a seconda dell'applicazione che deve essere testata:

- Test di compatibilità: garantiscono che l'applicazione lavori con differenti configurazioni in base a quelli che ha a disposizione l'utente
- Test di performance: valutano la capacità dell'applicazione di funzionare correttamente quando, ad esempio, più utenti la utilizzano contemporaneamente oppure il numero di input aumenta
- Test di stress: testano il corretto funzionamento dell'applicazione quando questa viene stressata con carichi di lavoro inusuali
- Test di carico: sono il complemento di quelli di stress e valutano il funzionamento dell'applicazione sotto normali carichi di lavoro

5.3.4 Test di regressione

Il test di regressione, già menzionato nei paragrafi precedenti, viene effettuato ogni volta che la procedura di un pezzo di programma viene modificata in seguito all'identificazione di un difetto; quando un errore viene corretto, nasce la possibilità che ne venga introdotto involontariamente uno nuovo: si ha perciò l'introduzione dell'incertezza circa la capacità dell'applicazione di ripetere nuovamente tutte le funzioni svolte in precedenza in modo corretto. Il test di regressione viene solitamente svolto in parallelo con altri test e può essere visto come un controllo di qualità per assicurarsi che il codice appena modificato continui a svolgere correttamente le funzioni che non sono state oggetto di modifica e che rispetti gli stessi requisiti verificati in precedenza. In conclusione, si può affermare che il test di regressione si assicura che il resto dell'applicazione non soggetta a modifiche non risulti affetta da errori nati dalla correzione di altri.

5.4 Test del tool sviluppato nell'attività di tesi

I concetti teorici enunciati nei paragrafi precedenti sono un'utile introduzione per poter capire le basi sulle quali è stata effettuata la fase di test del programma sviluppato in questo progetto di tesi. E' stato innanzitutto necessario definire una procedura dettagliata, o Test Case, in grado di indicare con precisione gli aspetti del programma che sono stati testati. Mentre un Test Plan descrive cosa c'è da testare, una Test Case definisce come effettuare un particolare test attraverso la definizione di una serie di requisiti che devono essere soddisfatti [14]: chi è incaricato a svolgere questo compito sa che il miglior modo per determinare la conformità del software rispetto ai requisiti che deve soddisfare è quello di programmare un certo numero di test, sia a livello unitario che di sistema. Nel fare ciò, il team di test deve tenere a mente le linee guida generali che caratterizzano questa fase:

- Lo scopo di ogni test è quello di essere eseguito nel modo più semplice possibile
- Concentrarsi inizialmente sui test positivi, volti a dimostrare che il software esegue correttamente quello per cui è stato realizzato
- Test già svolti in altri casi devono essere approfonditi, mentre ulteriori tipi di test devono essere realizzati per dimostrare che il software non sfora i vincoli che gli sono stati imposti svolgendo operazioni non previste (i cosiddetti test negativi)

- Dove possibile, diverse tipologie di test devono essere realizzate al fine di verificare la mancanza di problemi dal punto di vista della performance e dei requisiti di sicurezza del software
- Ulteriori tipologie di test dovrebbero essere svolte, dal punto di vista unitario, per raggiungere determinati obiettivi nella copertura di test del software

La maniera nella quale viene definita una strategia di test varia a seconda dell'azienda che la effettua: nonostante ciò, uno dei metodi più utilizzati è quello di creare degli appositi template, utilizzando il più delle volte dei fogli di calcolo quali Microsoft Excel sotto forma tabellare, nei quali vengono elencati tutti i test di tipo unitario e di sistema che vengono svolti, al fine di verificare il corretto funzionamento del programma testato.
Per questo lavoro di tesi si è deciso di concentrare l'attenzione sui test positivi, sia di tipo unitario che di sistema. Lo scopo è quello di dimostrare che il programma realizzato, nelle mani di un utente con una certa esperienza nel campo dell'analisi RAMS, fornisca le prestazioni ed i risultati che ci si attende di ottenere da un qualsiasi altro software commerciale. Futuri lavori di tesi potranno approfondire il discorso legato ai test, effettuando anche quelli negativi per trovare possibili limitazioni nell'uso del software.
Una nota ulteriore per quel che riguarda la fase di test è necessaria: l'attenzione si è concentrata sui test riguardanti gli alberi di guasto composti da elementi non riparabili. Questa scelta è stata fatta in quanto tale famiglia è quella più ricorrente nello studio dei fault trees, quindi si è deciso di approfondire il discorso su di essi. A margine di tali test è stata effettuata una prova su un semplice albero composto da elementi riparabili e testati, per dimostrare che la bontà dei risultati ottenuti è paragonabile a FaultTree+, pur non inserendo come vincolo stringente un errore relativo massimo dell'ordine di grandezza di 10^{-6}. Di seguito sono riportate le tabelle rappresentanti i risultati ottenuti.

Basic Events			
Basic Event ID	**Event Label**	**Q(t) error (%)**	**ω(t) error (%)**
BasicEvent	A_45HKPC1_A	**0.00000000E+00**	**0.00000000E+00**
BasicEvent1	B_45HKPC1_B	**0.00000000E+00**	**8.42397481E-09**
BasicEvent2	45KHPC	**0.00000000E+00**	**0.00000000E+00**

Figura 5.2: Confronto Basic Events Simulink-FaultTree+ componenti Fixed

Intermediate Events				
Intermediate Event ID	**Event Label**	**Q(t) [-] error (%)**	**ω(t) [h-1] error (%)**	**CFI [h-1] error (%)**
IntermediateEvent	GT1	**0.00000000E+00**	**0.00000000E+00**	**1.40400084E-09**

Figura 5.3: Confronto Intermediate Events Simulink-FaultTree+ componenti Fixed

Top Event			
Event Label	Q(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
TOPEVENT	0.00000000E+00	0.00000000E+00	0.00000000E+00

Figura 5.4: Confronto Top Event Simulink-FaultTree+ componenti Fixed

Basic Events			
Basic Event ID	Event Label	Q(t) error (%)	ω(t) error (%)
BasicEvent	A_45HKPC1_A	0.00000000E+00	0.00000000E+00
BasicEvent1	B_45HKPC1_B	0.00000000E+00	8.42397481E-09
BasicEvent2	45KHPC	0.00000000E+00	0.00000000E+00

Figura 5.5: Confronto Basic Events Simulink-FaultTree+ componenti riparabili

Intermediate Events				
Intermediate Event ID	Event Label	Q(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
IntermediateEvent	GT1	0.00000000E+00	0.00000000E+00	1.40400084E-09

Figura 5.6: Confronto Intermediate Events Simulink-FaultTree+ componenti riparabili

Top Event			
Event Label	Q(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
TOPEVENT	0.00000000E+00	0.00000000E+00	0.00000000E+00

Figura 5.7: Confronto Top Event Simulink-FaultTree+ componenti riparabili

Dai risultati si può notare che, pur avendo fatto un solo esempio e quindi non avendone la certezza assoluta, i risultati ottenuti dal confronto degli output a 9 cifre significative forniscono un errore massimo di un ordine di grandezza pari a 10^{-9}! Ciò significa che le formule utilizzate per entrambi i tipi di componente sono in linea con quelle di FaultTree+. Visto che sono

trattati componenti che prevedono un periodo di missione anche di anni, l'inaffidabilità gioca un ruolo marginale e di conseguenza non è stata valutata, in quanto parametro non utile alle analisi di sicurezza.

5.4.1 Test di tipo unitario realizzati

Seguendo quanto riportato in letteratura, la prima fase di test si è concentrata su prove di tipo unitario. Questa fase, a sua volta, è stata suddivisa in due tipologie di test principali:

- L'inserimento del numero corretto di Basic Events contenuti all'interno del template Excel creato per velocizzare il lavoro dell'utente
- Il corretto inserimento e salvataggio degli input e degli output contenuti nei Basic Events

Per ognuna delle due fasi sono stati indicati una serie di requisiti che, se rispettati, dimostreranno che il test del programma è stato superato con successo.

Importazione dei dati da template Excel

La descrizione del template Excel realizzato appositamente per il programma Simulink è già stata fornita nel capitolo 3, ma per maggiore comodità viene riportata di seguito la tabella corrispondente.

BASIC EVENTS										
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]
BasicEvent										
BasicEvent1										
BasicEvent2										

Figura 5.8: Template Excel per inserimento dati Basic Events

In base al numero di Basic Events che devono essere inseriti, lo user può aggiungere o rimuovere righe dalla tabella a suo piacimento. Nella colonna *Event Label* è possibile inserire il nome dell'evento che caratterizzerà il blocco in Simulink. Tramite menu a tendina appositamente realizzati per evitare errori di battitura, l'utente può scegliere il modello caratterizzante l'evento (*Model Type*) ed il tipo di componente (*Component Type*), mentre nelle colonne successive è possibile inserire i dati di input assegnati ad ogni evento. Una volta compilata la tabella, è possibile salvare il template e chiudere il file Excel. Dalla libreria Simulink, selezionando il comando *Excel–>FTA Tool* contenuto nella GUI del Control Panel, apparirà una finestra dalla quale selezionare il template appena creato: in automatico verrà generato un nuovo modello Simulink, al cui interno compariranno ordinati su una o più file tutti i Basic Events, insieme al blocco Control Panel.
L'obiettivo del primo test è ottenere un modello Simulink contenente l'esatto numero di blocchi Basic Events introdotti nel template Excel ordinati ed equidistanziati, oltre all'inserimento del blocco Control Panel. Per una valutazione che comprenda più scenari, un primo test è stato effettuato caricando nel template i Basic Events già presentati nell'esempio 4.2.5.5.8.(2) del capitolo 4, e successivamente un secondo test contenente modelli e tipi di componente diversi per ogni Basic Event. Di seguito viene riportata la Test Case realizzata, contenente i requisiti da soddisfare, con gli esiti ottenuti dal test.

4.2.5.5.8.(2)							
Test ID	Descrizione	Blocco	Modello	Tipo di componente	Risultato atteso	Passato/Fallito	Commenti
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent1	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent2	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent3	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent4	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent5	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
1	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	Control Panel			Corretto inserimento control panel in nuovo modello Simulink	PASSATO	

Figura 5.9: Risultati test esempio 4.2.5.5.8.(2)

BASIC EVENTS							
Test ID	Descrizione	Blocco	Modello	Tipo di componente	Risultato atteso	Passato/Fallito	Commenti
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent	Fixed		Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent1	Constant	Repairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent2	Constant	UnRepairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent3	Constant	Tested	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent4	Fixed		Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	BasicEvent5	Constant	Repairable	Corretto inserimento Basic Event in nuovo modello Simulink	PASSATO	
2	Inserimento di n°6 Basic Events ordinati ed equidistanziati + n°1 Control Panel	Control Panel			Corretto inserimento control panel in nuovo modello Simulink	PASSATO	

Figura 5.10: Risultati test con modelli/componenti di BEs diversi

Entrambi i test risultano superati, in quanto soddisfacenti i requisiti che si erano imposti all'inizio. In seguito al superamento di questo primo test, è stato possibile passare ad analizzare il secondo test unitario.

Inserimento corretto e salvataggio input e output Basic Events

Una volta constatato il corretto inserimento dei blocchi dei Basic Events e di quello del Control Panel, l'attenzione si è focalizzata sui valori di input e output forniti dagli stessi blocchi. L'obiettivo del test è di dimostrare che i valori inseriti nel template Excel ed attribuiti allo specifico Basic Event vengano correttamente riportati nel blocco creato in Simulink. Un ulteriore obiettivo è stato quello di dimostrare che i valori, una volta salvato, chiuso e riaperto il modello, risultino ancora correttamente inseriti all'interno delle GUI dei singoli blocchi. Infine, oltre agli input, questa fase di test comprende l'analisi del corretto salvataggio degli output, calcolati dall'utente attraverso il comando *Compute* presente all'interno di ogni GUI. Anche in questo caso, come per il test precedente, sono state effettuate due serie di test, la prima utilizzando come dati in input quelli dell'esempio 4.2.5.5.8.(2) e successivamente Basic Events comprendenti diversi modelli e tipologie di componente. Di seguito sono riportati i risultati del test.

4.2.5.5.8.(2)										
Test ID	Descrizione	Blocco	Modello	Tipo di componente	Failure rate λ [h-1]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Passato/Fallito	Commenti
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent	Constant	UnRepairable	3.56E-08		18		PASSATO	
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent1	Constant	UnRepairable	3.56E-08		18		PASSATO	
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent2	Constant	UnRepairable	3.56E-08		18		PASSATO	
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent3	Constant	UnRepairable	6.00E-09		18		PASSATO	
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent4	Constant	UnRepairable	3.813E-25		18		PASSATO	
3	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output	BasicEvent5	Constant	UnRepairable	2.652E-07		18		PASSATO	

Figura 5.11: Risultati test esempio 4.2.5.5.8.(2)

BASIC EVENTS													
Test ID	Descrizione	Blocco	Modello	Tipo di componente	Failure rate λ [h-1]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Q(t) [-]	F(t) [-]	ω(t) [h-1]	Passato/Fallito	Commenti
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent	Fixed		3.56E-08				1.236E-07	1.236E-07	3.56E-08	PASSATO	
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent1	Constant	Repairable	3.56E-08	10	3640					PASSATO	
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent2	Constant	UnRepairable	3.56E-08		18					PASSATO	
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent3	Constant	Tested	6.00E-09	10	18	1000				PASSATO	
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent4	Fixed		3.813E-08				2.236E-07	2.236E-07	3.813E-08	PASSATO	
4	Corretto inserimento e salvataggio dei valori importati da template Excel e degli output forniti.	BasicEvent5	Constant	Repairable	2.652E-07	10	18	1				PASSATO	

Figura 5.12: Risultati test con modelli/componenti di BEs diversi

Entrambi i test sono risultati superati, in quanto soddisfacenti i requisiti che si erano imposti all'inizio.

5.4.2 Test di sistema

Una volta effettuati e superati i test unitari, è stato possibile concentrare l'attenzione sui test di sistema, che riguardano il corretto funzionamento del programma oggetto di questa tesi. Anche in questo caso, sono stati realizzati due tipi di test con l'obiettivo di verificare:

- La corretta propagazione numerica delle informazioni dell'albero dei guasti, partendo dai Basic Events fino ad arrivare al Top Event
- La corretta esportazione dei risultati ottenuti in formato Excel

Per ognuna delle due fasi sono stati indicati una serie di requisiti che, se rispettati, dimostreranno che il test del programma è stato superato con successo. Gli output degli alberi oggetto di questa fase di test sono stati confrontati con gli equivalenti realizzati in FaultTree+.

Propagazione delle informazioni

La combinazione dei diversi Basic Events attraverso le porte logiche, come si sa, fornisce il risultato finale dell'analisi dell'albero dei guasti, ossia il Top Event del sistema. Affinché il risultato sia attendibile, i valori numerici devono combinarsi e propagarsi in maniera corretta, a seconda della porta logica alla quale sono collegati. Nella Fault Tree Analysis esistono infiniti esempi di alberi dei guasti, dai più semplici costituiti da due o tre Basic Events ai più complessi, costituiti da 15 o più blocchi elementari a formare diversi livelli prima di arrivare al Top Event; chiaramente la probabilità di errore di propagazione dei risultati risulta maggiore in questi ultimi: si è quindi scelto, per testare la bontà e la potenza del programma Simulink realizzato per questo lavoro di tesi, di realizzare prima un test su un albero più semplice, come quello dell'esempio 4.2.5.5.8.(2) e successivamente su uno molto più complesso, comprendente

ben 19 Basic Events. Avendo trattato in questi ultimi mesi in ambito lavorativo diverse Fault Tree Analysis, si è notato che in media un albero dei guasti è composto da un numero di Basic Events variabile solitamente tra 3 e 24 blocchi; la scelta quindi di effettuare i test sui due alberi enunciati in precedenza non è casuale, in quanto ricoprono la maggioranza dei casi di studio affrontati nella realtà.
Il primo test effettuato riguarda lo studio della corretta propagazione delle informazioni dell'albero presentato nell'esempio 4.2.5.5.8.(2), confrontato con lo stesso realizzato tramite libreria Simulink nel capitolo 4.

4.2.5.5.8.(2)										
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]
BasicEvent	A_45HKPC1_A	Constant	UnRepairable	3.56123456E-08		18				
BasicEvent1	B_45HKPC1_B	Constant	UnRepairable	3.56123456E-08		18				
BasicEvent2	C_45KHPC	Constant	UnRepairable	3.56123456E-08		18				
BasicEvent3	D_45SCP	Constant	UnRepairable	6.12345678E-09		18				
BasicEvent4	GATE3	Constant	UnRepairable	3.81312345E-09		18				
BasicEvent5	GATE4	Constant	UnRepairable	2.65212345E-07		18				

Figura 5.13: Template Excel albero 4.2.5.5.8.(2)

Partendo dai Basic Events ottenuti tramite template Excel (analizzati nel paragrafo dei test unitari) è stato ricreato l'albero inserendo le porte logiche necessarie, gli Intermediate Events ed il Top Event, oltre ai blocchi *Event Description* contenenti la descrizione di ogni evento.

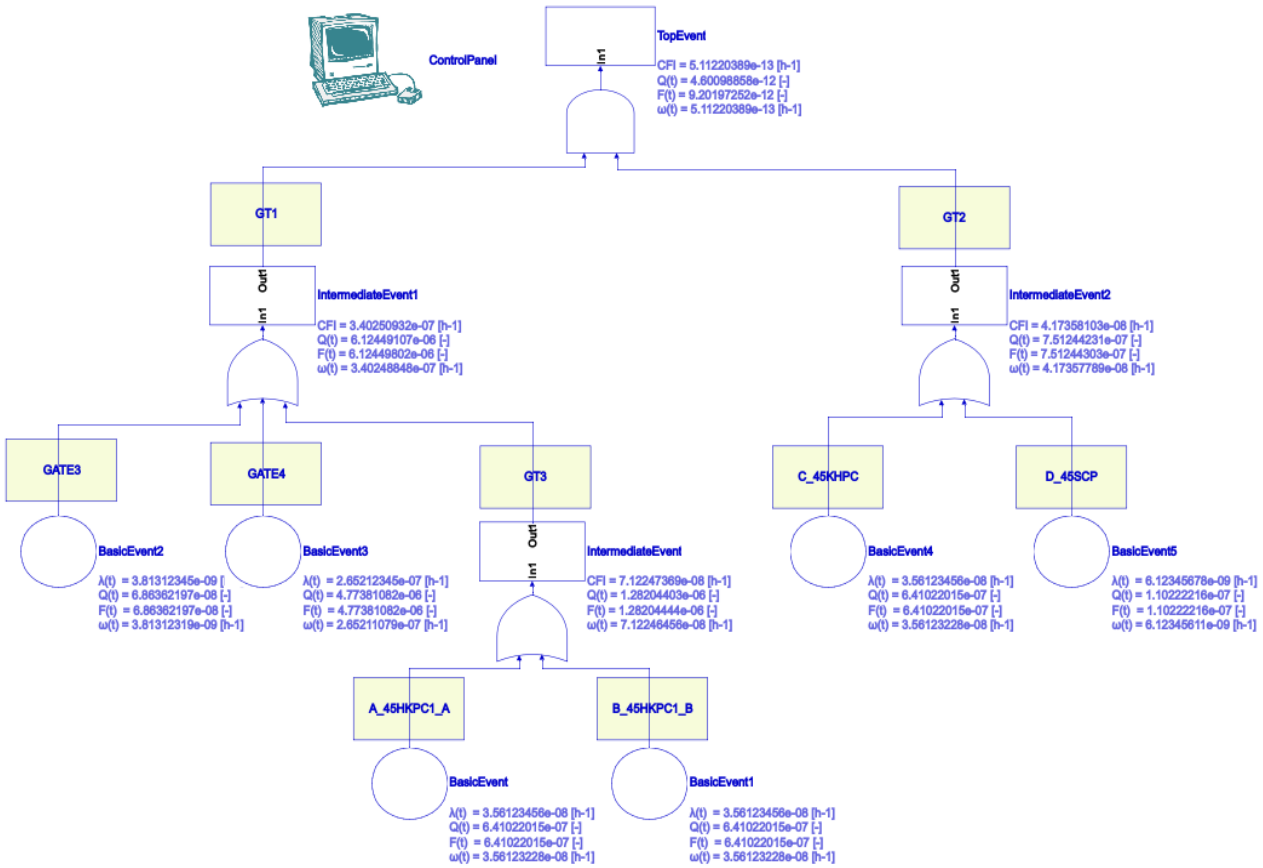

Figura 5.14: Albero esempio 4.2.5.5.8.(2)

Da sottolineare il fatto che la scelta di mostrare a schermo valori con 9 cifre significative non sia stata una scelta casuale: infatti questa è la massima precisione che fornisce FaultTree+ (mentre Matlab arriva fino a 16 cifre significative). Per permettere a Matlab di operare a 9 cifre significative, è stata creata un'apposita funzione chiamata *my_formatS*, grazie alla quale i valori forniti in output nei diversi alberi dei guasti sono presentati con 9 cifre significative. Gli output dell'albero ottenuto sono stati confrontati con quelli presentati nel capitolo 4: lo scopo del test infatti è quello di confermare che i risultati ottenuti dal programma Simulink realizzato mediante template Excel siano identici a quelli ricavati dall'albero del capitolo 4, il quale è stato realizzato immettendo i Basic Events non da template ma dalla libreria. I risultati ottenuti sono stati analizzati e riportati nella tabella successiva.

Basic Events									
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	Mission Time t [h]	Q(t) [-]	ω(t) [h^{-1}]	Q(t) [-] Template	ω(t) [h^{-1}] Template
BasicEvent	A_45HKPC1_A	Constant	UnRepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent1	B_45HKPC1_B	Constant	UnRepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent2	45KHPC	Constant	UnRepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent3	45SCP	Constant	UnRepairable	6.12345678E-09	18	1.10222216E-07	6.12345611E-09	1.10222216E-07	6.12345611E-09
BasicEvent4	GATE3	Constant	UnRepairable	3.81312345E-09	18	6.86362197E-08	3.81312319E-09	6.86362197E-08	3.81312319E-09
BasicEvent5	GATE4	Constant	UnRepairable	2.65212345E-07	18	4.77381082E-06	2.65211079E-07	4.77381082E-06	2.65211079E-07

Figura 5.15: Risultato test propagazione informazioni (Basic Events)

Intermediate Events									
Intermediate Event ID	Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] Template	F(t) [-] Template	ω(t) [h^{-1}] Template	CFI [h^{-1}] Template
IntermediateEvent	GT1	6.12449107E-06	6.12449802E-06	3.40248848E-07	3.40250932E-07	6.12449107E-06	6.12449802E-06	3.40248848E-07	3.40250932E-07
IntermediateEvent1	GT2	7.51244231E-07	7.51244303E-07	4.17357789E-08	4.17358103E-08	7.51244231E-07	7.51244303E-07	4.17357789E-08	4.17358103E-08
IntermediateEvent2	GT3	1.28204403E-06	1.28204444E-06	7.12246456E-08	7.12247369E-08	1.28204403E-06	1.28204444E-06	7.12246456E-08	7.12247369E-08

Figura 5.16: Risultato test propagazione informazioni (Intermediate Events)

Top Event								
Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] Template	F(t) [-] Template	ω(t) [h^{-1}] Template	CFI [h^{-1}] Template
TOPEVENT	4.60098858E-12	9.20197252E-12	5.11220389E-13	5.11220389E-13	4.60098858E-12	9.20197252E-12	5.11220389E-13	5.11220389E-13

Figura 5.17: Risultato test propagazione informazioni (Top Event)

Come si può notare, i risultati ottenuti con i due diversi metodi sono perfettamente identici, dai Basic Events fino al Top Event; questo dimostra che non solo gli input sono stati caricati correttamente dal template di Excel, ma anche che la propagazione numerica è avvenuta in modo corretto, in quanto i risultati finali ottenuti con i Top Event combaciano alla perfezione. Il test può quindi dirsi superato.

4.2.5.5.8.(2) Test Results						
Test ID	Descrizione	Blocco	Modello	Tipo di componente	Passato/Fallito	Commenti
5	Corretto inserimento degli input da Excel e corretta propagazione numerica fino al Top Event.	IntermediateEvent	Constant	UnRepairable	PASSATO	
5	Corretto inserimento degli input da Excel e corretta propagazione numerica fino al Top Event.	IntermediateEvent1	Constant	UnRepairable	PASSATO	
5	Corretto inserimento degli input da Excel e corretta propagazione numerica fino al Top Event.	IntermediateEvent2	Constant	UnRepairable	PASSATO	
5	Corretto inserimento degli input da Excel e corretta propagazione numerica fino al Top Event.	TOPEVENT	Constant	UnRepairable	PASSATO	

Figura 5.18: Test propagazione informazioni

Come test legato alla corretta funzione del tool Simulink sviluppato nel seguente lavoro di tesi rispetto ai risultati forniti da FaultTree+, è stato realizzato un albero dei guasti contenente 19 Basic Events, sempre attraverso l'utilizzo del template Excel; questo test risulta quindi essere il più importante tra quelli realizzati, in quanto si prefigge di dimostrare che i risultati ottenuti dal programma Simulink realizzato in questo lavoro di tesi sono identici o si discostano di poco rispetto a quelli forniti da un tool commerciale ed utilizzato massicciamente a livello mondiale come FaultTree+.
Il requisito che si vuole soddisfare mediante questo test è quello di verificare che l'errore relativo tra il programma Simulink e FaultTree+ legato agli output forniti rimanga nell'ordine di grandezza di 10^{-6}, confermando di conseguenza la corretta propagazione numerica delle informazioni fino al Top Event.

Albero 19 BEs										
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	MTTR [h]	Mission Time t [h]	Test Interval τ [h]	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]
BasicEvent	EV1	Constant	Unrepairable	3.56123456E-08		18				
BasicEvent1	EV2	Constant	Unrepairable	3.12612345E-08		18				
BasicEvent2	EV3	Constant	Unrepairable	9.71234567E-08		18				
BasicEvent3	EV4	Constant	Unrepairable	3.88712345E-08		18				
BasicEvent4	EV5	Constant	Unrepairable	9.71234567E-08		18				
BasicEvent5	EV6	Constant	Unrepairable	3.88712345E-08		18				
BasicEvent6	EV7	Constant	Unrepairable	3.56123456E-08		18				
BasicEvent7	EV8	Constant	Unrepairable	3.12712345E-08		18				
BasicEvent8	EV9	Constant	Unrepairable	9.71234567E-08		18				
BasicEvent9	EV10	Constant	Unrepairable	2.96812345E-08		18				
BasicEvent10	EV11	Constant	Unrepairable	9.71234567E-08		18				
BasicEvent11	EV12	Constant	Unrepairable	2.96812345E-08		18				
BasicEvent12	EV13	Constant	Unrepairable	3.12712345E-08		18				
BasicEvent13	EV14	Constant	Unrepairable	3.56123456E-08		18				
BasicEvent14	EV15	Constant	Unrepairable	3.56123456E-08		18				
BasicEvent15	EV16	Constant	Unrepairable	1.56412345E-08		18				
BasicEvent16	EV17	Constant	Unrepairable	2.22212345E-08		18				
BasicEvent17	EV18	Constant	Unrepairable	9.71234567E-08		18				
BasicEvent18	EV19	Constant	Unrepairable	2.96812345E-08		18				

Figura 5.19: Template Excel albero a 19 Basic Events

E' stato così possibile realizzare l'albero in Simulink, mostrato nella seguente figura:

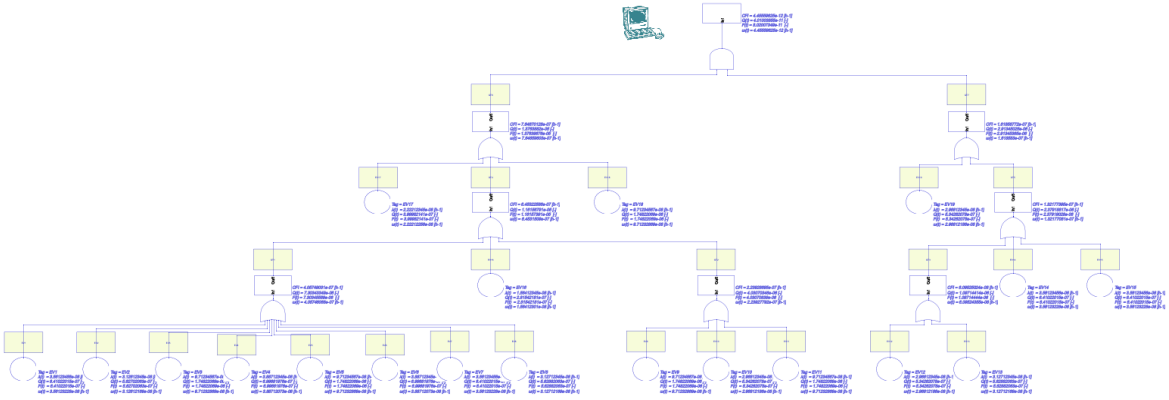

Figura 5.20: Esempio albero 19 Basic Events (Simulink)

Per determinare la correttezza del risultato ottenuto, lo stesso albero è stato realizzato utilizzando FaultTree+; in quanto il programma non consente di mostrare più grandezze contemporaneamente, al contrario di quello realizzato per questo lavoro di tesi in Simulink, è stato necessario inserire quattro figure.

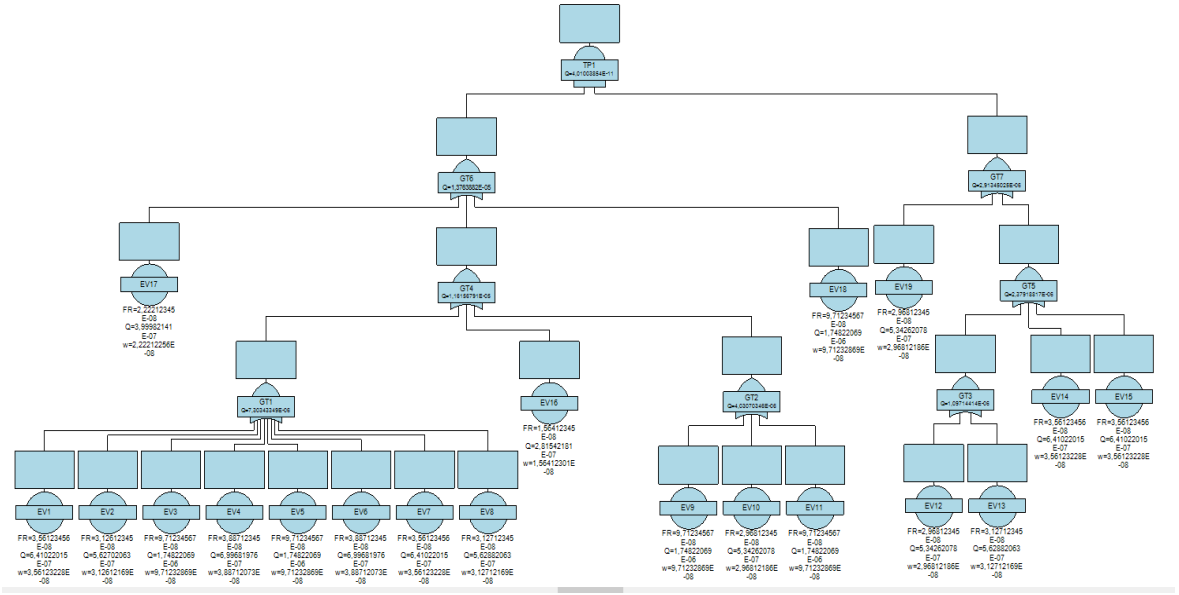

Figura 5.21: Indisponibilità albero FaultTree+

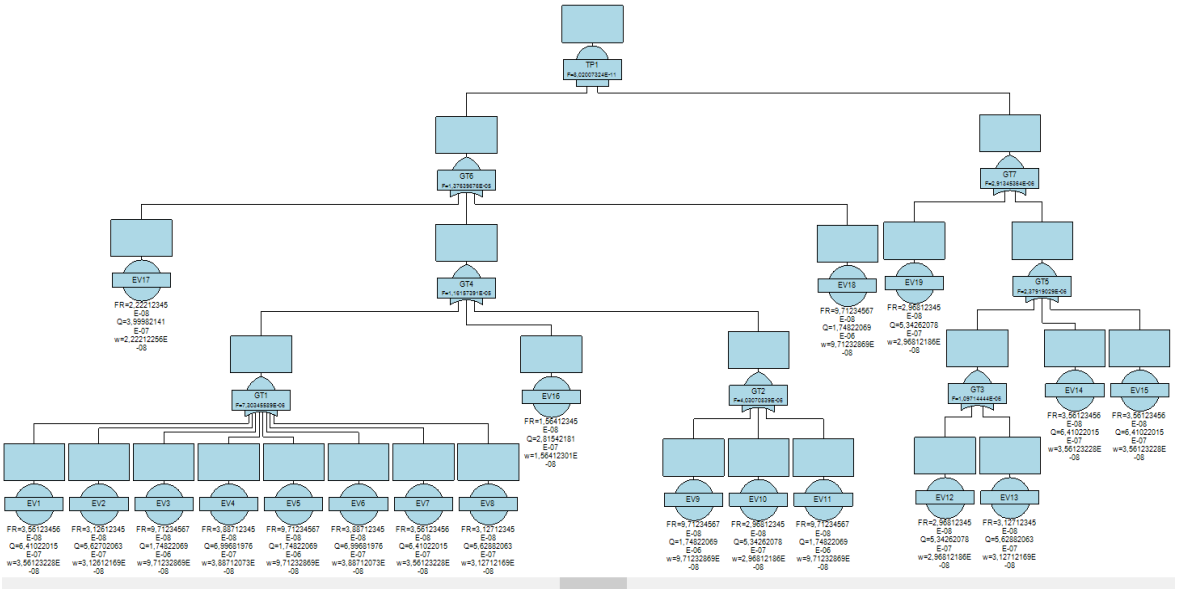

Figura 5.22: Inaffidabilità albero FaultTree+

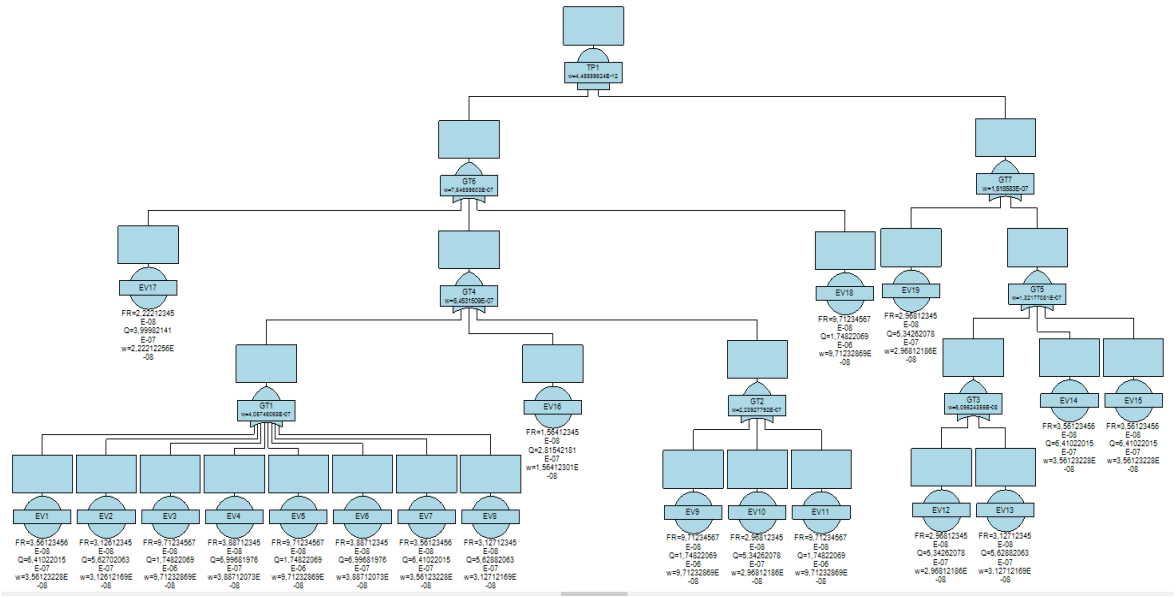

Figura 5.23: Frequenza di guasto albero FaultTree+

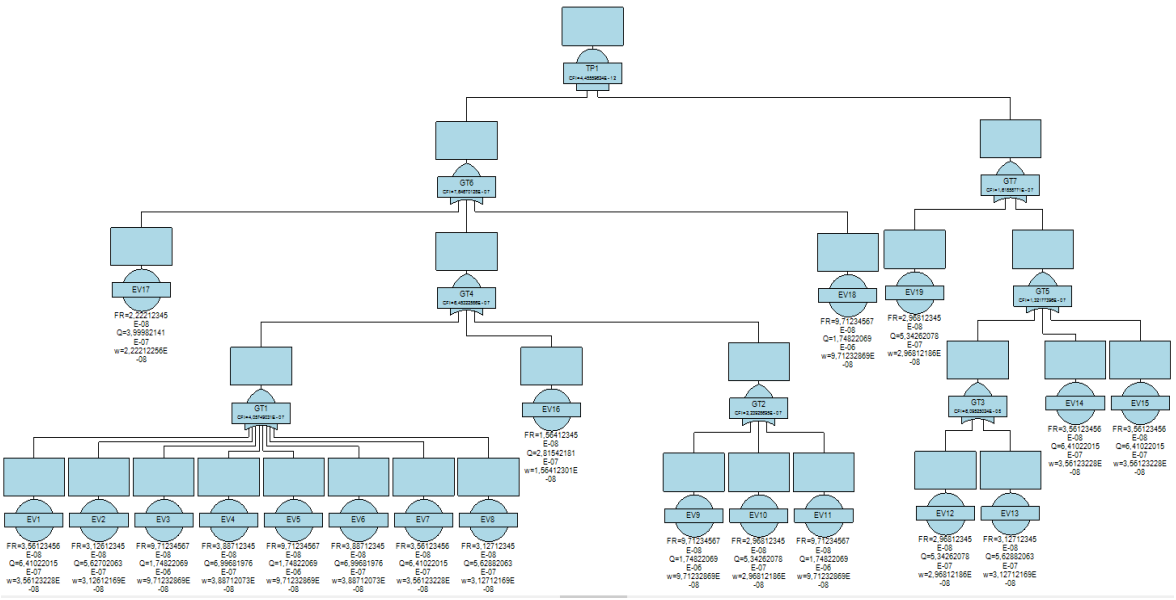

Figura 5.24: CFI albero FaultTree+

Vengono di seguito riportate le tabelle contenenti i valori ottenuti dai due alberi dei guasti per una maggiore chiarezza, seguite da quelle riportanti l'errore relativo tra gli output ottenuti con il programma Simulink e FaultTree+.

Basic Events									
Basic Event ID	Event Label	Model Type	Component Type	Failure rate λ [h^{-1}]	Mission Time t [h]	Q(t) [-]	ω(t) [h^{-1}]	Q(t) [-] FT+	ω(t) [h^{-1}] FT+
BasicEvent	EV1	Constant	Unrepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent1	EV2	Constant	Unrepairable	3.12612345E-08	18	5.62702063E-07	3.12612169E-08	5.62702063E-07	3.12612169E-08
BasicEvent2	EV3	Constant	Unrepairable	9.71234567E-08	18	1.74822069E-06	9.71232869E-08	1.74822069E-06	9.71232869E-08
BasicEvent3	EV4	Constant	Unrepairable	3.88712345E-08	18	6.99681976E-07	3.88712073E-08	6.99681976E-07	3.88712073E-08
BasicEvent4	EV5	Constant	Unrepairable	9.71234567E-08	18	1.74822069E-06	9.71232869E-08	1.74822069E-06	9.71232869E-08
BasicEvent5	EV6	Constant	Unrepairable	3.88712345E-08	18	6.99681976E-07	3.88712073E-08	6.99681976E-07	3.88712073E-08
BasicEvent6	EV7	Constant	Unrepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent7	EV8	Constant	Unrepairable	3.12712345E-08	18	5.62882063E-07	3.12712169E-08	5.62882063E-07	3.12712169E-08
BasicEvent8	EV9	Constant	Unrepairable	9.71234567E-08	18	1.74822069E-06	9.71232869E-08	1.74822069E-06	9.71232869E-08
BasicEvent9	EV10	Constant	Unrepairable	2.96812345E-08	18	5.34262078E-07	2.96812186E-08	5.34262078E-07	2.96812186E-08
BasicEvent10	EV11	Constant	Unrepairable	9.71234567E-08	18	1.74822069E-06	9.71232869E-08	1.74822069E-06	9.71232869E-08
BasicEvent11	EV12	Constant	Unrepairable	2.96812345E-08	18	5.34262078E-07	2.96812186E-08	5.34262078E-07	2.96812186E-08
BasicEvent12	EV13	Constant	Unrepairable	3.12712345E-08	18	5.62882063E-07	3.12712169E-08	5.62882063E-07	3.12712169E-08
BasicEvent13	EV14	Constant	Unrepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent14	EV15	Constant	Unrepairable	3.56123456E-08	18	6.41022015E-07	3.56123228E-08	6.41022015E-07	3.56123228E-08
BasicEvent15	EV16	Constant	Unrepairable	1.56412345E-08	18	2.81542181E-07	1.56412301E-08	2.81542181E-07	1.56412301E-08
BasicEvent16	EV17	Constant	Unrepairable	2.22212345E-08	18	3.99982141E-07	2.22212256E-08	3.99982141E-07	2.22212256E-08
BasicEvent17	EV18	Constant	Unrepairable	9.71234567E-08	18	1.74822069E-06	9.71232869E-08	1.74822069E-06	9.71232869E-08
BasicEvent18	EV19	Constant	Unrepairable	2.96812345E-08	18	5.34262078E-07	2.96812186E-08	5.34262078E-07	2.96812186E-08

Figura 5.25: Risultati caso di studio albero 19 BEs (Basic Events)

Intermediate Events									
Intermediate Event ID	Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] FT+	F(t) [-] FT+	ω(t) [h^{-1}] FT+	CFI [h^{-1}] FT+
IntermediateEvent	GT1	7.30343349E-06	7.30343349E-06	4.05746068E-07	4.05749031E-07	7.30343349E-06	7.30345589E-06	4.05746068E-07	4.05749031E-07
IntermediateEvent1	GT2	4.03070346E-06	4.03070839E-06	2.23927792E-07	2.23928695E-07	4.03070346E-06	4.03070839E-06	2.23927792E-07	2.23928695E-07
IntermediateEvent2	GT3	1.09714414E-06	1.09714444E-06	6.09524355E-08	6.09525024E-08	1.09714414E-06	1.09714444E-06	6.09524355E-08	6.09525024E-08
IntermediateEvent3	GT4	1.16156791E-05	1.16157391E-05	6.45315090E-07	6.45322586E-07	1.16156791E-05	1.16157391E-05	6.45315090E-07	6.45322586E-07
IntermediateEvent4	GT5	2.37918817E-06	2.37919028E-06	1.32177081E-07	1.32177395E-07	2.37918817E-06	2.37919029E-06	1.32177081E-07	1.32177396E-07
IntermediateEvent5	GT6	1.37638820E-05	1.37639676E-05	7.64659603E-07	7.64670128E-07	1.37638820E-05	1.37639676E-05	7.64659603E-07	7.64670128E-07
IntermediateEvent6	GT7	2.91345025E-06	2.91345365E-06	1.61858300E-07	1.61858772E-07	2.91345025E-06	2.91345364E-06	1.61858300E-07	1.61858771E-07

Figura 5.26: Risultati caso di studio albero 19 BEs (Intermediate Events)

Top Event								
Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]	CFI [h^{-1}]	Q(t) [-] FT+	F(t) [-] FT+	ω(t) [h^{-1}] FT+	CFI [h^{-1}] FT+
TOPEVENT	4.01003855E-11	8.02007349E-11	4.45559625E-12	4.45559625E-12	4.01003854E-11	8.02007324E-11	4.45559624E-12	4.45559624E-12

Figura 5.27: Risultati caso di studio albero 19 BEs (Top Event)

Di seguito gli errori relativi tra gli output dei due software.

Basic Events			
Basic Event ID	Event Label	Q(t) error (%)	ω(t) error (%)
BasicEvent	EV1	0.00000000E+00	0.00000000E+00
BasicEvent1	EV2	0.00000000E+00	0.00000000E+00
BasicEvent2	EV3	0.00000000E+00	0.00000000E+00
BasicEvent3	EV4	0.00000000E+00	0.00000000E+00
BasicEvent4	EV5	0.00000000E+00	0.00000000E+00
BasicEvent5	EV6	0.00000000E+00	0.00000000E+00
BasicEvent6	EV7	0.00000000E+00	0.00000000E+00
BasicEvent7	EV8	0.00000000E+00	0.00000000E+00
BasicEvent8	EV9	0.00000000E+00	0.00000000E+00
BasicEvent9	EV10	0.00000000E+00	0.00000000E+00
BasicEvent10	EV11	0.00000000E+00	0.00000000E+00
BasicEvent11	EV12	0.00000000E+00	0.00000000E+00
BasicEvent12	EV13	0.00000000E+00	0.00000000E+00
BasicEvent13	EV14	0.00000000E+00	0.00000000E+00
BasicEvent14	EV15	0.00000000E+00	0.00000000E+00
BasicEvent15	EV16	0.00000000E+00	0.00000000E+00
BasicEvent16	EV17	0.00000000E+00	0.00000000E+00
BasicEvent17	EV18	0.00000000E+00	0.00000000E+00
BasicEvent18	EV19	0.00000000E+00	0.00000000E+00

Figura 5.28: Risultati caso di studio albero 19 BEs (Basic Events)

Intermediate Events					
Intermediate Event ID	Event Label	Q(t) [-] error (%)	F(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
IntermediateEvent	GT1	0.00000000E+00	3.06704118E-06	0.00000000E+00	0.00000000E+00
IntermediateEvent1	GT2	0.00000000E+00	0.00000000E+00	0.00000000E+00	0.00000000E+00
IntermediateEvent2	GT3	0.00000000E+00	0.00000000E+00	0.00000000E+00	0.00000000E+00
IntermediateEvent3	GT4	0.00000000E+00	0.00000000E+00	0.00000000E+00	0.00000000E+00
IntermediateEvent4	GT5	0.00000000E+00	4.20311073E-09	0.00000000E+00	7.56558994E-09
IntermediateEvent5	GT6	0.00000000E+00	0.00000000E+00	0.00000000E+00	0.00000000E+00
IntermediateEvent6	GT7	0.00000000E+00	3.43235248E-09	0.00000000E+00	6.17822544E-09

Figura 5.29: Risultati caso di studio albero 19 BEs (Intermediate Events)

Top Event				
Event Label	Q(t) [-] error (%)	F(t) [-] error (%)	ω(t) [h-1] error (%)	CFI [h-1] error (%)
TOPEVENT	2.49374170E-09	3.11717851E-08	2.24436846E-09	2.24436846E-09

Figura 5.30: Risultati caso di studio albero 19 BEs (Top Event)

In seguito ai risultati ottenuti, si nota che per la maggior parte dei casi si ha, per uno studio a 9 cifre significative, un errore dello 0%, con un massimo di $3.06704118397033*10^{-6}$, rientrando quindi nei parametri definiti dai requisiti della Test Case. Si può quindi considerare il test legato alla corretta propagazione delle informazioni fino al Top Event come passato.

Test albero a 19 BEs				
Test ID	Descrizione	Blocco	Passato/Fallito	Commenti
6	Corretta propagazione numerica fino al Top Event.	IntermediateEvent	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent1	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent2	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent3	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent4	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent5	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	IntermediateEvent6	PASSATO	
6	Corretta propagazione simbolica fino al Top Event (Cut Set).	TOPEVENT	PASSATO	

Figura 5.31: Risultato test propagazione numerica albero 19 BEs

Un ultimo test è stato effettuato per definire il corretto salvataggio delle informazioni dei vari blocchi presenti nei modelli Simulink degli alberi creati in precedenza. L'obiettivo è dimostrare che le informazioni introdotte e propagate rimangono correttamente salvate all'interno delle GUI dei diversi blocchi, anche dopo che il modello è stato chiuso e riaperto. I risultati del test sono riportati nella seguente tabella.

Salvataggio informazioni albero 4.2.5.5.8.(2)			
Test ID	Descrizione	Passato/Fallito	Commenti
7	Corretto salvataggio informazioni del modello.	PASSATO	Tutte le informazioni contenute nei blocchi, dai Basic Events al Top Event, rimangono correttamente salvate all'interno del modello, anche a seguito della sua chiusura e riapertura.

Figura 5.32: Salvataggio risultati (albero 4.2.5.5.8.(2))

Salvataggio informazioni albero 19 Bes			
Test ID	Descrizione	Passato/Fallito	Commenti
8	Corretto salvataggio informazioni del modello.	PASSATO	Tutte le informazioni contenute nei blocchi, dai Basic Events al Top Event, rimangono correttamente salvate all'interno del modello, anche a seguito della sua chiusura e riapertura.

Figura 5.33: Salvataggio risultati (albero a 19 BEs)

Il test è correttamente superato, in quanto i dati vengono mantenuti salvati nelle GUI dei blocchi di appartenenza.

Esportazione risultati dell'albero in Excel

I risultati di Basic Events, Intermediate Events e Top Event ottenuti dall'analisi dell'albero dei guasti possono essere esportati in tabelle in formato Excel al fine di rendere più comoda e pratica la loro lettura per l'utente. Attraverso la schermata del Control Panel infatti è possibile, tramite il comando *FTA Tool–>Excel*, creare un file in formato Excel nel quale salvare gli output ottenuti dall'analisi; l'utente può scegliere a suo piacere la directory dove salvarlo. In questo modo, la lettura dei risultati ottenuti risulta immediata, evitando l'apertura dei diversi blocchi per valutare gli output ottenuti.

BASIC EVENTS				
Basic Event ID	Event Label	Q(t) [-]	F(t) [-]	ω(t) [h^{-1}]
BasicEvent	A_45HKPC1_A	6.41E-07	6.41E-07	3.56E-08
BasicEvent1	B_45HKPC1_B	6.41E-07	6.41E-07	3.56E-08
BasicEvent2	GATE3	6.86E-08	6.86E-08	3.81E-09
BasicEvent3	GATE4	4.77E-06	4.77E-06	2.65E-07
BasicEvent4	C_45KHPC	6.41E-07	6.41E-07	3.56E-08
BasicEvent5	D_45SCP	1.10E-07	1.10E-07	6.12E-09

Figura 5.34: Esportazione risultati in Excel (Basic Events)

ITERMEDIATE EVENTS					
Intermediate Event ID	Event Label	CFI $_{Cut}$ [h^{-1}]	$Q_{Cut}(t)$ [-]	$F_{Cut}(t)$ [-]	$\omega_{Cut}(t)$ [h^{-1}]
IntermediateEvent	IntermediateEvent	7.12E-08	1.28E-06	1.28E-06	7.12E-08
IntermediateEvent1	IntermediateEvent1	3.40E-07	6.12E-06	6.12E-06	3.40E-07
IntermediateEvent2	IntermediateEvent2	4.17E-08	7.51E-07	7.51E-07	4.17E-08

Figura 5.35: Esportazione risultati in Excel (Intermediate Events)

TOP EVENT								
Top Event ID	CFI_{SYS} [h^{-1}]	$Q_{SYS}(t)$ [-]	$F_{SYS}(t)$ [-]	$\omega_{SYS}(t)$ [h^{-1}]	CFI_{SYS} MCS [h^{-1}]	$Q_{SYS}(t)$ MCS [-]	$F_{SYS}(t)$ MCS [-]	$\omega_{SYS}(t)$ MCS [h^{-1}]
TopEvent	5.11E-13	4.60E-12	9.20E-12	5.11E-13	5.11E-13	4.60E-12	9.20E-12	5.11E-13

Figura 5.36: Esportazione risultati in Excel (Top Event)

Come si può notare dal seguente albero, utilizzato per realizzare il test, i valori ottenuti sono stati correttamente esportati e salvati nelle giuste celle del template.

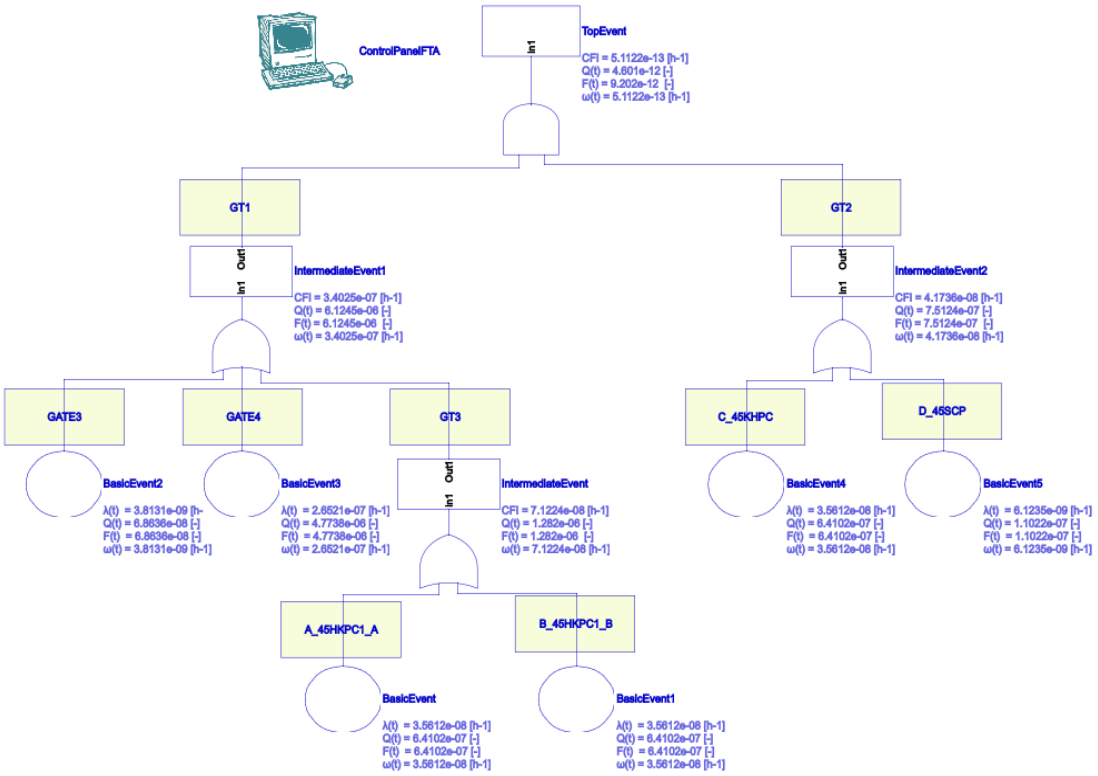

Figura 5.37: Albero utilizzato per il test

Alla luce dei risultati conseguiti, il test può considerarsi superato.

Export informazioni albero 4.2.5.5.8.(2)			
Test ID	**Descrizione**	**Passato/Fallito**	**Commenti**
9	Corretta esportazione degli output dei Basic Events nel foglio Excel a loro dedicato.	**PASSATO**	
9	Corretta esportazione degli output degli Intermediate Events nel foglio Excel a loro dedicato.	**PASSATO**	
9	Corretta esportazione degli output del Top Event nel foglio Excel a lui dedicato.	**PASSATO**	

Figura 5.38: Risultati test esportazione valori

Test di albero presentante Common Cause Failure

Perché la fase di test risulti davvero completa è stato effettuato un ultimo test, rappresentante un albero dei guasti affetto da Common Cause Failure. Come per gli altri tipi di test, l'albero è stato realizzato sia in Simulink che in FaultTree+: di seguito si può vedere il modello realizzato attraverso Simulink, nel quale i componenti affetti da CCF vengono evidenziati in automatico dal programma.

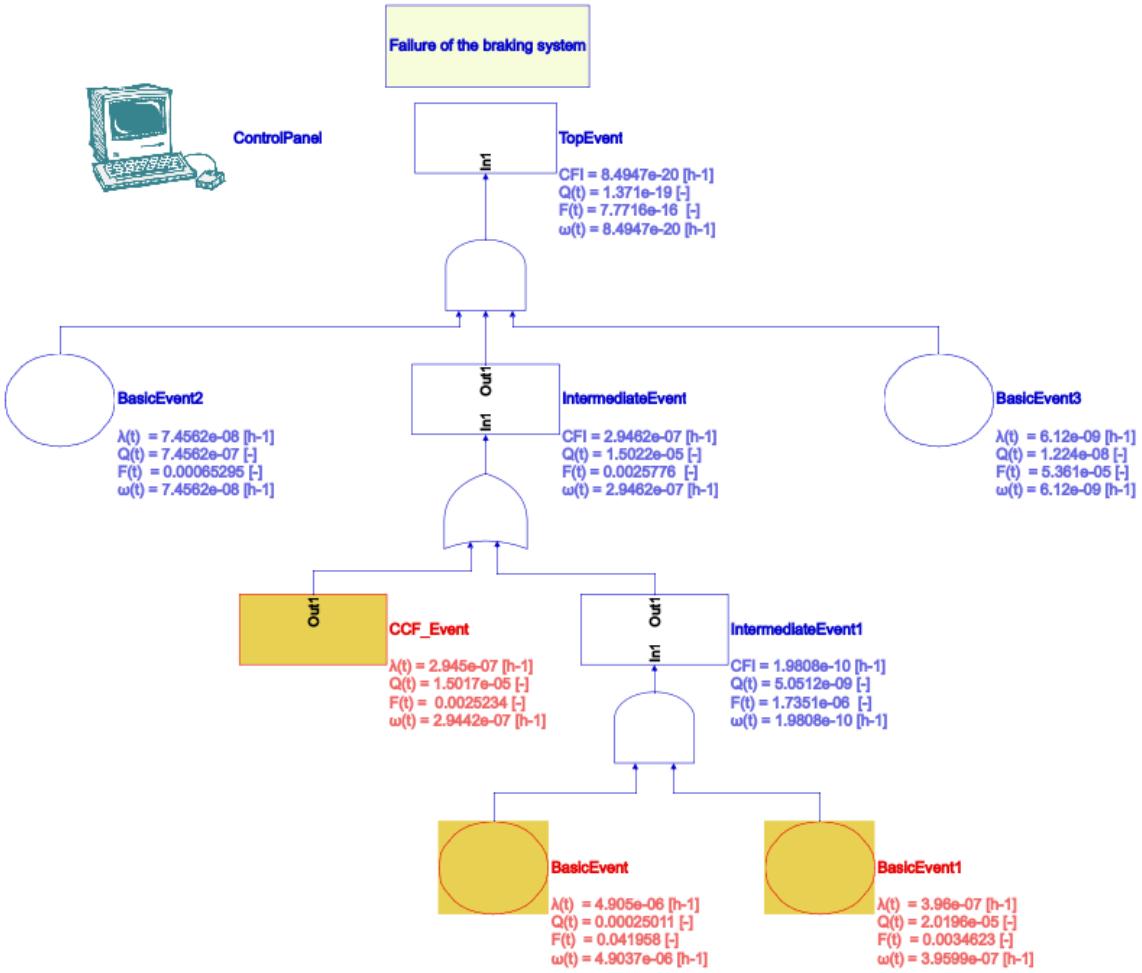

Figura 5.39: Albero con componenti affetti da CCF

La stessa struttura dell'albero non è rappresentabile in FaultTree+, in quanto le impostazioni con cui è realizzato non permetto di inserire graficamente un blocco legato al Common Cause Failure (infatti l'opzione CCF viene settata direttamente all'interno dei blocchi di interesse). Nonostante ciò, i risultati ottenuti dal confronto dei due software, realizzati con le stesse modalità e procedure di quelli precedenti, presentano un errore massimo dell'ordine di grandezza di 10^{-6}; il test risulta quindi passato.

Tutti i test svolti per verificare la correttezza del programma realizzato nel lavoro di tesi, da quelli unitari a quelli di sistema, risultano superati. Preme sottolineare il fatto che, lavorando con un numero di cifre significative pari al massimo consentito da FaultTree+, quasi la totalità dei risultati di comparazione degli output fornisce un errore dell 0%, con un massimo pari a 10^{-6}: ciò significa che nella maggior parte dei casi l'errore relativo è inferiore a 10^{-9}; questo errore probabilmente è dovuto non ad un inserimento errato delle formule nei diversi script Matlab, in quanto queste sono state prese direttamente dal manuale di FaultTree+, ma all'arrotondamento che Matlab attua ai propri valori.
In conclusione, è possibile affermare che il programma realizzato con Simulink fornisce risultati attendibili per alberi di piccole, medie ma anche di dimensioni di una certa importanza. Ovviamente ulteriori test potranno essere realizzati per testarne ulteriormente la bontà (ad esempio concentrando maggiormente l'attenzione su test che coinvolgano componenti riparabili e testati), soprattutto in ottica di futuri lavori volti a sviluppare ulteriormente le funzioni del programma.

Capitolo 6

Conclusioni e futuri sviluppi

Il lavoro di tesi presentato nei precedenti capitoli può essere definito come un primo ma importante passo verso un software che possa davvero competere ad armi pari con programmi commerciali come FaultTree+, i cui costi di licenza sono particolarmente elevati.
Lo sviluppo del tool incentrato attorno Matlab e Simulink è partito da zero: non esistevano infatti funzioni predefinite in grado di calcolare, in ambiente Matlab, grandezze come inaffidabilità, tasso di guasto ed indisponibilità. Non esisteva nemmeno una libreria Simulink che contenesse i blocchi fondamentali utili per rappresentare Basic Events, Intermediate Events ed il Top Event, tantomeno i blocchi più complessi come quello del Common Cause Failure; è stato necessario realizzare tutte e otto le interfacce grafiche attualmente presenti nella libreria, diverse delle quali costituite da centinaia di righe di codice, per poter dare la possibilità all'utente di inserire gli input necessari a realizzare le diverse analisi. Nella fase di impostazione di questo lavoro di tesi non si era ipotizzato che, in soli sei mesi, si potesse ottenere un programma in grado di svolgere tutte le funzioni attualmente disponibili.
Oggi il tool e' dotato di un'interfaccia grafica accattivante e molto intuitiva, che ne permette l'uso senza difficoltà anche ad un utente poco esperto. I valori in input possono essere caricati nei blocchi dell'albero immessi tramite libreria, ma anche inseriti direttamente in un template Excel in formato tabellare. Il programma, tramite un semplice comando presente nel Control Panel, carica il template e lo analizza creando da zero un nuovo modello contenente il numero esatto di Basic Events disposti ordinatamente su più righe con tutte le informazioni necessarie, sia numeriche che simboliche, già inserite. Una volta realizzato graficamente l'albero dei guasti e calcolati gli output degli eventi basici, attraverso un singolo comando è possibile far partire l'analisi numerica e simbolica, ottenendo così in pochissimi secondi tutti i valori legati agli Intermediate Events fino al Top Event. La propagazione delle informazioni potrebbe apparire quasi come una formalità, osservando la semplicità con cui gli output compaiono in poco tempo correttamente a fianco di ogni blocco, ma in realtà questa ha richiesto un elevatissimo numero di ore tra scrittura dei codici e successivi test svolti per dimostrare che i valori si propagassero e si combinassero correttamente. Anche la combinazione dei diversi input non è stata banale da realizzare: seppur le porte logiche utilizzabili siano solo due (AND e OR), l'unione dei valori non è riconducibile ad una semplice moltiplicazione o somma, in quanto il metodo utilizzato per la risoluzione degli alberi dei guasti, il Rare-Approximation, prevede che i risultati siano il prodotto di sommatorie e produttorie tra i diversi Basic ed Intermediate Events. Particolare enfasi deve essere posta riguardo i modelli utilizzabili per la risoluzione degli alberi ed i diversi tipi di componente analizzabili: infatti sebbene la fase di test si sia concentrata in modo particolare sullo studio di alberi costituiti da componenti non riparabili a tasso di guasto costante nel tempo, l'utente può eseguire analisi anche su componenti riparabili o testati, oltre che su modelli Fixed che prevedono la conoscenza come valori di input di inaffidabilità, indisponibilità e frequenza di guasto.
I risultati che si ottengono dall'analisi degli alberi dei guasti, ossia inaffidabilità, indisponi-

bilità, frequenza di guasto e CFI, vengono stampati tutti a schermo ordinatamente di fianco ad ogni blocco: questo è un vantaggio non da poco, in quanto l'utente non è costretto, come invece succede in FaultTree+, a selezionare ogni volta un output diverso ed eseguire nuovamente le analisi.
La fase di test ha infine confermato la bontà del lavoro svolto, in quanto ha dimostrato che i risultati ottenuti sono quasi, se non del tutto, uguali a quelli ricavati effettuando le stesse analisi in FaultTree+. Per poter verificare che il programma realizzato in Matlab e Simulink offra risultati paragonabili al suo corrispettivo della software house Isograph, si è deciso di impostare, anche con un po' di sana incoscienza, la precisione dei risultati del tool Simulink a 9 cifre significative, la precisione massima con cui FaultTree+ può essere configurato. I risultati ottenuti dal confronto tra gli output dei due programmi sono andati oltre le previsioni più ottimistiche, facendo registrare nella quasi totalità dei casi un errore pari allo 0% dopo 9 cifre significative, con un massimo dell'ordine di grandezza di 10^{-6}. Ed il dato assume una rilevanza ancora maggiore se si considera il fatto che i test sono stati effettuati su alberi con un elevato numero di Basic Events (ben 19) e diversi livelli costituiti da Intermediate Events prima di arrivare al Top Event, dimostrando non solo la corretta propagazione numerica dei valori, ma anche che la propagazione dell'errore si mantiene su livelli più che accettabili. Come ogni software che si rispetti, che ha come obiettivo finale quello di semplificare ed automatizzare il più possibile il lavoro dello user, anche per diminuire la possibilità di errore umano è stata implementata direttamente nel Control Panel la possibilità di esportare i risultati ottenuti in un template Excel in forma tabellare, non solo del Top Event, ma anche dei Basic Events e degli Intermediate Events, ordinatamente divisi in fogli separati.
In definitiva, il lavoro svolto nel corso della tesi ha reso possibile verificare sperimentalmente che si possono realizzare blocchi grafici collegati a funzioni in Matlab e Simulink per effettuare analisi con precisione paragonabile a prodotti commerciali, seppur gli sviluppi siano in buona misura di tipo prototipale. Per tal motivo, sebbene i risultati ottenuti siano apprezzabili sotto molti punti di vista, gli sviluppi futuri ai quali il programma può andare incontro per renderlo completo sono numerosi. Infatti, come spiegato ampiamente nei primi capitoli, un albero dei guasti presenta una casistica di blocchi e porte logiche molto ampia: basti pensare al fatto che nella libreria Simulink non sono state introdotte porte logiche come le PRIORITY-AND o le VOTE-GATE, e che alcuni eventi come quelli House non sono stati presi in esame. Lo stesso discorso può essere fatto riguardo ai metodi di risoluzione di una Fault Tree Analysis, in quanto per il presente lavoro di tesi è stato implementato solamente il Rare-Approximation mentre altri metodi, come ad esempio il metodo Esary-Proschan, non sono stati sviluppati: futuri lavori sul programma potrebbero concentrarsi nello sviluppo di tali aspetti, al fine di rendere più completo il tool, fino allo sviluppo di funzioni che implementino metodi statistici, come il metodo Montecarlo, in grado di simulare la reale vita di un componente considerando il tasso di guasto variabile nel tempo.
Un'ulteriore importante miglioria può essere apportata introducendo le Importance Measures, ossia quelle analisi volte ad identificare il peso relativo che ogni singolo componente ha sulle probabilità di guasto dell'intero sistema. Come già detto nel capitolo 5, i test effettuati sul tool si sono concentrati maggiormente sui componenti di tipo non riparabile: è quindi auspicabile che in futuro vengano approfondite le analisi riguardanti i componenti riparabili e testati, al fine di eliminare eventuali errori legati alla propagazione numerica. Il programma Simulink sviluppato in questo lavoro di tesi fornisce all'utente, oltre che risultati dal punto di vista numerico, la propagazione simbolica dei diversi Basic Events, dando inoltre la possibilità attraverso il comando presente nel blocco del Top Event, di ottenere il risultato in forma di Minimal Cut Set. Questa formula potrebbe essere in futuro utilizzata come un input dal quale far realizzare a Simulink in automatico un albero dei guasti "ridotto" secondo il Minimal Cut Set.
Per quel che riguarda il lato economico, ad oggi il programma per funzionare correttamente deve appoggiarsi forzatamente sia a Matlab che a Simulink, con quest'ultimo che può essere

definito "sprecato" in quanto utilizzato come foglio grafico statico, non sfruttandone perciò le potenzialità legate al calcolo dinamico. In realtà il tool può essere implementato esclusivamente in Matlab, anche dal punto di vista grafico: visto che Simulink può funzionare solamente se sul dispositivo utilizzato è installato anche Maltab, ma non vale il contrario, sviluppare il tool solamente in Matlab permetterebbe all'azienda fruitrice dello stesso un notevole beneficio economico, non dovendo compare la licenza di Simulink.
Infine, sempre nell'ottica di rendere il software più conforme agli standard che si prefigge la Model Automation, andranno ulteriormente migliorati i templates Excel sia per importare che per esportare i dati ottenuti dall'analisi dell'albero dei guasti; inoltre, fornire la possibilità di salvare i risultati in formati diversi, magari compatibili con altri software commerciali come FaultTree+, permetterebbe di effettuare analisi anche con dispositivi che non hanno a disposizione licenze software Matlab e Simulink.
Concludendo, l'obiettivo che si prefiggeva l'attività di tesi può dirsi raggiunto, in quanto si è realizzato e verificato sperimentalmente tramite diversi test che si possono ottenere risultati di rilievo riguardanti analisi di alberi di guasto con Matlab e Simulink, permettendo studi legati a sicurezza ed affidabilità utilizzando un tool diverso da quelli in commercio.

Bibliografia

[1] Mun Gyu Park, "RAMS Management of Railway Systems", University of Birmingham, 2013.

[2] Michael Stamatelatos, "Probabilistic Risk Assessment Procedures Guide for NASA Managers and Practitioners", NASA, 2002.

[3] CEI EN 50126, "Railway applications. The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS)", 2000.

[4] IEC 61508-5, "Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 5: Examples of methods for the determination of safety integrity levels", 2010.

[5] IEC 61508-1, "Functional safety of electrical/electronic/programmable electronics afety-related systems - Part 1: General requirements", 2010.

[6] https://elibrary.gsfc.nasa.gov/_assets/doclibBidder/tech_docs/25.%%20NASA_Fault_Tree_Handbook%20Copy.pdf

[7] http://issc2015.system-safety.org/T05_%20Fault_Tree_Introduction.pdf

[8] https://www.isograph.com/

[9] Holly Moore, "Matlab per l'ingegneria", Pearson Prentice Hall, 2008.

[10] Steven T. Karris, "Introduction to Simulink with engineering applications", https://www.u-cursos.cl/ingenieria/2007/1/IQ753/1/material_docente/bajar?id_material=133357, 2006.

[11] https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32014R1302&from=EN

[12] https://www.ife.no/en/ife/main_subjects_new/mto/safety-railways

[13] John E. Bentley, "Software Testing Fundamentals—Concepts, Roles, and Terminology", Wachovia Bank, Charlotte NC. http://www2.sas.com/proceedings/sugi30/141-30.pdf

[14] Padmini C., "Beginners Guide To Software Testing". http://www.softwaretestinggenius.com/download/bgstpadmini.pdf