POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Aerospaziale

Tesi di Laurea Magistrale

Ottimizzazione della schedulazione manutentiva di una flotta di propulsori aeronautici basata su simulazione Monte Carlo

Relatore:
Prof. Maggiore Paolo

Candidato:
Puggioni Marco
s278502

Aprile 2024

Abstract

L'industria aeronautica moderna è caratterizzata da sistemi complessi e costosi, la cui vita media è significativamente lunga, nell'ordine dei dieci-trenta anni. Gli studi rivelano che la quota maggioritaria del Costo totale del loro Ciclo Vita è associata alla fase di Operazione.

Questa fase è generalmente legata alla gestione di una flotta di velivoli, coordinandone le missioni e la manutenzione in modo da massimizzarne la disponibilità e minimizzarne il costo. In particolare, la manutenzione è in gran parte legata alla gestione di uno degli elementi principali del velivolo: il propulsore. Nasce per cui l'interesse per lo sviluppo di un programma ottimizzato di manutenzione per i velivoli nel complesso, e specificatamente per i propulsori.

Al giorno d'oggi, la prassi prevede l'utilizzo di un programma di manutenzione preventiva definito in fase di progetto, in base a regole e pratiche consolidate: tale approccio è noto come Reliability-Centered Maintenance. Questo paradigma, nato alla fine degli anni '70, ha riscosso un notevole successo grazie al focus mantenuto sui problemi manutentivi in fase di progetto del velivolo. Tuttavia, risulta una metodologia "statica" poiché, per sua intrinseca natura, non integra al suo interno informazioni relative all'effettivo stato del sistema di Supporto logistico che regge i processi di operazione e manutenzione.
Inoltre, modernamente il quadro complessivo si è notevolmente ampliato e complicato, con fornitori provenienti da diverse parti del globo, dinamismo introdotto dalla manutenzione on-condition, e requisiti di disponibilità sempre più stringenti: pertanto tali logiche tradizionali vengono messe in crisi, richiedendo una gestione più dinamica.
Attualmente manca un insieme di logiche e strumenti informatici adatti a gestire appieno le complessità di questo problema multidisciplinare: organizzazione dei voli; schedulazione, tempi e personale manutentivi; gestione dei siti manutentivi e dei magazzini sono solo alcune delle tematiche da tenere in considerazione. Per trovare una soluzione, monitorare lo stato effettivo del sistema è essenziale: l'idea è, pertanto, quella di sfruttare le moderne capacità tecnologiche permesse dall'Industria 4.0, che fa della digitalizzazione e della raccolta dati i suoi più incisivi punti di forza.

Questa tesi nasce nell'ambito di un progetto di ricerca per l'ottimizzazione della gestione operativa di una flotta di velivoli, con particolare focus su una flotta di propulsori.
La complessità del problema ha reso necessario lo sviluppo di un software multidisciplinare suddiviso in moduli: uno di essi, il Maintenance Planner, chiamato a gestire dinamicamente l'organizzazione manutentiva, è il modulo software su cui tale tesi si è incentrata.

L'ottimizzazione lavora principalmente sulla rischedulazione degli interventi di manutenzione programmata in modo da adattarla all'effettivo stato del sistema di Supporto logistico, input dagli altri moduli che affiancano il Maintenance Planner. Per valutare la bontà delle soluzioni proposte, l'*Ottimizzatore* si serve dell'azione combinata di altri due elementi: un *Simulatore Monte Carlo*, che riproduce l'attività di missione dei velivoli e la manutenzione dei loro propulsori; e un *Calcolatore dei Costi*, che quantifica economicamente lo scenario simulato.

Nella tesi vengono esposte le caratteristiche teoriche e implementative del Maintenance Planner, testandolo infine su un caso di studio realistico e analizzandone i risultati.

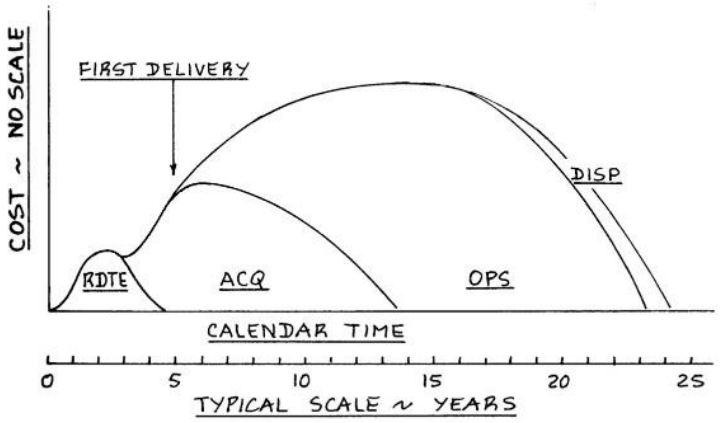
Indice

Capitolo 1 Introduzione

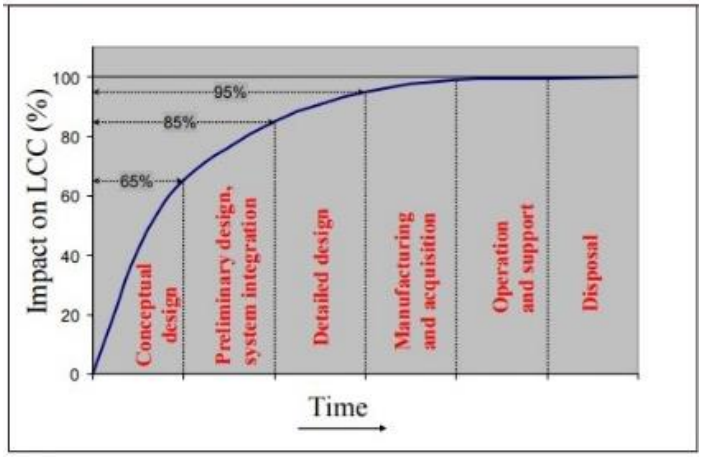
1.1 Il panorama industriale e il ruolo della manutenzione dei propulsori

L'obiettivo ultimo di un qualsiasi settore industriale, tanto per il fornitore quanto per il cliente finale, è ricercare il miglior bilanciamento fra qualità[1] del prodotto (bene materiale o servizio che sia) e il suo costo. Le peculiari discipline di studio e soluzioni al problema dipendono dallo specifico settore considerato e, dunque, dal tipo di prodotto stesso e sistema che ne permette la realizzazione.
Nel caso specifico dell'industria aeronautica, essa è caratterizzata da sistemi complessi e costosi, con una vita media significativamente lunga, nell'ordine dei dieci-trenta anni. In questo contesto, le RAMS (Reliability, Availability, Maintenability and Safety) e la Gestione del Costo sono ad oggi le discipline di maggiore interesse per raggiungere gli obiettivi di settore.

Nell'analisi del problema, una prassi consolidata è quella di suddividere la vita del sistema in momenti principali e studiarne le peculiarità per comprendere il loro impatto sul Costo del Ciclo Vita. Per tali momenti esistono diverse classificazioni, differenti per grado di specificità, ma comunque sempre afferenti alle seguenti quattro: Progetto, Produzione, Operazione (e Supporto), e Dismissione.



(a)



(b)

Figura 1) .a) Ripartizione dei costi nel ciclo vita tipico di un prodotto aeronautico ; .b) Impatto delle decisioni nel ciclo vita sul Costo del ciclo vita [1]

[1] Il termine "qualità" è qui inteso in senso generico con significato di "bontà generale delle caratteristiche di un prodotto". In gergo tecnico, invece, "qualità" ha un significato specifico e si riferisce in particolare al "grado con cui un insieme di caratteristiche di un prodotto o processo rispetta i requisiti".

Come evincibile dalla Figura 1.a, nel settore aeronautico le voci di maggior rilievo sono quelle di Produzione e Operazione. Come mostrato invece in Figura 1.b, le cause principali che vanno a definirne il costo sono dovute a scelte progettuali: per questa ragione, è necessario un notevole sforzo ingegneristico nella attenta e precisa definizione delle proprietà del sistema, visto da entrambi i suoi due lati di prodotto principale (in genere la peculiare macchina operatrice, il velivolo) e suo Supporto Logistico Integrato (ILS).
Nella pratica, la fase di Operazione&Supporto di un sistema aeronautico è generalmente legata alla gestione di una flotta di velivoli, coordinandone le missioni e la manutenzione in modo da ottenerne la massima disponibilità e minimizzarne il costo. In particolare, la manutenzione è in gran parte legata alla gestione di uno degli elementi principali del velivolo: il propulsore. Nasce per cui l'interesse per lo sviluppo di un programma ottimizzato di manutenzione per i velivoli nel complesso, e specificatamente per i propulsori.
A supporto di quanto appena detto, in Figura 2 si mostrano quantitativamente le porzioni di costo relative, rispettivamente, (.a) alla manutenzione[2] rispetto al totale di Operazione&Supporto, e (.b) alla manutenzione del motore rispetto al totale della manutenzione.

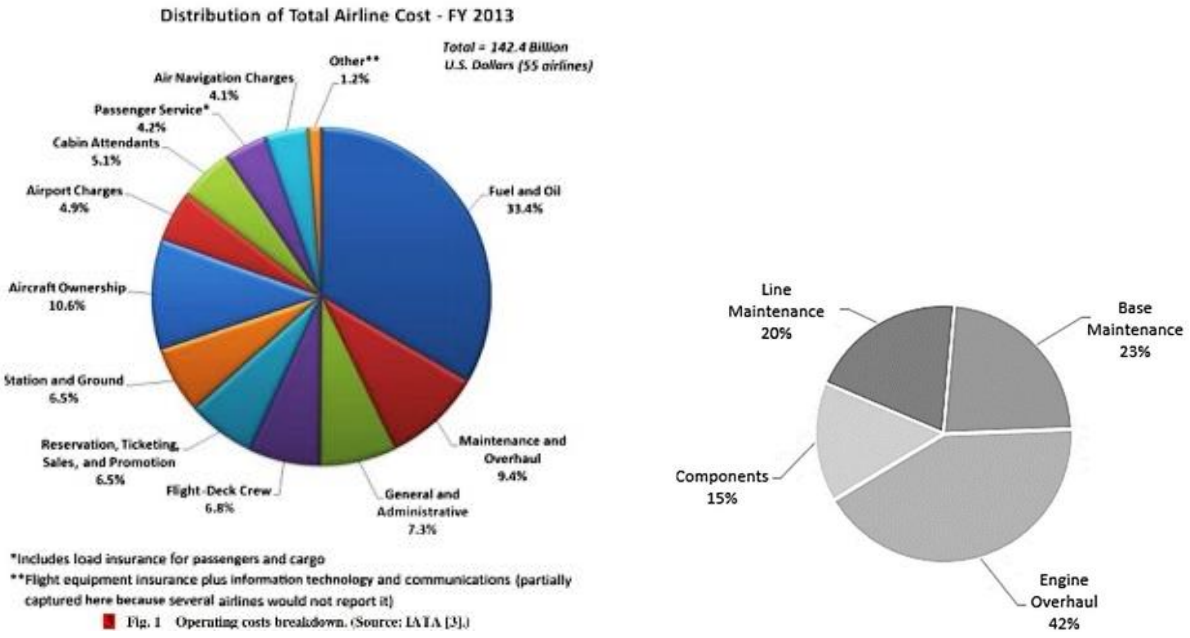


Figura 2) Distribuzione dei costi di…
.a) Operazione & Supporto [2]
.b) Manutenzione & Revisione [3]

1.2 Lo stato dell'arte e l'esigenza di una gestione logistica *dinamica*

Al giorno d'oggi, la prassi prevede l'utilizzo di un programma di manutenzione preventiva definito in fase di progetto, in base a regole e pratiche consolidate: tale approccio è noto come Reliability-Centered Maintenance (RCM). Esso, ufficialmente nato alla fine degli anni '70, ha riscosso un notevole successo grazie al focus mantenuto sui problemi manutentivi in fase di progetto del velivolo, valutando le possibili azioni manutentive sulla base dell'effetto che un certo guasto ha sulla funzione svolta dall'item considerato, piuttosto che (come prima) sulla sua semplice operatività.
L'analisi RCM produce come output un *programma* di manutenzione, ossia un documento che

[2] Per la precisione, in figura è riportata la dicitura "Manutenzione & Revisione" ("Maintenance and Overhaul"). Sebbene spesso la revisione sia concettualmente contenuta nelle operazioni di manutenzione, talvolta (come in questo caso) si parla della manutenzione con l'accezione più specifica di "insieme di attività di *ripristino* dell'operatività", separandola dunque dalle attività di *ispezione*, associate invece più specificatamente alla fase di revisione.

specifica quali azioni compiere sul velivolo (e le sue parti) e quando[3]: come tale, esso non è in sé la fase di supporto logistico all'operazione, ma solo l'insieme di logiche che ne guidano lo sviluppo. Pertanto, una differente problematica è la gestione effettiva della flotta, che si trova ad affrontare un complesso problema multidisciplinare: organizzazione dei voli; schedulazione, tempi e personale manutentivi; gestione dei siti manutentivi e dei magazzini sono solo alcune delle tematiche da tenere in considerazione. I dati a supporto per svolgere tale gestione sono tanti e di diverso tipo (ore volo; dati motore; costi; lead time; ratei di rimozione, di scarto e riparazione; limiti di accettabilità; modifiche di progetto; ecc.) oltre che essere forniti da diversi sistemi e da diverse utenze (e.g. in parte provengono dal campo, in parte dai depot).
Attualmente questa mole di dati viene impiegata per ricavare informazioni utili a definire piani di sbarco e previsioni di tipo logistico che spesso però, basandosi sull'uso di strumenti semplificati, non riescono a tenere conto di tutte le variabili impattanti e condizionano l'implementazione di strategie di manutenzione. Come conseguenza, la metodologia delineata risulta "statica" poiché fatica ad integrare al suo interno informazioni relative all'effettivo stato del sistema di Supporto logistico che regge i processi di operazione e manutenzione.
Inoltre, modernamente il quadro complessivo si è notevolmente ampliato e complicato, con fornitori provenienti da diverse parti del globo, dinamismo introdotto dalla manutenzione on-condition, complessità legate al promettente paradigma High Velocity Maintenance, e requisiti di disponibilità sempre più stringenti: dunque tali logiche tradizionali vengono messe in crisi, richiedendo una gestione più dinamica dell'intero sistema di Supporto Logistico Integrato. Per trovare una soluzione, monitorare lo stato effettivo del sistema (prodotto principale e associato ILS) è essenziale: l'idea è, pertanto, quella di sfruttare le moderne capacità tecnologiche permesse dall'Industria 4.0, che fa della digitalizzazione e della raccolta dati i suoi più incisivi punti di forza, e affiancarle a un robusto sistema automatizzato di analisi dati e ottimizzazione dello scenario logistico.

1.3 La tesi

In un tale contesto nasce questa tesi: essa è stata prodotta nell'ambito di un progetto di ricerca per l'ottimizzazione della gestione operativa di una flotta di velivoli, con particolare focus su una flotta di propulsori, dal punto di vista della disponibilità verso l'utente finale e dei costi di gestione.

1.3.1 Il progetto nel complesso

Sebbene esistano modelli matematici analitici per lo studio di problematiche di affidabilità, disponibilità, schedulazione manutentiva e altre, essi risultano però eccessivamente semplificati e non riescono a sfruttare la mole di dati nel complesso, mostrandosi incapaci di valutare il gran numero di fattori in gioco. Pertanto, la soluzione è stata ricercata nelle potenzialità dei metodi simulation-based, al giorno d'oggi ampiamente impiegati in tutti i settori industriali e permessi dalle nuove tecnologie computer-based sempre più performanti.
In particolare, l'obiettivo finale del progetto è la realizzazione di un software multidisciplinare chiamato *Operation-oriented Logistic-Support Optimizer (OLSO)*, suddiviso in tre moduli:

- Il *Maintenance Planner (MP)*, che si occupa di fornire un piano ottimizzato degli sbarchi motore (Removal Plan) ed un piano manutentivo generale (Maintenance Plan), valutando le problematiche di schedulazione manutentiva (influenzate principalmente da questioni legate alle RAMS) e il loro impatto sulla disponibilità.
- Lo *Spare Parts Manager (SPM)*, il cui scopo è l'ottimizzazione del processo logistico legato all'approvvigionamento delle parti di ricambio necessarie per l'esecuzione dei workscope manutentivi dei motori in servizio, focalizzandosi sia sui tempi che sui costi.

[3] "quando" viene inteso…
- sia in senso -semplicemente- *temporale* (i.e. attraverso condizioni basate su grandezze temporali, e.g. con l'imposizione di sostituzioni o revisioni al raggiungimento di un certo numero di *ore* volate),
- che in senso -più generico- *condizionale* (i.e. attraverso condizioni basate su grandezze non esclusivamente temporali, e.g. con l'imposizione di sostituzioni o revisioni al raggiungimento di un certo numero di *cicli* di operazione).

- Il *TAT*[4] *Forecaster (TATF)*, che definisce e ottimizza il tempo e costo necessario ad effettuare un determinato workscope in base all'effettiva disponibilità della fabbrica in quel momento.

Struttura dei moduli

Ognuno dei tre moduli è a sua volta costruito attorno a tre sotto-moduli: un core di simulazione che riproduce le dinamiche dello scenario trattato; un calcolatore dei costi che lo quantifica economicamente; e un ottimizzatore che gestisce l'ottimizzazione di parametri interni al modulo, e rappresenta dunque un loop di ottimizzazione *interno* (interno al modulo).

In particolare, riguardo i primi due sotto-moduli:

- Nel Maintenance Planner, l'ambiente simulativo (noto come *Simulatore manutentivo*) riproduce il comportamento della flotta di aerei, motori e componenti che operano sottoposti al Maintenance Plan, secondo un certo scheduling dei voli, assoggettati a fenomeni aleatori di usura, failure randomiche e alle dinamiche di magazzino, e rispettanti determinati vincoli e oneri contrattuali. Il Simulatore manutentivo produce come output principale la cronologia degli eventi manutentivi (sbarchi, sostituzioni, riparazioni, acquisti ecc.), da cui estrapolare la disponibilità dei motori.
  In questo modulo, il Calcolatore dei costi si occupa esclusivamente di quantificare eventuali penali ottenute a seguito del non rispetto, da parte del fornitore del servizio di gestione della flotta di motori, degli oneri contrattuali col cliente (in relazione, generalmente, ad una certa disponibilità mensile minima da garantire).
- Lo Spare Parts Manager si occupa di riprodurre le dinamiche di magazzino e fornitori di parti di ricambio, fornendo in output i tempi e i costi di approvvigionamento di tali parti in funzione dello stato attuale del sistema.
- Il TAT Forecaster si occupa di simulare lo stato dei siti manutentivi (soprattutto quelli relativi a manutenzioni/revisioni onerose) e produrre, in output, tempi e costi legati ai processi in essi svolti.

Le logiche di ottimizzazione vengono messe in pratica compiendo analisi di tipo iterativo con l'impiego di tutti e tre i moduli, secondo loop di ottimizzazione implementabili sia internamente al modulo, che esternamente a tutti. L'accoppiamento ottimizzazione-simulazione è pertanto portato avanti secondo paradigma Monte Carlo, e richiede un intenso carico computazionale.

La suddivisione del lavoro in *Development Levels*

Poiché l'OLSO completo è uno strumento complesso e lungo da sviluppare, il lavoro è stato suddiviso in vari step di qualità crescente detti *Development Levels (DLs)*, differenziati per grado di integrazione dei moduli e loro capacità di simulazione, analisi e ottimizzazione. La suddivisione è stata necessaria in modo da poter svolgere analisi anche nelle fasi intermedie del progetto.
Di seguito li si espone brevemente.

DL1) Al primo livello il modulo principalmente sviluppato è il Maintenance Planner, e in particolare il suo Simulatore manutentivo. Degli altri due moduli viene sviluppato una forma semplificata di modello dei costi che possa quantificare in maniera semplice ma efficace le spese relative ai TAT e all'acquisto/riparazione delle parti. In questa fase, il Maintenance Planner riceve in input i dati sugli spares[5] e sui TAT in forma "statica", ossia non aggiornati nel corso dell'ottimizzazione.
Sebbene sia una prima versione, interessanti analisi possono essere compiute già in tal sede, e in particolare, due gruppi di analisi/ottimizzazioni. Le si discutono di seguito.

[4] TAT (Turn Around Time): tempo per il completamento di un determinato processo o soddisfacimento di una richiesta. In tal caso, ad esempio, il tempo impiegato per riportare operativo un item mandato in riparazione, per completare una procedura di installazione motore sul velivolo o revisioni varie, o ancora l'attesa dopo un acquisto di una parte.
[5] "spare", forma concisa per "spare part" (parte di ricambio).

- *Ottimizzazione mediante anticipo:*
  È possibile indagare un primo metodo di ottimizzazione che non richieda la modifica di TAT e spares. Esso è basato sulla rischedulazione manutentiva delle attività programmate in modo da evitare che queste si accumulino nel medesimo periodo, facendo scendere la disponibilità sotto il valore minimo contrattuale. Poiché la rischedulazione avviene anticipando sostituzioni e/o revisioni rispetto al loro "naturale" corso, questa metodologia prende il nome di *ottimizzazione mediante anticipo*.
- *Analisi statiche di sensitività: TAT & spares*
  Seppure non sia possibile far variare in maniera dinamica i TAT e le parti di ricambio, è comunque possibile compiere analisi di sensitività su di essi, e da essi estrarre importanti informazioni sulla risposta dello scenario manutentivo. Tali analisi sono una base fondamentale per le ottimizzazioni più complesse che hanno luogo a DLs successivi, poiché permettono all'analista di familiarizzare con le quantità tipiche in gioco, e di spaziare (staticamente) sullo spazio di progetto in cui si muoveranno le ottimizzazioni più avanzate.

DL2) Al secondo livello si integrano nel Maintenance Planner capacità di analisi di dati di volo, per implementare logiche di predictive maintenance che influenzano il Maintenance Plan modificando dinamicamente le prospettive di vita dei componenti.
Vengono inoltre sviluppate prime versioni dei simulatori degli altri due moduli, ma non ancora integrati tutti assieme.

DL3) Al terzo e ultimo livello vengono completati e integrati i contributi di Spare Parts Manager e TAT Forecaster (a livello di tutti e tre i loro sotto-moduli) di modo che le loro valutazioni in un certo istante modifichino dinamicamente i parametri in input al Maintenance Planner nel corso della sua ottimizzazione. In particolare essi si occupano di valutare la fattibilità delle operazioni manutentive organizzate dal Maintenance Planner, al contempo cercando di ottenerla tramite l'ottimizzazione di leve interne al proprio dominio di competenza.

Per ogni DL viene previsto lo sviluppo dell'associato OLSO: *DL(x) ⇔ OLSO version (x)* .

1.3.2 Localizzazione del lavoro di tesi, obiettivi e risultati

Questa tesi si inserisce nel corso del progetto appena esposto localizzandosi nella prima fase di sviluppo (DL1).

Obiettivi

Il lavoro è stato portato avanti per raggiungere i seguenti obiettivi.

1) Sviluppo dei software: OLSO v.1
   Si richiede lo sviluppo del Simulatore manutentivo e del Calcolatore dei costi, sia in termini di modelli teorici (logiche manutentive e di assegnazione del costo, forma e contenuti dei database, ecc.) che implementazione pratica (script eseguibili), con un grado di dettaglio tale da poter compiere le analisi indicate nella descrizione del DL1 (ottimizzazione per anticipo, e analisi di sensitività a TAT e spares). Al loro fianco, è necessario sviluppare anche l'Ottimizzatore, sia in termini di Logiche (scelta delle leve di ottimizzazione, struttura iterativa ecc.) che di implementazione. Il tutto dev'essere poi integrato in un'unica piattaforma di integrazione dove i vari componenti andranno a formare la struttura che, nel complesso, rappresenta l' *Operation-oriented Logistic-Support Optimizer – versione 1* (OLSO v.1).
   L'OLSO così delineato deve infine poter ricevere gli input "dal campo" tramite fogli di lavoro e deve saper sviluppare, se richiesto, una reportistica per l'esportazione dei risultati in formati comuni (in particolare, fogli di lavoro), utili per la consultazione da parte di operatori umani o per il post-processing di altri software.

2) Analisi:
   Si richiede di svolgere una delle analisi di DL1, delineandone i passi sia a livello concettuale[6] che implementativo. Si è scelto di testare le potenzialità e i limiti della metodologia per anticipo.

Risultati

Il lavoro di questa tesi ha portato alla completamento dei suddetti obiettivi.
I software di Simulazione manutentiva e Calcolo dei costi sono stati scritti in linguaggio MATLAB ed esportati come script eseguibili (estensione *.exe*). Essi si interfacciano con l'esterno tramite file input (database) e output (reportistica) di tipo worksheet, in particolare sfruttando il formato di Microsoft Excel *.xlsx* .
Dell'Ottimizzatore è stata sviluppata la logica, applicata in particolare alla metodologia per anticipo. Per la sua implementazione non è stato sviluppato da zero uno script, per via dell'alta complessità dei problemi in questo campo, che avrebbero altrimenti richiesto uno studio approfondito dedicato (sottraendo risorse al resto del progetto). Piuttosto, si è optato per l'impiego di un modulo di ottimizzazione contenuto all'interno di un software commerciale di integrazione di processo; in particolare, una piattaforma SPDM. Al suo interno è possibile scegliere fra alcuni dei principali moderni metodi di ottimizzazione multi-variabile e multi-obiettivo.
Infine, i tre componenti sono stati integrati in una piattaforma SPDM dedicata, la stessa che ospita il modulo Optimizer.

Per completezza, si chiarisce che questa tesi si è in particolare concentrata sullo sviluppo di Simulatore manutentivo e Calcolatore dei costi nella loro interezza, più le logiche di ottimizzazione. L'integrazione dei processi e l'interfacciamento dell'Ottimizzatore (a livello di implementazione) sono invece stati presi in carico da un partner di progetto. Poiché essenziali per una comprensione del tutto, questa tesi presenterà brevemente anche questi[7].

1.4 Struttura espositiva

L'esposizione verrà suddivisa in tre blocchi:

- In un primo "Basi teoriche a supporto delle analisi", si espongono tutti i concetti teorici alla base delle analisi e degli strumenti implementati in questo lavoro. In particolare, dapprima si discute dei concetti fondamentali dell'ingegneria su come valutare la bontà ad ampio spettro di un sistema, approfondendo il Supporto logistico e la sua influenza sul sistema. La sotto-sezione successiva espone tutte le nozioni utili per comprendere le logiche su cui si basa il Simulatore manutentivo: introduzione sulla disciplina RAMS, trattando le parti che compongono la Dependability (Reliability, Maintainability ed Availability), seguito da un approfondimento sulla moderna manutenzione e i metodi con cui viene approcciata, nella pratica e a progetto. Nell'ultima sotto-sezione si presentano le nozioni utili per comprendere l'operato dell'Ottimizzatore, in particolare dando alcune basi su statistica e ottimizzazione.
- Nel secondo blocco "Caratteristiche dell'OLSO v.1" si espongono più nel dettaglio le proprietà del software, focalizzando l'attenzione sullo sviluppo concettuale (modelli, logiche, ecc.) più che sull'implementazione dettagliata, seppure alcuni richiami ad essa sono presenti qualora siano essenziali per comprendere gli strumenti con cui vengono mostrati i risultati.
- Nel terzo blocco "Caso di studio e risultati" si presenta l'analisi compiuta, con la quale si è testato il metodo di ottimizzazione mediante anticipi. Si mostrano i risultati e li si commentano.

Chiude la conclusione: si ripercorrono obiettivi e risultati di questo lavoro di tesi, commentandoli, e accenna agli sviluppi futuri dell'OLSO.

[6] La *logica di ottimizzazione*.
[7] Anche perché, seppure indirettamente, del lavoro è stato comunque fatto per affiancare il partner di progetto nei processi decisionali e durante l'implementazione discussa.

Capitolo 2 Basi teoriche a supporto delle analisi: concetti di RAMS, costi e metodi matematici

2.1 Il panorama dell'industria aerospaziale e l'importanza del Supporto logistico

Ridotta alla sua essenza, questa tesi tratta dell'ottimizzazione di problemi legati alla manutenzione. Inserita nel progetto di ricerca di cui fa parte, essa è la base (lo step DL1) per costruire il più complesso software Operation-oriented Logistic-Support Optimizer, che tratta numerose altre problematiche inerenti al *supporto logistico*[8].
"Supporto logistico": questo concetto è stato nominato più volte nell'introduzione, ma ancora non è stata data una chiara definizione di cosa sia e in che modo si relazioni col progetto di sistemi complessi. Pertanto, con questo sotto-capitolo introduttivo si vuole mostrare l'importanza del Supporto logistico nello sviluppo di un programma complesso come quelli aerospaziali, contestualizzandolo negli obiettivi principali della progettazione.

Nel seguito si vanno dunque a trattare, dopo alcune definizioni basilari, le proprietà principali con cui si valuta la bontà "a tutto tondo" di un sistema complesso (Capability, Dependability, Safety, Impatto ambientale e Costo), mettendole in relazione fra loro in modo da categorizzarle gerarchicamente. Dopodiché, si tratta di come il Supporto logistico vada ad influenzarle, e in che misura, mostrando in maniera chiara la sua importanza.
Alla fine, sarà chiaro al lettore in che modo questa tesi e i suoi risultati siano connessi con il miglioramento del sistema aeronautico nel suo complesso.

2.1.1 Concetti base dell'ingegneria

Si parta trattando i concetti fondanti di un progetto ingegneristico: "Sistema" e "Ciclo vita".

Sistema

"*Sistema*" è il termine più generico con cui ci si può rivolgere a un qualsiasi ente realizzato per raggiungere un qualche obiettivo, ed è definito come "l'insieme di un certo gruppo di elementi (materiali e non) e le relazioni che fra essi intercorrono". Il sistema è dunque il soggetto, ciò che compie le azioni e le subisce.
Solitamente un sistema complesso (come quelli aerospaziali) si suddivide in due elementi distinti:

- il *prodotto principale*, che svolge in maniera diretta i compiti (in gergo, le *funzioni*) per cui il sistema è stato creato; e
- il *supporto logistico*, che affianca il primo assicurandogli tutto il necessario per funzionare, rispettando i requisiti di sicurezza e rispetto ambientale nel modo più efficace ed efficiente possibile. Poiché in questo contesto il supporto viene inserito nella definizione stessa del sistema, si è soliti parlare di "supporto logistico *integrato (ILS)*[9] ".[10]

Nel caso aerospaziale il prodotto principale va a coincidere quasi completamente con la *macchina* peculiare: un velivolo nel caso aeronautico, un satellite (o altre macchine peculiari) nel caso spaziale. Riguardo l'ILS, poiché esso è inteso come supporto ad ampio spettro, tante e diversificate sono le azioni che lo riguardano: servizio clienti e gestione del personale sono due esempi di supporto non

[8] Da cui, in effetti, arriva il nome stesso.
[9] Precisamente, il "supporto logistico" è un concetto generale; invece, l'ILS è un insieme di attività specifiche e codificate, creato per chiarire i passi da seguire nello sviluppo di un programma complesso per ottimizzarlo sotto il punto di vista del supporto logistico.
Grammaticalmente, "ILS" è un nome proprio, non un nome comune; viceversa per "supporto logistico".
[10] Poiché in ultima analisi questa tesi tratta di ottimizzazioni che riguardano il supporto logistico (non il prodotto principale), alla fine di questo sotto-capitolo vi si dedica una sezione di approfondimento, così da averlo più chiaro nel complesso e, così, comprendere meglio anche come questa tesi si inserisce nel quadro complessivo.

direttamente legato al sistema principale (l'aereo); piuttosto, fra tutte la categoria più di interesse è quella della *manutenzione* del sistema primario e azioni di supporto ad essa, come fornitura e gestione di parti di ricambio, scrittura e gestione dei manuali di manutenzione, approvvigionamento e gestione degli strumenti ed equipaggiamenti di manutenzione, training dei manutentori ed altri.

Ciclo vita

Il **Ciclo vita** (Life-Cycle) è invece il concetto che scandisce temporalmente e tematicamente le azioni compiute dal sistema durante tutta la sua vita.
Esso viene canonicamente suddiviso in quattro fasi principali: Design (concettuale, preliminare e di dettaglio), Produzione, Operazione (& Supporto) e Dismissione. Si sottolinea che esse non avvengono in successione netta una dopo l'altra, ma è possibile che si *intersechino temporalmente*. Ciò vale fra le fasi di produzione-operazione, e operazione-disposal, mentre è tipico che la fase di design, specie quella dell'hardware (s'intende strutture e sistemi fisici), sia terminata prima dell'inizio della produzione: il motivo è l'alto costo associato alla riprogettazione, che rende necessario il modus "fatto una volta e fatto bene". In conclusione, però, si fa anche notare che esistono importanti eccezioni a questa regola e sono legate a un tipo di acquisizione (i.e. progettazione + produzione) detta *Evolutionary approach*, anche se circoscritte a particolari categorie di sotto-sistemi (tendenzialmente software) e prodotti/applicazioni (in genere, nel campo militare).

System Life-Cycle

Dopo averli definiti separatamente, si veda ora il *System Life-Cycle*, ossia l'entità data dall'unione dei due. Viene mostrata nella Figura 3.

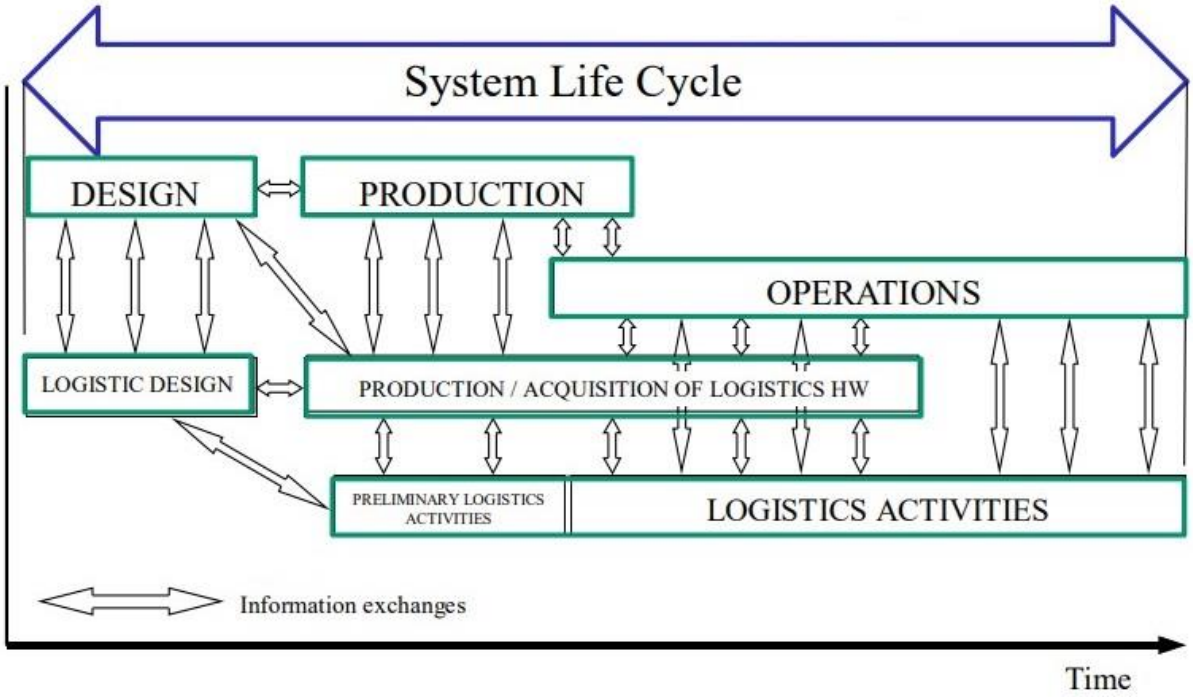


Figura 3) System Life-Cycle

Si possono distinguere chiaramente i rami facenti capo alle due facce del sistema: il *prodotto principale* e il *supporto logistico (integrato)*. Per ognuno dei due rami si riconoscono le fasi del ciclo vita, applicate/specializzate in base al caso particolare.
Da notare è come le fasi scambino informazioni fra loro, influenzandosi a vicenda: non si tratta dunque di compartimenti stagni e ciò è proprio tipico dei sistemi complessi. Si può notare anche la sovrapposizione parziale delle fasi del ciclo vita come si era sottolineato in precedenza.

2.1.2 Overall System Evaluation: gli obiettivi della progettazione

Compresi i concetti fondanti dell'ingegneria, si trattano ora quali siano gli obiettivi, ossia le parti salienti in cui si suddivide un progetto. Trattiamo in particolare di questioni nell'ambito aerospaziale, ma questi concetti si applicano anche più in generale al mondo dell'ingegneria nel complesso.

In generale

Ponendoci a una visione di ampio spettro, possiamo in generale riassumere le questioni legate alla progettazione suddividendo le proprietà principali nei seguenti gruppi, in base alle caratteristiche matematiche delle variabili che le rappresentano (negoziabili → obiettivi, non negoziabili → vincoli/requisiti) e alla loro desiderabilità (favorevole = desiderabile , sfavorevole = non desiderabile):

- *proprietà negoziabili favorevoli*, ossia desiderabili per il cliente (e quindi anche per il produttore), rappresentate di norma da variabili che si desidera massimizzare ;
- *proprietà negoziabili sfavorevoli*, ossia non desiderabili per il cliente (e quindi anche per il produttore), rappresentate di norma da variabili che si desidera minimizzare ;
- *requisiti non negoziabili*, sui quali cliente e produttore (anche volendo) non hanno possibilità di influire, poiché imposti dalla legge. Essi sono espressi tramite l'imposizione del rispetto di un certo standard minimo, ossia variabili che devono risultare maggiori/minori di una certa soglia minima/massima, pena il non ottenimento dei certificati di aeronavigabilità.

A compendio, si fa notare che la normativa, oltre che esprimersi tramite vincoli netti si/no, spesso ricorre all'imposizione di *tasse* in base a quali siano le performance del sistema riguardo una certa proprietà che essa reputa desiderabile (e.g. riguardo l'inquinamento), in modo da spingere "gentilmente" verso la sensibilizzazione ma senza imporre rigide soglie (anche se un qualche requisito minimo è quasi sempre presente). In quest'ultimo caso, comunque, queste variabili figurano come (sfavorevoli e) su cui il produttore ha una scelta, per cui verrebbero messe nella categoria precedente (negoziabili sfavorevoli).

In particolare

Nel particolare, queste proprietà sono canonicamente individuate dalle seguenti grandezze.

- Le **PROPRIETÀ FAVOREVOLI** sono:
  - ***Capability (C):*** quantifica la performance del sistema supposta la sua possibilità di operare. Quali performance sono da considerare nello specifico dipende da quale sia la particolare applicazione considerata. Ad esempio:
    a) nel caso di applicazioni aeronautiche civili, una quantificazione del payload (numero, volume, peso), il tempo per compiere una certa tratta, la lunghezza minima di decollo o atterraggio, ecc. ;
    b) nel caso di applicazioni aeronautiche militari, la rapidità con cui svolgere determinate missioni, peso e tipologia di payload trasportabile, velocità massima, capacità inerenti il combattimento e la manovrabilità, performance di guida e navigazione in condizioni avverse o ridotte, e altre capacità avanzate (stealth, capacità cooperativa, abilità di surveillance ecc.).
  - ***Dependability* (D):** misura nel complesso quanto si possa fare affidamento sul sistema, ossia la sua capacità di garantire al cliente le funzioni per cui è stato creato. Si tratta di una grandezza ad ampio spettro, che coinvolge vari concetti al proprio interno, in particolare quelli di:
    - *Reliability*, che misura l'abilità di un elemento di poter svolgere la propria funzione, sotto certe condizioni, per un determinato intervallo temporale (ma con accezione più particolare della dependability);
    - *Maintainability*, che misura la bontà del supporto manutentivo; e

- *Availability*, che quantifica l'abilità di un elemento di essere in uno stato tale da svolgere la propria funzione.

Esistono differenti tipologie di reliability, ed essa e la maintainability non sono indipendenti dall'availability. Andiamo dunque di seguito ad approfondire questi concetti.

- *Despatch Reliability* (R_d)*:* quantifica l'abilità del sistema di riuscire ad iniziare la missione quando chiamato a farlo. Di solito espressa matematicamente tramite una probabilità. Insieme alla Mission Reliability concorrono a influenzare l'affidabilità operativa.
- *Mission Reliability* (R_m)*:* quantifica l'abilità del sistema di riuscire a portare a termina la missione supposto di averla iniziata. Di solito è espressa matematicamente tramite una probabilità. Insieme alla Despatch Reliability concorrono a influenzare l'affidabilità operativa.
- *Basic Reliability* (R_b)*:* quantifica l'abilità del sistema di non incorrere in guasti (di qualunque natura), stanti determinate condizioni, per un determinato intervallo temporale.
- *Maintainability (M):* quantifica la bontà del supporto manutentivo, ossia la sua tempestività, la rapidità nell'esecuzione e la capacità di eseguirla correttamente.
- *Availability (A):* quantifica abilità di un elemento di essere in uno stato tale da svolgere la propria funzione sotto certe condizioni ad un certo istante | per un certo intervallo} di tempo supposto che vengano fornite le risorse esterne necessarie. Di solito è espressa matematicamente tramite una probabilità. Availability, Maintanability e Reliability sono quantità legate fra loro, e di solito la prima viene espressa in funzione delle altre due.

- Le **PROPRIETÀ SFAVOREVOLI** sono identificate tipicamente da una sola variabile che le pesa nel complesso: il ***Life-Cycle Cost*** (LCC), ossia una quantificazione monetaria del costo richiesto per "avere" (i.e. ottenere e possedere) il sistema, ossia progettarlo, produrlo, operarlo e dismetterlo.

- Infine, i **REQUISITI NON NEGOZIABILI**, imposti -come detto- dalla normativa (i.e. dalla legge) sono riguardanti la Sicurezza (Safety) e l'Impatto ambientale.
  - ***Safety:*** (intesa come proprietà) valuta il grado di libertà di un sistema da condizioni di *rischio*. Il rischio, inteso come livello associato a una *situazione*, è la misura del suo grado di indesiderabilità, valutato in base alla sua probabilità di accadimento e delle sue possibili conseguenze negative.
    Se inteso relativamente ad un *sistema*, il risk-level è una misura complessiva del grado di indesiderabilità di tutte le situazioni che caratterizzano l'operatività del sistema (i.e., le sue funzioni e l'ambiente in cui è inserito).
    Per completezza, "safety" è tradotto in italiano con "sicurezza", ma non è l'unico modo di intenderla. Al suo fianco anche si pone il differente concetto di *security*, ma quest'ultimo tratta di eventi negativi dovuti a deliberate azioni umane di sabotaggio, e seppure faccia parte del quadro delle questioni da tenere in conto, non rappresenta nella pratica un problema di cui ci si preoccupa approfonditamente nella progettazione del velivolo e suo supporto logistico. Pertanto, non lo tratteremo.
  - ***Impatto ambientale:*** valuta l'impronta del sistema sull'ambiente. Per un sistema aeronautico, misure tipiche sono la quantità di inquinanti gassosi (CO2, soot, ossidi d'azoto, solfuri, ecc.), non gassosi (olio idraulico, liquido refrigerante, carburante, ecc.) e inquinamento acustico.

Nel complesso

Nel complesso, la bontà del sistema viene valutata pesando gli effetti favorevoli rispetto a quelli sfavorevoli, stante il rispetto dei requisiti non negoziabili. Tradizionalmente la grandezza che rappresenta questo trade-off è detta ***System Effectiveness*** ed è definita come:

$$SE \overset{\text{def}}{=} \frac{C \cdot D(\bar{R}_{\mathrm{d}},\ \bar{R}_{\mathrm{m}},\ A(\bar{R}_{\mathrm{b}},\ M))}{LCC} = \frac{C \cdot \bar{R}_{\mathrm{d}} \cdot \bar{R}_{\mathrm{m}} \cdot A(\bar{R}_{\mathrm{b}}, M)}{LCC}$$

Relazione fra le proprietà principali

Si è mostrato che le proprietà fondamentali con cui valutare un sistema sono Capability, Dependability, Safety, Impatto ambientale e Costo. Si precisa che non si tratta di discipline indipendenti fra loro, ma al giorno d'oggi vanno a formare una struttura progettuale fortemente integrata, richiedendo una visione d'insieme e attenta gestione dei processi nel corso del life-cycle.

Classificazione gerarchica:

Circa la loro classificazione in ordine di ***importanza***, il peso relativo di queste parti del progetto cambia a seconda del momento storico e dell'applicazione particolare considerata[11]. La classificazione stessa inoltre dipende dai pesi con cui si valuta: uno dei criteri più utili è quello di quantificare l'importanza attraverso lo sforzo progettuale (in termini di tempo e risorse, dunque costo) richiesto per soddisfare le richieste del progetto, sia quelle del cliente (capability e dependability) che quelle normative (sicurezza e impatto ambientale).
Ad esempio, in ambito d' aviazione al giorno d'oggi:

- Nel *trasporto civile*, la sicurezza occupa senz'altro il primo posto della scala gerarchica. Il motivo è che è l'unico vincolo non negoziabile, poiché portato mediante severi requisiti imposti dalle autorità di competenza attraverso atti di legge e normative: non riuscire a soddisfare i -peraltro stringenti- requisiti di sicurezza significa non ottenere i certificati che ne permettono legalmente l'operazione del sistema, con le ovvie conseguenze del caso.
  Le problematiche di dependability sono fondamentali e seguono la safety per importanza. Poter contare su un sistema altamente disponibile permette di massimizzare gli introiti, con le intuitive conseguenze positive su un sistema basato sul profitto come quelli di tale settore. Allo stesso modo, la minimizzazione dei costi è un punto parimenti essenziale.
  Le questioni di impatto ambientale sono sempre più importanti al giorno d'oggi, ma rimangono secondarie poiché tendenzialmente più circoscritte e dipendenti dai sotto-sistemi singoli e meno dal sistema integrato.
- Negli ambiti *militare* e *operazioni speciali*, invece, il peso relativo della capability è superiore rispetto al trasporto civile, e si ha un rilassamento dei requisiti di sicurezza a causa dei contesti molto differenti che si stanno trattando, in termini sia di funzioni da svolgere che di ambiente operativo.
  La dependability rimane invece un fattore determinante, tendenzialmente guadagnando di peso rispetto al costo, seppure nelle ultime decine di anni anche l'ambito militare sta vedendo una forte attenzione anche su questo aspetto.
- In ambito di *categoria generale*, la richiesta del cliente di contenimento dei costi del sistema porta a rilassare i requisiti di Sicurezza. Infatti la tipologia di clienti e le loro capacità di spesa in questa categoria sono profondamente differenti da quelle delle grandi compagnie aeree del trasporto civile e dei ministeri della difesa del militare.
  Allo stesso modo, la richiesta di economicità esclude una gestione "stressata" della manutenzione, per la quale servono complicate infrastrutture e logiche di supporto, portando a livelli di dependability più bassi.

A livello di ***complessità e presenza nel progetto***, intesa come l'estensione e complicatezza dei processi in termini di temporali e organizzativi, Sicurezza e Costo sono i due driver principali di un progetto, poiché richiedono un complesso framework organizzativo e un'attenta gestione dei processi (di design, produttivi e operativi) per assicurare il rispetto degli obiettivi nel corso di tutta la vita del

[11] Esempi in aviazione: categoria generale, trasporto civile (passeggeri o merci), ambito militare, operazioni speciali (search and rescue, spegnimento incendi, ecc.) .

sistema. Non è un caso che nel corso del tempo tali strutture siano state approfonditamente indagate e perfezionate, andando di pari passo con la presa di coscienza della loro importanza da parte di legislatura/enti-normativi e aziende di progetto, portando alla creazione della disciplina *Safety Management* per la Sicurezza e una gestione del progetto detta *Design to Cost* per il Costo.
A seguire, e in similitudine, le problematiche di Dependability hanno anch'esse guadagnato l'attenzione generale e anche per loro è stato delineato un paradigma di progetto che vi ponga il focus: esso è conosciuto come *Design to Reliability.*

Infine, si fa notare che Safety e Dependability possiedono metodi per essere indagate che per molti versi si sovrappongono. La disciplina che se ne occupa è nota come RAMS:

RAM (Reliability, Availability, Maintainability) = Dependability ,
\+ S (Safety).

2.1.3 Il Supporto logistico e la sua influenza sulla System Effectiveness

Con la precedente sezione, si è terminata la presentazione delle nozioni principali per definire un sistema e valutarne la bontà. Per chiudere questa sezione introduttiva, rimane da mostrare in maniera chiara in che modo il Supporto logistico, uno dei due elementi fondamentali di un sistema complesso e soggetto dell'ottimizzazione di questa tesi, vada ad influenzare la bontà del sistema stesso (i.e., il suo impatto sulla System Effectiveness).
In particolare, l'esposizione viene portata avanti facendo riferimento specifico all'Integrated Logistic Support (ILS): un insieme di attività specifiche e codificate, creato per chiarire i passi da seguire nello sviluppo di un programma complesso di modo da ottimizzarlo sotto il punto di vista del supporto logistico. L'ILS rappresenta il paradigma moderno[12] con cui gestire le problematiche di supporto logistico, per cui descriverlo risponderà indirettamente anche alle questioni sul Supporto logistico che si vogliono mostrare (e che portano alla scrittura di questo paragrafo), dando in aggiunta interessanti informazioni sullo stato dell'arte delle metodologie progettuali.
Si vedrà che l'ILS si compone di numerosi aspetti, ma non tutti vengono impattati dalle ottimizzazioni di questo lavoro di tesi (e più in generale, dell'OLSO). Pertanto, in chiusura, viene mostrato in particolare su quali prodotti dell'ILS il lavoro di tesi, e l'OLSO, hanno effetto.

2.1.3.1 ILS – Integrated Logistic Support

L'Integrated Logistic Support è l'insieme delle attività, progettuali e fattuali[13], volte all'ottimizzazione del supporto logistico.
Più nello specifico, esso si occupa di:

- (Quando) intervenire durante tutto il ciclo-vita di un sistema, sia influenzando e guidando il progetto ingegneristico nella fase di design, che supervisionando e supportando le fasi di produzione, operazione e dismissione ;
- (Come) sviluppando strategie, ottenendo risorse umane e materiali ;
- (Perché) puntando alla riduzione del LCC e mantenendo, al contempo, il determinato livello di prestazione richiesto.

L'ILS può essere impiegato sia in relazione a un prodotto ancora da sviluppare, seguendolo fin dalle prime fasi del progetto, sia applicato a prodotti già esistenti per valutarne la bontà sotto il profilo del supporto logistico.

[12] Più correttamente, è *uno dei* paradigmi moderni. Un altro esempio è l'IPS (Integrated Product Support), anche se condividono numerosi punti in comune.
[13] Ossia, l'insieme di processi volti alla definizione del sistema e la sua conseguente realizzazione pratica.

Per analizzare l'operato dell'ILS, risulta comodo scomporlo nelle due porzioni che trattano i seguenti due aspetti[14]:

- Quando → l'ILS nel progetto ingegneristico (la System Engineering)
- Come → le attività (dette "prodotti") dell'ILS

Gli si dedica, nel seguito e a ciascuno, un paragrafo di breve approfondimento.

L'ILS nel progetto ingegneristico (la System Engineering)

Come detto, l'ILS è un insieme di attività che si inseriscono nel corso della vita di un programma (solitamente costoso e complesso), in particolare influenzandone il progetto. Vale la pena dunque riassumere brevemente quali siano queste fasi in cui si suddivide il processo per lo sviluppo di un prodotto, nel complesso noto come *System Engineering*, e indicare quali attività di supporto logistico gli si associno secondo i dettami del paradigma ILS.

System Engineering process	ILS Task/Product
Project planning	System engineering plan, ILS Plan, ILS Element Plan
Definizione dei requisiti	Concept of Operations, Concept of Support, Scenario Analysis
Analisi dei requisiti	Scenario and What-if analysis, ILS Requirements HFE/HSI Requirements
Progetto	Trade off studies, Risk Analysis, Product Element Requirements, Life Cycle Cost Analysis (LCCA), Preliminary Safety Analysis
Implementazione	Realizzazione di prodotti dell'ILS come addestramento, documentazione, procedure e accordi per la fornitura, LSA, LCCA, Human Factors Analysis
Integrazione	Realizzazione di prodotti dell'ILS come addestramento, documentazione, procedure e accordi per la fornitura, LSA, LCCA, Human Factors Analysis
Verifica	Verifica del soddisfacimento dei requisiti fissati relativi al prodotto e al supporto logistico
Transizione	Completamento prodotti logistici
Validazione	Valutazione dell'Availability e verifica dei requisiti
Fase operativa	Fornitura del supporto logistico durante la vita operativa; In Service Data Analysis (ISDA)
Manutenzione	Svolgimento di operazioni di manutenzione
Dismissione	Disposal Concept, trattamento rifiuti pericolosi

[14] Che non a caso sono simmetrici rispetto alla presentazione iniziale. Non viene approfondita la sezione sugli obiettivi ("perché") in quanto essi sono gli stessi già discussi in precedenza in relazione alla System Effectiveness.

Non si entra ulteriormente nel dettaglio delle fasi di System Engineering, poiché di interesse marginale per gli scopi di questa tesi. Per ulteriori informazioni, si possono consultare le fonti bibliografiche in calce al documento.

I prodotti dell'ILS

Le attività svolte e i prodotti realizzati dall'ILS sono dipendenti dalla particolare fase e prodotto considerati. Infatti, se per un nuovo prodotto potrebbe essere opportuno sviluppare l'intero gruppo di elementi ILS, per uno già esistente sarà meglio concentrarsi sul fornire una risposta a bisogni precisi.

Si indica con *Tailoring* il processo di adattamento della specifica metodologia di implementazione dell'ILS: si individuano le attività che devono essere svolte dipendentemente dalla dimensione e complessità del progetto, così come da eventuali accordi contrattuali.
La prima fase del Tailoring consiste nella categorizzazione delle attività ILS coerentemente con i vincoli e i requisiti progettuali. In particolare, si distingue in attività obbligatorie, opzionali e non applicabili. Successivamente, si procede all'allocazione delle risorse necessarie per le attività obbligatorie: il bilancio tra risorse necessarie e disponibili costituisce un vincolo per la programmazione delle attività opzionali. Ciascuna di tali attività deve sempre essere valutata nell'ottica costi-benefici. Si termina quindi con la selezione definitiva di attività che saranno svolte.

Dunque, l'ILS consiste nell'integrazione di tutte le funzioni e le attività necessarie per lo sviluppo del supporto al prodotto. Il paradigma ILS suddivide tali attività in 12 categorie codificate, che insieme costituiscono i cosiddetti *prodotti* dell'ILS. Le si presentino di seguito, dandone una breve spiegazione.

#	ILS product	Definizione, Obiettivi, Note varie
1	Computer resources	Identifica e pianifica le risorse hardware e software, la documentazione e il personale necessari alla gestione di sistemi HW e/o SW critici per il progetto.
2	Design influence	Consiste nell'integrazione dei parametri i quantitativi della System Engineering con i prodotti dell'ILS. Esso riflette l'interazione dei parametri di progetto con i requisiti di supporto. In tal modo i requisiti di supporto sono mutuati per assicurare che il prodotto raggiunga la desiderata Availability ed il raggiungimento dell'ottimizzazione e bilanciamento dei costi del prodotto e del supporto.
3	Facilities & Infrastructure (F&I)	Si tratta di strutture permanenti e semipermanenti necessarie per supportare ed utilizzare il prodotto. Sono necessari degli studi appositi per definire il tipo di strutture e la quantità di spazio, così come le condizioni ambientali, di sicurezza e gli equipaggiamenti necessari. Data la grande quantità di tempo, nonché i costi, necessari per l'acquisizione e/o realizzazione delle F&I è opportuno considerarle fin dalle prime fasi della progettazione.
4	Maintenance	Stabilisce i requisiti di manutenzione del prodotto lungo il suo Life Cycle. Tale fattore ha un grande impatto sulla pianificazione e sullo sviluppo degli altri elementi che costituiscono la catena di supporto logistico. Il suo obiettivo principale è pianificare e implementare la miglior strategia manutentiva per assicurare il requisito di disponibilità del prodotto al minor costo possibile.
5	Manpower & Personnel	Ha come obiettivo l'identificazione e la pianificazione del personale e del relativo livello di qualifica richiesto allo svolgimento delle varie task. In particolare, è necessario operare adeguatamente sia il prodotto che l'equipaggiamento necessario al completamento della missione e delle operazioni di supporto.

6	Packaging, Handling, Storage and Transportation (PHS&T)	Accanto alle attività associate con la manutenzione e riparazione del prodotto, ci sono alcuni aspetti supplementari che devono essere considerati. Questi, pur non essendo direttamente correlabili ad altri elementi dell'ILS, sono molto importanti per il corretto uso del prodotto.
7	Product Support Management	Consiste nella gestione del supporto al prodotto.
8	Supply Support	Rientrano in tale prodotto tutte le attività volte alla pianificazione e gestione dell'acquisizione delle risorse necessarie all'assicurare un supporto efficace al minor costo possibile. Ciò si traduce nell'avere disponibili i giusti ricambi, con il giusto tempismo e al giusto prezzo. È quindi necessario provvedere sia all'approvvigionamento iniziale che al rifornimento dell'inventario.
9	Support equipment	Tale prodotto mira all'acquisizione di tutto l'equipaggiamento di supporto necessario alle operazioni e alla manutenzione, assicurando quindi che il prodotto soddisfi i requisiti di Availability al minor costo possibile.
10	Sustaining Engineering	Essa sostiene il prodotto nel suo ambiente operativo, basandosi su alcune attività di analisi volte all'assicurare la continua operatività del prodotto. Inoltre, riguarda la risoluzione di mancanze individuate durante il suo ciclo vitale. Si rivela molto utile per individuare modi per migliorare le prestazioni. Le attività ad essa correlate sono legate alla verifica di mancanze tecniche del prodotto con annessa analisi delle possibili azioni correttive.
11	Technical Data	Si tratta di informazioni di natura scientifica o tecnica, escludendo però software o informazioni di carattere gestionale. Rientrano in tale elemento tutte quelle attività volte alla raccolta di tali informazioni ed alla produzione (revisione) di pubblicazioni tecniche.
12	Training and Training Support	L'obiettivo di tale prodotto è identificare le necessità di addestramento ed implementare una strategia per addestrare il personale in modo da fornirgli la capacità di operare e supportare il prodotto durante il suo intero ciclo vitale assicurando quindi un livello di prestazioni ottimale sia del prodotto che dell'apparato di supporto.

In conclusione, si segnala che ad ognuno dei prodotti discussi si associano una o più analisi peculiari. Comunque la loro trattazione esula dai fini semplificati di questa sezione, per cui si rimanda nuovamente alle fonti bibliografiche per ulteriori approfondimenti.

2.1.3.2 Il lavoro di tesi inserito nel quadro del Supporto logistico

Ridotta alla sua essenza, questa tesi si basa sull'ottimizzazione di problemi legati alla manutenzione. Inserita nel progetto di ricerca di cui fa parte, essa è la base (lo step DL1) per costruire il complesso software Operation-oriented Logistic-Support Optimizer. Dal suo nome stesso si evince che esso tratterà numerose altre problematiche inerenti al supporto logistico, e in particolare essi saranno specificamente relativi alla fase di operazione più che a quella di acquisizione (i.e. progetto e produzione).

Con riferimento ai 12 prodotti dell'ILS, si elencano quali di essi vengono trattati dall'OLSO (versione finale):

- Manutenzione
- Approvvigionamento (supply)
- Siti manutentivi
- Personale
- Attrezzature
- Dati tecnici

2.2 Basi di RAMS con focus sulla Dependability

In questo sotto-capitolo si va ad esporre alcuni concetti fondamentali nell'ambito della disciplina RAMS, e mantenendo in particolare il focus sulle problematiche di Dependability; in altre parole, sulla parte "RAM – Reliability, Availability, Maintainability" (escludendo dunque la Safety, disciplina che non ha influenzato le analisi di questa tesi).
Si conduce l'esposizione seguendo appunto le tre sotto-discipline citate.

Prima di iniziare, una precisazione: si tenga presente che le RAMS sono una disciplina ingegneristica ad ampio respiro, che al suo interno usa i risultati di numerose altre discipline. Fra esse spiccano, per importanza e presenza: probabilità e statistica, analisi sistemistica, analisi del rischio e fattori umani[15].

2.2.1 Definizioni

Come per tante altre discipline tecniche, anche le RAMS hanno un dizionario con numerosi termini dal significato specifico. Di seguito si vedano i principali per questa tesi.

Nota: si riportano i termini in inglese poiché la nomenclatura tecnica nasce in tale lingua;
inoltre, per alcuni di essi è difficile una traduzione univoca in italiano.

Terminologia data per assodata:

Generale:

- Being (ente), Element (elemento), Set/Group (insieme/gruppo);
- Property (proprietà), Attribute (attributo), Characteristic (caratteristica), Quality (qualità), Feature (tratto), Aspect (aspetto), Factor (fattore);
- Status (status), Situation (situazione), Condition (condizione);
- Occurrence (accadimento), Happening (avvenimento), Event (evento) [non tecnico];
- To… Cause (causare), Affect (produrre un effetto), Influence (influenzare);
- Circumstance (circostanza), Context (contesto), Framework (~circostanza), Condition (condizione), State (stato), Situation (situazione), Scenario (scenario).

RAMS-related[16]:

- Part, Component, Assembly, Equipment, Sub-system, System, Item[17];
- Function;
- Analysis, Assessment, Qualitative, Quantitative, Minimize;
- Redundancy.

Generici:

Event: (evento) [tecnico]
è un accadimento interno o esterno che la cui origine è distinta dal velivolo (e.g. condizioni atmosferiche).

[15] Questi ultimo punto, i fattori umani, sono tipicamente considerati in minor parte rispetto ai primi in relazione alle problematiche di Dependability, mentre assumono un importante ruolo nel caso della Safety.
[16] Per le definizioni, vedere la normativa AC 23.1309-1E .
[17] Si dà per scontata la definizione. Si vuole comunque ricordare il loro ordine gerarchico:
Part < *Component* < *Assembly* < *Equipment* < *Sub-system* < *System* ;
Item: indica un ente di livello gerarchico generico; tendenzialmente uno fra { Part → Equipment } .
Element: un qualsiasi ente che sottintende l'appartenenza a un gruppo gerarchicamente superiore.
Quindi, uno fra tutti meno il livello system.

Accadimenti negativi relativi all'umano:

Error: (errore)
è un'omissione o un'azione scorretta compiuta da un membro della crew o da personale manutentivo, oppure ancora un mistake in requisiti, design o implementazione.

Accadimenti negativi relativi ai componenti – Focus sull'operatività:

Fault: (-stato di- guasto)
è uno "stato di guasto", ossia uno stato caratterizzato dall'inabilità di operare come richiesto.
Si precisa che la fault è di solito specificatamente associata a, e detta di, un elemento di basso livello gerarchico, come un componente o un equipaggiamento, per i quale si identificano *funzioni* di basso livello.

Failure: (-evento di- guasto)
è un accadimento che ha effetto sull'operazione di un componente, parte o elemento in modo che esso non possa più funzionare come richiesto (intendendo sia la perdita di funzione che malfunzionamento).

Latent Failure: (failure latente)
Failure che non è stata resa nota alla flight crew o al personale manutentivo.

Malfunction: (malfunzionamento)
è la failure di un sistema, sottosistema, unità o parte che ne preclude la normale/usuale operazione; l'accadimento di una condizione in cui l'operazione eccede specifici limiti.

Failure condition: (condizione di guasto)
è una condizione avente un effetto sull'aereo e/o sui suoi occupanti, sia diretto che consequenziale, che è causato o contribuito da una o più failures o errori, considerando la fase di volo ed eventuali condizioni operative avverse, o ambientali, o eventi esterni.

Circostanze negative:

Adverse operating condition: (condizioni operative avverse)
un insieme di circostanze operazionali o ambientali applicabili al velivolo, combinate con failures o altre situazioni di emergenza che risultano in un incremento significativo del carico di lavoro della flight-crew rispetto al normale.

Effetti negativi:

Adverse effect: (effetto negativo)
è una risposta del sistema che risulta in un operazione non-desiderabile di un velivolo, sistema o sotto-sistema.

Altre:

In conclusione, si fa notare che tanti altri termini esistono e sono molto importanti nell'ambito RAMS, ma che non definiamo poiché i concetti associati non sono stati influenti nel corso della tesi. Per completezza, se ne elencano alcuni senza darne definizione:

- Conformance (conformità), Non-conformance (non-conformità), Anomaly (anomalia), Defect (difetto) ;
- Severity (gravità), Severity Level/Number (livello/numero di gravità), Criticality (criticità), Criticality Level/Number (livello/numero di criticità) ;
- Harm (danno), Injury (lesione, danno fisico), Damage (danno materiale) ;
- Incident (incidente), Accident (sinistro) ;
- Caution (~segnalazione), Warning (~segnalazione), Hazard (pericolo) .

2.2.2 Reliability

2.2.2.1 Introduzione

La Reliability[18] (letteralmente "Affidabilità") nasce, come disciplina a sé, negli Stati Uniti degli anni '50 in contesto militare, quando la complessità crescente dei sistemi elettronici stava iniziando a generare failure-rates che risultavano in availability fortemente ridotte e alti costi. L'esigenza di una soluzione spinse allo studio matematico del problema, approcciato inizialmente come un esercizio di probabilità applicata volto a studiare la probabilità di failure del sistema note le distribuzioni di probabilità degli elementi che lo compongono: il risultato è la scoperta che l'affidabilità del sistema, ossia la sua probabilità di essere funzionante ad un dato momento temporale, è essa stessa tempo-dipendente. Tutti i sistemi si guastano, prima o poi.

Nel tempo la presa di coscienza dell'importanza di questa disciplina ha spinto verso studi sempre più approfonditi e multidisciplinari. Al giorno d'oggi l'approccio probabilistico è solo una delle facce con cui si presenta la Reliability e, seppure rappresenti ancora il cuore del problema e delle metodologie quantitative, differenti figure professionali vi si interfacciano in maniera peculiare al proprio punto di vista:

- il matematico la vede come un esercizio di probabilità e statistica applicata ;
- il manager tenta di organizzare un Reliability-group, che spesso si rivela essere un'organizzazione staff ;
- l'esperto di controllo-qualità vede l'affidabilità come un'estensione dei propri sforzi ;
- il component-engineer è interessato a ricercare i componenti più affidabili ; mentre
- il system-engineer cerca soluzioni architetturali affidabili, semplificando il design.

Per arrivare a soluzioni progettuali valide, nessun approccio preso singolarmente è soddisfacente, ma devono concorrere tutti in sinergia fra loro.

In termini di influenza, la Reliability interessa ormai qualsiasi campo industriale , sia in termini di prodotti che di processi, soprattutto in relazioni a sistemi complessi. Temporalmente, si tratta di un problema cosiddetto "birth-to-death" per il sistema, interessandolo cioè dal conceptual design alla dismissione.

2.2.2.2 Tipologie di reliability

Esistono diversi tipi di affidabilità a seconda del punto di vista considerato. Esse sono state già incontrate nella sezione introduttiva sulle proprietà principali di un sistema, dove le erano state espresse a parole. Se ne riprenda la definizione, stavolta in forma probabilistica:

- *Basic Reliability* (R_b)*:* probabilità che il sistema non incorra in guasti (di qualunque natura) ;
- *Despatch Reliability* (R_d)*:* probabilità che il sistema non incorra in guasti che possano ritardare o precludere la partenza ;
- *Mission Reliability* (R_m)*:* probabilità che il sistema non incorra in guasti (critici o maggiori)[19] che possano impedire il compimento della missione, supposto che essa sia iniziata (probabilità condizionata) ;
- *Safety (S):* probabilità che il sistema non incorra in guasti critici durante una missione.

[18] Preferenzialmente, ci si riferirà con "Reliability" (in inglese, maiuscolo) alla disciplina, mentre con "affidabilità/reliability" (in italiano/inglese, minuscolo) alla proprietà.

[19] "*Critical*, *Major*, *Minor*, *Negligible*" sono le quattro categorie di *severity* in cui vengono catalogate le failures. Si tratta di una parte che è stata tralasciata, poiché ha per lo più influenza sulla Safety e in questa tesi hanno ruolo marginale.

Sebbene l'ultima categoria non sia canonicamente considerata un'affidabilità in senso tradizionale, viene riportata per mostrare la continuità logica fra affidabilità e sicurezza, che giustifica l'affermazione fatta qualche paragrafo scorso, secondo cui Reliability e Safety sono due discipline che possiedono numerosi punti in comune nel modo in cui vengono indagate.

Ad ogni tipo di affidabilità (safety inclusa) si associa una tipologia di failure. Le si mostrano nella Figura 4.

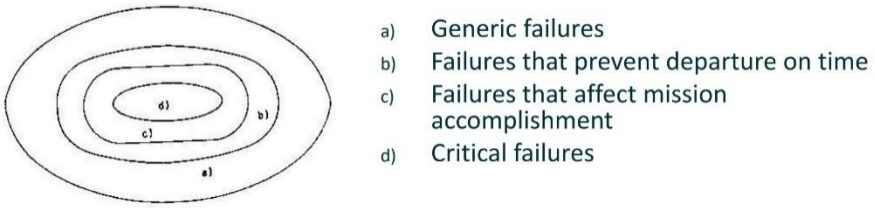


Figura 4) Tipologie di failure, e relazione fra il loro grado di probabilità e gravità

Si noti che non è una semplice e fortuita coincidenza il fatto che più le failures sono gravi e meno sono probabili: questa relazione è dovuta agli sforzi ingegneristici per limitare il rischio complessivo (quantità che pesa insieme probabilità e gravità di una failure).

2.2.2.3 Stima dell'affidabilità

Per poter governare le problematiche di Reliability, il passo fondamentale è quello di riuscire a stimare l'affidabilità di un sistema. Concettualmente si possono distinguere due metodi per trovare una soluzione a questa problematica:

1) (No Rlbt data) Arch $\xrightarrow{\text{Direct system testing}}$ System Rlbt (*estimated*)
2) Items Rlbt + Arch $\xrightarrow{\text{Calculation}}$ System Rlbt (*predicted*)

Approccio n.1:

Il primo è un metodo basato interamente sulla stima sperimentale: senza nessun dato di affidabilità pregresso, si *stima* attraverso esperimenti (e metodi matematici di indagine statistica) l'affidabilità globale del sistema. Questo approccio è fattibile però solo con sistemi molto semplici, gerarchicamente dell'ordine di componenti singoli o al limite equipaggiamenti. Per sistemi più complessi come sotto-sistemi, sistemi o addirittura velivoli completi la procedura è assolutamente improponibile, e per diverse ottime ragioni:

- Anzitutto, il sistema dev'essere costruito prima di poterlo testare. Nella quasi totalità dei casi però le analisi di Reliability sono interessate a stimarla per poter progettare il sistema stesso, che pertanto non è ancora stato definito. Chiaramente non è pensabile, per via di costi e tempistiche eccessivi, costruire svariati prototipi per ogni architettura da sottoporre a giudizio e poi scegliere.
- Anche supponendo che l'architettura sia stata in qualche modo fissata, i costi rimarrebbero troppo elevati per il semplice motivo che (per intuitive ragioni) i test sono di tipo distruttivo: ogni failure mode da indagare richiede lo scarto di componenti, con costi associati. L'anti-economicità dell'approccio è dovuta al grande numero di failure modes da dover indagare che possono caratterizzare un sistema, specie se complesso. Inoltre la situazione si aggrava ulteriormente, fino alla completa infattibilità pratica, considerando che molte delle condizioni rilevanti ai fini della sicurezza (e più in generale per affidabilità diverse da quella base) considerano scenari con failure multiple: se già indagare tutti i failure modes singolarmente è irragionevole, enormemente di più lo è pensare di testare anche tutte le loro possibili combinazioni.

- Infine, persino supponendo un ridotto numero di failure modes e loro combinazione, il numero di test da dover condurre sarebbe significativamente più alto di tali combinazioni, e questo è inevitabile per ottenere risultati statisticamente significativi.

Approccio n.2:

Dunque, per i motivi sopra citati, la procedura di primo tipo non è realizzabile se non per componentistica molto semplice. Come conseguenza, il secondo metodo è quello quasi sempre impiegato:

a) si parte dalla conoscenza pregressa di dati di affidabilità di elementi semplici (a loro volta ottenuta mediante iterazione dello stesso processo 2 a livelli gerarchicamente inferiori, oppure mediante metodo di tipo 1), unita alla conoscenza o ipotesi di una certa architettura ;

b) poi, attraverso calcoli di tipo probabilistico, si *predice* l'affidabilità globale.

Per la precisione, una tale metodologia è realizzabile in forma completa solo quando si considerano failure modes singoli oppure failure multiple non-interagenti. Per failure multiple è matematicamente necessario considerare anche le probabilità condizionate poiché, nella pratica, esse non sono sempre nulle[20]. Tuttavia, mentre si dispone dei dati di affidabilità dei componenti, le probabilità condizionate sono quantità non direttamente legate ai singoli elementi e dunque, non possedendole direttamente, vanno stimate: esse però dipendono da complesse questioni di accoppiamento legate all'architettura funzionale e quella fisica (ridondanze, problemi di installazione ecc.), alle condizioni di operatività e persino alle procedure di emergenza, che richiedono di considerare persino l'operatore umano inserito nel sistema.
Di nuovo, le difficoltà del metodo non risiedono tanto nella soluzione del problema matematico noti i dati (per la quale gli studi di calcolo di probabilità sono ormai maturi e ben noti), quanto piuttosto nella stima dei dati stessi.

Le difficoltà che emergono dalle problematiche di analisi delle failure multiple di tipo common-cause vengono approcciate con appositi strumenti di analisi, peculiari delle discipline RAMS. Non le tratteremo nel dettaglio, ma per darne un'idea se ne elencano le principali: Common Cause Analysis, Common Mode Analysis, Particular Risk Analysis, Zonal Safety Analysis.

Struttura espositiva seguente:

Cosa tratteremo:
Ai fini di questa tesi, le nozioni di probabilità che sono di interesse sono quelle base riguardanti la modellazione delle failures stesse: in pratica, variabili randomiche, distribuzioni di probabilità e parametri notevoli. Le si va dunque a trattare nel seguente paragrafo.

Cosa non tratteremo:
Tralasciamo invece tutta la parte di calcolo di probabilità composite (teoremi di joint & total probability, Bayes, ecc.) e modelli di affidabilità (e.g. Bayes, Binomiali, Stand-by, Multi-phase, Multi-mission, ecc.). Allo stesso modo, eviteremo tutto l'insieme di analisi tipiche delle RAMS, di cui le common-cause analyses sono solo una parte (altri esempi sono FMEA, FMECA, FTA, Check-List Analysis, What-if Analysis, HAZOP study, Ishikawa/Fishbone Analysis, Relationship plot, ecc.). Infine, tralasciamo anche i metodi per la stima delle affidabilità dei componenti in condizioni differenti da quelle nominali (Parts stress analysis, Parts count, ecc.).

[20] Le failures multiple problematiche non sono quelle che avvengono come casuale sovrapposizione di failure, problema questo probabilisticamente semplice da studiare. Piuttosto, lo sono le failures cosiddette "common cause", ossia per le quali il presentarsi di una aumenta (o diminuisce) la probabilità che se ne verifichi un'altra: le due failure vengono così accoppiate matematicamente da una quantità detta *probabilità condizionata*. Come si vede di seguito a questa nota, è proprio la stima di questa quantità ad essere difficoltosa, e richiede strumenti di analisi dedicati.

2.2.2.4 Basi di teoria della probabilità applicata alla Reliability

In questo sotto-paragrafo si vogliono discutere alcuni concetti fondamentali della probabilità matematica, poiché è su questa disciplina che si fondano le basi rigorose per gli studi di Reliability. In particolare, si vuole arrivare a mostrare le basi teoriche dietro la modellazione delle failures: le cosiddette *distribuzioni di probabilità*, studiandone gli elementi costituenti, la definizione, i parametri con cui vengono descritte, e infine passando in rassegna le principali distribuzioni impiegate nelle applicazioni di affidabilità.

Concetti teorici dati per assodati

Nella trattazione che segue, si considerano noti alcuni concetti base di probabilità, quali: spazio di campionamento Ω, campo dell'evento $\mathcal{F}$, il concetto di probabilità in sé, eventi semplici e composti, probabilità condizionata e indipendenza, e infine le regole fondamentali di teoria e calcolo di probabilità (teoremi della somma, moltiplicazione e della probabilità totale).

Variabili randomiche e Funzioni di distribuzione

Se il risultato di un esperimento con esito casuale è un numero (reale), allora si definisce tale quantità *variabile randomica (v.r.)*. Tali variabili sono spesso indicate con lettere greche come $\tau,\ \zeta,\ \varepsilon$.
Ci sono vari modi per definire una legge di distribuzione.

Cumulative Density Function (CDF) / (Cumulative) Distribution Function: F
Tra i più usati vi è la *funzione di distribuzione (cumulativa)* $F(t)$ della v.r. τ, definita come

$$F(t) = \Pr\{\, \tau \le t \,\}, \quad -\infty < \tau, t < \infty \,.$$

Per ciascun t, $F(t)$ restituisce la probabilità che la variabile random assuma un valore $\le t$.
Proprietà:

- Dato che $s > t$ implica $\{\tau \le t\} \le \{\tau \le s\}$, si tratta di una funzione non decrescente ;
- $F(-\infty) = 0$ e $F(\infty) = 1$;
- $F(t)$ è continua da destra ;
- La probabilità che una v.r. τ stia nell'intervallo $(\mathrm{a}, \mathrm{b}]$ è $Pr\{\, \mathrm{a} < \tau \le \mathrm{b} \,\} = F(\mathrm{b}) - F(\mathrm{a})$.

In ambito di Affidabilità, si assume la v.r. $\tau > 0$ come *failure-free time*, ed essa è distribuita secondo $F(t) = \Pr\{\tau \le t\}$ con $F(0) = 0$. La funzione $R(t) \stackrel{\text{def}}{=} 1 - F(t)$ rappresenta la *survival probability* ed è detta *reliability (function)*.

Probability Mass/Density Function (MDF/PDF): p, f
In base alla tipologia di v.r. è possibile definire una funzione di densità (continua o discreta).

- *V.r. discrete*: *MDF*
  una variabile randomica τ è discreta se può solo assumere un numero finito di valori, ossia se c'è una sequenza $t_1, t_2, \ldots$ tale che $p_k = \Pr\{\, \tau = t_k \,\}$ con $\sum_k p_k = 1$.
  La funzione (discreta) $k \mapsto p_k$ è detta *Probability Mass Function.*
  La CDF $F(t)$ di una v.r. τ discreta è una step function $F(t) = \sum_{k:\, t_k \le t} p_k$.
- *V.r. continue*: *PDF*
  la v.r. τ è assolutamente continua se $\exists\, f(x) \ge 0\ F(t) = \Pr\{\, \tau \le t \,\} = \int_{-\infty}^{t} f(x)\, \mathrm{d}x$;
  $f(t)$ è chiamata *probability density (function)* della v.r. τ , e si ha $f(t) \ge 0$, $\int_{-\infty}^{\infty} f(t)dt = 1$.
  Le funzioni $F(t)$ e $f(t)$ sono legate dalla relazione fondamentale $f(t) = \mathrm{d}F(t)/\mathrm{d}t$.
- *V.r. continue*: *PDF*
  Alcune funzioni di distribuzione sono miste, nel senso che esibiscono salti e crescite continue. Per tali funzioni si richiede una *definizione a tratti*.

Failure rate: λ

Si tratta di una funzione specificatamente legata a v.r. τ che rappresentano il verificarsi di un evento di failure (a seguito del quale l'item passa da uno stato operativo ad uno non operativo). Supposto un campione di items idealmente infinito al tempo $t = 0$, $\lambda(t)$ descrive "il numero di eventi di failure che avvengono (al tempo t) nell'unità di tempo rapportati al numero di items operativi in quel momento). In formule (definito $N_{\text{surv}}(t) = \mathrm{N}_0 - N_{\text{fail}}(t)$, con $\mathrm{N}_0 \to \infty$):

$$\lambda(t) \stackrel{\text{def}}{=} \frac{1}{N_{\text{surv}}(t)} \cdot \frac{\mathrm{d}N_{\text{fail}}}{\mathrm{d}t}(t) = \frac{1}{1 - F(t)} \cdot f(t) = -\frac{1}{R(t)} \cdot \frac{\mathrm{d}R}{\mathrm{d}t}(t)$$

Parametri numerici di variabili randomiche

Per caratterizzare una variabile randomica è necessario definire concetti come media, varianza, moda, quantile e mediana.

- **Expected Value / Mean**

  Definizione:

V.r. discrete	*V.r. continue*
Per una v.r. discreta τ che assume valori $t_1, t_2, \ldots$ con probabilità $p_1, p_2, \ldots$, si definisce *media* o *valore atteso* la quantità $E[\tau] \stackrel{\text{def}}{=} \sum_k t_k \, p_k$	La media di una v.r. continua τ con densità $f(t)$ è data da $E[\tau] = \int_{-\infty}^{\infty} t \cdot f(t) \, \mathrm{d}t$

  ammesso che la serie / l'integrale converga assolutamente.

  Alcune proprietà:

  - La media di una costante C è la costante stessa, ossia $E[\mathrm{C}] = \mathrm{C}$.
  - La media di una variabile randomica $\eta = u(\tau)$ è

    $$\tau \text{ discr: } E[\eta] = \sum_k u(t_k) \, p_k \quad , \quad \tau \text{ cont: } E[\eta] = \int_{-\infty}^{\infty} u(t) \, f(t) \, \mathrm{d}t$$

    ammesso che $u(t)$ sia continua e la serie e l'integrale convergano assolutamente.

    Due casi particolari:

    1. $u(x) = \mathrm{C}\, x \quad \Rightarrow \quad E[\mathrm{C}\tau] = \int_{-\infty}^{\infty} \mathrm{C}\, t \, f(t) dt = \mathrm{C}\, E[\tau]$
    2. $u(x) = x^{\mathrm{k}}$, che porta alla definizione del *momento* k-*esimo* di τ $\quad E\left[\tau^{\mathrm{k}}\right] = \int_{-\infty}^{\infty} t^k \, f(t) \, \mathrm{d}t$

- **Vita media:**

  Si tratta di un parametro tipicamente impiegato proprio in ambito di Affidabilità, e viene calcolato (per una variabile randomica continua) come

  $$T = \int_{-\infty}^{\infty} [\, 1 - F(t) \,] \, \mathrm{d}t \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} R(t) \, \mathrm{d}t$$

  Quando $\tau > 0$ rappresenta il *failure-free time*, T viene detto *Mean Time To Failure* perché, appunto, rappresenta il tempo operativo medio fra due failure consecutive (supposta nessuna operazione di mantenimento -preventivo- fra un guasto e l'altro).

- **Varianza**
  La varianza di una variabile randomica τ è una misura della dispersione della variabile randomica attorno alla sua media $E[\tau]$. La varianza è data da

$$Var[\tau] \stackrel{\text{def}}{=} E[\,(\tau - E[\tau])^2\,]$$

  Si riportano di seguito alcune proprietà importanti.

  - Per una variabile discreta è calcolata come

$$Var[\tau] = \sum_k (\,t_k - E[\tau]\,)^2\, p_k$$

  - Per una variabile continua è calcolata come

$$Var[\tau] = \int_{-\infty}^{\infty} (\,t - E[\tau]\,)^2\, f(t)\,\mathrm{d}t$$

  - In entrambi i casi vale

$$Var[\tau] = E[\tau^2] - (\,E[\tau]\,)^2$$

  - Se $E[\tau]$ o $Var[\tau]$ sono infinite, τ è detta a varianza infinita.

  - Per costanti C e A,

$$Var[\,\mathrm{C}\tau - \mathrm{A}\,] = \mathrm{C}^2\, Var[\tau] \quad \text{e} \quad Var[\mathrm{C}] = 0.$$

  Deviazione standard:
  La quantità $\sigma = \sqrt{Var[\tau]}$ è la *deviazione standard* di τ mentre, per $\tau \geq 0$, $k = \sigma / E[\tau]$ è il *coefficiente* di *variazione* di τ. La variabile randomica $(\,\tau - E[\tau]\,)/\,\sigma$, con media nulla e varianza 1, è definita *variabile randomica standardizzata*.
  La generalizzazione con legge esponenziale porta alla definizione del *k-esimo momento* centrale di τ

$$E[\,(\,\tau - E[\tau]\,)^k\,] = \int_{-\infty}^{\infty} (\,t - E[\tau]\,)^k\, f(t)\,\mathrm{d}t$$

- **Moda**
  Per una variabile randomica continua τ, la *moda* è il valore di t per cui $f(t)$ raggiunge il massimo. La distribuzione di τ è *multimodale* se $f(t)$ ha più di un massimo.

- **Quantile**
  Il *quantile* q è il valore t_q per cui $F(t)$ raggiunge il valore q ; in formule:

$$t_q = \inf\{\, t\colon\, F(t) \geq q \,\}$$

  e si ha $F(t_q) = q$ per una variabile continua.
  Il valore t_p per cui $1 - F(t_p) = Q(t_p) = p$ è detto *punto percentuale* .

- **Mediana**
  Il quantile 0.5 $(t_{0.5})$ è detto *mediana* .

Funzioni di distribuzione notevoli

In questa sezione vengono introdotte le più importanti funzioni di distribuzione usate nell'analisi di Affidabilità. La variabile t è qui usata per item non riparabili, ossia caratterizzati da un failure-free time $\tau > 0$. Per item riparabili, la variabile x è invece utilizzata.

- **Distribuzione esponenziale**

  Una variabile randomica positiva continua $\tau > 0$ ha una distribuzione esponenziale se la sua funzione di distribuzione è

  $$F(t) = 1 - e^{-\lambda t}, \qquad t > 0\,, \qquad \lambda > 0, \qquad (\ F(t) = 0 \ \text{ for } \ t \leq 0\ )$$

  La densità è data da

  $$f(t) = \lambda e^{-\lambda t}, \qquad t > 0\,, \qquad \lambda > 0, \qquad (\ f(t) = 0 \ \text{ for } \ t \leq 0\ )$$

  Il failure rate è definito da

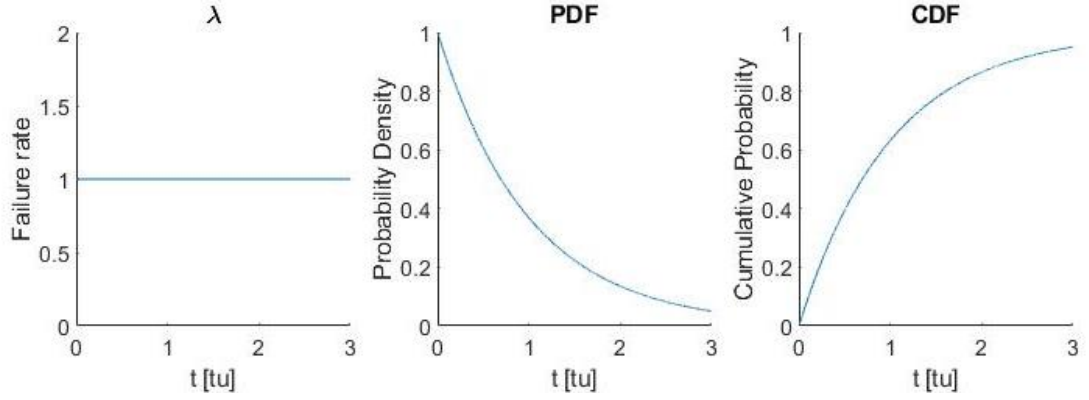
  $$\lambda(t) = \lambda\,, \qquad t > 0\,, \qquad (\ \lambda(t) = 0 \ \text{ for } \ t \leq 0\ )$$

  La media[21] , la varianza e il coefficiente di variazione sono rispettivamente

  $$E[\tau] = \frac{1}{\lambda} \quad , \quad Var[\tau] = \frac{1}{\lambda^2} \quad , \quad k = \sqrt{Var[\tau]}\ /\ E[\tau] = 1$$

  Per una distribuzione esponenziale, il failure rate è costante e uguale a λ. Questa proprietà è una importante caratteristica della distribuzione esponenziale che non compare in nessun'altra distribuzione continua. Si sottolineano inoltre:

  - *Memoryless property:*
    il comportamento dell'item nel futuro non dipende da quanto a lungo il componente sta operando, ossia $\Pr\{\ \tau > t + x_0\ \} = e^{-\lambda t}$
    In particolare, la probabilità che si rompa nel prossimo intervallo δt è costante e pari a $\lambda \delta t$.
  - *Failure rate costante a livello sistema:*
    se un sistema senza ridondanze ha elementi $E_1, E_2, \dots, E_n$ in serie e i failure-free times $\tau_1, \tau_2, \dots, \tau_n$ sono indipendenti ed esponenzialmente distribuiti con parametri $\lambda_1, \lambda_2, \dots, \lambda_n$, allora il failure rate di sistema λ è costante (indipendente dal tempo) e uguale alla somma dei failure rates degli elementi
    $$\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_n$$
    Portando a $R(t) = e^{-\lambda_1 t} \dots e^{-\lambda_n t} = e^{-\lambda t}$
    Il failure rate di sistema λ , pari a $\lambda = \sum \lambda_i$, è una caratteristica dei *modelli in serie* con elementi indipendenti e rimane valida per i failure rate tempo-dipendenti $\lambda_i = \lambda_i(t)$.



$(\lambda = 1)$

* [tu] = time unit

[21] Quando τ rappresenta un failure-free time, la media di una distribuzione esponenziale (genericamente detta MTTF) è specificatamente chiamata MTBF (Mean Time Between Failures).

- **Distribuzione Weibull**

Può essere considerata una generalizzazione della distribuzione esponenziale.
Una variabile randomica positiva continua $\tau > 0$ ha una distribuzione Weibull se la sua funzione di distribuzione è

$$F(t) = 1 - e^{-(\lambda t)^{\beta}} \quad , \quad t > 0\,, \quad \lambda, \beta > 0\,, \quad (\ F(t) = 0 \text{ for } t \leq 0\ )$$

La densità è data da

$$f(t) = \lambda\beta(\lambda t)^{\beta-1}\, e^{-(\lambda t)^{\beta}}, \quad t > 0\,, \quad \lambda, \beta > 0\,, \quad (\ f(t) = 0 \text{ for } t \leq 0\ )$$

Il failure rate è definito da

$$\lambda(t) = \beta\lambda(\lambda t)^{\beta-1}, \quad t > 0, \quad \lambda, \beta > 0, \quad (\lambda(t) = 0\ for\ t \leq 0)$$

con λ e β detti rispettivamente *parametro di scala* e *parametro di forma* .
Alcuni valori notevoli:

- Con $\beta = 1$, si ottiene la distribuzione esponenziale.
- Con $\beta > 1$, il failure rate $\lambda(t)$ è strettamente crescente, con $\lambda(+0) = 0$ e $\lambda(\infty) = \infty$
- Con $\beta < 1$, $\lambda(t)$ è strettamente decrescente, con $\lambda(+0) = \infty$ e $\lambda(\infty) = 0$

La media e la varianza sono

$$E[\tau] = \frac{\Gamma(1/\beta)}{\lambda\beta} \quad , \quad Var[\tau] = \frac{\Gamma(1 + 2/\beta) - \Gamma^2(1 + 1/\beta)}{\lambda^2}$$

con

$$\Gamma(z) = \int_0^{\infty} x^{z-1} e^{-x}\, \mathrm{d}x \quad , \quad z > 0$$

funzione Gamma completa.

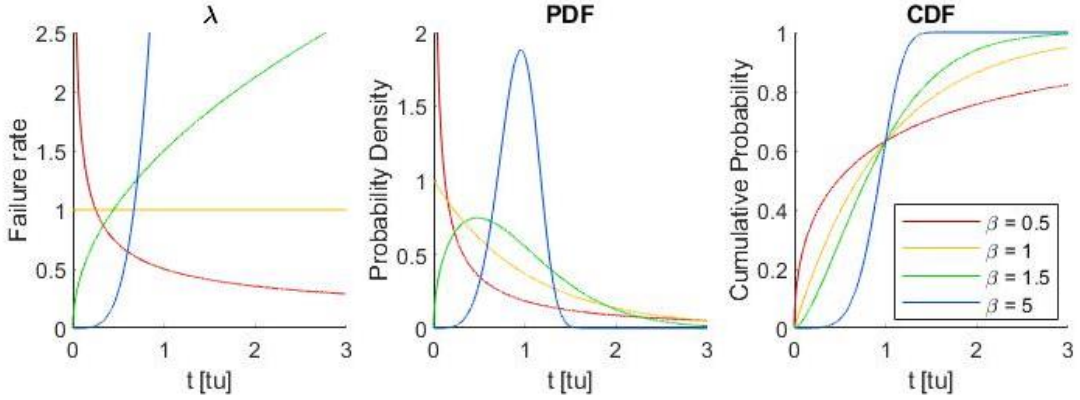
Applicazione alla Reliability:
Per un sistema senza ridondanza (modello serie) i cui elementi hanno failure-free times $\tau_1, \tau_2, \ldots, \tau_n$ indipendenti e distribuiti secondo una Weibull distribution, la funzione di affidabilità è data da

$$R(t) = \left(e^{-(\lambda t)^{\beta}}\right)^{n} = e^{-(\lambda' t)^{\beta}}, \quad t > 0, \quad R(0) = 1$$

con

$$\lambda' = \begin{cases} \lambda\, \sqrt[\beta]{n}, & \lambda_1 = \cdots = \lambda_n = \lambda \\ \sqrt[\beta]{\lambda_1^{\beta} + \cdots + \lambda_n^{\beta}}, & \lambda_1 \neq \cdots \neq \lambda_n \end{cases}$$

La distribuzione Weibull con $\beta > 1$ caratterizza componenti soggetti a wear-out o/e fatica.



$(\lambda = 1)$

- **Distribuzione normale**

Una variabile randomica continua $-\infty < \tau < \infty$ ha una distribuzione normale (Gaussiana) se la sua funzione di distribuzione è

$$F(t) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{(y-\mathrm{m})^2}{2\,\sigma^2}}\, \mathrm{d}y = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{(t-\mathrm{m})/\sigma} e^{-\frac{x^2}{2}}\, \mathrm{d}x, \quad -\infty < t, \mathrm{m} < \infty, \quad \sigma > 0$$

La densità è data da

$$f(t) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\frac{(t-\mathrm{m})^2}{2\,\sigma^2}}, \qquad -\infty < t, \mathrm{m} < \infty, \qquad \sigma > 0$$

Il failure rate è definito da

$$\lambda(t) = \frac{f(t)}{1 - F(t)}$$

La media , la varianza e il coefficiente di variazione sono rispettivamente

$$E[\tau] = \mathrm{m} \quad , \quad Var[\tau] = \sigma^2 \quad , \quad k = \sqrt{Var[\tau]}\,/\,E[\tau] = \sigma/\mathrm{m}$$

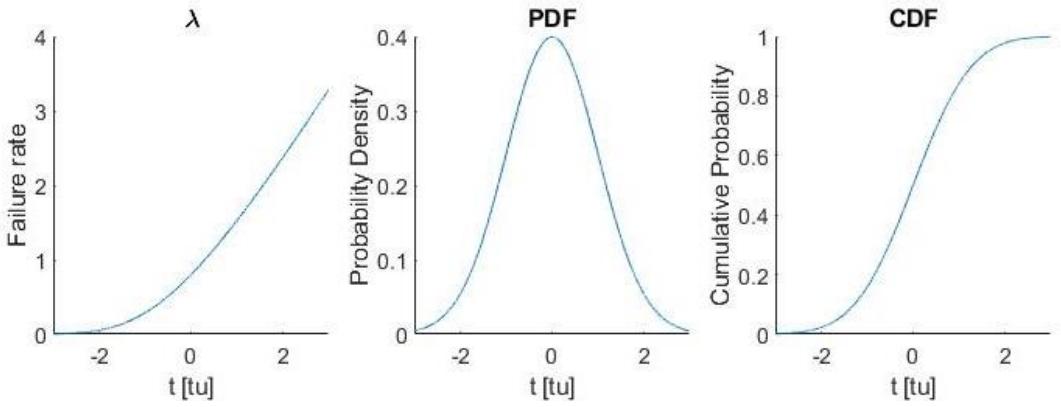
La densità è simmetrica rispetto alla linea $x = \mathrm{m}$ e la sua ampiezza dipende dalla varianza.
L'area sotto la curva di densità è:

- $\approx$0.6827 per l'intervallo $\mathrm{m} \pm \sigma$;
- $\approx$0.95450 per l'intervallo $\mathrm{m} \pm 2\sigma$;
- $\approx$0. 99730 per l'intervallo $\mathrm{m} \pm 3\sigma$;
- $\approx$0.9999367 per l'intervallo $\mathrm{m} \pm 4\sigma$;
- $\approx$0.9999932 per l'intervallo $\mathrm{m} \pm 4.5\sigma$;
- $\approx$0.99999943 per l'intervallo $\mathrm{m} \pm 5\sigma$.

Una distribuzione normale assume valori in $(-\infty, +\infty)$. Se ha parametri m e σ^2, $(\tau - \mathrm{m})/\sigma$ è normalmente distribuito con parametri 0 e 1, e la distribuzione si dice *standardizzata*.

Date due variabili τ_1 e τ_2 indipendenti e normalmente distribuite con parametri, rispettivamente, $(\mathrm{m}_1, \sigma_1^2)$ e $(\mathrm{m}_1, \sigma_1^2)$, allora $\eta = \tau_1 + \tau_2$ è normalmente distribuita con parametri $(\mathrm{m}_1 + \mathrm{m}_2, \sigma_1^2 + \sigma_2^2)$.
Tale legge si generalizza in: prese $\{\, \tau_i\colon\ (\mathrm{m}_i, \sigma_i^2)\,\}$, allora $\eta = \Sigma_i \tau_i$ con parametri $(\Sigma_i \mathrm{m}_i\ ,\ \Sigma_i \sigma_i^2)$.

Nella pratica è una delle più utilizzate, dato che la somma di un gran numero di variabili indipendenti converge, sotto condizioni deboli, ad una distribuzione normale (Central Limit Theorem).



$(\lambda = 1)$

- **Distribuzione log-normale**
  Una variabile randomica positiva continua $\tau > 0$ ha una distribuzione log-normale se il suo logaritmo è normalmente distribuito.

  La <u>funzione di distribuzione</u> risulta

$$F(t) = \frac{1}{\sigma\sqrt{2\pi}} \int_0^t \frac{1}{y} e^{-\frac{(\ln(\lambda y))^2}{2\,\sigma^2}} \mathrm{d}y = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{\ln(\lambda t)}{\sigma}} e^{-\frac{x^2}{2}}\, \mathrm{d}x \,,$$

$$t > 0, \qquad \lambda, \sigma > 0 \,, \qquad (\ F(t) = 0 \ \text{ for } \ t \leq 0\ )$$

  La <u>densità</u> è data da

$$f(t) = \frac{1}{t\,\sigma\sqrt{2\pi}}\, e^{-\frac{(\ln(\lambda t)^2}{2\,\sigma^2}}, \qquad t > 0, \qquad \lambda, \sigma > 0 \,, \qquad (\ f(t) = 0 \ \text{ for } \ t \leq 0\ )$$

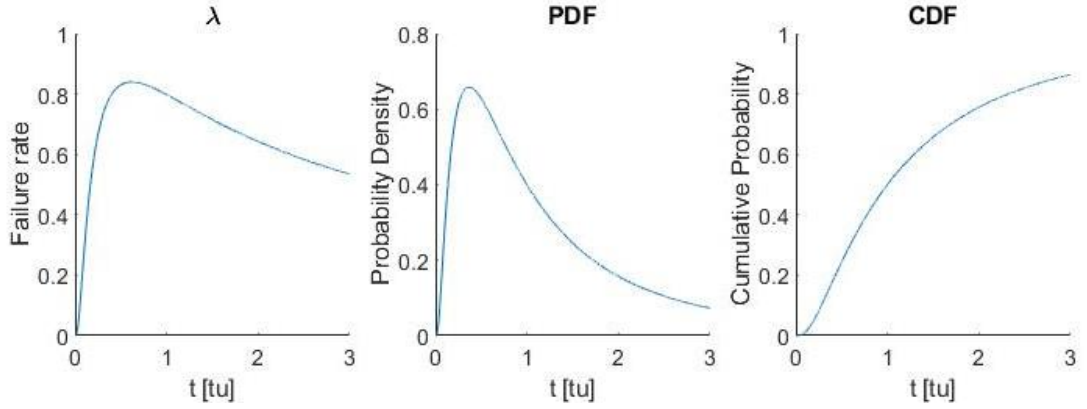
  Il <u>failure rate</u> è definito da

$$\lambda(t) = \frac{f(t)}{1 - F(t)}$$

  La <u>media</u> , la <u>varianza</u> e il <u>coefficiente di variazione</u> sono rispettivamente

$$E[\tau] = \frac{1}{\lambda} e^{\sigma^2/2} \quad , \quad Var[\tau] = \frac{1}{\lambda^2}\left(e^{2\sigma^2} - e^{\sigma^2} \right) \quad , \quad k = \sqrt{Var[\tau]}\,/\,E[\tau] = \sqrt{e^{\sigma^2} - 1}$$

  La densità è praticamente zero per alcuni t nell'origine, cresce rapidamente a un massimo e poi decresce velocemente. È utilizzata per rappresentare i repair times o negli accelerated reliability tests e compare laddove un gran numero di variabili indipendenti è combinato in modo moltiplicativo.



$(\lambda = 1)$

- **Altre distribuzioni:**
  Oltre quelle appena trattate, esistono anche altre distribuzioni notevoli ai fini delle RAMS, ma poiché non sono state impiegate direttamente in questa tesi non le si andrà a trattare.
  Per completezza, se ne elencano alcune: distribuzione… Gamma, Erlangiana, χ^2, Uniforme, Binomiale, di Poisson, Geometrica, Ipergeometrica, di Pearson, di Student.

2.2.3 Maintenance & Maintainability

La *Maintainability* (manutenibilità) è il secondo elemento principale della Dependability. Si tratta di una caratteristica inerente all'item, e descrive la semplicità, l'accuratezza, la sicurezza e l'economicità delle attività manutentive ad esso relative.
La *Maintenance* è l'insieme delle attività necessarie per riportare o mantenere un item in condizioni operative efficaci e sicure.

In quanto misura, la maintainability necessita di identificare delle grandezze con cui possa essere quantificata. Solitamente vengono impiegate definizioni probabilistiche legate a un qualche aspetto peculiare della manutenzione. Ad esempio, si può quantificare la maintainability in termini di…

- *Tempo:* come probabilità che il tempo manutentivo[22] non ecceda una certa soglia (*) ;
- *Frequenza:* come probabilità che la manutenzione non venga richiesta più di un certo numero di volte in un dato periodo temporale (*) ;
- *Costo:* come probabilità che il costo non ecceda un certo valore in un determinato periodo temporale (*) .

  (*) supposto che la manutenzione venga portata avanti seguendo le procedure prescritte, da personale adeguatamente competente, e fornite le risorse esterne necessarie .

Nella definizione è stato fatto riferimento al "tempo manutentivo": si tratta di una grandezza che ha varie interpretazioni a seconda dell'accezione, per cui è bene andare ad approfondire la questione, in quanto una delle metriche principali per caratterizzare la manutenzione.

Quantificazione mediante misure temporali

Definiamo di seguito alcune tipologie di "tempo" associate alla manutenzione.

- **Tempi relativi alla manutenzione in sé:**
  Possiamo suddividere concettualmente i tempi in "elementari" e "accorpati", dove i secondi consistono in combinazioni dei primi e sono utili a raggrupparli in set logicamente omogenei.

  Tempi elementari:

  - *MTTR (Mean Time To Repair) / MCT (Mean Corrective-maintenance Time):*
    tempo medio per compiere una manutenzione correttiva[23] ;
  - *MPT (Mean Preventive-maintenance Time):*
    tempo medio per compiere la manutenzione preventiva [24] ;
  - *LDT (mean Logistic Delay Time):*
    tempo medio per compiere i processi logistici, come acquisto e/o spedizione dell'item di ricambio al sito manutentivo interessato ;
  - *ADT (mean Administrative Delay Time):*
    tempo medio per compiere i processi amministrativi, come approvazione di una certa scelta manutentiva, recording dei dati manutentivi ecc.

[22] Più avanti si vedranno delle quantificazioni di tale tempo.
[23] È la manutenzione compiuta per *ripristinare* lo stato di un item ad una condizione di operatività efficace e sicura; avviene a seguito di una failure. Nel sotto-capitolo sulla manutenzione si approfondisce la questione.
[24] È la manutenzione compiuta per *mantenere* lo stato di un item ad una condizione di operatività efficace e sicura. Nel sotto-capitolo sulla manutenzione si approfondisce la questione.

Tempi accorpati:

- MAT (Mean Active-maintenance Time) / M:
  tempo medio legato alla manutenzione attiva, ossia l'insieme delle operazioni di manutenzione correttiva e preventiva.
- MDT (Mean Down-Time) / MTT (Mean Total Time):
  è il tempo totale (medio) per cui un item non è in condizioni di svolgere la propria funzione, ed è dato dall'unione dei tempi di manutenzione attiva, logistica e amministrativa.

Le tipologie di tempo manutentivo viste sono dunque i valori medi associati alle fasi di una manutenzione. Nello schema di Figura 5 si mostrano più in generale le fasi in cui si divide l'Operazione di un sistema (e.g. un velivolo) con focus sulla manutenzione, di cui i tempi appena visti rappresentano il ramo di "down-time".

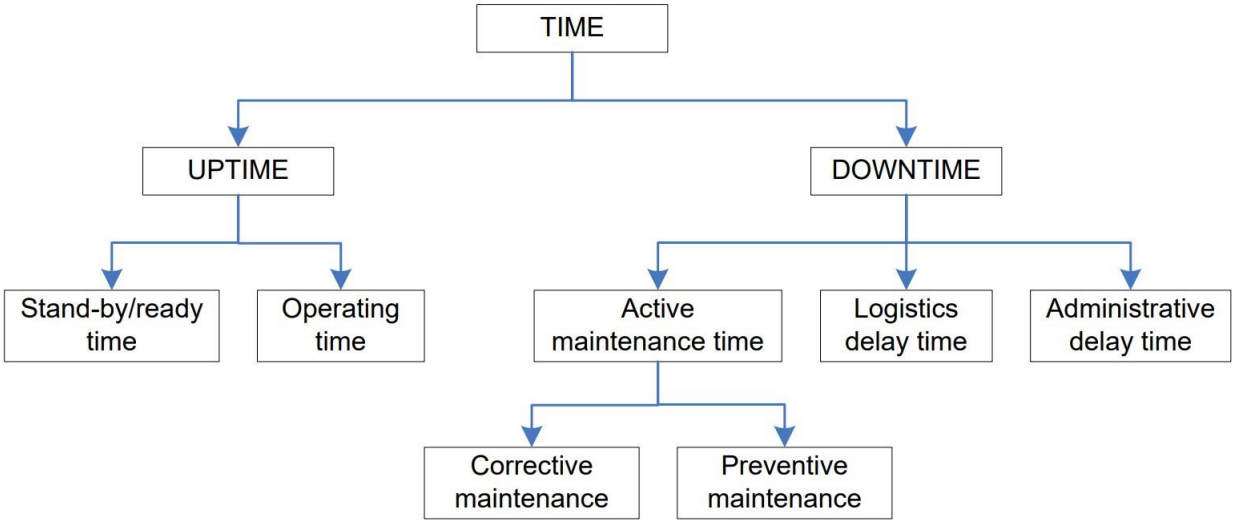


Figura 5) Suddivisione logica dei tempi legati all'Operazione di un sistema con focus sulla manutenzione.

- **Tempi relativi al periodo che intercorre fra le manutenzioni:**
  Per quantificare l'impatto della manutenzione sull'operatività del sistema, è utile considerare la relazione fra uptime e downtime (e sotto-categorie). Essa può essere espressa introducendo alcune nuove grandezze, fra cui tradizionalmente troviamo le seguenti:
  - *MTBM (Mean Time Between Maintenance):*
    tempo medio fra manutenzioni, include sia quella correttiva che preventiva, ed è strettamente legata alle proprietà di affidabilità del sistema (in particolare, a MTTF) .
  - *MTBR (Mean Time Between Replacement):*
    tempo medio fra sostituzioni dovute ad azioni manutentive, tipicamente genera requisiti sulle parti di ricambio.

- **Tempi manutentivi in relazione a quelli operativi:**
  È utile anche inquadrare i tempi manutentivi rispetto a quelli operazionali. Sotto questo punto di vista, una delle più tradizionali quantità è il
  - *MMH/FH (Maintenance Man-Hours per Flight-Hour):*
    quantità di ore-uomo (medie) necessarie per ogni ora-di-volo. Questa metrica è particolarmente utile sia per valutare l'active maintenance sia a livello di tempo che a livello di costi.

In ultimo si fa notare che tutte queste quantità medie sono relazionate alle proprietà di affidabilità del sistema, per cui è possibile la loro predizione (i.e. calcolo matematico) partendo dai risultati della Reliability ed unirli con i dati dei tempi manutentivi. La specifica metodologia non è particolarmente utile ai fini di questa tesi, per cui non le si dedica ulteriore spazio.

2.2.4 Availability

L'*availability* (disponibilità) è l'ultimo dei tre elementi della Dependability. Si tratta di un termine ad ampio spettro che esprime il rapporto dell'operatività effettiva rispetto a quella attesa di un item.

Definizione

Qualitativamente, viene spesso definita come

> " l'abilità di un item di svolgere la propria funzione sotto determinate condizioni in un certo istante (o per un certo intervallo) di tempo. "

Quantitativamente, essa può essere definita in differenti maniere; la prima categorizzazione le vede suddivise in:

- *Point Availability: PA*
  indica la *probabilità* che un sistema sia operativo (disponibile) in un dato momento (o "punto" nel tempo, da cui il nome). È una misura istantanea che può variare nel tempo a seconda delle condizioni operative del sistema.
- *Average Availability: AA*
  indica la *frazione* del tempo totale in cui un sistema è operativo (disponibile) durante un periodo di tempo specificato. È una misura aggregata che tiene conto sia del tempo di funzionamento che del tempo di inattività del sistema durante il periodo considerato.

Relazione con Reliability e Maintainability

La disponibilità non è una grandezza slegata da affidabilità e manutenibilità[25], ma le tre sono legate fra loro. Tendenzialmente, quando si tratta di predizione (i.e. calcolo matematico), è la disponibilità a venire calcolata a partire dalle altre due, e dunque possiamo concettualmente considerarla una grandezza derivata da esse. Tale calcolo è però spesso difficile, poiché bisogna considerare tutti i complessi fattori che influenzano l'affidabilità e la manutenibilità: caratteristiche dei componenti e dell'architettura di sistema, manutenzione, fattori umani e supporto logistico.

Tipologie di Point Availability

L'availability può essere espressa in maniere differenti a seconda della natura del sistema e/o del profilo di missione. Concentrandosi sulla point availability, si possono citare tre tipologie frequentemente impiegate; le si espone di seguito.

- *Inherent Availability* (A_I)*:* PA che considera come ideali il supporto logistico e amministrativo (LDT=ADT=0 → MDT=M), ed esclude i contributi di manutenzione preventiva (MPT=0 → M=MTTR & MTBM=MTBF), risultando in $A_I = \frac{MTTF}{MTTF+MTTR}$;
- *Achieved Availability* (A_A)*:* PA che considera come ideali solo supporto logistico e amministrativo ma include i contributi di manutenzione preventiva, risultando in $A_I = \frac{MTBM}{MTBM+M}$;
- *Operational Availability* (A_O)*:* PA che considera l'effettivo ambiente logistico e amministrativo ed anche la manutenzione preventiva, risultando nella forma più generale di availability espressa da $A_I = \frac{MTBM}{MTBM+M}$.

In questa tesi, l'availability principalmente presa in considerazione sarà la *Operational (point) availability* legata all'item *motore*, e per via delle complessità discusse più volte in precedenza, viene valutata non tramite predizione con calcoli (come quelli appena mostrati), ma tramite simulazione.

[25] e *supportabilità*, nel caso questa venga considerata a parte rispetto alla manutenibilità.

2.3 Basi di manutenzione moderna

Nello scorso sotto-capitolo sono stati esposti in termini generali le caratteristiche di affidabilità, manutenibilità e disponibilità di un sistema, e mostrato i parametri principali con cui possono essere quantificate. Queste nozioni sono essenziali per comprendere i parametri impiegati per valutare gli scenari simulati dall'OLSO.
Per chiudere il quadro delle nozioni importanti per la parte di simulazione[26], in questo sotto-capitolo si riprende la parte relativa alla manutenzione, approfondendola non tanto dal lato analitico (come visto prima) ma focalizzandosi sul lato pratico. L'esposizione tratterà i seguenti punti:

- i due paradigmi fondamentali della manutenzione (manutenzione correttiva e preventiva) ;
- la categorizzazione in funzione della complessità della manutenzione e luogo fisico in cui avviene (i Maintenance Levels) ;
- approfondimento dele attività elementari della manutenzione (i Maintenance Tasks) ;
- presentazione del processo base della manutenzione attiva (la Maintenance Action).

In conclusione, si tratta del moderno approccio al progetto di un sistema complesso con focus sui problemi manutentivi: la *Reliability Centered Maintenance*, su cui si fondano le attuali logiche manutentive.

2.3.1 Paradigmi manutentivi: corrective & preventive maintenance

Nella definizione di manutenzione[27] si fa esplicito riferimento ai suoi due momenti fondamentali: *riportare* o *mantenere* un item in condizioni operative efficaci e sicure. La prima catalogazione vede appunto la manutenzione suddivisa secondo i seguenti due paradigmi base.

- **Manutenzione correttiva:**
  detta anche *manutenzione non-schedulata*[28], include tutte le azioni (non-schedulate) svolte, a seguito di una failure di sistema, per *ripristinare* il sistema a una determinata condizione. La failure in questione può essere sia effettiva che sospetta.
  Le attività di manutenzione correttiva attiva vengono svolte in un ciclo manutentivo che consiste in alcuni step standard: identificazione e verifica della failure, localizzazione e isolamento della fault, disassemblamento per avere accesso all'item guasto, sua rimozione, sostituzione con un item di ricambio o riparazione in loco, riassemblamento, checkout, e infine verifica delle condizioni.
  Come quantificazione, la manutenzione correttiva è misurata in termini di frequenza ($MTBM_{\mathrm{u}}$ – Mean Time Beteween unsheduled Maintenance), tempo impiegato ($MTTR$ o M_{ct}), oppure ore-lavoro per ora-operativa (MLH/OH)[29].

- **Manutenzione preventiva:**
  detta anche *manutenzione schedulata*, include tutte le azioni (schedulate) di manutenzione svolte per *mantenere* il sistema in una determinata condizione.
  Le attività di manutenzione preventiva sono varie e non ricadono in uno specifico ordine; tra esse si trovano: ispezioni periodiche, monitoraggio delle condizioni, sostituzione degli item critici (prima della failure), calibrazioni periodiche ecc. Talvolta vengono incluse in questo gruppo anche alcune attività di routine dette di *servicing*: rifornimento di carburante e olio, ricaricamento batterie, lubrificazione, ecc. A volte comportano un downtime, altre no (e.g. svolte in stand-by).
  Misure tipiche sono in termini di frequenza ($MTBM_{\mathrm{s}}$), tempo (M_{pt}) e tempo di lavoro (MLH/OH).

[26] Questo sotto-capitolo chiude le nozioni per la simulazione, mentre i prossimi due saranno importanti per comprendere l'ottimizzazione e le anili ad essa associate.
[27] Manutenzione: insieme delle attività necessarie per *riportare* ("*restore*") o *mantenere ("retain")* un item in condizioni operative efficaci e sicure.
[28] Talvolta, anche *manutenzione di emergenza* ("*emergency maintenance*").
[29] Si tratta di grandezze già incontrate nella sezione sulla Maintainability.

La manutenzione preventiva ha come tratto caratteristico l'imposizione di scadenze[30] al termine delle quali delle azioni attive sono richieste, come revisioni o sostituzioni, in maniera da anticipare il presentarsi di una failure. A seconda di come queste soglie vengono stabilite, essa viene declinata in due sotto-paradigmi:

- Manutenzione preventiva tradizionale:
  per essa, le soglie vengono stabilite -a progetto- in maniera *statica*, ossia non cambiano nel corso dell'operazione dell'item. Con "preventive maintenance" di solito si indica questo tipo.
- Manutenzione predittiva:
  anche chiamata *on-condition maintenance*, per essa le soglie vengono stabilite in maniera *dinamica*, ossia non vengono definite a progetto ma imposte rispetto all'effettivo stato dell'item, e aggiornate nel tempo.

La manutenzione preventiva si basa sulla possibilità di imporre tali soglie su un item, e ciò è permesso solo se la sua fisica della failure le contempla, e nella capacità tecnologica (in termini di performance e costi) di poterle stimare. In particolare:

- Nella manutenzione preventiva tradizionale, si richiede che la failure presenti fenomeni di invecchiamento o usura, che si manifestano nel failure-rate come un incremento per valori crescenti della variabile indipendente (tempo, cicli, ecc.). Più l'incremento è repentino, migliore è la possibilità di stimare la soglia associata.
- Nella manutenzione predittiva, si richiede che la failure mostri sintomi rilevabili (deterioramento progressivo nel tempo, non failure improvvisa). È importante porre l'accento su entrambi i punti: i sintomi devono (ovviamente) esistere, ma altrettanto importante è il disporre di metodi di monitoraggio per rilevarli[31] e tecniche di analisi di dati per comprenderli. Inoltre, fondamentale è anche il disporre di tecniche di *predizione* dello stato dell'item nel futuro (non solo stima del suo stato presente), senza le quali è impossibile ricavare le soglie discusse.

2.3.2 Suddivisione per attività e luogo: *Maintenance Levels*

Una seconda suddivisione della manutenzione focalizza l'attenzione sul tipo di intervento (in termini di complessità) e luogo dove viene effettuato. Si hanno così i *Maintenance Levels (MLs)*; essi sono tre, e analizzabili in termini di:

- Dove) luogo di attività;
- Chi) quale personale e che livello di competenze sono richieste;
- Come) con che tipo di equipaggiamento;
- Cosa) che tipologie e complessità di tasks manutentivi.

Approfondiamoli di seguito.

- **ML1) Organizational Level:**
  Viene compiuto sugli elementi primi del sistema direttamente al sito operativo dell'utente finale. Generalmente include tasks svolti dall'organizzazione utente col proprio equipaggiamento, da personale solitamente legato all'operazione e all'uso di quell'equipaggiamento, e disponendo di minimo tempo disponibile per una manutenzione dettagliata di sistema.
  La manutenzione a questo livello è normalmente limitata a check periodici di performance d'equipaggiamento, ispezioni visuali, pulizia degli equipaggiamenti, qualche servicing, aggiustamenti esterni, e la rimozione e sostituzione di componenti.

[30] In termini di ore volate, numero di cicli ecc.
[31] Condizione associata al possedere metodi di indagine non distruttiva, e che siano economicamente vantaggiosi.

Il personale assegnato a questo livello di solito non ripara i componenti rimossi, ma li inoltra al livello intermedio (ML2).
Da un punto di vista manutentivo, a questo livello viene assegnato il personale con il più basso livello di competenze (e il design dei componenti di sistema deve tenere ciò in considerazione).

- **ML2) Intermediate Level:**
  I tasks di questo livello vengono compiuti da infrastrutture e organizzazioni specializzate di tipo mobile, semi-mobile e fisso. Gli items rimossi possono venire riparati attraverso la rimozione e sostituzione di moduli maggiori, assemblies, e/o parti. A questo livello può anche essere compiuta manutenzione schedulata richiedente il disassemblamento di equipaggiamenti.
  Il personale impegnato è solitamente più competente e meglio equipaggiato di quello a ML1 e si occupano di una manutenzione più dettagliata.

  Unità mobili o semi-mobili:
  sono spesso assegnate per fornire supporto ravvicinato ad elementi del sistema in stato operativo. Queste unità possono includere vans, trucks o portable shelters trasportanti equipaggiamenti di test e supporto e parti di ricambio, e una di esse può essere impiegata per supportare più di un sito operativo.
  Il loro obiettivo è fornire manutenzione sul campo (oltre quella già fornita a ML1) per facilitare il rapido restoring del sistema al suo stato operativo completo.

  Unità fisse:
  sono generalmente preposte al supporto sia di tasks di livello organizzativo che unità mobili o semi-mobili. Vengono interessati da task manutentivi che non possono essere compiuti dai livelli inferiori, per via di limitate competenze di personale o equipaggiamenti di test.
  Il personale altamente qualificato, gli equipaggiamenti di test e supporto, più ricambi e le migliori infrastrutture di questo livello spesso permettono la riparazione a livello di modulo e parte.
  Di solito sono posti in vicinanza dei siti operativi del sistema, ma all'interno di specifiche aree dedicate.
  Rapidi tempi manutentivi sono solitamente non così imperativi qui rispetto ai livelli inferiori.

- **ML3) Supplier / Manufacturer / Depot Level:**
  Esso costituisce il più alto livello di manutenzione, e supporta il compimento di tasks oltre le capacità del livello intermedio.
  Fisicamente, si può trattare di un'infrastruttura di riparazione specializzata che supporta un gran numero di sistemi, equipaggiamenti e software, oppure il sito stesso dell'azienda produttrice di un certo item. A questo livello i siti manutentivi sono sempre fissi, e la mobilità non è un requisito. In effetti, ivi vengono ospitati equipaggiamenti grossi e costosi, grandi quantità di ricambi, siti con capacità di environmental control e altre.
  In questo livello esistono e coesistono due piani organizzativi differenti: il primo gestisce un alto volume manutentivo mediante tecniche di linea d'assemblaggio, impiegando lavoratori relativamente poco qualificati per svolgere la grossa, e meno complessa, porzione del lavoro; si lascia così a un secondo piano organizzativo, formato da esperti altamente qualificati, il compito di concentrarsi in aree specifiche dove vengono svolti i lavori di alta precisione.
  Fra le attività più significative di questo livello si segnalano: revisioni e ricostruzioni complete, calibrazione di equipaggiamenti, ed altre varie azioni manutentive altamente complesse.
  Le infrastrutture sono generalmente locate in luoghi remoti e forniscono supporto per un alto numero di linee di produzione spaziando su una vasta area geografica, persino assumendo il ruolo di gestore dell'inventario per parti di ricambio e riparazione.

Questi tre livelli sono particolarmente importanti ai fini della simulazione manutentiva, poiché formano la base delle logiche manutentive su cui si fonda.

2.3.3 Le attività elementari: *Maintenance Tasks*

Più volte nel corso di questo documento è stato fatto riferimento ad attività manutentive come "riparazione, revisione, calibrazione, ecc", lasciando implicito il loro significato. Essi sono chiamati *maintenance tasks* e definiti come "attività specifiche di manutenzione sull'aeromobile o suoi elementi": rappresentano dunque le "attività semplici" alla base della manutenzione. Siccome si tratta di concetti fondamentali per comprendere come venga svolta la manutenzione, si dedica loro in questa sezione dello spazio per elencarne i principali, affiancandoli da una breve descrizione.
Li si riporta con i loro nomi inglesi, in ordine alfabetico.

- **Allignment/Adjustment:** consistono nel riportare il sistema ad un operatività accettabile attraverso l'adeguato allineamento (allignment) dei suoi componenti e/o la regolazione (adjustment) del controllo fino a he un determinato parametro di performance non viene portato in un range accettabile.
- **Calibration:** processo di verifica di un item rispetto ad uno standard. La calibrazione generalmente si applica all'equipaggiamento di misurazione di precisione (PME – Precision Measurement Equipment) e può essere portato a termine sia su base schedulata che su base non-schedulata conseguente ad una riparazione su un item PME.
- **Condition monitoring:** consiste nella supervisione continua dell'operatività di un sistema per assicurare la corretta operatività e il rilevamento di anormalità indicanti una failure avvenuta o in procinto di avveinre.
- **Functional test:** checkout operazionale di sistema, sia come verifica delle condizioni conseguente al completamento di una riparazione di item, oppure come requisito schedulato periodico.
- **Inspection:** azione, o serie di azioni, intraprese per assicurare che sussista un determinato requisito di qualità. Per compiere un'ispezione per una condizione desiderata, è possibile sia necessario la rimozione di un item per avere accesso attraverso la rimozione di altri items, o disassemblare parzialmente un item.
- **Overhaul:** azione, o serie di azioni, intraprese quando un item è completamente disassemblato, ricondizionato, rilavorato, etstato e riportato ad una condizione di servizio che soddisfi tutti i requisiti stabiliti nelle specifiche applicabili. La revisione può essere conseguente a requisiti sia schedulati che no, ed è generalmente compiuta a ML3 presso l'impianto di produzione/fornitore/deposito.
- **Removal:** consiste nell'estrazione di un item dall'entità immediatamente successiva nella scala gerarchica del sistema.
- **Removal & Reinstallation:** consiste nella rimozione di un item per manutenzione e reinstallamento dello stesso dopo sua riparazione.
- **Removal & Replacement:** consiste nella rimozione di un item per manutenzione e l'installazione (sostituzione) di un altro item ad esso affine, generalmente un item di ricambio dal magazzino. Una tale azione può essere conseguente ad una failure di componente o come risultato di un requisito di manutenzione preventiva (e.g. la periodica sostituzione di un item a vita limitata).
- **Repair:** costituisce una serie di attività correttive richieste per riportare un item ad una adeguata condizione di servizio. Può comprendere la rimozione e sostituzione di parti, l'alterazione dei materiali, fissaggio, sigillamento, rabboccatura/riempimento, ecc.
- **Servicing:** include attività manutentive associate con la pulizia di componenti e l'applicazione di lubrificanti, carburante, olio ecc. Può richiedere la rimozione disassemblamento, riassemblamento, regolazione e installazione, e può essere compiuto sia su base schedulata che no.
- **Troubleshooting:** coinvolge i processi logici che conducono all'identificazione della causa di un malfunzionamento di sistema, attraverso sua localizzazione e isolamento (include la diagnostica).

2.3.4 Processo di manutenzione correttiva attiva: *Maintenance Action*

Un'*azione manutentiva* è l'insieme di task manutentivi, in determinata o sparsa[32] successione, volto all'ottenimento di un qualche obiettivo manutentivo di alto livello. Nel caso di un'azione manutentiva correttiva, essa viene svolta con l'obiettivo di ripristinare la funzionalità efficace e sicura del sistema, e si compone tipicamente dei task indicati in Figura 6.

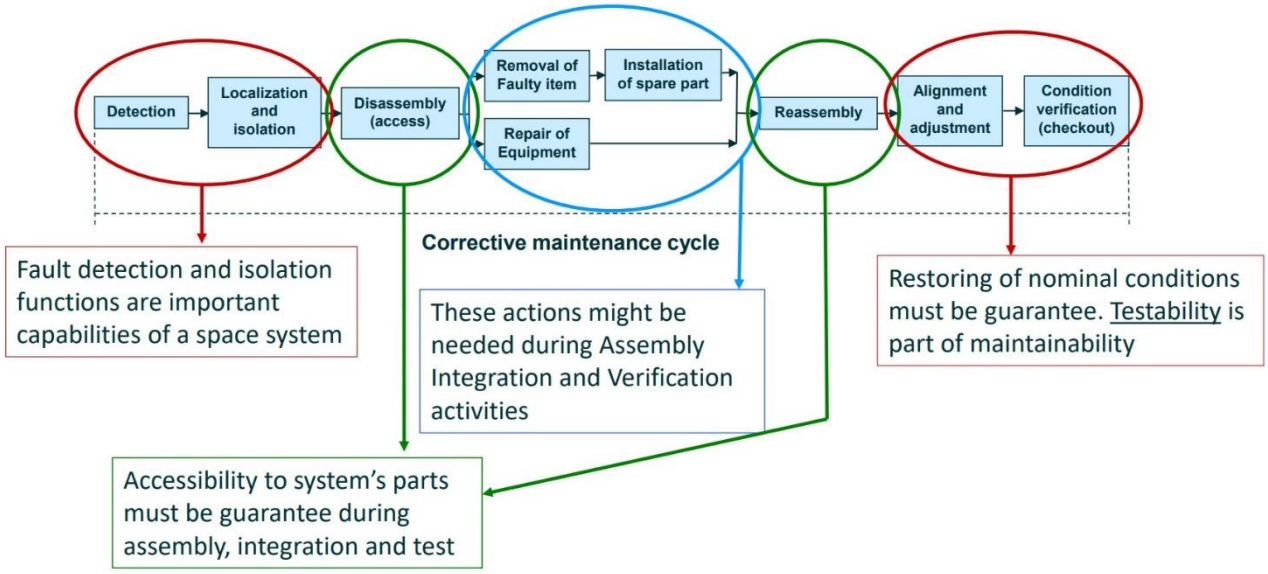


Figura 6) Schema della tipica azione manutentiva di tipo correttivo

Una manutenzione preventiva si compone tradizionalmente degli stessi task di quella correttiva, a meno dei primi due (Detection, e Localization & Isolation), visto che il componente interessato è già noto. È anche vero però che a seguito di una manutenzione a ML2, spesso vengono comunque compiute operazioni di controllo sugli equipaggiamenti principali , che possono impiegare ispezioni non segnalate nell'azione in figura.
L'azione/ciclo di manutenzione mostrata è importante per queste analisi poiché è uno dei punti fondamentali su cui si basa il Simulatore manutentivo.

2.3.5 Moderno approccio al progetto maintenance-based: *RCM*

Introduzione storica

Il primo degli approcci manutentivi, oltre quello di manutenzione correttiva, è stato quello della manutenzione preventiva -tradizionale- (anche detta *time-based maintenance*): esso è caratterizzato dalla stima del momento di failure basato sui dati storici di vita (in termini di tempo o cicli) di un certo item, per poter ricavare un programma di manutenzione preventiva stabilendo soglie allo scadere delle quali devono essere compiute operazioni di manutenzione (ispezione, revisione o sostituzione). Col tempo ci si è però resi conto che tale paradigma cade spesso in difetto delle sue due assunzioni principali, ossia le seguenti:

- che esista una correlazione forte fra failure rate ed età dell'item ;
- che sia sempre possibile determinare statisticamente una probabilità di failure, così da ricavarne le soglie di vita pre-failure.

Infatti, spesso gli item mostrano una vita significativamente più lunga di quella nominale su cui il metodo viene costruito, comportando ad esempio lo scarto di una parte la cui vita è in realtà lontana dal suo reale termine. D'altra parte, l'imposizione di soglie fortemente conservative è necessaria per gestire l'incertezza nella stima della soglia stessa, talvolta assai importante.

[32] Ad esempio, nel caso in cui i sotto-processi (maintenance tasks) possano essere compiuti in parallelo.

Nel frattempo, al fianco di questa metodologia tradizionale si sviluppava un nuovo paradigma: la manutenzione on-condition, che prometteva di ridurre sensibilmente i costi evitando manutenzioni non necessarie (per i motivi sopra discussi) attraverso la stima dello stato effettivo di un certo item, piuttosto che rifarsi ad un comportamento medio associato, tra l'altro, a dati e metodologie spesso datati. Questa nuovo paradigma porta con sé un cambio di prospettiva, e viene permesso dalle moderne tecnologie di acquisizione[33] e processamento digitale dei dati (che nel primo contesto industriale non erano ancora esistenti).

Tuttavia, la manutenzione on-condition non soppianta completamente quella preventiva tradizionale, che rimane valida ad esempio nel caso di items con marcato comportamento ad usura. Si tratta dunque di definire delle logiche decisionali per scegliere la miglior combinazione di task manutentivi per supportare un determinato sistema. Nell'affrontare queste e altre problematiche di progetto integrato dell'insieme "sistema più sua manutenzione", un nuovo paradigma è nato alla fine degli anni '70: la Reliability-Centered Maintenance (RCM).

Caratteristiche generali della RCM

La RCM integra varie strategie manutentive: la -Traditional- *Preventive Maintenance (PM)*, *Predictive Testing and Inspection (PT&I)*, la *Reactive Maintenance* (anche chiamata *Repair*) e la *Proactive Maintenance* al fine di aumentare la probabilità che la macchina o il componente funzionino come richiesto durante il corso del ciclo-vita progettato, con la minima necessità di manutenzione e downtime. Non si tratta di strategie applicate in modo indipendente, ma di una integrazione ottimale al fine di trarre vantaggio dai rispettivi punti di forza. La RCM richiede inoltre di considerare le conseguenze di una failure ad un dato componente e di rispondere a quesiti come:

- Cosa fa l'equipaggiamento, ossia, quali sono le sue funzioni?
- Quali sono le failure funzionali che hanno più probabilità di avvenire?
- Quali sono le loro conseguenze più probabili?
- Cosa può essere fatto per ridurre la probabilità di una failure, identificarne l'inizio o ridurre le sue conseguenze?

L'obiettivo è ridurre il Life-Cycle Cost (LCC) della macchina o dell'equipaggiamento permettendo lo svolgimento della sua funzione con l'affidabilità e disponibilità richieste. In altre parole, è la determinazione della più idonea tecnica di manutenzione, efficace dal punto di vista dei costi, che minimizzi il rischio di failure e crei un ambiente sicuro di lavoro. Ciò è ottenuto tramite identificazione dei failure mode e delle conseguenze per ciascun sistema.

In Figura 7 sono riportate alcune applicazioni di ciascuna strategia.

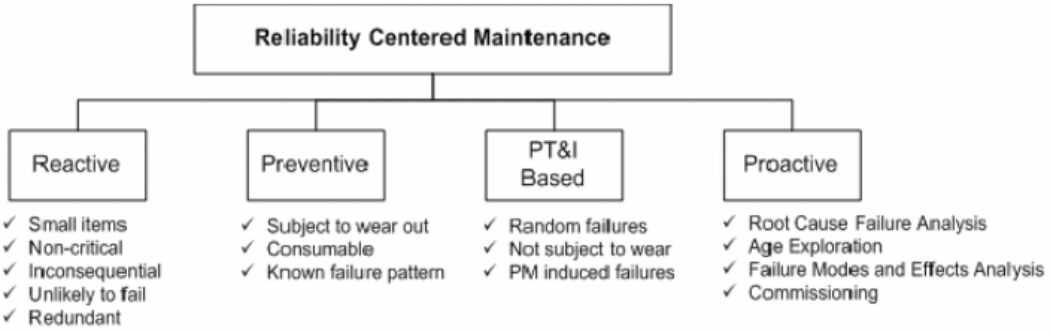


Figura 7) Componenti della RCM

[33] Dunque, problematiche legate alla *misurazione* (sicuramente non distruttiva, ma anche economicamente sostenibile) di parametri indicativi dello stato dell'item.

In termini di obiettivi, la RCM:

- assicura la realizzazione di livelli di sicurezza e affidabilità adeguati all'equipaggiamento;
- ristabilisce tali livelli quando si verifica un deterioramento;
- ottiene informazioni per migliorare la progettazione di quelli items che rivelano essere inadeguati;
- raggiungere questi obiettivi al minimo costo, includendo costi di manutenzione, supporto e costi conseguenti alla failures durante l'operatività.

Le tipologie di RCM

Il programma di RCM può essere implementato in modi differenti: la scelta dipende dell'utente finale e da aspetti quali conseguenze della failure, probabilità di failure, dati storici e Risk tolerance (Mission Criticality). Si distinguono due categorie di RCM: rigorosa e intuitiva.

- **Rigorosa:**
  anche nota come *Classical RCM*, fornisce la maggior parte delle conoscenze e dei dati riguardanti le funzioni del sistema, le modalità di guasto e gli interventi di manutenzione per risolvere guasti funzionali. Si tratta della tipologia che produce la documentazione più completa, ma ad un alto costo di analisi.
  Questo tipo di analisi è normalmente utilizzata su ciascun item che debba essere incluso in un nuovo e costoso sistema[34]; viceversa, è raramente impiegato per enti la cui costruzione e modi di guasto sono ben noti e compresi. Viene impiegata industrie come aeronautica, spazio, difesa e nucleare, in generale dove una failure funzionale può risultare in una significativa perdita di vite umane, con impatto sulla sicurezza nazionale o ambientale.
  È basata sulla FMEA (Failure Modes and Effects Analysis), e include il calcolo delle probabilità di failure del sistema e della sua affidabilità, con pochissima influenza da dati storici di performance. L'analisi determina gli appropriati tasks di manutenzione e i requisiti di riprogettazione al fine di identificare ciascun failure mode e le sue conseguenze.

- **Intuitiva:**
  anche nota come *Streamlined* o *Abbreviated RCM* ed è più appropriata per le infrastrutture dato il costo elevato di una analisi rigorosa, il basso impatto della failure sul sistema e la loro ridondanza. Identifica e implementa attività di manutenzione condition-based. Attività di poco valore sono eliminate sulla base di dati storici e sugli input del personale di manutenzione. Richiede la presenza di un esperto delle tecnologie PT&I.
  Poiché non è indicata per sistemi complessi, costosi e safety-critical come quelli aerospaziali, non viene impiegata negli ambiti che riguardano questa tesi (i.e. ambito aeronautico).

Qualsiasi approccio adottato va validato e riesaminato.

I principi della RCM

I principi della RCM sono i seguenti:

- **Function-Oriented:** tutela delle *funzionalità* del sistema/equipaggiamento, non semplicemente l'operabilità in sé ;
- **System-Focused:** tutela delle funzionalità del sistema piuttosto che del singolo componente ;
- **Reliability-Centered:** focus sulla variabilità temporale del failure rate in funzione dell'età degli items ;

[34] Come un velivolo o un sistema spaziale.

- **Riconoscimento dei limiti di progettazione:** riconoscimento del fatto che l'affidabilità intrinseca del sistema sia dovuta prima al design che alla manutenzione, ossia, la manutenzione può solo raggiungere e mantenere il livello di affidabilità delle apparecchiature previsto dalla progettazione. Tuttavia, la RCM riconosce anche che il feedback di manutenzione può migliorare il progetto originale e si adopera nel processo di Age Exploration (EA).
- **Safety, Security e Cost-effectiveness**
- **Failure come qualsiasi condizione insoddisfacente:** con "failure" va inteso non solo un guasto che porti alla perdita di una funzione, ma anche un evento che comporti una perdita di performance non accettabile.
- **Logic Tree per la definizione dei Maintenance Tasks:** sono il caposaldo attorno a cui si sviluppano, nella pratica, gli output della RCM. Si tratta di schemi decisionali che guidano l'analista verso la composizione di un piano manutentivo (preventivo) su misura al sistema considerato. Un esempio è riportato in Figura 8.

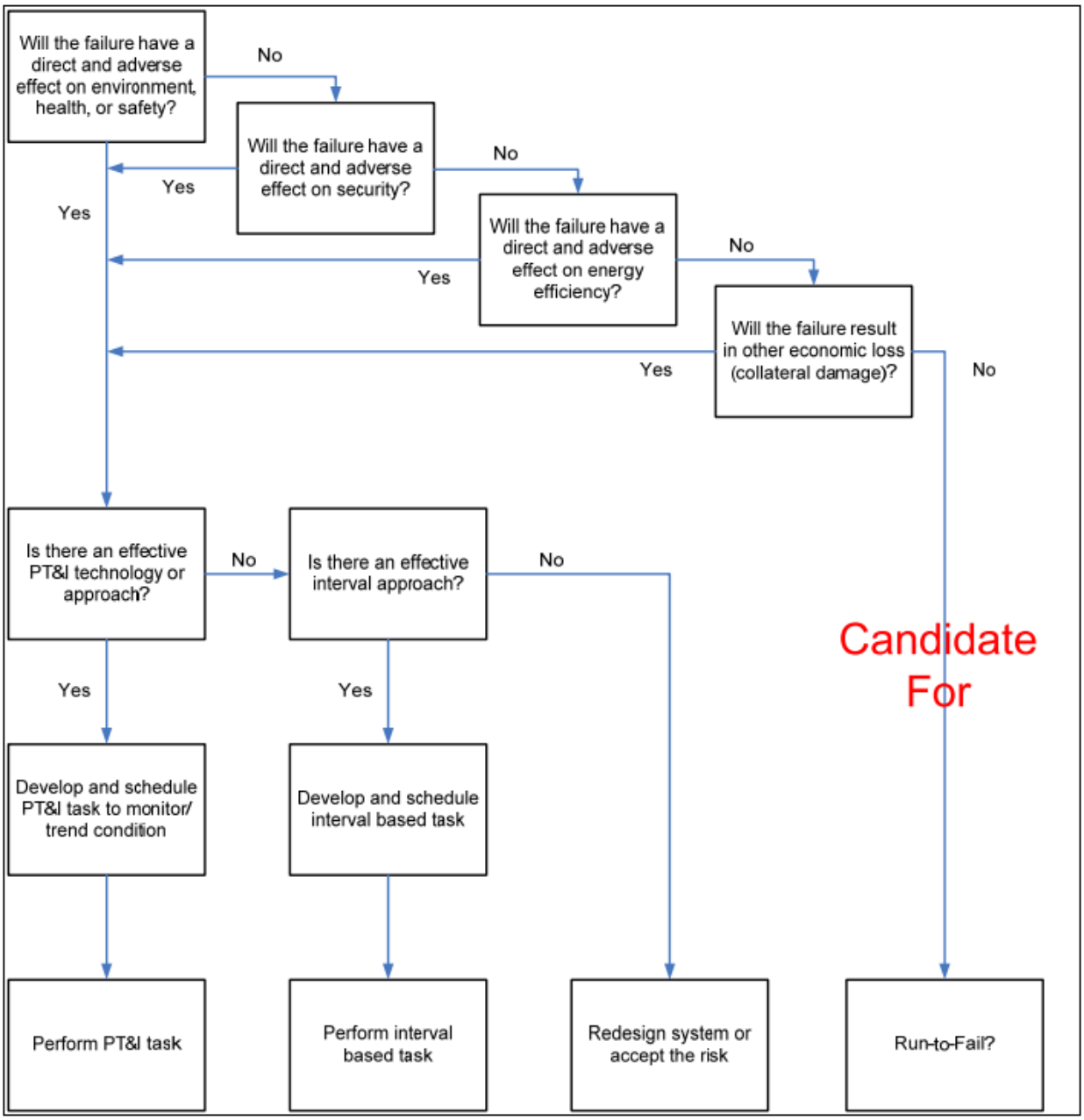


Figura 8) RCM Logic Tree

- **Applicabilità dei task:** i tasks devono riguardare i modi di guasto e considerarne le caratteristiche;
- **Efficacia dei task:** devono ridurre la probabilità di failure ma anche essere cost-effective.

Tre dei precedenti principi riguardano la definizione di tasks. È possibile definire differenti tipi di attività: time-directed PM, condition-directed PT&I e failure-finding come aspetto della Proactive Maintenance. La prima tipologia riguarda attività programmate quando opportuno. La seconda, attività eseguite quando si verificano le condizioni per cui sono necessarie. La terza riguarda attività che individuano funzioni nascoste che sono state oggetto di failure ma che non hanno prodotto evidenze e che normalmente sarebbero oggetto di time-directed task. Normalmente l'approccio Run-to-Failure è accettabile sono per alcuni equipaggiamenti e la scelta deve essere ben ponderata.

Le analisi della RCM

L'analisi di manutenzione si svolge attraverso i seguenti step:

1) Identificazione del sistema e delle condizioni al contorno: input, output, risorse, limiti ;
2) Identificazione dei sottosistemi e dei componenti ;
3) Esame delle funzioni: primarie vs di supporto, continue vs intermittenti, attive vs passive ;
4) Definizione di failure e failure mode: nascoste, potenziali ;
5) Identificazione delle conseguenze della failure: qualità delle operazione per gli aspetti ambientali, di salute o sicurezza, costi.

Al termine di una analisi di manutenzione, sono quattro le possibilità:

- Si definiscono azioni interval- (time o cycle) based ;
- Si definiscono azioni Condition-based directed (PT&I) ;
- Non si definisce alcuna azione e si scegli di rimandare la riparazione alla failure successiva ;
- Non si definisce alcuna azione di manutenzione ma si considera una riprogettazione o introduzione di ridondanza.

La RCM e questa tesi

Come appena visto, la RCM è un ottimo strumento per costruire un piano manutentivo adattandolo al meglio al sistema che deve supportare, e assumendo l'importante (e attivo) ruolo di feedback verso di esso per suggerirne modifiche al design. Tuttavia, il piano così creato risulta indipendente dall'effettivo stato del Supporto logistico che regge tali processi di manutenzione: la RCM compie, nelle sue analisi, l'ipotesi che esso riesca a fornire il supporto ai processi manutentivi.
Nella realtà però non sempre è così: risulta importante l'effettivo stato delle industrie, magazzini, sistemi di trasporto ecc. Per questa ragione, per compiere un ulteriore passo avanti nella gestione della manutenzione, è necessario integrarla assieme agli altri momenti del Supporto logistico. Questa tesi mira, con l'OLSO v.1, a porre le basi per poter superare questi limiti portando all'integrazione desiderata.

2.4 Basi di analisi statistica

In questo sotto-capitolo si vogliono esporre alcuni concetti principali dell'analisi statistica, che sono alla base di alcune questioni affrontate durante il lavoro di tesi per rispondere ai problemi legati all'intrinseca aleatorietà del sistema studiato.

Introduzione

La matematica statistica riguarda essenzialmente situazioni in cui, data una popolazione di elementi statisticamente identici e indipendenti con proprietà statistiche sconosciute, misure riguardo queste proprietà sono fatte su un campione randomico della popolazione e, sulla base dei dati raccolti, conclusioni vengono tratte sui restanti elementi della popolazione. La matematica statistica relaziona osservazioni (realizzazioni) di un dato evento (casuale) in una serie di tentativi indipendenti alla ricerca del modello probabilistico più adatto a descrivere l'evento considerato (approccio induttivo). I metodi usati si basano sulla teoria della probabilità e i risultati ottenuti possono essere formulati solo in termini probabilistici. Per semplificare la notazione, i termini *campione*, *media* e *indipendente* sono utilizzati per i concetti di "campione randomico prelevato da una popolazione larga e omogenea", "valore atteso" e "campione mutualmente, statisticamente e stocasticamente indipendente".

Metodi empirici

I metodi empirici permettono una valutazione rapida della funzione di distribuzione, della media, della varianza e degli altri momenti che caratterizzano una variabile randomica. Un vantaggio delle funzioni di distribuzione empiriche è nella possibilità di utilizzarne la rappresentazione su particolari grafici, i probability-plot papers, come check visivo sulla correttezza del modello assunto.
Questa metodologia non è stata impiegata esplicitamente nella tesi; la si riporta per completezza.

Stima parametrica

Si supponga che il tipo di funzione di distribuzione $F(t)$ di una variabile randomica τ sia conosciuto. Si ha inoltre $F(t) = F(t, \theta_1, \dots, \theta_r)$, noto nella sua forma funzionale. I parametri $\theta_1, \dots, \theta_r$ necessitano invece di essere stimati. Tale stima deve essere fatta sulla base delle osservazioni mutualmente (statisticamente) indipendenti $t_1, \dots, t_n$ della variabile randomica τ. Una distinzione è fatta tra stima puntuale e di intervallo.

- **Point estimation:**
  Si supponga innanzitutto che $F(t)$ dipenda solo dal parametro θ, assunto come costante sconosciuta. Una stima puntuale per θ è una funzione
  $$\hat{\theta}_n = u(t_1, \dots, t_n)$$
  delle osservazioni $t_1, \dots, t_n$ della variabile randomica τ. La stima viene detta
  - *Unbiased*, se $E[\hat{\theta}_n] = 0$;
  - *Consistente*, se $\lim_{n\to\infty} \Pr\{ |\hat{\theta}_n - \theta| > \varepsilon \} = 0$ per ogni $\varepsilon > 0$;
  - *Fortemente consistente*, se $\Pr\left\{ \lim_{n\to\infty} \hat{\theta}_n = \theta \right\} = 1$;
  - *Efficiente*, se $E\left[\left(\hat{\theta}_n - \theta \right)^2 \right]$ è un minimo rispetto a tutte le possibili stime puntuali di θ.
  - *Sufficiente* (statisticamente per θ), se $\hat{\theta}_n$ racchiude l'informazione completa su θ, ossia se la distribuzione condizionale di τ per un dato $\hat{\theta}_n$ non dipende da θ.

  Diversi metodi sono noti per la stima di θ, fra cui il metodo dei momenti, quantili, minimi quadrati e massima probabilità (quest'ultimo particolarmente usato in applicazioni ingegneristiche).

- **Interval estimation:**
  La point estimate fornisce rapidamente una stima ma non indicazioni sulla differenza tra stima e parametro reale. Più informazioni possono essere ottenute con una *stima di intervallo*. In tal caso, si definisce un intervallo randomico $[\hat{\theta}_l, \hat{\theta}_u]$ tale da includere il valore vero del parametro sconosciuto θ con una data probabilità γ. Tale intervallo è anche definito intervallo di confidenza e i suoi estremi *limiti di confidenza inferiore* e *superiore*. γ è il *confidence level* e la sua interpretazione è la seguente: in un numero crescente di campioni indipendenti di dimensione n, la frequenza relativa ai casi in cui gli intervalli di confidenza includono il parametro sconosciuto converge al livello di confidenza $\gamma = 1 - \beta_1 - \beta_2$, con $0 < \beta_1 < 1 - \beta_2 < 1$, β_1 e β_2 errori di probabilità relativi all'intervallo di stima.
  Questa metodologia, più che quella della point estimate, è stata impiegata nelle analisi di tesi.

Statistical Hypotesis Testing

Si veda lo statistical hypotesis testing applicato ad un caso tratto dalle analisi di affidabilità. Nella tesi è stato impiegato in particolare per risolvere problemi emergenti dalla tecnica Monte Carlo: si porta qui un problema differente come un'efficace presentazione degli elementi della metodologia.

L'obiettivo è risolvere il seguente problema:

> " sulla base della natura del problema o dell'esperienza, viene formulata una ipotesi H_0 per una proprietà statistica della variabile randomica considerata. Si stabilisca una regola (o test) tale da permettere l'accettazione o il rifiuto di H_0 sulla base delle realizzazioni indipendenti della variabile randomica in un opportuno campionamento. "

Se R è l'affidabilità sconosciuta di un item, sono possibili tre ipotesi H_0:

1.a) H_0: $R = R_0$;
1.b) H_0: $R > R_0$;
1.c) H_0: $R < R_0$.

Per capire se il failure-free time di un oggetto è distribuito secondo una distribuzione esponenziale $F_0(t) = 1 - e^{-\lambda t}$ con parametro λ sconosciuto o $F_0(t) = 1 - e^{-\lambda_0 t}$ con parametro λ_0 noto, la seguente ipotesi H_0 può essere formulata:

2.a) H_0: la funzione di distribuzione è $F_0(t)$;
2.b) H_0: la funzione di distribuzione è differente da $F_0(t)$;
2.c) H_0: $\lambda = \lambda_0$, ammesso che la funzione sia esponenziale ;
2.d) H_0: $\lambda < \lambda_0$, ammesso che la funzione sia esponenziale ;
2.e) H_0: la funzione è $1 - e^{-\lambda t}$, con parametro λ sconosciuto .

Si è soliti suddividere le ipotesi in parametriche e non parametriche, con ipotesi semplici o composte. Quando si testa una ipotesi, due tipi di errori sorgono:

1. Tipo I , quando si rigetta una ipotesi H_0 vera: la probabilità di questo errore è definita con α.
2. Tipo II, quando si rigetta una ipotesi H_0 falsa: la probabilità di questo errore è definita con β.

Se lo spazio di campionamento è diviso in due set complementari, $\mathcal{A}$ di accettazione e $\bar{\mathcal{A}}$ di rifiuto, gli errori di tipo I e II sono dati da

$$\alpha = \Pr\{ \text{sample in } \bar{\mathcal{A}} \mid H_0 \text{ true } \}$$
$$\beta = \Pr\{ \text{sample in } \mathcal{A} \mid H_0 \text{ false} \}$$

Entrambi gli errori sono possibili e non possono essere minimizzati contemporaneamente. Spesso α, è selezionato e il test viene stabilito in modo tale che per un certo H_1, β sia minimizzato.
La quantità α è detta *significance level*, mentre $1 - \beta$ è definita *potere del test*.

2.5 Basi di ottimizzazione

Nella presente appendice si descriverà il concetto di ottimizzazione su cui si basa l'Ottimizzatore impiegato negli studi, per poi passare ad una rapida descrizione delle famiglie di algoritmi di ottimizzazione in esso presenti, nonché dei principali algoritmi di ognuna di esse.

2.5.1 Concetto di ottimizzazione

In generale, per *ottimizzazione del design* si intende "trovare l'elemento migliore considerando un insieme di criteri predefiniti da un insieme di soluzioni praticabili". Il processo parte da un progetto esistente, che dovrebbe essere migliorato il più ragionevolmente possibile, tenendo conto del tempo disponibile, delle risorse e della conoscenza e capacità del progettista di formulare il problema. Nella valutazione di un progetto di solito non c'è un singolo aspetto che riassuma l' "essenza" del progetto stesso, ma l'ingegnere deve bilanciare una moltitudine di fattori (come prestazioni, durata, costo, ecc.) e di parametri necessari a misurare questi fattori l'uno rispetto all'altro, per arrivare a quella che ritiene sia la migliore combinazione. In altre parole, spesso non è opportuno fissare un'unica metrica nell'ottimizzazione del design, ma è meglio affrontare il problema considerando contemporaneamente una serie di aspetti da migliorare, ovvero più obiettivi di progettazione.

Un concetto importante nell'ottimizzazione multi-obiettivo è l' *ottimalità paretiana*. Una soluzione multi-obiettivo perfetta è impossibile; una soluzione ragionevole consiste nell'indagare su un insieme di compromessi ugualmente validi. Un *Fronte di Pareto* è un insieme di soluzioni che non sono dominate l'una rispetto all'altra, cioè nessun obiettivo di progetto può essere ulteriormente migliorato senza pregiudicarne un altro. In altre parole, mentre si passa da una soluzione paretiana a un'altra, c'è sempre una certa quantità di sacrificio in uno o più obiettivi per raggiungere un certo livello di guadagno negli altri. Quindi, matematicamente parlando, ogni punto *ottimo di Pareto* è ugualmente accettabile per un problema di ottimizzazione multi-obiettivo. In termini pratici, gli insiemi di soluzioni Pareto ottimali sono spesso preferiti alle soluzioni singole quando si considerano problemi della vita reale poiché la soluzione finale del decisore è sempre un compromesso basato su una preferenza personale o oggettiva.

Per molti problemi multi-obiettivo la dimensione dell'insieme ottimo di Pareto è enorme, se non infinita, quindi, identificarlo nella sua interezza è computazionalmente irrealizzabile. I progettisti devono quindi concentrarsi su un sottoinsieme della Pareto stessa, che dovrebbe essere il più uniformemente distribuito e diversificato possibile e catturare l'intero spettro di Pareto, comprese le estremità delle funzioni obiettivo, per fornire al decisore un quadro completo di compromessi.

2.5.2 Algoritmi di ottimizzazione

Nell'Ottimizzatore sono disponibili diversi algoritmi per l'ottimizzazione sia stocastica che deterministica, nonché algoritmi ibridi che combinano i vantaggi (ma anche alcuni inconvenienti) di entrambi gli approcci. Non esiste infatti una "ricetta universale" per una corretta ottimizzazione. La scelta della strategia o combinazione di strategie più adatta dipende dalla conoscenza e comprensione da parte dell'ingegnere della teoria della tecnica di ottimizzazione e del problema di progettazione (ad esempio, configurazione del risolutore, parametrizzazione del problema, scelta corretta di obiettivi e vincoli, ecc.). Quest'ultimo potrebbe non essere strettamente correlato al funzionamento degli algoritmi, ma può avere un impatto significativo sull'esito del problema di ottimizzazione.

Concetti importanti relativi al funzionamento degli algoritmi di ottimizzazione sono la loro *robustezza*, *accuratezza* e *velocità di convergenza*. La robustezza di un ottimizzatore si riferisce alla sua capacità di trovare l'estremo assoluto della funzione obiettivo anche partendo lontano dalla soluzione finale, senza convergere prematuramente ad un ottimo locale. L'accuratezza di un ottimizzatore si riferisce alla sua capacità di avvicinarsi il più possibile all'estremo della funzione

obiettivo. Il tasso di convergenza di un ottimizzatore si riferisce alla sua capacità di trovare la soluzione ottimale, cioè di raggiungere la convergenza, dopo il minor numero di valutazioni possibili.

2.5.2.1 Algoritmi Evolutivi

Gli Algoritmi Evolutivi (EA) si basano sull'imitazione dei processi genetici degli organismi biologici e affrontano i problemi di ottimizzazione implementando un processo ripetuto di (piccole) variazioni stocastiche seguite da selezione: in ogni generazione (o iterazione), vengono generati nuovi figli (o soluzioni candidate) dai loro genitori (soluzioni candidate già valutate), la loro bontà viene valutata e i figli migliori vengono selezionati per diventare i genitori della generazione successiva.
I principali algoritmi facenti parte di questa famiglia sono:

1. MOGA-II
2. NSGA-II
3. Evolution Strategy
4. Many-Objective Algorithm

Il Multi-Objective Genetic Algorithm (MOGA-II) è la versione proprietaria dell'algoritmo genetico multi-obiettivo che utilizza un elitarismo multi-ricerca intelligente ed efficiente in grado di preservare soluzioni eccellenti (di Pareto o non dominate) senza convergere prematuramente in un ottimo locale. L'elitismo migliora la convergenza dell'algoritmo e assicura che la bontà di ogni nuova generazione sia maggiore della bontà della generazione madre.

Il Non-dominated Sorting Genetic Algorithm II (NSGA-II) implementato si basa sull'algoritmo genetico di ordinamento non dominato II (NSGA-II) sviluppato dal prof. K.Deb et al. del Kanpur Genetic Algorithms Laboratory (KanGAL). Questi algoritmi sono in generale impegnativi dal punto di vista computazionale, soprattutto nel caso di popolazioni molto grandi, poiché l'identificazione di individui appartenenti al primo fronte non dominato richiede il confronto di ogni soluzione con ogni altra soluzione. Tuttavia, NSGA-II implementa una strategia di ordinamento intelligente non dominata che richiede molti meno calcoli. Per ogni soluzione viene calcolato un conteggio delle dominazioni: quelle con conteggio delle dominazioni 0 appartengono al primo fronte. Quindi il conteggio del dominio di tutte le rimanenti soluzioni dominate viene ridotto di 1 e quelli risultanti con il conteggio del dominio 0 sono classificati al secondo fronte. Questa procedura viene ripetuta fino a quando tutti i design non sono stati ordinati. NSGA-II implementa inoltre una tecnica che garantisce la diversità e la diffusione delle soluzioni sul fronte di Pareto, permettendo quindi una migliore distribuzione ed uniformità delle soluzioni sulla Pareto stessa.

L'approccio degli Evolution Strategy (ES) è stato utilizzato per la prima volta presso l'Università tecnica di Berlino. Durante la ricerca delle forme ottimali dei corpi in un flusso, i classici tentativi con le coordinate e le ben note strategie basate sul gradiente non hanno avuto successo. Quindi, l'idea è stata concepita per procedere strategicamente. Ingo Rechenberg e Hans-Paul Schwefel hanno proposto l'idea di provare cambiamenti casuali nei parametri che definiscono la forma, seguendo l'esempio dell'evoluzione naturale. Di solito, c'è un'enorme differenza tra l'ottimizzazione matematica e l'ottimizzazione nelle applicazioni del mondo reale. Pertanto, gli ES sono stati inventati per risolvere problemi di ottimizzazione tecnica in cui solitamente non sono disponibili funzioni obiettivo analitiche.

Il Many-Objective algorithm (MANY) è un algoritmo genetico progettato per l'ottimizzazione di problemi con quattro o più obiettivi indipendenti. Questi problemi non possono essere risolti in modo efficiente con le strategie classiche per una serie di motivi, come l'alto costo computazionale e le difficoltà derivanti dal confronto e dalla selezione di soluzioni in più di tre dimensioni.

2.5.2.2 Ottimizzatori euristici

I principali algoritmi facenti parte di questa famiglia sono:

1. SIMPLEX
2. MOSA
3. MOGT
4. MOPSO

L'algoritmo di ottimizzazione Simplex implementato è il metodo Nelder-Mead utilizzato per trovare il minimo o il massimo di una funzione obiettivo in uno spazio multidimensionale. Simplex viene utilizzato per risolvere problemi di ottimizzazione non lineare mono-obiettivo. Non calcola derivate, il che lo rende più robusto dei metodi basati su gradiente.
Questo algoritmo euristico utilizza il concetto di simplesso, che è un poliedro con N+1 vertici in uno spazio N-dimensionale (N rappresenta il numero di variabili di input). Confronta i valori della funzione obiettivo a N+1 vertici e sposta gradualmente il poliedro verso il punto ottimale sostituendo iterativamente il vertice peggiore con un punto spostato attraverso il baricentro degli N punti rimanenti. L'algoritmo si arresta quando raggiunge l'accuratezza fissata con un opportuno indice o il numero massimo di valutazioni, a seconda di quale sia raggiunto per primo.
Il movimento del simplex è dato da quattro operazioni: riflessione, espansione, contrazione e contrazione multipla. Si inizia sempre con la Riflessione e i successivi movimenti dipendono esclusivamente dall'idoneità del nuovo punto rispetto ad altri vertici simplex.

L'algoritmo Multi-Objective Simulated Annealing (MOSA) è un metodo stocastico ispirato dal processo di ricottura in metallurgia, una tecnica che prevede il riscaldamento e il raffreddamento controllato del materiale per modificarne le proprietà fisiche modificandone la struttura interna. Ad alte temperature le molecole del materiale si muovono liberamente. Quando il materiale si raffredda, la sua nuova struttura si fissa, facendo sì che il materiale mantenga le sue nuove proprietà. Nel MOSA la temperatura è utilizzata come variabile esterna: all'inizio il suo valore è impostato alto e viene gradualmente ridotto man mano che l'ottimizzazione avanza. Quando la temperatura è elevata, all'algoritmo è consentito con una certa frequenza di accettare soluzioni peggiori della soluzione attuale. In questo modo MOSA può esplorare lo spazio di progetto ed evitare di rimanere bloccati in un ottimo locale. Al diminuire della temperatura, aumenta anche la probabilità che vengano accettate soluzioni peggiori. Ciò consente all'algoritmo di concentrarsi sull'area di ricerca in cui si trovano le soluzioni ottimali. Questo processo di raffreddamento graduale è ciò che rende MOSA molto efficace nel trovare una soluzione quasi ottimale di problemi con molti ottimi locali.

L'algoritmo Multi-Objective Game Theory (MOGT) si basa sulla Game Theory, che è stata formulata matematicamente da JF Nash nei primi anni '50 e ha trovato la sua prima applicazione in economia, in particolare per risolvere problemi relativi alle decisioni che hanno qualche effetto su ambiti diversi e spesso in contrasto tra loro. MOGT è particolarmente efficiente con problemi altamente vincolati e non lineari. Questo algoritmo si basa su un gioco competitivo tra giocatori. Ci sono tanti giocatori quanti sono gli obiettivi del problema di ottimizzazione e ogni giocatore ha il compito di ottimizzare l'obiettivo assegnatogli. Il numero di obiettivi non può essere maggiore del numero di variabili di input.

Il Particle Swarm Optimization (PSO) è una tecnica di ottimizzazione stocastica basata sulla popolazione sviluppata dal Dr. Eberhart e dal Dr. Kennedy nel 1995, ispirata al comportamento sociale dello stormo di uccelli. Il Multi-Objective Particle Swarm Optimization (MOPSO) è un algoritmo iterativo di ottimizzazione multi-obiettivo che condivide molte somiglianze con le tecniche di calcolo evolutivo come gli algoritmi genetici (GA), ma a differenza di GA, MOPSO non ha operatori di evoluzione (come crossover e mutazione). È un po' meno robusto del GA, ma lo compensa con una maggiore velocità di convergenza che lo rende una buona alternativa al GA quando le valutazioni richiedono un maggiore sforzo computazionale. MOPSO implementa una strategia di

elitismo e conserva le migliori soluzioni in un archivio chiamato *élite set*. L'élite set viene aggiornato con nuove soluzioni che sostituiscono qualsiasi altra soluzione che dominano. In questo modo l'archivio viene mantenuto libero da dominazioni. Se l'élite set raggiunge la sua capacità massima e nessuna soluzione domina un'altra, le soluzioni in eccesso vengono scartate secondo un criterio di selezione casuale. In MOPSO ogni potenziale soluzione è chiamata *particella* (particle) e la popolazione di soluzioni è chiamata *sciame* (sworm). Le particelle volano attraverso lo spazio di progetto con una certa velocità seguendo le attuali particelle ottimali chiamate *guide*.

2.5.2.3 Algoritmi multi-strategia

I principali algoritmi facenti parte di questa famiglia sono:

1. MEGO
2. FAST
3. pilOPT

Il Multi-Objective Efficient Global Optimization (MEGO) è un ottimizzatore multi-obiettivo surrogato basato su *processi gaussiani* (GP). Trova l'optimum globale in base al rispetto del cosiddetto "criterio di riempimento" per la selezione dei design. Per problemi con un solo obiettivo il criterio di riempimento è la massimizzazione del cosiddetto "Miglioramento atteso". Per problemi multi-obiettivo il criterio di riempimento è la massimizzazione di un'estensione del "Miglioramento atteso" basato sul calcolo dell'ipervolume. Il "Miglioramento atteso" è calcolato con modelli GP ed è massimizzato con un algoritmo genetico come parte del problema di ottimizzazione interna. A seguito di questa ottimizzazione interna, vengono aggiunti in modo iterativo nuovi design nella regione di interesse.

Il FAST è un algoritmo di ottimizzazione che combina strategie di ottimizzazione reali e basate su modelli virtuali (Response Surface Model – RSM). Sia l'ottimizzazione reale che quella virtuale sono eseguite da uno degli algoritmi evolutivi o euristici disponibili nell'Ottimizzatore per la risoluzione di problemi singoli e multi-obiettivo. I design generati dall'algoritmo FAST vengono scambiati tra i suoi ottimizzatori reali e virtuali ad ogni generazione, aumentando la loro robustezza e migliorando la convergenza. I nuovi design aggiunti al database FAST, siano essi provenienti dall'ottimizzazione reale o virtuale, vengono utilizzati attivamente sia per migliorare l'accuratezza degli RSM addestrati sia per arricchire la popolazione dell'ottimizzatore reale.

Il pilOPT è un algoritmo auto-adattativo multi-strategia che combina i vantaggi della ricerca locale e globale e bilancia in modo intelligente l'ottimizzazione reale e basata su RSM (virtuale) nella ricerca del fronte di Pareto. pilOPT offre prestazioni notevoli anche quando si gestiscono funzioni di output complesse e problemi vincolati. È generalmente consigliato per problemi con più obiettivi, ma può anche gestire problemi con un singolo obiettivo. pilOPT è stato progettato per sfruttare in modo efficiente il tempo di valutazione disponibile (determinato dal solutore integrato) eseguendo costantemente i processi supportati dalle sue diverse strategie di ottimizzazione interna. Inoltre, regola dinamicamente il rapporto tra le valutazioni dei design reali e virtuali in base alle loro prestazioni. Questo lo rende particolarmente adatto per simulazioni computazionalmente pesanti. Si noti che questo può essere percepito come un comportamento computazionalmente intensivo (anche perché ha una componente interna basata su RSM), specialmente quando le valutazioni vengono eseguite localmente o quando la soluzione del problema non sembra essere eccessivamente complessa, ma consente a pilOPT di trovare i design ottimali ben distribuiti sul fronte paretiano.

Capitolo 3 Caratteristiche dell'OLSO (v.1)

Come visto nel primo capitolo, questa tesi si è incentrata su due obiettivi fondamentali:

I) La realizzazione della prima versione di un software per l'ottimizzazione del supporto logistico durante la fase di Operazione di un sistema aeronautico, con focus alla gestione della flotta di propulsori. Nome: *Operation-oriented Logistic-Support Optimizer – v.1 (OLSO v.1)* .

II) Il compimento di una delle analisi di Development Level 1, in particolare l'analisi del metodo di ottimizzazione per anticipo.

In questo capitolo si va a presentare il software OLSO v.1, focalizzando l'attenzione sulle sue caratteristiche concettuali (funzionalità e metodi) più che sull'implementazione specifica (forma e contenuti dettagliati dei file, script ecc.). Lo scopo è presentare il funzionamento del software così da poter comprendere i risultati dell'analisi, trattate invece nel prossimo capitolo.

Struttura modulare dell'OLSO v.1

Da come è stato pensato l'OLSO in versione completa (v.3), esso è suddiviso nei tre moduli principali (Maintenance Planner, Spare Parts Manager, TAT Forecaster) e ognuno di essi è suddiviso a sua volta in tre sotto-moduli (Simulatore, Calcolatore dei costi, Ottimizzatore). L'OLSO v.1 è però una versione semplificata in cui fra i tre simulatori, solo quello del Maintenance Planner è presente: pertanto, il senso di avere una struttura divisa in moduli e sotto-moduli viene fortemente ridotto.
Per questa ragione, l'OLSO v.1 è stato sviluppato secondo una struttura leggermente diversa, eliminando il livello di modulo e mantenendo solo i sotto-moduli, così come mostrato in Figura 9.

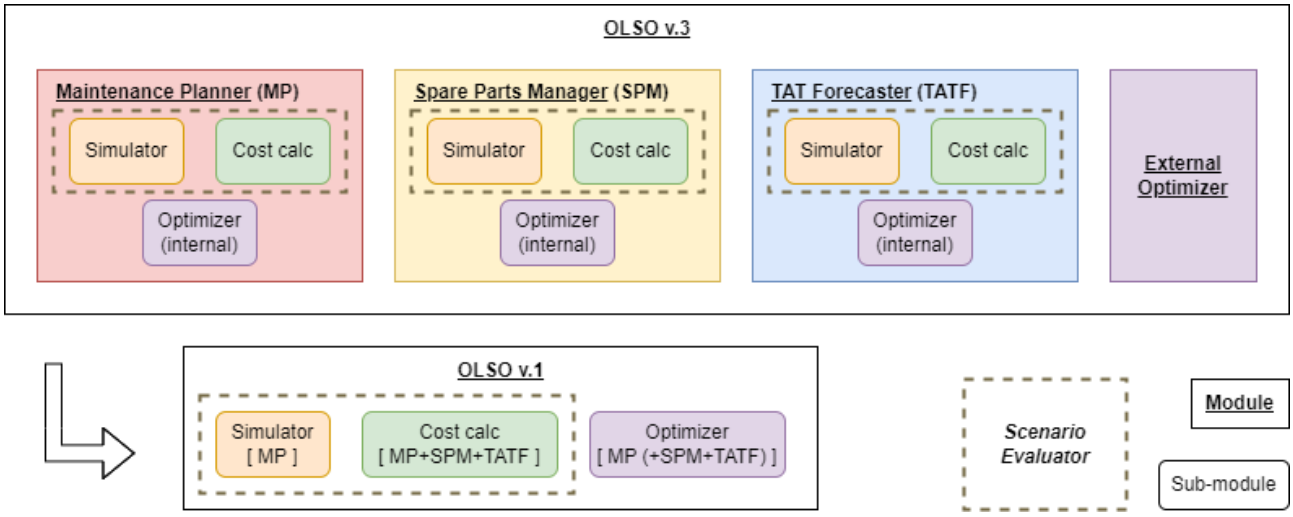


Figura 9) Struttura modulare dell'OLSO e sua semplificazione.
Sono mostrati solo i moduli di processo, e non anche quelli dei dati.

In particolare:

- il Simulatore contiene le dinamiche dell'ambiente operativo (voli e manutenzione), relativi come visto al funzionamento del Maintenance Planner ;
- il Calcolatore dei costi quantifica, attraverso un modello semplificato basato su delle Cost Estimation Relationships (CERs), tutti e tre gli elementi significativi dei moduli: penali (MP), parti (SPM) e TATs (TATF) ;
- l'Ottimizzatore è costruito per affrontare le analisi di DL1, e in particolare quello implementato in questa tesi è rivolto nello specifico all'ottimizzazione per anticipo, focus di questa trattazione[35].

[35] Per questa ragione, in figura l'indicazione dei moduli "SPM" e "TATF" è messa fra parentesi.

Funzionamento concettuale dell'OLSO v.1

Prima di entrare nella spiegazione dettagliata delle caratteristiche interne di ogni modulo, si vuole dare una panoramica sul suo funzionamento. Si veda la Figura 10: essa mostra lo schema concettuale dell'OLSO v.1 specializzato in particolare per l'ottimizzazione secondo metodo degli anticipi[36].

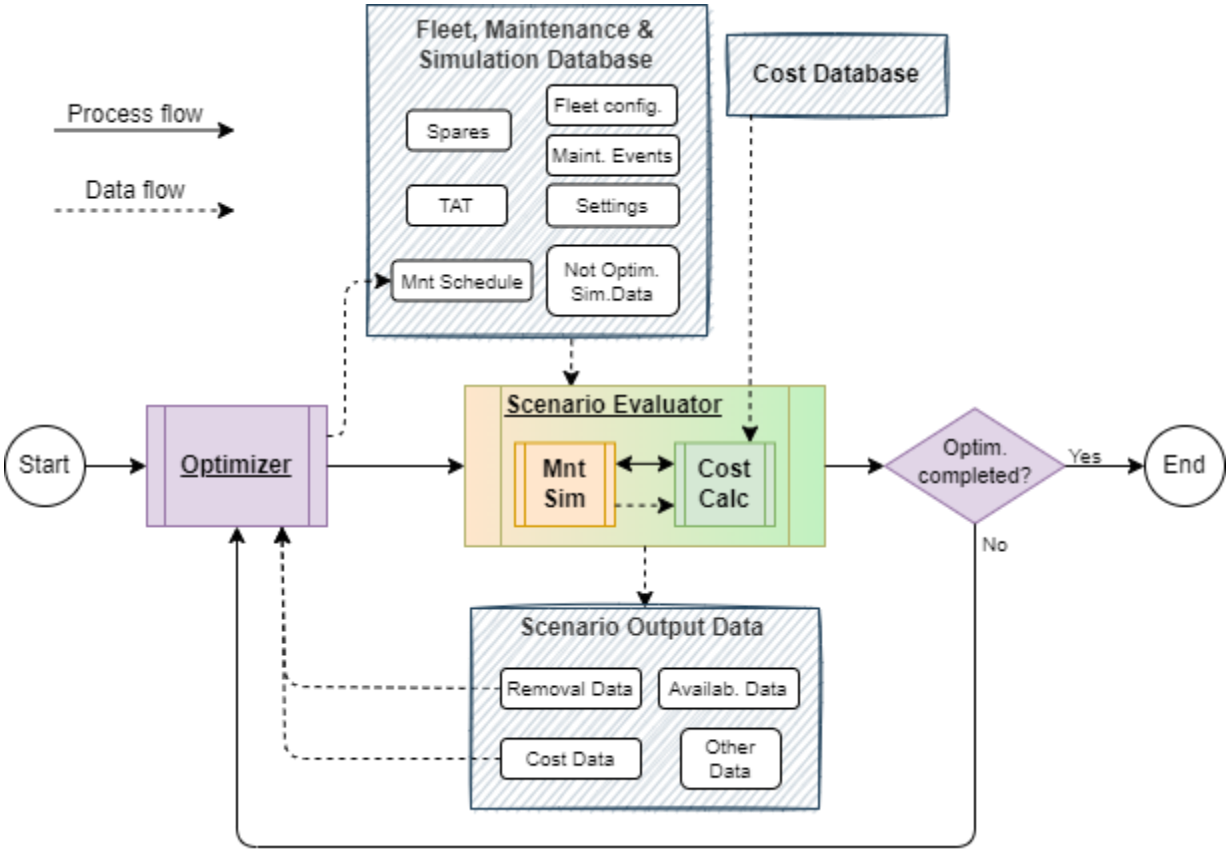


Figura 10) Schema concettuale del funzionamento dell'OLSO v.1 per l'ottimizzazione secondo metodo degli anticipi.

Il funzionamento complessivo è più semplicemente comprensibile se si considera l'Ottimizzatore come elemento principale, con lo *Scenario Evaluator* (Simulatore manutentivo + Calcolatore dei costi) come modulo di supporto a fornirgli i dati per compiere le decisioni.
L'ottimizzazione lavora sulla schedulazione degli eventi di manutenzione programmata, in particolare scegliendo di quanti giorni anticipare un certo evento (sostituzione o revisione). Non tutti gli eventi sono però ugualmente impattanti sullo scenario, per cui anzitutto si analizza lo scenario "di base", ossia quello non ottimizzato, da cui vengono selezionati solo quelli (schedulati) con la più alta probabilità di accadimento. Si identificano così un certo numero di *items di anticipo*, e il *numero di giorni d'anticipo* associati ad ognuno di essi compongono il set di variabili di ottimizzazione.
Come obiettivo di ottimizzazione è stato scelto il *cost per flight hour (CPFH)*, definito come il costo totale (medio) della manutenzione per ora-velivolo volata. Tale grandezza viene fornita, per ogni optimization design indagato, dallo Scenario Evaluator: esso vi estrae informazioni attraverso una metodologia basata su simulazione Monte Carlo e, dopo adeguato post-processing dei dati delle singole iterazioni, fornisce all'Ottimizzatore i dati in termini di distribuzione di CPFH.
Confrontato il CPFH di optimization trial con quello dello scenario non-ottimizzato, e applicate considerazioni statistiche, l'Ottimizzatore sceglie il nuovo optimization design, e il ciclo ricomincia.

[36] Ci si è concentrati su di esso per adattarlo alle analisi che si volevano svolgere.
Al capitolo conclusivo "Conclusioni e sviluppi futuri" sarà mostrato lo schema relativo all'OLSO v.1 che implementa anche logiche per le altre due ottimizzazioni di DL1, ossia la riduzione dei TAT e la modifica del numero di parti a magazzino (in maniera statica, poiché i moduli completi SPM e TATF non sono ancora presenti).

Struttura espositiva del capitolo

La struttura espositiva di questo capitolo è affrontata seguendo quella (semplificata) dell'OLSO v.1:

.1 si affronta dapprima il Simulatore (detto "manutentivo" poiché relativo solo al Maintenance Planner), analizzato sia in termini di modello che dando qualche informazione sui processi implementativi (pre-processing, processing e post-processing);

.2 si passa al Calcolatore dei costi, analizzato in termini di modello dei costi, processo di assegnazione dei costi e analisi statistica finale ;

.3 si tratta l'Ottimizzatore, esponendone il processo logico, ossia la successione di passi nel ciclo iterativo che sfrutta l'operato di Simulatore e Calcolatore dei costi ;

.4 si espongono brevemente i punti fondamentali relativi ai File Input/Output, in particolare mostrando i grafici principali in preparazione all'analisi, che segue al capitolo successivo ;

.5 infine si mostra il risultato finale, l'OLSO v.1, dove tutti i precedenti elementi vengono integrati nello stesso ambiente.

3.1 Il Simulatore manutentivo

Concisamente, possiamo riassumere l'operato del Simulatore manutentivo come segue:

> " Il Simulatore manutentivo è un software di simulazione event-based discreta, che riproduce nel tempo il comportamento della flotta di aerei, motori e componenti che operano sottoposti a un certo scheduling dei voli, assoggettati a fenomeni aleatori di usura, failure randomiche e alle dinamiche di magazzino, e rispettanti determinati vincoli e oneri contrattuali.
> Il Simulatore manutentivo produce come output principale la cronologia degli eventi manutentivi (sbarchi motore, sostituzioni, riparazioni, acquisti ecc.) e lo status di ogni velivolo, motore e componente, da cui estrapolare la disponibilità dei motori. "

3.1.1 Il Modello di simulazione

Il modello di simulazione si costruisce attorno a quattro elementi:

- Soggetti: sono gli enti che compiono e subiscono le azioni nella simulazione. Sono suddivisi in tre gruppi di ordine gerarchico sequenziale; in ordine discendente si hanno: velivoli, motori e items[37]. Ogni soggetto ha una doppia essenza: quella di *tipo (type)* e quella *seriale (serial)*[38].
- Stato: associato ad un ente seriale, ne definisce lo stato in un dato momento della simulazione. Ad esempio, indicano il luogo (virtuale) dove si trovano: in operazione o no, montati o no, in riparazione, al magazzino, ecc.
- Eventi: accadimenti della simulazione che causano un cambiamento di stato in uno o più enti. Essi sono suddivisibili in: "*eventi di fermo*", che causano l'interruzione dell'operatività del velivolo e richiedono un'azione manutentiva sul motore; ed "eventi non di fermo", per tutti gli altri. Esistono altre sotto-categorie, che verranno discusse in seguito.
- Logiche Manutentive: definiscono le azioni da compiere a seguito di un evento. Ad esempio, definiscono se un item è riparabile o meno.

Nel seguito si approfondiscono i due elementi "Soggetti" ed "Eventi".

[37] Tradizionalmente tradotto con "componenti", si preferisce lasciarlo in forma inglese perché "componente" viene usato in questo testo come traduzione di "component", che ha un significato più specifico. Infatti, *item* è un qualsiasi elemento (hardware, in tal caso) dotato di una qualche individualità, e può essere di vari livelli gerarchici: part, component, equipment. Vedere la sezione sulle definizioni RAMS.

[38] *Type*: archetipo concettuale di un ente, definito dall'insieme delle sue caratteristiche.
Serial: unità specifica di un ente. Ogni seriale è appartenente a un tipo, ma non vale il contrario.

Soggetti:

- **Velivoli:** si può simulare una flotta con un numero a piacere di velivoli. Ogni velivolo può avere un numero a piacere di motori. Il tipo di velivolo influenza il tipo e numero di motori che monta.
- **Motori:** vi possono essere un numero a piacere di motori, alcuni montati sul velivolo, altri no.
  I motori vengono ulteriormente suddivisi per tipo in base alla versione, detta *Tranche*. Il tipo di motore (tranche inclusa) definisce il numero e tipo di item che deve montare.
- **Items:** vi possono essere un numero a piacere di items, alcuni montati sul motore, altri no.
  Come per i motori, anche il tipo degli items è influenzato sia dal *tipo base* che dalla *tranche*, ed entrambe devono essere coerenti col motore su cui devono essere montati.
  Gli items si suddividono in *accessori* o *moduli*, e questo influisce sulle associate logiche di manutenzione. Anche il motore nella sua interezza viene visto come modulo, ma questo è utile più che altro nell'implementazione pratica software, qui non particolarmente interessante.
  I moduli possono possedere elementi detti *Life-Limited Parts (LLPs)*, ossia parti che hanno una vita massima stabilita in termini di ore di volo, al cui raggiungimento necessitano una sostituzione.

Eventi:

- **Eventi di fermo:**
  Sono eventi che causano l'interruzione dell'operatività del velivolo e richiedono un'azione manutentiva sul motore. Ogni tipo di item ha associati un certo numero di eventi di fermo, ed essi possono essere suddivisi nelle seguenti categorie.
  - **Eventi schedulati:** dette anche *scadenze programmate*, sono eventi di fermo definiti dal piano manutentivo.
    Esistono due tipi di eventi schedulati:
    - ***Overhaul:*** indica un evento di *revisione*, e si associa solo ad un modulo.
    - ***LLP removal:*** è un evento che causa la sostituzione di una Life-Limited Part.
      Una rimozione di LLP può avvenire per due ragioni distinte:

      Primary) la LLP non ha un numero di ore di volo residue tali da poter compiere la successiva missione ;

      Secondary) a fronte di una riparazione motore che ne causa lo sbarco, la LLP non soddisfa un requisito sul numero di ore minimo per essere rimesso in servizio, e tale valore di requisito è detto *Minimum Issued Service Life (MISL)*.
  - **Eventi non schedulati:** sono rappresentati dalle *failures* e dal *subsidiary damage*.
    - ***Failures)*** Ogni item-type può avere un numero variabile di failure modes, e ognuno di essi è modellato da una distribuzione di probabilità mediante curve di Weibull (tramite fornitura dei due parametri, di scala e di forma).
    - ***Subsidiary damage)*** Rappresenta una (hidden) failure scoperta nelle ispezioni post-fermo. Viene modellato tramite la probabilità di trovare l'item guasto a fronte di un fermo motore.

  Inoltre, gli eventi di fermo possono essere classificati in:
  - *Primary*, se sono la causa diretta del fermo velivolo (e.g. eventi schedulati e failures) ;
  - *Secondary*, se avvengono a seguito di controlli successivi a un fermo avvenuto per un qualche evento primary distinto (come quelli sulla MISL o il subsidiary damage).

- **Eventi non di fermo:** comprendono una gamma variegata di eventi, come spostamenti (virtuali) di luogo, ritorno in condizioni operative, montaggio di un ente sull'ordine gerarchico superiore, ecc.

3.1.1.1 Il processo manutentivo

Si considera una flotta di velivoli sottoposta a un certo piano di volo, che definisce quando ognuno di essi deve compiere una determinata missione. I velivoli operativi, ovvero quelli in grado di volare, compiono le missioni assegnategli, accumulando ore di volo sui propri componenti. Quando uno di essi subisce una failure, o si verifica una scadenza programmata (sia essa un fine vita di parte o revisione TBO), il velivolo viene fermato e vengono rimossi i componenti interessati e compiuti controlli per valutare la rispondenza ai requisiti sulla MISL
Se sono già disponibili a magazzino i sostituti dei componenti rimossi, essi vengono subito montati. A seconda della tipologia di componente interessato (se accessorio o modulo), si hanno delle differenze nella gestione manutentiva del motore, espresse da un codice detto *Maintenance Level*:

- *ML1:* un'operazione di rimozione e sostituzione che può essere eseguita con motore montato sul velivolo viene caratterizzata dal codice "*ML1*", e la possono realizzare solo gli accessori.
- *ML2:* viceversa, se almeno un modulo è interessato, per la sua rimozione e sostituzione è necessario sbarcare il motore, e una tale operazione è detta "a *ML2*".
  Sia operazioni a ML1 che a ML2 per essere concluse impiegano un tempo detto TAT_{exch}, differente a seconda dell'item. Nel caso di motore, rappresenta il tempo per il rimontaggio sul velivolo.
- *ML3:* gli item, se sono riparabili (come i moduli), rimossi vengono mandati in riparazione, e una tale operazione manutentiva è caratterizzata dal codice "*ML3*", e risulta generalmente molto lunga. Se non sono riparabili (come accessori o LLP) se ne ordinano di nuovi. In ognuno dei due casi il tempo che ci mettono per tornare operativi è detto TAT_{rep}, al termine del quale vanno al magazzino. Eventi di revisione sono anch'essi associati a ML3.

Se invece non sono disponibili tutti i componenti per la sostituzione, il motore non può tornare operativo, e viene messo da parte in attesa dei componenti necessari. Se un altro motore a magazzino è disponibile, si compie la sostituzione completa, che prende un tempo pari al TAT_{exch} del motore. Per come è stato implementato il modello, se un velivolo ha allo stesso tempo la possibilità di sostituire i componenti sul proprio attuale motore (e aspettare il suo ritorno) oppure sostituire l'intero motore con un alto a magazzino, sceglie sempre la seconda via, poiché più rapida.
La simulazione del comportamento della flotta sarà affrontata nel dettaglio nel prossimo paragrafo.

3.1.2 L'implementazione: fasi di processo

Si è terminata la presentazione del Modello di simulazione. Di seguito si vanno a dare alcune note di implementazione, suddividendo l'esposizione nei tre momenti di pre-processing, processing e post-processing. Le si riassume brevemente:

- ***Pre-processing:*** è la fase di raccolta dati e dei calcoli preliminari, utile per le operazioni successive. Come primo passo si leggono i file di input e li si analizza con apposite function di controllo per accertarsi che non vi siano eventuali errori o violazioni di ipotesi. Successivamente si svolgono dei semplici calcoli per ottenere le condizioni iniziali della simulazione.
- ***Processing:*** è la fase di calcolo vero e proprio.
  Il simulatore manutentivo riproduce giorno per giorno le azioni dei velivoli e di manutenzione, registrando la cronologia degli eventi. La simulazione termina quando si arriva all'ultimo giorno calendariale richiesto nei file di input.
- ***Post-processing:*** fase di rielaborazione dei risultati del processing, e creazione dei files di output.

3.1.2.1 Pre-processing

Dopo l'avvio del modulo di Simulazione, la prima azione è quella di creazione del database simulativo che servirà per le successive fasi, poiché vi sono contenute tutte le informazioni su velivoli, motori e items, in termini di scheduling delle missioni, manutenzione e dati di setting della simulazione. Per farlo, si caricano i dati di interesse leggendoli dai vari file di input.

Lettura ed estrazione dati generali, di manutenzione e configurazione

Si apre il file dei dati di input e come primo passo lo si legge nella sua interezza, caricando in forma tabellare le informazioni in MATLAB; successivamente si passa all'estrazione dei dati dalla forma tabellare in una più consona ad essere maneggiata durante la simulazione, riordinando le informazioni in modo da catalogarle in funzione di precisi codici[39], assegnati a velivoli, motori e items in modo da identificarli in maniera univoca e semplice.

La catalogazione delle informazioni riguardanti i soggetti manutentivi (velivoli, motori e items) avviene secondo tre grandi gruppi:

- dati *Generali*: riguardano sostanzialmente i nomi propri di velivoli, motori e items;
- dati sugli *Eventi manutentivi*: riguardano i componenti e i loro failure modes, gli eventi schedulati di revisione o limiti di vita (per eventuali parti a vita limitata), le ore di volo già accumulate, e infine le informazioni sui tempi di manutenzione (i TAT).
- dati sulla *Configurazione*: specificano quali (o quali tipi) di items sono associati a un certo (o certo tipo di) motore, o quali (/quali tipi di) motori sono associati a un certo (/un certo tipo di) velivolo.

Controllo del database

Si operano dei controlli per assicurarsi che le ipotesi principali di database manutentivo vengano rispettate e, in caso contrario, il software si arresta generando un errore che specifichi il tipo di violazione.

Generazione scheduling delle missioni

Come secondo passo si procede a creare lo scheduling delle missioni. Esse vengono generate in maniera randomicamente sparsa durante gli anni, ma in modo da rispettare i requisiti (indicati nel file di input) sul numero massimo di missioni per un certo velivolo in un certo anno.

Estrazione dati di anticipo

Si procede leggendo i dati di ottimizzazione legati all'anticipo, dal foglio "*Ant_data*". Partendo da essi, si memorizzano i giorni in cui si desidera imporre la rimozione per un certo item.

Estrapolazione dei dati di inizio simulazione

Infine, come ultimo step prima del processing, si impostano le variabili di inizio simulazione, ad esempio si calcola lo Stato dei vari soggetti manutentivi al primo giorno. I dati qui impostati hanno la principale caratteristica di essere costanti al variare delle iterazioni, poiché non dipendono da variabili aleatorie; gli eventi di failure, infatti, non vengono trattati qui, ma generati nuovi all'inizio di ogni nuova iterazione, così come sarà spiegato di seguito.

[39] Per fare ciò si è fatto uso di una rigida nomenclatura, che per semplicità di trattazione verrà qui omessa. All'occorrenza, verranno spiegati i nomi più importanti nel caso sia più avanti indispensabile nominarli. Si ricorda che comunque eventuali termini sono consultabili nel Glossario o nella Lista degli acronimi.
Alcuni esempi sono i codici di nome *IC*, *ESC*, *IPSC* e *AQC*. In generale finiscono tutti in '*C*' (*Code*, codice)

3.1.2.2 Processing

La fase di processing simula il sistema oggetto di studio e ne salva i dati per rimaneggiarli nella fase successiva. Poiché il sistema studiato possiede una forte componente aleatoria (le failures), è necessario andare ad analizzarlo compiendo più volte la simulazione sul periodo di interesse, così da estrapolarne dati da interpretare secondo una logica statistica, e cogliere così le caratteristiche stocastiche del problema. Questa tipologia di approccio alla simulazione viene chiamata *Monte Carlo based*.
Tutte le iterazioni sono dunque ovviamente strutturate in maniera identica fra loro.

Di seguito, si analizzerà dunque anzitutto il contenuto di una di esse. Al suo termine, per completezza, come piccola appendice si riporteranno alcune note tecniche sull'implementazione dei vari cicli di iterazione.

La singola iterazione

La prima fase della singola iterazione Monte Carlo è l'assegnazione randomica delle failures ai vari items. È importante che essa venga fatta ora, e all'inizio di ogni nuova iterazione, per accertarsi che la simulazione sia effettivamente aleatoria fin dai primi momenti e diversa dalle altre iterazioni. Successivamente, dopo la fine di questo breve pre-processing di singola iterazione, si può passare alla simulazione temporale dello scenario manutentivo.

Si procede giorno per giorno, processando l'operatività dei velivoli e le manutenzioni a cui sono sottoposti. Si noti che la precisione[40] della simulazione è (all'incirca[41]) quella del giorno, cioè essa riesce a distinguere (rispetto alla controparte reale) due eventi nella corretta maniera, e in particolare nella corretta sequenza, quando questi sono ragionevolmente distanti rispetto alle 24 ore. Ad esempio, le missioni vengono schedulate indicandone il giorno, ma non l'ora precisa; così, anche le procedure manutentive non hanno luogo ad un momento preciso della giornata.
Per compensare questa mancanza di rigore nella sequenzialità degli eventi, si è pensato di imporre per ipotesi una loro determinata successione, anche e soprattutto per ottenerne un beneficio computazionale e di semplicità nella scrittura del codice, oltre che concettuale. È così che si è scelto di suddividere la giornata in vari ma precisi momenti, i cui due principali sono il "giorno" e la "sera"[42]: nel primo si compiono le missioni e possibili rimozioni/sostituzioni, nel secondo si affrontano operazioni di sostituzione su motori o velivoli (che non sono stati interessati da eventi durante il giorno) e, come ultima azione della giornata, si aggiornano i dati di status dei soggetti manutentivi in vista del giorno che verrà l'indomani.
Si affrontino ora nel dettaglio i vari momenti della giornata in cui si articola la simulazione.

[40] Il concetto di precisione è qui utilizzato in una forma più intuitiva che rigorosamente matematica, ma è comunque utile per comprendere il significato delle sue implicazioni.
[41] "all'incirca" proprio in conseguenza del non-rigore nella definizione del concetto di precisione, per la quale non possiamo assegnare un valore piuttosto che un altro.
[42] Ovviamente, la valenza temporale di "giorno" e "sera" è legata alla misura in cui siano un espediente per immaginare con più semplicità la sequenza di eventi, più che un rimando al fatto che avvengano davvero in quei momenti della giornata.

- **<u>Momento 1)</u> <u>IL GIORNO:</u>** si compiono le missioni, rimozioni e sostituzioni

  - **1.1) Si compiono le missioni**

    Gli aerei operativi compiono le ore di volo che gli spettano. Terminate le missioni, tornano all'hangar.

    Durante la missione, solo un evento di failure può far fermare il velivolo. Se succede, si porta anzitempo il velivolo all'hangar delle rimozioni. La terminazione della missione (dal momento in cui la failure viene identificata) e il conseguente ritorno alla base vengono considerati, per ipotesi, istantanei.

  - **1.2) Ritorno all'hangar: identificazione, rimozione e sostituzione**

    - *1.2.I)* *<u>Identificazione:</u>*

      Tornati all'hangar, si compiono i controlli per assicurarsi che vengano identificati gli eventuali item che hanno subito una failure, e che tutti gli item con scadenze programmate rispettino i vincoli, ovvero abbiano una vita (programmata) residua che gli permetta sia di rispettare la condizione sulla MISL che di poter completare la prossima missione. Eventuali subsidiary damage failures vengono identificate in questa fase.

      La ricerca appena esposta procede con i seguenti due momenti:

      .a) ricerca rimozioni 'primary' e 'primary-secondary';

      .b) ricerca rimozioni 'secondary'.

    - *1.2.II)* *<u>Rimozione:</u>*

      Si rimuovono gli items identificati nella fase (I) e si mandano in officina se riparabili, altrimenti se ne ordinano di nuovi.

    - *1.2.III)* *<u>Sostituzione:</u>*

      Si controlla a magazzino se tutte le parti sono disponibili, e se sì, si procede immediatamente con la loro sostituzione (senza attendere il momento 2.1 della giornata).

  - **1.3) Rimozioni di Anticipo**

    Alla fine delle missioni del giorno, ma prima della "sera", si controlla la lista degli anticipi per eventuali rimozioni di questo tipo.

    Se il motore interessato si è già fermato oggi prima di questa fase 1.3, l'anticipo vien fatto direttamente in quella sede, comparendo fra le rimozioni della giornata (in fase 1.2). Altrimenti, la rimozione avviene ora e con essa tutti i controlli come se fosse la fase 1.2.

- **<u>Momento 2)</u> <u>LA SERA:</u>** gestione di magazzino e sostituzioni arretrate

  - **2.1) Sostituzioni arretrate**

    Si controlla il magazzino per vedere se sono arrivate le parti necessarie ai motori non operativi, (siano essi montati o meno sul velivolo), o motori ai velivoli non operativi; se sì, si procede alla loro installazione sul motore/velivolo di interesse.

  - **2.2) Aggiornamento Stato per il giorno dopo**

    Terminate tutte le operazioni del giorno, si controlla se eventuali soggetti manutentivi hanno concluso la propria operazione di riparazione o installazione, e se sì, gli si aggiorna lo Stato, che diventerà influente a partire dalla mattina successiva.

Alla fine della simulazione, vengono compiuti alcuni controlli per la correttezza dei risultati.

Ultima iterazione: simulazione senza failures

Completate tutte le iterazioni previste, prima di passare al post-processing, si fa partire un'ultima simulazione. Essa è nota come "simulazione senza failures", ovvero – come dice il nome – che riproduce il comportamento del sistema come se i suoi item non possano subire eventi unscheduled. In pratica si ottiene una simulazione senza variabili aleatorie. Una volta completata, i dati ad essa relativi vengono salvati a parte rispetto a quelli delle iterazioni canoniche, e non andrà a concorrere insieme ad esse al calcolo dei parametri statistici.
L'obiettivo di una tale simulazione è quello di avere un metro di paragone per comprendere l'influenza degli eventi randomici sul sistema, e così avere uno strumento in più per analizzare il problema.

3.1.2.3 Post-processing

La fase di post-processing si può dividere a sua volta in tre grandi parti.

- Manipolazione dei dati di simulazione:
  per estrapolarne i risultati statistici di interesse; nel caso di simulazione ottimizzata, avviene anche il computo delle variazioni (i "delta") delle grandezze fra simulazione ottimizzata e non.
  L'estrazione delle informazioni dall'insieme di iterazioni è un processo fondamentale per sintetizzare le caratteristiche dello scenario. È in questa sezione che vengono calcolati i *percentili della distribuzione di disponibilità* e la suddivisione in *livelli di priorità/probabilità* degli items nel tempo, nonché l'*Availability di down*, il *down score* e tre grandezze legate al conteggio delle *ore di volo volate*, *residue post-sostituzione* e *residue a fine simulazione*[43].

- Creazione e salvataggio dei grafici:
  per una più semplice interpretazione dei risultati da parte di un operatore umano.
  I grafici creati sono i seguenti:
  - *Eng Avbt:* mostra l'availability mensile dei motori, in termini dei percentili principali, media e simulazione senza failures;
  - *Items-NotOp:* mostra la probabilità che un certo item sia non-operativo in un dato giorno.

- Salvataggio dei dati su file di output:
  per la loro consultazione a esecuzione terminata dell'applicativo o, in caso di simulazione non ottimizzata, per essere utilizzati dal simulatore stesso per calcolare le variazioni rispetto a quella ottimizzata.
  I file prodotti sono i seguenti:
  - *Output-Data:* è il vero file di output per la consultazione da parte di un operatore umano, e contiene tutti gli output rilevanti della simulazione, catalogati in diversi fogli di lavoro a seconda del tipo di variabile;
  - *Input-Database:* viene compilato solo dalla simulazione non ottimizzata per indicare gli items da anticipare, e salvare i dati di output principali utili per il calcolo delle variazioni della simulazione ottimizzata rispetto a questa.

Si tralascia la presentazione dettagliata dei file di Input e Output, poiché non particolarmente interessante per comprendere l'analisi e i suoi risultati. Piuttosto, poiché essa presenta questi ultimi attraverso grafici, dedichiamo il prossimo paragrafo a mostrarli e spiegarli.

[43] Tutte queste grandezze sono importanti ai fini dei calcoli di ottimizzazione. Poiché non abbiamo ancora affrontato questa parte (lo faremo fra due sotto-capitoli), non entriamo in ulteriore dettaglio, per ora.

3.1.3 Grafici dei risultati di simulazione

I grafici sono efficaci strumenti per l'analisi compiuta da operatori umani. Sono stati pensati per rappresentare il comportamento del sistema cogliendone gli aspetti statistici fondamentali, essendo chiamati a sintetizzare i risultati di una simulazione compiuta su più iterazioni (dell'ordine del centinaio, talvolta migliaio).

3.1.3.1 Engine Monthly Availability (and percentiles distribution of data)

Con questo grafico si mostra l'availability (mediata sul mese calendariale) dei motori. Siccome la simulazione consta di numerose iterazioni, un grafico che mostri semplicemente la curva per ognuna di esse non sarebbe particolarmente significativo, poiché eccessivamente pieno di linee. Per rendere più leggibile, si è andati a rappresentare solo i percentili legati alla distribuzione dell'availability.

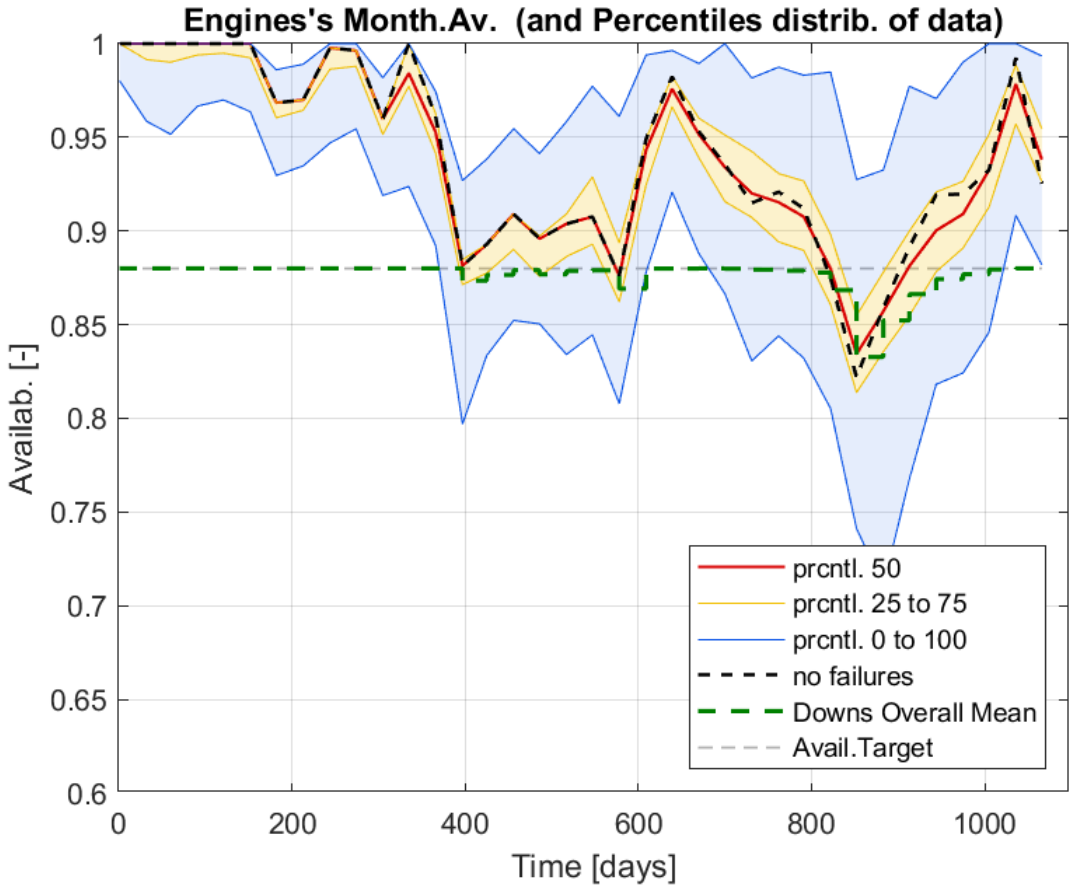


Figura 11) Grafico impiegato per la visualizzazione della disponibilità nel tempo

Come risultato si sono possono vedere diverse linee rappresentate, e il loro significato è il seguente:

- **Linee colorate continue:** rappresentano ognuna un determinato percentile della distribuzione nel tempo. Siccome questi sono simmetrici rispetto al 50°, è venuto comodo segnare visivamente l'aera compresa fra di essi, dello stesso colore delle linee che la racchiudono.
- **Linea nera tratteggiata:** andamento dell'availability della simulazione senza eventi non schedulati. Essendo sempre uguale a sé stessa (poiché esente da aleatorietà), ne basta una sola.
- **Linea grigia tratteggiata:** rappresenta l'*Availability Target*, ovvero il minimo valore di disponibilità mensile permessa dal contratto prima che si incorra in penali.
- **Linea verde tratteggiata:** è una linea che serve a visualizzare l'entità delle penali mediate fra le varie iterazioni, ed è ottenuta come spiegato di seguito. Per ogni mese, si considera la distribuzione di availability al variare delle iterazioni; si sottrae ad essa il valore dell'Availability Target,

ottenendo la cosiddetta *Availability di down*, che se minore di zero implica il pagamento di penale. Dopodiché, si vanno a escludere tutti quanti i valori positivi, così da lasciare la sola influenza delle disponibilità che genereranno penali; infine, si media su tutte le iterazioni (tutte, non solo quelle rimaste dopo la scrematura). Il risultato esteso a tutti i mesi porta a formare una successione sempre minore di zero che, se moltiplicata per il costo mensile della penale, porta ad ottenere il valore di quest'ultima sul mese. Ecco perché si è detto che serve a visualizzare l'entità delle penali. Per visualizzarla con semplicità, la si è traslata centrandola sull'Availability Target.

3.1.3.2 Items Non-Operativity Probability

Il nome 'ipscNop' significa "IPSC[44] Not Operative": il grafico serve a visualizzare, con le linee orizzontali, la probabilità (espressa in termini di frequenza di accadimento) con cui un determinato item di un certo motore (identificato dal codice *IPSC – Item Position Serial Code*) si trova, in un dato giorno di simulazione, ad essere inefficiente.
Per ottenerlo, si parte dalla singola iterazione: per ogni IPSC si memorizza, all'inizio di ogni giorno di simulazione, se il componente è inefficiente (1-true) o no (0-false). Si ripete la procedura ad ogni iterazione, ottenendo una funzione binaria 0|1 di tre variabili: *IPSC*, *giorno*, *iterazione*. Una volta concluse, nel post-processing si va a calcolare la frequenza con cui ad un certo IPSC e giorno il componente risulti inefficiente, mediando i valori 0|1 della funzione sulla variabile i*terazione*. Si ottiene una funzione a variabile reale compresa fra 0 e 1, al variare di *IPSC* e *giorno*.
Per la sua rappresentazione in un grafico 2D si è mappato con il colore il valore della funzione: rosso per valori prossimi a 1 (inefficienza molto probabile), poi a scalare arancione, giallo, … , magenta (inefficienza poco probabile) e nessun colore (inefficienza molto poco probabile, < 0.1).

Le bande verticali servono invece a rappresentare la gravità del momento di down, secondo un codice di colori analogo al caso precedente: rosso molto grave, … , magenta/rosa/nulla poco grave.

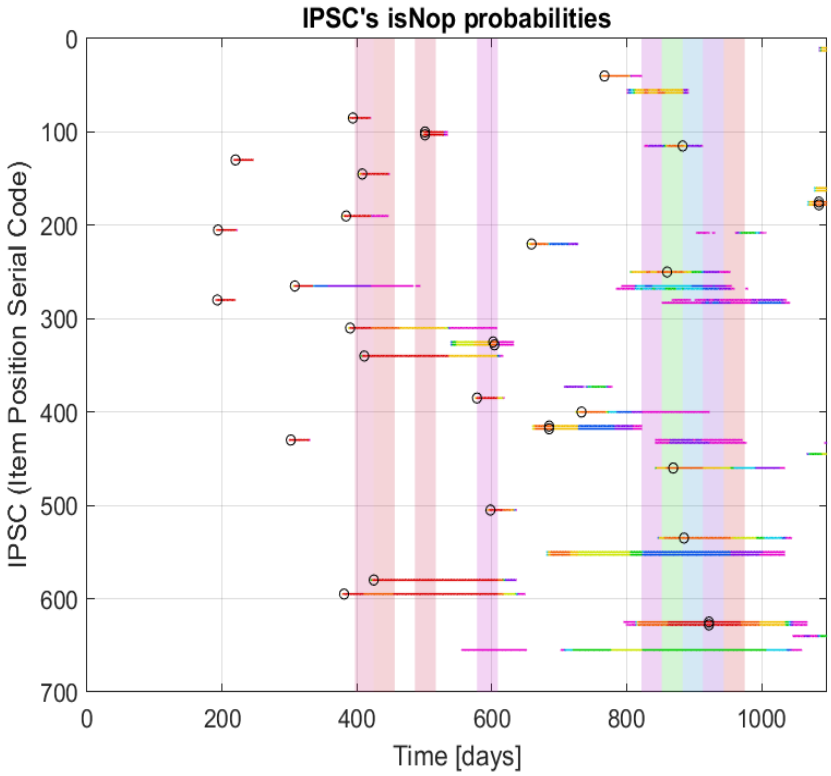


Figura 12) Grafico dell'operatività nel tempo dei componenti (simulazione non ottimizzata)

[44] IPSC – Item Position Serial Code: codice numerico con cui si identifica in maniera univoca *la posizione* di un certo item su un determinato motore, contando secondo un ordine preciso: prima si dispongono i motori in ordine di codice seriale (ESC – Engine Serial Code), poi per ognuno di essi si considera la sua configurazione di items (IC – Item type Code); si indicizzano gli item così disposti e il numero assegnatogli è proprio l'IPSC.

Si mostrano due figure: la prima relativa a una simulazione non ottimizzata, la seconda ad una ottimizzata. I cerchi neri rappresentano il momento di inefficienza più probabile per un certo item e con probabilità almeno superiore all'80%, relativo alla simulazione non ottimizzata e riportato sia nel suo grafico, che in quello della simulazione ottimizzata. In essa inoltre vengono segnati, con cerchio blu, il momento di anticipo desiderato. La distanza fra cerchi blu e neri è indicazione del numero di giorni di anticipo desiderati per quel dato item.

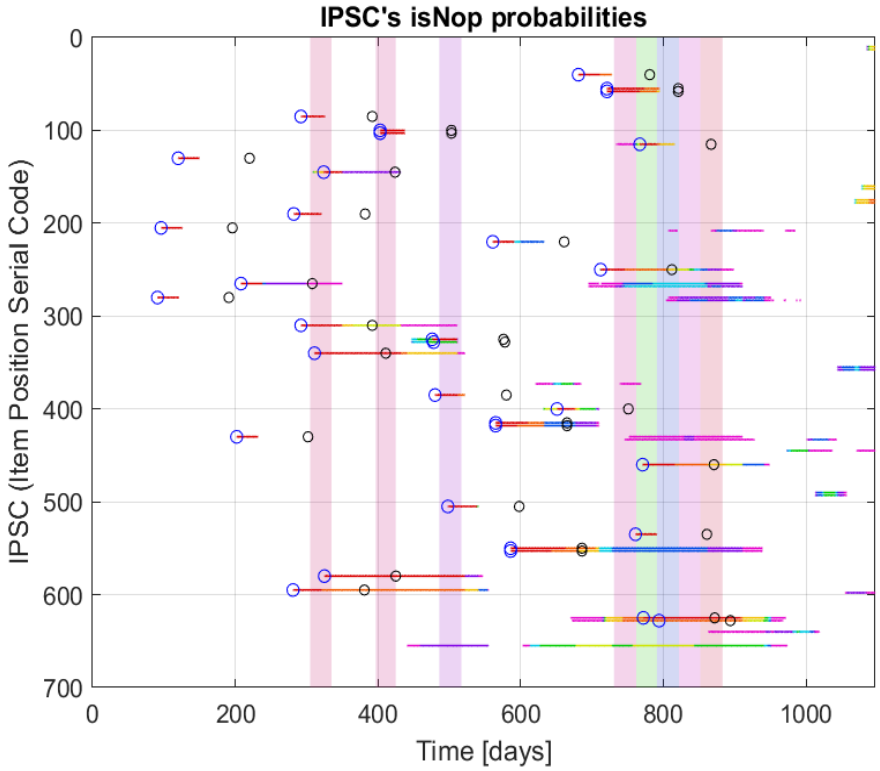


Figura 13) Grafico dell'operatività nel tempo dei componenti (simulazione ottimizzata)

In conclusione, si specifica che gli ultimi due non sono gli unici grafici generati per mostrare l'operatività degli items. Questi in particolare la mostrano catalogandoli (nel tempo) secondo un certo ordine rappresentato dall'indice/codice IPSC; altri grafici la rappresentano secondo una catalogazione differente, per numero seriale dell'item (ISC), non tanto quello della sua posizione sul velivolo; altri ancora focalizzano l'attenzione solo sugli eventi schedulati[45].

[45] Quest'ultimo è utile perché permette di visualizzare le grandezze prese in considerazione dall'ottimizzazione, che seleziona appunto gli *items anticipabili* sulla base della probabilità di non-operatività ma solo legata agli eventi schedulati.

3.2 Il Calcolatore dei costi

In questo capitolo si affrontano nel dettaglio le caratteristiche delle basi teoriche che sostengono il Calcolatore dei costi, trattando il modello su cui si fonda e le strutture dati su cui si appoggia, i metodi di assegnazione del costo durante la simulazione e le metodologie di indagine statistica per l'interpretazione stocastica dei risultati di simulazione.

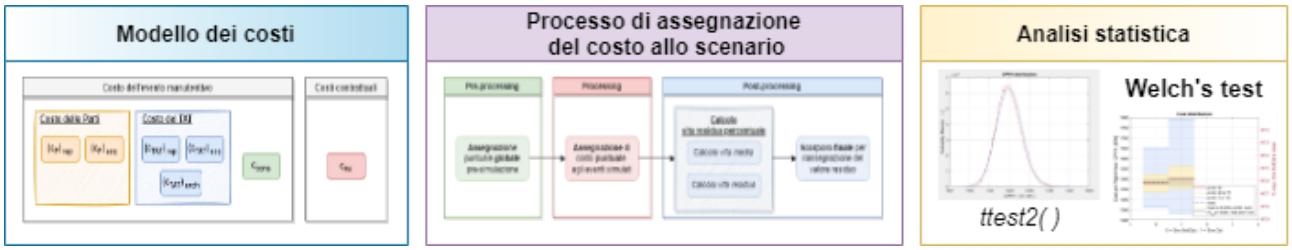


Figura 14) Schema dei componenti teorici fondamentali alla base del Cost calculator.

3.2.1 Sviluppo concettuale

Prima di iniziare, si noti che lo scopo del Calcolatore dei costi è sì quello di costificare lo scenario manutentivo in modo da ottenere un risultato in termini "assoluti" poter compiere analisi postume, dunque deve essere *accurato*, ma oltre ciò bisogna tener presente che esso deve anche supportare adeguatamente la fase di ottimizzazione di DL1. Pertanto, deve efficacemente saper cogliere le differenze di costo indotte dalle manovre di ottimizzazione (ovvero variazione di alcuni dati di input) anche sottili, e in questo senso deve dunque essere *preciso* relativamente alle suddette. Il calcolo dei costi dovrà perciò specializzarsi anche per modellare adeguatamente queste possibilità di ottimizzazione.

Le leve di ottimizzazione di DL1 a cui il Cost calculator dev'essere sensibile sono:

- la localizzata *riduzione dei TAT* manutentivi (tempi intesi come riparazione, sostituzione, testing e logistici delle parti) ;
- rischedulazione delle operazioni manutentive programmate con secondo metodo degli anticipi.

Per il raggiungimento delle caratteristiche appena esposte come obiettivo finale, il lavoro svolto è consistito nei seguenti passi fondamentali:

- sviluppo della logica di base, il *modello dei costi*, con cui costificare le operazioni di acquisto, riparazione e sostituzione delle parti, nonché di eventuali penali in caso di non rispondenza ai requisiti contrattuali definiti col cliente;
- ampliamento delle capacità di predizione della spesa, permettendo la stima dei costi associati a operazioni non comuni quali la contrazione dei TAT manutentivi;
- sviluppo del *database dei costi* associato ai principali scenari trattati;
- sviluppo di metodi di controllo della correttezza del modello simulativo (manutenzione più costi) attraverso l'utilizzo di variabili di supporto e calcolo teorico dei costi;
- infine, studio e implementazione di metodi matematici di statistica, volte all'estrapolazione di parametri di popolazione (e.g. media e varianza) e loro maneggiamento per irrobustire il processo di ottimizzazione della manutenzione.

3.2.2 Il Modello dei costi

Il Modello dei costi definisce l'insieme delle varie categorie con cui ripartire la spesa da quantificare. Identificare il modello significa dunque delineare quali eventi di spesa sono importanti da considerare, dipendentemente ovviamente anche dalle capacità simulative impiegate, e creare una struttura di dati e formule (CERs) che possa quantificare il costo ad essi associato.

L'idea alla base del Calcolatore dei costi è quella di costificare un determinato scenario manutentivo calcolando la cumulativa della spesa tramite assegnazione di un costo ad ogni evento manutentivo[46]. Siccome in linea concettuale è possibile pensare ad una manutenzione come composta di due soggetti, l'oggetto fisicamente sostituito e le operazioni di manutenzione, si è scomposto il costo secondo due voci principali: il *costo della parte* e il *costo della manutenzione* (quest'ultima valutata attraverso i TAT). Oltre questo, vengono identificati due scenari manutentivi: la *sostituzione ex-novo* della parte o la sua *riparazione.* Ad ognuna delle due voci principali è associato un costo per ognuno dei due scenari. Oltre questi, per completare il quadro delle voci associate alla manutenzione, si è andati anche a considerare la spesa associata all'utilizzo dei materiali di consumo (i cosiddetti *consumables*) per il compimento del workscope in questione, e quella che accompagna le operazioni di rimozione e installazione (di componenti/motori sul motore/velivolo). Vengono così a delinearsi sei campi con i quali descrivere la spesa associata a ogni evento manutentivo. Al loro fianco, si aggiungono infine eventuali costi di natura contrattuale, da sostenere come penale per il mancato raggiungimento di uno o più vincoli di contratto.
Di seguito si mostra uno schema che riassume le sette categorie di costo discusse.

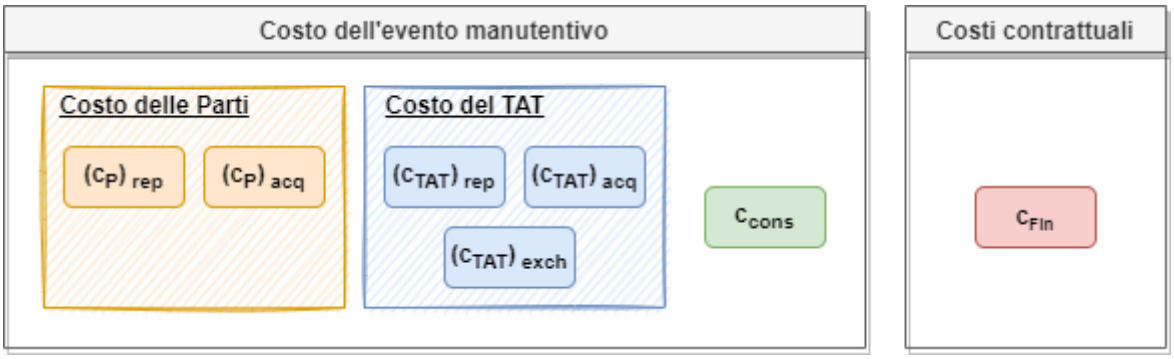


Figura 15) Schema delle voci di costo che compongono il *modello dei costi*.

3.2.2.1 Il costo della manutenzione

Mentre l'assegnazione di un costo alla parte (sia esso di riparazione o acquisto) è concettualmente semplice, poiché essa è un oggetto fisicamente esistente, lo stesso non si può dire per la costificazione della manodopera e della logistica (a cui ci si riferirà spesso in seguito, più semplicemente, come *costo del TAT*). Per risolvere il problema, si è ricorsi ad uno strumento, una formula matematica, che riuscisse a cogliere gli aspetti fondamentali della questione: da una parte, ci si aspetta che al crescere del tempo di manutenzione cresca anche la spesa associata; dall'altra, come spiegato nella sezione sugli obiettivi del Calcolatore dei costi inerentemente alle necessità legate all'ottimizzazione, vi è l'esigenza di calcolare l'effetto della riduzione forzata dei tempi manutentivi, attendendosi che ad una diminuzione dei TAT corrisponda un aumento della spesa, poiché si richiede di compiere una medesima serie di attività in un tempo minore.

Seguendo questi punti chiave, è stata creata la CER seguente:

---

[46] Con *evento manutentivo* si intende uno fra i seguenti, la cui replicabilità è ammessa dal simulatore manutentivo: riparazione o acquisto di una parte o modulo, rimozione o installazione di un modulo (o motore) sul motore (o velivolo). Per la precisione, nel contesto di nomenclatura del simulatore manutentivo, un evento manutentivo può anche possedere specifiche extra, come il motivo che ha causato la rimozione; a tal riguardo si trovano: inefficienza causata da eventi non pianificati (failure o subsidiary damage) o schedulati (revisione per *TBO – Time Between Overhaul* o fine vita di *LLP – Life Limited Part*).
Per una trattazione completa, si rimanda ai deliverable D1.2.1 e D1.4.1.

$$c_{\mathrm{TAT}} = k_{\mathrm{base}}\, TAT_{\mathrm{base}}\, {p_{\mathrm{TAT}}}^{\alpha}$$

dove:

Grandezza	Unità di misura	Spiegazione
c_{TAT}	[cu[47]]	costo della manodopera associata a un certo intervento ("rep"= riparazione; oppure "exch"= sostituzione)
k_{base}	[cu/days]	coefficiente di costo (medio) per giorno di fermo
TAT_{base}	[days]	tempo base per compiere un certo intervento manutentivo, ovvero quello che si avrebbe in assenza di riduzione forzata
TAT	[days]	tempo effettivo per compiere un certo intervento manutentivo, sia esso con o senza riduzione forzata
$p_{TAT} = \frac{TAT_{\mathrm{base}}}{TAT}$	[-]	coefficiente di riduzione del TAT (maggiore di uno se presente riduzione forzata)
α	[-]	esponente di penalizzazione dell'intervento manutentivo (gestisce l'aggravo di spesa associata a una medesima riduzione percentuale del TAT)

Per la precisione, sono state sviluppate formule differenti a seconda dell'applicazione, ma tutte si rifanno come forma a quella appena vista. In particolare, a seconda del tipo di intervento manutentivo, ci si può trovare davanti a una riparazione o una sostituzione (nel senso di rimozione, installazione e testing). Le si approfondiscano di seguito.

- **Riparazione/Acquisto (TAT_rep)**
  Rappresenta il tempo necessario alla riparazione di un item riparabile (dunque, un motore o un modulo), oppure il tempo di approvvigionamento di una parte a vita limitata o un accessorio.
  - TATrep) *Moduli & Motore:*
    Per i TAT_rep di moduli e motore, poiché rappresentano effettivamente dei tempi di riparazione, la formula è esattamente quella mostrata prima.
  - TATacq) *Accessori & LLP:*
    Poiché gli accessori e le LLP a fine vita non vengono riparati ma vengono comprati ex-novo, in teoria non esiste nessun "lavoro di manutenzione" su di essi. Se si volesse diminuire il costo di acquisizione bisognerebbe pagare semplicemente di più il componente. Dunque il costo del TAT è in questo caso inteso come costo di surplus dell'item. Deve essere zero (surplus nullo) quando $p_{TAT} = 1$. Pertanto si ottiene una formula:
    $$c_{\mathrm{TAT}} = c_{\mathrm{New}}\, ({p_{\mathrm{TAT}}}^{\alpha} - 1)$$
    dove c_{New} è il costo base della parte.

- **Sostituzione (TAT_exch)**
  La formula è la stessa dell'inizio, con l'accortezza che c'è la possibilità per gli accessori di avere nella simulazione un "TAT_exch = 0": questo farebbe esplodere il calcolo. Pertanto si considera, per sanare la questione, che il TAT_exch (del calcolo dei costi) non può essere minore di "0.5". Insomma si sostituisce il TAT_exch di simulazione con quello dei costi nella seguente maniera: $TAT_{\mathrm{cost}} = \max\{\, TAT_{\mathrm{sim}}\, , 0.5\,\}$.

[47] *cu*: "cost unit", generica unità di costo (e.g. €, $, k€ ecc.)

3.2.2.2 Introduzione delle penali

A livello di modello dei costi, un passo fondamentale è ancora da compiere per chiudere il quadro delle voci di spesa indispensabili per una corretta analisi, specie se si focalizza l'attenzione sul gestore diretto della manutenzione dei motori: tale mancanza è l'introduzione delle quote di costo associate alle penali da pagare al gestore della flotta di velivoli, cliente del primo, nel caso in cui non vengano rispettate le condizioni contrattuali sulla disponibilità-motori minima da assicurare in un determinato lasso temporale (generalmente, ragionando per mensilità).

A livello pratico:

- anzitutto, si considera l'availability dei motori (nel caso implementato, quella mensile calendariale);
- se essa scende sotto al valore definito nel contratto (in genere intorno all' 80-90%) si calcola di quanto è stata questa violazione;
- infine, secondo una qualche funzione, gli si associa un costo, tenendo conto che per legge non può essere superiore al 10% del costo della manutenzione nel medesimo periodo considerato.

Nel caso implementato, la funzione in questione è di proporzionalità diretta fra entità del *down* di availability (A_{down}) e costo della penale associata (c_{Fin}). La legge matematica è perciò semplicemente:

$$c_{\mathrm{Fin}} = k_{\mathrm{Fin}}\, A_{\mathrm{down}}$$

A livello parametrico serve dunque un solo coefficiente che regola questa dipendenza, ovvero il coefficiente di proporzionalità fra i due, k_{Fin} , espresso in $\left[\frac{\mathrm{cu}}{\%}\right]$.

3.2.2.3 Il problema delle ore di volo

L'introduzione delle penali permette di abbandonare un'ottimizzazione basata sul doppio obiettivo availability-costo, passando ad una più semplice ottimizzazione mono-obiettivo: il costo totale, giacché esso, grazie al contributo delle sanzioni, ingloba già le informazioni relative alla diminuzione non tollerabile di availability. Questo comporta uno sgravo del problema, ammettendo una soluzione in forma forte.
Tuttavia, un'ottimizzazione basata direttamente sul costo totale assoluto non è efficace, poiché esso dipende dal quantitativo di ore volate: l'Ottimizzatore tende a valorizzare scenari a basso quantitativo di ore volate, ad esempio accumulando tutte le manutenzioni in un medesimo periodo e generando code che diminuiscono sensibilmente l'availability media. Questo effetto è parzialmente compensato dall'effetto delle penali, che però sono soggette a una saturazione (imposta per legge al 10% del costo della manutenzione): l'effetto complessivo trova comunque un equilibrio spostato verso l'abbassamento dell'availability, portando a dover scartare i risultati di ottimizzazione.

La soluzione

Per sanare la problematica, è necessario reintrodurre l'informazione (diretta o indiretta) sulle ore di volo. Le alternative sono le seguenti due, principalmente.

- Soluzione diretta (scartata)
  La possibilità più immediata è una di tipo diretto: introdurre, nell'ottimizzazione, un nuovo obiettivo di massimizzazione delle ore di volo, o quantomeno un vincolo affinché la simulazione ottimizzata abbia tale valore almeno superiore a quella non ottimizzata. Così facendo, però, si ricadrebbe nella problematica di avere due obiettivi separati, per cui questa soluzione va scartata, poiché va contro i propositi iniziali di avere un problema mono-obiettivo.

- Soluzione indiretta (perseguita)
  Il metodo alternativo consiste nel reintrodurre l'informazione sulle ore di volo in maniera indiretta, contenendola all'interno dei costi, giovandone così com'era stato per le penali. Due possibilità possono essere considerate.
  - *Il **costo netto**:*
    Si sostituisce nell'ottimizzazione il costo totale (lordo) con un *costo totale netto*, valutato partendo dal primo e aggiungendovi (/scorporandovi) una quota di spesa proporzionale alle ore di volo perse (/guadagnate), agendo in un modo analogo a quanto compiuto nel caso di riassegnazione del valore residuo. Come lato positivo, questa grandezza risultata di immediata comprensione, essendo una quantità assoluta come il costo totale; di contro, per il suo computo fa affidamento sul processo di riassegnazione che, per quanto preciso, introduce sempre una naturale incertezza per via delle ipotesi su cui è costruito.
  - *Il **costo per ora volata**:*
    L'alternativa è di calcolare un *costo per ora-velivolo di volo*, ottenuto dividendo il costo totale lordo per il numero totale di ore-velivolo[48] volate. Sebbene meno intuitivo del primo, alla fine si opta per questa grandezza come nuova variabile di ottimizzazione poiché nasce da una divisione esatta, senza introduzione di ipotesi di calcolo di alcun tipo, e dunque più affidabile.

In ultimo, si potrebbe argomentare che questo nuovo metodo di ottimizzazione potrebbe ancora essere suscettibile alla seguente problematica: poiché l'ottimizzatore che lo implementa ragionerebbe con il concetto di ora di volo generica, cumulabili da un qualsiasi velivolo nella flotta indipendentemente dalla sua tipologia, esso potrebbe tendere a ricercare soluzioni che facciano volare maggiormente velivoli con item meno costosi, e viceversa, non curandosi di eventuali esigenze ulteriori. Non ci si preoccupa della questione, poiché le flotte di velivoli prese ad esame in questo studio sono tipicamente composte da una sola tipologia di velivolo (o comunque con caratteristiche molto simili), e dunque di items che monta, senza significative differenze a livello di costo di componenti. In ogni caso, ci sarebbero dei metodi per sanare la problematica anche se quest'ipotesi cadesse, ma poiché reputata valida non li si è approfonditi ulteriormente.

3.2.3 Il processo di assegnazione dei costi

Con "processo di assegnazione dei costi allo scenario manutentivo" si intende tutto quell'insieme di passaggi che permettono il computo della spesa dello scenario considerato, partendo dal modello dei costi che ne specifica la catalogazione in base al tipo di evento accaduto (questi ultimi a loro volta dipendenti dalle capacità del simulatore).

La procedura di assegnazione del costo durante la simulazione è conseguenza diretta della scelta di associare a ogni intervento manutentivo una determinata spesa. Nella sua forma più semplice, si parte infatti da inizio simulazione con un costo totale nullo; progredendo con essa, si calcolano di volta in volta le spese associate ad ogni evento manutentivo che si verifica (riparazioni, acquisti, sostituzioni), andando a sommare il nuovo costo al totale accumulato fino a quel momento. Alla fine della simulazione, si disporrà di un costo totale dello scenario, suddiviso secondo una certa logica desiderata (e.g. i sette campi di costo, gli items ed eventualmente conservando anche l'andamento temporale). Il processo così descritto rappresenta il cuore caratteristico del calcolo (in Figura 16, la sezione di Processing, in rosso), ma da solo non è sufficiente per poter dare una stima affidabile nei termini di accuratezza e precisione discussi all'inizio di questo capitolo. Si vadano dunque ad approfondire, nei seguenti paragrafi, gli altri due momenti fondamentali per un corretto calcolo.

[48] È importante specificare che il calcolo viene compiuto sulle *ore-velivolo*, perché si possono considerare anche altre grandezze quali le *ore-motore* o le *ore-item*, in questo caso meno significative per lo scopo in questione.

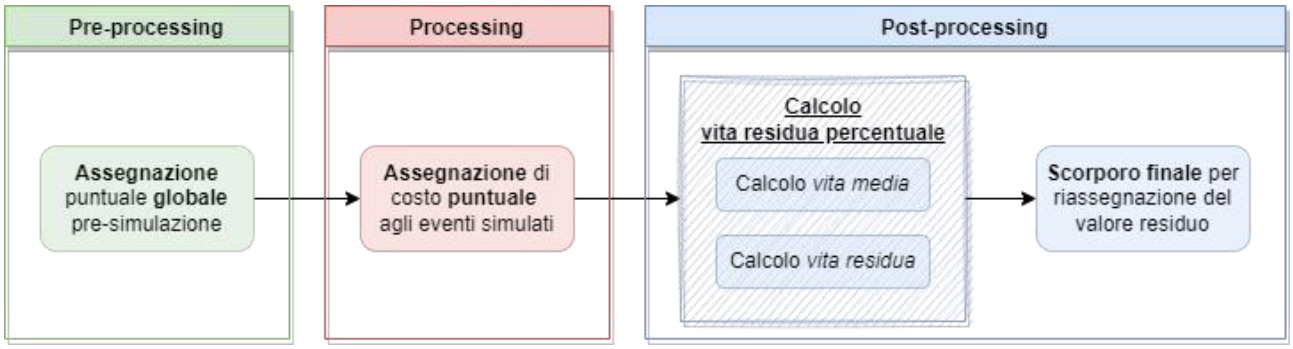


Figura 16) Schema del processo di assegnazione dei costi allo scenario, espresso nella sua versione finale.

3.2.3.1 Il calcolo assoluto dei costi

Una delle limitazioni maggiori del processo di calcolo dei costi appena descritto, è che esso non tiene in considerazione la spesa associata ai componenti già montati sui motori all'inizio della simulazione. Ciò implica che, alla fine di essa, il costo totale ottenuto sia quello di utilizzo solo dei componenti sostituiti ai primi: il valore numerico non è ancora il costo "assoluto" dello scenario.
Un costo assoluto non solo aiuta nella comprensione generale dello scenario, ma è utile anche per ribaltare queste nozioni verso il database dei costi stesso, permettendo di renderlo più realistico[49]. Inoltre, come visto nel paragrafo 3.2.2.2, la sua importanza è cruciale per il corretto dimensionamento delle penali, che vengono assegnate in relazione al costo totale di gestione della flotta di motori in un certo periodo.
L'implementazione scelta consiste nel calcolare il costo dei primi items in maniera analoga agli altri: dopo una prima assegnazione puntuale prima dell'inizio della simulazione, alla fine di essa si opera il ricalcolo del valore residuo, considerando non solo la porzione di vita effettivamente residua dell'item considerato, ma restituendo anche quella di inizio simulazione, poiché associata a un periodo precedente a quello simulato (vedasi prossimo paragrafo).

3.2.3.2 Il ricalcolo del valore residuo dei componenti

Affinché il modello dei costi sia sufficientemente sensibile da poter cogliere anche le sottili differenze di costo indotte da pochi giorni di anticipo, esso dev'essere in grado di distinguere due scenari caratterizzati dai medesimi eventi ma differenti per il momento in cui essi sono avvenuti. Infatti, gli effetti di un anticipo non sono di cambiare sensibilmente il numero di manutenzioni in un determinato tempo di osservazione, ma appunto il momento in cui esse avvengono: ciò che cambia a fine simulazione nei due casi è l'età del componente, maggiore nel caso anticipato che nell'altro.
Per sanare la problematica, si è può operare un ricalcolo del valore residuo dei componenti a fine simulazione, scorporandola dal totale: tale residuo è la manifestazione, a livello di costi, dell'aspettativa media di vita residua del componente. Il metodo dunque consiste nella stima, a fine simulazione, della vita rimanente di ogni singolo componente come percentuale rispetto all'aspettativa da nuovo. A livello di costi, si scorpora dal totale la medesima percentuale, stavolta associata al costo delle operazioni manutentive legate all'item preso in considerazione. Questo procedimento è sensibile anche a variazioni dell'ordine dei giorni nel computo totale del costo.

[49] Infatti, da fonti bibliografiche è nota l'entità del costo di operazione di un velivolo come quelli impiegati negli scenari considerati: il vincolo principale, dunque, è far sì che tale costo totale atteso, fortemente legato ai valori inseriti nel database, sia congruente con quello calcolato a livello di simulazione.

Primo approccio

Consiste nel trascurare le effettive distribuzioni di probabilità di guasto dei failure mode, assimilandole tutte quante a delle distribuzioni puramente randomiche (dunque, ad andamento esponenziale), con MTBF pari per ognuna al proprio vero valore di MTBF del failure mode considerato. In questo modo è stato possibile ricavare con semplicità l'MTBR globale del componente a partire da quelli degli eventi che esso possiede, applicando la legge:

$$MTBR_{\text{tot}} = \left[\sum_{j=1}^{N_e} \frac{1}{MTBR_j}\right]^{-1}$$

dove:
- j conta il j-esimo evento del componente selezionato; N_e è il numero totale di eventi;
- $MTBR_{\text{tot}}$ è relativo all'intero componente;
- $MTBR_j$ è relativo al j-esimo evento del componente selezionato, e viene calcolato a partire dalla distribuzione di Weibull che lo rappresenta (attraverso i suoi parametri di scala η e di forma β) secondo la formula $MTBO_j = \eta_j\, \Gamma\left(1 + 1/\beta_j\right)$, con $\Gamma(x)$ che è la *funzione gamma*.

Tale metodo è immediato e risulta esatto per gli accessori, poiché essi hanno solo failure mode puramente randomici ($\beta = 1$); tuttavia, risulta meno robusto per i moduli con valori di parametro di forma non prossimi all'unità e soprattutto in caso di presenza di una o più scadenze programmate (per le quali ricordiamo che $\beta \sim \infty$), siano esse per TBO o per LLP.
Per risolvere la problematica si è deciso di calcolare (sotto ipotesi meno rigide) in maniera esatta la vita media del componente a partire dalle curve Weibull dei suoi failure mode. Se ne esponga il metodo di seguito.

Calcolo migliorato

Fatte le seguenti considerazioni:
- sotto l'ipotesi di massimo una sola scadenza programmata e
- ipotizzando che il suo counter si resetti ad ogni revisione (vero nel caso di tutti i failure mode e dell'evento di revisione TBO, ma non per forza per una LLP),

applicando le formule della matematica probabilistica, si ottiene la *vita media* dell'item come:

$$MTBM_{\text{tot}} = \int_0^{LOF} R_{\text{tot}}(t)\ dt$$

Si portano le seguenti note:
- la vita media è ottenuta come integrale della *funzione globale di affidabilità* $R_{tot}(t)$,
- a sua volta ottenuta come produttoria delle *reliability function* dei vari failure mode del componente interessato $\left(R_{\text{tot}}(t) = \prod_{j=1}^{N_f} R_j(t) \right)$,
- ognuna delle quali è il complementare a uno della *cumulative density function* associata al failure mode in questione $\left(R_j(t) = 1 - CDF_j(t) \right)$,
- quest'ultima ottenibile direttamente dai parametri di scala e forma della curva Weibull

$$\left(CDF_j(t) = CDF_{\text{weibull}}\left(t\,;\, \eta_j, \beta_j \right) = 1 - e^{-(t/\eta_j)^{\beta_j}} \right)$$

- In ultimo, l'integrale viene calcolato fra *0* e il *LOF* (*Limite in Ore di Funzionamento*) della (unica per ipotesi) scadenza programmata del componente, e che vale ∞ se esso non ne possiede.

Si noti che il nuovo calcolo ora risulta esatto per i componenti senza LLP (solo failure mode e un evento di Overhaul), e se presenti sarebbe complicato calcolare la vita media generale, che dipenderebbe anche da altri fattori quali la MISL. Infine, si consideri che, in questo metodo, vengono trascurati gli eventi di subsidiary damage, che introdurrebbero altrimenti una complessa interdipendenza fra items.

3.3 L'Ottimizzatore

L'Ottimizzatore è, al suo cuore, un algoritmo iterativo che impone i valori di un certo set di variabili (numeriche, in tal caso) in funzione dell'andamento dello storico degli obiettivi di ottimizzazione e del set stesso.
Da soli questi tre elementi non sono però sufficienti per ottenere delle analisi significative, poiché gli algoritmi disponibili per le analisi sono di tipo deterministico, ossia non considerano incertezze negli obiettivi di ottimizzazione. A causa delle failures, per loro natura aleatoria, gli scenari simulati saranno soggetti a dell'incertezza che, in qualche modo, va maneggiata. Pertanto, al fianco dei tre elementi base dell'Ottimizzatore, è necessario porre degli altri *Elementi di supporto*.

Nel prossimo paragrafo si vadano a discutere brevemente gli elementi dell'Ottimizzazione, dedicando spazio ai paragrafi successivi per l'approfondimento degli elementi di supporto.

3.3.1 Gli elementi dell'ottimizzazione

Algoritmo di ottimizzazione

Le logiche specifiche dipendono dall'algoritmo scelto, ed essi possono essere di diverse tipologie e anche molto complicati. Ne abbiamo esposto i principali nel paragrafo dedicato alle Basi di ottimizzazione, per cui non li riaffronteremo. Presentiamo invece gli altri due elementi costitutivi dell'Ottimizzatore: leve/variabili di ottimizzazione e Obiettivi di ottimizzazione, così come definiti per le analisi di questa tesi (metodo degli anticipi).

Leve/variabili di ottimizzazione

L'ottimizzazione lavora sulla schedulazione degli eventi di manutenzione programmata, in particolare scegliendo di quanti giorni anticipare un certo evento schedulato (sostituzione o revisione).
Non tutti gli eventi sono però ugualmente impattanti sullo scenario, per cui anzitutto si analizza lo scenario "di base", ossia quello non ottimizzato, da cui vengono selezionati solo quelli (schedulati) con la più alta probabilità di accadimento. Si identificano così un certo numero di *items di anticipo*, e il *numero di giorni d'anticipo* associati ad ognuno di essi compongono il set di variabili di ottimizzazione.

Obiettivi di ottimizzazione

Come obiettivo di ottimizzazione è stato scelto il *cost per flight hour (CPFH)*, definito come il costo totale (medio) della manutenzione per ora-velivolo volata. Tale grandezza viene fornita, per ogni optimization design indagato, dallo Scenario Evaluator: esso vi estrae informazioni attraverso una metodologia basata su simulazione Monte Carlo e, dopo adeguato post-processing dei dati delle singole iterazioni, fornisce all'Ottimizzatore i dati in termini di distribuzione di CPFH.

Elementi di supporto all'ottimizzazione

Nei sotto-capitoli dedicati a Simulatore Manutentivo e Calcolatore dei costi sono stati visti i metodi con cui le variabili di input/output all'Ottimizzatore vengono ottenute/impiegate. Per le questioni sull'incertezza degli scenari, i dati di input all'Ottimizzatore non vengono però impiegati in maniera diretta. Ci sono due processi fondamentali ancora da discutere che riguardano la *Gestione dell'incertezza*, e possiamo riassumerli nei seguenti due punti:

- *Analisi statistica sugli Output di Simulazione:* spiega il problema statistico e la sua soluzione ;
- *Scelta del numero di iterazioni:* questione accessoria alla soluzione individuata al punto prima.

Ad ognuno di essi vi si dedicato un paragrafo di approfondimento, subito di seguito.

3.3.2 Analisi statistica

Dopo aver affrontato il modello dei costi e come applicarlo a uno scenario manutentivo, si è finalmente in grado di ottenere da esso un risultato di costo. In ottica di singola iterazione, ciò è sufficiente nel senso che è possibile estrarre da essa tutte le informazioni di cui si necessita. Tuttavia, a causa dell'aleatorietà che caratterizza il sistema, lo studio limitato a un singolo caso non è abbastanza per poter inferire riguardo proprietà medie dello stesso. Pertanto, ancora uno studio dev'essere compiuto per chiudere il quadro della comprensione, a livello di costi, del sistema presentato, e cioè un'analisi che permetta di sintetizzare le informazioni provenienti da più iterazioni per poter gestire la naturale incertezza dei risultati.
Nel seguito, si va dapprima ad affrontare approfonditamente il problema, spiegandone la genesi e le implicazioni, e successivamente esponendo le soluzioni ricercate e implementate a tal proposito.

3.3.2.1 Il problema

La necessità dell'analisi

È possibile che, nel processo di ottimizzazione, a seconda del tipo di scenario considerato i margini di costo su cui operare siano sottili: è dunque necessario sviluppare metodi che permettano di affinare l'ottimizzazione dei costi, rendendola più precisa.

Il problema principale

A rendere complicato il raggiungimento di un tale obiettivo, il maggiore antagonista è l'oscillazione dei parametri di ottimizzazione (in tal caso, di costo), indotti dall'incertezza intrinseca nel sistema. Infatti, l'Ottimizzatore compie le proprie valutazioni sulla base di tali dati come obiettivo da migliorare, assumendoli deterministici[50]. Dunque, esso risente negativamente della randomicità poiché rende complicato distinguere se un certo scenario favorevole sia dovuto a un effettivo miglioramento associato agli input, oppure semplicemente al caso. Per far fronte a questa problematica, è necessario dover sviluppare degli adeguati metodi di indagine statistica.

Comprensione del caso specifico

La logica di ottimizzazione, per come è stata impostata, si basa sull'ottimizzazione della media aritmetica della variabile di ottimizzazione calcolata dalle varie iterazioni, in questo caso il *cost per flight hour* (*cpfh*). L'obiettivo è dunque quello di saper discernere se un certo *cpfh medio* calcolato in una simulazione ottimizzata sia o meno favorevole (ossia minore) rispetto a quello della simulazione ottimizzata: se lo è, si considerano positivi gli input che hanno generato lo scenario, altrimenti viceversa e si ricerca verso input differenti.
Quanto affrontato è una questione classica dell'analisi statistica, appartenente all'ampia gamma dei "problemi di testing di ipotesi nulla sulla media di (una o più) popolazioni". Per affrontarli, è stata tradizionalmente sviluppata una famiglia di soluzioni, applicate tutte al medesimo problema ma sotto ipotesi più o meno stringenti riguardanti le distribuzioni (ignote) che generano i campioni osservati e il tipo e la numerosità dei dati a disposizione, ottenendo metodi più o meno efficaci a seconda dello scenario considerato. Il primo passo è riconoscere le caratteristiche del caso trattato.
Quanto in esame si configura, nella sua forma generale, come:

> " l'asserimento, sotto specificati valori di precisione, della diseguaglianza della media di popolazione di due variabili randomiche, valutate attraverso due insiemi di realizzazioni di uguale o differente numero, estratti da popolazioni distribuite quasi-normalmente e in generale con diversa (anche se simile) varianza. "

[50] Non è stato infatti possibile trovare uno strumento con integrate funzioni per il trattamento di problemi di ottimizzazione stocastica, dove cioè i parametri presi come obiettivo sono a loro volta sottoposti ad incertezza, come in questo caso.
Il software impiegato (modeFRONTIER) è infatti al più capace di valutare la robustezza del sistema valutando gli output a partire da distribuzioni stocastiche di input, ma non il contrario, potenzialità necessitata in questo caso.

3.3.2.2 La soluzione

Ricerca della soluzione

Tutti gli studi indagati per la sua soluzione hanno il denominatore comune che, come è anche atteso intuitivamente, la precisione aumenta con la grandezza dei campioni di cui si dispone, in altre parole con il numero di iterazioni compiute; inoltre, le soluzioni più robuste a livello statistico sono quelle che impiegano due campioni di egual grandezza. In questi termini la soluzione ideale in termini di sicurezza sarebbe avere due simulazioni (quella ottimizzata e quella non) con uguale e alto numero di iterazioni. Tuttavia, per questioni legate al tempo di calcolo è possibile avere alti valori di numero di iterazioni solo per la simulazione non ottimizzata, poiché viene compiuta una sola volta in tutta l'analisi di Maintenance Plan; viceversa, per quella ottimizzata non è possibile aumentarne eccessivamente il numero, essendo ripetuta più volte (nell'ordine delle centinaia di cicli) e dunque estremamente più incisiva. Ne deriva che bisogni operare una scelta fra le due condizioni ideali.

Soluzione trovata

Risulti conveniente rinunciare all'uguaglianza dei campioni, e in letteratura la soluzione a questo problema è portata attraverso un metodo noto come *t-test di Welch*: si tratta di un adattamento del *t-test di Student*, ma più affidabile quando i due campioni sono estratti da popolazioni con varianze diverse e hanno numerosità (uguale o) diversa. Inoltre, entrambi mantengono -nella loro forma esatta- l'ipotesi di distribuzione normale, anche se il t-test di Welch è comunque piuttosto robusto alla presenza di asimmetria in essa. Per tutte queste ragioni, è stato scelto come soluzione.

A livello di **implementazione**, è stata utilizzata una funzione, integrata nella libreria di default di Matlab, chiamata *ttest2*. Essa realizza un t-test a doppio campione (di grandezza differente) permettendo la scelta di parametri accessori quali:

- la *significatività*[51] (*significance level*) del test;
- il *tipo di ipotesi alternativa* desiderata, con scelta fra '*both*' (test contro l'ipotesi alternativa che le due popolazioni abbiano media diversa), '*right*' (test contro l'ipotesi alternativa che la prima popolazione abbia media maggiore della seconda) o '*left*' (che la prima sia minore della seconda);
- il *tipo di varianza* delle due popolazioni, se assunta uguale o diversa (ma comunque ignota).

Come output, restituisce l'esito del test, il suo *p-value*[52] e l'*intervallo di confidenza*[53] per la *differenza delle medie di popolazione*[54] associata al livello di confidenza scelto in input.

Comunque, prima di applicare i metodi appena esposti, è necessario accertarsi che le ipotesi su cui essi si fondano siano rispettate. In particolare, si deve valutare se la condizione di normalità della distribuzione di popolazione sia sufficientemente rispettata. Prima di iniziare l'ottimizzazione sarà dunque necessario compiere uno studio preliminari per accertarsi di ciò.

---

[51] La *significatività* è definita come la probabilità, per uno studio di testing dell'ipotesi nulla, che quest'ultimo rigetti l'ipotesi nulla nel caso che essa sia però vera. Essa rappresenta dunque il rateo di *falsi positivi* (anche detti *errori di prima specie*) e viene indicata convenzionalmente con α.
Inoltre, spesso si fa riferimento al suo complemento a uno, ovvero il *confidence level* $\gamma = 1 - \alpha$, che rappresenta la probabilità di non rigettare l'ipotesi nulla quando questa è vera (dunque, si tratta del rateo di *veri negativi*).

[52] Il *p-value* di un test di significatività è la probabilità di ottenimento di un risultato estremo almeno quanto quello osservato, data per vera l'ipotesi nulla.
Il risultato è detto *statisticamente significativo* per gli standard dello studio se $p \leq \alpha$, nel qual caso si dice che l'ipotesi nulla (e.g. che due medie misurate siano statisticamente uguali) viene rigettata e assunta vera l'ipotesi alternativa (e.g. che le due medie siano effettivamente diverse).

[53] L'i*ntervallo di confidenza* rappresenta il range, associato alla variabile (ignota) di cui si vuole stimare il valore, tale per cui si riscontra che il reale valore ricade proprio in tale regione con frequenza pari al *confidence level* con cui l'intervallo è stato calcolato.

[54] In questo specifico caso, qualora l'intervallo di confidenza contenga lo zero allora l'ipotesi non viene rigettata, e viceversa se l'intervallo non lo contiene.

3.3.2.3 Applicazione dei risultati

Gli studi appena esposti hanno permesso di ricavare una soluzione al quesito iniziale:

> “ disponendo di due medie di *cost per flight hour*, ognuna calcolata da un certo campione di iterazioni (uno dalla simulazione ottimizzata e l’altro dalla non ottimizzata), queste medie sono “sufficientemente differenti in valore” da poter imputare questa differenza agli input piuttosto che al semplice caso? ”

La soluzione trovata consiste nell’implementare un approccio statistico detto t-test di Welch: inseriti tutti i dati (ovvero i due campioni) e definiti i vari parametri, il metodo restituisce in output il responso cercato, ossia se i risultati siano “statisticamente significativi” (rispetto al metodo scelto) o no.
Dal test sono possibili due esiti. Rimane da mostrare come muoversi in entrambi i casi. Si esponga di seguito tale procedura, applicandola agli usi che se ne fanno nelle analisi di questa tesi.

- Nel caso la differenza delle medie risulti statisticamente significativa (quindi l’ipotesi nulla di uguaglianza delle medie venga rigettata), i risultati della simulazione ottimizzata vengono passati all’ottimizzatore, che li prende in considerazione per compiere le successive scelte per migliorare lo scenario.

- Viceversa, se il test non rigettasse l’ipotesi nulla (ovvero considerasse i risultati non statisticamente significativi), vorrebbe dire che il rumore nei dati è troppo intenso per poter evincere una precisa tendenza degli input selezionati a migliorare o peggiorare la situazione: da ciò ne deriva l’impossibilità per l’ottimizzatore di impiegare le informazioni in questione per reindirizzare efficacemente la ricerca di soluzioni migliorative. In questo caso, pertanto, si ignorano i risultati ottenuti, poiché di scarsa significatività.
  Riguardo il come agire in seguito all’accadimento di un tale scenario, ci sono diverse opzioni. Fra le principali troviamo le seguenti:
  - o si arresta definitivamente il processo di simulazione, dichiarando la combinazione input come “impossibile da valutare” e ricercando verso altre di esse, ignorando dunque del tutto i risultati di simulazione appena ottenuti;
  - oppure si continua a simulare, arricchendo il campione ottimizzato di realizzazioni, il che permetterà di aumentare la precisione con cui viene calcolata la media estratta da esse e perciò la tendenza del metodo a restituire un risultato significativo, e dunque impiegabile.

Per quanto visto finora, si comprende che la soluzione ideale per risolvere il problema dell’incertezza sia, semplicemente, disporre di un alto numero di iterazioni. Tuttavia questa soluzione porta con sé i relativi aggravi computazionali, e dunque sebbene è la migliore soluzione “ideale”, non lo è nella pratica.
Nel prossimo paragrafo si discute più nel dettaglio questa problematica e le soluzione trovata.

3.3.3 Scelta dinamica del numero di iterazioni

Per via della natura fortemente aleatoria del sistema studiato, e delle alte precisioni richieste dai calcoli, si necessita un significativo numero di iterazioni Monte Carlo affinché il singolo scenario venga inquadrato nei giusti modi. Sommato a ciò, tanti design di ottimizzazione vanno indagati per via del grande spazio di progetto su cui si spazia. Ciò comporta però una grossa quantità di tempo di calcolo per produrre i propri risultati: si richiede dunque un qualche metodo per limitarlo.

Con i vari tentativi di ottimizzazione compiuti durante gli studi in corso d'opera, una delle peculiarità osservate con certezza è stata che l'Ottimizzatore all'inizio delle proprie analisi, quando genera randomicamente le prime combinazioni di input esplorando lo spazio di progetto, spesso produce scenari chiaramente peggiori di quello non ottimizzato. Inoltre si consideri che, per quanto influente possa essere l'oscillazione del costo, più uno scenario ottimizzato è positivo (o negativo) in termini di media di popolazione, tanto più è probabile che servano meno iterazioni per accettare l'ipotesi alternativa[55], e con essa i risultati di simulazione.
Sintetizzando insieme questi due concetti appena esposti, viene naturale concluderne che all'inizio del processo ottimizzativo servano meno iterazioni, per ogni simulazione, rispetto alla sua fine. Pertanto se si potesse scegliere, secondo un qualche criterio, il numero di iterazioni da compiere di volta in volta, sarebbe possibile risparmiare prezioso tempo di calcolo evitando di allungarlo dietro scenari di cui si evince fin da subito che non siano favorevoli.

Nasce così la necessità di disporre di una ***scelta dinamica del numero di iterazioni***, e la logica con cui progredire con la simulazione o arrestarla, è basata sul test statistico già visto in precedenza. Il processo così realizzato si compone dei seguenti punti cardine.

1. Come primo passo si compie la simulazione non ottimizzata, con un alto numero di iterazioni N_{notOptm} affinché si abbia un'ottima precisione sul calcolo della sua media di *cpfh*, così come spiegato nel paragrafo precedente sull'analisi statistica;
2. In seguito, si compie la prima[56] simulazione ottimizzata scegliendo un numero iniziale di iterazioni N_1, concluse la quali si opera un primo processo di post-processing in cui vengono utilizzati i dati ricavati per compiere il t-test.
   - Se il suo risultato è di rigetto dell'ipotesi, vuol dire che lo scenario ottimizzato simulato è sufficientemente incisivo da poterne decretare la positività o negatività: in tal caso il processo simulativo si arresta e si può passare direttamente alla fase 4 di questa lista.
   - Se invece l'ipotesi nulla non venisse rigettata, significherebbe che l'attuale numero di iterazioni non è sufficiente per asserirne l'accettabilità: in tal caso servirebbe continuare a simulare, da cui il passaggio al punto 3.
3. Siccome servono più iterazioni, se ne sceglie un nuovo numero e si continua a simulare, accumulando realizzazioni di *cpfh* e affiancandole a quelle già ottenute dal passo 2. Concluso questo nuovo *gruppo simulativo* si ricompie il post-processing intermedio e il t-test, secondo la logica vista nel punto 2: se l'ipotesi viene rigettata si interrompe l'esecuzione, passando al punto 4; altrimenti si ricomincia dal punto 3 e andando avanti finché o non si raggiunge una condizione di rigetto o un limite massimo di iterazioni N_{max} precedentemente fissato.
4. Conclusa la fase simulativa, si compie il post-processing finale con generazione dei file di output e terminando l'esecuzione dello Scenario Evaluator. Si passa quindi il controllo all'Ottimizzatore, che valuterà i nuovi output di costo e la loro attendibilità, scegliendo un nuovo scenario ottimizzato e ripartendo dal punto 2.

[55] L'ipotesi in questione fa riferimento al test esposto al paragrafo precedente sull'accettazione o rigetto dell'ipotesi nulla di uguaglianza delle medie (di simulazione ottimizzata e non).
[56] Si discute della prima, ma il discorso vale identicamente per qualsiasi altro ciclo di ottimizzazione.

A livello di implementazione, il numero di iterazioni ottimizzate a ogni gruppo simulativo viene assegnato secondo la seguente legge che sfrutta parametri inseriti dall'utente:

- il primo gruppo ha un numero di iterazioni $N_1 = N_{\text{start}}^{\text{Optm}}$ definito nel file di input;
- ogni gruppo seguente al primo ha $N_k = N_{\text{MAX}}^{\text{Optm}}/10$ iterazioni, che si sommano a quelle già compiute, con $N_{\text{MAX}}^{\text{Optm}}$ anch'esso inserito dall'utente nel file di input.

3.3.3.1 Metodologia per la scelta del numero di iterazioni

Un passo fondamentale per una corretta esecuzione del processo simulativo è la scelta dei parametri sul numero di iterazioni da eseguire, dunque:

- N_0^{notOptm} , numero di iterazioni della simulazione non ottimizzata;
- $N_{\text{start}}^{\text{Optm}}$, numero di iterazioni iniziale della simulazione ottimizzata;
- $N_{\text{MAX}}^{\text{Optm}}$, numero massimo di iterazioni della simulazione ottimizzata.

Tutti e tre hanno influenza sul tempo di calcolo e sulla sicurezza statistica dei risultati. Riguardo quest'ultimo punto, ad esempio, maggiore è il numero delle iterazioni e: minore è l'influenza di eventuali *outliers* che si potrebbero presentare; oppure ancora, maggiore è la robustezza all'allontanamento da alcune delle ipotesi su cui il metodo si fonda[57].

In particolare, ad essere principali sono N_0^{notOptm} e $N_{\text{MAX}}^{\text{Optm}}$, poiché:

1) influenzano il grado di precisione che si vuole raggiungere nell'identificazione dell'accettabilità col t-test;
2) dipendono dalle caratteristiche stesse del sistema studiato.

Sull'ultimo punto non si può avere influenza, ma insieme a quanto esposto dal primo riletto in senso opposto, è possibile evincerne l'utilità ai nostri propositi:

> " definita, secondo certi parametri e criteri, la "precisione" da raggiungere con le analisi, e note (o ipotizzate) le caratteristiche del sistema statistico in analisi, si possono ricavare i valori del numero di iterazioni tali da rispettare queste condizioni. "

L'obiettivo rimasto è dunque trovare questa relazione.

Implementazione

Tale relazione è stata ricercata sia bibliograficamente a livello teorico, sia fra eventuali librerie Matlab preesistenti, trovando in esse una funzione built-in di nome *sampsizepwr* – "sample size and power (of the test)". Per come è stata impiegata[58], ha permesso di calcolare in output le grandezze dei campioni desiderate, richiedendo in input i seguenti parametri:

- *testType* , tipo di test in questione (z-test, t-test a uno o due variabili, chi-quadro test ecc.);
- *p0* , parametri validi sotto l'ipotesi nulla (quali di preciso dipendono dal tipo di test scelto);
- *p1* , parametro valido sotto l'ipotesi alternativa (quale di preciso, dipende dal tipo di test);
- *tail* , modo per intendere la significatività, se *one-tailed* (destro o sinistro) o *two-tailed*;
- *pwr* , potenza del test;
- *signLvl* , significatività del test;
- *ratio* , per un test a due variabili, è il rapporto desiderato fra i due campioni.

[57] Ad esempio, l'ipotesi di normalità della distribuzione di probabilità delle popolazioni in esame. Il metodo descritto infatti risulta *esatto* per qualsiasi grandezza dei campioni ma *solo* in caso questa condizione sia rispettata. Se non lo è, altre proprietà matematiche vengono incontro tendendo a sanare la problematica (quali, ad esempio, quanto espresso dal Teorema del Limite Centrale), ma risultano tanto più robuste quanto numeroso è il campione considerato.

[58] La funzione ha infatti diversi output a seconda degli input. La descrizione completa delle sue capacità esula dai nostri interessi e verrà dunque omessa. Si vedano i riferimenti bibliografici per approfondimenti.

Considerato il caso che interessa queste analisi[59], si è andati a compilare nel seguente modo i campi appena esposti:

- *testType = 't2'* , ovvero selezionando un t-test a due variabili normalmente distribuite con campioni a varianza raggruppata[60], ignota e *uguale*.

  Nota sull'ipotesi di uguaglianza delle varianze:
  l'ideale sarebbe poter scegliere l'esatto test di Welch, ovvero un 't2' con varianze generiche (*non* uguali). Tuttavia non è possibile fra le scelte, perciò si è optato per quello ad esso più affine. La correttezza di quanto scelto è comunque suggerita dal fatto che ci si aspetta che le varianze non siano poi tanto differenti fra loro nello stretto range di osservazione che caratterizza questo tipo di sistema (nell'intorno del punto percentuale).

- *p0 = [mean_notOptm, std_notOptm]*, ovvero fornendo la media e la deviazione standard di popolazione della prima variabile (quella della simulazione non ottimizzata). Ovviamente, i dati di popolazione non sono disponibili, pertanto li si deve approssimare: una soluzione adeguata sarebbe condurre una simulazione una tantum con un numero molto alto di iterazioni, con la quale evincere tali dati dalle osservazioni, approssimandoli con quelli campionari.
  Nel caso delle analisi relative ai risultati finali esposti in questo deliverable, tali dati sono stati ottenuti dalla già vista simulazione non ottimizzata, con 10^5 iterazioni.

- *p1 = mean_Optm* , media di popolazione della seconda variabile, che si desidera distinguere dalla prima con il grado di precisione specificato dagli associati parametri inseriti (seguenti nella lista). Il suo valore viene scelto in base alla conoscenza dei margini permessi dall'ottimizzazione, e congruentemente ad essi.

- *Tail = 'both'* , poiché siamo interessati a un test two-sided per l'ipotesi alternativa delle due medie diverse fra loro.

I campi rimanenti (potenza e significatività del test, e rapporto fra le grandezze dei due campioni) definiscono la precisione statistica desiderata e vanno scelti secondo un trade-off con il tempo di calcolo, a sua volta dipendente sia dalla potenza di calcolo disponibile che dalla complessità attesa dell'ottimizzazione, nota a priori da esperienza pregressa.
Svolte queste indagini, si è andati a compilare nella seguente maniera i campi discussi:

- *pwr = 90%* ,
- *signLvl = 2%* ,
- *ratio =* 16.5.

In conclusione, impiegando la funzione *sampsizepwr* con i suddetti dati di input, si sono ottenute le grandezze campionarie richieste, ovvero:

$$N_{\mathrm{I}} = N_0^{\mathrm{notOptm}} = 10\text{^}4 \quad \text{e dunque} \quad N_{\mathrm{II}} = N_{\mathrm{MAX}}^{\mathrm{Optm}} = 3 \cdot 10^3 \ .$$

[59] Si veda la sua definizione al paragrafo "Analisi statistica", alla fine della sottosezione "Comprensione del caso specifico".
[60] "varianza raggruppata", probabilmente meglio nota col suo nome inglese "*pooled variance*".

3.4 L'integrazione dei processi

Sono state definite nel dettaglio le caratteristiche dei tre elementi dell'OLSO v.1; possiamo mostrare ora il flusso operativo (spesso chiamato *workflow*), che struttura la sequenza e le relazioni fra di essi. Si riporta di seguito uno schema che ne rappresenta il risultato finale.

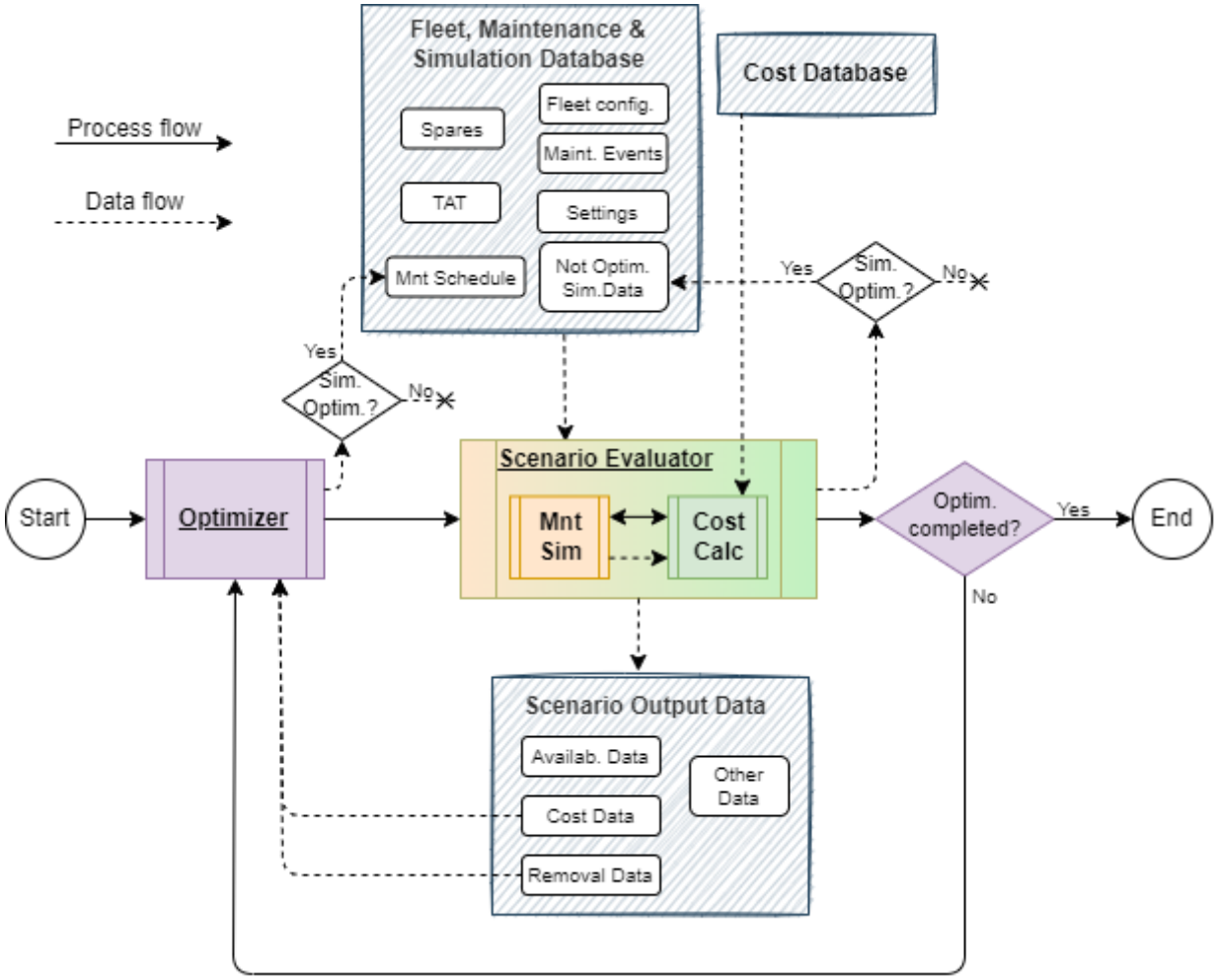


Figura 17) Schema concettuale del funzionamento dell'OLSO v.1 per l'ottimizzazione secondo metodo degli anticipi

Si tratta dello stesso schema visto all'inizio di questo capitolo, con in più l'indicazione specifica dei passi per trattare la prima simulazione, quella non-ottimizzata.
In linea con l'obiettivo di esposizione finalizzata alle analisi, si omette la spiegazione dettagliata delle caratteristiche implementative dell'integrazione. Al fine di rendere almeno l'idea di come appare uno dei workflow su modeFRONTIER, se ne mostra uno nella Figura 18.

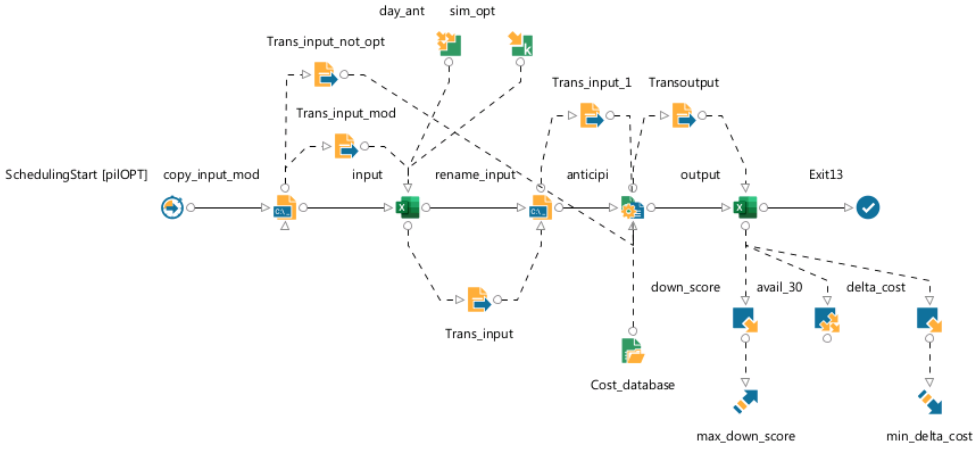


Figura 18) Workflow in modeFRONTIER, esempio di integrazione di processo.

Capitolo 4 Caso di studio e risultati

In questo capitolo si affronta il secondo obiettivo di questa tesi: l'analisi di una delle ottimizzazioni di Development Level 1, in particolare quella relativa alla metodologia per gli anticipi, per la quale sono stati specificatamente sviluppati Scenario Evaluator e Optimizer.
L'analisi della metodologia è stata portata avanti su un caso di studio realistico, da ora in poi chiamato *test case*.

L'esposizione viene articolata in tre sezioni:
- dapprima si descrive il caso di studio considerato, descrivendone gli input così da comprendere con chiarezza le caratteristiche dello scenario di partenza ;
- successivamente si trattano le analisi preliminari, necessarie per definire alcuni degli input esposti nella sezione precedente ;
- infine si mostrano i risultati, con i vari grafici e quantità fondamentali, commentandoli.

4.1 Panoramica degli input

Le analisi sono state condotte su una flotta che, anche se virtuale, è stata modellata in similitudine ad una flotta realmente esistente. Per ragioni di segretezza, non verranno qui esposti esplicitamente gli esatti valori numerici in gioco, anche perché non si ritiene siano di particolare interesse. In ogni caso, essi verranno illustrati a livello generale, di modo da avere chiare comunque le dimensioni e le caratteristiche dello scenario.
A livello generale possiamo descrivere la flotta impiegata nei seguenti punti fondamentali:
- la flotta è di medie dimensioni, costituita da più di 200 motori e più[61] di 80 velivoli ;
- i velivoli sono tutti bimotore e della medesima tipologia.;
- per ciascun velivolo sono pianificate mediamente 150 ore di volo annue;
- i motori sono tutti della stessa tipologia (generazione/configurazione) e ogni motore è composto da una decina di moduli, sostituibili a ML2 (Maintenance Level 2).

Si vada nel seguito a descrivere più nel dettaglio il contenuto dei dati, il processo di loro estrapolazione e i motivi per cui sono stati scelti tali, suddividendo l'esposizione nei due gruppi di input (Dati di simulazione e Dati di costo).

4.1.1 Dati di simulazione

Si vada a descrivere le caratteristiche della flotta in termini di soggetti (velivoli, motori e items), ed eventi a cui sono sottoposti.

- **Flotta: dimensione ed età**
  Nel scegliere il tipo di flotta si è tenuto conto di due fattori:
  - Dimensione: è necessario scegliere una flotta non troppo ridotta perché questo rende scarsamente applicabile il metodo degli anticipi per problemi di possibilità di manovra.
  - Età: per avere realisticità, è necessario considerare una flotta non-nuova, con un certo numero di items in riparazione ad inizio simulazione.

- **Manutenzione: TATs**
  Quantitativamente, i TAT considerati sono piuttosto elevati: si attende che ciò avrà sicuramente un forte impatto sulla manutenzione, allungando i tempi di riparazione e rendendo così più critica la situazione media.

[61] Il numero esatto non viene divulgato per ragioni di riservatezza.

- **Eventi manutentivi:**
  sono stati apportati dei cambiamenti rispetto alla flotta reale, in modo da adattare il file di input alle ipotesi dell'OLSO v.1. In particolare, una sola scadenza programmata. Nel caso dunque un certo componente ne avesse più d'una, si è andati a considerare solo la prima: tale scelta è stata arbitraria, non essendoci particolari differenze nell'utilizzare una piuttosto che un'altra.

- **Utilizzo flotta: scheduling dei voli**
  Non si disponeva di uno scheduling e dunque lo si è generato, tenendo le 150 afh/(anno*velivolo) come valore medio della distribuzione.

- **Settings:**
  Si tratta di parametri vari di simulazione. Ne esistono di tanti tipi; qui riportiamo solo quelli significativi per la comprensione dell'analisi.

  - Simulazione:

Parametro – Valore	Significato
Durata simulazione = 3 anni	Lunghezza temporale della simulazione.
MISL = 100 hours	Minimum Issued Service Life: numero di ore residue minime da garantire a fronte di una riparazione motore effettuata a ML2 o a ML3. Non si applica per le riparazioni a ML1. Scelto come valore tipico contrattuale per questa tipologia di motore.
EFH per mission = 1.5 hours/mission	Ore medie per missione. Come per le ore medie di volo annue, anche questo dato è stato scelto tramite esperienza in base alla tipologia di motore selezionato.
Max Missions per day = 2 missions/day	Numero massimo di missioni che un velivolo può compiere in una giornata. Anche questo viene scelto in base alla tipologia di motore selezionato.
Availability minima = 0.86	Availability (media mensile) minima permessa dal contratto per i motori. È stata scelta in maniera tale da generare dei momenti di down, di modo da testare la logica di ottimizzazione.

  - Ottimizzazione:

Parametro – Valore	Significato
*N sim **not-optim*** = $\mathbf{10^5}$	Numero di iterazioni Monte Carlo per la simulazione non ottimizzata. Il motivo di questa particolare scelta è esposto nel dettaglio nel prossimo paragrafo.
*N sim **optim** (Start)* = **200**	Numero iniziale di iterazioni Monte Carlo per una simulazione ottimizzata. È stato scelto come compromesso: da una parte l'essere sufficientemente alto da poter inferire statisticamente in maniera sensata sulle caratteristiche medie di scenario; dall'altre l'essere non troppo alto per non intaccare negativamente il tempo di calcolo.
*N sim **optim** (MAX)* = $\mathbf{5 \cdot 10^3}$	Numero massimo di iterazioni Monte Carlo per una simulazione ottimizzata. Anche questo parametro è stato scelto in base alle analisi mostrate di seguito.
***Significance** Level* = 2 %	Significance Level del two-sided test per decretare se la simulazione ottimizzata ha o meno media uguale a quella della simulazione non ottimizzata. Scelto arbitrariamente, ma in modo da essere sufficientemente basso da non incappare in troppi falsi positivi, e sufficientemente alto da non allungare eccessivamente i tempi di calcolo.

4.1.2 Dati di costo

I dati di costo sono notoriamente fra quelli più sensibili all'interno di un'azienda, pertanto è stato impossibile ottenerli da condizioni effettive per questioni di riservatezza. Anche bibliograficamente la loro ricerca è stata particolarmente complessa e non si è riusciti a trovare dei dati soddisfacenti allo scopo. Pertanto, si è andati a compilare il database dei costi seguendo alcune indicazioni di massima fornite dal partner di progetto di riferimento, e sfruttando alcuni dati noti (discussi in seguito).
Ad esempio, il costo dei TAT (imposti scegliendo *kBase*_rep e *kBase_exch*) è stato assegnato conoscendo il salario medio di un manutentore e ipotizzando il numero di ore/uomo necessarie per una certa operazione manutentiva, noto il valore (realistico) dei TAT.

Dopo aver completato una prima stesura del database, lo si è andato a validare con una simulazione di prova in modo da ottenere il costo totale della manutenzione (non considerando le penali), e accertandosi che esso fosse congruente almeno nell'ordine di grandezza a quanto atteso dagli studi bibliografici, e così è stato.
A tal proposito, tale valore di riferimento sul costo medio dell'operazione di un motore[62] è stato calcolato partendo anzitutto dalla nozione approssimativa del life cycle cost per l'intero velivolo. Dopodiché, attraverso la conoscenza delle percentuali tipiche legate alla manutenzione e quelle specificatamente relative al sistema propulsivo, si è ricavato il valore desiderato.
In ogni caso, il valore in termini assoluti del costo della manutenzione non è particolarmente di interesse per l'ottimizzazione, che viene influenzata invece dai rapporti fra le varie grandezze in gioco.

Per quanto riguarda il costo delle penali, è stato considerato il massimale di penale associato a un valore pari al 5% di down di availability. Il valore è stato scelto in base alle peculiarità di simulazione e come queste avrebbero influenzato un ipotetico scenario realistico con esse come caratteristiche. Infatti, considerata la non presenza di rescheduling dei voli, è risultato plausibile richiedere un availability relativamente alta, da cui un massimale di penale più vicino al target di availability.

4.2 Analisi preliminari

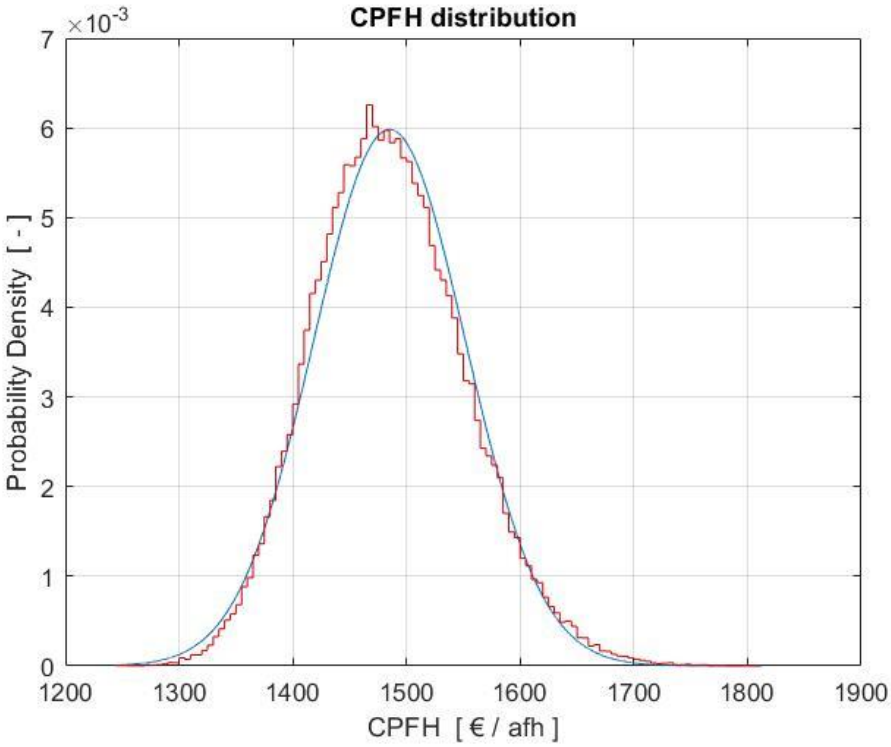
Prima di simulare è necessario compiere alcune analisi preliminari, propedeutiche (per ragioni diverse) all'analisi principale. Esse sono:

- *Verifica della quasi-normalità della distribuzione di costo di ottimizzazione:*
  necessaria per assicurare il rispetto delle ipotesi alla base dei metodi di indagine statistica ;
- *Definizione del numero di iterazioni:*
  necessario per la simulazione, garantendo determinati requisiti di significatività statistica ;
- *Analisi dei margini di ottimizzazione:*
  per accertarsi che esista la possibilità di applicazione del metodo degli anticipi al caso considerato.

4.2.1 Verifica della quasi-normalità della distribuzione di costo di ottimizzazione

Come primo studio, si è andati a confermare che anche nel nuovo scenario considerato, la distribuzione di costo di ottimizzazione (il cost per flight hour) ricalcasse in maniera adeguata la forma di una curva normale.
Per ottenere quanto voluto, si è attuata una simulazione non ottimizzata con numero molto alto di iterazioni (10^5) e la si è andata a confrontare con la distribuzione normale di media e deviazione standard pari a quelle del campione simulato. I risultati dell'analisi sono mostrati nella figura seguente (in rosso la distribuzione simulata, in azzurro la gaussiana associatagli).

[62] Considerato l'ambito in questione, qui non divulgato.



La distribuzione di costo calcolata (curva rossa) risulta leggermente spostata "verso sinistra" rispetto alla sua controparte normale. Tale risultato è in linea con le aspettative, poiché i costi tendono a distribuirsi nella pratica secondo distribuzioni normali o log-normali; in questo caso l'asimmetria sinistra è comprensibilmente dovuta a una componente log-normale.
Per quantificare numericamente quanto le due curve siano differenti si è andato a calcolare la *skewness*[63] della distribuzione. Il risultato è stato reputato soddisfacente e si è potuto procedere con le fasi successive.

4.2.2 Definizione del numero di iterazioni

La procedura è stata portata avanti secondo quanto discusso nella sezione dedicatagli nel capitolo scorso.
Per il *test case*, è stato selezionato un t-test a due variabili di tipo two-sided, con potenza al 90% e significance level al 2%, e sono stati fornite le caratteristiche del sistema non ottimizzato in termini di media e deviazione standard, secondo quanto ottenuto dalla prova su 10^5 iterazioni. Inoltre, si è deciso di non avere un numero di iterazioni non ottimizzate superiore a 10^5. Riguardo la precisione con cui distinguere le due medie, un valore dello 0.2% è stato selezionato: esso è legato alle previsioni dei margini di ottimizzazione, che verranno indagati nel dettaglio nel prossimo paragrafo.

Infine, per ottenere quanto cercato, poiché non era possibile inserire manualmente il numero di iterazioni non ottimizzate ma solo agire sul *ratio* fra le due, attraverso un metodo di tipo trial and error si è arrivati al seguente risultato:

- $ratio = 16.7$
- $N_{\text{iter}}^{\text{sim.Opt.MAX}} = 6\,005$
- $N_{\text{iter}}^{\text{sim.notOpt.}} = 100\,283$

da cui conseguono i risultati poi effettivamente impiegati di $6 \cdot 10^3$ e 10^5.

[63] Come visto nella sezione relativa alle Basi di probabilità, la skewness è una misura del grado di asimmetria di una distribuzione (in termini di PDF).

4.2.3 Analisi dei margini di ottimizzazione

L'analisi dei margini di ottimizzazione è probabilmente il più importante degli studi preliminari, poiché pone i criteri per stabilire se il sistema è incline o no all'ottimizzazione mediante il metodo degli anticipi. Per farlo, ci si pone in una condizione semplice, tale da poter essere trattata con metodi analitici, e in particolare con l'impiego di semplici calcoli algebrici.

L'idea è di porsi in uno scenario per cui, anticipando di un giorno la manutenzione di un determinato item, si riesce a guadagnare completamente l'availability che in quel giorno era in down (proporzionalmente all'influenza del motore considerato). Si calcolano quindi le quote di costo positive e negative che gli si associano.

- Con *quota negativa* si intende l'ammontare di risparmio ottenuto dall'eliminazione del momento di down, ed è pari costo della penale per punto percentuale (normalizzata al giorno) moltiplicata per la percentuale di availability associata a un singolo motore.
- Con *quota positiva* si intende l'ammontare del costo sopportato a seguito dell'anticipo stesso, che causa lo scarto di ore utili dell'item. Per ottenerlo, si calcola dapprima il costo per ora-motore di volo legato alle revisioni/riparazioni, e quello relativo alle parti a vita limitata. In seguito, per ogni item anticipabile (ovvero dotato di almeno una scadenza programmata), si va a calcolare il costo associato ad un suo giorno di anticipo, considerando anche l'influenza degli eventi di subsidiary damage, che accoppiano il costo con quello degli altri componenti, e il quantitativo medio di ore volate in un giorno.

Quanto appena esposto è stato dunque implementato sottoforma di codice MATLAB, generando uno script che, applicato ai dati del *test case*, ha riportato i risultati mostrati con la figura seguente.

```
Il costo delle penali al giorno (per motore in down) è:    cFin/day =  505.95  €/(day*EngDown)

Il costo per anticipare un certo Item di un giorno è:
    IC    ITEM     CostPerDayOfAnt
    __    _____    _______________

     1    "M01"           NaN
     2    "M02"           NaN
     3    "M03"        65.841
     4    "M04"        68.303
     5    "M05"        113.59
     6    "M06"           NaN
     7    "M07"           NaN
     8    "M08"        57.577
     9    "M09"           NaN
    10    "M10"           NaN
    11    "ENG"           NaN
```

Figura 19) Risultati dell'analisi dei margini di ottimizzazione applicata al test case.

Come si nota, ogni motore in down causa una penale di $505.95\ €/(day * eng_{down})$, e questo dato rappresenta la quota negativa di costo. Guardando al costo dell'anticipo, esso è ovviamente calcolato solo per i quattro componenti con scadenze programmate, per gli altri vi è il valore 'Not a Number'. Tutti e quattro hanno un costo inferiore al primo, pertanto si considera rispettata la condizione necessaria di ottimizzazione.
Si può dunque procedere alla simulazione, per vedere all'atto pratico come si comporta l'ottimizzazione dello scenario considerato. Durante il processo, ci si attende che i reali margini siano inferiori, a causa di una serie di allontanamenti dalle ipotesi del calcolo semplificato, pertanto non è per nulla semplice, comprendere l'esito dell'ottimizzazione senza una simulazione completa.

4.3 Risultati

A seguito della definizione degli input di scenario e il completamento di tutte le analisi preliminari, è stato possibile far partire l'ottimizzazione del test case. Per la precisione, ci sono stati una serie di altri approfondimenti specificatamente relativi all'ottimizzazione, dalla scelta dell'algoritmo, al suo ausilio con definizione di metodi ad hoc che lo supportassero. Conclusi anche questi si è fatto partire ufficialmente il test case, e in questa sezione del deliverable se ne riportano i risultati principali.

4.3.1 Scenario di partenza

Come primo passo, si espongono anzitutto i risultati dello scenario di partenza, ovvero quello non ottimizzato, mostrandone i grafici di output e vedendo qualche dato numerico principale.

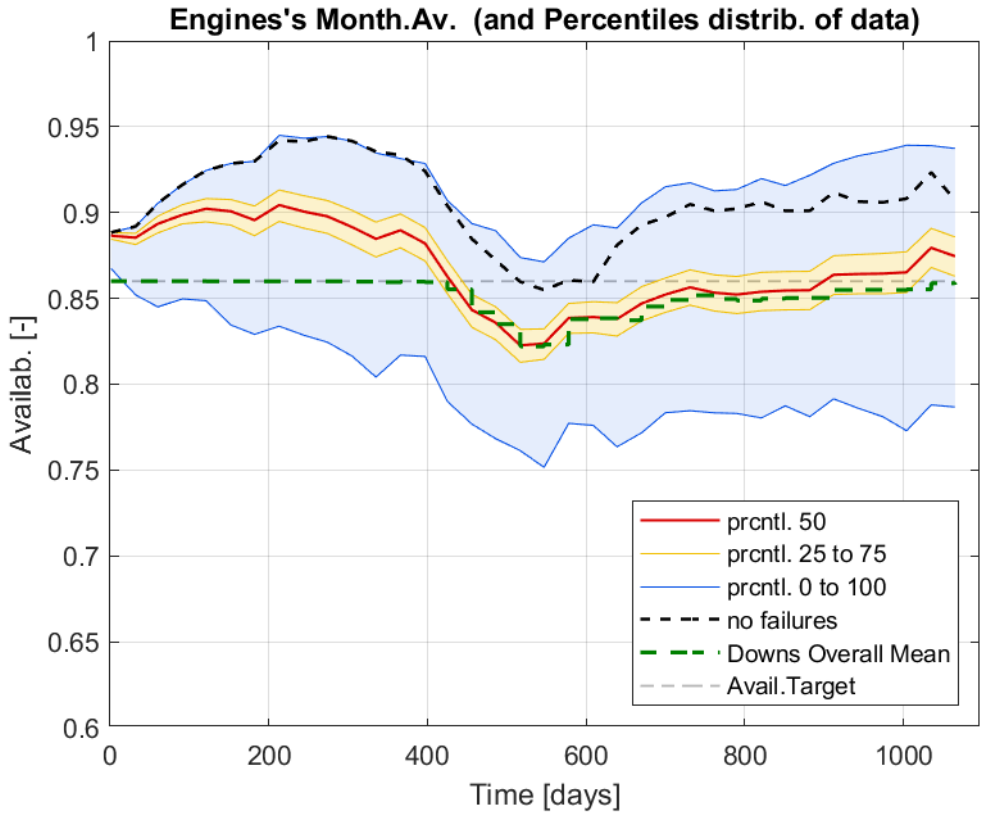


Figura 20) Simulazione non ottimizzata, andamento dell'availability-motori mensile nel tempo.

A livello di momenti di down, se ne configura uno solo e posto a circa un anno e mezzo dall'inizio del periodo considerato. L'ottimizzatore dovrà cercare di spostare le manutenzioni in modo da far alzare quel pezzo di curva, e nel farlo andrà ad abbassare il primo con l'accortezza di non farlo scendere ovviamente al di sotto del limite contrattuale dell'86%.

Il grafico sulla distribuzione di costo non è particolarmente significativo e poi verrà comunque ripreso in seguito, per cui per ora lo si tralascia. Sufficienti informazioni sulla simulazione non ottimizzata sono state ottenute già dall'analisi preliminare sulla distribuzione di costo.

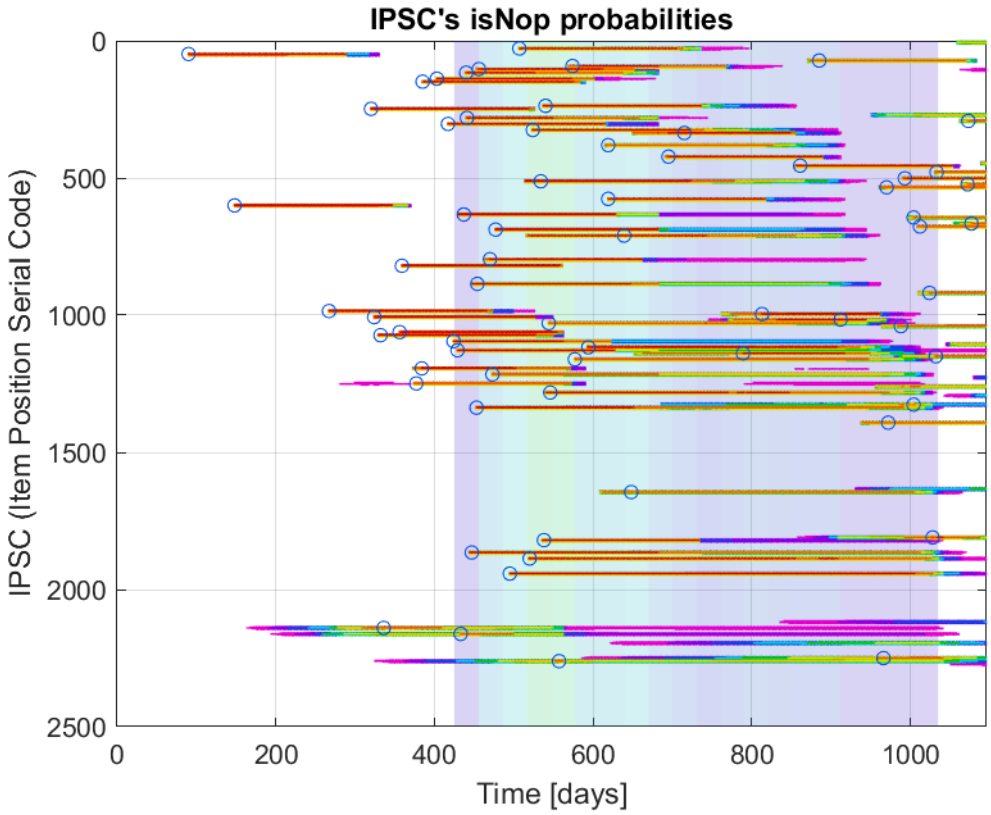


Figura 21) Simulazione non ottimizzata, probabilità di inefficienza dei componenti nel tempo.

Il grafico in Figura 21 serve più che altro a mostrare l'influenza del subsidiary damage. È dal prossimo con le scadenze programmate che si evincono più concretamente le caratteristiche legate agli anticipi.

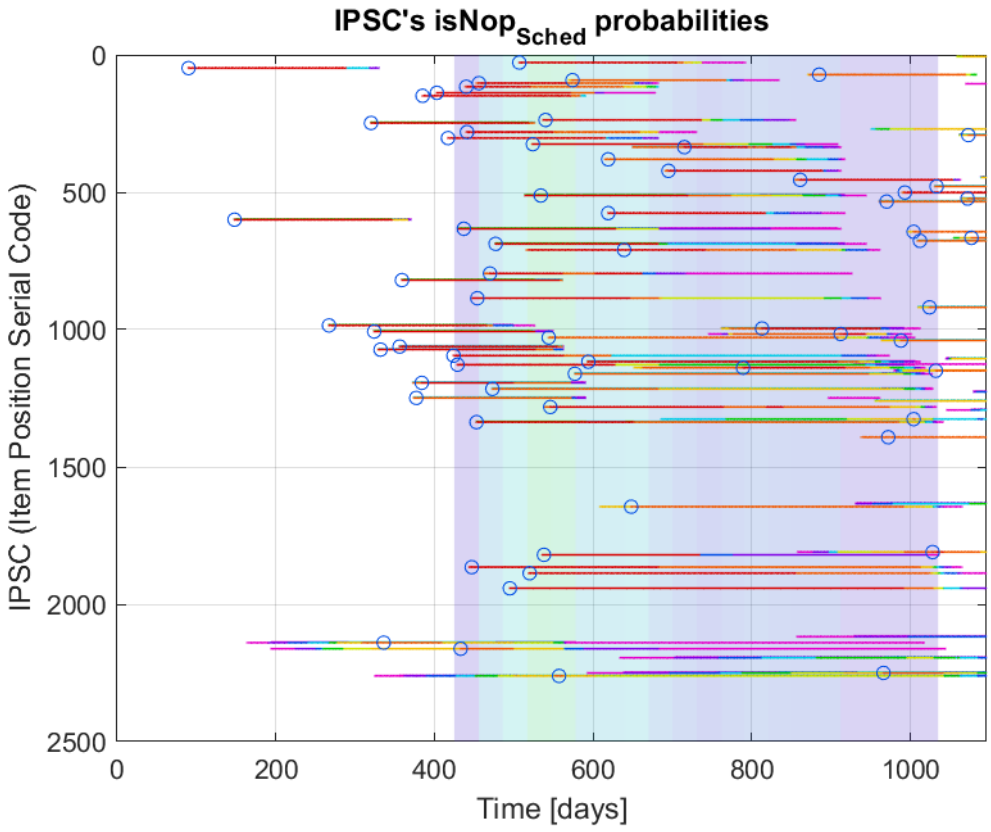


Figura 22) Simulazione non ottimizzata, probabilità di inefficienza, nel tempo, dei componenti per scadenze programmate.

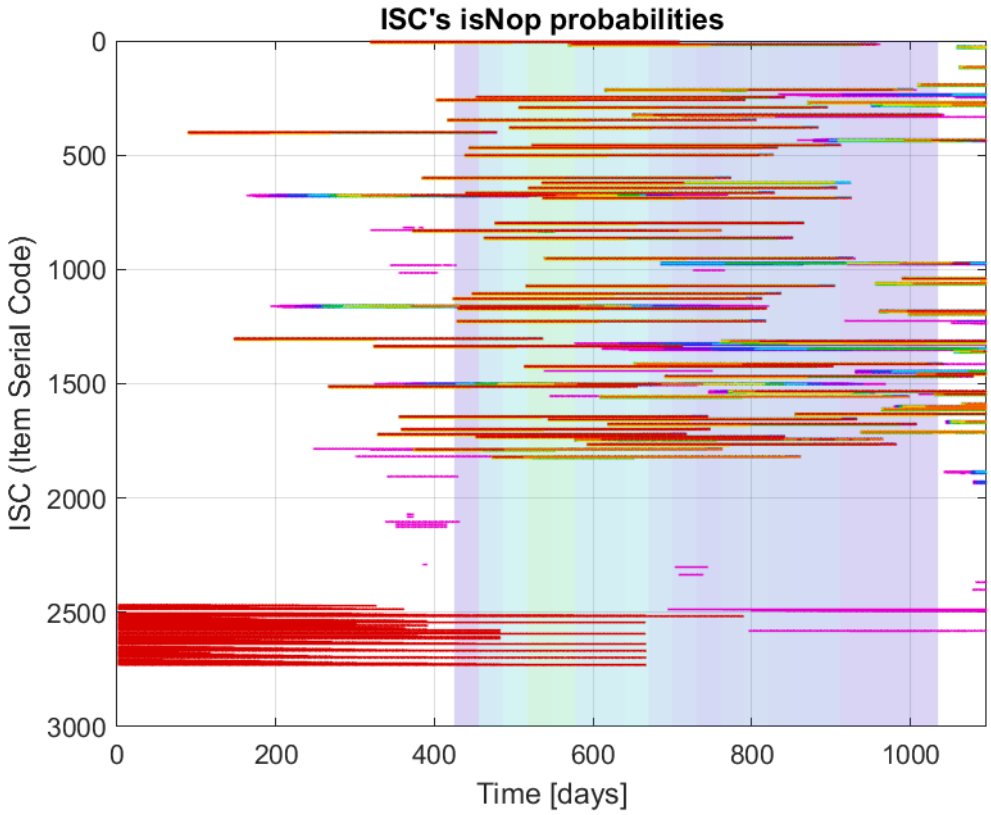


Figura 23) Simulazione non ottimizzata, probabilità di inefficienza dei componenti nel tempo (catalogazione per Serial Number).

A livello di dati numerici, si trova che la simulazione ha calcolato un $cpfh = 1485.11\ €/fh$, di cui il 1.66% sono relativi alle penali. Questo significa che al massimo è possibile ottenere tale valore come risparmio totale (precisamente, più basso considerando anche il costo degli anticipi e le complessità del caso pratico).
Riguardo i momenti di down, è stato ricavato un valore di $downScore = -2.10$.

Come componenti anticipabili, ne sono stati suggeriti 66 in tutto:
- 18 di livello 1[64],
- 24 di livello 2,
- 15 di livello 3,
- 9 di livello 4.

Circa la loro composizione, si ha:
- 57 moduli 'M05' (con LLP) ;
- 8 moduli 'M03' (con scadenza per Overhaul) ;
- 1 modulo 'M04' (con LLP) .

A giudicare dalla prevalenza di 'M05' e dalle analisi di margini di ottimizzazione, che lo vedono essere purtroppo anche il meno incline all'anticipo, ciò lascia pensare che questo potrebbe rendere più complicata la procedura di ottimizzazione.

[64] Con "Livello" si intende il *livello di probabilità di non operatività*, che cataloga la probabilità più alta che un certo item non sia operativo durante tutto il periodo simulato. I livelli vengono assegnati nel seguente modo:
- Livello 1: probabilità nell'intervallo (0.95 , 1]
- Livello 2: probabilità nell'intervallo (0.90 , 0.95]
- Livello 3: probabilità nell'intervallo (0.85 , 0.90]
- Livello 4: probabilità nell'intervallo (0.80 , 0.85]

4.3.2 Processo di ottimizzazione

Al termine della simulazione non ottimizzata, il Maintenance Planner ha fatto partire la vera analisi al proprio cuore. L'ottimizzazione (algoritmo impiegato: pilOPT) ha generato 1500 design, e i risultati di questo processo sono ottimamente rappresentati dai seguenti due grafici:

- il primo è un *history chart* che mostra l'andamento del *delta_cpfh*[65] al variare del *design ID*, e dunque con l'avanzare dello studio;
- il secondo è un *bubble chart* volto a rappresentare in ascisse il *delta_cpfh* e in ordinate il *downScore* (è presente anche un suo zoom nella parte di maggiore interesse).

Li si riportano di seguito.

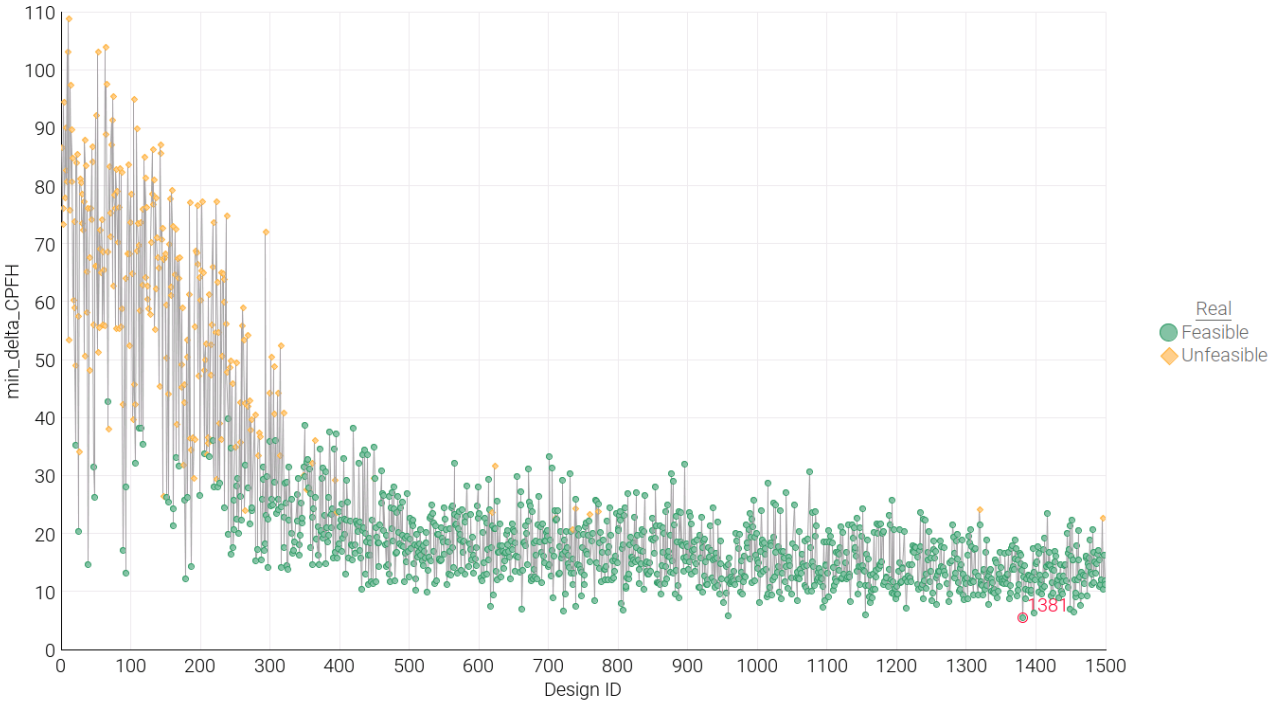


Figura 24) ***History chart*** **che mostra l'andamento del** ***delta_cpfh*** **al variare del** ***design ID.***

Da quanto si nota, c'è stato un chiaro processo di miglioramento della soluzione nel tempo. Come atteso, le prime simulazioni sono caratterizzate da alti valori di *delta_cpfh*: questo perché nelle prime fasi l'algoritmo tenta combinazioni randomiche (o quasi-randomiche) delle variabili di ottimizzazione in modo da esplorare lo spazio di design, e quindi c'è un'alta probabilità di incorrere in scenari peggiori di quello di partenza. Infatti, si ricorda che, solo i design che hanno mostrato un *downScore* migliore (meno negativo in segno) vengono considerati *feasible*, in caso contrario li si ignora ai fini dell'ottimizzazione.
Con l'avanzare dell'analisi l'algoritmo seleziona quei design (feasible) che hanno dimostrato di essere promettenti e indaga nei loro pressi, migliorando via via la soluzione. All'incirca al design n.300 viene trovata una via più stabile, portando all'ottenimento di un numero preponderante di soluzioni feasible. Proseguendo su di essa l'algoritmo continua nel miglioramento, anche se dal design circa n.500 esso risulta essere più blando, ma comunque continuo.
Come detto, il processo è stato arrestato al design n.1500 e si è concluso con soluzioni nell'intorno dei $10 - 12\ €/afh$, e con miglioramento dello scenario a livello di availability. Se si fosse proseguita l'ottimizzazione è probabile che si sarebbero trovati risultati ancora più promettenti.

Per comprendere meglio a livello quantitativo i risultati in termini di disponibilità e costo si può andare a visionare i seguenti due grafici.

[65] *delta_cpfh:* differenza fra il *cpfh* (medio) della simulazione ottimizzato rispetto a quella non-ottimizzata.

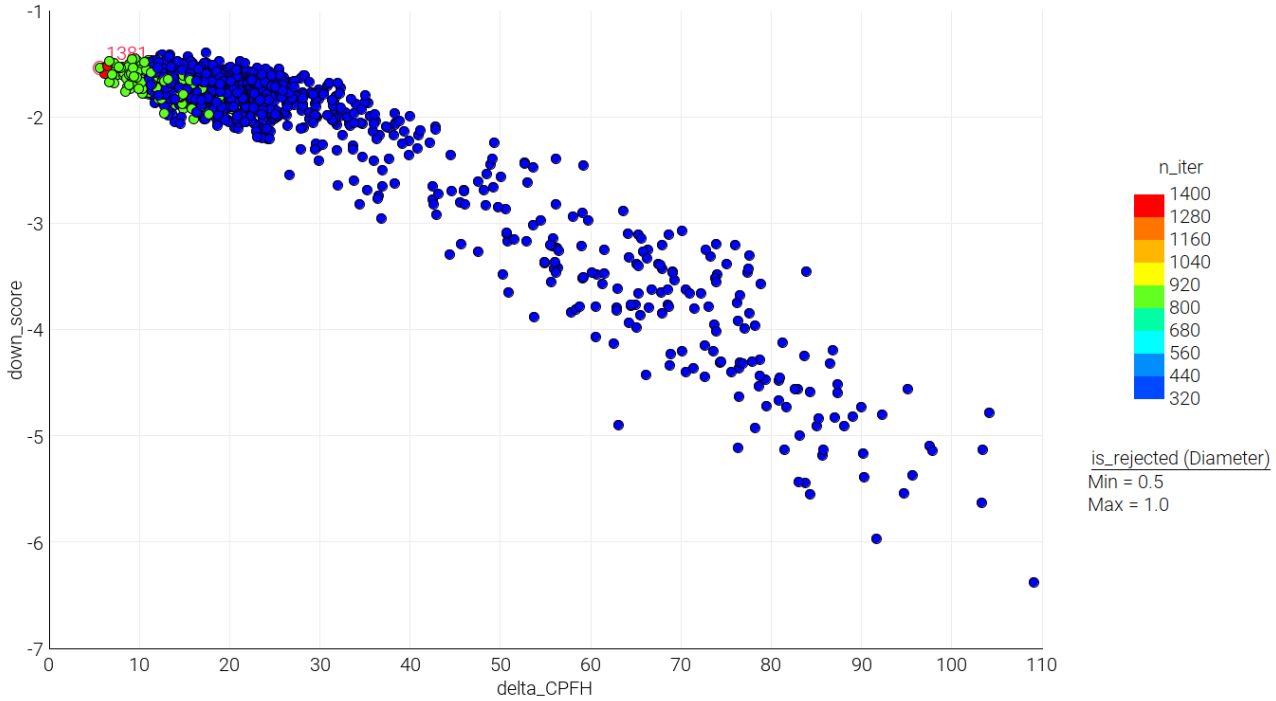


Figura 25) ***Bubble chart*** **volto a rappresentare in ascisse il** ***delta_cpfh*****, in ordinate il** ***downScore*** **e mediante il colore il numero di iterazioni della simulazione.**

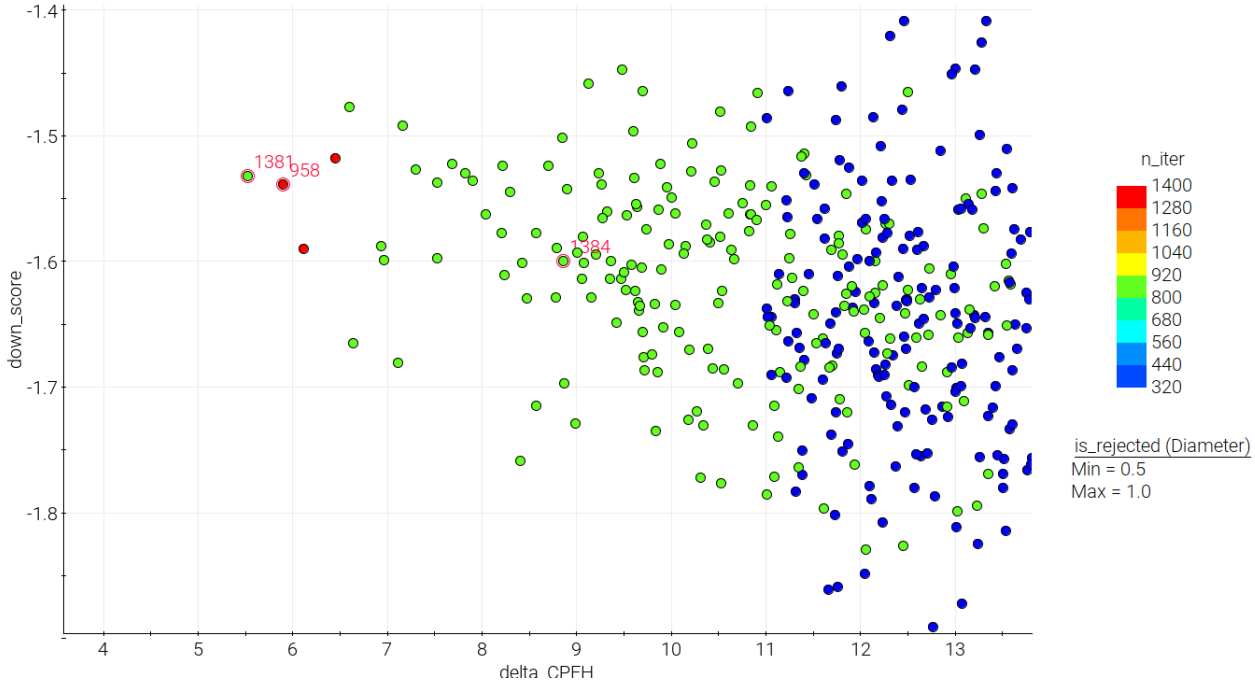


Figura 26) Zoom del grafico in Figura 25 nella sua parte di maggiore interesse (la punta in alto a sinistra).

Anche da questo grafico si evince chiaramente la tendenza a convergere verso una direzione comune migliorativa (bassi *delta_cpfh* e *downScore* meno negativi). La "scia" in basso a destra è probabilmente dovuta ai primi design scartati. Proseguendo verso la punta le soluzioni si addensano in particolare a partire da $downScore > -2.1 \div 2.3$, e questo plausibilmente perché solo per $downScore > -2.1 = (downScore)_{\text{simNotOpt}}$ esse diventano feasible, facendo concentrare l'attenzione dell'algoritmo da quel punto in poi. Infine, i design della punta sono gli ultimi indagati.

A livello di numero di iterazioni, non stupisce che la punta del grafico sia caratterizzata da valori più alti di $N_{\text{iterations}}$: ciò perché si va verso la neutralità di *delta_cpfh*, necessitando maggiore precisione per il superamento della condizione di rigetto dell'ipotesi nulla sulla differenza delle medie (ottimizzata e non). Circa il $10 \div 15$ % dei design hanno richiesto un aumento del numero di iterazioni rispetto al loro valore di base ($N_{\text{iter}}^{\text{sim.Opt.Start}}$) pari a 200, portandosi così a 800 ($N_{\text{iter}}^{\text{sim.Opt.Start}} + N_{\text{iter}}^{\text{sim.Opt.MAX}}/10$); solo 3 design hanno avuto bisogno di ulteriori 600 rilevamenti ciascuno.

In generale, si può notare che la punta del grafico tenda ad orizzontalizzarsi, segno di una difficoltà da parte dell'algoritmo di ottimizzazione a ottenere valori ulteriormente migliorativi in termini di disponibilità motori.

4.3.3 Scenario conclusivo

Dopo aver visto il processo di ottimizzazione, mostrando il modo con cui l'algoritmo è avanzato verso soluzioni via via più migliorative, si riportino di seguito i risultati in termini di scenario ottenuto.

4.3.3.1 Andamento ricavato

Si noti che, vista la relativamente lenta tendenza migliorativa delle fasi finali, gli scenari relativi agli ultimi design di ottimizzazione (quelli sulla punta) sono in gran parte simili fra di loro. Per mostrare i risultati a livello di andamento generale, dunque, si può prendere uno qualsiasi di essi.
A tal proposito, si è deciso di scegliere il design n.1381, che è anche il migliore fra tutti in termini di cost per flight hour, ed è stato considerato il migliore in generale. A proposito particolare della scelta del punto di ottimo, invece, la problematica è più complessa, e verrà discussa nel prossimo sotto-paragrafo.

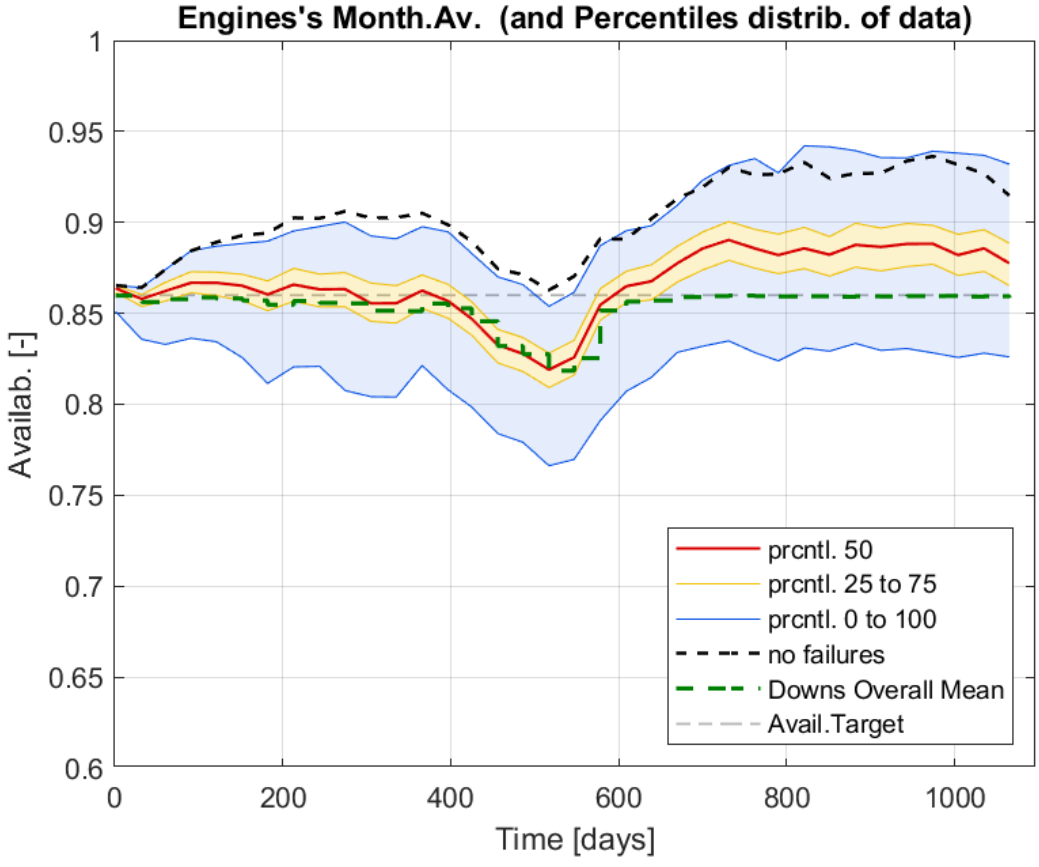


Figura 27) Simulazione ottimizzata n.1381, andamento dell'availability-motori mensile nel tempo.

Anche a prima vista risulta chiara la differenza col caso di partenza. L'algoritmo ha, come atteso, cercato di ridurre l'importanza del momento di down centrale spostando le manutenzioni e abbassando la parte anteriore della curva, allineandola quasi perfettamente al minimo. Anche l'andamento posteriore della stessa è variato, venendo sensibilmente sollevato oltre il minimo di availability e permettendo il recupero di tutta la parte prima in down in quei pressi, e ciò è probabilmente il motivo che ha causato il miglioramento del *downScore*.
Nel complesso il momento centrale di down è rimasto, come già ci si aspettava sarebbe successo, per via della complessità dello scenario di partenza. Si nota che comunque il suo picco negativo non è peggiorato.

Comunque, per entrare maggiormente nella comprensione di come ha lavorato l'algoritmo e valutarne la soluzione, è necessario andare a visionare i grafici successivi sulla disponibilità, che indicano inoltre in maniera grafica l'importanza e la distribuzione degli anticipi, supportando questa analisi anche con i dati numerici. Queste procedure finali di analisi degli scenari ottimizzati da parte di un tecnico esperto sono essenziali per confermare la corretta esecuzione del lavoro da parte del software.

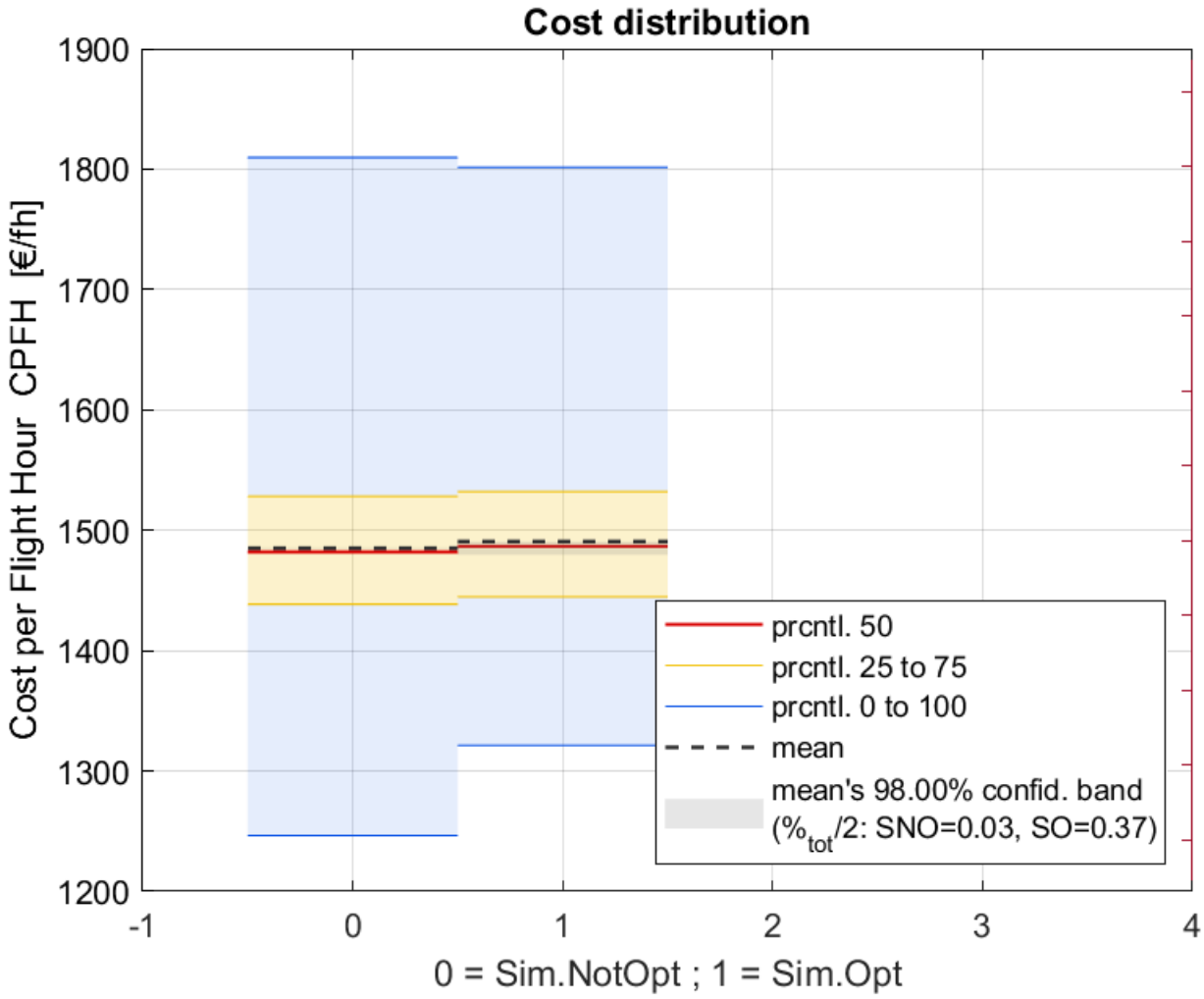


Figura 28) Simulazione ottimizzata n.1381, distribuzione del costo di ottimizzazione (*cpfh*)

Dal grafico di costo si possono notare le larghezze di banda di confidenza impiegate in questi design con 800 iterazioni. Per il resto, i dati di costo sono ben esposti anche nei fogli di lavoro di output e si prediligono questi per compiere analisi.

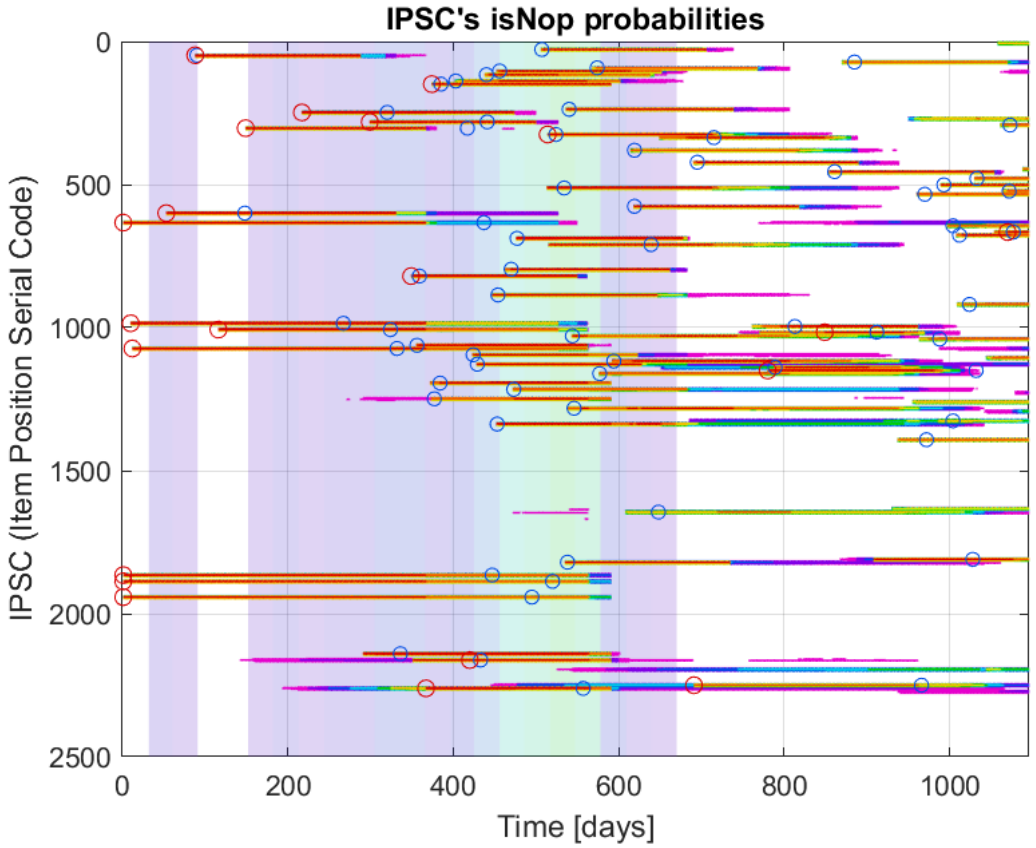


Figura 29) Simulazione ottimizzata n.1381, probabilità di inefficienza dei componenti nel tempo.

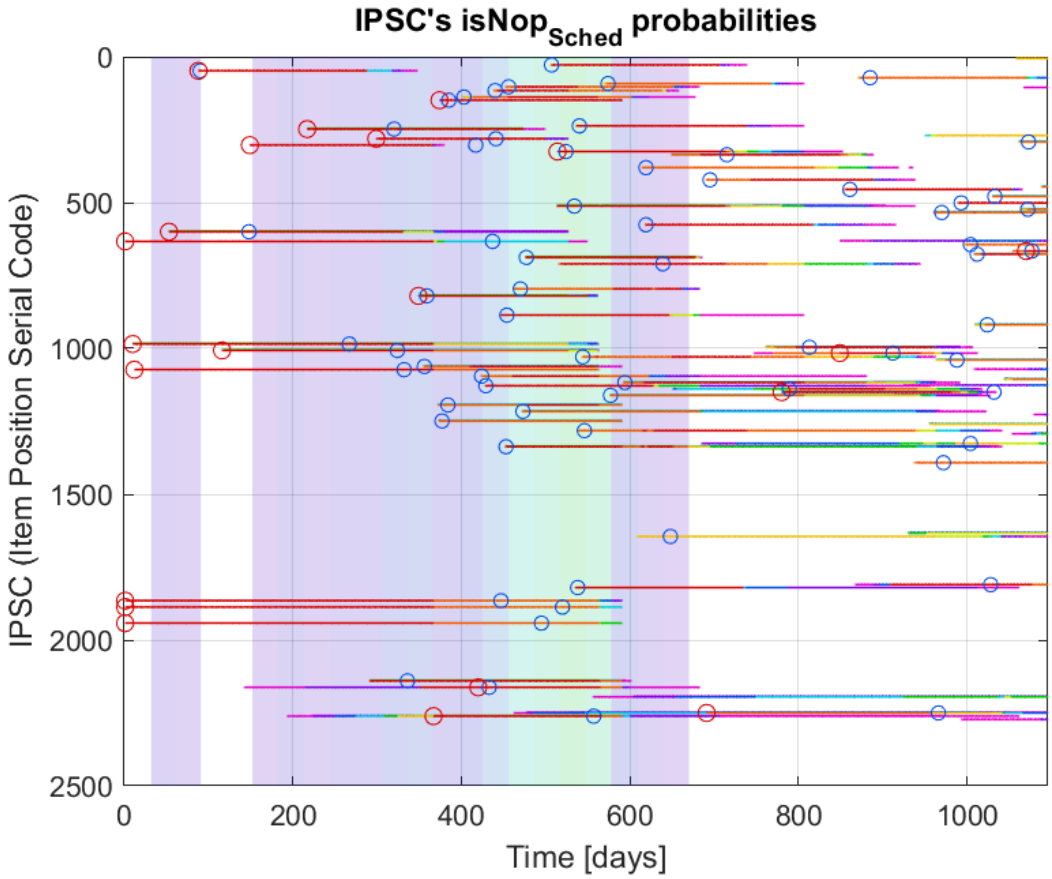


Figura 30) Simulazione ottimizzata n.1381, probabilità di inefficienza, nel tempo, dei componenti per scadenze programmate.

Da queste due immagini, e soprattutto dalla seconda, è possibile vedere la distribuzione di anticipi per cogliere gli aspetti fondamentali della soluzione ricercata dall'algoritmo di ottimizzazione. Si ricorda che i cerchi blu rappresentano il giorno più probabile di rimozione di un certo item nello scenario di partenza, mentre i cerchi rossi indicano il giorno di anticipo desiderato per quello stesso componente.
I risultati riportano che sono stati anticipati 21 componenti (dei 66 suggeriti) per un totale di 4121 giorni di anticipo. Come composizione degli items anticipati, essi si categorizzano come:

- 12 (su 18) di livello 1,
- 4 (su 24) di livello 2,
- 3 (su 15) di livello 3, e
- 2 (su 9) di livello 4.

Un numero preponderante di item ad alta probabilità di rimozione è in sintonia con quanto atteso nonché auspicato.
Inoltre, si nota che l'algoritmo ha anticipato sei items all'inizio della simulazione: a giudicare però dalla lunghezza temporale delle inoperatività che ciò ha comportato (ben più lunga dei 200 giorni di TAT_exch per l'installazione e il testing del modulo), non sembra essere stata una scelta particolarmente proficua. È comunque probabile che lasciando evolvere ulteriormente il calcolo prima o poi l'Ottimizzatore avrebbe aggiustato anche questo comportamento.

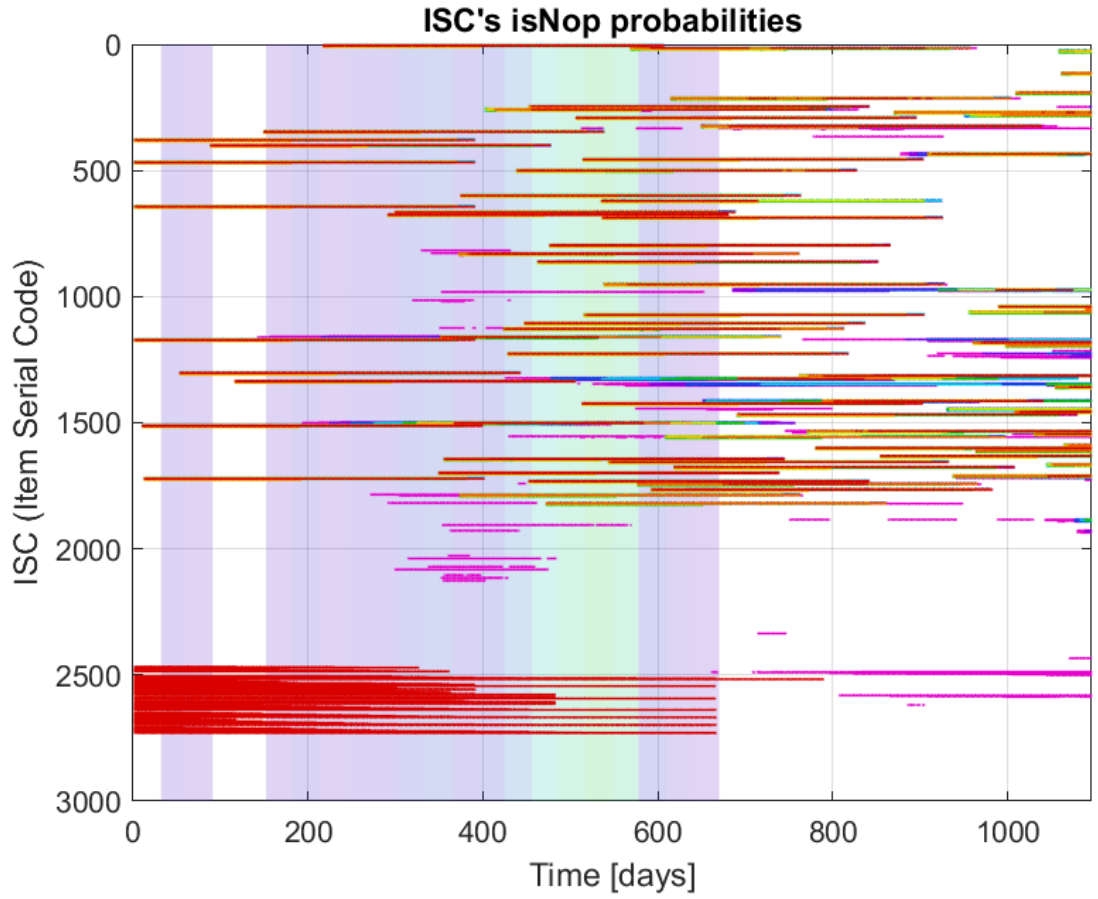


Figura 31) Simulazione ottimizzata n.1381, probabilità di inefficienza dei componenti nel tempo.

Questo grafico può mostrare l'operatività dei componenti sotto un differente punto di vista, anche se in questo caso non si notano effetti particolari.

4.3.3.2 Scelta dello scenario ottimo

Il processo di ottimizzazione, come si è potuto vedere, genera una grande quantità di scenari e se, come in questo caso, le differenze fra di essi non sono particolarmente accentuate, sorge la difficoltà di scelta di quale di essi considerare come "ottimo" e prenderlo come soluzione dell'ottimizzazione.

Ancora una volta, il problema è generato dall'aleatorietà intrinseca nel sistema, che fa oscillare i parametri del costo di ottimizzazione. In linea generale, pertanto, si può affrontare la questione portando le stesse osservazioni già discusse quando si è analizzato il problema stocastico sulle medie della simulazione ottimizzata e non. Se si desidera distinguere due scenari con una certa precisione, si può impiegare un t-test; nel caso esso non andasse a buon fine, con l'aumento del numero di iterazioni si potrebbe sanare la problematica. Ora però, quest'ultima soluzione non è percorribile poiché ci si ritrova ormai a processo concluso, e ogni simulazione ha già il proprio numero di iterazioni. Si potrebbe pensare di riprendere alcuni degli scenari più promettenti e aggiungervi iterazioni, ma questo equivarrebbe a continuare la procedura di ottimizzazione e pertanto la si esclude come possibilità.
Come si vede la scelta dell'ottimo è un problema tutt'altro che semplice, e richiede un'approfondita e ulteriore analisi dedicata solo ad esso. Nel caso qui considerato, però, si è optato per l'impiego di un metodo qualitativo, come esposto di seguito.

Se si trattasse di un problema deterministico, la soluzione ottimale sarebbe quella col minore *cpfh*, ovvero il design n.1381[66]. Introducendo però il criterio che maggiori sono le iterazioni della simulazione e maggiore è la sicurezza dello stesso, verrebbe da scegliere come candidato ideale il design n.958, che ha 1400 iterazioni (600 più del 1381) e un *cpfh* di poco inferiore. Ebbene, se si guardassero solo da questo punto di vista allora si, il secondo dovrebbe essere preferito al primo. Si può però introdurre una nuova osservazione per prendere una decisione: ovvero sfruttare il fatto che le iterazioni singole sono in realtà inserite all'interno di un processo di ottimizzazione, con tanti altri design simili ad essi che sono stati valutati. Se si giudica dunque la bontà della singola simulazione in relazione a quelle nei suoi pressi, si protende invece per scegliere la 1381, poiché si trova in una zona più avanzata del processo di ottimizzazione, la cui media complessiva locale di *cpfh* risulta migliore di quella nei pressi del 958.
È per tale motivo che lo si è scelto come soluzione ottimale, rimanendo comunque consci della necessità di tenere a mente la problematica aleatoria e di ricercare metodi più precisi che, partendo dalle considerazioni fatte, riescano ad approcciare la questione in modo quantitativo.

[66] Si rimanda all'history chart in Figura 24 per tutti i riferimenti ai design citati in questo paragrafo.

4.3.4 Conclusioni sul test case

Nel complesso, si considerano soddisfacenti l'esito delle analisi condotte e le capacità dell'OLSO v.1 di ricercare soluzioni migliorative in termini di availability, anche se dai risultati si è mostrata una difficoltà nella gestione della spesa. Non è semplice l'individuazione delle cause di tali difficoltà, e si ritiene sia stato dovuto a molteplici fattori. Possiamo suddividerli in: complessità di scenario, difficoltà dell'algoritmo di ottimizzazione e di logica della stessa. Indaghiamoli nel dettaglio di seguito.

Complessità di scenario

Sicuramente, un fattore determinante è la complessità dello scenario di partenza. Essa può essere considerata a livello di availability e di costi:

- nel primo caso se non vi sono adeguati margini per anticipare le manutenzioni, saturando la capacità del sistema di riceverle senza incorrere in nuovi momenti di down;
- il secondo caso invece rende difficile l'ottimizzazione per via del costo degli anticipi, causato da sfavorevoli condizioni di prezzo del componente interessato. Riguardo quest'ultimo punto, ad esempio, rivolgendosi specificatamente al test case si è già fatto notare come tra i componenti anticipabili la maggior parte fossero M05, che con la sua LLP ad alto costo orario lo rende poco incline all'anticipo stesso.

L'algoritmo di ottimizzazione

In secondo luogo, l'algoritmo di ottimizzazione può incontrare difficoltà nel ricercare la corretta combinazione di input che migliori lo scenario. Questo è alimentato fortemente dal numero di variabili di ottimizzazione e dalle caratteristiche stocastiche del sistema studiato. Seppure gli algoritmi utilizzati siano preposti a ricercare l'ottimo anche con molti gradi di libertà, l'ulteriore introduzione dell'aleatorietà può giocare un ruolo particolarmente complicato, allungando i tempi di calcolo o conducendo a risultati sub-ottimi. Nel caso specifico del test case, si è visto come la convergenza del metodo fosse blanda: è probabile che concedendole altro tempo sarebbe ulteriormente migliorata, tuttavia il tempo di calcolo stesso è un bene da tenere in attenta considerazione. Potrebbe dunque essere necessario sviluppare dei metodi di ricerca della soluzione che permettano di ottenerle più rapidamente, sfruttando algoritmi che impiegano un maggior numero di informazioni per prendere le proprie decisioni e progredire con l'ottimizzazione. Ad esempio, si potrebbe supportare l'attuale algoritmo valutando i giorni di rimessa in servizio dei componenti.
Comunque sia, l'OLSO è da questo punto di vista piuttosto versatile grazie alla modularità con cui è stato sviluppato: nelle prossime versioni del software, si desiderassero sviluppare diversi e affinati sistemi di ottimizzazione, la loro interazione con il resto del software rimarrebbe pressoché inalterata, rendendo semplice la loro introduzione.

La logica di ottimizzazione

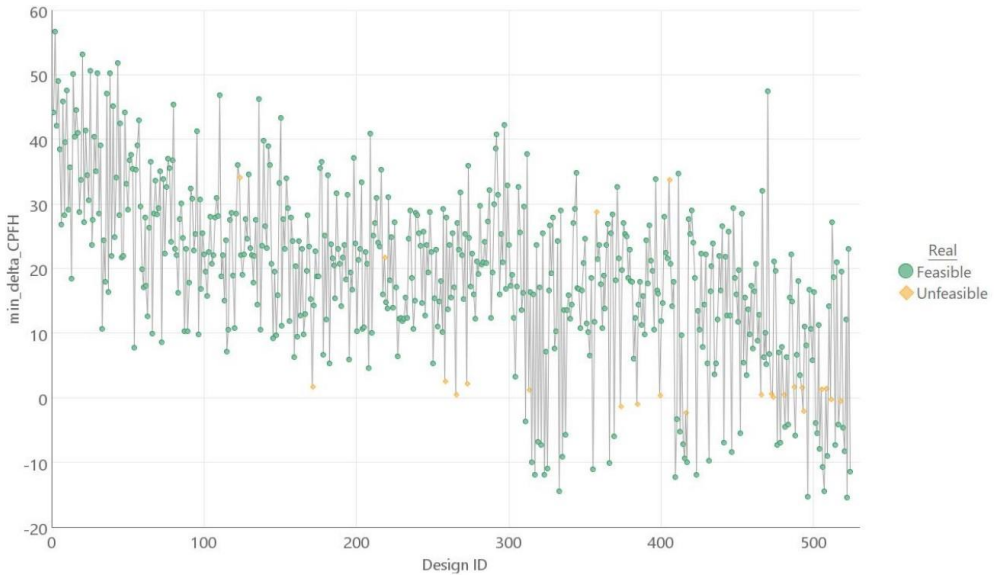
Infine, è da tenere in considerazione anche l'eventualità per cui lo scenario, per le ragioni viste in precedenza nella sezione dedicatagli, non sia sufficientemente adeguato al miglioramento mediante tecnica dell'anticipo. In tal caso bisognerebbe affiancarlo da altri metodi, come la riduzione forzata dei TAT o la richiesta di nuove parti di ricambio. Sebbene attualmente non siano state implementate nell'OLSO v.1, il simulatore è stato comunque pensato per essere parametrico, e quindi versatile, incline a potersi adattare all'introduzione di differenti logiche di ottimizzazione e simularle efficacemente.

4.3.4.1 Comparazione con un secondo scenario di partenza

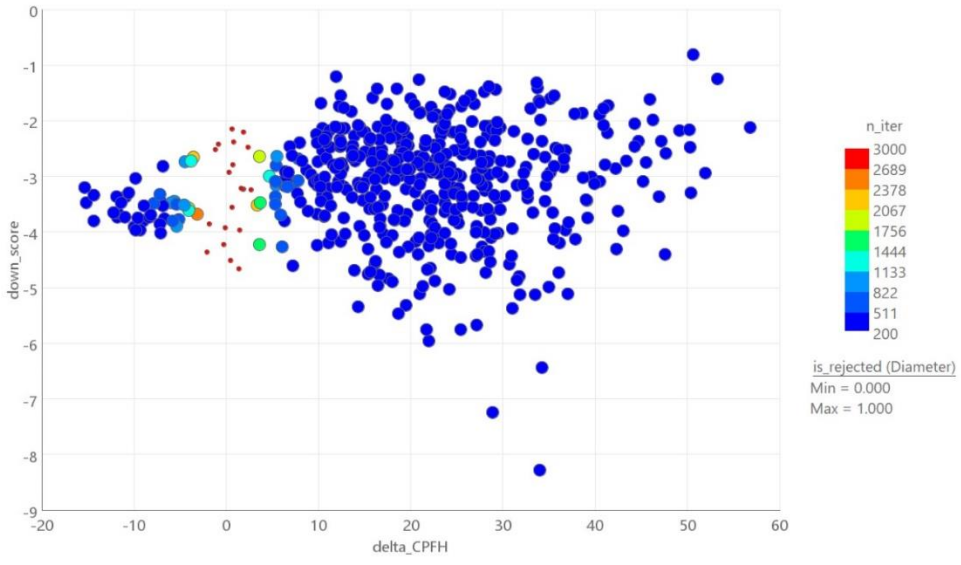
Per concludere l'analisi dei risultati, e in particolare a supporto dei commento circa le difficoltà relative alla complessità di scenario, si vuole mostrare come in effetti al variare delle condizioni iniziali (di flotta, utilizzo, età ecc.) la soluzione ottimizzata riesca ad essere significativamente differente, e in particolare migliore.

Questo secondo scenario considerato parte da una flotta (di velivoli, motori e parti) di dimensione e composizione simile a quella del test case. A cambiare sono i costi delle parti, leggermente inferiori a quelli del test case, ma soprattutto le condizioni iniziali di flotta che presentano, in analogia al test case, un forte momento di down (*downScore* = -5.4) ma più nel futuro; inoltre, l'availability media nel periodo pre-down risulta più alta. Tali condizioni sono più favorevoli all'ottimizzazione mediante anticipi. Per brevità non si mostrano in maniera dettagliata gli input di questo nuovo scenario; si considerano sufficienti le indicazioni appena date.

Si mostrano in Figura 32 i risultati principali dell'ottimizzazione (con algoritmo pilOPT).



(a)



(b)

Figura 32) Risultati di ottimizzazione del secondo scenario:
a) History chart – Andamento dell'ottimizzazione ; b) Bubble chart – Risultati in termini di availability e costi (relativi)

Come si può notare, dopo la tipica fase di assestamento iniziale, l'ottimizzazione riesce a trovare soluzioni migliorative in termini sia di availability che (a differenza del test case, anche) di costi. Inoltre, l'ottimizzazione è stata interrotta a 525 design, ma il chiaro trend che si visualizza lascia prevedere che migliori risultati siano ottenibili se si proseguisse con le analisi.
La porzione di design scartati a cavallo della regione di zero-CPFH sono dovuti al rigetto per scarsa significatività statistica (e più iterazioni non sono state permesse poiché già raggiunto il limite massimo di 3000).
Circa la ricerca dell'ottimo, valgono le medesime considerazioni fatte per il test case: a causa delle oscillazioni del costo, la condizione $delta\_cpfh < 0$ non è sufficiente a poterne concludere l'assoluta certezza di essere un design migliorativo. Bisogna guardare al processo di ottimizzazione nel suo complesso, e calcolare la probabilità di ottenere un certo numero di design migliorativi esclusivamente per via del caso, e concludere la validità dei risultati in base a questo valore. Insomma, alla fine della simulazione bisogna attuare una complicata analisi per la scelta dell'ottimo, che potrebbe non coincidere semplicemente con il design che mostra il *delta_cpfh* minimo.

Capitolo 5 Conclusioni e sviluppi futuri

In conclusione, si ripercorrono brevemente obiettivi e risultati di questo lavoro di tesi.

Obiettivi

Il lavoro è stato portato avanti per raggiungere i seguenti obiettivi.

1) Sviluppo dei software: OLSO v.1
   Si richiedeva lo sviluppo di due software, Simulatore manutentivo e Calcolatore dei costi, sia in termini di modelli teorici (logiche manutentive e di assegnazione del costo, forma e contenuti dei database, ecc.) che implementazione pratica (script eseguibili), con un grado di dettaglio tale da poter compiere le analisi indicate nella descrizione del DL1 (ottimizzazione per anticipo, e analisi di sensitività a TAT e spares). Al loro fianco, è necessario sviluppare anche l'Ottimizzatore, sia in termini di Logiche (scelta delle leve di ottimizzazione, struttura iterativa ecc.) che di implementazione. Il tutto dev'essere poi integrato in un'unica piattaforma di integrazione dove i vari componenti andranno a formare la struttura che, nel complesso, rappresenta l' *Operation-oriented Logistic-Support Optimizer – versione 1* (OLSO v.1).
   L'OLSO così delineato deve infine poter ricevere gli input "dal campo" tramite fogli di lavoro e deve saper sviluppare, se richiesto, una reportistica per l'esportazione dei risultati in formati comuni (in particolare, fogli di lavoro), utili per la consultazione da parte di operatori umani o per il post-processing di altri software.
2) Analisi:
   Si richiede di svolgere una delle analisi di DL1, delineandone i passi sia a livello concettuale che implementativo. Si è scelto di testare le potenzialità e i limiti della metodologia per anticipo.

Risultati

Il lavoro di questa tesi ha portato alla completamento dei suddetti obiettivi.

Sviluppo dei software:

I software di Simulazione manutentiva e Calcolo dei costi sono stati scritti in linguaggio MATLAB ed esportati come script eseguibili (estensione *.exe*). Essi si interfacciano con l'esterno tramite file input (database) e output (reportistica) di tipo worksheet, in particolare sfruttando il formato di Microsoft Excel *.xlsx* . Il loro funzionamento è stato validato da partner di progetto esterni.
Dell'Ottimizzatore è stata sviluppata la logica, applicata in particolare alla metodologia per anticipo. Per la sua implementazione non è stato sviluppato da zero uno script, per via dell'alta complessità dei problemi in questo campo, che avrebbero altrimenti richiesto uno studio approfondito dedicato (sottraendo risorse al resto del progetto). Piuttosto, si è optato per l'impiego di un modulo di ottimizzazione contenuto all'interno di un software commerciale di integrazione di processo; in particolare, una piattaforma SPDM. Al suo interno è possibile scegliere fra alcuni dei principali moderni metodi di ottimizzazione multi-variabile e multi-obiettivo.
Infine, i tre componenti sono stati integrati in una piattaforma SPDM dedicata, la stessa che ospita il modulo Optimizer.

Analisi:

L'analisi finale richiesta, ossia il testing della metodologia per anticipi, è stata compiuta (e affiancata da altre analisi di supporto per problematiche specifiche).
I risultati hanno mostrato che in base alla tipologia scenario la logica può essere efficace e portare ad un miglioramento sia in termini di costi che di availability, tuttavia le caratteristiche di scenario possono rendere più o meno complicata l'ottimizzazione. In particolare, l'analisi del test case ha ricavato soluzioni migliorative in termini di availability a scapito del costo per ora volata; mentre il secondo scenario, più incline all'ottimizzazione mediante anticipi, ha ricavato soluzioni migliorative su entrambi i punti di vista.

Nel complesso:

Nel complesso, si giudicano soddisfacenti i risultati di questo lavoro. In particolare, vanno valutati tenendo a mente che si tratta di un primo stadio di sviluppo di un software più grande e complesso. In ottica di sviluppi futuri, lo step immediatamente successivo è quello di completare l'OLSO v.1 introducendo la capacità di gestire un'ottimizzazione statica anche su TAT e parti di ricambio. Per questo tipo di analisi, il Simulatore manutentivo e il Calcolatore dei costi implementati per questa tesi sono già pronti (poiché sviluppati già per questo scopo); si tratta di modificare l'Ottimizzatore dandogli la capacità di gestire anche variabili legate al numero di parti a magazzino e TAT manutentivi.

Per sviluppi ancora successivi, i cosiddetti Development Level 2 e 3, si richiede la produzione di moduli a sé stanti, Spare Parts Manager e TAT Forecaster, integrandoli nell'OLSO in modo che le loro valutazioni in un certo istante modifichino dinamicamente i parametri in input al Maintenance Planner nel corso della sua ottimizzazione. In particolare essi si occupano di valutare la fattibilità delle operazioni manutentive organizzate dal Maintenance Planner, al contempo cercando di ottenerla tramite l'ottimizzazione di leve interne al proprio dominio di competenza. Il software completo sarà in grado di gestire dinamicamente le problematiche di Supporto logistico, sia attivo (manutenzione diretta) che di supporto, migliorando l'operatività del sistema sotto il duplice fronte di availability e costo operativo.

Capitolo 6 Lista degli acronimi e Glossario

6.1 Lista degli Acronimi

Acronimo	Significato
ADT	Administrative Delay Time
Ant	Anticipo (usato nei nomi, ad esempio di variabile)
afh	Aircraft Flight Hour
ATN	Aircraft Tail Number
CPFH | cpfh	Cost Per Flight Hour (sempre inteso per *ora-velivolo*)
EHM	Engine Health Monitoring
eis	Entry Into Service
ESN	Engine Serial Number
GA	Genetic Algorithm
GP	Gaussian Process
IC	Item Code
imc	Iterazione Monte Carlo (usato nei nomi di variabile)
IPSC	Item Position Serial Code
ISN	Item Serial Number
LDT	Logistic Delay Time
LLP	Life Limited Part
LOF	Limite in Ore di Funzionamento
LRU	Line Replaceable Unit
MANY	Many-Objective Algorithm
MDT	Mean Downtime
MEGO	Multi-objective Efficient Global Optimization
MISL	Minimum Issued Service Life
MOGA	Multi-Objective Genetic Algorithm
MOGT	Multi-Objective Game Theory
MOPSO	Multi-Objective Particle Swarm Optimization
MOSA	Multi-Objective Simulated Annealing
MLx	Maintenance Level X (‘x’ è un numero intero da 1 a 3)
Mnt	Maintenance
MP	Maintenance Plan | Maintenance Planner
MTBF	Mean Time Between Failures
MTBM	Mean Time Between Maintenance
MTBR	Mean Time Between Replacements

MTTF	Mean Time To Failure
notOpt | notOptm	Not optimized (dicasi di “simulation”, nei nomi di variabile o file)
NSGA	Non-dominated Sorting Genetic Algorithm
Opt | Optm	Optimization (e derivati, e.g. “Optimizator”)
PI	Process Integration
Pred	*Prediction (e derivati, e.g. “Predictive”; a volte anche in italiano)*
PSO	Particle Swarm Optimization
Sim	*Simulation (e derivati, e.g. “Simulator”; a volte anche in italiano)*
SN	*Serial Number (riferito ai componenti)*
SPDM	Simulation Process Data Management
SW	Software (usato in schemi, principalmente)
TAT	Turn Around Time
TBO	Time Between Overhauls
TSO	Time Since Overhaul
v. | vers.	Versione (seguita dal numero identificativo)

6.2 Glossario

Soggetto	Spiegazione
Aircraft Tail Number (*ATN*)	Numero di serie o matricola del velivolo.
Algoritmo di ottimizzazione	Qui inteso relativamente all'OLSO v.1, è l'insieme delle logiche con cui migliorare lo scenario manutentivo a livello di disponibilità e costo. Si costruisce attorno alla riduzione dei TAT e al metodo degli anticipi.
Applicativo/Modulo/ Software di Simulazione	Software che simula e costifica lo scenario manutentivo. Formato dal Maintenance Simulator e Cost calculator.
Availability di down / A_{down}	Grandezza calcolata come differenza dell'*availability* meno l'*availability target*, azzerando dove il tutto è maggiore di zero. In pratica: $A_{down} = \min\{ 0 , A - A_{target} \}$
Availability Target / Availability minima	Valore minimo di availability motori mensile al di sotto del quale si incorre in penali.
Confidence interval / Intervallo di confidenza	Rappresenta il range, associato alla variabile (ignota) di cui si vuole stimare il valore, tale per cui si riscontra che il reale valore ricade proprio in tale regione con frequenza pari al *confidence level* con cui l'intervallo è stato calcolato.
delta_	Prefisso usato nei nomi delle variabili e che indica la variazione, della grandezza in questione, nel passaggio dalla simulazione non ottimizzata a quella ottimizzata. In pratica: $delta\_x = x_{opt} - x_{notOpt}$.
Down / Momento di down	Porzione temporale in cui si registra un'availability inferiore al requisito minimo imposto dal contratto, a seguito della cui mancanza si incorre in penali da pagare al cliente.
Down score	Grandezza scalare che identifica la gravità complessiva dello scenario considerato. È calcolato come integrale nel tempo dell'*availability di down* pre-elevata per un certo coefficiente di penalizzazione. Dunque: $down\_score = -\int_0^{T_{sim}} \left[abs\left(A_{down}(t)\right)\right]^{\alpha_{down}} dt$
Engine Serial Number (*ESN*)	Numero di serie o matricola del motore.
Entry Into Service (*eis*)	Giorno di simulazione a partire dal quale il motore/componente entra in servizio ed è in grado di accumulare ore di funzionamento.
History plot	Tipo di grafico in cui lungo l'asse x compare il tempo.
Item Code (*IC*)	Codice numerico che identifica in maniera univoca, *all'interno della simulazione,* un certo *tipo* di item. Unità diverse con stesso Item Code si differenziano per codice seriale (ISC).
Item Position Serial Code (*IPSC*)	Codice numerico con cui si identifica in maniera univoca un certo item su un determinato motore, contando secondo un ordine preciso: prima si dispongono i motori in ordine di codice seriale (ESC), poi per ognuno di essi si considera la sua configurazione di items (IC); si contano i componenti così disposti e il numero assegnatogli è proprio l'IPSC.

Item Serial Number (ISN) / Serial Number (SN)	Numero seriale del componente.
Limite in Ore di Funzionamento (*LOF*)	Valore massimo di ore di volo cumulabili, per un item a scadenza programmata, prima della scadenza in questione. Nel caso di evento per TBO, è il *TBO* stesso; nel caso di LLP, è la vita del componente.
MATLAB	Software di programmazione e calcolo ingegneristico di proprietà dell'azienda MathWorks, che si basa sull'omonimo linguaggio di programmazione (MATLAB, appunto). Tale software è stato utilizzato in questa attività per sviluppare il *Modulo di simulazione*.
Maintenance Level (*ML*)	Codice, da 1 a 3, che identifica le caratteristiche della gestione manutentiva del motore o del componente. *ML1*: operazione di rimozione e sostituzione che può essere eseguita con motore montato sul velivolo; tipica solo degli accessori; *ML2*: rimozione che causa lo sbarco del motore; *ML3*: operazioni di manutenzione per le quali è necessario smontare il modulo nelle diverse parti che o compongono, e.g. per pulizia e riparazioni.
Maintenance Planner / Maintenance Planner software	È uno dei tre moduli dell'OLSO: un software che, compiendo simulazioni e applicando certe logiche, ottimizza il piano manutentivo in termini di availability. Definibile anche come: è l'applicativo che implementa il Maintenance Plan (che invece è una logica, un algoritmo)
Minimum Issued Service Life (*MISL*)	Rappresenta il minimo quantitativo di ore residue (per una parte con scadenza programmata) che devono essere garantite a fronte di una riparazione motore che causa lo sbarco del motore
p-value	Il *p-value* di un test di significatività è la probabilità di ottenimento di un risultato estremo almeno quanto quello osservato, data per vera l'ipotesi nulla. Il risultato è detto *statisticamente significativo* per gli standard dello studio se $p \leq \alpha$, nel qual caso si dice che l'ipotesi nulla (e.g. che due medie misurate siano statisticamente uguali) viene rigettata e assunta vera l'ipotesi alternativa (e.g. che le due medie siano effettivamente diverse).
Penale	Somma di denaro che il gestore della manutenzione dei motori deve pagare al suo cliente in caso di mancato rispetto di uno o più requisiti contrattuali, in questo studio specificatamente relativi alla disponibilità-motori mensile minima da assicurare.
Politecnico di Torino	È uno dei partner di questa attività (A1.4).
Removal Plan	Piano delle rimozioni (schedulate) nel tempo. È l'output principale del Maintenance Planner.
Significance Level / Significatività	Viene definita come la probabilità, per uno studio di testing dell'ipotesi nulla, che quest'ultimo rigetti l'ipotesi nulla nel caso che essa sia però vera. Essa rappresenta dunque il rateo di falsi positivi (anche detti errori di prima specie) e viene indicata convenzionalmente con α.
Simulatore manutentivo /	Programma che implementa le logiche di simulazione manutentiva, che riproduce il comportamento della flotta di aerei, motori e componenti che

Maintenance simulator	operano sottoposti al Maintenance Plan, secondo un certo scheduling dei voli, assoggettati a fenomeni aleatori di usura, failure randomiche e dinamiche di magazzino, e rispettanti determinati vincoli e oneri contrattuali.
Soggetto manutentivo	Qui utilizzato nel senso di "generico ente fra velivoli, motori e items".
Subsidiary Damage / Secondary Inspection Failure	Si tratta di un evento per cui, durante le operazioni di controllo a ML2 (Maintenance Level 2), viene rilevato un danno che necessita la sostituzione di un modulo che non ha causato in maniera diretta lo sbarco del motore. Per questa ragione, il subsidiary damage è anche detto *secondary inspection failure*.
Test Case	Test ufficiale dell'OLSO v.1 applicato ad un caso realistico, volto a valutare la bontà delle logiche di ottimizzazione.
Time Between Overhauls (*TBO*)	Numero massimo di ore di volo cumulabili, da un item che possiede l'omonimo evento schedulato, prima della sua revisione obbligatoria.
Time Since New (*TSN*)	Numero di ore accumulate dalla condizione di "nuovo" dell'oggetto.
Time Since Overhaul (*TSO*)	Numero di ore accumulate dall'ultima revisione dell'oggetto.
Turn Around Time	Tempo per il completamento di un determinato processo o soddisfacimento di una richiesta. In tal caso, ad esempio, il tempo impiegato per riportare operativo un item mandato in riparazione, per completare una procedura di installazione motore sul velivolo o revisioni varie, o ancora l'attesa dopo un acquisto di una parte
Workflow	Flusso di lavoro che sintetizza l'insieme dei *flussi di dati* e *processi* (e le loro relazioni) che nell'insieme permettono di svolgere determinati compiti. Spesso usato con l'accezione particolare in riferimento agli omonimi modelli (e associati schemi visivi) sviluppati nella piattaforma SPDM.

Capitolo 7 Bibliografia

[1] J. a. U. o. K. Roskam, Airplane Design: Part 8 - airplane cost estimation: design, development, manufacturing and operating, DARcorporation, 1985.

[2] M. Fioriti, V. Vercella e N. Viola, «Cost-Estimating Model for Aircraft Maintenance,» *JOURNAL OF AIRCRAFT,* 2018.

[3] «FY2013 Maintenance Cost Preliminary Analysis,» IATA, Montreal, 2014.

[4] E. Hunt, «Fast Jet Cost per Flight Hour Assessment,» IHS Jane's, 2012.

[5] «Cos'è l'Industria 4.0?,» https://www.ibm.com/it-it/topics/industry-4-0.

[6] «Design to Cost and Life Cycle Cost,» in *Flight Mechanics Panel Symposium*, Amsterdam, Netherlands, 22 May 1980.

[7] «RCM Guide: Reliability-Centered Maintenance Guide for Facilities and Collateral Equipment,» NASA, 2008.

[8] A. Birolini, Reliability Engineering, Theory and Practice, 8 a cura di, Springer.

[9] D. H. G. M. D.C. Brauer, *Depot Maintenance Handbook,* Defense Technical Information Center, 1985.

[10] Honeywell.com, «Three Ways High Velocity Maintenance are Changing MRO,» [Online]. Available: https://aerospace.honeywell.com/us/en/about-us/blogs/three-ways-high-velocity-maintenance-are-changing-mro.

[11] B. Kobren, «Integrated Logistic Support (ILS) and Integrated Product Support (IPS) Elements - A study in contrasts,» https://www.dau.edu/blogs/integrated-logistics-support-ils-and-integrated-product-support-ips-elements-study-contrasts, 2014.

[12] A. Maron, *Integrated Logistic Support: Applicazione e ottimizzazione delle metodologie ISDA e FRACAS in ambito aeronautico,* Politecnico di Torino, 2019/20.

[13] M. Scully, «High Velocity Maintenance,» Air&Space Forces Magazine, 2009.

[14] K. Willcox, *16.885 Aircraft Systems Engineering Cost Analysis,* MIT Aerospace Computational Design Laboratory, 2004.