Capitolo 6

Testing dell'applicazione

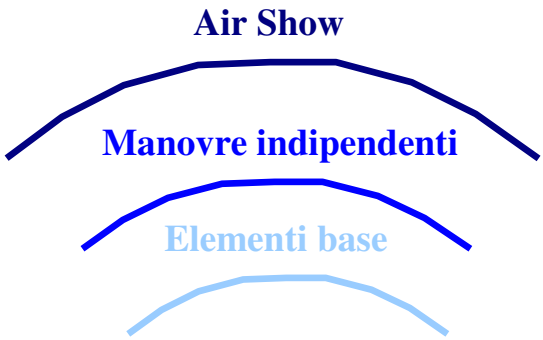
6.1 Introduzione

La fase di test costituisce un aspetto fondamentale della progettazione del software, infatti procede di pari passo con l'implementazione dell'applicazione. Ovviamente la documentazione prodotta per la fase di test è risultata a volte informale e non verrà riportata, per cui lo scopo di questo capitolo è illustrare quali sono state le metodologie di test adottate.

La fase di test è stata ovviamente facilitata dall'implementazione modulare dell'applicazione,che mi ha permesso di concentrarmi di volta in volta sul test di un singolo modulo. Ho cercato per ovvi motivi di tempo di limitare l'implementazione di stubs e drivers sfruttando dove possibile la linea di comando di MATLAB o la definizione parziale dei moduli di livello superiore.

6.2 Progetto di test

L'applicazione realizzata presenta 3 livelli di astrazione principali:



Il primo modulo ad essere implementato è stato quello relativo alla costruzione delle manovre indipendenti, che in fase iniziale ha funzionato da *driver* per l'esecuzione dei moduli che costituiscono il livello più basso d'astrazione.

Inizialmente le funzionalità del modulo relativo al livello intermedio di astrazione erano le seguenti:

- Aggiungere un Line block
- Aggiungere un Circle block
- Aggiungere un Follow block

Tale strategia implementativa mi ha permesso di testare la procedura di costruzione della sequenza di blocchi elementari senza dover implementare gli stub relativi ai moduli di livello inferiore *(line block,circle block,follow block).*

Il modulo relativo al follow block è stato testato considerando come traiettoria da inseguire la manovra indipendente stessa e verificandone l'output su un riferimento cartesiano creato su una window temporanea.

All'implementazione del livello più basso di astrazione è seguita l'implementazione delle altre funzionalità del modulo di astrazione intermedia, per la parte relativa alla sola manovra, escludendo quindi quelle atte a manipolare la struttura dati *'AirShow'*.

Nell'ultima fase dell'implementazione è stato realizzato il modulo di più alto livello, per il quale sono state definite tutte le funzionalità, per testare le quali era necessario avere definiti sia il livello intermedio sia il livello più alto dell'astrazione.

Il test sulla correttezza dell'interfaccia tra moduli di un livello e quelli del livello immediatamente superiore, è stato eseguito in parte durante l'implementazione e in parte nella fase finale del progetto.

La fase di test è stata facilitata dalla possibilità di riportare nel *command window* di MATLAB i risultati intermedi relativi alla manipolazione di strutture dati e all'esecuzione delle procedure implementate, permettendo di valutare il flusso di esecuzione in risposta ai comandi operati.

6.3 Criteri di test

Il test dei moduli, divisi per livello di astrazione, è stato condotto secondo i seguenti criteri:

- Black – Box Testing
- White – Box Testing

Il criterio Black-box testing permette di testare le funzionalità del software ignorando il flusso di manipolazione delle strutture dati e delle variabili utilizzate. Il white-box testing permette invece di analizzare le strutture dati interne e la copertura del codice testato.

La verifica di un modulo secondo la tecnica del black-box testing è consistita in:

- Individuare i valori di test a partire dalla descrizione funzionale
- Analizzare le relazioni esistenti tra valori di ingresso e risultati
- Verifica della consistenza dei risultati con quelli attesi

Gli errori tipici rilevati con questa tecnica sono stati:

- Funzionalità non corrette
- Errori di interfaccia
  - Passaggio dei parametri
  - Non completezza dell'output

La tecnica del black-box testing è stata facilitata dalla possibilità di lanciare opportuni messaggi di inconsistenza atti ad indicare errori dovuti alla mancanza o alla non completezza dei parametri di ingresso/uscita ad un modulo.

In particolare l'uso dell'istruzione

```
try
……..<< codice del modulo da testare>>
catch
……<< messaggio di errore generico -> sintassi o inconsistenza delle
        strutture dati>>
end
```

è stato prezioso in quanto ha permesso di individuare più agevolmente il punto in cui si è verificato l'errore, non pregiudicando la possibilità di proseguire il test anche di altre funzionalità nella stessa sessione, comportando così notevole risparmio di tempo.

La tecnica del white-box testing è stata utilizzata per testare:

- La corrispondenza tra input e modifica delle strutture dati
- La copertura del codice
- La predisposizione degli opportuni messaggi all'utente in conseguenza delle sue azioni

La corrispondenza tra input e modifica delle strutture dati è stata facilitata dall'uso del command window di MATLAB per analizzare lo stato intermedio delle strutture dati durante l'esecuzione e rilevare facilmente le inconsistenze.

La copertura del codice è stata valutata predisponendo, lungo il flusso dell'esecuzione, dei messaggi di avvenuta esecuzione della parte di codice a cui il messaggio si riferisce, ed anche in questo caso ho sfruttato il *command window* MATLAB.

Con questa tecnica di testing gli errori più frequentemente rilevati sono stati

- Errori di dimensionamento *(array, campi delle strutture dati …)*
- Errori di inizializzazione
- Errori nelle condizioni di uscita dai cicli
- Errori nella scansione dei vettori