TestStand & Requirement Gateway: un approccio integrato di Software Engineering nei sistemi di test automatici

L. Magni - PRAGMA ENGINEERING

LA SFIDA

Realizzare un sistema test di nuova generazione (NxGen ATS) applicando i principi di software engineering nell'intero ciclo di progettazione, sviluppo, rilascio e validazione dell'architettura software del sistema.

LA SOLUZIONE

Effettuare scelte architetturali hardware e software secondo l'approccio NxGen ATS impiegando, per quanto concerne il software, un test manager quale NI TestStand, ambienti ADE quali NI LabVIEW e NI LabWindows/CVI ed un tool quale NI Requirements Gateway per verificare e tracciare i requisiti di test del sistema.

Breve riassunto

La realizzazione di un sistema di test di nuova generazione (NxGen ATS) richiede l'impiego di appropriati strumenti di sviluppo al fine di soddisfare i requisiti architetturali di base e di consentire la corretta implementazione del processo di software engineering dell'intero sistema.

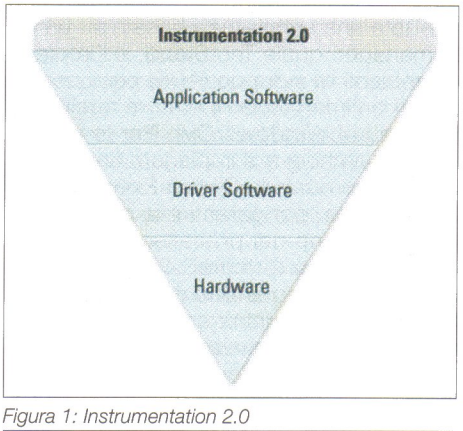

Figura 1: Instrumentation 2.0

L'adozione di TestStand quale test manager consente di agevolare la corretta applicazione del processo di software engineering massimizzandone i vantaggi ed i risultati, inoltre l'impiego del Requirements Gateway come tool per la verifica e la tracciabilità dei requisiti consente di integrare nell'intero processo la fase di verifica e validazione del sistema.

Articolo

La realizzazione di un sistema di test deve necessariamente partire da una accurata definizione architetturale sia in termini hardware che software, che deve consentire sia la rispondenza ai requisiti elettrici/funzionali del dispositivo da collaudare che il soddisfacimento dei goal tecnico/economici di tipo progettuale. In particolare, per i sistemi di test Nuova Generazione (NxGen ATS) l'architettura generale del sistema si basa su quanto definito nell'approccio "Instrumentation 2.0" (Figura 1) dove il ruolo cardine ai fini dell'implementazione del sistema è svolto dal software più che dalla strumentazione. Questo tipo di approccio oltre che a garantire il soddisfacimento delle necessità funzionali dei moderni sistemi di test (misure definibili dall'utente, elaborazione dei dati in real time, interfacce MMI custom, etc.) deve consentire la rispondenza alle esigenze dell'ingegneria di test quali: flessibilità, prestazioni, qualità di misura, manutenibilità, espandibilità, riuso, costo etc. Infine, da una recente analisi relativa ai costi di realizzazione dei sistemi di test (Figura 2) è emerso che almeno il 30% dei costi sono imputabili al processo di sviluppo del software di test.

Le scelte architetturali, le esigenze e le considerazioni che ne derivano devono portare ad un accurata progettazione del software e del suo sviluppo (software engineering) definendo in primis gli strumenti più efficaci che consentono la realizzazione del sistema rispettando i requisiti del collaudo ed attenendosi alle scelte progettuali e di sviluppo riducendo l'impatto economico dello stesso.

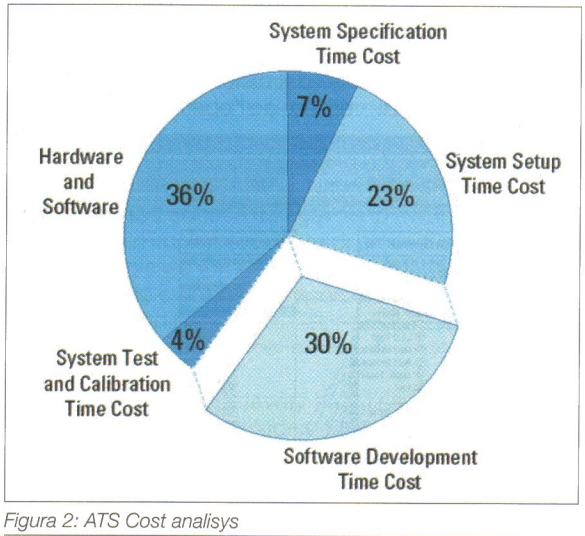

Figura 2: ATS Cost analisys

Questo approccio sistematico e rigoroso (spesso sottovalutato o disatteso), sebbene richieda uno sforzo metodologico e progettuale iniziale consistente (e quindi un costo),

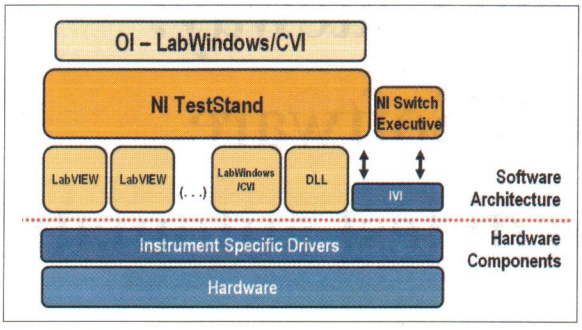

Figura 3: Architecture

porta a degli indubbi vantaggi implementativi e realizzativi che conducono al pieno soddisfacimento non solo dei requisiti del test ma anche delle esigenze ingegneristiche prima citate e ciò è tanto più rilevante quanto maggiore è la complessità del sistema da realizzare specie in ambito aerospaziale e militare dove i sistemi di test hanno tempi di vita molto lunghi e spesso necessità ricorrenti di update a seguito di revisioni del DUT.

La realizzazione, caso di studio del presente articolo, consiste nello sviluppo di un ATS per un dispositivo avionico. In sintesi i requisiti più rilevanti ai fini del test prevedono:

- Comunicazione con il DUT via CAN bus (comandi e monitoraggio dello stato)
- Gestione di comandi discreti verso il DUT
- Stimolazione analogica (inerente i segnali provenienti da sensori esterni al fine di simulare differenti condizioni del veivolo)
- Acquisizione analogica di più canali simultanei (per la verifica delle attuazioni imposte dal DUT)
- Commutazione di linee di segnali discreti e/o analogici (per simulare condizioni critiche sui segnali e verificare il comportamento del DUT)
- Esecuzione del ciclo di Burn-In del DUT (ESS di verifica dell'operatività del DUT)

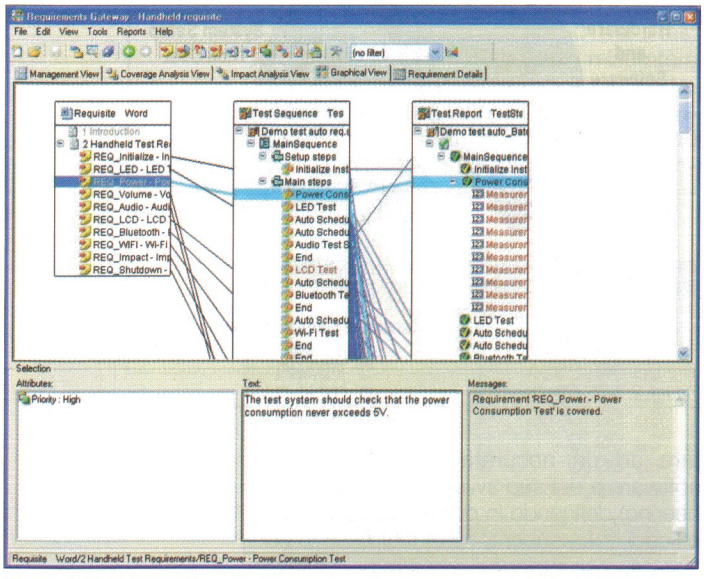

Figura 4: Requirement

Architettura COTS

La scelta realizzativa del sistema si è rivolta sia in termini hardware che software a componenti COTS (Commercial Off-The-Shelf) riducendo al minimo l'impatto custom hw

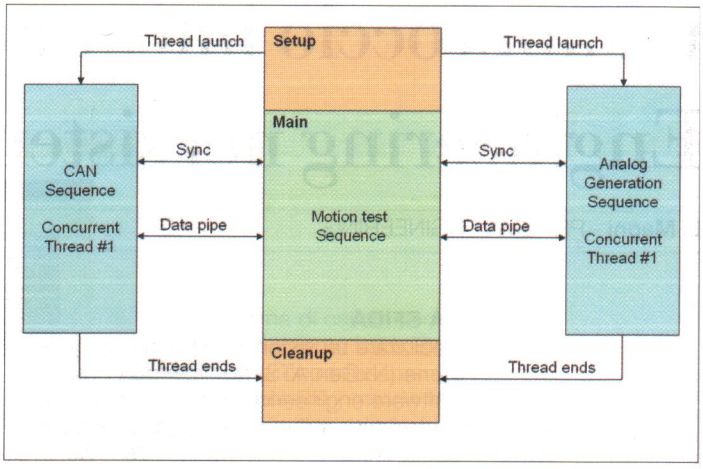

Figura 5: Cuncurrent threads

(JIG di interfaccia per lo sbroglio delle linee DUT), spesso difficilmente riproducibile, garantendo intrinsecamente sia per l'hw che per il sw quei fattori di prestazioni, efficacia, longevità, manutenibilità, espandibilità ed aggiornabilità del sistema precedentemente menzionati (Figura 3).

Per quanto concerne l'architettura hardware questa è stata imperniata su un sistema PXI dotato delle schede necessarie (CAN, I/O simultaneo di segnali analogici e switching) a soddisfare tutte le necessità di interfacciamento e gran parte delle necessità di misura (specie se critiche in termini prestazionali e di accuratezza) affiancato da un minimo di strumentazione tradizionale (PSU e DMM).

Per quanto concerne il software si è adottata una architettura basata su un test manager, quale TestStand, e l'impiego di ambienti di sviluppo (ADE) per il codice di test e l'interfaccia operatore quali LabVIEW e LabWindows/CVI. Per le necessità di switching si è impiegato uno specifico engine quale lo Switch Executive (NiSE) ed infine, per garantire la corretta implementazione del processo di sviluppo del sistema, si è ricorsi all'utilizzo di un tool quale il Requirements Gateway per tracciare ed esaminare la copertura dei requisiti dell'intero sistema che ha consentito di soddisfare a pieno il processo di verifica e validazione di quanto realizzato (Figura 4).

Implementazione

Lo sviluppo software del banco è stato eseguito interamente ed in maniera nativa su TestStand ovvero concentrando tutto il flusso all'interno dello stesso minimizzando lo sviluppo di codice (negli ADE scelti) e soprattutto atomizzandolo il più possibile in modo da poter sostenere l'integrazione del codice unicamente nelle sequenze di test implementate su TestStand.

In tal modo si sono potuti massimizzare i benefici esposti dalle caratteristiche di TestStand e del suo framework, in particolare in termini di:

> Modello di processo sequenziale customizzato.

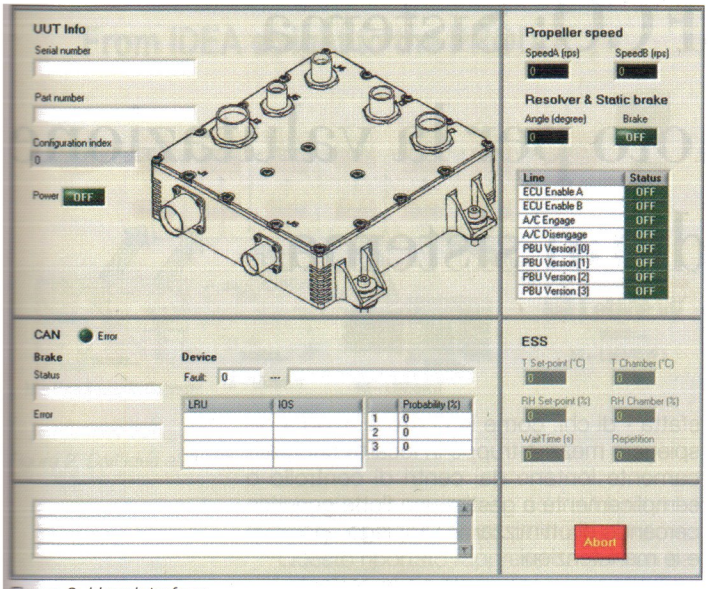

Figura 6: User interface

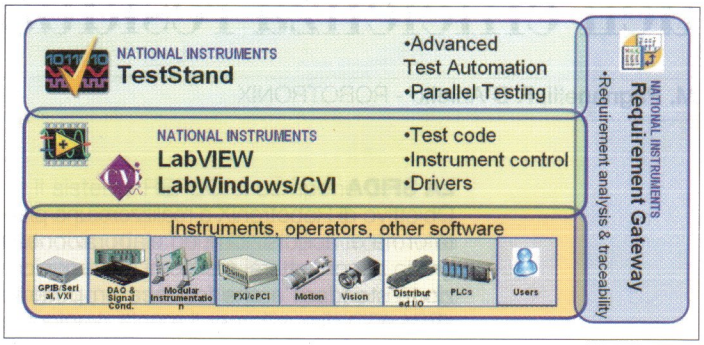

Figura 7: ATS Implementation

> Esecuzione concorrente di Thread multipli (Figure 5 e 6) in particolare per i task di:
- CAN management
- Analog generation
- Signal Acquisition
- User interface.

> Reporting customizzato in XML.
> Database management.
> Documentazione delle sequenze.

L'impiego di LabVIEW e Labwindows/CVI ha consentito di assolvere a tutte le necessità di acquisizione, elaborazione ed user interface richieste dal sistema in maniera efficace in termini di tempo e di prestazioni ottenibili. Inoltre la loro presenza "atomica" ne ha consentito un rapido debug prima della loro integrazione nelle sequenze di test accelerando la fase di validazione del sistema.

L'impiego della tecnologia IVI ha consentito di gestire la strumentazione tradizionale in maniera nativa su TestStand senza ricorrere allo sviluppo di codice dedicato ed assicurando l'intercambiabilità futura di quei componenti senza ricorrere ad alcuna modifica del codice.

L'impiego di Switch Executive ha permesso di "incapsulare" la gestione e la configurazione degli apparati di commutazione dentro il MAX (Measurement & Automation Explorer) svincolandola dal codice e rendendola accessibile in maniera nativa sempre da TestStand. In tal modo una variazione dell'hardware di switch o della sua configurazione non richiede modifiche al codice ma eventualmente "impatta" unicamente sulla sequenza.

Tramite gli adapter di TestStand inoltre è stato possibile integrare DLL preesistenti necessarie al fine della comunicazione con la camera climatica rendendone quindi possibile il controllo e l'integrazione della gestione del ciclo termico internamente alla sequenza di test.

Il tool del Requirement Gateway ha consentito di verificare i requisiti di seguirne la loro implementazione nelle sequenze di TestStand e nei report generati dallo stesso. Il suo impiego ha consentito di verificare la progressione delle attività in base all'indice di copertura dei requisiti e di fornire un report completo relativo alla loro tracciabilità nel codice (Fig. 7).

Conclusioni

Le scelte architetturali basate su hardware e software COTS di National Instruments hanno consentito una applicazione "virtuosa" delle metodologie di progettazione, sviluppo e realizzazione del sistema di test rendendo possibile ottenere i migliori vantaggi offerti da questa scelta architetturale con il minimo sforzo e quindi con un costo contenuto. L'approccio di software engineering è stato intrinsecamente supportato e coadiuvato (se non talvolta sostenuto) dall'adozione della stessa architettura software. In particolare l'utilizzo di TestStand come engine del sistema ha portato (seguendo le opportune metodologie di sviluppo) a sistematizzare l'implementazione delle procedure di test.

L'utilizzo integrato di TestStand combinato con IVI, NiSE, e gli ADE di National Instruments, ha reso efficace ed agevole lo sviluppo anche nell'implementazione di procedure di test complesse (processi concorrenti) minimizzando lo sviluppo del codice e massimizzando l'indipendenza del codice dall'hardware sottostante (per ciò che è gestito via IVI e NiSE).

Il Requirements Gateway ha infine consentito di integrare e chiudere il ciclo di sviluppo con una appropriata fase di verifica e validazione che ha dato evidenza sia dell'attività in itinere sia dello sviluppo finale documentando (matrice di tracciabilità) l'implementazione stessa dei requisiti.

Prodotti utilizzati
LabVIEW, LabWindows/CVI, PXI/CompactPCI, TestStand, Industrial Communication, Data Acquisition, Serial, Switch Executive, Requirement Gateway, Switches