Creazione ed analisi di una metodologia per la creazione di interfacce utente per Sistemi Embedded di fascia bassa

Enrico Giordano, *Studente Magistrale,UniVR, VR386687*

***Abstract*—La progettazione di interfacce utente per Sistemi Embedded ha avuto un notevole sviluppo per renderle sempre pi gradevoli ed efficienti, avvicinandosi sempre pi ai concetti di portabilit e mantenibilit tipica dei sistemi General Purpoise. Si tende sempre pi ad investire su processori di fascia media per poter utilizzare tecnologie tipicamente Client-Server, in modo da poter trattare le applicazioni basate su grafica in maniera pi modulare possibile in accordo con le ultime tecnologie, soprattutto usando la rete e protocolli di rete standard, come HTTP. Questo per risulta problematico in tanti sistemi di fascia bassa, in cui l'ottimizzazione essenziale per la corretta progettazione, ma anche per i vincoli non funzionali da rispettare, rendendo scorrette le tecnologie attualmente utilizzate. Si vuole quindi proporre un metodo per la progettazione di questi sistemi, con un framework appositamente creato per analizzare le performance e la velocit di progettazione.**

I. Introduzione

Esistono due categorie di Sistemi Embedded in commercio:

- *dispositivi ciechi*, che non sono dotati di schermo e non permettono quindi di presentare un'interfaccia;
- *dispositivi visuali*, che sono dotati di schermo e permettono di presentare un'interfaccia.

Nonostante ci, esistono varie tecniche per presentare un'interfaccia utente per comandare il dispositivo o semplicemente presentare lo stato della macchina. Il metodo pi antico e tipico dei sistemi embedded consiste nel creare un terminale testuale comandato via comandi seriali a caratteri; un esempio pu essere il terminale comandato via UART. Con la definizione di protocolli di rete, si sentita la necessit di utilizzarli per garantire robustezza e sicurezza, oltre ad avere la possibilit di avere un'interfaccia multiutente e facilmente portabile ed implementabile. Ultimamente quindi utilizzare un'interfaccia di rete con protocollo standard come HTTP uno standard *de facto*.

Il problema per sorge quando si hanno risorse talmente limitate da non permettere l'utilizzo di alcune tecnologie tipicamente moderne, come la programmazione ad oggetti o stack di rete completi; quindi nasce la necessit di avere delle implementazioni pi ottimizzate e semplici, sia per dispositivi ciechi, sia per dispositivi visuali.

A. Sistemi Embedded di fascia bassa

Dal momento che un'azienda sceglie di utilizzare un tipo di Sistema Embedded di fascia bassa per motivi ragionevoli, cio i costi di produzione, il progettista embedded deve tenere conto di vari vincoli, tra cui:

- *vincoli realtime*, ovvero il sistema deve rispondere entro un certo limite di tempo;
- *vincoli di memoria*, ovvero non eccedere con il codice compilato dal punto di vista di spazio occupato;
- *vincoli di consumo*, ovvero non si deve consumare troppa energia per eseguire un'operazione.

Rispettare questi vincoli molto difficile, quindi solitamente si spende molto tempo e quindi risorse per progettare una *business logic* che implementa la logica del circuito, ma ancora pi tedioso progettare un'interfaccia grafica per comandarla e controllarla.

Come caso di studio di questo progetto, stato scelto un processore di fascia bassa della NXP, ovvero LPC1788, che permette di essere sia dispositivo cieco, sia dispositivo visuale, grazie la sua GPU integrata per gestire un dispositivo come uno schermo LCD Touch.

B. Librerie utilizzate

Poich i Sistemi Embedded da commercializzare necessitano di progettazione abbastanza veloce, in accordo con le regole del *time to market*, varie aziende mettono a disposizione diverse librerie commerciali ottimizzate per gestire al meglio le interfacce o, pi in generale, la progettazione stessa del sistema. In generale, per progettare un Sistema Embedded, necessario conoscere nella sua interezza il microprocessore, per vengono fornite delle API per astrarre il pi possibile il device.

Per la progettazione di interfacce per sistemi di fascia bassa, commercialmente si utilizza una libreria grafica chiamata EmWin, offerta dalla SEGGER. Questa si basa sul paradigma di programmazione ad eventi, in modo da evitare di utilizzare necessariamente un sistema operativo. La progettazione pu avvenire in due modalit diverse:

- *hard coded*, ovvero scrittura completamente manuale sia della grafica sia della business logic associata;
- *soft coded*, ovvero disegnando l'interfaccia il "GuiBuilder", programma in dotazione, per poi scrivere manualmente le funzioni associate all'interazione con i widget.

Per la progettazione della parte di rete, esiste una libreria leggera e ottimizzata che implementa il protocollo TCP/IP, ovvero LWIP. Questo si propone come standard per i disposivi di fascia bassa che necessitano di rete ma non hanno

abbastanza risorse disponibili per ospitare l'intero stack di rete TCP/IP.

Per la progettazione rapida di Sistemi Embedded, esiste un framework chiamato uEZ (si pronuncia come "Muse" in Inglese) che offre un'astrazione dell'hardware per progettare ad alto livello il sistema.

Per queste librerie necessario implementare il porting dell'architettura desiderata, per poi avere un codice "portabile" per tutti i sistemi che utilizzano le stesse librerie per funzionare. Inoltre sono altamente ottimizzate per non sprecare risorse sia di memoria sia energetiche, in modo da progettare un sistema commercializzabile.

II. Il framework per il caso di studio

Poich si vuole dare una metodologia di sviluppo efficiente per la creazione di sistemi con interfaccia utente, necessario l'utilizzo di un framework. Questo stato creato a partire

III. Analisi della metodologia

IV. Analisi delle risorse utilizzate

Appendix A
Proof of the First Zonklar Equation

Some text for the appendix.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.