Ingegneria del Software

Progettazione architetturale (II)

Antonino Staiano
e-mail: **antonino.staiano@uniparthenope.it**

Stili di controllo (I)

- I modelli per strutturare un sistema si occupano della sua scomposizione in sottosistemi
- Perché ciascun sottosistema si integri in un tutt'uno per funzionare correttamente come un sistema è necessario che esso sia controllato
  - I suoi servizi devono essere forniti al posto giusto ed al momento giusto
- I progettisti dovrebbero organizzare i sottosistemi secondo un modello di controllo che integri il modello strutturale usato

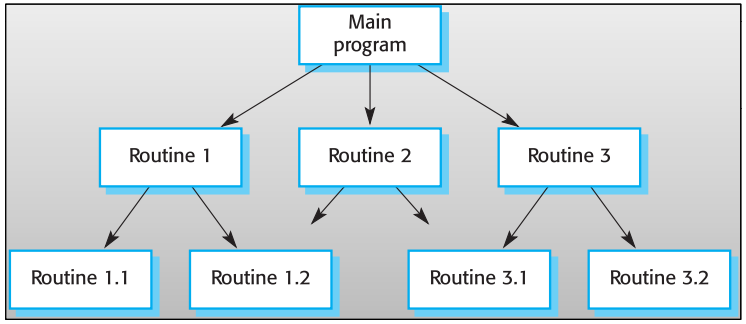
Stili di controllo (II)

- Riguardano il flusso di controllo tra i sottosistemi
  - Gli stili di controllo sono usati in congiunzione con gli stili strutturali
- Sono usati in genere due stili di controllo
  - Controllo centralizzato
    - Un sottosistema ha una responsabilità completa per il controllo e l'avvio e l'arresto di altri sottosistemi
  - Controllo basato su eventi
    - Ogni sottosistema può rispondere ad eventi generati esternamente da altri sottosistemi o dall'ambiente del sistema

Controllo centralizzato

- Un sottosistema di controllo assume la responsabilità per la gestione dell'esecuzione degli altri sottosistemi
- Ricadono in due classi a seconda che i sottosistemi controllati siano eseguiti sequenzialmente o in parallelo
  - Modello call-return
    - Modello di subroutine top-down dove il controllo parte in cima alla gerarchia delle subroutine e si sposta verso il basso. Applicabile ai sistemi sequenziali
  - Modello manager
    - Applicabile ai sistemi concorrenti. Un componente del sistema controlla l'arresto, l'avvio e la coordinazione degli altri processi del sistema. Un processo è un sottosistema che può essere eseguito in parallelo ad altri processi. Può essere implementato nei sistemi sequenziali come una istruzione case in cui una routine di gestione richiama particolari sottosistemi a secondo dei valori di alcune variabili di stato

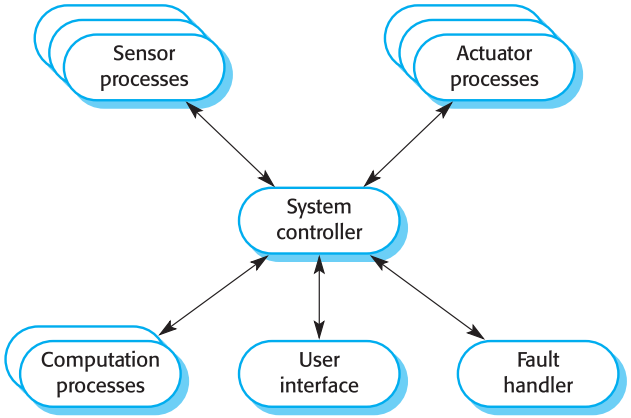
Modello call-return



Modello manager

- E' possibile avere un modello di controllo a gestione centralizzata per un sistema concorrente
  - Utilizzato in semplici sistemi real-time che non hanno vincoli di tempo molto stretti
    - Il controller centrale deve gestire l'esecuzione di un insieme di processi associati a sensori ed attuatori
    - Il processo di controllo decide quando i processi devono partire o fermarsi a seconda delle variabili di stato del sistema
      - Verifica se altri processi stanno producendo informazioni da elaborare o se hanno bisogno di informazioni per farlo
      - Poiché il controllo itera continuamente interrogando i sensori e gli altri processi per eventi o cambi di stato, il modello è chiamato modello a ciclo di eventi

Modello di controllo centralizzato per un sistema real-time



Sistemi guidati da eventi

- Guidati da eventi generati esternamente dove il timing degli eventi è fuori il controllo dei sottosistemi che elaborano l'evento
- Per evento si intende un segnale che può cambiare in una gamma di valori o anche un comando selezionato da un menu
  - La distinzione tra un evento e un input semplice è che la temporizzazione dell'evento è completamente fuori dal controllo del processo che lo gestisce

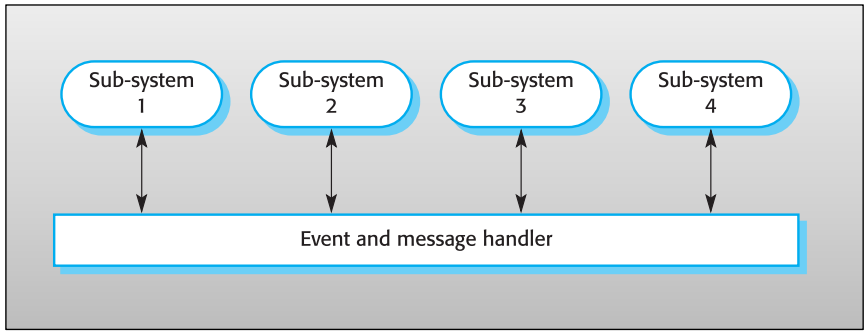
Sistemi guidati da eventi

- Due modelli principali guidati da eventi
  - *Modelli broadcast*. Un evento è inviato in broadcast a tutti i sottosistemi. Qualsiasi sottosistema che può gestire l'evento può rispondere
  - *Modelli guidati da interruzioni*. Usati nei sistemi real-time dove gli interrupt sono individuati da un gestore e passati a qualche altro componente per l'elaborazione
- Altri modelli guidati da eventi includono sistemi di produzione (usati in AI, ad esempio)

Modello Broadcast

- Efficaci nell'integrare sottosistemi distribuiti su diversi computer in una rete
- I sottosistemi segnalano un interesse a specifici eventi e quando questi si verificano il controllo viene trasferito al sottosistema che può gestirli
- Rispetto al modello centralizzato, la politica di controllo non è integrata nel gestore degli eventi e dei messaggi
  - I sottosistemi decidono di quali eventi hanno bisogno ed il gestore si assicura che questi siano loro inviati
  - Tutti gli eventi possono essere trasmessi a tutti i sottosistemi ciò implica un grosso overhead di elaborazione
    - Il gestore conserva un registro dei sottosistemi e degli eventi a cui sono interessati. I sottosistemi generano eventi, il gestore individua gli eventi, consulta il registro e li invia ai sottosistemi che hanno dichiarato di esserne interessati

Broadcasting selettivo



Modello Broadcast: vantaggi

- L'evoluzione è semplice
  - Un nuovo sottosistema per la gestione di particolari classi di eventi può essere integrato attraverso la registrazione dei propri eventi con il gestore
  - Ogni sottosistema può attivare qualsiasi altro sottosistema senza conoscerne il nome o la posizione
    - I sottosistemi possono essere implementati su macchine distribuite in modo trasparente per gli altri sottosistemi

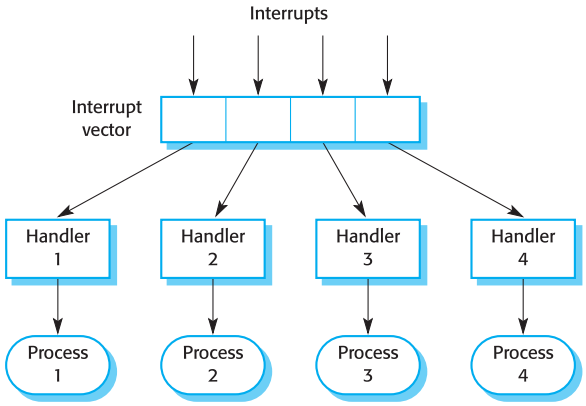
Modello Broadcast: svantaggi

- I sottosistemi non sanno se e quando gli eventi saranno gestiti
  - Quando un sottosistema genera un evento non sa quali altri sottosistemi hanno registrato un interesse per tale evento ed è possibile che diversi sottosistemi si registrino per lo stesso evento
    - Si creano dei conflitti quando i risultati della gestione dell'evento sono disponibili

Sistemi interrupt-driven

- Sono particolarmente adatti per sistemi real-time in cui è necessario fornire risposte immediate ad eventi generati esternamente
  - Sistema di sicurezza di un auto che individua un incidente ed attiva tempestivamente, ad esempio, l'airbag
- Esistono tipi di interruzione noti ed un gestore definito per ciascuno di essi
  - Ogni tipo di interruzione è associato a una posizione di memoria che contiene l'indirizzo del suo gestore
  - Quando viene ricevuta un'interruzione, un selettore hw trasferisce immediatamente il controllo al suo gestore che avvia o arresta altri processi in risposta all'evento segnalato dall'interruzione
- Tale modello può essere combinato con quello di gestione centralizzata
  - Il gestore centrale gestisce le normali operazioni del sistema e usa il controllo interrupt-based per le emergenze

Controllo Interrupt-driven



Controllo Interrupt-driven: vantaggi e svantaggi

- Vantaggio
  - Consente risposte molto rapide agli eventi da implementare
- Svantaggio
  - Complesso da programmare
  - Difficile da convalidare
    - Può essere impossibile replicare gli schemi di temporizzazione delle interruzioni durante la fase di test del sistema

Attività di Progettazione di un sistema

Attività di progettazione del sistema: dagli oggetti ai sottosistemi

- La progettazione del sistema consiste nel trasformare il modello di analisi nel modello di progetto prendendo in considerazione i requisiti non funzionali descritti nel documento dei requisiti
- Illustriamo queste attività con un esempio: *MyTrip*
  - Sistema di pianificazione di percorsi stradali per automobilisti
  - Partiamo con un breve modello di analisi per MyTrip
    - Successivamente descriviamo l'identificazione degli obiettivi di progetto e il progetto della decomposizione iniziale in sottosistemi

Modello di analisi per MyTrip

- Usando MyTrip un automobilista può pianificare il proprio viaggio dal proprio computer di casa attraverso un servizio di pianificazione viaggio sul Web (*PlanTrip*)
  - Il viaggio viene salvato sul server per successive consultazioni

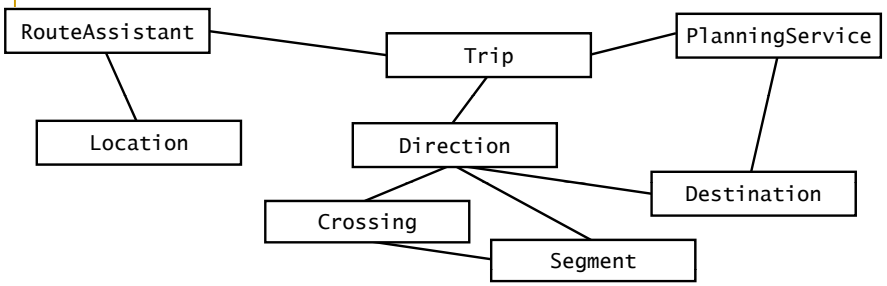
Nome caso d'uso	PlanTrip
Flusso di eventi	1. Il Driver attiva il proprio computer e si logga nel servizio Web di pianificazione viaggio 2. Il Driver immette i vincoli per il viaggio come sequenza di destinazioni 3. Sulla base di un db di mappe, il servizio di pianificazione calcola la via più breve per visitare le destinazioni nell'ordine specificato. Il risultato è una sequenza di segmenti che legano una serie di attraversamenti ed una lista di direzioni 4. Il Driver può revisionare il viaggio aggiungendo o rimuovendo destinazioni 5. Il Driver salva il viaggio pianificato per nome nel db del servizio di pianificazione per successivi accessi

Caso d'uso ExecuteTrip

- A questo punto l'automobilista va in macchina ed inizia il viaggio, mentre il computer di bordo fornisce le direzioni sulla base
  - delle informazioni di viaggio del servizio di pianificazione e
  - la posizione corrente indicata dal sistema GPS di bordo

Nome caso d'uso	ExecuteTrip
Flusso di eventi	1. Il Driver avvia l'automobile e si logga nel sistema di bordo di assistenza al viaggio 2. Avvenuto il log-in, il Driver specifica il servizio di pianificazione ed il nome del viaggio da eseguire 3. L'assistente di bordo ottiene la lista delle destinazioni, le direzioni, i segmenti e gli attraversamenti dal servizio di pianificazione 4. Data la posizione attuale, l'assistente di bordo fornisce al l'automobilista il successivo insieme di direzioni 5. Il Driver arriva a destinazione e spegne l'assistente di bordo

Modello di analisi per MyTrip



Crossing	Punto geografico in cui si incontrano diversi segmenti
Destination	Rappresenta una località in cui il Driver desidera andare
Direction	Dato un Crossing ed un Segment adiacente, una Direction descrive in linguaggio naturale come guidare l'auto su un dato Segmento
Location	Posizione dell'auto nota a partire dal sistema GPS
PlanningService	Web server che suggerisce un viaggio, collegando un numero di destinazioni in forma di Crossing e Segment
RouteAssistant	Fornisce le direzioni all'automobilista, data la Location corrente e il prossimo Crossing
Segment	Rappresenta la strada tra due Crossing
Trip	Sequenza di Direction tra due Destination

Requisiti non funzionali per MyTrip

- Inoltre, durante la scoperta dei requisiti, il nostro cliente ha specificato i seguenti requisiti non funzionali per MyTrip:
  1. MyTrip è in contatto con il PlanningService via un modem wireless. Assumiamo che il modem funzioni correttamente alla destinazione iniziale
  2. Una volta che il viaggio è iniziato, MyTrip dovrebbe dare le direzioni corrette anche se il modem fallisce nel mantenere una connessione con PlanningService
  3. MyTrip dovrebbe minimizzare il tempo di connessione per ridurre i costi operativi
  4. La ripianificazione è possibile solo se è possibile la connessione con PlanningService
  5. Il PlanningService può supportare almeno 50 differenti automobilisti e 1000 viaggi

Identificare gli obiettivi di progetto

- Gli obiettivi di progetto identificano le qualità su cui il nostro sistema deve focalizzarsi
- Si possono inferire numerosi obiettivi dai requisiti non funzionali o dal dominio dell'applicazione
  - Mentre altri saranno scoperti dal cliente
- Devono comunque essere definiti esplicitamente in modo che ogni importante decisione di progettazione può essere fatta in modo consistente seguendo lo stesso insieme di criteri

Obiettivi per MyTrip

- Sulla base dei requisiti non funzionali, identifichiamo gli obiettivi di progetto *affidabilità* e *tolleranza agli errori per la perdita di connessione*
- Successivamente identifichiamo *sicurezza* poiché numerosi automobilisti accederanno allo stesso server per la pianificazione viaggi
- Aggiungiamo *modificabilità* poiché vogliamo fornire la possibilità ai guidatori di selezionare un proprio servizio di pianificazione viaggi

Obiettivi di progetto per MyTrip

- Affidabilità: MyTrip dovrebbe essere affidabile [generalizzazione del requisito 2]
- Tolleranza agli errori: MyTrip dovrebbe essere tollerante agli errori rispetto alla perdita della connessione con il servizio [requisito non funzionale 2 riformulato]
- Sicurezza: MyTrip dovrebbe essere sicuro, cioè, non deve consentire ad altri guidatori o utenti non autorizzati di accedere ai viaggi di un guidatore [dedotto dal dominio applicativo]
- Modificabilità: MyTrip dovrebbe essere modificabile per usare diversi servizi stradali [anticipazione di cambiamento da parte degli sviluppatori]

Obiettivi di progetto

- In generale, selezioniamo gli obiettivi di progetto dalla lunga lista di qualità desiderabili che abbiamo visto nella lezione precedente in cui i criteri di design erano organizzati nei cinque gruppi
  - Prestazioni
  - Affidabilità
  - Costi
  - Manutenzione
  - End user
- I criteri Prestazioni, affidabilità e end user solitamente sono specificati nei requisiti o inferiti dal dominio applicativo
- I criteri costi e manutenzione sono dettati dal cliente e dal fornitore

Identificare i sottosistemi

- L'individuazione dei sottosistemi durante la progettazione del sistema è simile al modo di trovare gli oggetti durante l'analisi
- Ad esempio, alcune tecniche di identificazione degli oggetti di analisi come le euristiche di Abbott possono essere applicate all'identificazione dei sottosistemi
- Inoltre, la decomposizione in sottosistemi viene revisionata costantemente laddove sono esaminate nuove questioni
  - Diversi sottosistemi sono fusi in un sottosistema, un sottosistema complesso è suddiviso in parti, sono aggiunti altri sottosistemi per gestire nuove funzionalità
  - Le prime iterazioni sulla decomposizione in sottosistemi possono introdurre cambiamenti drastici nel modello di progetto del sistema. Questi sono solitamente meglio gestiti attraverso dei brainstorming

Sottosistemi per MyTrip

- La decomposizione iniziale in sottosistemi dovrebbe essere derivata dai requisiti funzionali
  - In MyTrip identifichiamo due gruppi principali di oggetti: quelli coinvolti durante il caso d'uso PlanTrip e quelli coinvolti nel caso d'uso ExecuteTrip
  - Le classi Trip, Direction, Crossing, Segment e Destination sono condivise tra i due casi d'uso. Tale insieme di classi è altamente accoppiato poiché sono usate come un tutt'uno per rappresentare un viaggio
    - Decidiamo allora di assegnarle con PlanningService a PlanningSubsystem e le restanti classi sono assegnate a RoutingSubsystem
    - Otteniamo così una sola associazione tra i due sottosistemi. E' da osservare che tale decomposizione è un repository in cui PlanningSubsystem è responsabile per la struttura dati centrale

Identificazione dei sottosistemi

- Un'altra euristica per l'identificazione dei sottosistemi consiste nel mantenere insieme oggetti funzionalmente in relazione
- Un punto di partenza è assegnare gli oggetti partecipanti identificati in ciascun caso d'uso ai sottosistemi. Alcuni gruppi di oggetti, come il gruppo *Trip* in *MyTrip* sono condivisi e usati per comunicare le informazioni da un sottosistema ad un altro
- Possiamo creare sia un nuovo sottosistema per accomodarli che assegnarli al sottosistema che li crea

Euristiche

- Assegnare gli oggetti identificati in un caso d'uso nello stesso sottosistema
- Creare un sottosistema dedicato per gli oggetti usati per muovere i dati tra i sottosistemi
- Minimizzare il numero di associazioni che attraversano i confini dei sottosistemi
- Tutti gli oggetti nello stesso sottosistema dovrebbero essere funzionalmente legati

Decomposizione per MyTrip