Università degli Studi di Padova
Corso di Laurea
Magistrale in Informatica

Relazione del progetto per il corso di Sistemi Concorrenti e Distribuiti

Simulatore concorrente e distribuito di una competizione di Formula 1

Studente: Cacco Federico
Matricola: 624686
Data: 18 settembre 2013
Versione: 1.2

Diario delle modifiche

Tra parentesi sono indicate le numerazioni delle sezioni corrispondenti alla versione precedente.

MODIFICA	AUTORE	DATA	VER
Prima versione del documento	Cacco Federico	28-07-2013	1.0
Correzione errori ortografici Migliorata la leggibilità Aggiunte sezioni 4.3, 5.1 e 5.3.4 Modificati e integrati sezioni 1.2, 3.1, 3.3.1, 4.2, 5.2 (5.1) 5.3 (5.2), 5.4 (5.3), 5.5 (5.4), 5.6.1 (5.5.1), 5.6.2 (5.5.2), 5.7 (5.6)	Cacco Federico	12-09-2013	1.1
Corretti errori ortografici Aggiunto schema dello schieramento in griglia nella sezione 5.2	Cacco Federico	18-09-2013	1.2

Indice

CAPITOLO 1

Introduzione

1.1 Scopo del progetto

Il progetto didattico per il corso di Sistemi Concorrenti e Distribuiti consiste nell'analisi, e la successiva risoluzione, delle problematiche relative alla progettazione di un simulatore concorrente e distribuito di una competizione paragonabile ad una gara Formula 1.

Il sistema da simulare dovrà prevedere:

- Un circuito, possibilmente selezionabile in fase di configurazione, dotato almeno della pista e della corsia di rifornimento. Entrambe dovranno essere soggette a regole congruenti di accesso, condivisione, tempo di percorrenza, condizioni atmosferiche, ecc.
- Un insieme configurabile di concorrenti, ciascuno con caratteristiche specifiche di prestazione, risorse, strategia di gara, ecc.
- Un sistema di controllo capace di riportare costantemente, consistentemente e separatamente, lo stato della competizione, le migliori prestazioni (sul giro, per sezione di circuito) e anche la situazione di ciascun concorrente rispetto a specifici parametri tecnici
- Una particolare competizione, con specifica configurabile della durata e controllo di terminazione dei concorrenti a fine gara.

Si nota subito la natura concorrente del problema, in quanto tra i vari aspetti di cui bisogna tener conto nella progettazione vi sono quelli legati ai sorpassi tra piloti impegnati a percorrere lo stesso tratto di pista e la gestione dell'accesso alla corsia dei box.

Analogalmente il progetto si presta anche ad alcune valutazioni dal punto di vista della distribuzione, data la presenza di almeno 2 blocchi distinti: un primo che si occupa della simulazione della competizione e un secondo che si occupa della rappresentazione grafica del suo stato.

Addizionalmente per ottenere una simulazione verosimile della competizione vi sono poi tutta una serie di parametri di cui tener conto, come ad esempio le condizioni meteo, le differenti abilità dei piloti e le varie caratteristiche delle vetture.

1.2 Terminologia

Per chiarezza di seguito verranno spiegati alcuni termini usati all'interno della relazione, che altrimenti potrebbero ad una prima lettura sembrare ambigui.

Tracciato Con il termine tracciato si fa riferimento alla carreggiata della pista su cui corrono le vetture di Formula 1, comprende anche la corsia dei box.

Circuito Con circuito fa riferimento a un tracciato dove sia stata stabilita una griglia di partenza.

Gara Gara rappresenta l'evento a cui partecipano i piloti di formula uno. Si tratta di un circuito su cui sono state impostati un numero totale di giri da eseguire e delle condizioni meteo.

Concorrente Si tratta dell'entità che partecipa alla gara, ogni concorrente è composto da un pilota e da una vettura a lui associata.

Stato del pilota Con stato del pilota si intende l'insieme dei valori che caratterizzano un pilota e la relativa vettura (ovvero nome, numero, costruttore, benzina nel serbatoio e stato degli pneumatici) e quelli relativi alla sua gara, ovvero i tempi degli intermedi del giro corrente, il tempo totale di gara. il numero di soste fatte e il numero di giri conclusi.

Stato del sistema o stato della gara Con stato del sistema, o stato della gara, si intende l'insieme dei valori che consentono di descrivere le caratteristiche della gara (nome circuito, numero di giri e condizioni meteorologiche), unite alle informazioni relative allo stato di ogni pilota partecipante a tale gara in modo da poter ricavare la classifica corrente e i distacchi dei vari piloti dal primo in classifica.

CAPITOLO 2

Analisi dei requisiti

I requisiti funzionali obbligatori del progetto sono elencati nella tabella 2.1

CODICE	**DESCRIZIONE**
RFOBB-01	Presenza di un circuito nella competizione
RFOBB-02	Presenza di piloti nella competizione
RFOBB-03	Presenza di un sistema di controllo della competizione
RFOBB-04	Il circuito deve essere dotato di una pista e della corsia di rifornimento
RFOBB-05	La pista deve essere soggetta a regole congruenti di accesso
RFOBB-06	La corsia dei box deve essere soggetta a regole congruenti di accesso
RFOBB-07	La pista e la corsia box devono essere condivisibili tra i piloti
RFOBB-08	La pista e la corsia box devono avere tempi di percorrenza verosimili
RFOBB-09	La pista e la corsia box devono essere soggette a condizioni atmosferiche
RFOBB-10	I concorrenti devono possedere personali caratteristiche di prestazione
RFOBB-11	I concorrenti devono possedere una strategia di gara
RFOBB-12	I concorrenti devono possedere una vettura con specifiche caratteristiche prestazionali

RFOBB-13	Il sistema di controllo deve riportare lo stato della competizione
RFOBB-14	Il sistema di controllo deve riportare le prestazioni e o stato dei piloti
RFOBB-15	Il sistema di controllo deve tener traccia delle migliori prestazioni
RFOBB-16	Durata e condizione meteo della gara devono essere configurabili
RFOBB-17	Presenza di un controllo di terminazione dei concorrenti a fine gara

Tabella 2.1: Requisiti funzionali obbligatori

I requisiti funzionali opzionali del progetto sono invece elencati nella tabella 2.2

CODICE	**DESCRIZIONE**
RFOPZ-01	Il circuito può essere scelto in fase di configurazione
RFOPZ-02	I piloti devono poter essere configurabili

Tabella 2.2: Requisiti funzionali opzionali

CAPITOLO 3

Progettazione

Durante la prima fase di progettazione si è cercato di individuare un modello per rappresentare lo scenario dato e le sue principali componenti.

3.1 Modello e componenti

3.1.1 Modello

Nello sviluppo del progetto per prima cosa è stato necessario stabilire con che modello rappresentare lo scenario proposto. Dato che lo scopo di tale progetto si focalizza sullo studio degli aspetti di natura concorrente e distribuita, è stato scelto di utilizzare un modello a tempo discreto piuttosto che uno a tempo continuo.

Questo perché, volendo appunto analizzare aspetti di natura concorrente, non sarebbe interessante valutare l'evoluzione del sistema eseguito mediante una successione di avanzamenti unitari, dato che ciò porterebbe alla mancanza di un'esecuzione in tempo reale e verrebbero quindi a mancare molti aspetti legati allo studio della concorrenza tra le varie entità.

Il fatto di aver optato per un modello a tempo discreto ha comportato il dover decidere in quali istanti dovesse essere fornito uno stato coerente del sistema. Rapportandosi alla realtà della Formula 1, dove lo stato di ogni pilota viene rilevato al termine di determinati settori della pista (chiamati intermedi), si è scelto di procedere in modo analogo ovvero aggiornando lo stato del sistema ogni volta che un pilota attraversa un determinato punto del tracciato.

L'aggiornamento del sistema risulta quindi legato ad una dimensione spaziale (la posizione del pilota nella pista) e non temporale (il tempo trascorso), e ciò ha implicato che non sia più il tempo, ma la distanza percorsa

la variabile indipendente tramite la quale vengono calcolati tutti gli altri dati.

Scegliere il tempo come variabile indipendente non sarebbe stata una scelta molto conveniente, in quanto per calcolare in quali istanti di tempo determinare lo stato del sistema si sarebbero dovuti calcolare tutti gli istanti in cui un pilota sarebbe uscito da un segmento. Viceversa questa sarebbe stata una scelta valida nel caso in cui si fosse deciso di determinare uno stato coerente del sistema ad intervalli regolari di tempo invece che in base alla distanza percorra da un pilota.

Dato che lo stato del sistema viene aggiornato in modo discreto, è inoltre necessaria la presenza di un modello di concorrenza per far si che esso evolva in modo corretto tra uno stato e l'altro.

3.1.2 Componenti

Successivamente si è pensato a come scomporre il sistema, individuandone le varie componenti fondamentali.

Per prima cosa sono state individuate 2 partizioni ben distinte, una chiamata F1Engine legata al funzionamento dell'applicazione e una chiamata F1ControlPanel legata alla visualizzazione dello stato della competizione.

La motivazione di questa prima suddivisione risiede nella necessità di poter poi distribuire l'intero sistema, creando 2 partizioni che possano cooperare eseguendo però in modo indipendente l'una dall'altra.

Per ogni partizione del progetto sono state poi individuati i seguenti componenti:

- F1Engine
  - Gara
  - Concorrenti
  - StartUp
- F1ControlPanel
  - Monitor

Questi componenti verranno poi descritti in modo più approfondito per chiarire il loro compito all'interno del sistema.

3.2 Gara

Verrà ora descritta la struttura del componente *Gara*, con lo scopo di fornire una prima organizzazione gerarchica dei vari componenti e averne quindi una prima visione d'insieme.

Gara rappresenta l'evento alla quale partecipano i vari concorrenti, ed è composta da un circuito, una condizione meteorologica e un numero di giri da percorrere.

Essa non svolge alcun ruolo attivo e la sua unica funzione è quella di accogliere i piloti impegnati nella competizione, la gestione della parte concorrente è infatti poi effettuata mediante dei suoi sotto-componenti.

Alla base di ciò nella progettazione di *Gara* si è tenuto conto i questi aspetti:

- Dovrà essere specificata una lunghezza del circuito
- Il circuito deve avere una larghezza variabile durante la sua percorrenza, non in tutti i tratti sarà ad esempio possibile eseguire un sorpasso per via dello spazio limitato
- Nel circuito dovranno essere presenti sia tratti curvi che rettilinei, che avranno quindi diverse velocità di percorrenza
- Dovranno esserci dei punti per il rilevamento delle prestazioni
- Dovrà essere presente una corsia per i box, con relative regole di accesso
- Le prestazioni possono essere influenzate dalle condizioni meteorologiche

Come scritto la componente *Gara* in se non svolge alcun ruolo attivo, e trattandosi di una semplice collezione di dati non necessità di particolari accorgimenti come ad esempio possono richiederne una risorsa protetta o un'entità passiva-reattiva. Per questo motivo è possibile modellare tale componente come un semplice record di dati che verrà condiviso tra tutti i piloti, contenente i seguenti campi:

- *Circuito*
- *Numero di giri*
- *Condizioni meteo*

Numero di giri è un semplice campo numerico, mentre *Condizioni meteo* può assumere i valori asciutto o bagnato. Quest'ultimo andrà poi a influenzare il tempo necessario ad un pilota per compiere un giro di pista, in quando con pista bagnata le prestazioni decrescono sensibilmente.

Circuito rappresenta invece il tracciato su cui si svolge la gara, comprendendo anche la relativa griglia di partenza. Come per *Gara* anche quest'ultimo non ha nessun ruolo attivo e può nuovamente essere rappresentato come un semplice record di dati con i seguenti campi:

- *Nome del circuito*
- *Numero di segmenti*
- *Lista dei segmenti*
- *Griglia di partenza*
- *Lunghezza del giro*

Nome del circuito e *Lunghezza del giro* sono dei semplici campi che contengono appunto il nome del circuito e la lunghezza di ogni giro (corsia dei box esclusa)

Dato che la transizione tra 2 stati coerenti del sistema viene effettuata quando un pilota attraversa determinati punti del circuito si ha la necessità di fissare tali punti. Siccome che in un reale circuito di Formula 1 si possono distinguere tratti di pista contigui con caratteristiche simili (come ad esempio curve, rettilinei o tratti che consentono o meno ai vari piloti di sorpassarsi), si è deciso di suddividere il tracciato in vari segmenti, ovvero tratti di pista contigui con simili caratteristiche. Un tracciato è dunque rappresentato come usa successione di questi ultimi, salvata in *Lista dei segmenti*.

Anche *Griglia di partenza* non è un semplice campo dati, ma si tratta di una entità un po' più complessa in quanto rappresenta la griglia di partenza nella quale i piloti si schierano nell'attesa della partenza.

Per chiarezza uno schema della struttura del componente *Gara* è riportato nella figura 3.1

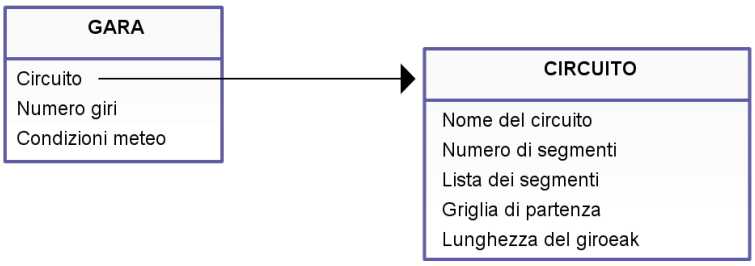

Figura 3.1: Schema di *Gara*

Ora che è stata data una visione d'insieme dell'intera struttura di *Gara*, è possibile descriverne con più chiarezza i componenti principali.

3.2.1 Circuito

Circuito rappresenta il tracciato su cui si svolge la competizione, e viene rappresentato con una successione di segmenti e una griglia di partenza.

Sulla base di ciò esso contiene quindi le seguenti proprietà:

- Nome del circuito
- Numero di segmenti
- Lista dei segmenti
- Griglia di partenza
- Lunghezza del giro

3.2.2 Lista dei segmenti

Questo parametro contiene la lista dei segmenti di cui è composto il circuito. Nella sua composizione dovranno essere rispettati i seguenti vincoli:

- I primi 3 segmenti rappresentano l'ingresso ai box, la corsia box e l'uscita dai box
- Il rettilineo col traguardo deve essere il quarto segmento
- Presenza di almeno 4 intermedi, l'ultimo dei quali coincidente col traguardo

Inoltre si assume che la linea del traguardo sia collocata all'inizio del quarto segmento, che l'ingresso dei box sia collocato all'inizio del rettilineo del traguardo e che l'uscita box sia collocata alla fine del rettilineo del traguardo.

3.2.3 Segmenti

La loro funzione, oltre a quella di descrivere il tracciato su cui si svolge la gara, è quella di gestire la parte di concorrenza relativa ai sorpassi tra piloti determinando i loro tempi di attraversamento in modo coerente rispetto alla presenza o meno di altri piloti.

Le caratteristiche in base alla quale vengono raggruppati i tratti di pista contigui per dare origine ai segmenti sono:

- Numero di corsie del tratto di pista
- Tipologia del tratto di pista (accelerazione, frenata, curva, entrata ai box, uscita dai box, corsia dei box)

Numero di corsie specifica quanti piloti possono attraversare quel tratto di pista contemporaneamente uno di fianco all'altro. *Tipologia del tratto di pista* invece rappresenta il modo in cui si deve comportare un pilota quando lo attraversa, le varie tipologie sono accelerazione, frenata, curva, entrata ai box, uscita dai box e corsia dei box.

Un segmento è dunque rappresentato da un record contenente i seguenti campi:

- Codice segmento
- Tipologia
- Lunghezza
- Velocità massima
- Numero di corsie
- Presenza fotocellula
- Risorsa protetta

Codice segmento

Serve per poter indicizzare, e quindi distinguere, i vari segmenti tramite un codice univoco crescente che ne determina il loro ordinamento nella composizione del circuito.

Per semplicità di implementazione si è decisa la seguente indicizzazione (dove i è l'indice) dei segmenti:

- i=1: il segmento di entrata ai box
- i=2: il segmento della corsia dei box
- i=3: il segmento di uscita dai box
- i=4: il segmento del rettilineo principale (quello in cui è presente il traguardo)
- i>4: gli altri segmenti che vengono dopo il traguardo

Tipologia

Un segmento deve essere in grado di rappresentare il più fedelmente possibile il tratto di pista a lui corrispondente. Compatibilmente con i requisiti del progetto sono state individuate 3 tipologie di segmento:

- Segmenti di accelerazione
- Segmenti a velocità costante
- Segmenti di decelerazione

I segmenti di accelerazione modellano tratti in cui i piloti accelerano, ovvero rettilinei e uscita dai box. I segmenti di decelerazione modellano invece ingressi in curva, staccate e corsia di entrata ai box. Infine i segmenti a velocità costanti modellano curve e corsia dei box.

Lunghezza

Descrive semplicemente la lunghezza in metri del segmento.

Velocità massima

Il significato di questo dato varia in base alla tipologia del segmento.

Nel caso in cui esso sia associato ad un segmento di accelerazione indica la massima velocità che la vettura può raggiungere, questo è utile ad esempio per modellare rettilinei non completamente dritti.

Nei segmenti di decelerazione la velocità massima indica invece la velocità che dovrà avere la vettura al termine del segmento, per poter poi effettuare una curva o entrare in un segmento a velocità controllata come ad esempio la corsia dei box.

Infine nei tratti a velocità costante il valore rappresenta la velocità massima con cui il segmento potrà essere attraversato, e avrà lo stesso valore del precedente tratto di decelerazione.

Un tratto a velocità costante deve essere sempre preceduto da un tratto di decelerazione o di accelerazione che abbia la sua stessa velocità massima. Concettualmente è molto simile ad un tratto di accelerazione con velocità massima limitata, però tale distinzione risulterà poi comoda a livello implementativo per risparmiare inutili calcoli di accelerazione.

Tutti i valori della velocità massima sono solo indicativi e servono per dare una descrizione del segmento, i reali valori di attraversamento verranno calcolati anche in base alle caratteristiche e allo stato di vettura e pilota, come sarà spiegato nella sezione 3.3.4

Numero di corsie

Rappresenta il numero di corsie presenti nel segmento, che influenzano la possibilità di poter effettuare sorpassi da parte dei piloti. Il loro numero determina infatti quanti piloti possono trovarsi contemporaneamente nello stesso tratto di segmento.

In un segmento con una sola corsia sarà dunque impossibile effettuare sorpassi, mentre saranno più agevoli al loro aumentare.

Le corsie d'ingresso e uscita dai box, la corsia dei box e le curve hanno sempre il numero di corsie pari a 1.

Presenza fotocellula

Un circuito di Formula 1 è solitamente suddiviso in 4 intermedi, alla fine dei quali è presente un dispositivo per rilevare i tempi dei vari piloti. Questo parametro indica se alla fine del segmento deve essere rilevato tale tempo, e in caso affermativo specifica anche il numero dell'intermedio alla quale tale rilevazione fa riferimento.

Risorsa protetta

Ogni segmento contiene al suo interno una risorsa protetta, che ha essenzialmente due compiti. Il primo è quello di rendere possibile un accesso ai segmenti congruente ai vari vincoli imposti, il secondo è quello di permettere un ordinamento dei piloti in base alla loro posizione.

Una spiegazione dettagliata del suo funzionamento sarà data nella sezione 5.3.

3.2.4 Griglia di partenza

La griglia di partenza è il luogo dove i piloti prendono posto attendendo l'inizio della gara, ed è anch'essa modellata mediante una risorsa protetta. Il suo funzionamento verrà spiegato nella sezione 5.2

3.3 Piloti

I piloti sono le entità che dovranno gareggiare nella competizione attraversando il circuito. Lo scopo di ognuno d'essi è quello di percorrere tutti i giri previsti nella gara nel minor tempo possibile, il tutto in modo coerente con le caratteristiche del circuito.

Dato che nella realtà ogni pilota ha le proprie abilità (come ad esempio la prontezza di riflessi o la capacità di valutare i punti esatti in cui effettuare le staccate), anche in questo caso ad ogni pilota dovranno essere associate delle caratteristiche, chiamate skill, che possano modellarne il comportamento in pista. Le skill che ogni pilota possiede sono:

- Numero del pilota
- Nome del pilota
- Skill di accelerazione
- Skill di decelerazione

La skill di accelerazione e la skill di decelerazione indicano rispettivamente la capacità del pilota di iniziare l'accelerazione il più presto possibile e la capacità del pilota di frenare il più tardi possibile prima di un inserimento in curva. Bassi valori in queste skill implicheranno che il pilota inizia ad accelerare troppo tardi e a frenare troppo presto rispetto all'istante ottimale, peggiorando così le sue prestazioni nel giro di pista.

Ad ogni pilota viene poi associata una vettura, anch'essa caratterizzata da delle skill che ne influenzeranno le prestazioni in pista. Esse sono:

- Costruttore

- Coefficiente di accelerazione
- Coefficiente di decelerazione
- Velocità massima
- Tenuta di strada
- Consumo degli pneumatici
- Affidabilità

I coefficienti di accelerazione e di decelerazione indicano le prestazioni di accelerazione e frenata della vettura, vetture con alti coefficienti avranno quindi migliori capacità di accelerazione e decelerazione che comporteranno prestazioni migliori.

La tenuta di strada viene invece usata per calcolare i tempi di percorrenza nelle curve e quanto peggiorano le prestazioni in caso di pista bagnata.

Il consumo degli pneumatici invece stabilisce in quanti giri essi degradano e hanno quindi bisogno di essere sostituiti ai box. Vetture con un basso degrado degli pneumatici potranno compiere più giri senza doversi fermare ai box.

L'affidabilità specifica invece la probabilità che ha una vettura di riuscire a compiere l'intera gara senza essere soggetta a gusti che porterebbero ad un ritiro del pilota che la utilizza.

Ogni pilota inoltre possiede un propria strategia, che rappresenta l'elenco dei giri in cui fermarsi per effettuare il cambio gomme. E' importante bilanciare il vantaggio di avere sempre pneumatici in buone condizioni con lo svantaggio di doversi fermare per sostituirli.

Infine piloti che utilizzano vetture con un basso consumo di carburante potranno completare la gara immagazzinando alla partenza un minor quantitativo d'esso, rendendo così la vettura più leggera e performante.

3.3.1 Simulazione accelerazione

Dato che simulare la reale curva di accelerazione di una vettura di Formula 1 è un'operazione molto complessa, è stato deciso di semplificarne la sua rappresentazione. L'aspetto chiave da tenere in considerazione è il fatto che la forza di accelerazione varia in funzione della velocità, più precisamente più bassa è la velocità e maggiore è l'accelerazione che si riesce ad imprimere alla vettura.

Come scritto in 3.1.1 la variabile indipendente adottata nel sistema è lo spazio percorso, questo implica che il tempo necessario ad attraversare un segmento e la velocità raggiunta da una vettura nel suo attraversamento devono essere calcolati in funzione di esso.

La funzione scelta per rappresentare la velocità v in funzione della distanza x percorsa è la seguente:

$$v(x) = 80 * log[(\frac{x}{100}) + 1]$$

La funzione inversa per calcolare lo spazio percorso x in funzione della velocità raggiunta v è invece:

$$x(v) = 100 * [10^{(v/80)} - 1]$$

Il tempo necessario all'attraversamento viene poi calcolato partendo dall'ipotesi di attraversare il segmento ad una velocità costante pari alla media tra la velocità di ingresso e quella d'uscita, ma una descrizione più dettagliata del suo calcolo verrà data in 3.3.4.

Nella figura 3.2 viene rappresentato il grafico della funzione usata per simulare il valore della velocità raggiunta in funzione della distanza percorsa, si nota come l'accelerazione diminuisca all'aumentare della velocità.

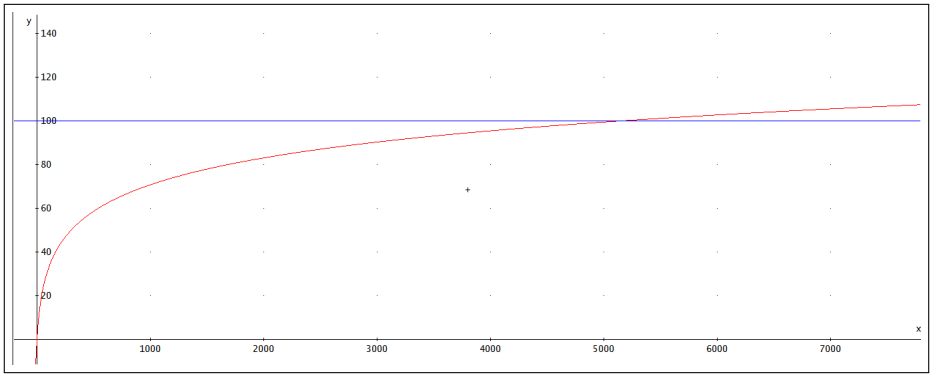

Figura 3.2: Andamento della velocità in base alla distanza percorsa di una vettura con velocità iniziale pari a 0m/s

Tale grafico presuppone che la vettura parta da una velocità iniziale di 0m/s, ma questo si verifica però solo alla partenza. Per poter calcolare la variazione di velocità della vettura in funzione della distanza percorsa anche nel caso in cui essa abbia una velocità iniziale v_0 diversa da 0, è sufficiente traslare il grafico fino a intersecare tale valore v_0 con l'asse delle ordinate. Un esempio è mostrato in figura 3.3

3.3.2 Simulazione decelerazione

La simulazione della decelerazione risulta molto più semplice rispetto a quella dell'accelerazione in quanto si può approssimare il fatto che durante la frenata la forza impressa dal freno sia costante.

La simulazione di una decelerazione viene effettuata calcolando per prima cosa lo spazio necessario alla frenata, in modo da determinare per quanto

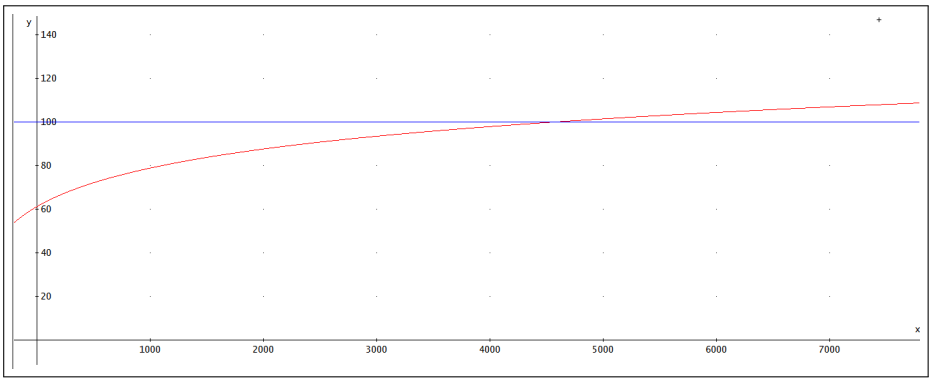

Figura 3.3: Andamento della velocità in funzione della distanza, assumendo che la vettura abbia una velocità iniziale v_0 di 60m/s

spazio il pilota continuerà ad accelerare e per quando spazio eseguirà la frenata. Ad ogni vettura è associato quindi un coefficiente $EffDec$, che misura la forza sua forza di decelerazione

La formula con la quale essa viene calcolato lo spazio di frenata x necessario per effettuare una variazione di velocità ΔV alla velocità iniziale v_i è la seguente:

$$x(\Delta V) = \frac{1}{2} * EffDec * \frac{\Delta V^2}{EffDec^2} + v_i * \frac{\Delta v}{EffDec}$$

Il tempo necessario per effettuare la frenata sarà invece

$$t(\Delta v) = \frac{\Delta v}{DecEff}$$

3.3.3 Simulazione tratto a velocità costante

Il questo caso per calcolare il tempo di attraversamento di usano le leggi del moto rettilineo uniforme, dove v è la velocità di attraversamento e l la lunghezza del segmento

$$t(v) = \frac{l}{v}$$

3.3.4 Azioni del pilota

Nel loro ciclo di vita i piloti compiono le seguenti azioni:

- Schieramento nella griglia di partenza
- Partenza
- Attraversamento dei segmenti

- Eventuali soste ai box
- Terminazione della gara

Tali azioni verranno descritte a seguire, ma senza fare alcun riferimento a concetti di concorrenza, in quanto saranno poi esposti nel capitolo 5

Schieramento nella griglia di partenza

Questa è la prima fase della gara, semplicemente una volta che il pilota è stato creato viene posizionato nella griglia di partenza nell'attesa di iniziare la gara.

Partenza

Una volta tutti i piloti sono schierati nella griglia di partenza il sistema resta in attesa di un input da parte dell'utente per poter dare inizio alla gara.

Appena ricevuto il segnale i piloti partono e iniziano l'attraversamento dei segmenti.

Attraversamento dei segmenti

In questa fase i piloti attraversano i vari segmenti del tracciato nel tentativo di completare tutti i giri nel minor tempo possibile.

L'attraversamento di un segmento si svolge in modo analogo sia che il tratto di pista sia di accelerazione, di frenata o a velocità costante (curva). Nei vari casi cambia solo il modo con cui viene determinato il tempo di attraversamento.

Nel caso in cui il pilota sia impegnato in un tratto a velocità costante il tempo di attraversamento viene calcolato nel seguente modo:

1. In base alla velocità del pilota e alla lunghezza del segmento viene calcolato il tempo di attraversamento previsto
2. Si calcolano poi delle penalità che verranno aggiunte a tale tempo, che sono:
   - Condizioni meteo e tenuta di strada (max 10% del tempo calcolato)
   - Stato degli pneumatici (max 4% del tempo calcolato)
   - Peso della benzina nella vettura (0,05% in più del tempo per ogni litro nel serbatoio)

   Tali penalità vengono poi sommate al tempo di attraversamento

3. Il pilota chiede l'accesso alla risorsa protetta del segmento comunicando il tempo atteso per l'attraversamento. La risorsa risponde comunicando il tempo effettivo di attraversamento sulla base della presenza o meno di altri piloti

Nel caso invece in cui il pilota sia impegnato in un tratto di accelerazione i passi per calcolare il tempo necessario al suo attraversamento sono:

1. In base alle caratteristiche del pilota viene calcolato lo spazio di ritardo con cui il pilota inizierà l'accelerazione. Tale spazio potrà essere al massimo il 10% della lunghezza del segmento

2. Viene quindi calcolato lo spazio effettivo di accelerazione, che sarà quindi tanto maggiore quanto il pilota sarà abile nell'accelerare il prima possibile

3. In base dello spazio di accelerazione effettivo viene calcolata la velocità massima che il pilota avrà alla fine del tratto di accelerazione, tale velocità sarà ridotta al massimo del 10% sulla base delle prestazioni in accelerazione della vettura, e non sarà comunque superiore alla velocità massima della vettura

4. Sulla base della velocità di entrata del segmento e di quella di uscita si calcola il tempo di attraversamento

5. Si calcolano poi delle penalità che verranno aggiunte a tale tempo, che sono:

   - Condizioni meteo e tenuta di strada (max 10% del tempo calcolato)
   - Stato degli pneumatici (max 4% del tempo calcolato)
   - Peso della benzina nella vettura (0,05% in più del tempo per ogni litro nel serbatoio)

   Tali penalità vengono poi sommate al tempo di attraversamento

6. Il pilota chiede l'accesso alla risorsa protetta del segmento comunicando il tempo atteso per l'attraversamento. La risorsa risponde comunicando il tempo effettivo di attraversamento sulla base della presenza o meno di altri piloti

Infine nel caso in cui il pilota sia impegnato in un tratto di decelerazione il procedimento è il seguente:

1. Viene calcolato lo spazio di frenata in base alla skill di frenata dell'auto

2. In base alle caratteristiche del pilota viene calcolato lo spazio di anticipo con cui il pilota inizierà la decelerazione. Tale spazio potrà essere al massimo il 10% della lunghezza del segmento Maggiore è questo spazio maggiore sarà l'anticipo della frenata, fatto che causerà una diminuzione delle prestazioni.

3. Viene quindi calcolato lo spazio effettivo di frenata, che sarà quindi tanto minore quanto il pilota sarà abile nel frenare il più tardi possibile

4. In base allo spazio di frenata effettivo viene calcolato il tempo di attraversamento atteso, tale valore verrà poi modificato in base alle skill del pilota e alle caratteristiche della vettura

5. Si calcolano poi delle penalità che verranno aggiunte a tale tempo, che sono:

   - Condizioni meteo e tenuta di strada (max 10% del tempo calcolato)
   - Stato degli pneumatici (max 4% del tempo calcolato)
   - Peso della benzina nella vettura (0,05% in più del tempo per ogni litro nel serbatoio)

   Tali penalità vengono poi sommate al tempo di attraversamento

6. Il pilota chiede l'accesso alla risorsa protetta del segmento comunicando il tempo atteso per l'attraversamento. La risorsa risponde comunicando il tempo effettivo di attraversamento sulla base della presenza o meno di altri piloti

Soste ai box

I piloti avranno poi una loro personale strategia di gara, questo consentirà loro di effettuare i sorpassi non solo durante la percorrenza del giro, ma anche mediante le soste ai box. La strategia di gara specifica la benzina iniziale che la vettura avrà nel serbatoio a inizio gara, e i giri in cui ci si fermerà per il cambio degli pneumatici.

Entrambi i fattori sono importanti. Bisognerà sia caricare il minimo quantitativo di benzina necessario per portare a termine la gara (avendo così la vettura il più leggera possibile) e sia schedulare al meglio le soste ai box per il cambio pneumatici, tenendo presente che con pneumatici usurati le prestazioni decrescono in maniera significativa ma allo steso la sosta ai box per la loro sostituzione è un'operazione che richiede una certa quantità di tempo.

Dato che nelle competizioni reali le soste ai box sono diventate estremamente veloci (specialmente dopo il divieto di poter effettuare rifornimenti di carburante), nel sistema progettato una sosta ai box viene simulata mediante l'attraversamento della corsia box a velocità ridotta.

Fine della gara

Molte volte nelle competizioni di Formula 1 un pilota termina la propria gara non perché abbia completato tutti i giri previsti, ma a causa di altri eventi come ad esempio incidenti, gusti o altro.

Volendo riportare questa caratteristica anche nel progetto, sono stati fissati 3 motivi per i quali un pilota può terminare la propria gara, e sono:

- Completamento di tutti i giri previsti
- Fine della benzina
- Guasto nella vettura

Ogni volta che si conclude il giro viene fatto un controllo per determinare se il pilota ha abbastanza benzina per completare un altro giro. In caso affermativo esso procede normalmente effettuando il giro successivo, in caso contrario esso si ritira dalla competizione terminando così la sua gara.

In base all'affidabilità della vettura inoltre un pilota ha una certa probabilità che durante la percorrenza del giro la sua vettura possa rompersi causandone il suo ritiro.

3.3.5 Invio dello stato del pilota

Il pilota inoltre in determinati momenti invia delle informazioni sul suo stato al monitor, che poi le elabora per la loro visualizzazione a video.

Le informazioni trasmesse da un pilota sono le seguenti:

- Schieramento del pilota nella griglia di partenza
- Comunicazione del tempo fatto registrare in un determinato intermedio
- Ingresso ai box
- Uscita dai box
- Comunicazione della benzina rimasta e dello stato degli pneumatici
- Comunicazione di gara terminata

Nella sezione 4.2 si spiegherà come queste informazioni vengono effettivamente inviate al monitor.

3.4 Monitor

Il monitor è una semplice interfaccia grafica che raccoglie le informazioni sullo stato dei piloti ordinati secondo la loro posizione.

Essa è modellata come un'entità reattiva, in quando il suo unico compito è quello di ricevere i messaggi sullo stato della competizione e riportarli a video.

Le informazioni vengono visualizzate in una tabella dove ad ogni riga è associato un pilota, per ognuno di essi viene visualizzato:

- Nome Pilota
- Numero
- Costruttore
- Tempo primo intermedio
- Tempo secondo intermedio
- Tempo terzo intermedio
- Tempo al traguardo
- Distacco dal primo
- Numero giri fatti
- Stato del pilota (in gara, ai box o ritirato)
- Numero soste ai box fatte
- Carburante presente nel serbatoio
- Usura delle gomme
- Strategia del pilota (elenco dei giri in cui si fermerà)
- Tempo di gara (Visualizzato solo quando il pilota termina la competizione)

Vengono inoltre visualizzate informazioni come nome del circuito, tempo meteorologico, numero di giri previsti, stato della gara e miglior giro effettuato

Il modo con cui tale informazioni vengono raccolte è spiegato nella sezione 4.2

3.5 StartUp

Il componente StartUp si occupa della creazione di Gara e piloti e dell'avvio della competizione. Esso è l'unico metodo invocato dal main di F1Engine e ha il seguente ciclo di vita:

1. Creazione della gara
2. Creazione dei piloti
3. Attesa schieramento dei piloti sulla griglia di partenza
4. Attesa della conferma da parte dell'utente per la partenza
5. Attesa della fine della gara
6. Terminazione a fine gara

CAPITOLO 4

Distribuzione

4.1 Partizionamento

Il progetto svolto si presta ad alcune considerazioni sull'aspetto della distribuzione. Si può infatti notare che esso è strutturato in 2 parti ben distinte, una contenente la logica della competizione (Piloti, Gara e StartUp), e una contenente il monitor per la visualizzazione dello stato della competizioni.

Inoltre entrambi i blocchi sono indipendenti uno dall'altro. I piloti possono infatti gareggiare anche senza la presenza del monitor, mentre quest'ultimo può eseguire anche senza che nessun pilota stia gareggiando (ovviamente non mostrerà nessun dato).

Tale suddivisione è inoltre giustificata dal fatto che le parti che richiedono un certo carico computazionale (seppur modesto) sono quelle relative ai piloti e al monitor, mentre Gara, trattandosi di una collezione di record di memoria e di risorse protette, ha solamente un ruolo passivo e richiede perciò una capacità di calcolo molto inferiore.

Queste considerazioni hanno portato quindi a scomporre il progetto in 2 nodi così composti:

- Nodo 1 (F1Engine):
  - Piloti
  - Gara
  - StartUp
- Nodo 2 (F1ControlPanel):
  - Monitor

4.2 Comunicazione

Una parte fondamentale della distribuzione riguarda la scelta del modello da usare per lo scambio delle informazioni tra i 2 nodi F1Engine e F1ControlPanel.

Nel nostro caso le informazioni scambiate hanno una sola direzione in quanto il monitor nel nodo F1ControlPanel, che ha il solo compito di ricevere le informazioni dal nodo F1Engine e mostrarle a video, non manda alcun messaggio di risposta verso il nodo F1Engine.

Questo perché dato che quasi la totalità delle informazioni visualizzate vengono inviate dal pilota, non è desiderabile che esso attenda la risposta di avvenuta ricezione del messaggio da parte del monitor, in quando questo trasformerebbe l'operazione di invio messaggio in una operazione bloccante influenzandone così in modo indesiderato il suo comportamento.

Tale aspetto potrebbe essere risolto creando un modulo dedicato solamente alla gestione dei messaggi spediti al monitor da parte dei piloti che sia disaccoppiato da quest'ultimi, in modo da poterne rinviare i messaggi senza bloccarne l'esecuzione in attesa di una risposta. Ciò avrebbe richiesto tuttavia una progettazione più complessa che avrebbe portato al superamento delle ore di lavoro previste per la realizzazione del progetto.

Il verificarsi di una informazione non ricevuta o ricevuta in ritardo non causerebbe comunque un'inconsistenza dello stato della competizione a livello logico, ma solo una inconsistenza dei dati mostrati a video. Tale fenomeno si risolverebbe poi mano a mano che la gara avanza e che i nuovi dati aggiornati vengono ricevuti.

Tali considerazioni suggeriscono quindi la possibilità di usare una comunicazione asincrona nello scambio dei messaggi tra i 2 nodi basata su chiamate a procedure remote, con semantica oneway best-effort.

4.3 Middleware e interfaccia

La realizzazione della distribuzione tra 2 nodi comporta il dover utilizzare un middleware per permettere la loro comunicazione. Dato che il nodo F1ControlPanel svolge il solo ruolo di servente, in quanto tutte le sue azioni svolte (che consistono solamente nella visualizzazione dei dati) sono conseguenza di messaggi inviati dal nodo F1Engine, è ragionevole utilizzare un middleware di distribuzione che adotti un modello cliente-servente. In questa prospettiva il nodo F1ControlPanel si occuperà di avviare il lato servente fornendo tutte le informazioni al lato cliente (in questo caso il nodo F1Engine) per poter interagire con esso.

Diviene poi necessario stabilire un'interfaccia di comunicazione, che renda possibile l'esecuzione di tutte funzionalità assegnate al monitor elencate in 3.4. Tale interfaccia dovrà quindi contenere le seguenti chiamate, dove tra parentesi sono indicati i relativi parametri:

- Aggiunta di un pilota al monitor (numero, nome, costruttore della relativa vettura, strategia, quantità di benzina a bordo)
- Notifica del tempo realizzato in un intermedio (numero del pilota, tempo realizzato, numero giri, numero intermedio, ticket, tempo attuale di gara)
- Notifica di entrata ai box di un pilota (numero del pilota entrante)
- Notifica di uscita ai box di un pilota (numero del pilota uscente)
- Invio dei dati della gara (nome del circuito, lunghezza, numero di giri previsti, condizioni meteo)
- Notifica di gara partita
- Notifica dello stato della vettura di un pilota (numero pilota, stato gomme, livello carburante nel serbatoio)
- Notifica di fine gara da parte di un pilota (numero pilota, tempo di gara, ragione che ha causato il termine della gara)

Questa interfaccia consente di scambiare col monitor tutti i dati necessari per la visualizzazione delle informazioni sullo stato della gara.

Uno schema della distribuzione e dei canali di comunicazione è riportato nella figura 4.1

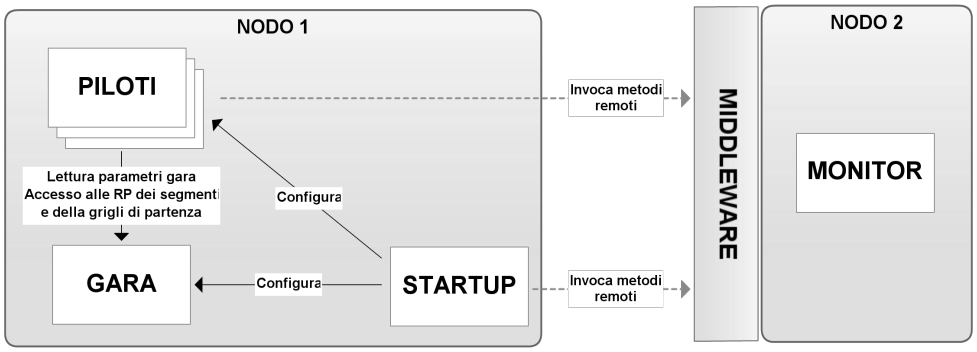

Figura 4.1: Schema di distribuzione e dei canali di comunicazione

CAPITOLO 5

Concorrenza

Gli aspetti del progetto in cui entrano in gioco problemi di natura concorrenziale sono i seguenti:

- Schieramento nella griglia di partenza
- Attraversamento dei segmenti da parte dei piloti

Essi riguardano solamente il nodo F1Engine, in quanto F1ControlPanel consiste solamente nel monitor per la visualizzazione dei dati e non presenta aspetti di natura concorrente relativi al progetto.

5.1 Organizzazione gerarchica dei thread di F1Engine

Durante l'esecuzione del nodo F1Engine i vari thread vengono creati con questa sequenza:

1. All'avvio del nodo F1Engine viene avviato il thread principale StartUp
2. StartUp crea e configura Gara
3. StartUp crea vari thread Pilota, uno per ciascun pilota
4. I thread Pilota si configurano e si schierano sulla griglia di partenza
5. StartUp attende che tutti i thread Pilota siano schierati e che l'utente dia il via alla gara
6. Una volta che la gara è partita i piloti iniziano a gareggiare tra loro

Riguardo all'organizzazione gerarchica i thread Pilota appartengono allo scope del thread StartUp, ed ogni thread Pilota è al di fuori dello scope di ogni altro thread Pilota. Uno schema di tale struttura gerarchica è mostrato in figura 5.1

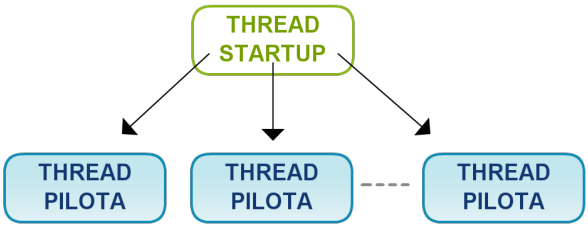

Figura 5.1: Organizzazione gerarchica dei thread di F1Engine

5.2 Schieramento nella griglia di partenza

La griglia di partenza è stata modellata come una risorsa protetta, in modo che fosse possibile potervi accodare i piloti nell'attesa della partenza. Questo è stato necessario in quanto la gara può partire solo una volta che tutti i piloti sono stati schierati.

La risorsa protetta contiene 3 canali d'accesso:

- Uno pubblico sempre aperto chiamato Place_On_Grid
- Uno privato controllato da una guardia chiamato Wait_To_Start, inizialmente chiusa
- Uno pubblico controllato da guardia chiamato Wait_To_Pilots, inizialmente chiusa

Il protocollo d'accesso alla griglia di partenza è il seguente, ed è composto da 2 serie di eventi concorrenti. La prima comprendente il thread StartUp e la seconda comprendente i thread Pilota creati da StartUp:

Thread StartUp:

- Una volta che ha creato tutti i thread Pilota StartUp chiede accesso al canale Wait_To_Pilots, inizialmente con guardia chiusa
- Una volta che tutti i thread Pilota sono schierati la guardia di Wait_To_Pilots viene aperta e StartUp si pone in attesa del segnale di inizio gara da parte dell'utente
- Una volta ricevuto tale segnale viene aperta la guardia Wait_To_Start

Thread Pilota:

- Un thread Pilota viene creato e chiede di schierarsi nella griglia di partenza tramite il canale Place_On_Grid, il numero di piloti in griglia viene aumentato di uno
- Pilota viene poi riaccordato nel canale Wait_To_Start, inizialmente chiuso
- Quando il numero dei piloti schierati è uguale al numero di piloti che devono prendere parte alla gara la guardia di Wait_To_Pilots viene aperta
- Tutti i thread Pilota attendono ora l'apertura della guardia Wait_To_Start da parte di StartUp per poter iniziare a gareggiare

Una schematizzazione di questo protocollo è mostrata in figura 5.2, dove in giallo è indicato il flusso d'esecuzione del thread StartUp e in verde quelli relativi ai thread Pilota.

5.3 Acceso ai segmenti

L'accesso ai segmenti è la parte cruciale di tutto il sistema in quanto tramite essi viene gestita l'intera dinamica della gara, ovvero tempi di percorrenza del circuito, sorpassi e accesso ai box. Fornisce inoltre il meccanismo di ordinamento dei piloti durante la competizione.

Come detto un segmento è composto da una o più corsie gestite mediante una risorsa protetta per garantirne l'accesso in mutua esclusione, esse consentono ai piloti di entrare nello stesso segmento contemporaneamente e di effettuare sorpassi. Se un segmento è composto da una sola corsia i piloti dovranno necessariamente accodarsi in un'unica fila e attraversare il segmento senza alcuna possibilità di effettuare sorpassi.

L'accesso viene modellato mediante una risorsa protetta associata a ciascun segmento, che contiene la rappresentazione delle sue corsie ed il suo stato. Lo stato consiste nell'associare ad ogni corsia del segmento l'istante in cui l'ultimo pilota che vi ha chiesto l'accesso uscirà da essa, nel caso in cui nessun pilota abbia ancora attraversato la corsia a tale istante viene dato valore 0.

Tale rappresentazione permette ad un pilota di capire immediatamente se una corsia è libera o no, in quanto se il valore memorizzato nel relativo stato è minore dell'istante in cui il pilota ne chiede l'accesso vorrà dire che tale corsia sarà completamente libera, in quanto l'ultimo pilota che la occupava sarà già uscito da essa.

Inoltre ad ogni segmento è assegnato un contatore, che eroga dei ticket decrescenti ai piloti ogni volta che ognuno di essi esce dal segmento. In questo modo è possibile ordinare i piloti con relativa semplicità senza dover tenere in considerazione alcun aspetto legato al calcolo dei tempi.

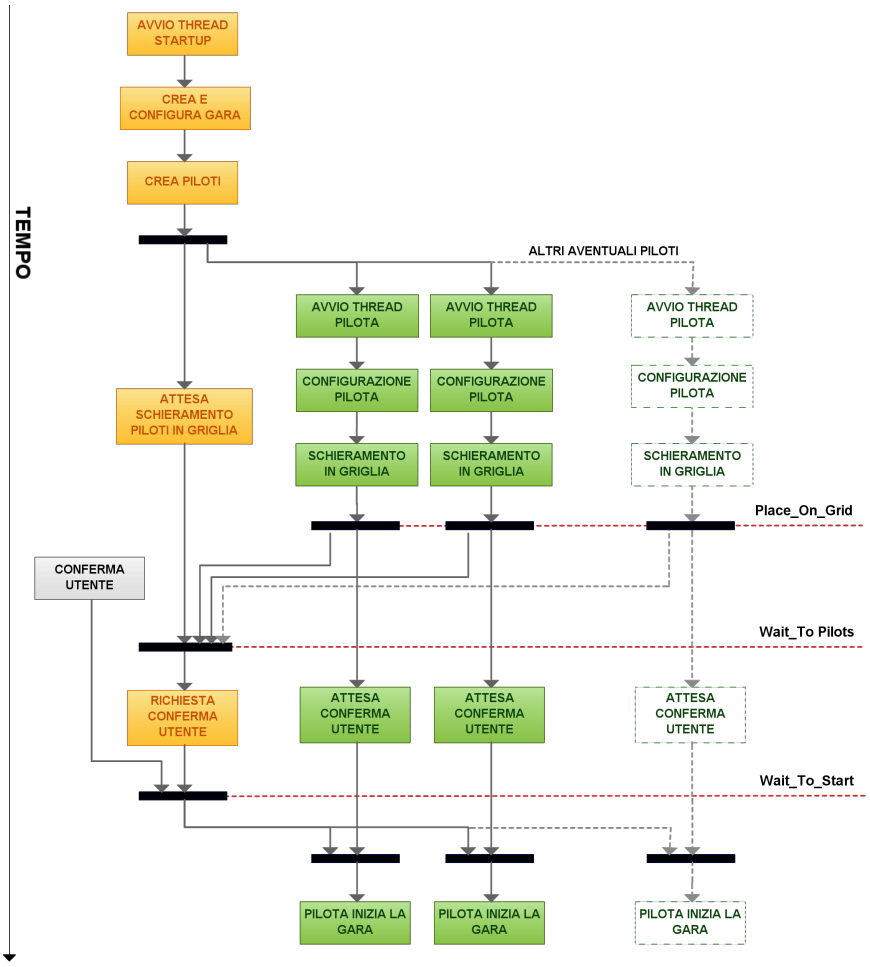

Figura 5.2: Schema del protocollo di schieramento nella griglia di partenza

I piloti vengono infatti ordinati, nel momento in cui escono dai segmenti aventi la fotocellula per il rilevamento dei tempi, in base al giro in cui sono, al segmento in cui si trovano e al ticket in loro possesso. Il caso più complesso si ha quando si confrontano 2 piloti nello stesso giro usciti dallo stesso segmento, rendendo così necessario analizzare l'istante temporale in cui sono usciti. Con l'utilizzo dei ticket sarà invece sufficiente vedere chi dei 2 ha il ticket più alto, chi lo avrà sarà quello uscito prima dal segmento.

Il protocollo d'accesso ai segmenti da parte di un pilota non varia in base alla loro tipologia, restando la medesima per ognuna di esse:

1. Reperimento delle informazioni del segmento
2. Calcolo del tempo di attraversamento ideale T_i in base al suo stato e alle caratteristiche del segmento

3. Accesso in mutua esclusione alla risorsa protetta del segmento
4. Chiamata alla procedura d'ingresso al segmento della risorsa protetta del segmento, che risponderà comunicando il reale tempo di attraversamento T_r calcolato in base al suo stato
5. Aggiornamento dello stato della risorsa protetta
6. Rilascio della risorsa protetta
7. Sospensione del pilota per un tempo T_r
8. Accesso in mutua esclusione alla risorsa protetta
9. Chiamata alla procedura di uscita dal segmento per assegnamento ticket
10. Aggiornamento della risorsa protetta
11. Rilascio della risorsa protetta

T_i non fa nessuna considerazione sugli aspetti relativi alla concorrenza, ma calcola il tempo assumendo che il segmento non sia occupato da nessuna vettura. Questo tempo serve alla risorsa protetta come base di partenza per calcolare T_r

Il procedimento con cui viene calcolato T_r può essere spiegato mediante un esempio, poi facilmente generalizzabile.

Supponiamo quindi che all'istante $T_0 = 0$ un pilota P_c debba attraversare un segmento S con 2 corsie che si trovano nel seguente stato:

- Corsia 1 occupata da un pilota P_a con istante d'uscita $T_{Pa} = 16$
- Corsia 2 occupata da un pilota P_b con istante d'uscita $T_{Pb} = 8$

T_{C1} e T_{C2} sono gli istanti in cui l'ultimo pilota libera rispettivamente la corsia 1 e la corsia 2, quindi si avrà $T_{C1} = T_{Pa}$ e $T_{C2} = T_{Pb}$

Uno schema dello stato del segmento è rappresentato nella figura 5.3

Ora a seconda del valore di T_i relativo al pilota P_a si possono ipotizzare tre possibili scenari, che sono

- $T_i < T_{Pb}$
- $T_{Pb} < T_i < T_{Pa}$
- $T_i > T_{Pa}$

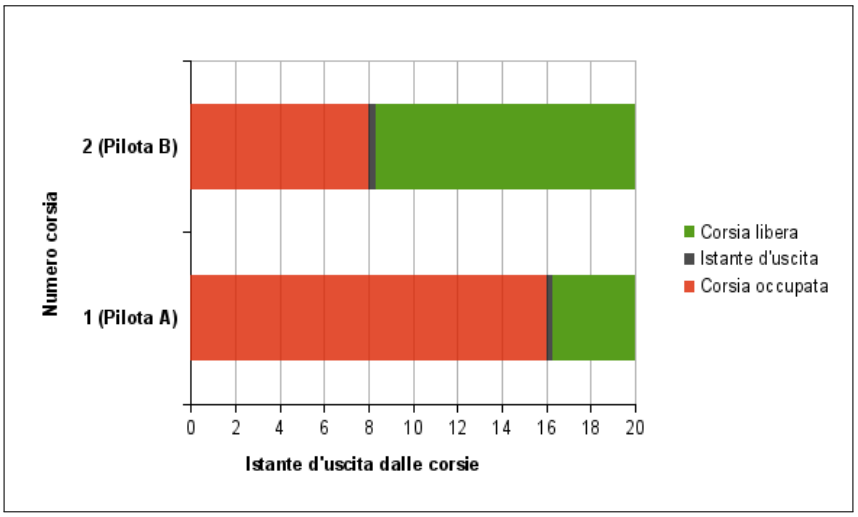

Figura 5.3: Stato del segmento S all'istante in cui il pilota C chiede l'accesso

Occorre anche introdurre un tempo chiamato T_{min}, ovvero il distacco minimo che 2 vetture possono avere stando sulla stessa corsia senza sovrapporsi. L'utilizzo di tale parametro verrà chiarito nella sezione 5.7. Ai fini della spiegazione teorica esso può comunque essere posto uguale a 0, in quanto non influisce sulla scelta della corsia su cui far accodare un pilota.

Vediamo ora come si calcolerà il tempo d'uscita del pilota P_c per ognuno di questi 3 casi.

5.3.1 Caso $T_i < T_{Pb}$

In questo caso il pilota P_c uscirebbe sia prima di P_b che di P_a, questo tuttavia non è possibile in quanto entrambe le corsie sono già occupate da piloti che la liberano dopo di lui, dato che $T_i < T_{Pb} < T_{Pa}$.

Il pilota P_c verrà dunque inserito nella corsia con l'istante d'uscita minore, ovvero C_2, e quindi attraverserà il segmento S usando tale corsia in tempo $T_r = T_{c2} + T_{min}$ uscendo subito dopo il pilota P_2. L'ordine d'uscita dal segmento sarà dunque $P_b -> P_c -> P_a$ (nella realtà P_c riesce a sorpassare P_a solo se $T_r = T_{Pa} + T_{min} < T_{Pa}$, in caso questo non sia verificato ci si ritroverà nella stessa situazione del caso $T_i > T_{Pa}$)

Bisognerà poi aggiornare lo stato del segmento, ponendo $T_{C2} = T_r$ in quanto ora P_c è l'ultimo pilota ad uscire dalla corsia C_2, uno schema del nuovo stato della risorsa protetta relativa al segmento S dopo che il pilota P_c ha effettuato l'accesso è riportato nella figura 5.4.

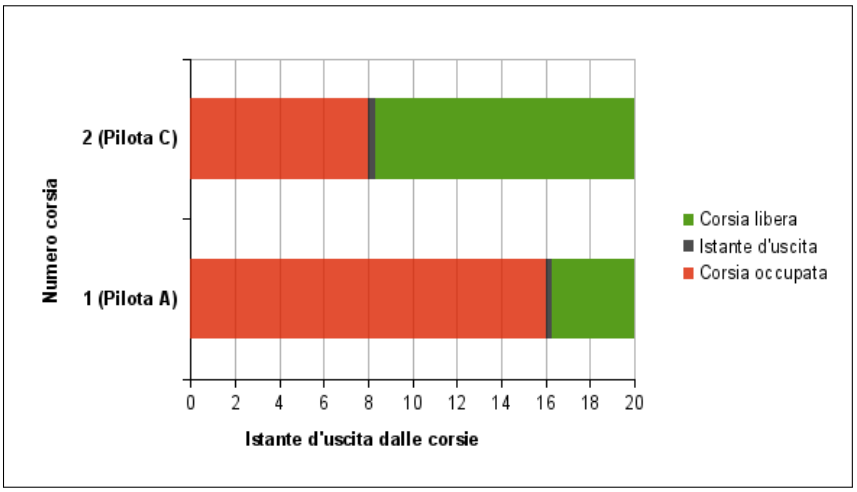

Figura 5.4: Caso 1: Stato del segmento S dopo che il pilota P_C ha effettuato l'accesso nel caso $T_i < T_{Pb}$

5.3.2 Caso $T_{Pb} < T_i < T_{Pa}$

In questo caso il pilota P_c uscirebbe dopo P_b e prima di P_a, questo è compatibile con lo stato del segmento in quando P_c vede la corsia C_2 virtualmente libera poiché il pilota che la impegna esce in ogni caso prima di lui e non comporta nessun ostacolo alla sua percorrenza.

P_c viene dunque inserito nella corsia con il tempo d'uscita minore, ovvero C_2, e dato che $T_i < T_{C2}$ potrà attraversare il segmento in un tempo $T_r = T_i$. Nello stato della risorsa protetta del segmento verrà posto a $T_{C2} = T_r$, mentre T_{C1} resterà invariato. L'ordine d'uscita dal segmento sarà poi $P_b -> P_c -> P_a$, e il pilota P_c avrà effettivamente superato P_a come previsto.

Uno schema del nuovo stato della risorsa protetta relativa al segmento S dopo che il pilota P_C ha effettuato l'accesso è illustrato nella figura 5.5.

5.3.3 Caso $T_i > T_{Pa}$

In questo caso il pilota P_c uscirebbe sia dopo P_a che dopo P_c, per lui il segmento è virtualmente libero in entrambe le corsie in quanto nessuno dei 2 piloti gli crea intralcio. P_c attraverserà dunque il segmento in tempo $T_r = T_i$ utilizzando comunque la corsia con il tempo d'uscita minore, ovvero C_2.

Lo stato della risorsa protetta associata al segmento S sarà modificata ponendo $T_{C2} = T_r$, come rappresentato in figura 5.6, e l'ordine d'uscita sarà come previsto $P_b -> P_a -> P_c$.

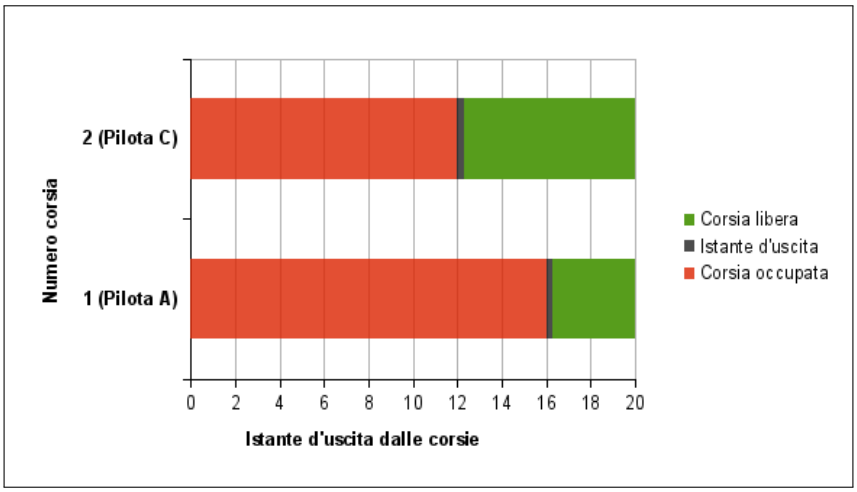

Figura 5.5: Caso 2: Stato del segmento S dopo che il pilota C ha effettuato l'accesso nel caso $T_{Pb} < T_i < T_{Pa}$

5.3.4 Altri casi particolari

Generalizzare l'esempio appena fatto ad altri casi particolari è molto semplice.

Volendo rifarsi al caso in cui ci sia un pilota P_C che voglia attraversare in tempo T_i un segmento S di una sola corsia occupata da P_a con tempo d'uscita T_{Pa}, si possono avere solamente i 2 casi in cui $T_i < T_{Pa}$ o $T_i > T_{Pa}$.

Data la presenza di una sola corsia in ogni caso il pilota P_a verrebbe accodato in ogni caso a P_a, ma varierebbe il metodo con cui verrebbe calcolato il tempo di attraversamento T_r. Nel primo caso attraverserebbe il segmento in tempo $T_r = T_{C1} + T_{min}$, e si imposterebbe $T_{C1} = T_r = T_{C1} + T_{min}$, mentre nel secondo caso il pilota P_C attraverserebbe il segmento in tempo $T_r = T_i$, e si imposterebbe $T_{C1} = T_r = T_i$.

Nel caso in cui un pilota P_C volesse attraversare un segmento S dove una o più corsie non siano state ancora attraversate da alcun pilota, P_C verrebbe accodato nella corsia con instante d'uscita uguale a zero, con relativa modifica allo stato del segmento. Se tali corsie fossero più di una verrebbe scelta quella con indice minore.

Va fatto notare che nel caso in cui 2 piloti volessero uscire da un segmento nello stesso istante, dato che sarebbe comunque necessario stabilire un ordine d'uscita, verrebbe data la precedenza a quello entrato prima nel segmento.

5.4 Stallo

Affinché in un sistema possa esserci possibilità di stallo devono verificarci queste 4 condizioni necessarie e sufficienti:

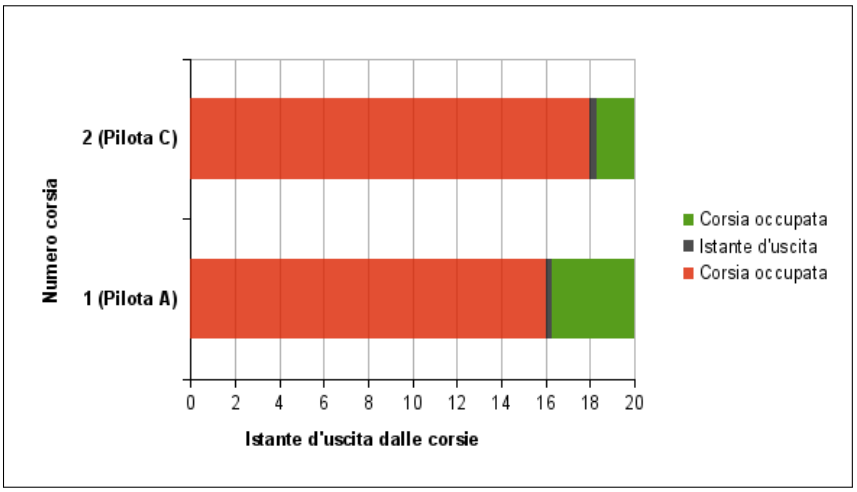

Figura 5.6: Caso 3: Stato del segmento S dopo che il pilota C ha effettuato l'accesso nel caso $T_i > T_a$

1. Accesso esclusivo a risorsa condivisa
2. Inibizione di prerilascio
3. Accumulo di risorse
4. Condizioni di attesa circolare

Cerchiamo di analizzare ora queste 4 condizioni per determinare se nel sistema progettato possa esserci possibilità di stallo.

5.4.1 Accesso esclusivo a risorsa condivisa

Questa condizione è una parte fondamentale del sistema, in quanto non sarebbe possibile garantire uno stato coerente della competizione nel caso in cui 2 processi potessero accedere contemporaneamente allo stessa corsia dello stesso segmento modificandone lo stato.

Riportandoci ad una caso reale sarebbe come se 2 vetture occupassero la stessa posizione nella stessa corsia di un segmento contemporaneamente, cosa ovviamente impossibile e da evitare.

5.4.2 Inibizione del prerilascio

Anche questa condizione non può essere evitata, in quanto sarebbe scorretto consentire ad un processo di potersi appropriare di una risorsa protetta in uso da un altro processo.

Nel caso reale ciò sarebbe paragonabile alla situazione in cui un pilota eseguisse un sorpasso senza aver a disposizione una corsa libera. Anche

questo deve essere vietato e quindi non è possibile rinunciare all'inibizione del prerilascio.

5.4.3 Accumulo di risorse

Nel sistema progettato un processo per poter avanzare con l'esecuzione necessita al massimo dell'acquisizione di una sola risorsa protetta, quindi viene a mancare il caso in cui esso possa accumulare solo una parte di risorse attendendone altre prima di continuare.

5.4.4 Condizioni di attesa circolare

Data la natura circolare di un tracciato di Formula 1 si potrebbe pensare che esistano le condizioni necessarie affinché possa verificarsi attesa circolare. Questo avviene quando si ha una catena chiusa ad anello di processi, dove ogni processo attende una risorsa in possesso del successivo processo.

Nel sistema qui progettato questo evento è impossibile. Dopo la partenza della gara l'unico caso in cui un processo si sospende in attesa che un altro processo liberi una risorsa si ha nell'accesso in mutua esclusione ai segmenti, nel momento in cui si ottengono i reali tempi di attraversamento e se ne aggiorna lo stato. Tuttavia il processo che possiede il lock del segmento non è in attesa di alcun altro processo per continuare, in quanto esegue solamente operazioni matematiche con dati noti e aggiornamenti di variabili di cui possiede già il lock e i nuovi valori. Viene quindi a mancare la catena di processi che mantengono un lock attendendo altri processi.

L'altro momento i cui un processo si sospende è nella simulazione dell'attraversamento di un segmento. Anche in questo caso però non può esserci attesa circolare in quanto il risveglio non è vincolato da nessun altro processo.

Vi sono poi delle sospensioni da parte dei piloti nella griglia di partenza ma anche in questo caso non si ha attesa circolare, in quanto tale operazione (il posizionarsi sulla griglia di partenza) avviene senza alcuna sospensione e senza dover acquisire il lock di qualche risorsa.

In conclusione dato il non verificarsi della condizione di accumulo risorse e dell'attesa circolare possiamo affermare che il sistema progettato è esente da possibili stalli.

5.5 Starvation

La starvation si verifica quando un processo non viene mai eseguito restando sempre in uno stato di wait, a causa dell'utilizzo di politiche di accodamento dei processi non FIFO.

La starvation è quindi evitata implementando il sistema garantendo che tutti gli accodamenti dei processi siano regolati da una politica FIFO.

5.6 Avvio e terminazione

Di seguito verrà spiegato come sono stati gestiti l'avvio e la terminazione dei vari componenti.

Come già detto i 2 nodi devono poter eseguire in modo indipendente, anche se ovviamente è necessario che entrambi siano attivi per avere un completo funzionamento del sistema.

5.6.1 F1ControlPanel

Questo deve essere il primo nodo ad essere eseguito, in quanto come visto in 4.2 si occupa della creazione del lato servente del middleware.

Avvio

L'avvio del nodo non ha particolari aspetti rilevanti, in quanto si tratta semplicemente di un pannello che visualizza le informazioni ricevute dall'altro nodo.

Le operazioni eseguite sono:

- Inizializzazione del middleware
- Creazione del pannello per visualizzare dai ricevuti
- Ascolto per la ricezione dei messaggi

Terminazione

Come per l'avvio anche la terminazione del nodo F1ControlPanel non ha aspetti molto rilevanti. La sua terminazione è infatti decisa dall'utente, in quanto siccome contiene tutte le informazioni della gara è desiderabile che tali informazioni siano visibili anche al suo termine.

Caduta o mancanza del nodo F1Engine

La caduta del nodo F1Engine non ha ripercussioni sull'esecuzione di F1ControlPanel. Tale nodo resterà comunque attivo anche se ovviamente non riceverà più alcun aggiornamento sullo stato della gara. Potrà comunque essere utilizzato

per visualizzare i dati di una nuova gara eventualmente avviata tramite un nuovo nodo F1Engine.

5.6.2 F1Engine

Questo deve essere eseguito solo dopo aver avviato il nodo F1ControlPanel, in quanto necessita del middleware da lui creato per poter inviare i messaggi per le chiamate alle procedure remote.

Avvio

L'avvio del nodo consiste nelle seguenti operazioni:

1. Avvio del metodo main di F1Engine che invoca StartUp
2. StartUp crea Gara, Circuito, Segmenti e i task Pilota
3. Attesa dello schieramento dei piloti
4. Attesa del via da parte dell'utente
5. Partenza dei piloti

Nella figura 5.7 sono chiariti gli scope dei vari componenti lanciati.

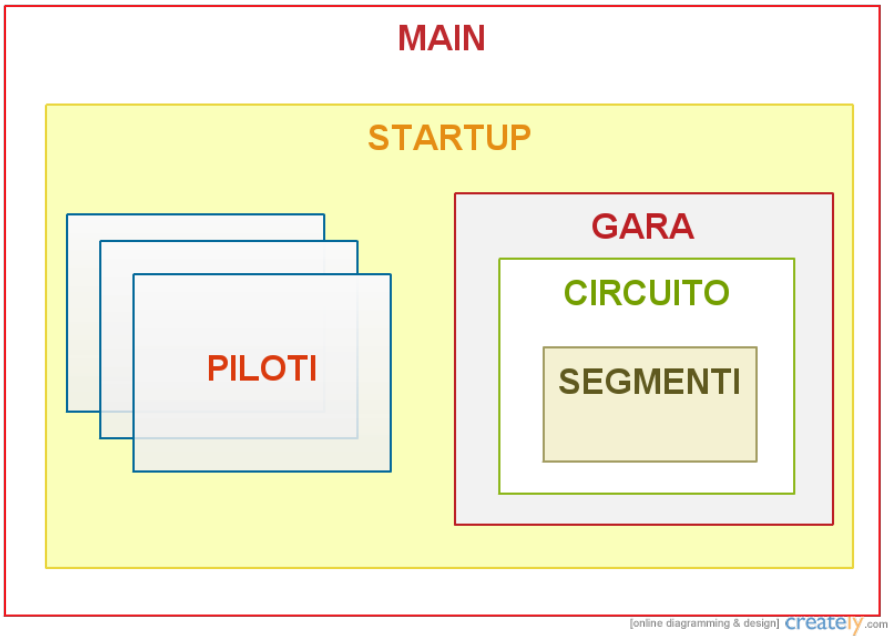

Figura 5.7: Scope del nodo F1Engine

Terminazione

La terminazione inizia quando tutti i piloti concludono la gara secondo questa serie di eventi:

1. L'ultimo pilota conclude la gara e termina la propria esecuzione
2. A questo punto tutti i task creati da StartUp sono terminati
3. Gara non è più riferita da nessun componente al di fuori di StartUp
4. StartUp può terminare la sua esecuzione
5. Main ha invocato solamente StartUp, una volta che esso termina può a sua volta terminare

Il metodo main dunque termina dopo che tutti i piloti hanno concluso la loro gara, facendo così terminare tutto il nodo F1Engine.

Caduta o mancanza del nodo F1ControlPanel

La caduta o la mancanza del nodo F1ControlPanel non pregiudica il corretto svolgimento della gara, semplicemente F1Engine continuerà la propria esecuzione senza però poter mandare alcuna informazione sul proprio stato al monitor, ma visualizzando tali informazioni nel terminale anche se in modo poco pratico e non completo.

5.7 Problemi legati alla granularità del tempo

Nelle prime esecuzione del sistema si sono notate delle problematiche relative alla granularità con cui il sistema operativo gestisce lo scorrere del tempo.

Ciò accadeva quando in un segmento S fossero presenti almeno 2 piloti P_a e P_b con un tempi di risveglio molto vicini, ad esempio $T_{Pa} = T_0$ e $T_{Pb} = T_0 + \epsilon$

Nel caso in cui ϵ sia minore della risoluzione con cui il sistema operativo discretizza il tempo, lo scheduler non è più in grado di distinguere quale pilota vada svegliato prima e quale vada svegliato dopo, causando spesso il loro risveglio in ordine errato.

Questo sostanzialmente comportava il verificarsi di sorpassi non possibili (ad esempio su segmenti con una sola corsia con molti piloti ravvicinati in fila indiana), e sono state pensate 2 possibili soluzioni per evitarli

La prima soluzione consisteva nel creare un sistema di prenotazione in modo che un pilota potesse prenotare la mutua esclusione della risorsa protetta del segmento successivo. Se la risorsa protetta associata ad un segmento riceveva la richiesta di mutua esclusione da parte di un pilota diverso dal primo in lista nell'elenco delle prenotazioni, allora tale pilota doveva

essere riaccordato nella stessa coda. Tale processo di riaccodamento proseguiva finché il pilota che chiedesse l'accesso in mutua esclusione alla risorsa protetta fosse effettivamente quello in cima alla lista di prenotazione.

Tale strategia, corretta dal punto di vista logico, comporta l'accumularsi di ritardi dovuti alla grande capacità di calcolo richiesta per eseguire un riaccodamento, in quanto si tratta di una operazione molto onerosa.

La soluzione alternativa, che è quella che è stata poi implementata nel sistema, è molto più semplice ed è basata sul fatto che in una reale competizione i Formula 1 due piloti su una stessa corsia non possono avere un distacco inferiore ad un certo limite, in quanto ciò significherebbe che essi sono sovrapposti.

Anche nel sistema progettato è stato quindi inserito un distacco minimo T_{min} tra i piloti, che garantisce che 2 o più di essi non si risveglino mai in istanti così vicini da non essere ben gestiti dallo scheduler. Il distacco minimo è stato scelto di 0.05 secondi, sia perché questa è pressapoco la distanza minima che 2 vetture possono avere senza sovrapporsi, sia perché varie simulazioni di gare effettuate sull'ambiente utilizzato per i test non hanno riportato sorpassi indesiderati con T_{min} impostato a tale valore o valori superiori.

CAPITOLO 6

Implementazione

La presenza di 2 nodi distinti e con diverse funzioni ha portato a diverse considerazioni riguardo alla scelta dei linguaggi di programmazione da utilizzare nell'implementazione.

6.1 Nodo 1

Come già detto il nodo 1 contiene la gara, i piloti e il relativo StartUp e gestisce tutti gli aspetti legati alla concorrenza e alla temporizzazione.

É stato quindi scelto di implementare questo nodo mediante il linguaggio Ada, che assicura un buon supporto sia riguardo alla concorrenza che riguardo alla distribuzione.

6.2 Nodo 2

Il nodo 2 ha caratteristiche del tutto diverse rispetto a quelle del nodo 1, in quanto non gestisce nessun aspetto legato alla concorrenza ma solamente quelli legati alla visualizzazione grafica.

Per questi motivi è stato scelto di utilizzare Java per la sua implementazione, in quando garantisce un buon supporto all'aspetto grafico con la presenza di vari tool ne consente la progettazione mediante semplice drag-and-drop. Questo linguaggio garantisce inoltre un buon supporto alla distribuzione.

6.3 Middleware

Come Middleware è stato scelgo di utilizzare Corba, per via della sua diffusione e per il buon supporto dato ad entrambi i linguaggi utilizzati.

6.4 Accorgimenti

In fase di implementazione sono stati tuttavia necessari degli accorgimenti che hanno leggermente modificato la struttura iniziale del progetto, riportato in figura 4.1.

I problemi sono sorti quando più piloti utilizzano contemporaneamente il middleware Corba per inviare messaggi al monitor, in quando ci sono state delle difficoltà nell'impostare Corba con una tasking policies che supportasse più thread per il lato Ada.

Per non superare ulteriormente il limite delle ore da dedicare al progetto si è scelto quindi di modificarne leggermente la struttura, introducendo una nuova entità chiamata Sender.

Essa ha il compito si serializzare i messaggi inviati al monitor, in modo che non ve ne siano più di spediti contemporaneamente. La sua realizzazione è stata modellata come una risorsa protetta, con un procedura esposta per ogni messaggio che il pilota o lo StartUp possono inviare al monitor.

Il fatto che si tratti di procedure garantisce che esse siano eseguite in mutua esclusione e quindi serializzate. Esse comunque devono solo consegnare il messaggio al middleware, usando ugualmente una politica best-effort e non introducono ritardi evidenti in fase di esecuzione.

Lo stesso problema si è ripercosso anche nel nodo 2, in quanto poteva capitare che venissero chiamati più metodi del monitor allo stesso tempo causando degli errori nella visualizzazione. Anche questo è stato risolto eseguendo tutti i metodi per l'aggiornamento dell'interfaccia grafica in mutua esclusione, e anche in questo caso non ci sono ripercussioni in quanto nel peggiore dei casi l'unico inconveniente che si avrebbe sarebbe un leggero ritardo nell'aggiornamento dell'interfaccia grafica, difficilmente rilevabile e che non ne compromette la consistenza.

Uno schema con l'architettura dopo l'introduzione dell'entità Sender è rappresentato nella figura 6.1

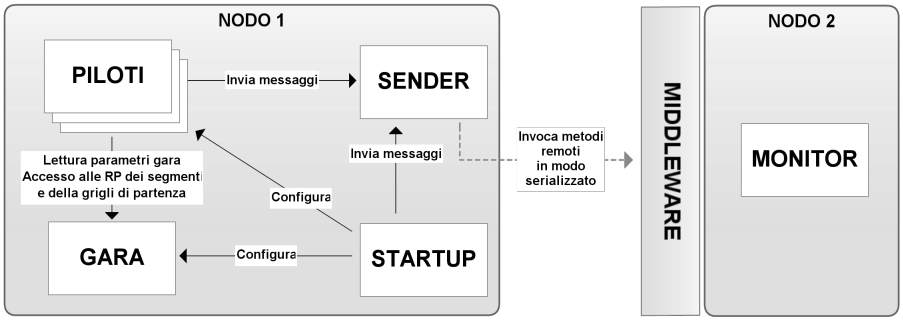

Figura 6.1: Schema di distribuzione dopo l'aggiunta dell'entità Sender

CAPITOLO 7

Compilazione, configurazione ed esecuzione

7.1 Ambiente d'esecuzione

Il progetto è stato testato sia su sistema operativo Ubuntu 13.04 che su sistema operativo virtualizzato Ubuntu 10.04.

Oltre alle librerie standard del sistema operativo sono necessari i seguenti pacchetti:

- polyorb-server
- libpolyorb2
- libpolyorb1-dev
- oracle-jdk-7 o una distribuzione Java equivalente

7.2 Compilazione

Nel progetto sono presenti 2 script, chiamati CompileEngine.sh e CompilePanel.sh che servono per compilare rispettivamente la partizione F1Engine e la partizione F1ControlPanel in modo indipendente.

7.3 Configurazione

Il progetto prevede tutta una serie di file di configurazione per personalizzare la competizione Questi file sono salvati nella cartella conf di F1Engine, e possono essere di 4 differenti tipi:

- file .trk, contenuti nella sotto cartella circuits_set

- file .car, contenuti nella sotto cartella cars_set
- file .plt, contenuti nella sotto cartella pilots_set
- file .conf, contenuti nella cartella conf

7.4 File .trk

Ognuno di questi file descrive un circuito su cui è possibile svolgere una competizione, ogni riga contiene i parametri che descrivono un singolo segmento (quindi la prima riga contiene i parametri del primo segmento, la seconda quelli del secondo segmento e così via)

I parametri per ogni segmento sono:

- Segment_Type (dec, acc, const, box)
- Length (Integer 1..∞)
- Speed (Float 1..∞)
- Num_Lane (Integer 1..2)
- Has_Time_Check (Positive 0..4)

Segment_Type

Identifica il tipo di segmento (acc => accelerazione, const => curva, dec => decelerazione, box => box). I box devono occupare i primi 3 segmenti (decelerazione, box e accelerazione) e devono avere complessivamente una lunghezza pari al segmento di partenza (che è alla terza posizione).

Length

Indica la lunghezza del segmento in metri.

Speed

Indica la velocità (m/s) massima in caso di un segmento di accelerazione, la velocità di uscita in caso di un segmento di decelerazione o la velocità di percorrenza in caso di segmento a velocità costante. Se un segmento di accelerazione o di decelerazione precede un segmento a velocità costante, dovranno avere l'attributo Speed uguale.

Num_Lane

Indica il numero di corsie del segmento.

Has_Time_Check

Indica se il segmento viene usato per la rilevazione dei tempi, devono essere in tutto 4. Indica anche il numero dell'intermedio.

7.5 File .car

Questi file di configurazione stabiliscono le possibili vetture che possono essere assegnate ai vari piloti. Ogni file definisce una vettura, e viene indicato un suo parametro per ogni riga col seguente ordine:

- Manufacturer : String;
- Coeff_Acceleration : (Integer 1..10)
- Coeff_Deceleration : (Integer 1..10)
- Max_Speed: (Integer 1..100)
- Coeff_Roadholding: (Integer 1..10)
- Coeff_Tire_Wear: (Integer 1..10)
- Consuption: (Float 1..∞)
- Max_Fuel_Level: (Float 1..∞)
- Reliability: (Integer 1..100)

Manufacter

Indica la casa costruttrice della vettura.

Coeff_Acceleration

Indica il coefficiente di accelerazione, che è massimo se vale 10 e minimo se vale 1.

Coeff_Deceleration

Indica il coefficiente di decelerazione, che è massimo se vale 10 e minimo se vale 1.

Max_Speed

Indica la velocità massima che l'auto può raggiungere in metri al secondo.

Coeff_Roadholding

Indica il coefficiente di tenuta, che è massimo se vale 10 e minimo se vale 1.

Coeff_Tire_Wear

Indica il coefficiente di usura gomme, più alto è e meno si consumano.

Consuption

Indica il consumo dell'auto, misurato in litri per chilometro.

Max_Fuel_Level

Indica la capienza del serbatoio in litri.

Reliability

Indica l'affidabilità della vettura.

7.6 File .plt

I file .plt descrivono un pool di piloti da cui è possibile scegliere quelli che effettivamente parteciperanno alla competizione.

Ogni file descrive un diverso pilota, e in ogni riga si specifica una sua skill con il seguente ordine:

- Name : (String);
- Number : (Positive)
- Skill_Acceleration : (Integer 1..10)
- Skill_Deceleration: (Integer 1..10)

Number

Indica il numero del pilota.

Name

Indica il nome del pilota.

Skill_Acceleration

Indica la skill di accelerazione, che è massimo se vale 10 e minimo se vale 1.

Skill_Deceleration

Indica la skill di decelerazione, che è massimo se vale 10 e minimo se vale 1.

7.7 File .conf

Ognuno di questo file descrive una competizione. Nella prima parte si specificano, uno per riga, le seguenti opzioni della competizione:

- Nome del file con la configurazione del circuito
- Numero di giri previsti (Positive)
- Condizioni meteo (dry, wet)

Nella seconda parte del file invece si elencano tutti i piloti che partecipano alla competizione, specificando quale vettura è a loro assegnata, la quantità di benzina a inizio gara e i giri in cui entrare a i box per cambiare gli pneumatici.

Per ogni riga verranno dunque indicati:

- Nome del file del pilota
- Nome del file della vettura assegnata al pilota
- Litri di benzina caricati a inizio gara (Positive)
- Elenco dei giri in cui fermarsi ai box (Elenco di Positive)

Nel progetto sono già inclusi vari file di configurazione, con la quale è possibile testare il tutto. Il file test_1lane.conf contiene una configurazione che avvia una competizione senza soste ai box in cui il circuito è interamente ad una corsia, e nonostante la presenza di auto lente nelle prime posizioni si può notare come esse facciano da tappo in quanto su tale tracciato non è possibile eseguire sorpassi.

Il file test_2lane.conf avvia invece una competizione con soste ai box in un circuito con la presenza alcuni di segmenti con 2 corsie, dove si può notare l'esecuzione di sorpassi ai danni dei piloti più lenti.

7.8 Esecuzione

Per semplificare la fase di test si ipotizza che entrambi i nodi siano eseguiti sullo stesso terminale, in modo che lo IOR del server dei nomi possa essere condiviso mediante un semplice file di testo accessibile da entrambi i nodi.

Nel progetto sono presenti 2 script per lanciare il software.

Per prima cosa bisognerà avviare lo script con nome StartControlPanel.sh che ha il compito di avviare il pannello per la visualizzazione delle statistiche e il server dei nomi per il la distribuzione.

Una volta avviato il pannello sarà necessario lanciare lo script StartEngine.sh, che si occuperà di avviare il nodo F1Engine usando lo IOR precedentemente salvato dal nodo F1ControlPanel. Lo script StartEngine.sh deve inoltre ricevere in ingresso una stringa contenete il nome del file .conf desiderato.

Un esempio di una possibile esecuzione partendo dalla cartella Workspace del progetto è la seguente:

```
cd F1ControlPanel
sh StartControlPanel.sh
cd ..
cd F1Engine
sh StartEngine.sh test_2lane.conf
```