Railway Simulation

Moreno Ambrosin

Università degli studi di Padova

Facoltà di Scienze MM. FF. NN.

Corso di laurea in Informatica

Settembre 2013

Indice

Descrizione

Simulatore software concorrente e distribuito, per la simulazione di un sistema ferroviario composto da:

- *treni*, appartenenti a diverse categorie.
- *viaggiatori*, che operano azioni elementari come acquisto di *biglietti*, salita a bordo e discesa dai treni, attesa presso il *binario*.
- *stazioni* composte da
  - *binari* interni di sosta per i *treni*;
  - *piattaforme* di attesa per i *viaggiatori*;
  - una *biglietteria* interna, accessibile ai *viaggiatori*;
  - un *pannello informativo*.
- *segmenti* che collegano le diverse *stazioni*.
- un *controllo centrale* che mantiene lo stato di ciascun *treno* e di ciascun *viaggiatore* in transito.

Requisiti di alto livello (1)

Treno

- (`RT1`) Ciascun *treno*, appatiene ad una delle seguenti categorie: `FB` o `REG`.
  - (`RT1.1`) La categoria `FB` ha priorità più alta rispetto a `REG` nell'accesso a *stazione*, *piattaforme* e *binari*.
- (`RT2`) Ciascun *treno* è caratterizzato da un identificativo univoco, e possiede capacità e velocità massime, e mantiene stazioni di partenza e destinazione.
- (`RT3`) Ciascun *treno* effettua continuamente un tragitto simmetrico di andata e ritorno, definito da un *percorso* costituito da più *tappe*
  - (`RT3.1`) Il tragitto di ciascun *treno* è scandito da una *tabella oraria*, che definisce,per ciascuna *tappa*, l'orario di partenza.

Requisiti di alto livello (2)

- (RT3.2) Ciascuna *tappa* definisce:
  - *stazioni* di partenza e destinazione
  - *piattaforme* di partenza e destinazione
  - azione da compiere all'arrivo presso la destinazione, tra STOP e PASS
  - il prossimo *segmento* da utilizzare.
- (RT4) I *treni* di tipo FB sono a prenotazione.

Viaggiatore

- (RV1) Ciascun *viaggiatori* possiede una *stazione* di partenza ed una di destinazione.
- (RV2) Per poter salire a bordo di un *treno*, ciascun *viaggiatori* deve prima acquistare un *biglietto* presso la *biglietteria* della *stazione* di partenza.

Requisiti di alto livello (3)

- (RV3) Una volta arivato a destinazione, ciascun *viaggiatore* attende un tempo casuale prima di ritornare alla stazione di partenza.
- (RV4) Il viaggio di ciascun *viaggiatore* può comprendere cambi di *treno*.

Segmento

- (RS1) Ciascun *segmento* è ha un identificativo univoco, una lunghezza e una velocità massima di percorrenza; esso collega esattamente due *stazioni* diverse.
- (RS2) Un *segmento* ha percorrenza bidirezionale; inoltre più *treni* possono percorrere il *segmento* nello stesso senso di marcia (percorrenza multipla).

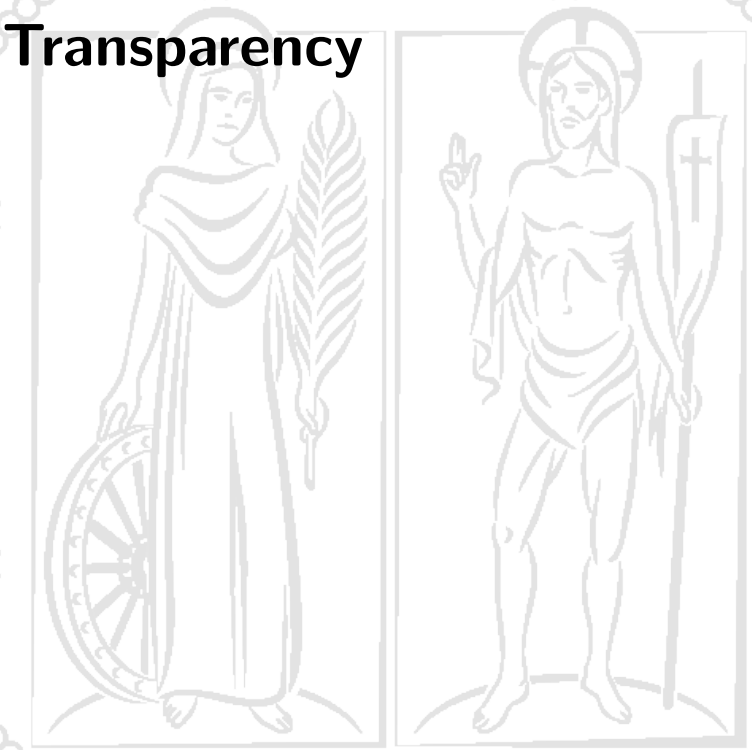
Requisiti di alto livello (4)

Stazione

- (RST1) Ciascuna *stazione* è caratterizzata da un identificativo univoco.
- (RST2) Ciascuna *stazione* contiene:
  - Un certo numero N di *binari* di sosta per i *treni*, a percorrenza bidirezionale e ad accesso mutuamente esclusivo.
  - Un certo numero N di *piattaforme* per l'attesa dei passeggeri.
  - Un *pannello informativo* che riporta informazioni su *treni* in arrivo, in transito, e che hanno appena superato la *stazione*.
  - Una biglietteria accessibile ai *viaggiatori*, per l'acquisto del biglietto necessario.

Caratteristiche desiderabili

- **Distribution Transparency**
- **Openess**
- **Scalability**



Caratteristiche desiderabili - Distribution Transparency

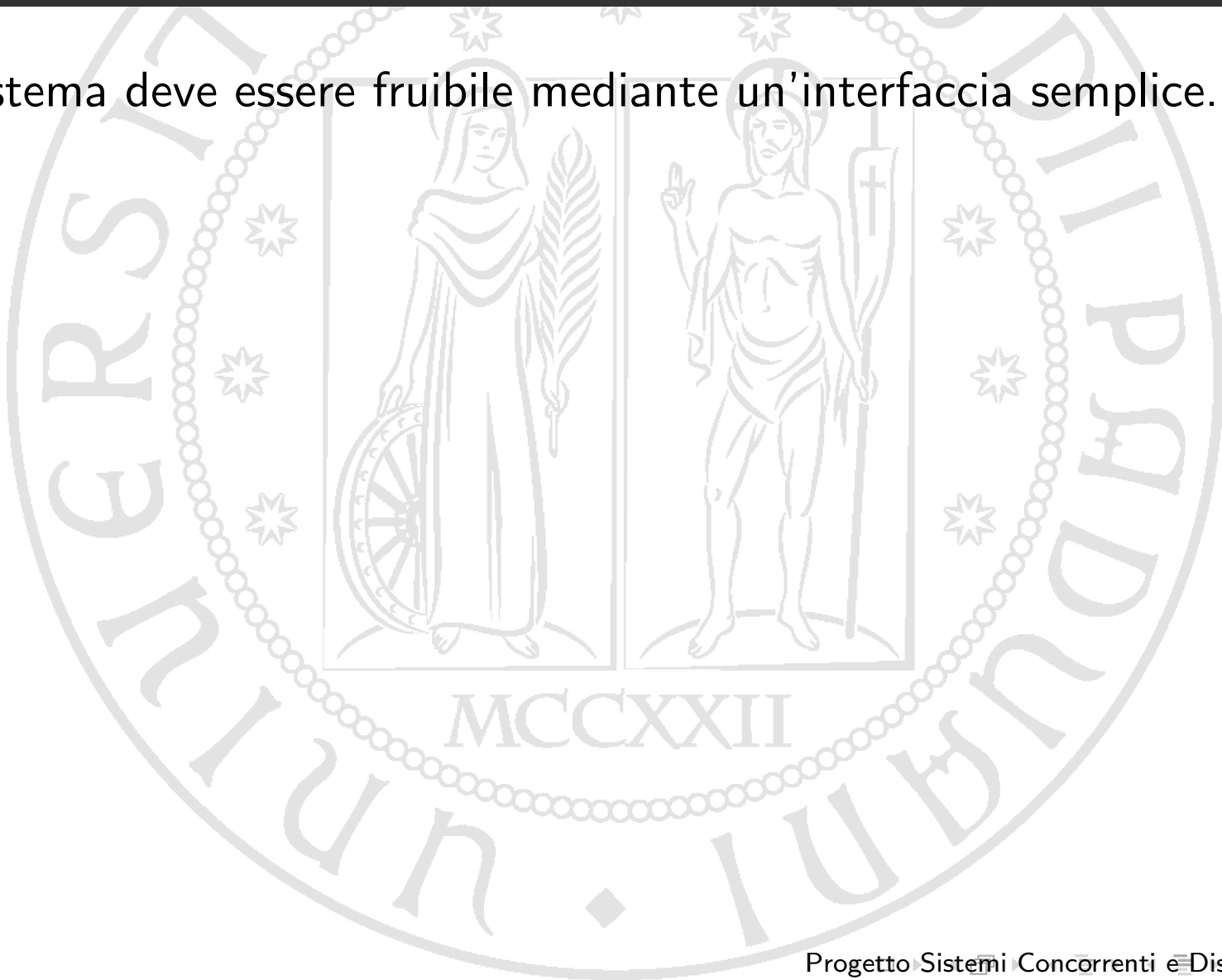
Il sistema appare all'utilizzatore come un sistema monolitico.

- Deve fornire un unico metodo per accedere alle risorse (*Access Transparency*).
- Deve nascondere all'utente la locazione fisica delle risorse (*Location Transparency*) e la loro migrazione (*Migration Transparency*).
- Deve nascondere all'utilizzatore finale la possibilità che le risorse siano replicate (*Replication Transparency*).
- Deve nascondere all'utente l'eventuale accesso concorrente ad una stessa risorsa condivisa (*Concurrency Transparency*).
- Deve rendere il sistema trasparente rispetto ai malfunzionamenti (*Faliure Transparency*).

Il grado di trasparenza da adottare dipende dal problema.

Caratteristiche desiderabili - Openess

Il sistema deve essere fruibile mediante un'interfaccia semplice.



Scelete progettuali

Il grado di distribuzione da utilizzare va scelto con cura.

- Pro e contro per diverse soluzioni possibili.
- Analisi delle implicazioni.

Analisi di tre soluzioni a grado di distribuzione diverso.

Soluzione A - Descrizione (1)

Primalla Soluzione valutata: tutte le entità risiedono su un diverso nodo di calcolo.

- Ciascuna *stazione*, *treno* e *viaggiatore* esegue su uno specifico nodo di calcolo.
- Le *stazioni* espongono un'interfaccia remota per l'iterazione con *treni* e *viaggiatori*.
- Unica entità *biglietteria*, accessibile ai *viaggiatori* tramite interfaccia esposta dalle *stazioni*.
- Unice entità di *controllo centrale*, che espone interfaccia remota a *treni* e *stazioni*.
- Server dei nomi presso il quale le entità si registrano all'avvio del sistema.

Soluzione A - Descrizione (2)

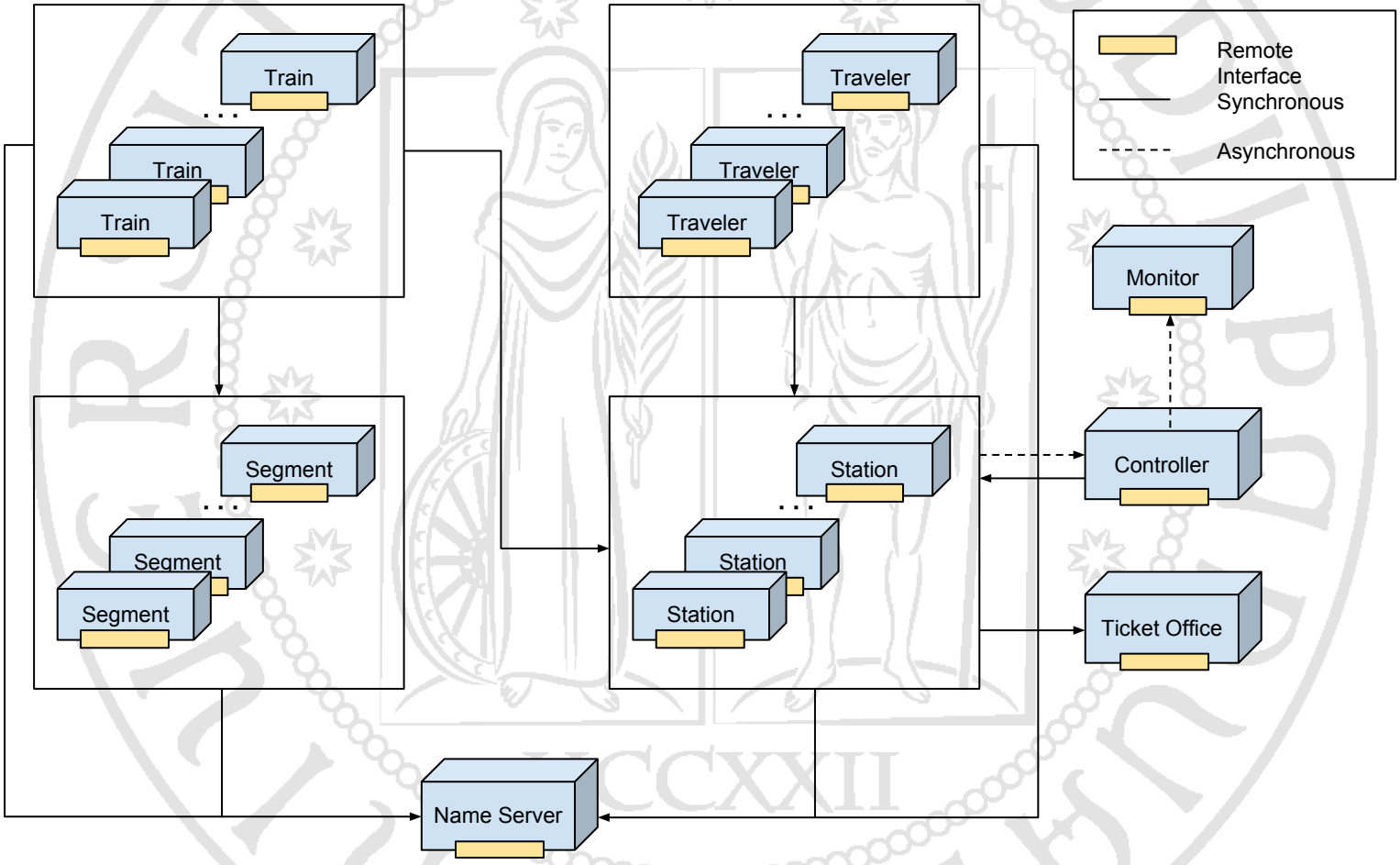


Figura: Grafico informale che illustra l'architettura di distribuzione di massima dellalla Soluzione presentata.

Soluzione A - Valutazione (1)

PRO:

- Sistema scalabile in dimensione rispetto a *treni*, *stazioni* e *viaggiatori*.
- Robusto rispetto a malfunzionamenti di *treni*, *stazioni* e *viaggiatori*, con impatto ridotto sull'intero sistema.
- *Distribution Transparency* rispetto alle entità *Monitor*.

Soluzione A - Valutazione (2)

CONTRO:

- Sincronizzazione in distribuito.
- Differenza tra i clock fisici delle macchine che compongono il sistema possono creare inconsistenze.
- La *biglietteria* deve mantenere tutta l'informazione relativa alla topologia del sistema ferroviario, e allo stato di prenotazione dei *treni*.
- Rischio di comunicazioni errate, a causa dell'inaffidabilità della rete; conseguente perdita di performance.
- Elevato traffico di rete.
- Terminazione e avvio del sistema complesse.

Soluzione B - Descrizione (1)

Secondalla Soluzione valutata: entità di simulazione in un unico nodo di calcolo.

- Solo *controllo centrale* e *monitor* distribuiti.

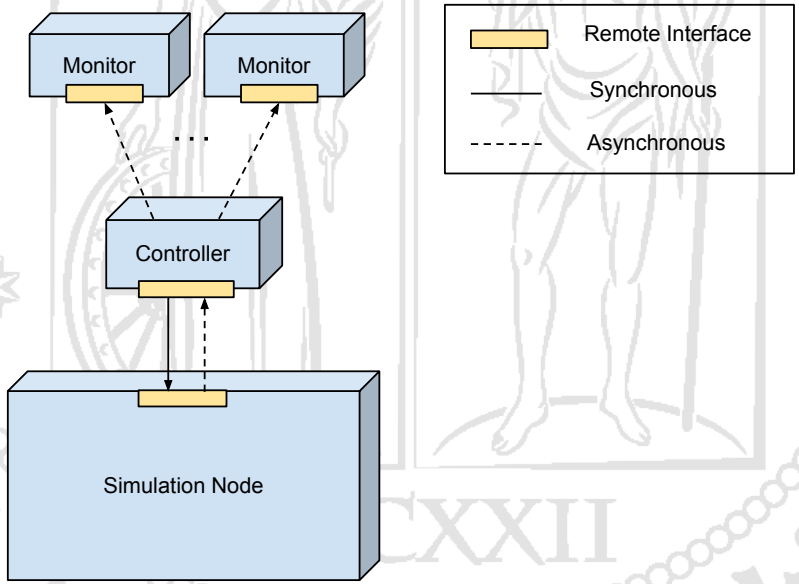


Figura: Grafico informale che illustra l'architettura di distribuzione di massima dellalla Soluzione presentata.

Soluzione B - Valutazione (1)

PRO:

- Seplicità implementativa.
- Interazione tra entità risolta in locale mediante meccanismi di concorrenza.
- Utilizzo di un unico riferimento temporale; assenza di inconsistenze dovute a clock fisici non sincronizzati.
- Semplicità di Avvio e Terminazione dell'intero sistema.
- Comunicazione tra le entità di simulazione affidabili.
- Traffico di rete minimo.

Soluzione B - Valutazione (1)

CONTRO:

- Di scarso interesse.
- Fragile rispetto a malfunzionamenti.
- Non scalabile.
- Conoscenza su Topologia e Stato in un unico luogo.
- Carico computazionale elevato sul singolo nodo di simulazione.

Soluzione Adottata - Descrizione (1)

- Soluzione intermedia che mitiga le problematiche riscontrate nelle Soluzioni A e B.
- Introduzione di *regioni* per distribuire il carico computazionale su nodi diversi
- Ciascuna *regione* risiede su un proprio nodo di calcolo.
- In ciascuna *regione* vi sono un certo numero di *stazioni*, di *segmenti*, di *treni* e di *viaggiatori*.
- Necessario l'utilizzo di un *Server dei Nomi* per lookup degli indirizzi delle varie *regioni*.

Soluzione Adottata - Descrizione (2)

- Introduzione di più livelli di *biglietterie*:
  - *interne alle stazioni:* Forniscono da interfaccia per accedere alle biglietterie regionali.
  - *regionali:* Hanno conoscenza della topologia interna a ciascuna regione; si occupano della creazione dei *biglietti*.
  - *centrale:* Mantiene lo stato di prenotazione dei *treni* `FB`
- L'entità di *controllo centrale* è centralizzata, e raccoglie lo stato della simulazione; esso si occupa inoltre delle procedure di Avvio e Terminazione del sistema.

Soluzione Adottata - Valutazione (1)

PRO:

- L'utilizzo delle *regioni* è un buon compromesso per controllare la complessità del sistema.
- Il sistema è scalabile rispetto alla dimensione.
- Distribuisce la conoscenza relativa alla topologia del sistema ferroviario a livello di regioni (non più centralizzata).
- La *biglietteria* centrale mantiene solo lo stato relativo alle prenotazioni dei *treni* `FB`.

Soluzione Adottata - Valutazione (2)

CONTRO:

- Rimane il problema dei clock fisici non sincronizzati, anche se con impatto minore rispetto alla Soluzione A.
- Maggiore complessità di Avvio e Terminazione del sistema rispetto alla Soluzione B.
- Meno robusta ai fallimenti rispetto alla Soluzione A .
- Maggiore traffico di rete rispetto alla Soluzione B

Soluzione Adottata - Requisiti

Requisiti aggiuntivi

- (`RT5`) I *treni* possono compiere tragitti che attraversano più *regioni*.
- (`RV5`) I *viaggiatori* possono viaggiare attreverso più *regioni*.

Soluzione Adottata - Problematiche

1. Come rappresentare il trasferimento di *treni* e *viaggiatori* tra *regioni*?
2. Come gestire le problematiche relative al tempo?
3. Come organizzare avvio e terminazione del sistema?

Soluzione Adottata - Problematiche - Trasferimento remoto (1)

- Sono introdotte *stazioni di gateway*.
- Luogo unico dal quale i *treni* possono uscire dalla regione corrente e accedere a un'insieme di *regioni*.
- Una volta che un *treno* accede ad una *stazione di gateway* e viene trasferito su una regione R', esso riprenderà il suo *percorso* dalla *stazione di gateway* corrispondente presso R'.

Soluzione Adottata - Problematiche - Trasferimento remoto (2)

- Sono posti dei vincoli sull'utilizzo di *stazioni di gateway*:
  - Siano g ed g' *stazioni di gateway* appartenenti rispettivamente a due *regioni* R ed R'. Allora se da g è possibile raggiungere g' in modo diretto (e viceversa), allora $!\exists g''$ appartenente a R tale per cui da g'' si può raggiungere g'.
    $\Rightarrow$ In questo modo, dal punto di vista dei *treni*, g e g' saranno la stessa *stazione* logica.

Soluzione Adottata - Problematiche - Trasferimento remoto (3)

- I *binari* di sosta per i *treni*, interni a ciascuna *stazione di gateway*, sono unidirezionali. Inoltre, ciascuna *stazione di gateway* avrà $p_1, p_2, ..., p_m$ *binari* per permettere l'uscita dalla *regione* corrente, e $p_{m+1}, p_{m+2}, ..., p_k$ *binari* per consentire l'arrivo dei *treni*provenienti da altre stazioni. I *binari* $p_1, p_2, ..., p_m$ avranno il loro corrispettivo presso le altre *stazioni di gateway* connesse per permettere l'arrivo dei treni provenienti dalla *regione* corrente. Il numero di *binari* (e *piattaforme*) date $g_1, g_2, ..., g_n$ *stazioni di gateway*, è $sum_{i=1}^{n} m_i$.
  $\Rightarrow$ Questo permette di mantenere locale alle regioni la competizione per l'accesso al *binario*.