

# Uno schema di coordinamento vagamente ispirato alla Movement Authority per ERTMS/ETCS LV 1 e LV 2

**Andrea Ceccarelli**  
andrea.ceccarelli@unifi.it

Progetto per il corso di Sistemi Operativi

## NOTA

QUANTO DI SEGUITO HA  
SOLO SCOPO ESPLICATIVO

IL TESTO DEL PROGETTO, A  
CUI E' RICHIESTO  
ATTENERSI IN MODO  
SCRUPOLOSO, E' PRESENTE  
SUL SITO DEL CORSO

# Un po' di background



Cosa è l'ERTMS/ETCS e la Movement Authority  
(<https://it.wikipedia.org/wiki/ERTMS>)

**ERTMS/ETCS (European Rail Traffic Management System/European Train Control System)** è un sistema di gestione, controllo e protezione del traffico ferroviario e relativo segnalamento a bordo

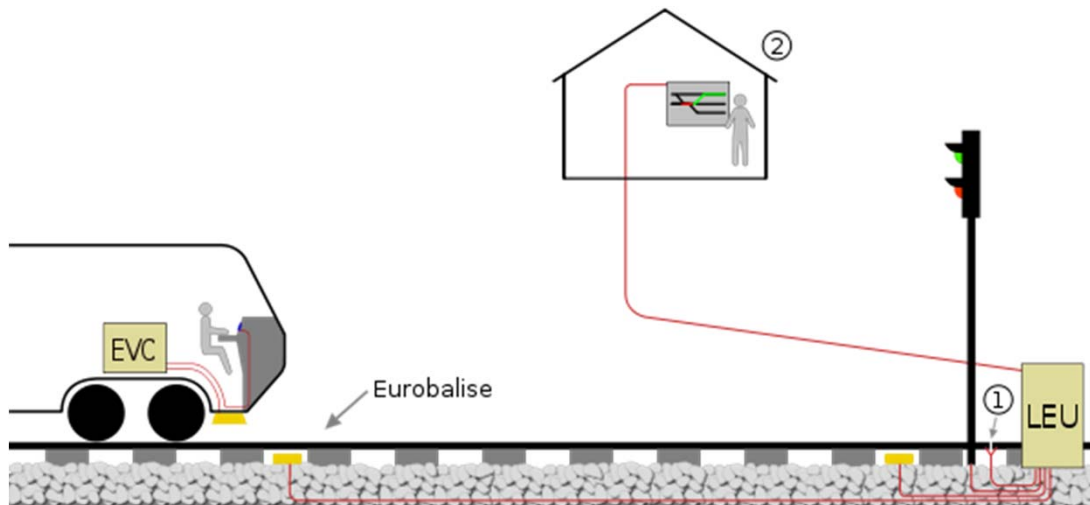
- Scopo: garantire l'interoperabilità dei treni soprattutto sulle nuove reti ferroviarie europee ad Alta velocità.

# ETCS Livello 1

(<https://it.wikipedia.org/wiki/ERTMS>)

Autorizzazione al movimento (*Movement Authority, MA*) e le corrispondenti informazioni sul percorso vengono trasmesse al treno e visualizzate in cabina al macchinista mediante **boe (balises)**

- Maggiori dettagli: <https://it.wikipedia.org/wiki/ERTMS>

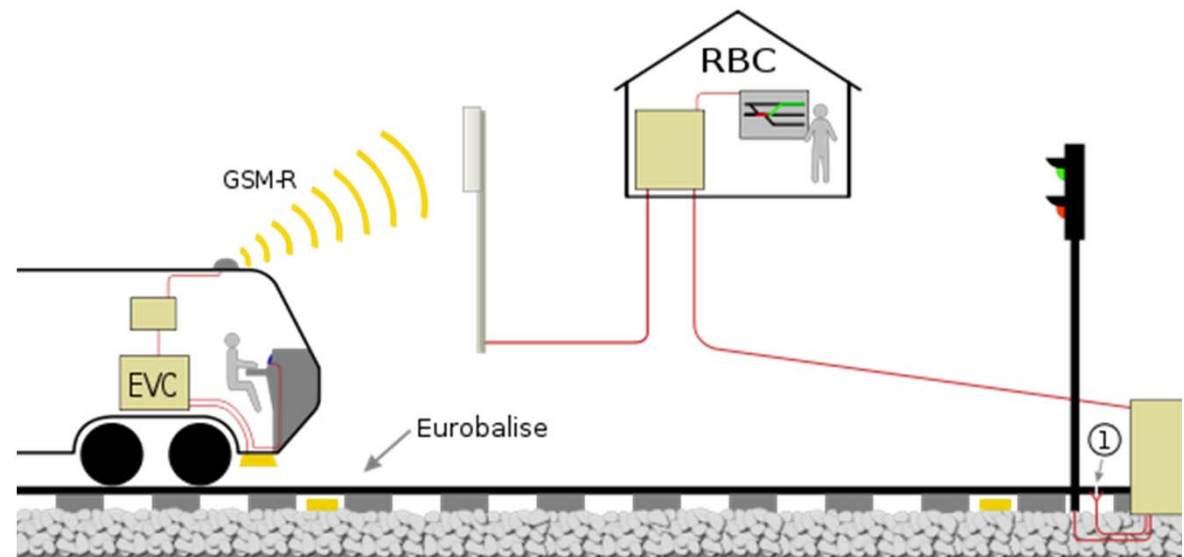


## ETCS Livello 2

(<https://it.wikipedia.org/wiki/ERTMS>)

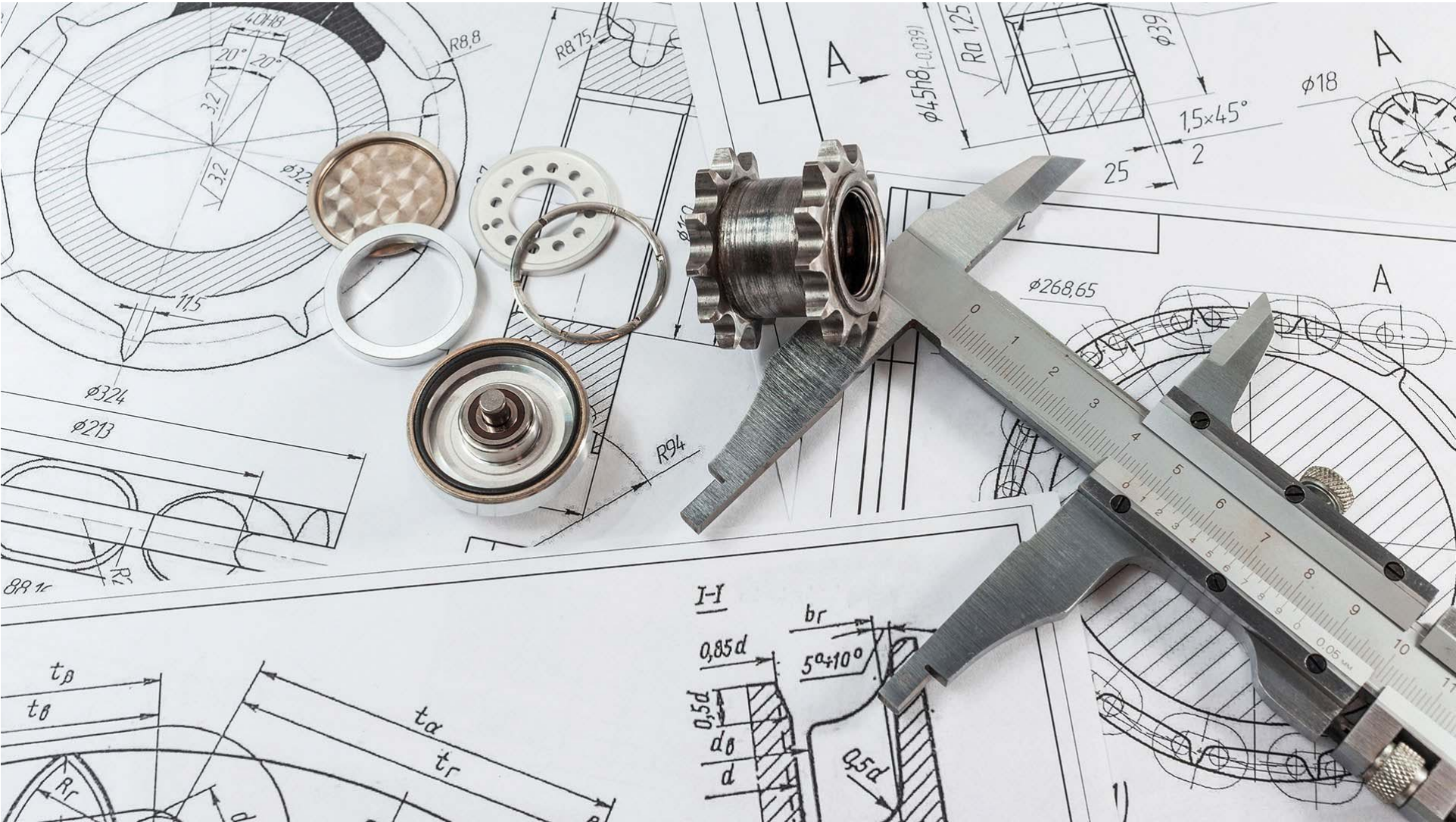
Permette la gestione del distanziamento dei treni tramite un sistema trasmittente a terra e un posto centrale (Radio Block Center)

- conoscendo lo stato della linea e degli altri treni, invia in continuità, tramite un collegamento GSM-R, informazioni ai treni relative all'autorizzazione all'avanzamento.





# Presentazione del progetto



## Obiettivo

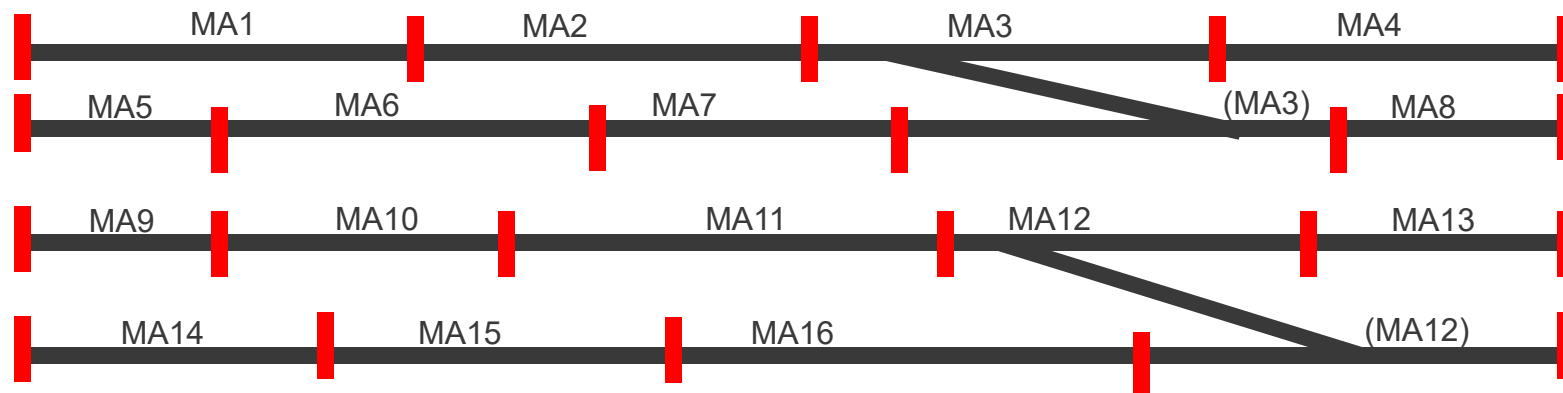
Obiettivo del progetto è rappresentare 5 treni che, per realizzare la loro missione, attraversano segmenti di binario, ricevendo il permesso di accesso a tali segmenti di binario da balises o RBC. La missione di ciascun treno è raggiungere una stazione definita.



## Segmenti di binario

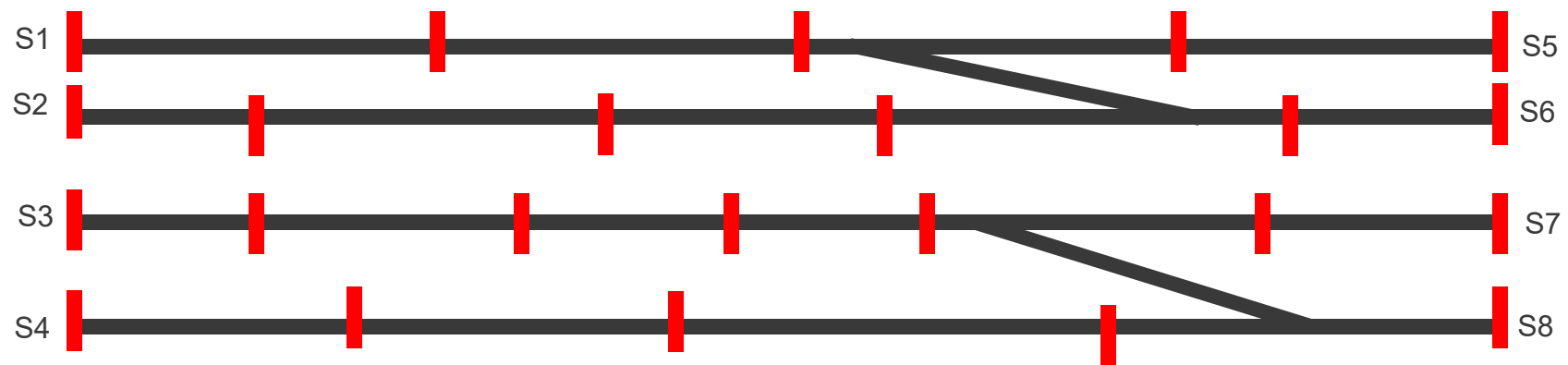
16 segmenti di binario, identificati con  $MA_x$ , quindi da MA1 a MA16.

- Ciascun segmento è delimitato da 2 boe, con l'eccezione dei segmenti che contengono interconnessioni. Questi ultimi sono delimitati da 4 boe



# Stazioni

Si definiscono otto stazioni, una per ciascun terminale di binario. Le stazioni sono indicate da S1 ad S8.



## Treni e percorsi

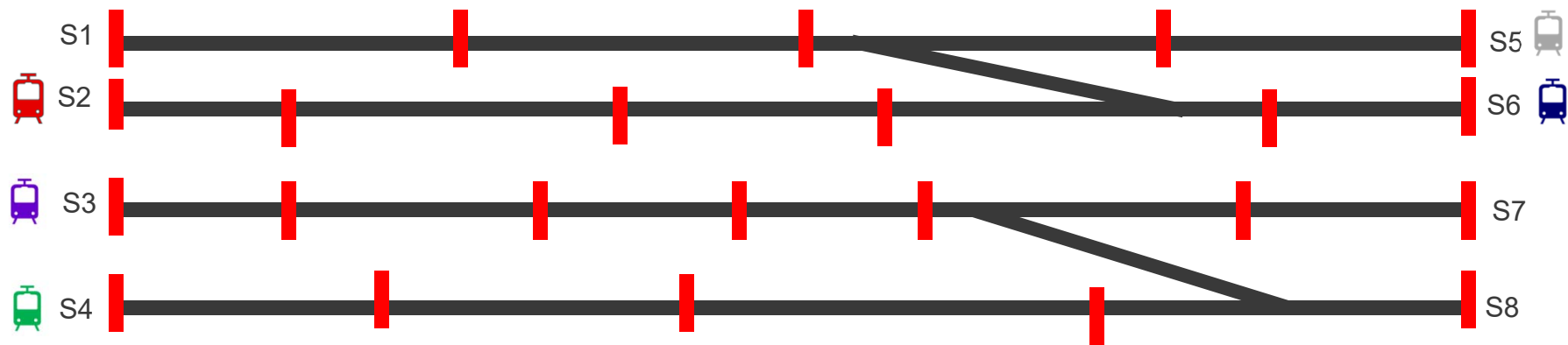
Si definiscono 5 treni, numerati da T1 a T5, e i relativi itinerari (stazione di partenza e di destinazione) come riportato nella tabella seguente.



TRENI		Partenza e destinazione
T1		Da S2 a S6
T2		Da S3 a S8
T3		Da S4 a S8
T4		Da S6 a S1
T5		Da S5 a S1

# Regole di percorrenza

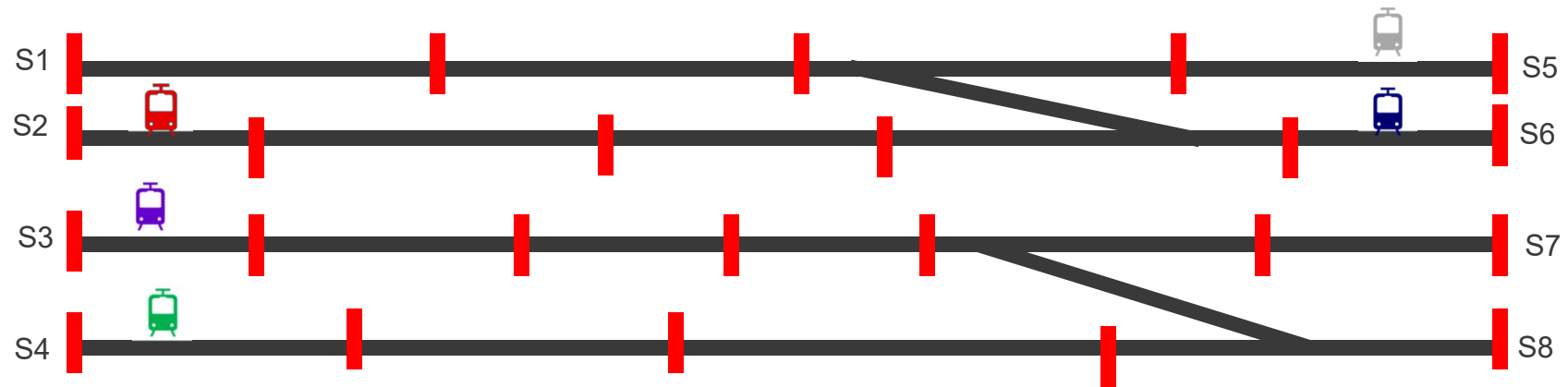
- ▶ Tutti i treni si avviano nello stesso momento, ovvero partono tutti insieme
- ▶ I treni vanno solo avanti
- ▶ Ciascun treno conosce il proprio itinerario, ovvero la sequenza di segmenti  $MA_{\underline{x}}$  che dovrà attraversare
- ▶ Per attraversare un qualsiasi segmento  $MA_{\underline{x}}$ , si impiegano 3 secondi.
- ▶ Ciascun treno chiede, al termine dell'attraversamento del segmento  $MA_{\underline{x}}$ , il permesso di attraversare il segmento successivo, tramite boe (ETCS LV1) o chiedendo ad RBC (ETCS LV2)
- ▶ La missione di un treno termina quando raggiunge la stazione di destinazione.


# Esempio di esecuzione: istante 0



TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

## Esempio di esecuzione: istante 3

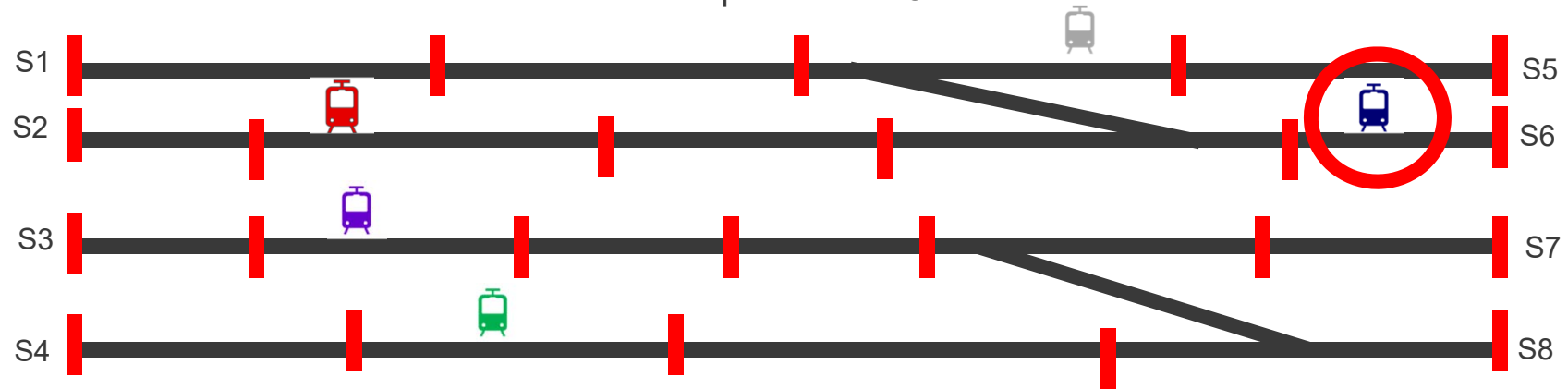


TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1



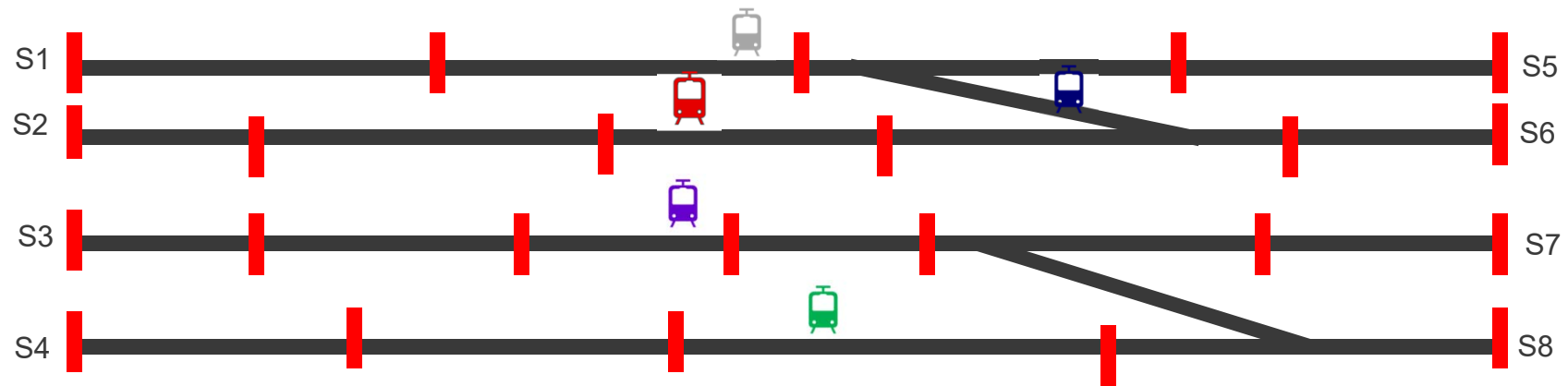
## Esempio di esecuzione: istante 6


Sia T5 che T4 vogliono accedere ad MA3; in questo caso, va avanti T5. A seconda di dettagli implementativi, T4 potrebbe precedere T5.



TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

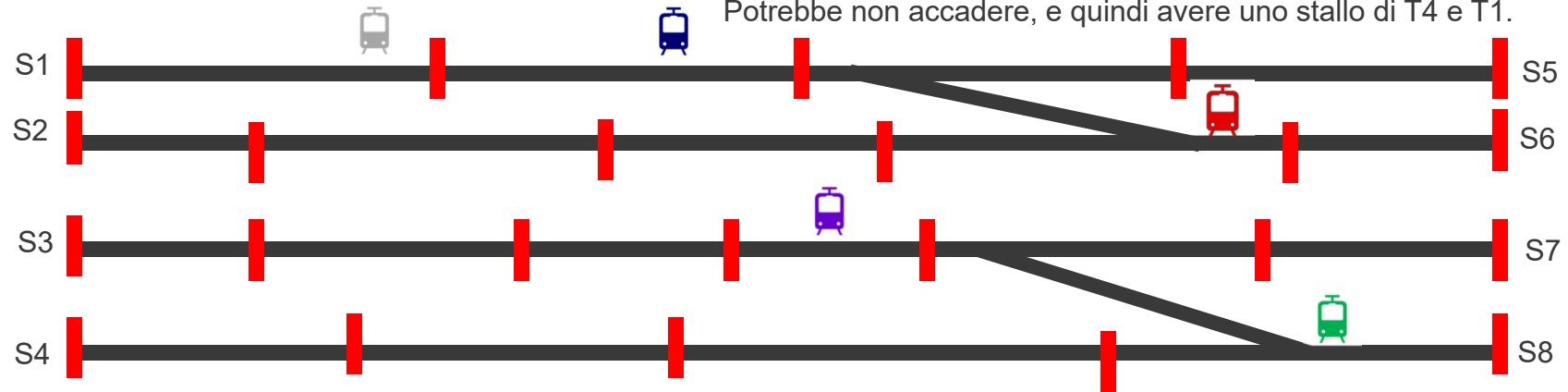
## Esempio di esecuzione: istante 9



TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

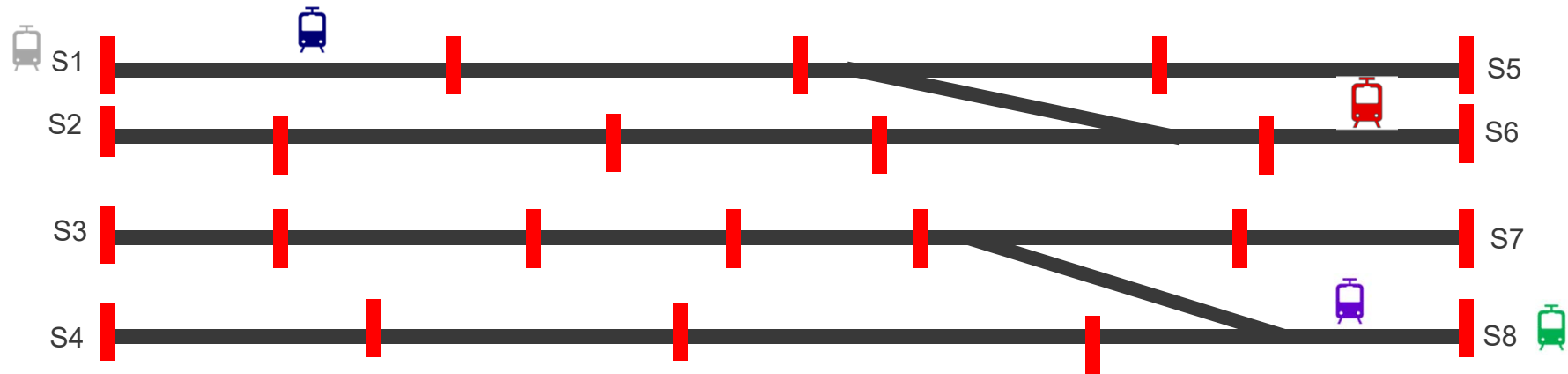
# Esempio di esecuzione: istante 12

Questo è il caso ideale in cui le autorizzazioni sono state chieste nell'ordine ottimale (prima T5, poi T4 e quindi T1). Potrebbe non accadere, e quindi avere uno stallo di T4 e T1.



TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

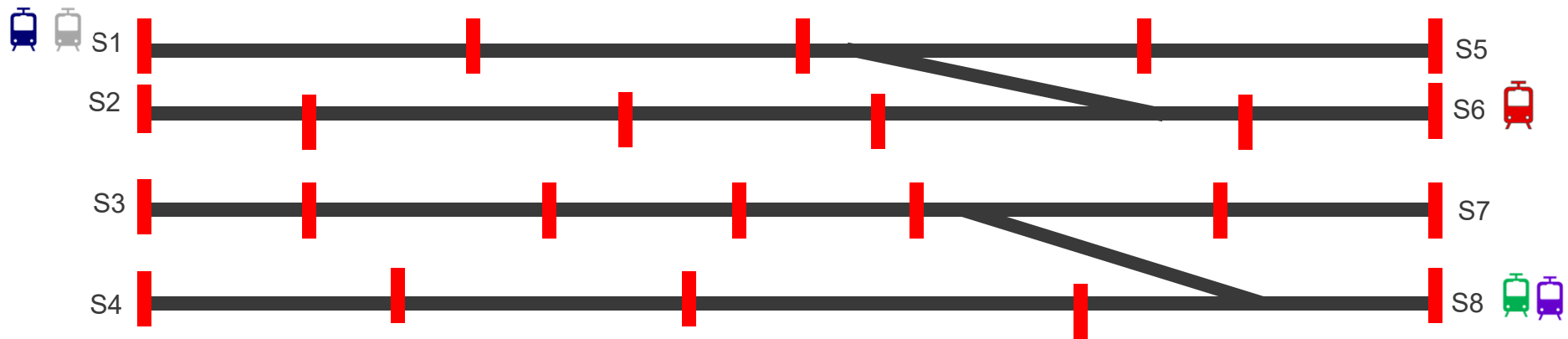
# Esempio di esecuzione: istante 15



Anche in questo caso, T2 ha chiesto l'autorizzazione dopo T3;  
Alternativamente, T2 avrebbe dovuto attendere il turno successivo.

TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

# Esempio di esecuzione: istante 18 (arrivo)



TRENI	Partenza e destinazione
T1 	Da S2 a S6
T2 	Da S3 a S8
T3 	Da S4 a S8
T4 	Da S6 a S1
T5 	Da S5 a S1

## Richieste implementative

Se non diversamente specificato, le seguenti richieste implementative sono da intendersi prescrittive, equivalenti alla parola **MUST** secondo l'**RFC 2119**.

*MUST This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.*



## Richieste implementative – Avvio del programma

Il programma è eseguito in due modalità, selezionabili tramite due opzioni:

- Con l'opzione ETCS1

Esempio: `./nomeprogramma ETCS1`

- Usando due shell: in entrambe, l'applicativo è da eseguire specificando l'opzione ETCS2 e l'opzione RBC

Esempio:

shell1: `./nomeprogramma ETCS2`

shell2: `./nomeprogramma ETCS2 RBC`

L'opzione ETCS1 avvia il programma per la versione ETCS1, l'opzione ETCS2 avvia il programma per la versione ETCS2.

## Richieste implementative - 1

I cinque treni sono rappresentati con 5 processi, che chiameremo nel seguito TRENI. I TRENI sono figli di un unico processo che chiameremo nel seguito PADRE.

Ciascun TRENO, all'avvio, legge il proprio percorso (ovvero, l'insieme di segmenti MAX e stazioni che deve attraversare) da un file di testo, e lo mantiene in memoria, in una struttura dati

- I percorsi per i TRENI sono forniti nei file CSV T1, T2, T3, T4, T5

Tutti i TRENI avviano la propria missione contemporaneamente, al meno di inevitabili ritardi computazionali.

## Richieste implementative - 2

Ciascun segmento  $MA_x$  è rappresentato tramite un file di testo, di nome  $MA_{numero\ file}$  (ad esempio,  $MA1, MA2, \dots$ ).

- Questi file sono creati dal PADRE e inizializzati con uno "0" al loro interno. Sono impostati per avere accesso in lettura e scrittura da parte di tutti, e privilegi dei file posti a 777.
- un TRENO, durante la missione, si posiziona in un dato segmento  $\rightarrow$  cambia lo stato del file  $MA_x$  corrispondente da 0 ad 1.
- Quando un treno abbandona un segmento, il TRENO cambia lo stato del file  $MA_x$  corrispondente da 1 a 0.

## Richieste implementative – opzione programma "ETCS1"

- ▶ Un treno accede al segmento successivo solo se il corrispondente file  $MA_x$  contiene il valore 0.
  - Altrimenti, rimane nel segmento corrente.
- ▶ (opzionale) *Si consiglia di considerare l'opportunità di effettuare il lock dei file per letture/scritture concorrenti.*

# Richieste implementative – opzione programma "ETCS2"

L'informazione è mantenuta da un **server socket RBC**. Il server socket RBC, o semplicemente RBC nel seguito, controlla l'intera esecuzione. RBC è implementato come un applicativo separato e individuale, che esegue in una shell separata dal resto dei processi PADRE e TRENO.

- ▶ All'avvio, RBC riceve dal PADRE tutti i percorsi dei treni, in altre parole le informazioni contenute nei file T1, T2, T3, T4, T5.
- ▶ RBC crea una struttura dati in cui mantiene lo stato di tutti i segmenti MAX e le stazioni. Si nota che RBC non accede ai file MAX creati dal PADRE.
- ▶ Durante la missione, ciascun treno:
  - Contatta RBC per l'autorizzazione a muoversi nella sezione successiva (MAX o Stazione)
  - Se riceve l'autorizzazione, si muove
  - Se l'autorizzazione è negata, rimane fermo
- ▶ *Requisito opzionale: in caso di informazione discordante tra RBC e boe (ovvero, le informazioni dell'RBC non corrispondono con i requisiti delle boe), il TRENO rimane fermo.*

## Richieste implementative - Logging

- Ciascun processo TRENO riempie un file di log T1.log, T2.log, T3.log, T4.log, T5.log

- Ciascun file di log indica

- Settore MA $\underline{x}$  in uso (oppure stazione), prossimo settore (oppure stazione), data
- A titolo di esempio:

[ATTUALE: MA5], [NEXT: S6], 27 Aprile 2018 16:14:13

- Il processo RBC scrive un file RBC.log che contiene

- Le autorizzazioni concesse e negate, i destinatari, e la data

A titolo di esempio:

DESTINATARIO: T4; COLLOCATO: S6; RICHIESTA: MA8, AUTORIZZATO: SI, DATA: 27 Aprile 2018 16:15:22

DESTINATARIO: T4; COLLOCATO: MA8; RICHIESTA: MA3, AUTORIZZATO: NO, DATA: 27 Aprile 2018 16:15:25

DESTINATARIO: T4; COLLOCATO: MA8; RICHIESTA: MA3, AUTORIZZATO: SI, DATA: 27 Aprile 2018 16:15:28



# Regole per lo svolgimento o consegna del progetto

Il progetto deve essere svolto in gruppi di 2-3 persone. Gli studenti lavoratori possono svolgere il progetto individualmente.

E' necessario consegnare:

- Il **codice sviluppato** inclusi tutti i file necessari alla sua compilazione.
- Una **relazione** sul progetto. Ogni gruppo di lavoro dovrà produrre **una sola relazione**, i cui autori saranno quindi tutti i membri del gruppo stesso. La relazione dovrà essere in formato **pdf**.
- Codice e relazione dovranno essere forniti in un archivio **.zip** oppure **.tar.gz**, caricato sul sito del corso seguendo l'apposito link che verrà reso disponibile alla pagina del corso.
- *Non si accettano consegne in formati o modalità differenti da quelli sopra indicati*

# Relazione

La relazione dovrà contenere:

- **Informazioni sugli autori** (per ciascun componente del gruppo: Nome, Cognome, Numero di matricola, indirizzo e-mail) e la data di consegna
- **Tutte le istruzioni necessarie per compilare i file sorgenti e lanciare in esecuzione il programma.**
- **Descrizione dell'implementazione.** La relazione dovrà essere esaustiva delle scelte progettuali e implementative.
- **Evidenza del corretto funzionamento del programma.** Ciò può essere realizzato riportando e commentando estratti del file di log, cioè commentando tracce dell'esecuzione.

# Q&A Time

