11. Train Traffic Control (TTC) and Information System

Contents:

"Everything should be made as simple as possible, but not simpler." **Albert Einstein**

Build models that are suited to the current goals.
What are the goals when we build the models?

A8	B8	C8	D8	E8	F8	G8	H8
A7	B7	C7	D7	E7	F7	G7	H7
A6	B6	C6	D6	E6	F6	G6	H6
A5	B5	C5	D5	E5	F5	G5	H5
A4	B4	C4	D4	E4	F4	G4	H4
A3	B3	C3	D3	E3	F3	G3	H3
A2	B2	C2	D2	E2	F2	G2	H2
A1	B1	C1	D1	E1	F1	G1	H1

Homework

Add to the previous mobile robots problems the constraints:
The land is partitioned in four zones: N-E, N-W, S-E, and S-W.
The robots can communicate in each zone using the wireless link provided by the zone's router.
The robots in different zones can communicate using the zones' servers (see the mobile entities problem).
The servers of different zones can communicate each other (without constraints).
Each zone has its own coordinator.
The coordinators of different zones can communicate freely (variant two: with their neighbors) using the services of their zone servers.
The frontier resources are managed by the zones' coordinators.
(see Specification.Distributed PN)

	North-East				North-West				
North-East	A8	B8	C8	D8	E8	F8	G8	H8	North- West
	A7	B7	C7	D7	E7	F7	G7	H7	
	A6	B6	C6	D6	E6	F6	G6	H6	
	A5	B5	C5	D5	E5	F5	G5	H5	
South-East	A4	B4	C4	D4	E4	F4	G4	H4	South-West
	A3	B3	C3	D3	E3	F3	G3	H3	
	A2	B2	C2	D2	E2	F2	G2	H2	
	A1	B1	C1	D1	E1	F1	G1	H1	
	South-East				South-West				

Level 1: robots are moved from one intersection to another without collision.
Level 2: the deadlocks are detected.
Level 3: the deadlock appearances are removed.

➔Levels of difficulties
➔Levels of competence
➔Levels of efficiency

Communication link:
1) Any to any
2) N-E←→N-W←→S-W←→S-E←→N-E

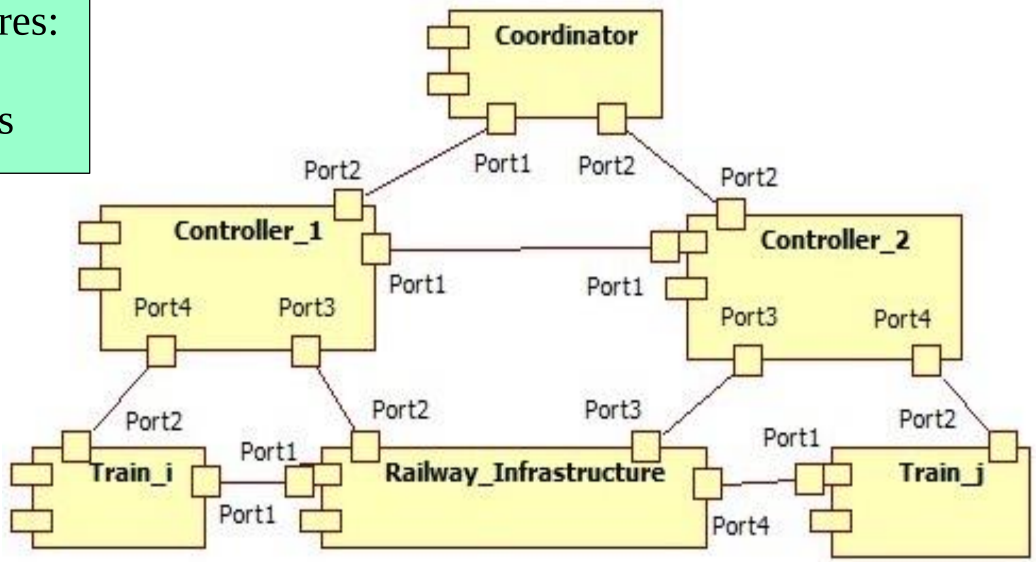
3. Railway System Models

Rationales of modeling:

- To understand the railway system
  - The structure
  - The behavior
- To specify the plants (infrastructure and trains) and verify
  - The structure
  - The behavior
- To specify the requested behavior
- To synthesize and verify the monitoring system
- To synthesize and verify the control and the management system
- To synthesize and verify the information system
- For implementation of control, management and information system
- To verify and improve the safety system
- To sustain the maintenance and the software update
- To determine the system resilience and to improve it
- To implement the traffic resilience.

Model features:
- structure
- interactions

Fig. Railway system component diagram model 1



Someone's level of understanding depends on education, self-education and training.

Someone's power of understanding depends on the models used for thinking.

Modeling ←→ implementation = Simulation ←→ Verification ←→ Testing ←→Maintenance

There are different models depending on the goals and the needed details.

Petri nets can model:

- The railway infrastructure (lines, platforms, interlockings, traffic lights, eurobalises, euroloops, other trackside signals, etc.)
- The traffic coordinator (centralized, distributed, scheduler)
- The railway plant control
- The train move (behavior) and the controller (engine onboard computer)
- The relations between the previous mentioned components

Kinds of Petri Net Models:

- Enhanced Time Petri Nets
- Unified Enhanced Time Petri Nets
- Object Enhanced Real-Time Petri Nets

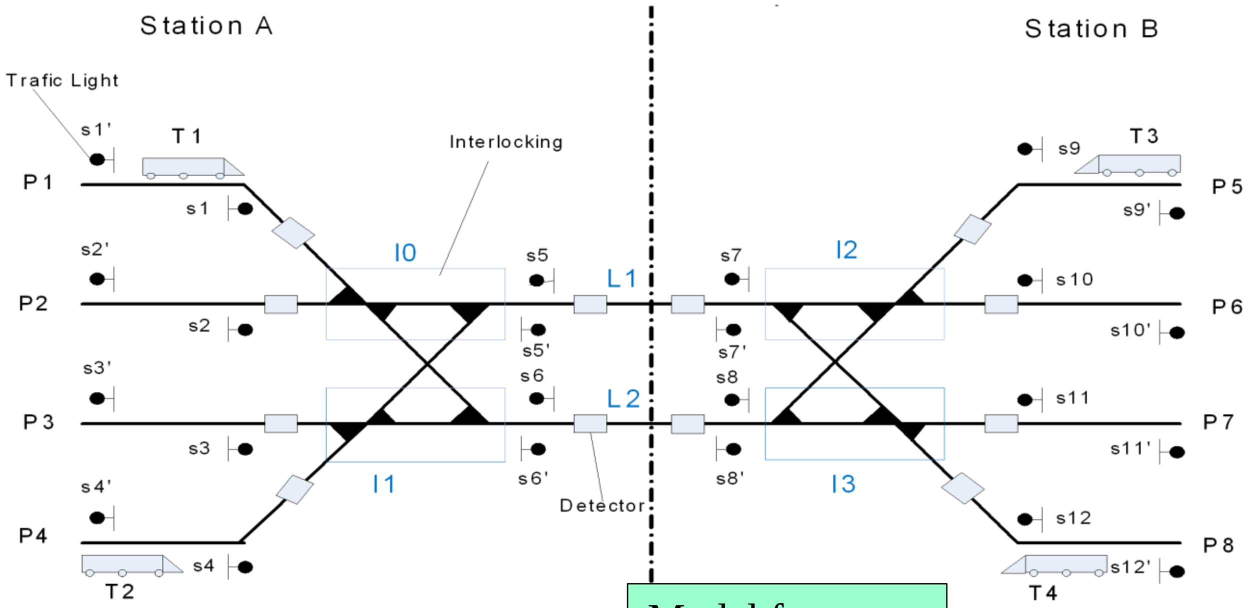
A PN model includes the structure and the behavior. Some annotations are used for filling the information that are not included.

PNs sustain the analysis and the verification of models' structures and behaviors.

Stochastic Petri net can catch the uncertainties.

Approaches of railway system management and control:

- *Teleoperation* – human operators (i.e. dispatchers) control remotely and directly the interlockings;
- *Supervisory control* – a set of cooperative supervisors interacts with local controllers to read and set the interlockings etc.;
- *Multi-agent system* – a set of fixed agents cooperate with mobile agents (implemented on train engines)



Model features:
- structure

Software of a multi-agent system

Agent definition:

An ***agent*** is a task that:

- Has a mission
- Has decision autonomy
- Has communication capabilities
- Can adapt itself to its environment

All the approaches of TTC have to solve the problems:
1) Resource reservation and allocation ←→ resource scheduling
2) Interlocking and train monitoring
3) Interlocking settings
4) Infrastructure and train reliability
5) Infrastructure and train traffic resilience

The agents can communicate and collaborate.

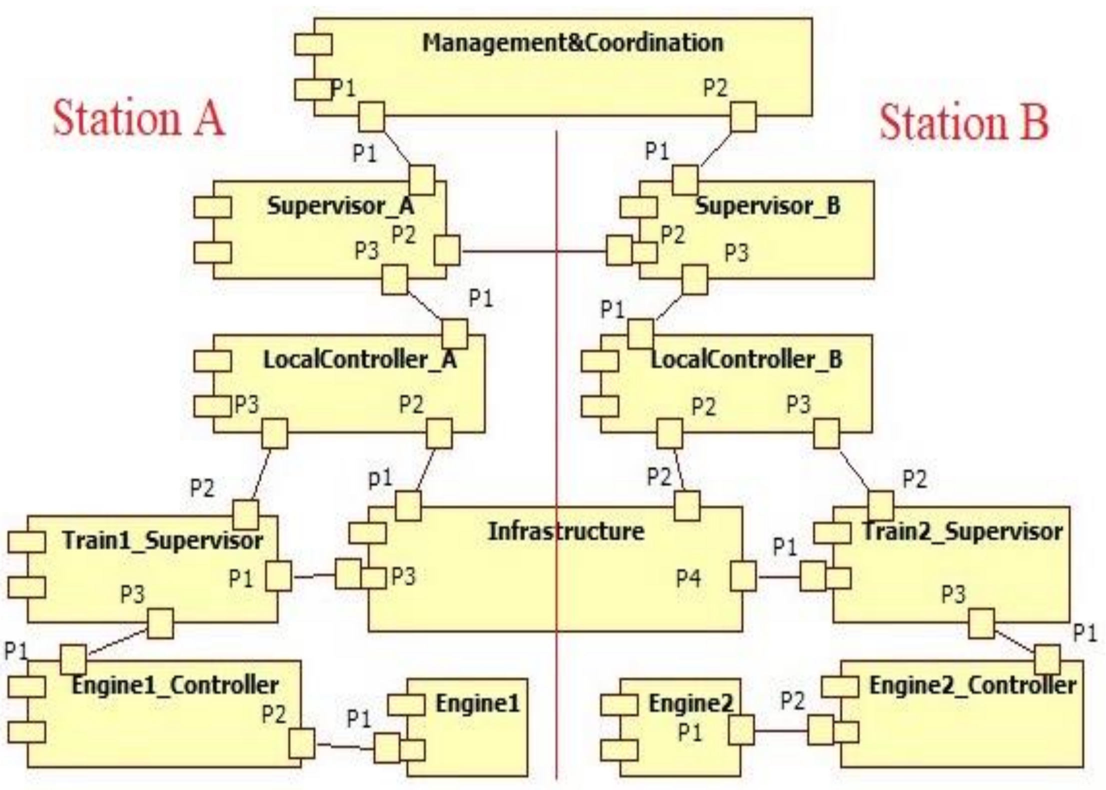
Human operators are included in the loops: → i. e. these are kinds of *Cyber-Physical Systems*.

There are mobile agents (on onboard engine unit) and fixed agents implemented in TCS.

Models of railway system:

- Infrastructure (lines, platforms, interlockings, balises, euroloops = detectors, traffic lights, other side track devices, communication channels etc.)
- Train Traffic Control and Supervision System (TTCSS)
  - Supervisor
  - Local controller
- Train (train agent or supervisor, engine controller, train movement)
  - Train supervisor
  - Engine controller
  - Engine (not a subject of this course)
- Inter-component communication (included in infrastructure)
- Railway traffic behavior

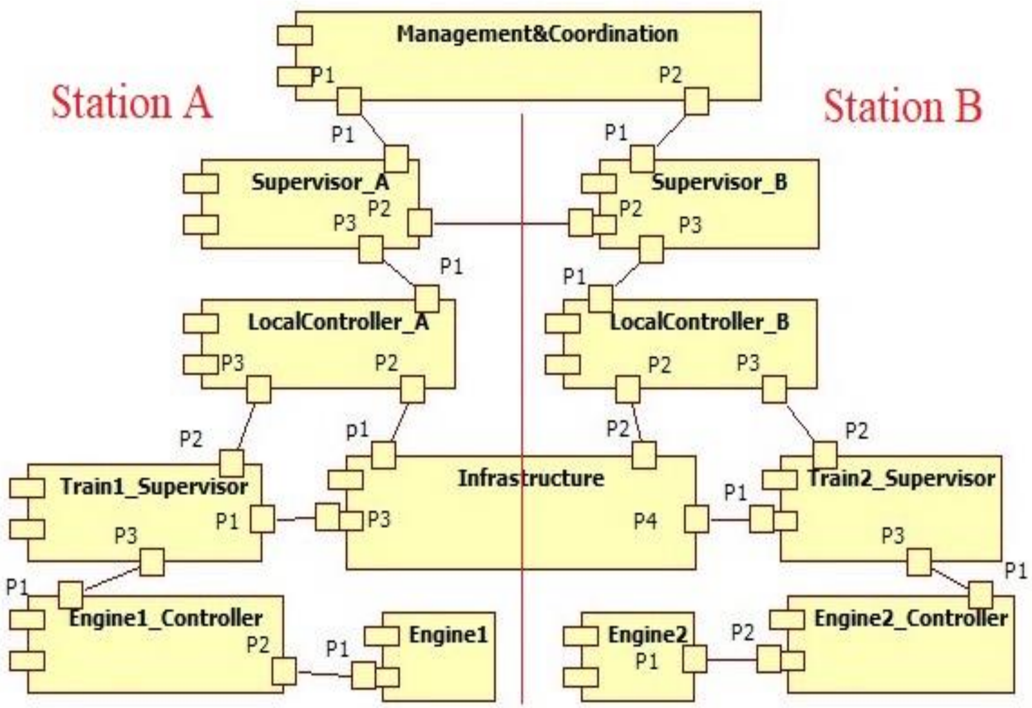
Fig. Railway system component diagram model 2



Model features:
- structure
- interactions

Why should be used such models? What are the benefits?

Teleoperation and supervisory control



For safety reasons there are double communication channels and double decision makers.
The human operators supervise and can take the control of TCS.

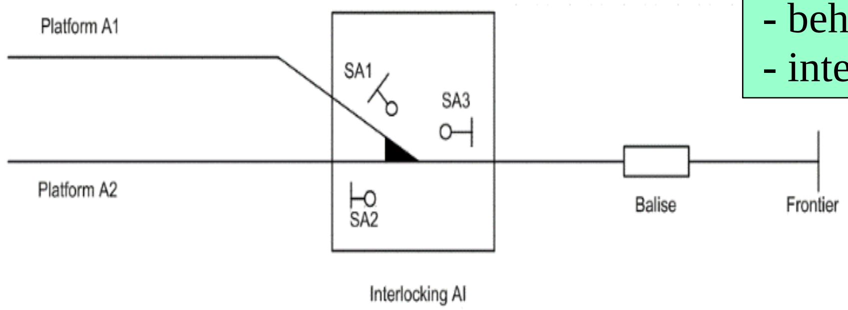
Human operators (dispatchers) set the interlockings side track devices.
The dispatchers send (through wireless channels) movement commends to train supervisors.
Train supervisors obey the received commends.

Supervisory control: TCS supervisors replaced the human operators performing the same operations. TCS controllers receive information from devices, set the interlockings and side track devices.

There are multiple stages of TTCSS development and implementation.

Infrastructure models

Fig. 1. Simple Infrastructure.



Model features:
- structure
- behavior
- interactions

Fig. 2. Simple Infrastructure OER-TPN model.

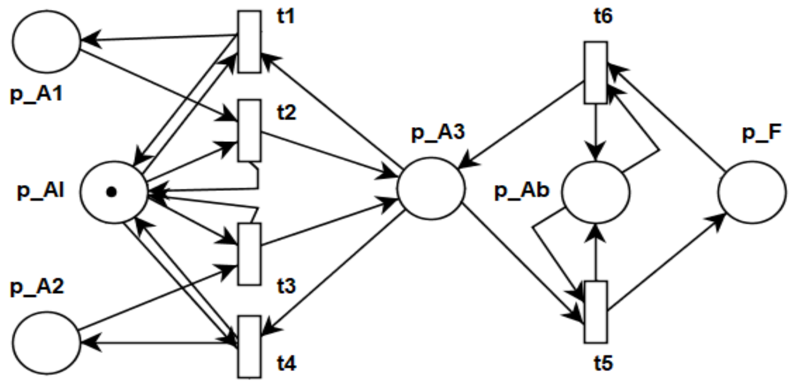


Fig.2. represents OETPN model of the infrastructure drawn in Fig.1 corresponding to the Station A. Place p_{A1} corresponds to the platform 1. The rest of the places are defined in the attached table.

Node	Significance	Token/place type	Place Role
p_{A1}	Platform A1	Train	Input/output
p_{AI}	Interlocking A	Control signal	Input/output
p_{A2}	Platform A2	Train	Input/output
p_{A3}	Line A	Train	State
p_{Ab}	Balise of A	Control signal	Input/output
P_F	Frontier	Train	State

A token of type train can be set in the places p_{A1}, p_{A2}, p_{A3} and P_F and this are denoted by

$type(p_{A1}) = type(p_{A2}) = type(p_{A3}) = type(p_F) = Train$.

The other types are: $type(p_{AI}) = Interlocking$ and $type(p_{Ab}) = Balise$.

As a matter of fact, these specify the types of the token that can be set in the mentioned places.

The place of the type *Train* contains information relative to if it contains a train in the current moment of time and the train identifier. Interlocking and Balise store information sent by controller and provide information to trains. The trains obey to Interlocking and have to obey to Balise requirements.

Homework:

Add details to place classes to make possible the next request:

Specify the transition significances and add their guards, mappings and delays determined by the information stored in *Train* tokens.

When a train is moved from one place to the next place? Where are stored the information to execute the corresponding transition?

How can be modeled a transition delay (depending of segment length and the train maximum speed) of a train that releases a line? (i. e. delay = length/trainSpeed)

How can be modeled that a train has a movement direction?

(e.g. frontier → platform_A; platform_B → frontier)

Fig. 2. Infrastructure

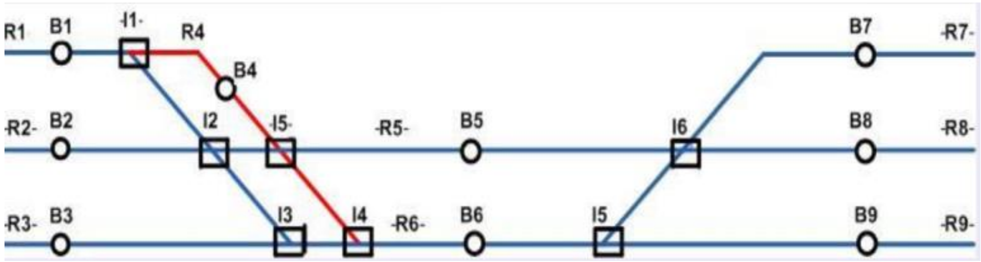


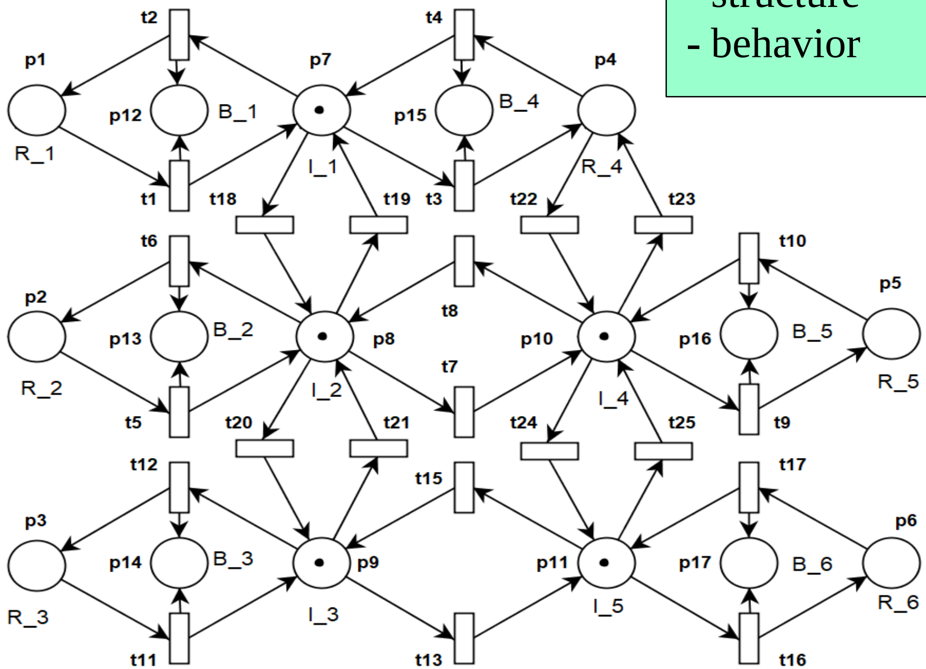
Fig. 2 is an example of a part of a railway system that links two stations. R_i (i=1, 2, …, 9) denote the resources representing the platforms and lines. I_i (i=1, 2, .., 6) are interlockings and B_i (i=1, 2, …, 9) are balises.

Each resource is endowed with a balise used for communication between engine onboard computer and TCS (Traffic Control System).

Fig 3 shows a part of the OETPN that models the network linking the resources R1, R2, R3, R4, R5 and R6.

Fig.3 A part of OETPN that models the network linking the resources R1, R2, R3, R4, R5 and R6.

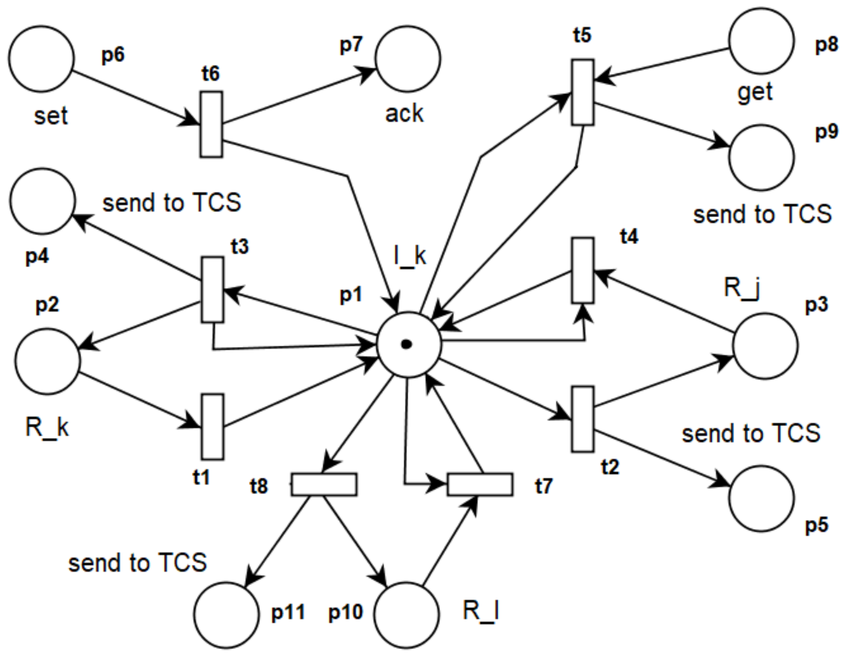
Model features:
- structure
- behavior



The links with TCS are ignored from the representation reasons to avoid the loading of the figure with excessive information. These were added in the next model.

The balises (B1, B2, …, B6) used for communication between TCS and trains supervisor are loaded by trains when they pass over or by TCS if some information have to be sent to (train) engine onboard computer.

Fig. 4. Interlocking model.



The onboard computer are programed to react and to receive information if the driver does not do it.

The OETPN model of an interlocking I_k linking the resources R_i, R_j and R_k is represented in Fig. 4.

TCS can set the linked resources through the place p_6 and receive the confirmation acknowledgement by the place p_7.

TCS can request the linked resources or the interlocking state using the place p_8 and receive the information by p_9.

The trains can cross the interlocking from a resource to another one and the move is signaled by places p_4, p_5 or p_{11}.

The places types are *Type*(p_1)=*Interlocking*, *Type*(p_2)=*Type*(p_3)= *Type*(p_{10})= *Line*; *Type*(p_4) = *Type*(p_5) = *Type*(p_{10}) = *TrainInfo*; *Type*(p_7) = *Type*(p_9) = *InterlockingInfo*.

Each resource (including the interlocking) can temporarily aggregate a Train object.

Model features:
- structure
- behavior

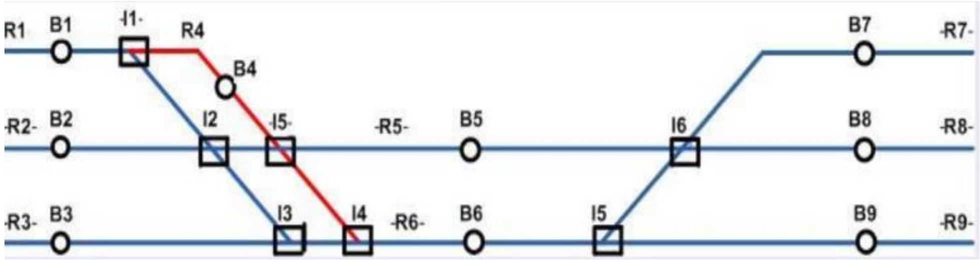
Train information:

- Identifier
- State (move direction, maximum speed, current speed, etc.)

Homework: Fill the OETPN models with the missing details (types, guards, mappings, eet, let).

The *train agent* has a mission described by
*Route = Station_A*Station_B*⋯.*

The train supervisor has the route too.



Two platforms can be linked by sequences of segments.

E. g. R1 → R7

*LinkSequenceSet(R1, R7) = {R1*I1*R4 * I5 * R5*I6*R7, R1*I1*I2*R5*I6*R7, R1*I1*I2*I3*I4*R6*I5*I6*R7, ⋯ }*

A train has *locally assigned a path* chosen from the *LinkSequenceSet*. It is described by the sequence of segments:

σ = *R1*I1*R4*I5*R5*I6*R7*

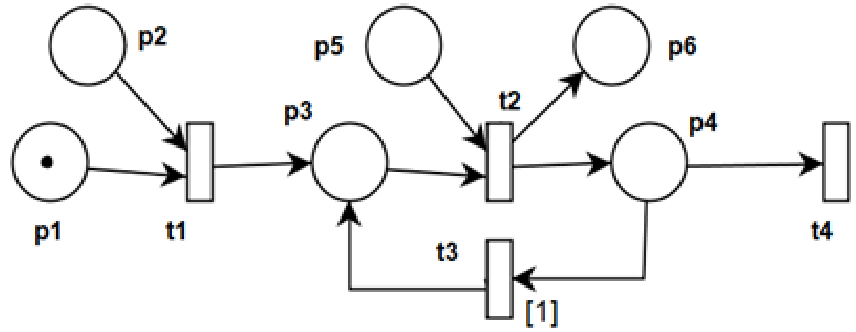
A train assigned (reserved) path is implemented by TCS (human operator, local controller) before starting the train.
The assigned path is reserved for a specified period of time and it is included in the *movement authority* together with the maximum speed and the gradient of each segment.

A train loses the reservation (releases) the resources after exiting them. The released resources can be allocated to other trains.

Train models

Trains with fixed blocks
Simple train OER-TPN model
Simulation ➔ train move prediction.

Model features:
- structure
- behavior
- interactions



A Token in p1 means that the train is created on the station platform. A token in p2 means that the train is allowed to start due to the fact that the controller has set accordingly the semaphores if the resources that are used for moving to the next station are free.

When the transition t1 is executed, a token is set in p3 marking the train moving state.
The transition t2 is conditioned by the information red from the place p5 linked to an input channel receiving information from infrastructure (i.e. interlocking or balise).

Node	Significance	Token/place type	Place Role
p_1	Initialization	Train_Parameters	State
p_2	Resource info	Resource	Input
p_3	Train state	Train_State	State
p_4	Train state	Train	State
p_5	Control signal	Control signal	Input
p_6	Monitoring	Train_Information	Output

The transitions t2 and t3 are executed periodically changing the train position. When it reaches the resource limit it becomes a passive object and the infrastructure executor moves it to the next resource as an active object.

When the train reaches the infrastructure frontier, it is sent as an object to the next infrastructure.

When the train reaches the final destination, the physical train can be removed from the physical system and then the software train is removed from the software infrastructure too.

Where can be used the above presented model? ➔ To conceive the software implemented on engine.
Homework:
1. Endow the train simple model with information related to distance (current local position) and speed.
2. Endow the train simple model to react to the (logic) traffic lights (i.e. semaphores).

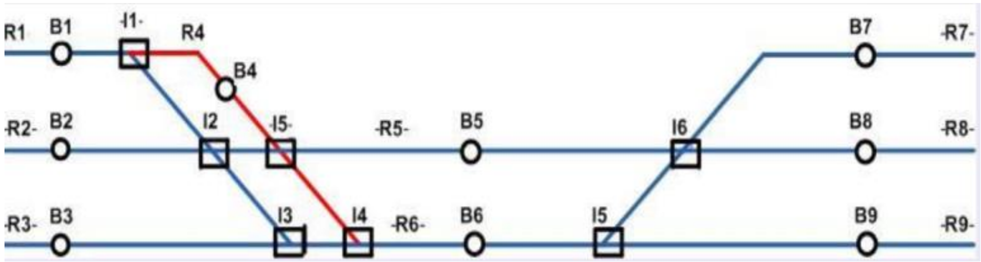
Detailed train model (supervisor, controller, engine)

Solution for mobile continuous block.

(rom. blocuri alunecătoare)

When mobile continuous blocks (between two trains) are used, the trains are supposed to be linked through a wireless communication channel.

When the wireless communication is lost, the mobile blocks procedure is replaced by fixed blocks.



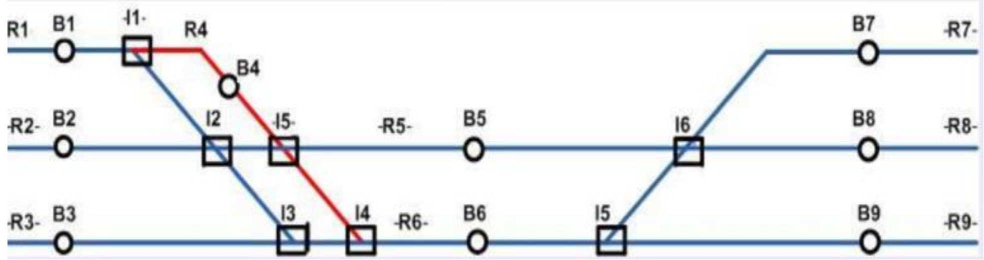
A line resource is assigned to a set of train with the same move direction.

A follower train supervisor (or agent) has to be informed when the followed train changes its move parameters (speed, acceleration).

Model implementation: *type(resource) = TrainList*

Trains with mobile continuous blocks

Complex train model is shown in the attached figure.

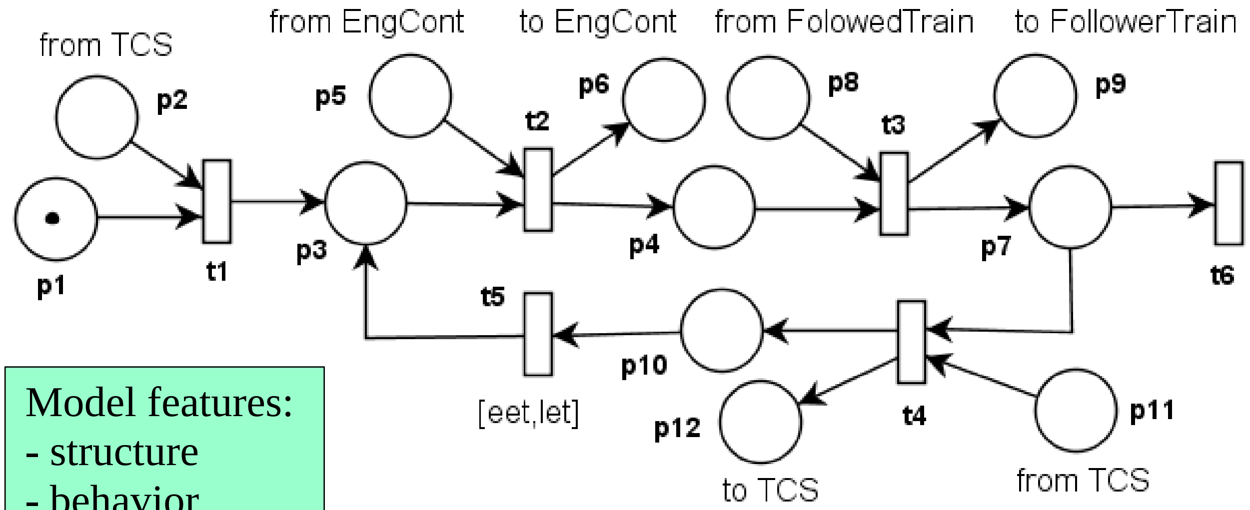


Places p5 and p6 are linked to train Engine Controller. These places are used to get and set information from/to infrastructure.

Engine Controller receives move parameters and sends current train movement state.

Place p8 is used to get movement information (localization, speed, acceleration etc.) from the followed train.

Place p9 is used to send information to a potential follower train.



Model features:
- structure
- behavior
- interactions

Places p11 and p12 is used for changing information with the supervisor of the zone (station) where the train is currently moving.

Where can be used the above presented model? ➔
To conceive the software implemented on engine.

Homework

Conceive Engine (move) Controller model.

Add to both models: types, guards, mappings, eet and let.

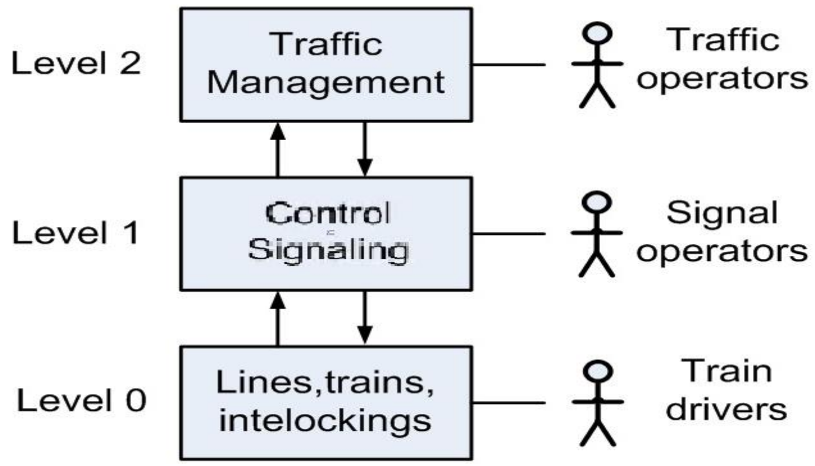
Add the behavior: the train move speed is limited by the segment maximum accepted speed and by the geometrical distance relative to the followed train.

4. Train Traffic Control

It is achieved by:

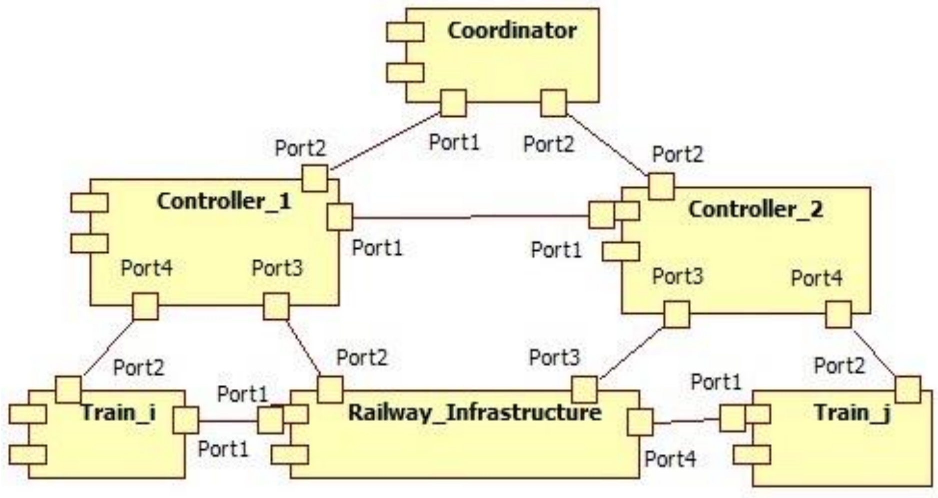
- Monitoring
- Teleoperation or
- Supervisory control or
- Multi-agent system

Model features:
- structure
- interactions



Involved activities:

- Train scheduling = Resource scheduling = path reservation
- Signal control (command) = controller task
- Communication: TCS supervisors- engines supervisors – railway infrastructure

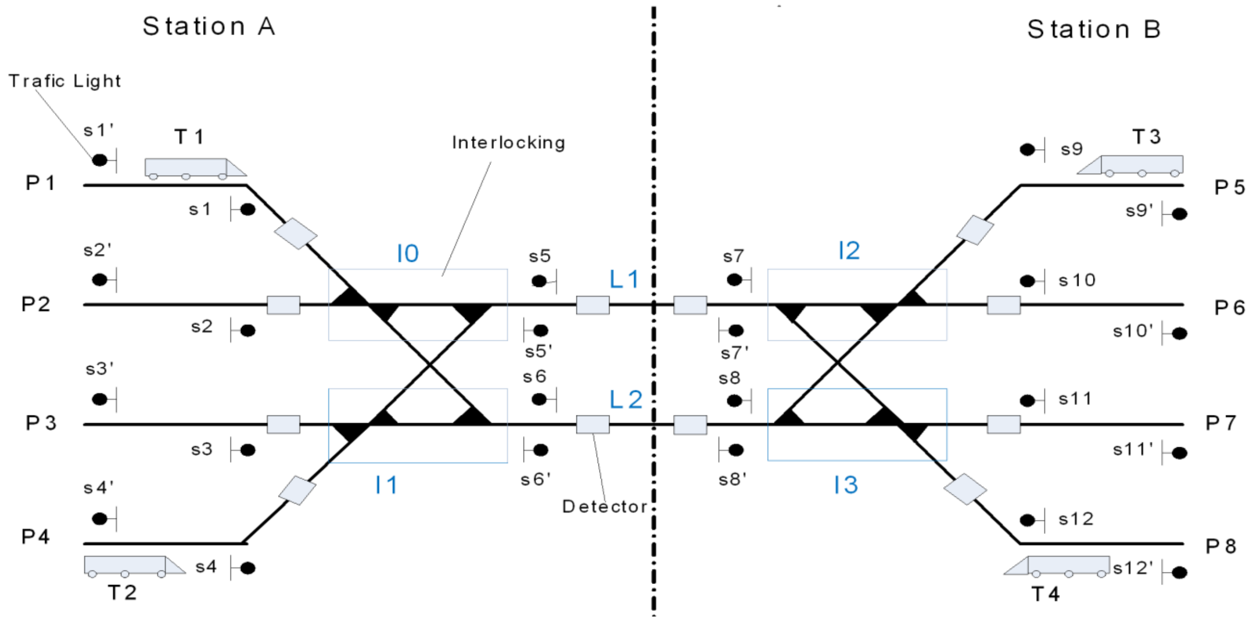


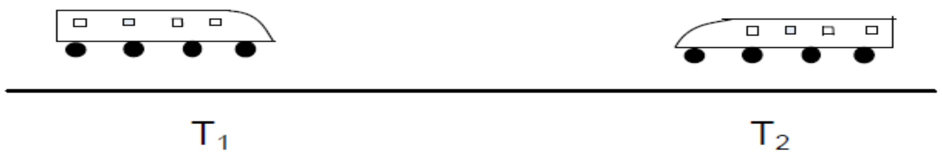
Scheduling

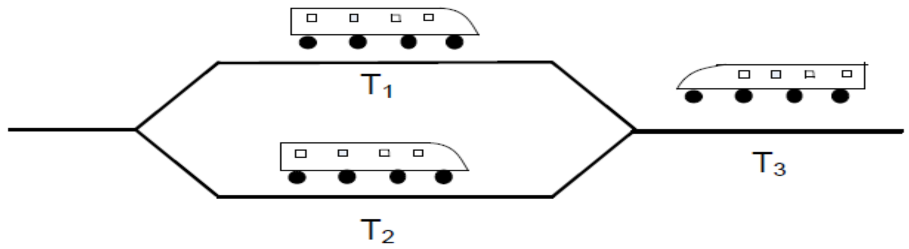
Train deadlocks
Scheduling role: avoid collisions, deadlocks and starving.

Long time (distance) scheduling → routes

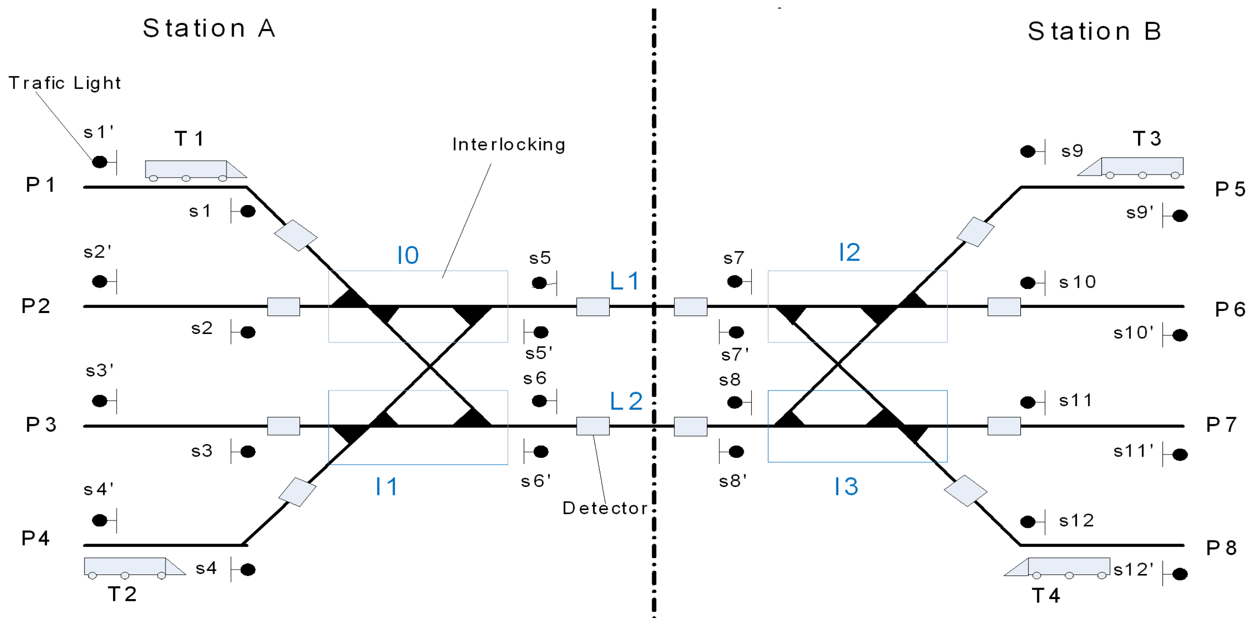
Short time (distance) scheduling → paths between neighbors railway stations or waiting segments.







Scheduling



The task of scheduling consists of determining the movement information for each train:

- the day and time on which the train should run
- the route of the train (the stations through the network)
- departure from and arrival times at stations
- maximum, average speed

Scheduling:

- long distance scheduling – route scheduling
- short distance scheduling – the path between two neighbor stations

Scheduling result: *train time table* = *train movement diagram* = a survey of all scheduled trains that run on the same portion of a line.

Headway represents:
time interval or distance between two vehicles, automobiles, ships, or railroad or subway cars, traveling in the same direction over the same route.
It is an available duration for producing of two consecutive events (departure, arrival etc.)

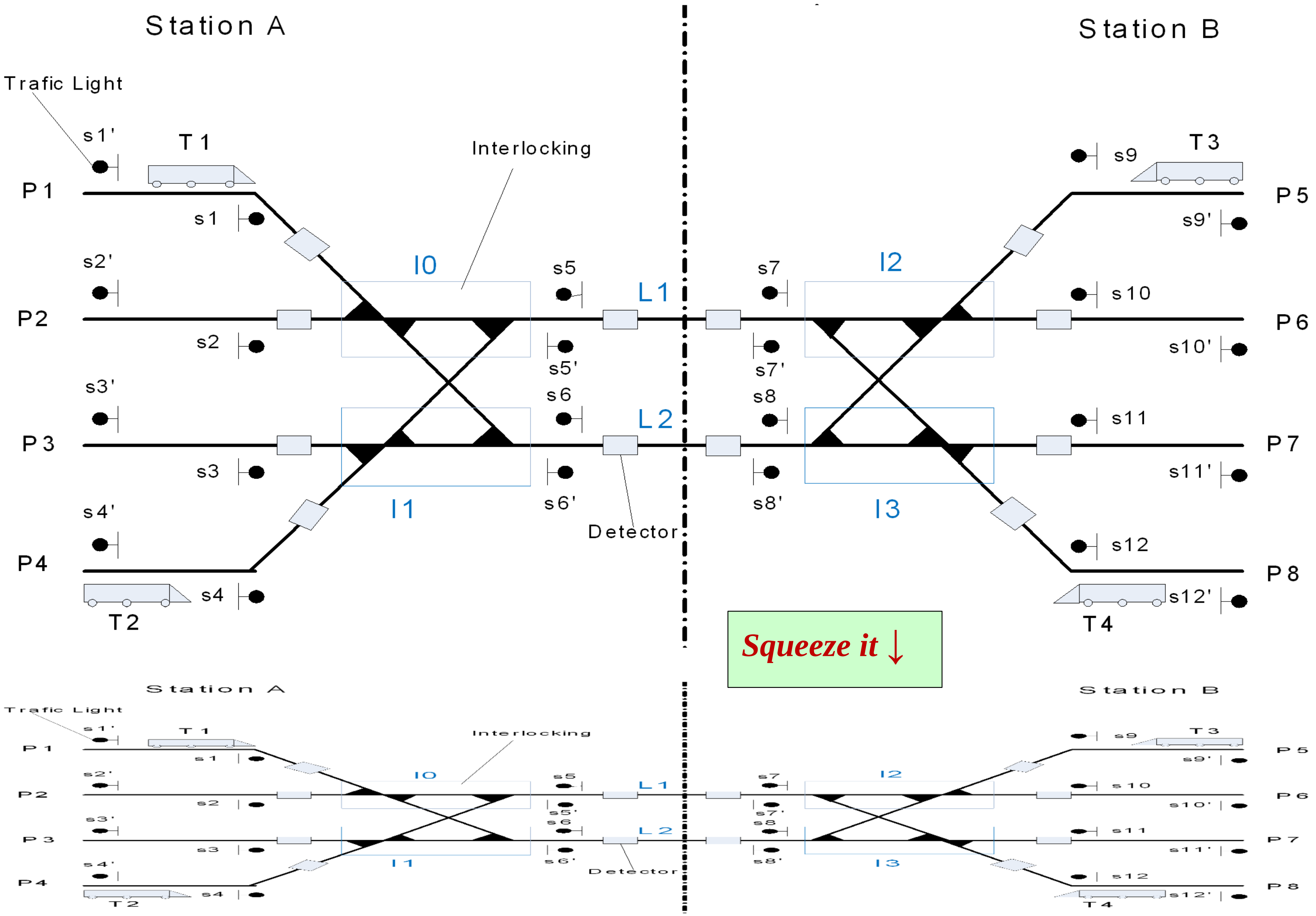
- *depart-depart headway*
- *depart-arrival headway*
- *arrival-arrival headway*

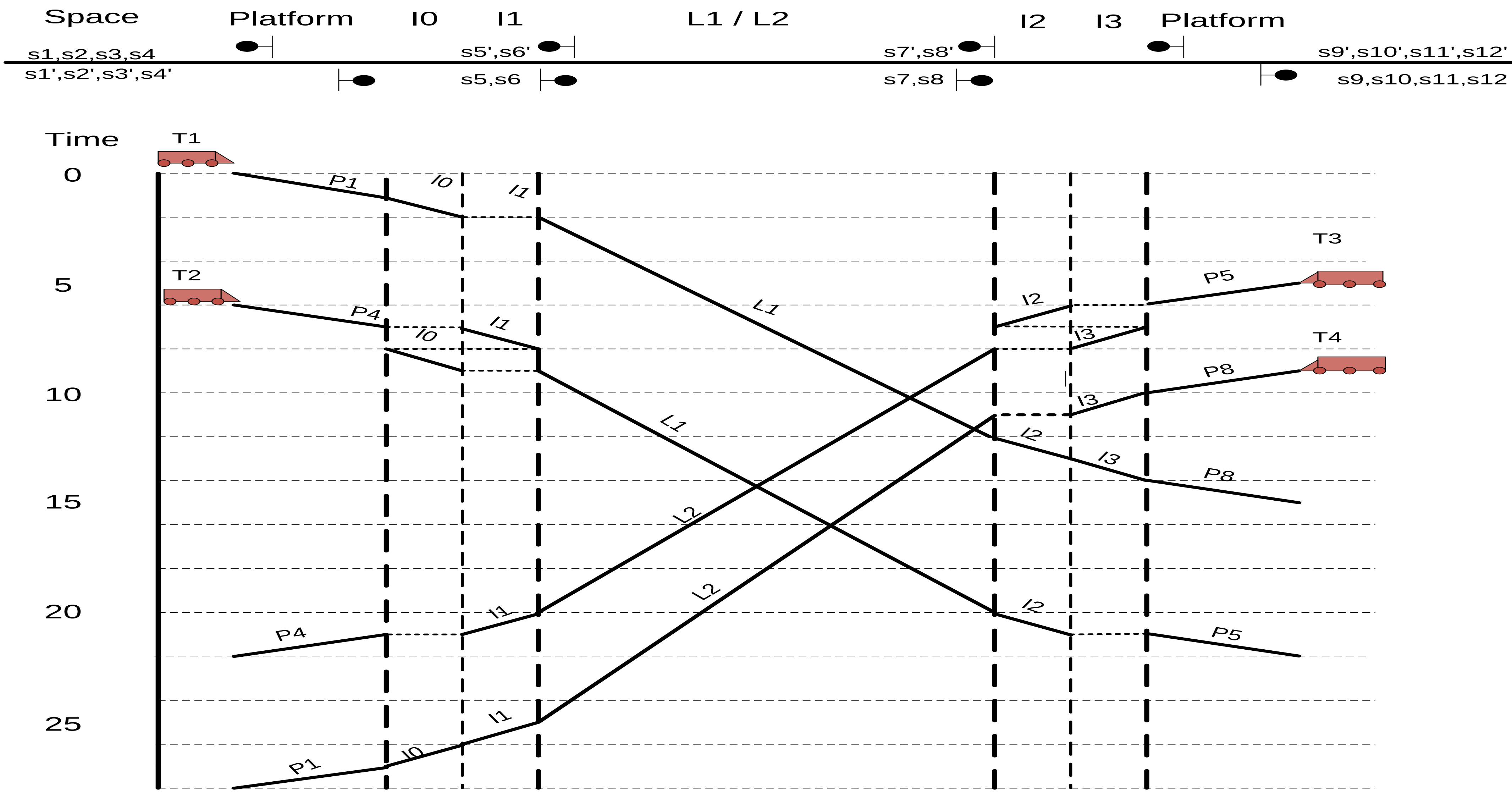
The *schedulead headway* between two trains consist of minimum line headway plus a required buffer time to compensate small train delays on the particular line.

Scheduling Running Time: The pure running time between scheduled stops

The *dwell time* (timpul de oprire) at scheduled stops.

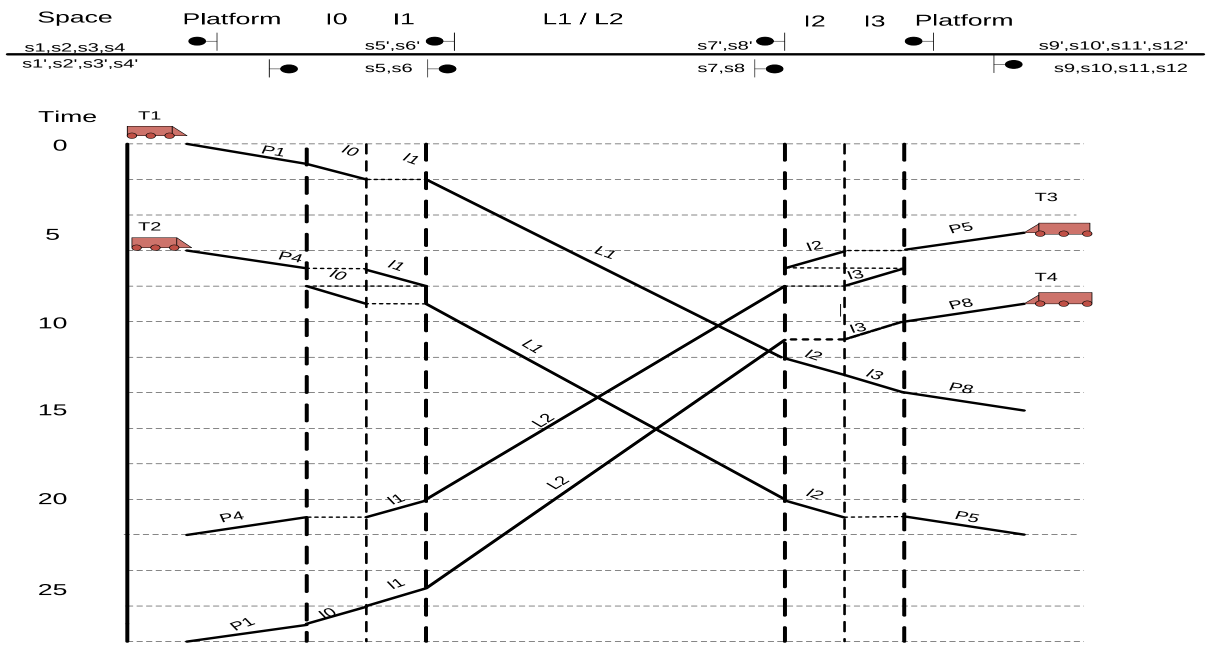
Recovery time & Scheduled waiting time


Station A
Station B
Trafic Light
Interlocking
Detector
T 1
T 2
T 3
T 4
P 1
P 2
P 3
P 4
P 5
P 6
P 7
P 8
s 1'
s 1
s2'
s2
s3'
s3
s 4'
s 4
s5
s5'
s 6
s6'
s7
s7'
s8
s8'
s9
s9'
s 10
s10'
s 11
s11'
s 12
s12'
I0
I1
I2
I3
L 1
L 2
Squeeze it ↓



The train scheduling between two neighbor stations

Model features:
- behavior
- resource allocation



Time Min.	R1	R2	R3	…	…	…	Rn
1	T1				T2		
2		T1		T2			
3		***T1***	***T2***				
…		***T2***	***T1***				
….		***T2***	***T1***				
….	T2			T1			
….	T2			T1			

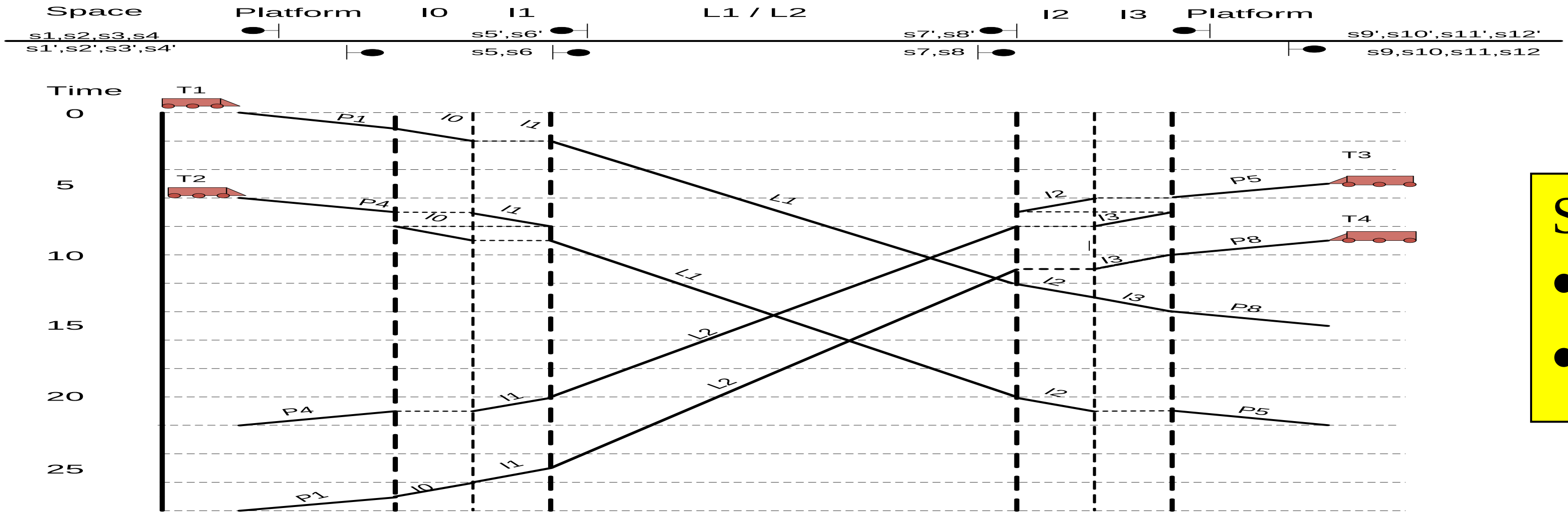
Scheduling results: *Train Time Table (TTT)*

Notations:

- Ti – train identifier
- Ri – resurce identifier (lines, platforms, interlockings)

Inferences from TTT: interlockings and traffic lights settings

The trains schedule can be represented on a resource table:



Resource	P1	P2	P3	P4	P5	P6	P7	P8	I0	I1	L1	L2	I2	I3
Min.														
0	T1	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	-	-	-	-	-	-	T1	-	-	-	-	-
2	-	-	-	-	-	-	-	-	-	-	T1	-	-	-
3	-	-	-	-	-	-	-	-	-	-	T1	-	-	-
4											T1			
5								T3			T1			
6													T3	
7														T3
8												L2		

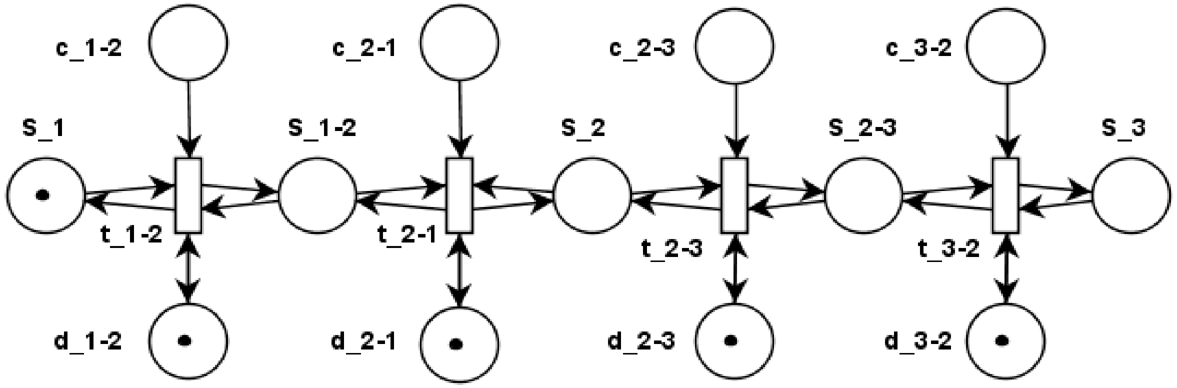
The resource table contains the train schedules for each minute.

Scheduling performed:
- offline
- online

Railway Traffic Control Problems

- train movement (routing) planning
- inter-station path scheduling (and reservation)
- traffic control
- traffic monitoring
- safety warning and protection
- train traffic information
- train traffic resilience

Moving (Mobile) Entity (ME)



A physical (here a train) ME moves on the (requested) route specified by the sequence:

$route= s_1[w_1]*s_2[w_2]*s_3[w_3]$

where s_1, s_2 and s_3 represent the stations (modeled by places) and w_1, w_2, w_3 the rest (dwell) times.

Model role: to simulate the train moves and to determine its arrival times.

Model features:
- structure
- interactions
- behavior

The ME's *train moving states* are modeled by s_{1-2}, s_{2-3}.
The moves are conditioned by *control signals* (move, wait) c_1-2, c_2-1, c_2-3 and c_3-2.

The *structure parameters* (maximum speed, gradient, segment length etc.) are given by the tokens placed in d_{1-2}, d_{2-1}, d_{2-3} and d_{3-2}.

ME has maximum speed capability denoted by v_M The structure allows the maximum speed specified by v_{1-2_M}, v_{2-1_M}, v_{2-3_M} and v_{3-2_M} stored in the tokens injected in d_{1-2}, d_{2-1}, d_{2-3} and d_{3-2}. These maximum speeds can be changed due to the environment conditions to lower speeds denoted by v_{1-2_m}, v_{2-1_m}, v_{2-3_m} and v_{3-2_m} respectively.

Beside the maximum speed, the tokens stored in *d_1-2, …, d_3-2* contains a parameter specifying the traveling distance between the places *s_1, s_2* and *s_3* respectively.
Let ME_d denote the ME achieved dynamics (in the case that control signals c_1-2, …, c_2-3 allow freely the move) described by:

*ME_d= s_1[w_1]*s_1-2[w_1-2] *s_2[w_2]*s_2-3[w_2-3]* s_3[w_3]*

Unlike the demanded waiting times *w_1*, *w_2* and *w_3*, the durations *w_1-2*, *w_2-3* are calculated considering that ME moves in the states s_1-2 and s_2-3 with the maximum speed between its own (i.e. v_M) and the maximum allowed speed (i.e. v_1-2_m, v_2-3_m or v_1-2_M, v_2-3_M). The mappings eet_t and let_t use for the segments s_1-2, s_2-3 the speeds v_1-2_M and v_1-2_m respectively.

Let's have a *traffic (environment) assessment or estimator* that evaluates the environment conditions and stores in the places *d_1-2, …, d_3-2* the probable maximum speeds on the given segments v_1-2_p, v_2-3_p.

Using these probable speeds, the probable arrival times can be determined. Some estimators need the time when the resource is demanded to assess with higher accuracy the probable maximum speeds.

The free move was assumed in the previous calculus, but the *control system* can delay some departures due to the temporarily lack of free (unreserved) paths.

These delays should be added to eet_t and let_t mappings used for calculus of state elements s_1-2, s_2-3 durations.

Fig. Infrastructure model

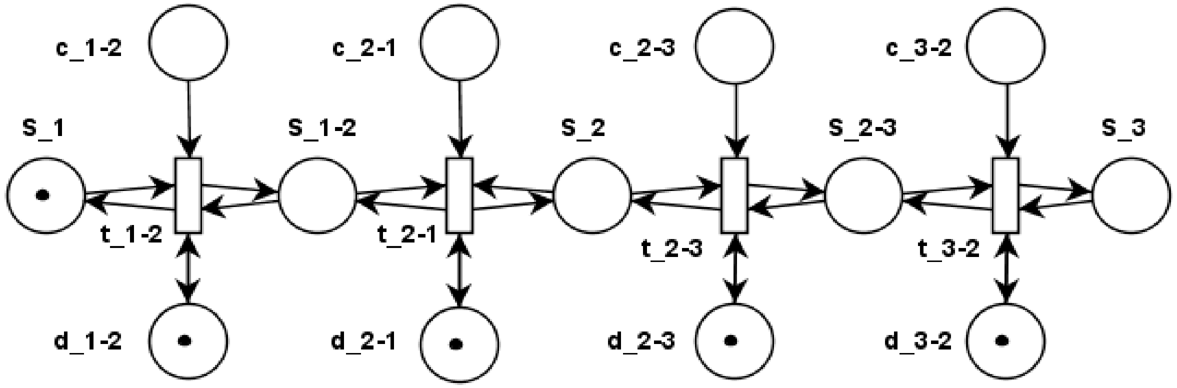
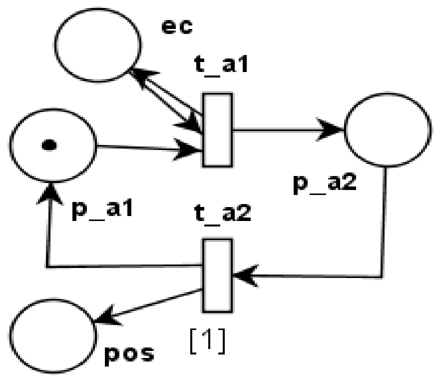


Fig. A Simplest train model

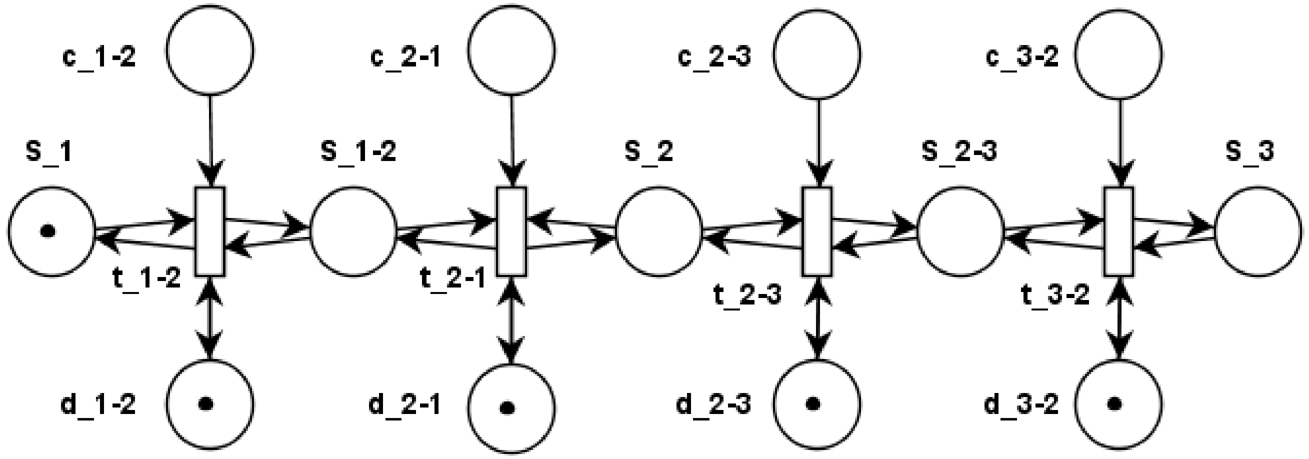


In a simulation with a set of moving trains can be determined the trains' positions using a simple train model. The place *ec* can be used to set information from train's environment (control signals, segment maximum speed etc.) or receive information from the train state. The place *pos* can be used to provide the train localization.

The train model includes parameters of train movement capabilities.

Implementation: a task for infrastructure and a task for each train.

Real-Time Moving Entity
(R-TME)



A R-TME fulfills the R-T constraints even if the environment conditions are changing during travel.

The R-T constraints can be given by specifying the departure and arrival times. These are related to start and end transitions:

requestedRoute= t_1-2[w_1-2]*t_2-1[w_2-1]*t_2-3[w_2-3]*t_3-2[w_3-2]

Actually, the possible route becomes:

possibleRoute= t_1-2[eet_1-2, let_1-2]*t_2-1[eet_2-1,let_2-1]* t_2-3[eet_2-3, let_2-3]*t_3-2[[eet_3-2,let_3-2]

Open loop control

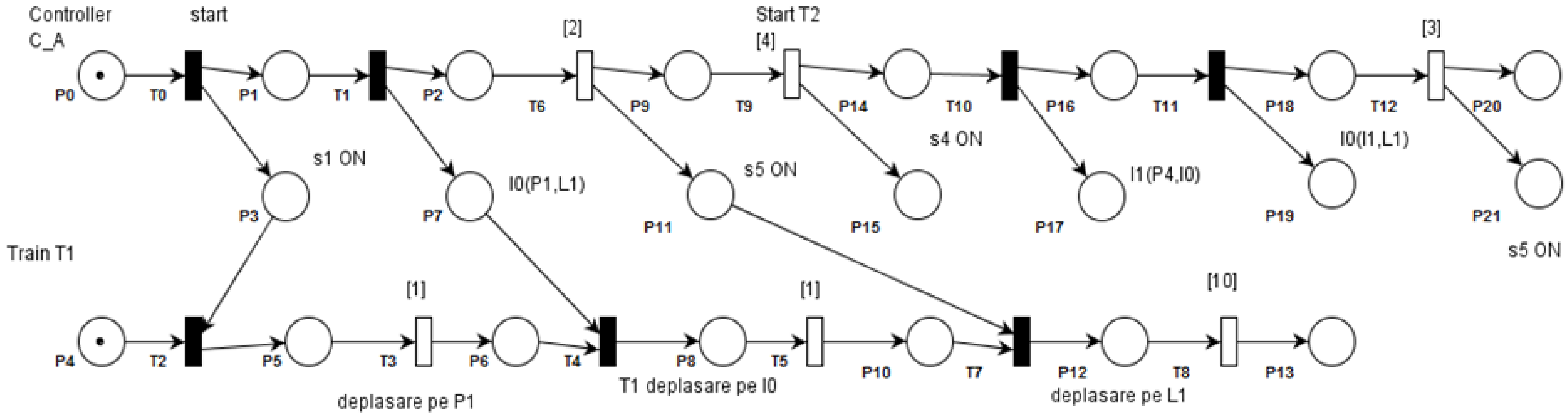


Fig. 5. Time Petri Net of TTC Open loop.

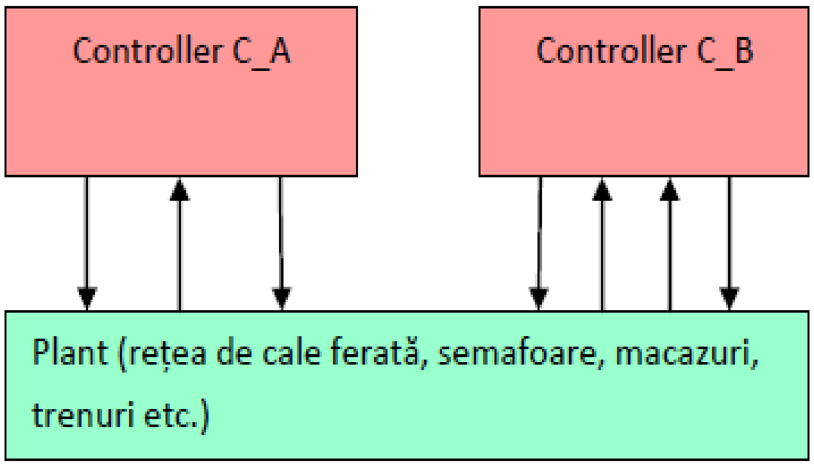
The controller implements the train schedule.
The trains are not identified and their moves are not traced.
The resource states are not monitored.

Notice: Each train benefits of a controller thread that applies the control signals and monitor the train movement until the train exits the controller's assigned zone.

Closed loop control

Each resource R_i (i=1,2,…) has added a detector d_i (i=1,2,…). Each platform (resource) P_i (i=1,2,…) has added a detector d'_i (i=1,2,…). A detector can signal the move or the presence of a train on it.

Assignments: Trains T1 and T2 move on L1. Train T3 and T4 move on L2.



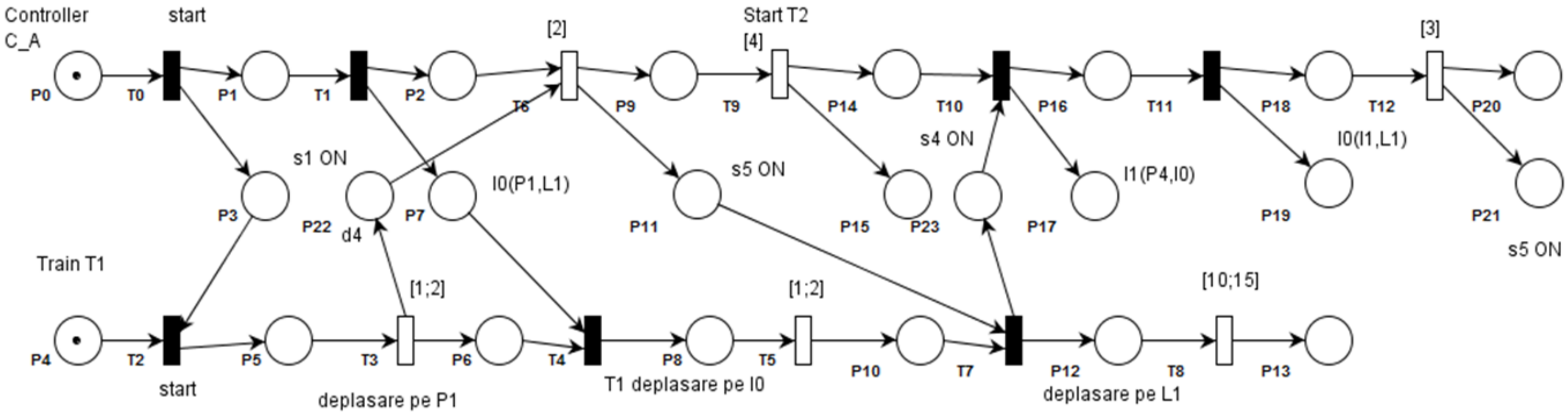


Fig. 8. Time Petri Net of TTC ***Closed Loop*** – independent controllers.

Train Traffic Control. Closed Loop - Coordinated Controllers (Cooperative controllers)

The trains have significant delays.

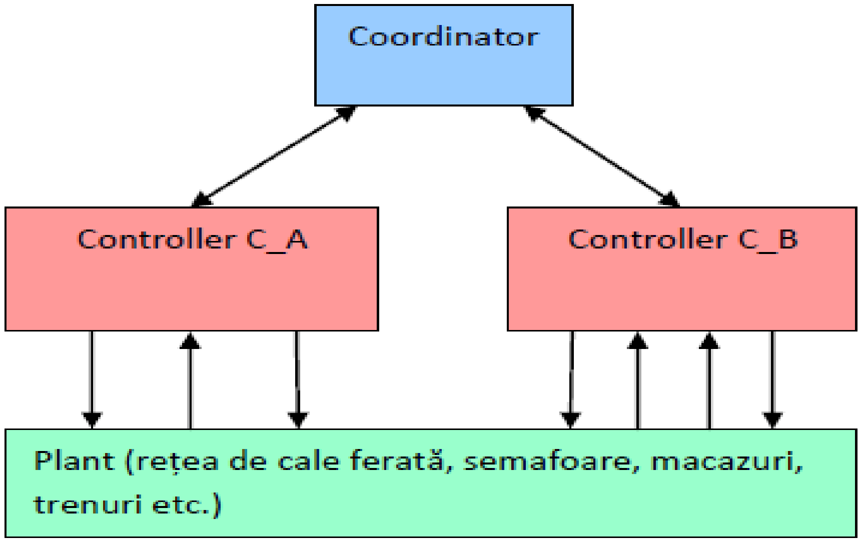


Fig. 9. TTC System Architecture - closed loop.

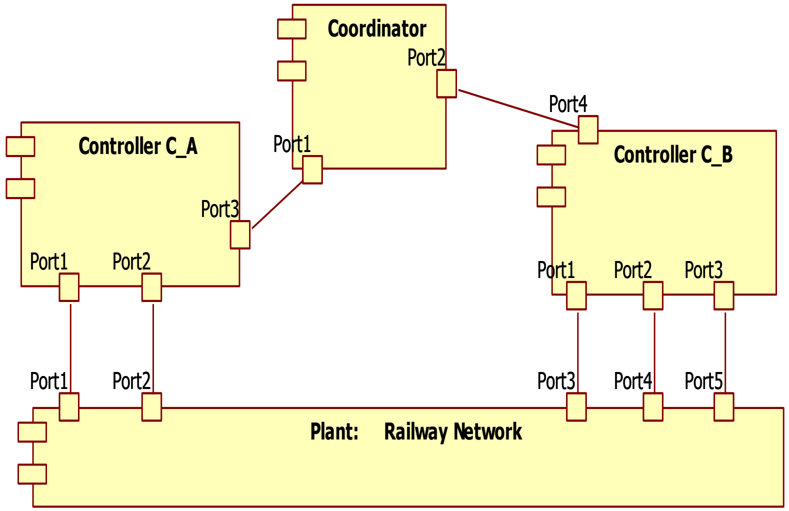
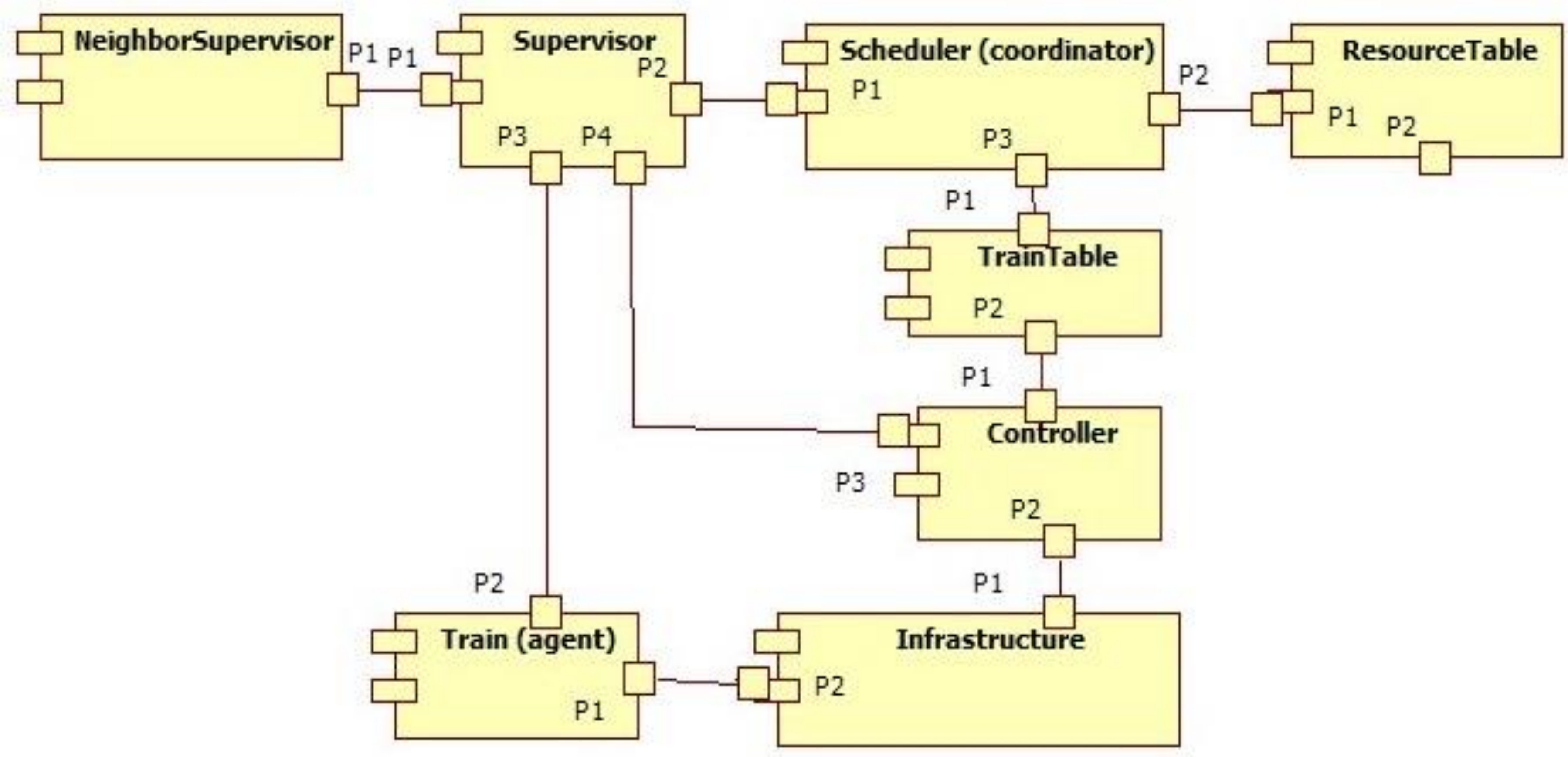


Fig. 9. TTC System Architecture - closed loop.

Fig.10. Component diagram of TTC – closed loop.



Supervisor request a train scheduling according to its route and *Resource Table*.
Scheduler performs the demands setting the *Train Table*.
Controller sets the involved control signals.
Train agent (or Supervisor) receives the *movement authority* and starts its journey.
Travelling across the frontier needs the cooperation with *Neighbor Supervisor*.

Coordination algorithm:

initialize: the critical resource states as not reserved;
while(*true*)
  receive a message from a *supervisor or train agent*
    if the message is *request(resource, train)*
      if the resource is *released*
          answer *true*;
          mark the resource *reserved*;
      else answer *false*;
    if the message is *release(resource, train)*
        mark the resource *released*;
        notify the complementary supervisor (train agent ~Agent_i) about the *crossing event*;
end while;

Scheduler algorithm:

input: PathList, trainSpecification;

initialization: mark the train *not scheduled*;

output: train schedule;

wait(*start*);

while (train is *not scheduled*)

  do

    choose a path from the PathList and try to reserve it;

    if reservation is obtained *request(critical resource, train)*;

      receive(answer);

      if (*answer* is *true*) mark the train *scheduled*;

      else cancel the reservation;

  while train is *not scheduled* or not all the paths from the PathList were used;

  if (train is *scheduled*)

    load the train schedule on the Train (Scheduled) Table;

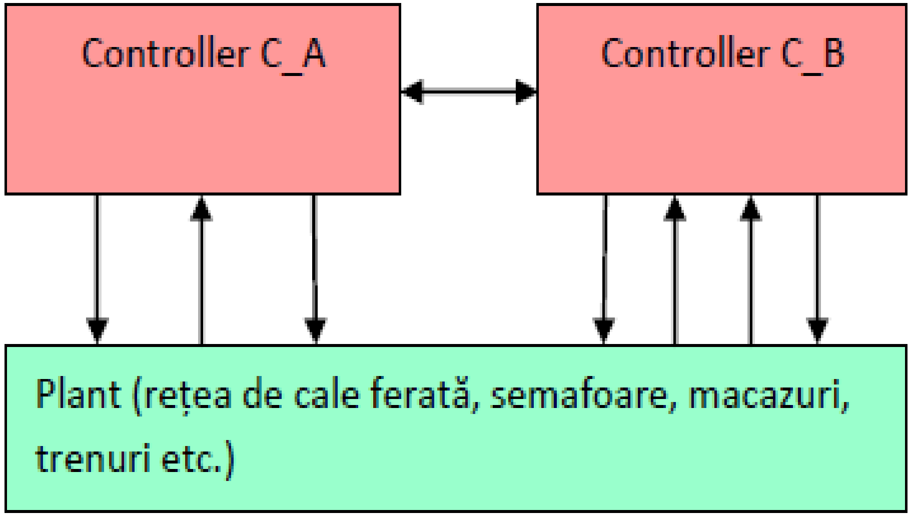
  else *wait* a period and try a new reservation;

end while;

What are the *critical resources*? ← Those used for crossing the frontier of the two zones.

Train Traffic Control. Closed Loop - Heterarchical Supervisors (Cooperative supervisors) ➔ No coordinator!

The trains have significant delays.



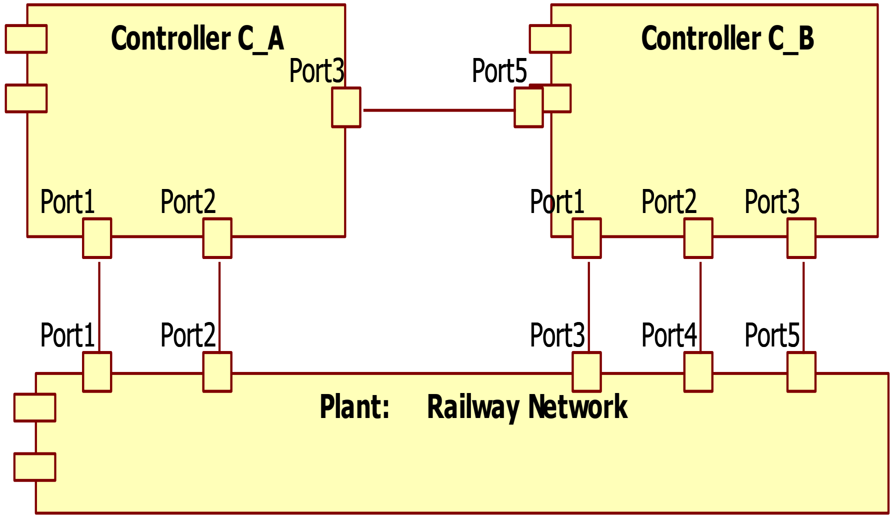


Fig. 11. TTC System Architecture - closed loop.

Fig.12. Component diagram of TTC .

Heterarchical_agent_i request algorithm: \\i.e. supervisor agent i

input: *trainPathList*, *trainSpecification*;
initialization: mark the train *not scheduled*
output: train schedule;
wait(start);
while (train is *not scheduled*)
    do //path reservation
        * choose a path from the trainPathList and try to reserve it;
        if the reservation is obtained *request(critical resource, train, periodOfTime)*;
            receive(answer);
        if (*answer* is *true*) mark the train *scheduled*;
        else release the reservation;
 while train is *not scheduled* or not all the paths from the *trainPathList* was used;
 if (train is *scheduled*)
    * load the train *schedule* on the Train Schedule Table;
    wait(cross event);\\from local controller
    signal(cross event);\\the complementary supervisor agent
 else wait a period and try a new reservation;
end while;

5. Monitoring

It is a part of Automatic Train Protection (ATP)

Use-Case name: **Monitoring**

Summary:

- ➔ Monitorizează toate evenimentele petrecute sau semnalate sistemului de control.
- ➔ Furnizează sistemului de control informaţii despre evenimentele produse.
- ➔ Afişează pe ecran (într-o fereastră grafică) starea curentă a sistemului.

Dependency: Control şi Information.

Actors: DataBase

Pre-condition: Monitoring este startat după intrarea în funcţiune a Simulatorului.

Description:

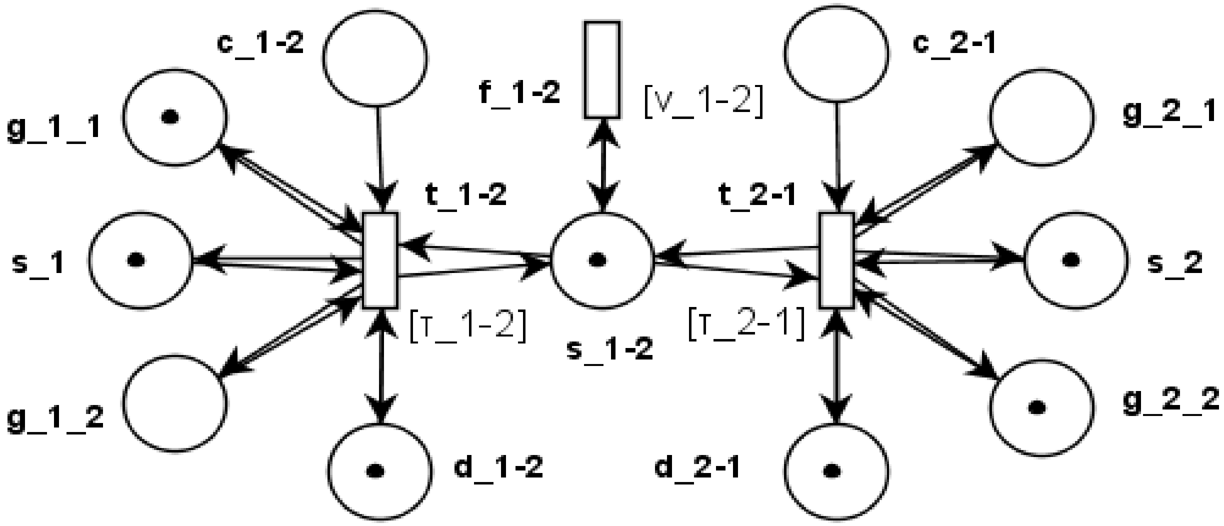
- ➔ Se fac legăturile cu sistemul de control.
- ➔ Este semnalat despre toate modificările porturilor de intrare şi ieşire.
- ➔ Furnizează informaţii şi semnalizări către Control şi Information
- ➔ Lucrează concurent cu Control, Information şi Simulator.
- ➔ Informaţiile stocate sunt de forma:
  - o EventIdentifier
  - o Place
  - o Time
  - o EventProducer (sau EventCause)
- ➔ Comunică la cere componentei Control toate informaţiile cerute.
- ➔ Afişează pe ecran:
  - o structura subreţelei de cale ferată corespunzătoare gării,
  - o informaţii despre fiecare tren cum ar fi:

        - identificatorul
        - poziţia
        - viteza
        - starea
    - starea macazurilor
    - starea semafoarelor
    - semnalizările detectoarelor
    - cererile de transmitere şi recepţionare a trenurilor dintr-o gară în alta
    - răspunsurile sistemului de control la cererile de transfer a trenurilor, inclusiv a celui vecin

Alternatives: Nu există.

Post-condition: - Îşi opreşte activitatea la închiderea aplicaţiei înainte de simulator, dar după Control.

6. Information System



It provides information (departure, arrival time) to travelers.

Homework: conceive a software application for information system. Requirements:

- specification
- design diagrams
- example of relevant codes

Fill for the attached OER-TPN model the missing information (or conceive a new one):

- Types
- Guards,
- Mappings
- Eet, let

s_1, s_2: stations
s_1-2 : line between stations
g_1_1, g_1_2: gates
d_1-2, d_2-1: environment conditions (distance, speed, etc.)
c_1-2, c_2-1: controller signals
f_1-2: position updater

*

***

*****

END

*****

***

*