

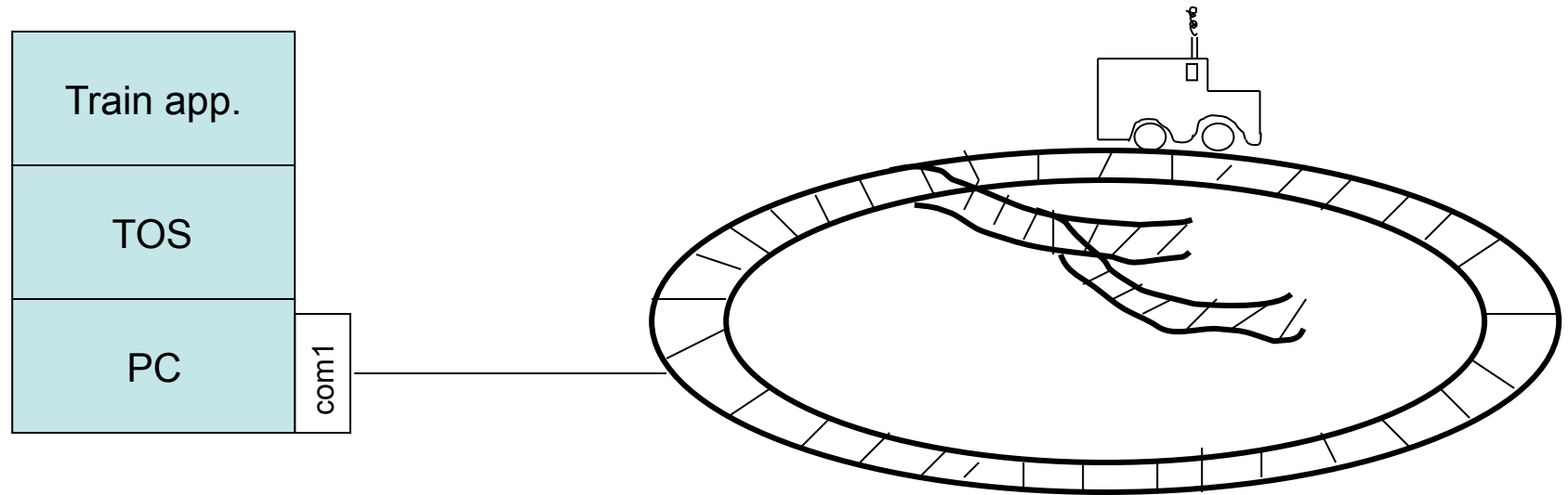
# *Model Train*



# Objectives

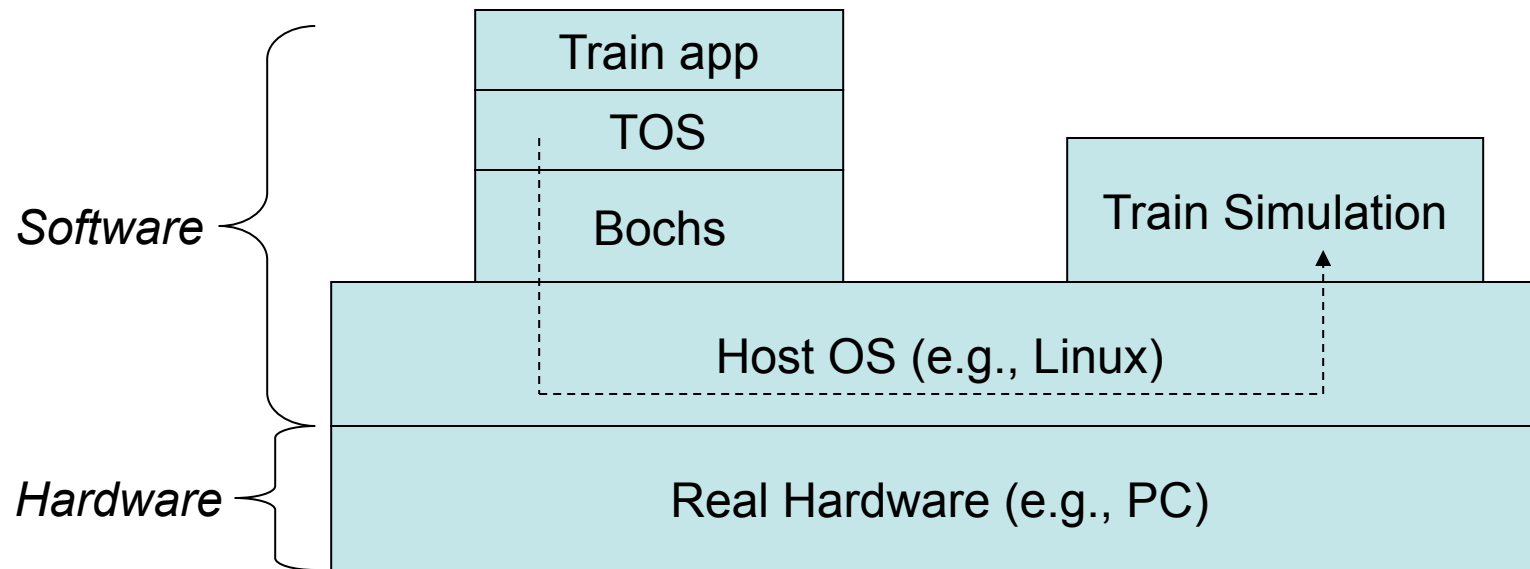
- Introduce the train simulator
- Using the model train
- Hints for writing the train application

# Train Setup

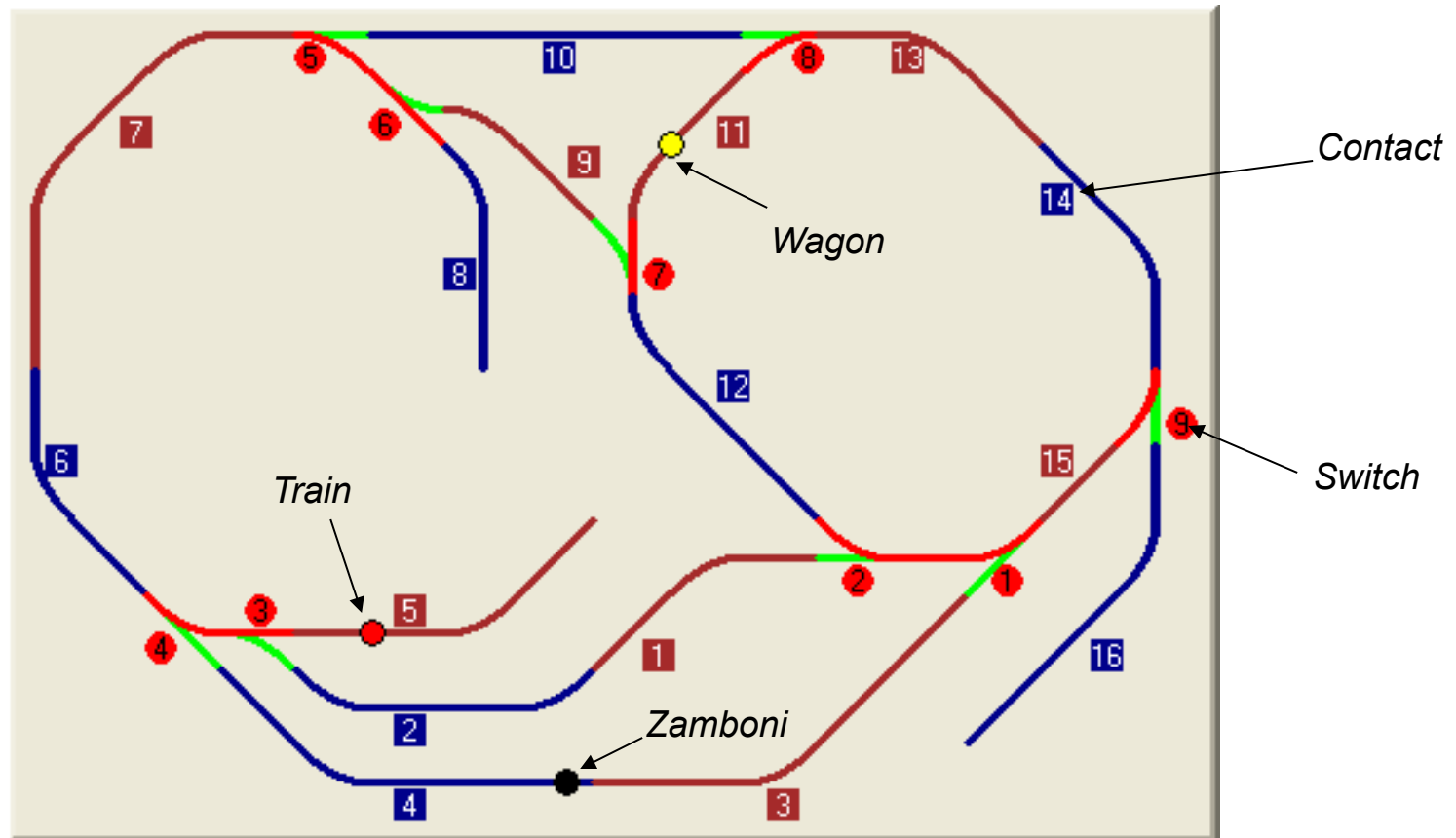


- Train application runs on top of TOS
- TOS implements various operating system functions including a serial line device driver
- Serial line (com1) of the PC is connected to the train
- Commands that control the train are sent via the serial line

# Train Simulation



# Labels and Switch Settings



- Numbers in square are contact IDs
- Numbers in circle are switch IDs.
- Color green stands for 'G' setting of a switch. Color red stands for 'R' setting of a switch.

# Train Commands (1)

Command	Function	Examples	Note
<i>L#D{CR}</i> #: vehicle ID	Change the direction of a vehicle. Train ID is 20.	To change the direction of the train: “L20D\015”	You cannot change the direction of any vehicle other than the train.
<i>L#S#{CR}</i> 1 <sup>st</sup> #: vehicle ID 2 <sup>nd</sup> #: speed, 0-5	Change the speed of a vehicle.	To change the speed of the train to 4: “L20S4\015” To stop the train: “L20S0\015”	1. You cannot change the speed of any vehicle other than the train. 2. On the real system, speeds 1-3 do not work well.

Notation:

- #: a number.
- {CR}: carriage return. (see Note 1.)

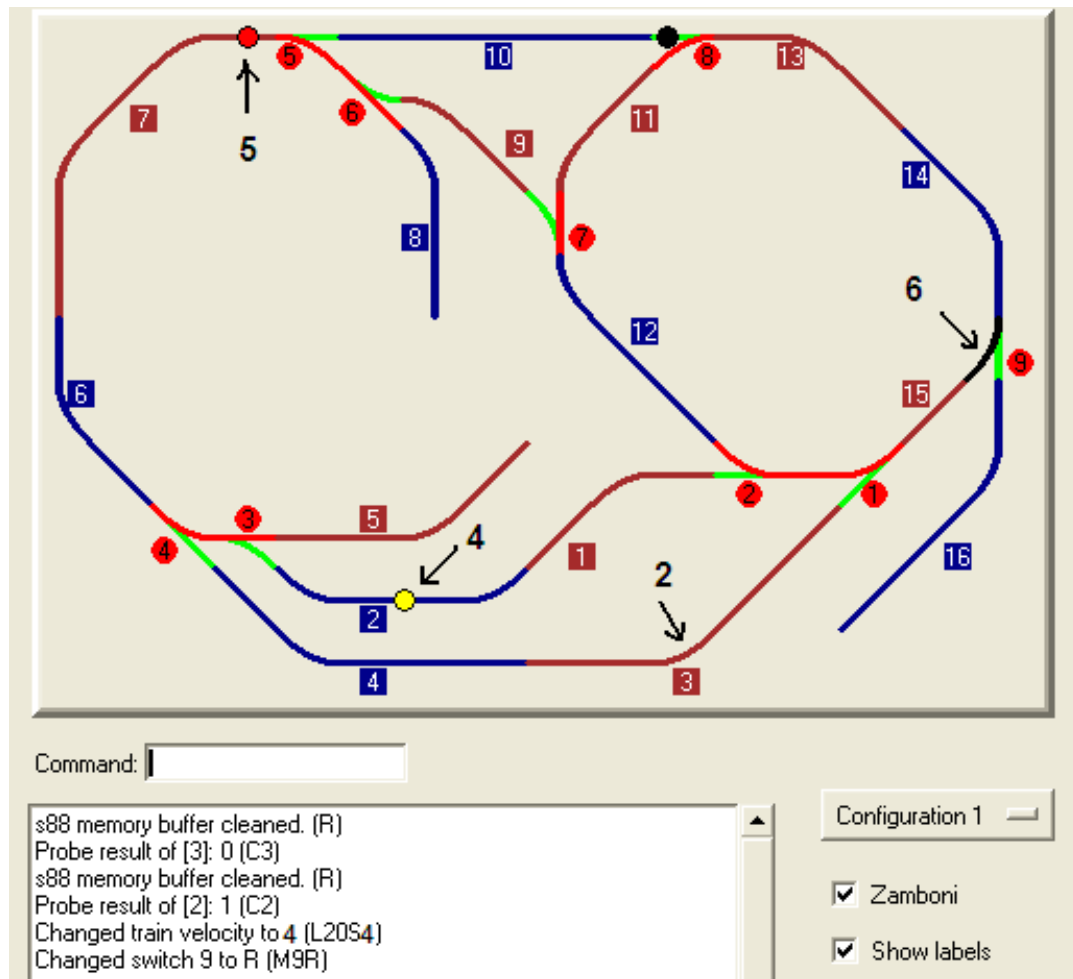
# Train Commands (2)

Command	Function	Examples	Note
<i>M#x{CR}</i> #: switch ID X: 'R' or 'G'	Change a switch to 'G' or 'R'.  'G' and 'R' are the two possible settings of a switch.	To set switch 5 to "R": 'M5R\015'	The initial setting of a switch can be either 'G' or 'R'
<i>R{CR}</i>	Clear the s88 memory buffer. s88 is the device that controls the contacts.	"R\015"	This command is required for every "C" command.
<i>C#{CR}</i> #: contact ID	Get the status of a contact. "*1\015" is returned if there is a vehicle on the contact. Otherwise, "*0\015" is returned. (see Note 2)	To know if any vehicle is on contact 3: "C3\015"	1. A contact is a track segment with a sensor. 2. Must be preceded by a "R" command.

# An Example

Commands sent:

1. `"R\015"`  
Clear memory buffer
2. `"C3\015"`  
Probe contact 3.  
Result is 0.
3. `"R\015"`  
Clear memory buffer
4. `"C3\015"`  
Probe contact 2.  
Result is 1.
5. `"L20S4\015"`  
Set train speed to 4
6. `"M9R\015"`  
Set switch 9 to red.





# Note –About the Commands

## 1. {CR} -- carriage return

- The ASCII of carriage return is 13. In C, you can use ‘\015’.
- In the simulator, if a command is from the keyboard, ‘\015’ is not needed, i.e. you only type “L20S5” to set train speed. But if a command is sent from TOS, “\015” is needed, i.e. you must send “L20S5\015” (6 characters)

## 2. “C” command

The result is a string: “\*0\015” or “\*1\015”, where the first character is ‘\*’ and the third character is a carriage return. The information you want is the second character: ‘0’ for empty contact and ‘1’ for occupied contact.

## 3. Pause between commands

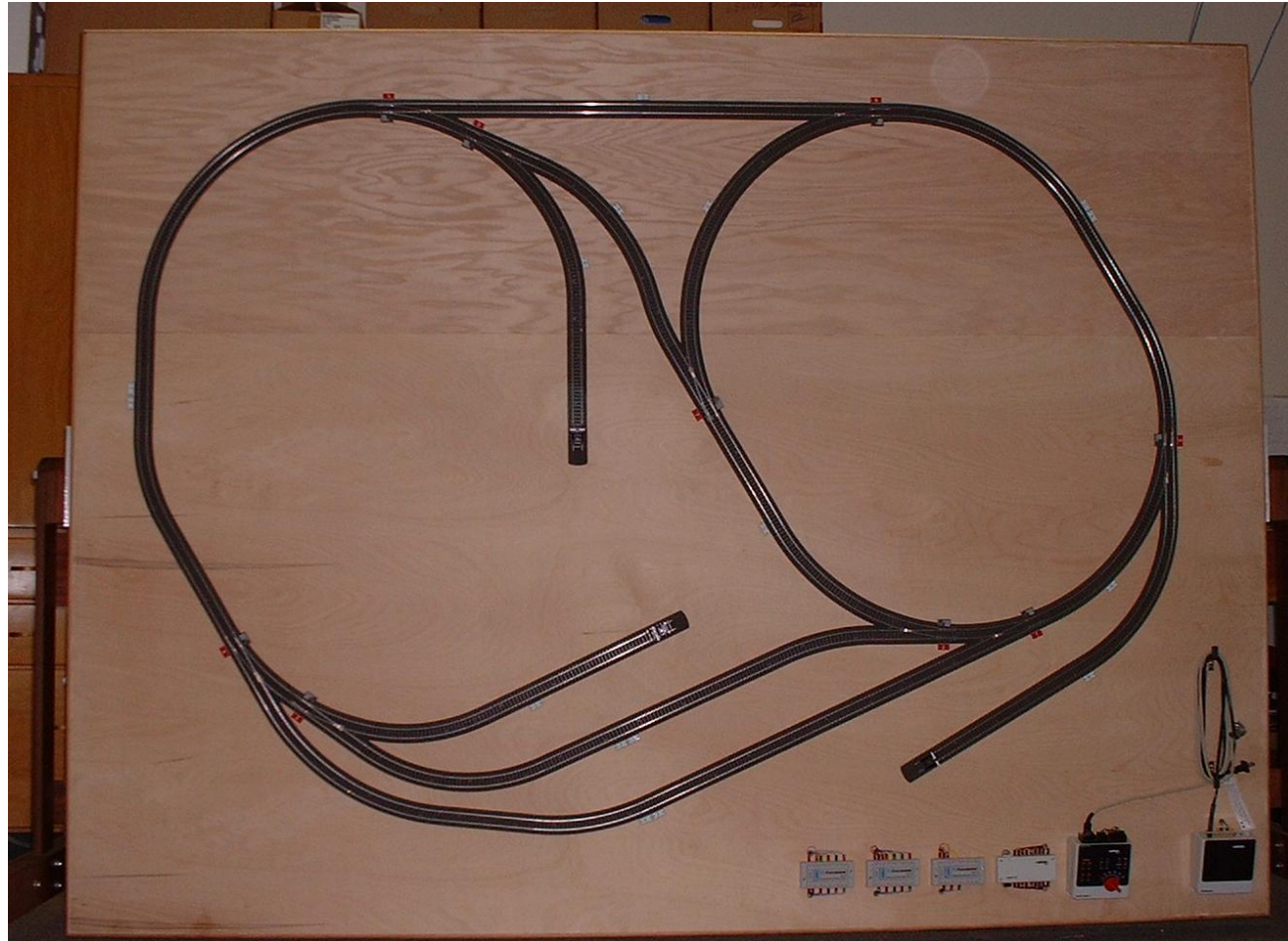
A pause is required between commands. You can do so by sleeping, for example, 15 ticks. (Hint, define a variable for the number of ticks to sleep. You may have to change it when you try to run on the real system)

# *Train Application*

# Train Application

- There are four different configurations.
- For each configuration, Zamboni can be enabled or disabled.
- Therefore, there are 8 different permutations for the initial setup.
- Your train app has to find out at runtime, which configuration was selected (hint: use the „C“-command to probe a track segment).
- The goal is to retrieve the abandoned wagon: the train has to rendezvous with the wagon and return to the home base while avoiding Zamboni

# Model Train



# Note – About the Real System

## 1. Reset the system

1. Press the “off”, “stop”, and “go” buttons at the same time.

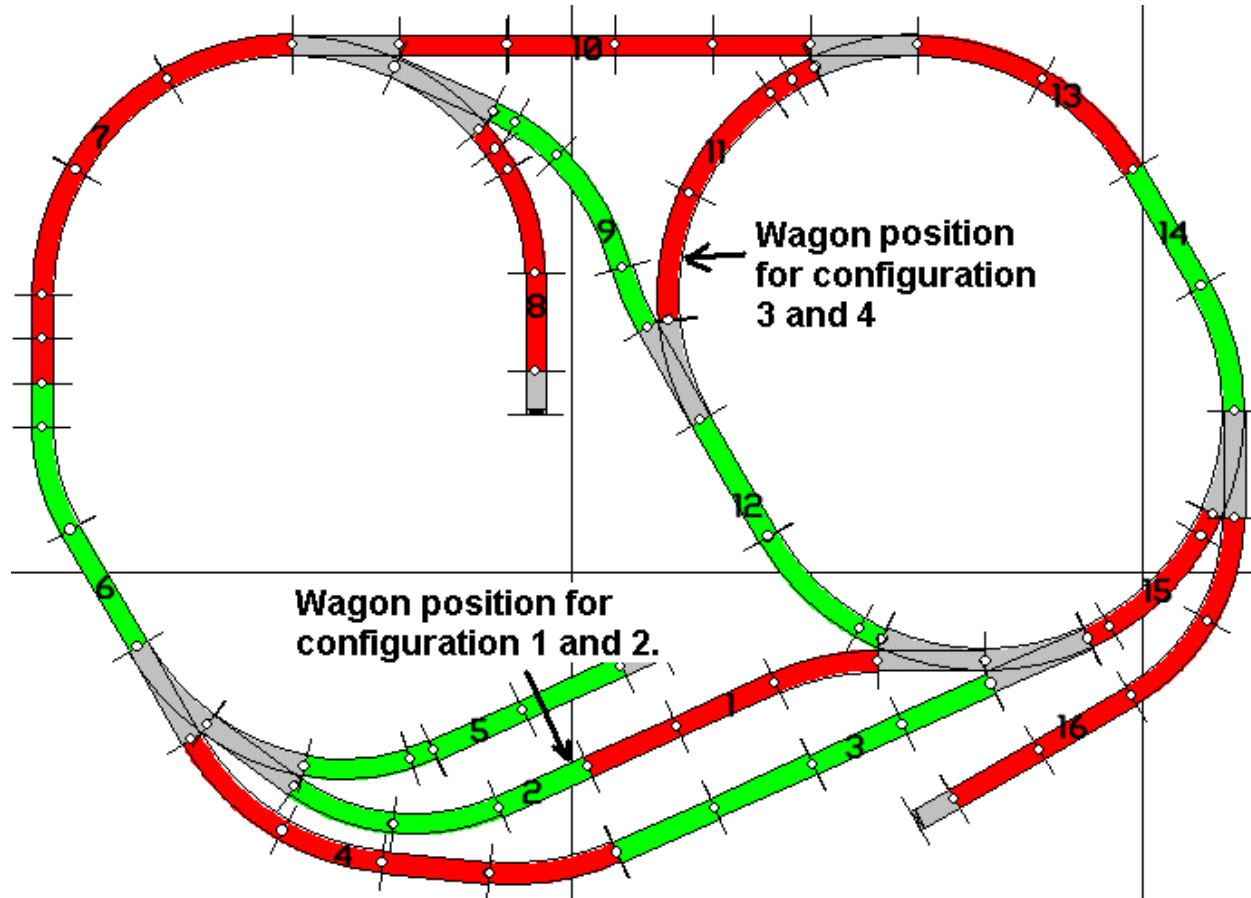
2. Press “go”.

## 2. Place the wagon

In order to get the wagon attached to the train, you much place it at certain positions. See the next slide.



# Wagon Positions



# Hints

## 1. Initialize the switches

- The first thing you should do in your application is to set the switches so that Zamboni, if exists, will not run out of track.

## 2. Timing issues

- The timing of the simulator is not exactly the same as the timing of the real system
- Play the game based on probing results, but not the time.
- Leave as much “margin” as possible. For example, do not get too close to Zamboni; do not set a switch just before a vehicle reaches it.

## 3. Application design

When trying on the real system, you may need to modify your application somehow. So, design you application in a way in which modifications are easy to make. (Define and use functions, define a variable for sleep time, etc.)



# Assignment 11

- Implement the train application.
- For further details see:  
<http://pear.sfsu.edu/csc720/train.html>