Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

ASSESSING TRAM SCHEDULES USING A LIBRARY OF SIMULATION COMPONENTS

Elisangela Mieko Kanacilo
Alexander Verbraeck

Systems Engineering Group
Faculty of Technology, Policy and Management
Delft University of Technology
Jaffalaan, 5, 2628BX, Delft, THE NETHERLANDS

ABSTRACT

Assessing tram schedules is important to assure an efficient use of infrastructure and for the provision of a good quality service. Most existing infrastructure modeling tools provide support to assess an individual aspect of rail systems in isolation, and do not provide enough flexibility to assess many aspects that influence system performance at once. We propose a library of simulation components that enable rail designers to assess different system configurations. In this paper we show how we implemented some basic safety measures used in rail systems such as: reaction to control objects (e.g. traffic lights), priority rules, and block safety systems.

1 INTRODUCTION

A **Tram System**, also referred as **Streetcar** or **Trolley System** (Edwards 1992, Vuchic 2004) is a type of rail system driving on tracks in streets, sharing space with cars, buses and pedestrians.

As tram systems represent public services, the level of service is an important requirement. When planning services for transit systems in general, planners should look for strategies that will provide cost effective operation and adequate services for the public (Harris and Godward 1992, Sussman 2000, Vuchic 2004, Edwards 1992).

The task of creating schedules is of huge importance as this defines how good the system fulfill the passenger demand, with a certain minimum frequency of service. A good level of service means that transport service is reliable, easily accessible in time and space and provides a safe, fast and comfortable ride at a reasonable price (Wiransinghe 2003).

Several commercial simulation tools are available to support the assessment of tram schedules. The disadvantages of these tools are that they assess schedules in isolation and do not consider or do not provide enough flexibility to assess other aspects at the same time. For example, it might be that the strategies used to improve timetables and the ones used to decrease delay propagation will not produce a good system performance, when these measures are combined in the same scenario.

As advocated by Ceder (2003), different aspects of the rail system should be analyzed at the same time, to exploit the system's capability to the greatest extent and maximize the system's productivity and efficiency. Therefore, we propose a library of simulation components to enable schedule makers in assessing timetables at different system configurations before commissioning them.

In this paper, we give a detailed explanation of how we modeled some basic rules used in rail systems, such as vehicle reaction to control objects, e.g. traffic lights, priority measures and block system as a safety mechanism. Section 2 covers the challenges in creating tram schedules for which the designers have to take several aspects into account. In section 3, we show the Java-based library of rail-specific simulation components, based on the DSOL simulation environment (Jacobs, Lang, and Verbraeck 2002, Lang, Jacobs, and Verbraeck 2003, Jacobs 2005) that was created to research tram schedules in relation to control systems. Section 4 illustrates the use of the library for a concrete case study that was carried out for a tram company, followed by an evaluation and conclusions in section 5.

2 CHALLENGES IN CREATING TRAM SCHEDULES

Scheduling is one major concerns of transit management (Black 1995). It directly affects operating costs and schedules can vary widely in their efficiency. Schedule of rail lines must satisfy two requirements (Vuchic 2004):

- it must provide adequate transporting capacity for the passenger volume (number of transit units)
- it must offer a certain minimum frequency of service

In addition, when creating schedules, planners aim at a good system performance. There are many factors affecting the system performance of rail systems, such as travel time, average speed, punctuality, regularity of vehicles and others (Edwards 1992, Black 1995).

Operation strategies, such as setting speed limits and priority rules, also influence the performance and they should be combined and assessed together with scheduling decisions. Assessing these strategies in combination with schedule allows for a better exploration of alternatives for a good service planning according to the system capacity.

Several tools to for rail-based infrastructure modeling have been described in literature. RailSys is a supporting tool to create timetables for rail networks where a block safety mechanism is in use. Rudolph and Demitz (2003) and Demitz, Hübschen, and Albrecht (2004) describe the application of RailSys to a rail network and how it can improve the timetable of a system. Kraft (1982) proposed a model to analyze the effects of unexpected delays in the scheduling for the purpose of analyzing jam capacity. Higgings, Ferreira, and Kozan (1995) analyze the risks of delay for each train and for the schedule as a whole, by giving weigh to delays. SIMONE is a Dutch simulation package used to assess timetables of large scale rail networks. In Middelkoop and Bouwman (2000) and Middelkoop and Bouwman (2001) the authors apply SIMONE to evaluate and compare many traffic scenarios in The Netherlands. SIMON (Wahlborg 1996, Bergmark 1996) is a Swedish software package used to simulate the whole train network, and UX-SIMU (Kaas 2000) is used in Denmark to simulate rail systems traffic.

Although existing tools are very useful to test one aspect of the control design in isolation, like for example timetable assessment, delay propagation, or traffic analysis, they do not provide support for, or it becomes hard to analyze all these aspects at the same time in the model. For example, it might be that the combination of control strategies used to improve timetables and the ones used to decrease delay propagation, will not produce a good system performance, when they are combined in the same scenario.

As advocated by Ceder (2003), different aspects of the rail system should be analyzed at the same time, to exploit the systems capability to the greatest extent and maximize the systems productivity and efficiency.

In the next section, we present a library of simulation components that we propose to support the assessment of tram schedules and other operational strategies in a joint model.

3 LIBRARY OF SIMULATION COMPONENTS

We propose a library of simulation components organized in a service oriented way (Papazoglou 2003). In Kanacilo and Verbraeck (2006), we explained how we structured the library in service oriented architecture. Simulation components have been developed in Java and they are clustered by the type of services they offer. See Figure 1.

The left side of the figure contains the components for input processing. These are components to check and validate input data, with the XML input file parser from which the whole model will be created as the most important component. In the middle the components to create the simulation model are present. These are separated into layers:

- **Physical Layer** creates the physical objects, like for example traffic light and sensors.
- **Control Layer** creates the logic behind each physical objects, for example control part of the traffic light, which in fact defines the different types of traffic light (different logic control).
- **Statistics Layer** creates graphs and other objects to support the calculation of statistics during the simulation run.

On the right side we have the components to support output analysis, like the animation of object interaction and visualization of statistics. For more details about the architecture, see Kanacilo and Verbraeck (2006).

There are some objects in rail systems that are better represented by the DESS formalism (continuous), such as vehicles, while others objects are better represented by DEVS (discrete) formalism, such as sensors and traffic lights. Therefore, to model the interaction among these objects, we use the DEVSDESS formalism which is able to support both DEVS and DESS behavior. More about modeling formalisms can be found in Zeigler, Praehofer, and Kim (2000).

The communication between objects is done through the publish/subscribe mechanism, which is an asynchronous type of communication. The strength of asynchronous communication is that the sender does not block while waiting for an answer of the receiver. Instead, the communication is implemented in such a way that the sender can handle a callback that comes in later. For modeling transportation systems and safety systems this is important, because trams continue to drive, for instance, when an answer for a request does not come back immediately. We used the publish/subscribe mechanism to model the basic safety measures of rail systems. The safety measures described here are: reacting to control objects, modeling of priority rules, and the block system, which will all be explained in the following subsections.

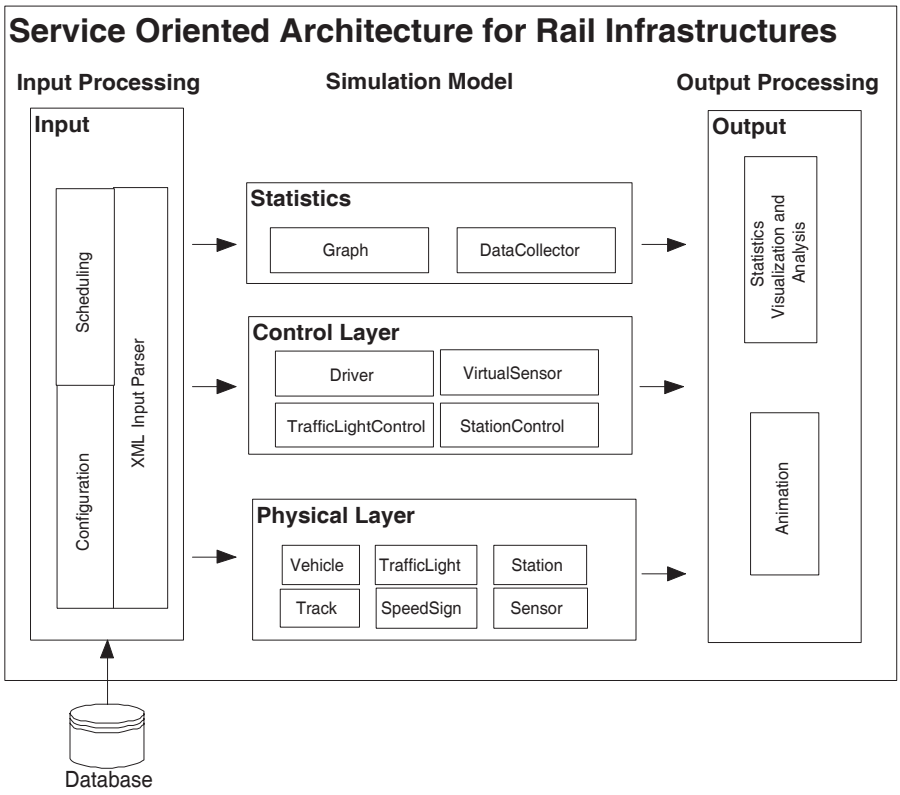

Figure 1: Service oriented architecture.

3.1 Reacting to Control Objects

Control objects refer to any object in the system that will result in changes to the vehicle speed, such as traffic lights, speed signs or other vehicles on the same track.

To make sure the vehicle will stop in time for these objects, a minimum speed-dependent *braking_distance* is calculated for each vehicle type. This calculation is done just once at the beginning of the simulation and stored in a vector for efficiency purposes. The actual braking distance is calculated by interpolating the braking distance between the two nearest speeds. This is done per vehicle type as it takes into account the acceleration/deceleration rate of the vehicle, which is dependent on the vehicle characteristics such as the vehicle weight.

During the simulation, the vehicle is always ”looking for” control objects within a certain distance d ahead, where d= 1.5 x *braking_distance* meters to assure a safe ride, and to accommodate for reaction time of the simulated driver of the vehicle. This does not mean the vehicle will start to brake immediately after a control object is detected, for example in case this control object is a red traffic light, but it is just to make the vehicle “aware” of the obstacle. The vehicle will only start braking, when the distance between the vehicle and the control object is equal to the *braking_distance*

This *awareness* is implemented using a publish-subscribe mechanism (see Kanacilo and Verbraeck 2006). When the traffic light is detected by the vehicle, the vehicle subscribes itself to the *Change_State* event of the traffic light. This means that if the state is changed, regardless what the new state is (green, red or yellow), the vehicle (and all other objects subscribed to this *Change_State* event) will be notified. According to what the new state is, the vehicle will take appropriate action. This reaction could be accelerate or keep constant speed if the state is green, or start braking if the state is changed to red. See Figure 2.

After the vehicle has passed the traffic light location, the vehicle removes itself from the *Change_State* event list, as the state of this traffic light is not relevant anymore to the behavior of the vehicle.

3.2 Modeling Priority Rules

As explained in Vuchic (2004), priority measures for rail systems can be of three types: passive, active and signal preemption. These vary accordingly to the time the traffic light will change to or remain in green phase favoring traffic of the tram line for which the priority is required. In this section, we explain how the signal preemption has been modeled in the library.

Signal preemption gives absolute priority to a certain traffic direction. It means that every time the vehicle coming from the priority direction approaches the traffic light, it

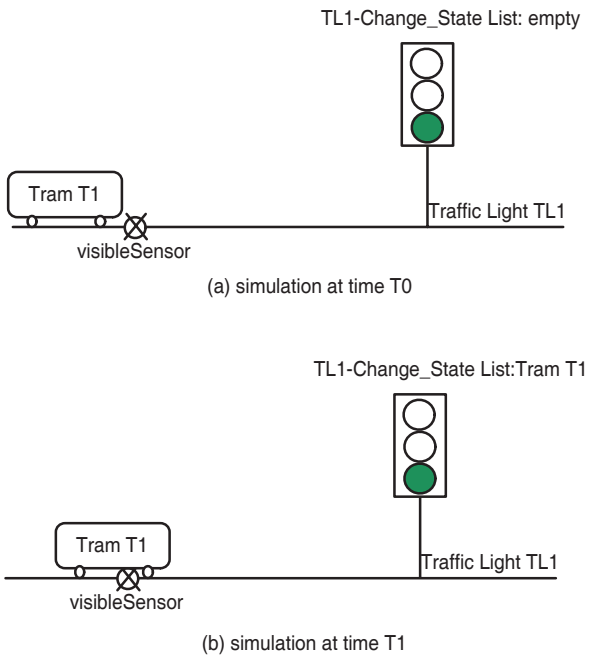

Figure 2: Publish-subscribe mechanism.

will trigger the green phase to start. For safety reasons, the green phase only starts when the way is clear.

Consider the situation of Figure 3. Four directions are possible in this crossing as shown in the figure. Three absolute traffic lights guard this crossing. Absolute traffic lights remain in the red state by default, and every time a vehicle approaches, it triggers a sensor that requests access to the crossing. Only when the access is granted by the control system, the traffic light turns to green. As all three traffic lights are of the absolute type, requests are processed on a first come, first served basis, if no priority is defined for certain tram lines. To implement the priority, requests are stored in a map. If priority is defined, the request map is sorted based on the priority set for the vehicle line. When more than one request is done before access is given (green light), multiple entries will reside in the map, and the tram with the highest priority will be granted access first. We define an *absolute block* as a length of track guarded by an absolute traffic light.

To model this crossing, four objects were used: three absolute blocks which control the accessibility to the area guarded by each absolute traffic light, and one so-called *switch block* to control the crossing, which is based on the occupancy of all three absolute blocks. The switch block object is the central object in charge of processing the requests, determining which traffic light will change to green phase, and when it will change. The vehicle sends a request to access the crossing to the absolute block it wants to enter. The controller for the absolute block delegates the request to the switch block, passing a pointer to the absolute block that sent the request. When the controller

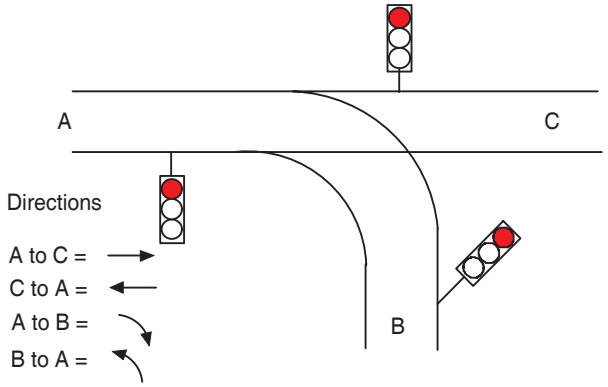

Figure 3: Priority at intersection.

of the switch block grants access to a vehicle, it sends the approval by allowing the absolute block to change the phase of its respective traffic light. As the vehicle is subscribed to the state change of that traffic light, it can start to accelerate again (or to continue driving if it did not decelerate yet) and enter the crossing in a safe way. As long as the vehicle does not receive an answer, it will decelerate as soon as it hits its braking distance to stop just in time for the absolute traffic light. In this way, a safe system can be created where vehicles only trigger sensors and drivers of the vehicles look at the traffic lights, and there is no direct object-object interaction between vehicles and control systems or traffic lights, just like in reality.

3.3 Block System

The block system concept is a safety mechanism commonly used in rail systems. When this mechanism is in use, (part of) the infrastructure is divided into blocks. At the beginning of each block, there is a traffic light controlling the access to the block. Therefore, a block consists of the distance between two successive signals, which will be called traffic lights in this paper. The block length is not fixed and it varies among rail companies. For a tram system, for instance, blocks have a much shorter length than for a high-speed rail system, due to the shorter vehicles, lower weight, and shorter braking distance.

The safety policy determines how many vehicles are allowed to be present in the block at the same time. The number depends on the length of the vehicle and on the length of the block, but normally it is set at one vehicle at a time. For high-speed train safety systems, the policy is often that the maximum speed in a predecessor block for an occupied block is set to half-speed, which is often indicated by a yellow signal. For tram systems, with much lower speeds, and shorter braking distances, the policy can even be set to allow two vehicles in one block at a time, to avoid congestion. The second vehicle to enter the block is then also warned by a yellow signal.

When assuming that just one vehicle is allowed per time in a block, the block control system works as follows:

- when the vehicle enters a block, the traffic light turns to red, to indicate to the following vehicle to wait and keep the safety distance.
- when the vehicle leaves the first block, the traffic light of the first predecessor block turns to yellow. Yellow allows a vehicle to enter into the block, but with a slower speed, as there is another vehicle in the next block.
- when the vehicle enters in the third block, the traffic light of the second predecessor block turns to green, if there is no vehicle in it.

Figure 4 illustrates the block system. This safety mechanism was modeled by creating a Java class Block having as its main attributes: traffic light, red sensor and a pointer to the predecessorBlock. When the red sensor of a block is triggered (i) it changes the state of its traffic light to red, (ii) it changes the state of the traffic light of its predecessor block to yellow, (iii) it changes the state of the traffic light of the predecessor of its predecessor block to green, if the current state is yellow. This is shown in the situation I of Figure 4. If the state of the traffic light of the second predecessor block is not yellow, it can only be red, meaning that there is a vehicle in the first predecessor block, as shown in situation II of Figure 4.

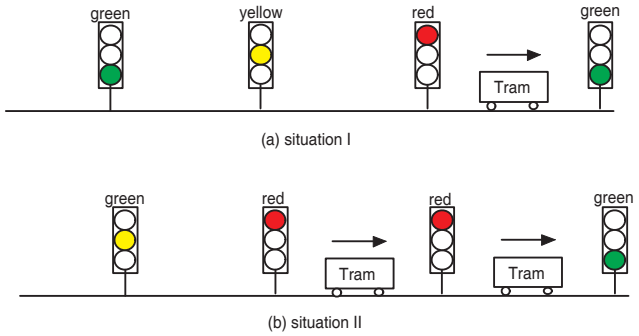

Figure 4: Block system.

In this way of modeling, a block changes the state of its own traffic light and of its neighbors when necessary. An important prerequisite is that predecessor and successor blocks must be set when instantiating or loading the model.

4 CASE STUDY

4.1 Case Description

In the city of The Hague in The Netherlands, a light rail viaduct in the Beatrixlaan is an important part of the Randstad Rail project, which consist of offering transport services using light rail vehicles between The Hague and Zoetermeer. Because of this initiative, the tram operator HTM is currently extending its rail infrastructure and consequently, the control system needs to be (re)designed.

The Beatrix Project focuses on a specific part of this infrastructure extension and it consists of studying a crossing of tracks that will be formed with the building of the new part of the infrastructure, which looks very much like the situation of Figure 3. The complexity is that tram stops (stations and platforms) are present that will influence the effectiveness of control measures (see Figure 5). The light rail vehicles travel between the south and the west, and share the tracks with the traditional trams that use the east-west tracks.

With the addition of tracks and light rail vehicles, control strategies for the surrounding area need to be redesigned as the traffic will be influenced by the new tram lines. These new tram lines will be operated by light rail vehicles and, compared to traditional trams they differ in length, width, acceleration and deceleration. Adjustments to the tram timetables, different speed limits, location of traffic lights and other alternatives of strategies need to be assessed. Infrastructure capacity will also vary according to the applied control strategies. For example, if speed limits are set to different values, the average of travel time will vary and consequently the throughput of vehicles will vary as well. Therefore, a careful analysis is necessary to decide which strategies will provide a satisfactory system performance.

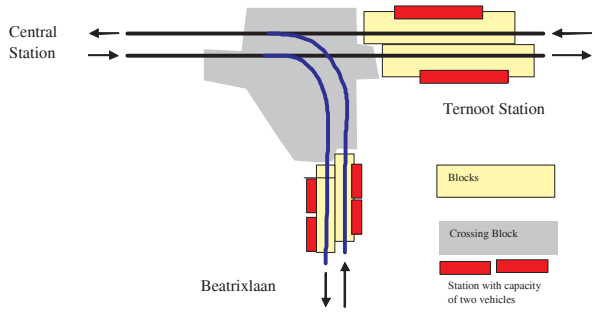

Figure 5: Beatrixlaan crossing.

4.2 Performance Indicators

In order to assess how the differences in timetables influence the system performance, we selected a number of performance indicators as a basis for the analysis. For all the experiments described later in this paper, the following performance indicators were used:

Travel Time: This is the total time a vehicle takes to drive through the system. Travel time is from the time the vehicle enters in the system until the time it leaves the system. If it is necessary that the vehicle waits until tracks are cleared to enter in the system (for safety reasons), this

Table 1: Generation for the expected scenario.

Vehicle line	Directions	Frequency
Line 7	Central Station - Ternoot	8 vehicles per hour (every 7.5 minutes)
Line 2	Central Station - Ternoot	12 vehicles per hour (every 5 minutes)
Line 7	Ternoot - Central Station	8 vehicles per hour (every 7.5 minutes)
Line 2	Ternoot - Central Station	12 vehicles per hour (every 5 minutes)
Line 3 (RR)	Central Station - Beatrixlaan	12 vehicles per hour (every 5 minutes)
Line 6 (RR)	Central Station - Beatrixlaan	12 vehicles per hour (every 5 minutes)
Line 3 (RR)	BeatrixLaan - Central Station	12 vehicles per hour (every 5 minutes)
Line 6 (RR)	BeatrixLaan - Central Station	12 vehicles per hour (every 5 minutes)
Total	88 vehicles per hour - No priority	

waiting time is not counted. Travel time is measured per line per direction.

$$Traveltime = EntryTime - DeletionTime \quad (1)$$

where EntryTime = time the vehicle enters in the system, and DeletionTime = time the vehicle is deleted from the system.

Speed Average: it is the speed average each vehicle drove through the system. It is calculated per line and per direction. It is calculated using the following formula:

$$Speedaverage = DistanceTraveled/TravelTime \quad (2)$$

where DistanceTraveled = distance traveled by the vehicle from the entry point until the deletion point, and TravelTime = see formula (1).

Queue Time: This is the time a vehicle waits to enter in the system until tracks are cleared. It is measured using the following formula:

$$Queuetime = EntryTime - GenerationTime \quad (3)$$

where EntryTime = time the vehicle enters in the system, GenerationTime = time the vehicle is created.

All the performance indicators in use in this project are measured per line per direction. Specific measurement objects have been created in the statistics layer of the library in Figure 1 to allow the measurement of these performance indicators.

4.3 Experiments

As we are analyzing a non-existing system, there are no real data to compare the results. For this reason, we set two scenarios that will serve as a reference for data analysis. We used one scenario where only light rail vehicles are generated and one where only traditional trams are driving through the system. In both scenarios, vehicles are generated periodically, but at very long time intervals. This is an ideal scenario, as apart from the dwell time at stations, no other factor will delay the trip. These scenarios provide the data of an ideal situation and the other scenarios will be analyzed making a comparison of how much they deviate from the ideal case. For simplification, we refer to light rail vehicles as RR vehicle, which means Randstad Rail. Many different experiments have been carried out. In this paper, we will show two scenarios with the expected number of vehicles only.

The scenario consists of running the system with the expected number of vehicles in the system. This means 88 vehicles in total, among Randstad Rail vehicles and traditional trams. The number of vehicles generated per directions is (names refer to the locations mentioned in Figure 5):

- Central Station - Ternoot: 20 vehicles per hour
- Ternoot - Central Station: 20 vehicles per hour
- Central Station - BeatrixLaan: 24 vehicles per hour
- BeatrixLaan - Central Station: 24 vehicles per hour

Table 1 shows the number of vehicles that is generated per line. In this scenario, no lines have priority in accessing the crossing. The access is given according to the order of request.

We carried out experiments without priority for any type of vehicle or line, and experiments where the RR vehicles for lines 3 and 6 have priority in accessing the crossing.

4.4 Analysis of Results

In this section, we present the results gathered from the scenarios defined earlier. The results from the first two reference experiments with only RR vehicles and only trams have been described in Tables 2 and 3.

Table 2: Averages for RR vehicles only.

Line	Travel Time	Speed	Queue Time
Line 3	157.532 s	8.031 m/s	0.000 s
CS-Beatrix	9.85 s	0.46 m/s	0.00 s
Line 3	160.903 s	7.801 m/s	0.000 s
Beatrix-CS	8.25 s	0.395 m/s	0.00 s

Table 3: Averages for tram vehicles only.

Line	Travel Time	Speed	Queue Time
Line 2	164.753 s	6.328 m/s	0.000 s
CS-Ternoot	8.727 s	0.536 m/s	0.000 s
Line 2	128.163 s	8.076 m/s	0.000 s
Ternoot-CS	12.156 s	0.4133 m/s	0.000 s

Comparing to ideal situation, the scenario with expected capacity and no priorities from which the results are displayed in Table 4 shows an increase in the average of travel time for the direction Ternoot-CS. This is explained by two factors: increase in the number of vehicles driving through the system makes the crossing to be busy with more frequency plus the fact that vehicles driving in this direction only request access to the crossing after they have left Ternoot station, which is at a short distance from the absolute traffic light guarding the crossing. Therefore, the request is made very late and when the crossing is occupied, vehicles have to wait for a longer time until tracks are cleared.

In addition, some vehicles of line 7 driving in the direction Ternoot-CS present an increase on the queue time. This is caused because vehicles have to wait longer to have access to the crossing and because vehicles keep a safety distance from each other. Therefore, new vehicles cannot enter in the system until tracks are cleared.

Although this scenario shows a small decrease on the performance, this delay can be avoided by adjusting timetables. The animation indicates that vehicles of all directions are generated at time intervals that make them arrive almost at the same time at the crossing while there are intervals without any vehicle in the system. Changing the timetables in order to make vehicles arrive at the crossing at different times can solve this conflict and can reduce the delay.

When comparing the results of the prioritized experiment in Table 5 to the experiment without priority in Table 4, it seems that giving priority to RR vehicles do not influence the system performance. This is expected, as the delay encountered in the non-prioritized experiment is caused by lines 2 and 7, which are operated by traditional trams. This result was not expected by the experts from HTM, who thought that priorities would make quite some difference.

4.5 Case Study Conclusions

In this case study we used a rail simulation architecture to support the measurement of a rail infrastructure capacity. Experiments showed that the current infrastructure can support 88 vehicles per hour distributed among 4 lines driving in both directions. The scenario discussed in this paper describes the system configuration that presented a good performance taking into account the performance indicators discussed earlier. Although results show that the infrastructure supports 88 vehicles per hour, adjustments to timetables

Table 4: Averages for realistic scenario, no priority.

Line	Travel Time	Speed	Queue Time
Line 2	136.063 s	7.607 m/s	0.000 s
CS-Ternoot	9.311 s	0.503 m/s	0.000 s
Line 7	151.472 s	6.954 m/s	0.000 s
CS-Ternoot	24.320 s	1.096 m/s	0.000 s
Line 2	136.063 s	7.607 m/s	0.000 s
Ternoot-CS	9.311 s	0.5032 m/s	0.000 s
Line 7	151.472 s	6.954 m/s	2.898 s
Ternoot-CS	9.311 s	0.441 m/s	0.000 s
Line 3	157.880 s	8.010 m/s	0.000 s
CS-Beatrix	9.006 s	0.441 m/s	0.000 s
Line 6	156.040 s	8.097 m/s	0.000 s
CS-Beatrix	7.018s	0.341 m/s	0.000 s
Line 3	163.480 s	7.674 m/s	0.000 s
Beatrix-CS	8.148 s	0.375 m/s	0.000 s
Line 6	167.032 s	7.499 m/s	0.000 s
Beatrix-CS	11.818s	0.238 m/s	0.000 s

are necessary. Based on statistical data and on the animation of the scenario (see Figure 6 for an example of the user output during the run), one could notice an overlapping on the timetables. At certain time intervals, vehicles coming from all directions have to compete to get access to the crossing as they reach the area almost at the same time, while there are time intervals where no vehicles are driving in the system. Distributing the vehicles departures better, the system performance can be improved by reducing the delay caused by the waiting time until the crossing is cleared.

Contrary to the expectations of the tram operator, giving priority to RR vehicles to the crossing does not increase the throughput of this type of vehicles as the average of travel time remains almost the same. Instead, it delays the traffic of traditional trams operating in other lines (lines 2 and 7). The implementation of other types of priorities could have a bigger impact on the traffic in the studied area, like for example, give priority to all vehicle lines departing from Central Station, but further tests need to be done in order to assess the impact of this strategy on the system operation.

5 CONCLUSIONS

Simulation showed to be very adequate to test alternatives of strategies to operate a rail infrastructure. Different system configurations can be tested and the performance of the system can be analyzed through animation and through the statistics gathered during the simulation.

The rail simulation library applied in this project performed well, but improvements can be made. For example, calculate automatically some performance indicators, such as average of travel time, instead of only providing data for later calculation. The addition of other performance indicators to complement the output analysis would be desirable.

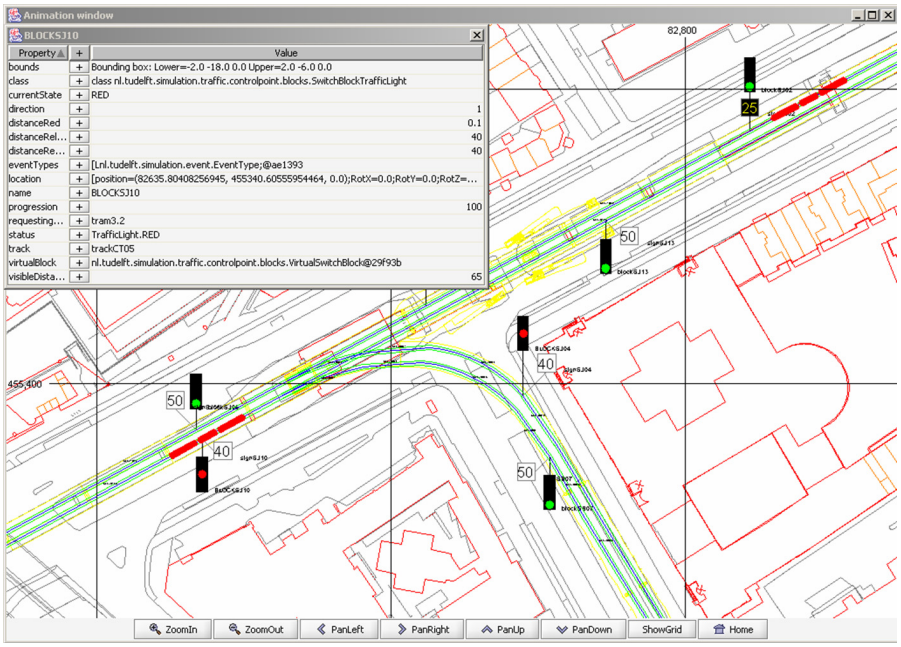

Figure 6: Screen-dump of the library in action for the case study example.

Table 5: Realistic scenario, priority for RR vehicles.

Line	Travel Time	Speed	Queue Time
Line 2	136.078 s	7.612 m/s	0.000 s
CS-Ternoot	10.197 s	0.550 m/s	0.000 s
Line 7	150.492 s	6.999 m/s	0.000 s
CS-Ternoot	23.479 s	1.141 m/s	0.000 s
Line 2	136.078 s	7.612 m/s	0.000 s
Ternoot-CS	10.20 s	0.550 m/s	0.00 s
Line 7	150.492 s	6.999 m/s	2.899s
Ternoot- CS	23.479 s	1.287 m/s	3.202 s
Line 3	158.392 s	7.981 m/s	0.000 s
CS-Beatrix	9.3162 s	0.4536 m/s	0.000 s
Line 6	156.698 s	8.059 m/s	0.000 s
CS-Beatrix	7.723 s	0.377 m/s	0.000 s
Line 3	164.882 s	7.611 m/s	0.000 s
Beatrix-CS	8.330 s	0.372 m/s	0.000 s
Line 6	167.200 s	7.493 m/s	0.000 s
Beatrix-CS	6.141 s	0.259 m/s	0.000 s

ACKNOWLEDGMENTS

We acknowledge the support of the HTM Personenvervoer NV for this research <www.htm.net>. This research project has been funded by the BSIK-NGI program <www.nginfra.nl>.

REFERENCES

Bergmark, R. 1996. Railroad capacity and traffic analysis using simon. In *Computers in Railways V*, ed. R. J. H. G. S. J. Allan, C. A. Brebbia and S. Sone, 549–562. Southampton, United Kingdom: WIT Press.

Black, A. 1995. *Urban mass transportation planning*. McGraw-Hill.

Ceder, A. 2003. *Advanced modeling for transit operations and service planning*, Chapter Public Transport Timetabling and Vehicle Scheduling, 31–57. Elsevier Science.

Demitz, J., C. Hübschen, and C. Albrecht. 2004. Timetable stability - using simulation to ensure quality in a regular interval timetable. In *Computers in Railways IX*, ed. J. Allan, C. A. Brebbia, R. J. Hill, G. Sciutto, and S. Sone, 549–562. Southampton, United Kingdom: WIT Press.

Edwards, J. 1992. *Transportation planning handbook*. Englewood Cliffs: Prentice-Hall.

Harris, N. G., and E. W. Godward. 1992. *Planning passenger railways: A handbook*. Transport Publishing Co Ltd.

Higgings, A., L. Ferreira, and E. Kozan. 1995. Modelling single line train operations. *Transportation Research Record 1484, In Railroad Transportation Research*:9–16. Washington DC.

Jacobs, P. H. M. 2005. *The DSOL simulation suite - enabling multi-formalism simulation in a distributed context*. Ph.D. thesis, Delft University of Technology, Delft, The Netherlands.

Jacobs, P. H. M., N. A. Lang, and A. Verbraeck. 2002. D-SOL; a distributed java based discrete event simulation architecture. In *Proceedings of the 2002 Winter Simulation Conference*, ed. E. Ycesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 793–800. Piscataway,

New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Kaas, A. H. 2000. Punctuality model for railways. In *Computers in Railways VII*, ed. C. A. B. G. S. J. Allan, R. J. Hill and S. Sone, 809–816. Southampton, United Kingdom: Wit Press.

Kanacilo, E. M., and A. Verbraeck. 2006. Simulation services to support the control design of rail infrastructures. In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1372–1379. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Kraft, E. R. 1982. Jam capacity of single track rail lines. In *Transportation Research Forum Proceedings*, Volume 23, 461–471.

Lang, N. A., P. H. M. Jacobs, and A. Verbraeck. 2003. Distributed open simulation model development with DSOL services. In *Proceedings of the 15th European Simulation Symposium 2003 - Simulation in Industry*, 210–218. Germany: SCS European Publishing House.

Middelkoop, D., and M. Bouwman. 2000. Train network simulator for support of network wide planning of infrastructure and timetables. In *Computers in Railways VII*, ed. J. Allan, R. J. Hill, C.A.Brebbia, G. Sciutto, and S. Sone, 267–276. Southhampton, UK: WIT Press.

Middelkoop, D., and M. Bouwman. 2001. Simone: Large scale train network simulations. In *Winter Simulation Conference*, ed. B. Peters and J. Smiths, 1042–1047. Psicataway, New Jersey, EUA: IEEE Press.

Papazoglou, M. P. 2003. Service oriented computing: concepts, characteristics and directions. In *Proceedings of the 4th IEEE International Conference on Web Information Systems*, 3–12: IEEE Computer Society.

Rudolph, R., and J. Demitz. 2003. Simulation of large railway networks. In *World Congress on Railway Research 2003*, 644–652. Edinburgh, United Kingdom.

Sussman, J. 2000. *Introduction to transportation systems*. Norwood,MA.: Artec House, Inc.

Vuchic, V. R. 2004. *Urban transit: Operations, planning and economics*. John Wiley & Sons, Inc.

Wahlborg, M. 1996. Simulation models: Important aids for baverkert's planning process. In *Computers in Railways V*, ed. R. J. H. J. Allan, C. A. Brebbia, Volume 1, 175–181. Southampton, United Kingdom: Wit Press.

Wiransinghe, S. C. 2003. *Advanced modeling for transit operations and service planning*, Chapter Initial Planning for Urban Transit Systems, 1–29. Oxford, UK: Pergamon.

Zeigler, B. P., H. Praehofer, and T. G. Kim. 2000. *Theory of modeling and simulation*. Second Edition ed. Academic Press.

AUTHOR BIOGRAPHIES

ELISANGELA MIEKO KANACILO is a Ph.D. candidate at the Systems Engineering Group of Delft University of Technology. Her PhD research is focused on developing a simulation environment to support the control system design of rail infrastructures. Her email address is <e.m.kanacilo@tudelft.nl> and her web page is <www.tbm.tudelft.nl/webstaf/elisangelak>.

ALEXANDER VERBRAECK is chair of the Systems Engineering Group of Delft University of Technology, and a part-time full professor in supply chain management at the R.H. Smith School of Business of the University of Maryland. He is a specialist in discrete event simulation for real-time control of complex transportation systems and for modeling business systems. His current research focus is on development of generic libraries of object oriented simulation building blocks in C++ and Java. His e-mail address is <a.verbraeck@tudelft.nl>, and his web page is <www.tbm.tudelft.nl/webstaf/alexandv>.