Simulating the Oslo Subway by hierarchic, coloured, object-oriented, timed Petri Nets with viewpoints

Thor Kristoffersen, Anders Moen, Hallstein Asheim Hansen

NWPT 2002

Norwegian Computing Center 2002-20-11

Abstract

Abstract

In this paper we discuss some difficulties in modelling and extracting relevant information from large concurrent systems. We present the concept of 'viewpoint', how one can observe high level behaviour from low level behaviour of Petri Nets. The fruitfulness of viewpoints in simulating and monitoring real railway systems is demonstrated, by an experimental Petri Net tool, called Andromeda, supporting hierarchical, coloured, object-oriented, timed Petri Nets, with viewpoints. The Oslo Subway is modelled as a large modular coloured Petri Net, using all Andromeda's features. To test our approach, large scale simulations of the subway traffic have been performed.

What Petri Nets contributes to Logistics and Workflow

The theory has been demonstrated useful in

- simulating
- visualising
- modelling
- executing
- monitoring
- analyzing

workflow and logistic processes.

Challenges with large scale concurrent systems

- observation of the structure and behaviour
- construction of large systems using components

Solutions to these problems are respectively

- **viewpoints**, abstraction mechanisms on both structure and behavior
- **object-oriented PetriNets**, providing tools to define standard classes, e.g., general stations and special stations that inherit behaviour from the general station.

The subway model

Oslo Subway 2002:

- 5 lines
- 101 stations
- trains run at 15 minutes intervals
- 69,8 million travels per year
- 10624 persons travelling per hour on average
- 118,7 kilometers of railway.

The system is chosen as a benchmark for investigating two mechanisms of abstraction, **object orientation** and **views**.

Implementation

Implementations split into two:

- Andromeda Petri Net engine written in Common Lisp
- Railway model programmed by object-oriented Petri Nets

The model of Oslo Subway runs on top of Andromeda, containing;

- a **low level engine**, performing basic Petri Net transactions,
- the **class system**, providing means to define Petri Nets by net classes, providing both inheritance and encapsulation,
- the **class editor**, that offers visual editing of Petri Net classes
- the **engine visualizer**, that allows visual views of an engine at run-time.

Our Petri Net model of Oslo Subway contains 2926 places, 2117 transitions and 6971 arcs. **Petri Net classes**: Stations, platform and segments **Tokens**: passengers and trains

Andromeda in action

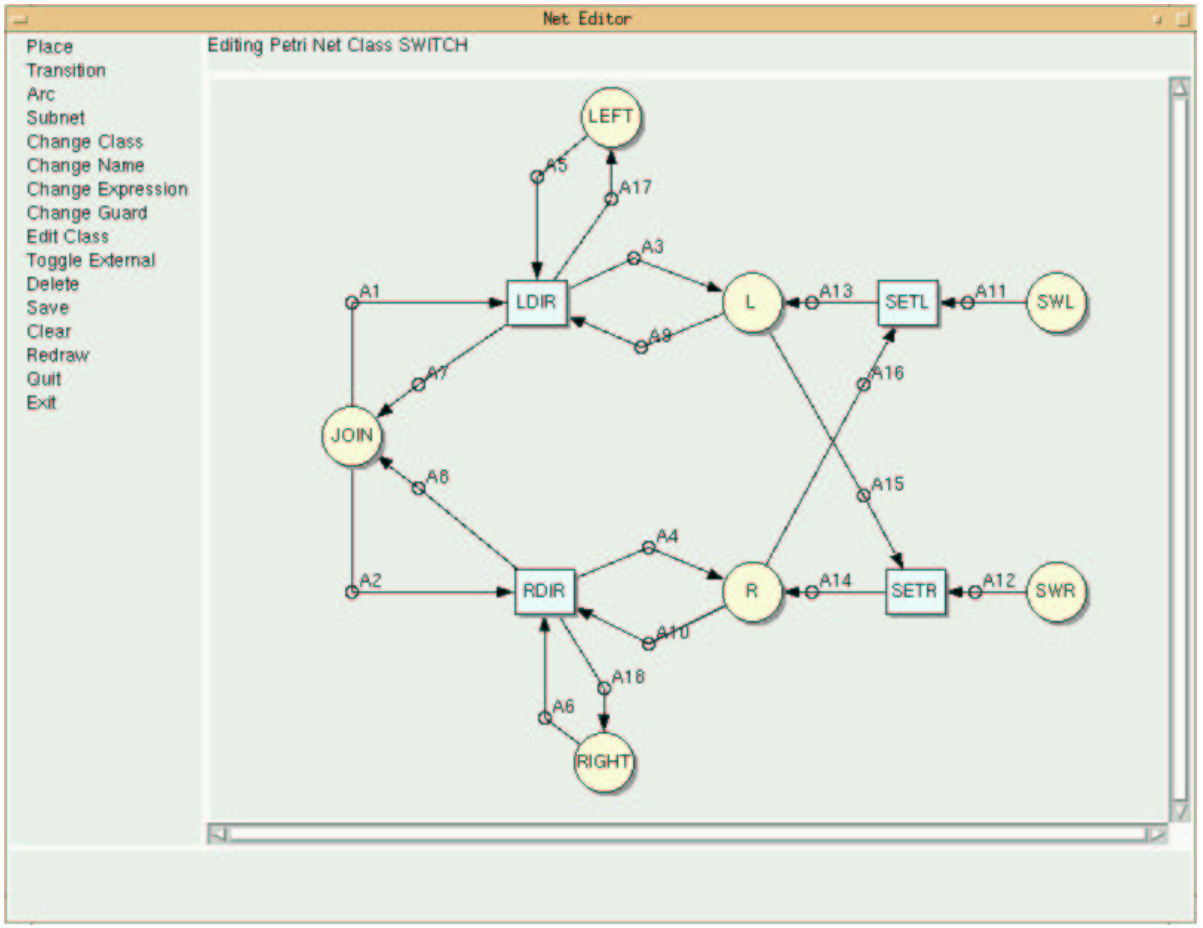

Figure 1: The net editor of Andromeda in use 3

Railway components

A **Raiload Net** is composed of

- railroad segments
- switches
- end segments
- platforms.

A **Railway System** contains

- railroad net
- trains moving in the net
- passengers travelling in the trains
- signalling system
- power supply.

The trains

The basic entity, the **train**, is represented as a composite coloured token, with four slots,

- **train number**
- **main direction** deciding the main direction the train moves, e.g. on Sognsvannsbanen direction east
- **local direction** refers to the current state of the train, moving either forwards, backwards or in stop state, and finally a set of
- **passengers** inside the train.

In Figure 2 we show a basic railway segment consisting of three segment states s_1, s_2 and s_3, representing three physical points on the railroad, and **move** transitions t_1, t_2, t_3 and t_4, moving the train forwards or backwards: 2:

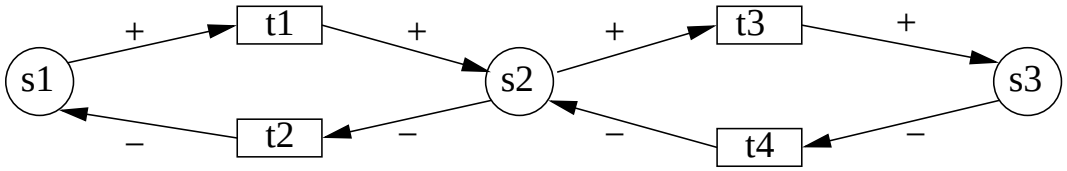

Figure 2:

Switch components consist of three segment places, **join**, **left** and **right**, semaphore places L and R, controlling the routing of the tokens over the switch, see Figure 3.

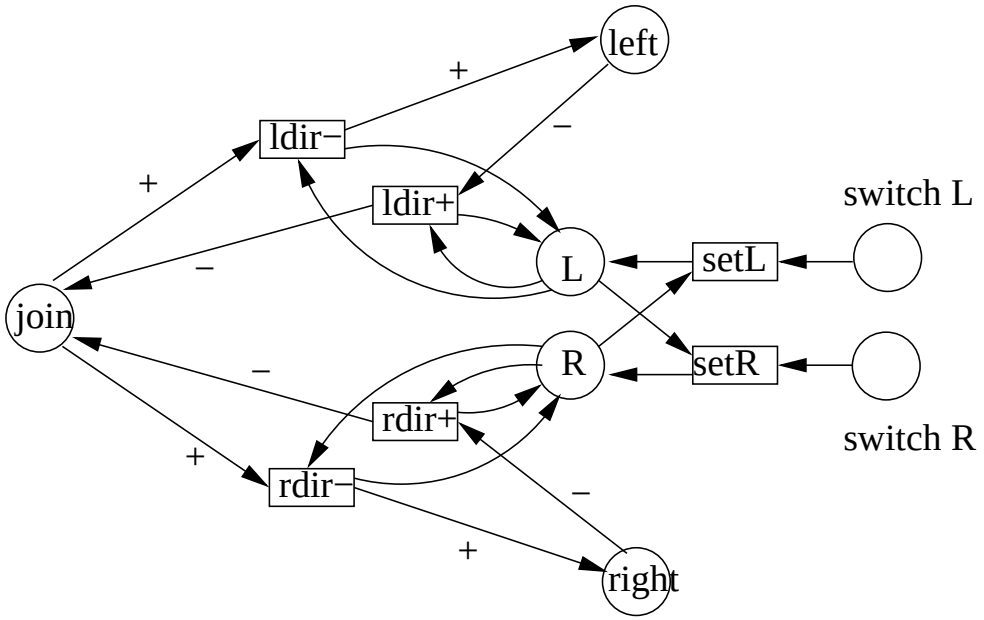

Figure 3: The switch component.

Initial marking: mutex pair (L, R) carries exact one token

+ and − denote the way routing of trains are performed

Signalling system

A signaling system is extracting information from the segment states. A signal light is modelled as the mutex net: Signal lights red, yellow and green meaning respectively full stop, drive slowly and drive

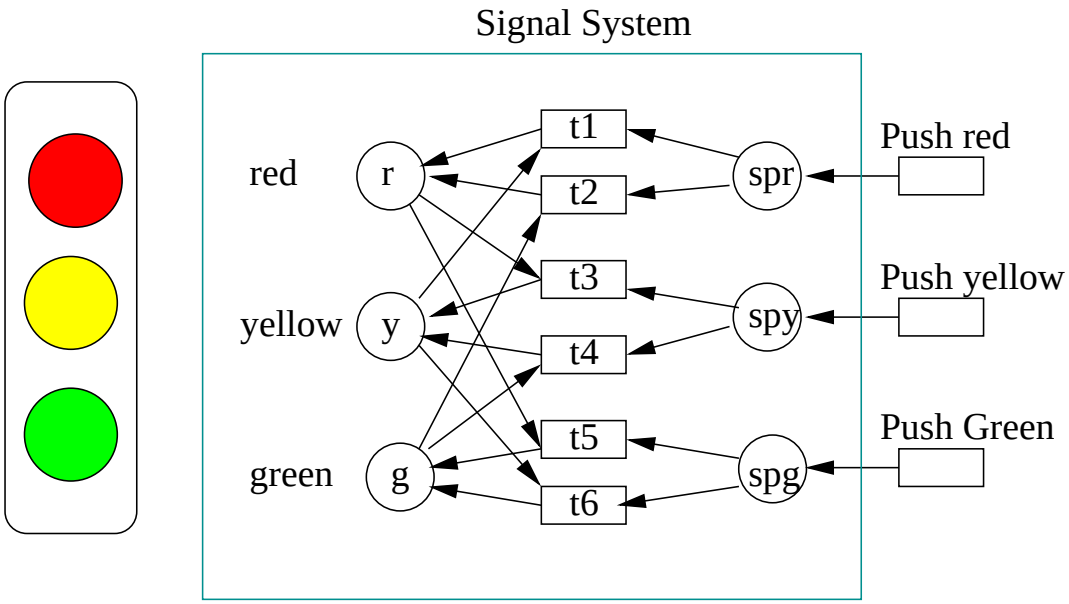

Figure 4:

(Intermediate detailed model for a station contains over 3000 transitions from segment places to signal nets)

Communication

At runtime, a monitoring and controlling system sends requests to an external application or receiving messages from the application enabling transitions to fire:

- "message out" tokens
- "message in" tokens

Definition 1 *The connection with the external environment takes place through the input/output interfaces, the set of I/O places.*

Definition 2 *We say that a Petri Net system is* **open** *if it is connected to an external environment.*

The control of trains, their positions and the handling of switches can be done by putting tokens into the I/O places.

Time

Travelling by train takes time. Unforeseen events happen, and cause extraordinary delays. In a realistic model of train systems, both standard and stocastic delays must be accounted for.

We have implemented timed Petri Net as specified in Jensen [1997].

token @ + delay

taking account or both standard and stocastic delays
In our model delays are specified:

- in the railroad segments between stations
- at the stations, slowing down speed by arrival and departure
- the time taken to get passengers onboard and out of the trains

Viewpoints

One problem of simulating and monitoring railway systems, is to understand their behaviour.

Example of viewpoints: The railway leader central

- equipped with monitoring screens
- alarms detect equipment and possible collisions, that provides relevant and security critical information to the train leaders.

Problem: Relevant information should be provided so that train leaders could

- monitor train position
- discover trains that are too late
- alarms for detect malfunction of equipment and possible collisions, etc.

Viewpoints continued

- To be precise about behaviour is nice, but it does not necessarily entail that we understand a complex system (too large!)
- But since low level behaviour is represented we can **extract** high level behaviour, by mappings of abstraction
- Introduce the concept of **viewpoint**, to denote these mappings.
- A viewpoint has two aspects, **visual** and **functional**
- The visual aspect of a viewpoint is the graphical representation of the viewpoint.
- The basic entities in a viewpoint are the **observable places**, the places carrying information we need to monitor.
- The functional aspect of a view is the functional algebra over the observable places. The functional view is constructed from viewpoint functions listening to the marking of observable places.

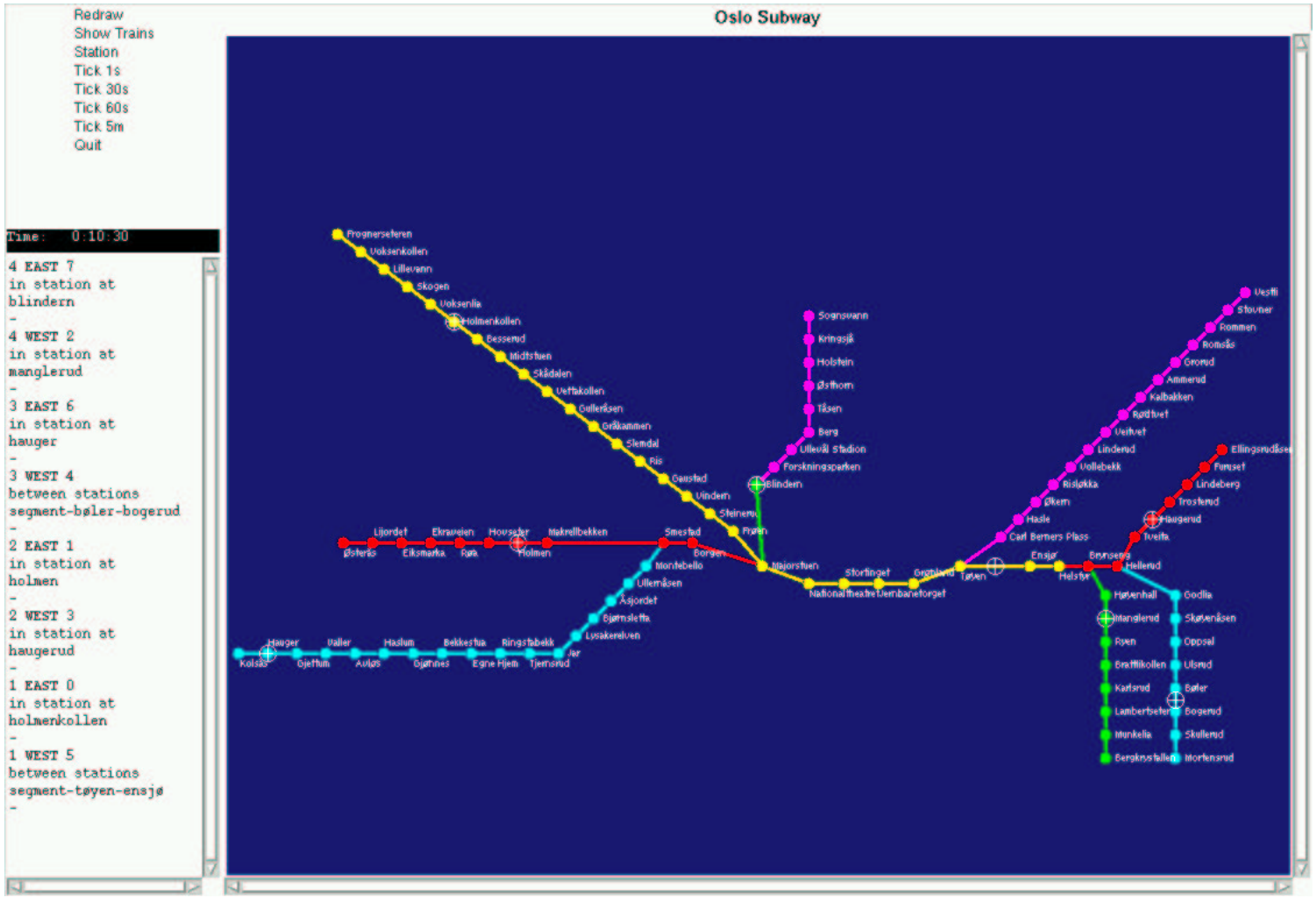

Figure 5: The visualizer for Oslo Subway, train-in-line viewpoint.

Related Work

- Railway industry: closed industry, difficult to find standards for terminology
- Not much work on Petri Net models for railway systems, except Hielscher et al. [1998] Hielscher and Urbszat [1997] (and Nisse Husberg's group)
- We have built PetriNet system from scratch: Why not choose Design/CPN as core technology?
  - wanted ownership to our code
  - wanted to use our tool to experiment with concepts
- In Jensen [2002] various techniques for obtaining information from a simulation is discussed. E.g. 'business chart'.
- The work of van der Aalst et. al, in the papers Basten and van der Aalst [1999] and Basten and van der Aalst [2001], but limited to the structure of the hierarchies.

Experiences

- A prototype Andromeda+Libraries has been implemented, on which we can do simulations and some simple performance analysis
- implemented Viewpoint on top of the Net, showing the actual subway net with trains moving
- Object-orientation turned out to be useful while constructing railway components, and the inheritance mechanism applied to nets speeded up the time to model railway components, since we used variants of Petri Nets patterns several times by subclassing
- To build an error free signalling system is difficult: Experimented with specifying the behaviour of the signal light at a station in a minimal logical language $\longrightarrow$ generating Petri Net code from the spesification $\longrightarrow$ connect this (Petri Net) code to the actual railroad segments.

Further work

- Formalize the concept of viewpoint
- Provide Railroad components providing local collision detection
- Give definition of a Rail Way System in terms of Petri Nets
- Use this definition to define properties on the net, e.g. collision signalling behaviour
- Use real I/O as input, save histories of real behaviour, to simulate the behaviour with the external environment, realistic simulations.
- Extend the perspective of viewpoint to filtration of execution