Petri Nets

Djura Smits

August 22, 2011

1 Introduction

Petri nets is a mathematical modeling language developed for describing distributed systems. We are considering replacing the finite state machines used in the situations framework [1]. Naturally, finite state automata are not directly convertible to petri nets, so there are are a couple of issues that need to be addressed.
There are several aspects that we would like to have in a description of behavior:

- *Time*
- *Non-determinism*
- *Modularity*

  Is this the right word?
- *Availability*
-

2 Petri Nets

A Petri net is a mathematical modeling language used for the description of distributed systems. A petri net is a bipartite graph consisting of two types of nodes: places and transitions. These nodes are connected by directed arcs. An arc can run from either a place to a transition, or from a transition node to a place, but never from a place to a place, or between two transitions.

Activity in a Petri net is expressed by by the movement of tokens from place to place, through transitions. Input arcs (from place to transition) denote which places need to contain tokens in order to enable the transition. When a transition is enabled, it consumes the tokens from the input places, and produces tokens in the place indicated by the output arc.

3 Advantages of Petri Nets over Finite State Automata

Petri nets hold several advantages over finite state automata. First of all, Petri nets allow for concurrent behavior. This would enable our pedestrians to do several tasks at once. For example, it could be possible to model our pedestrians' Petri nets so that a token would represent an arm or a leg. That way, a pedestrian could execute multiple basic tasks independently. It would not be possible to achieve this kind of behavior with a finite state machine, unless a state would be described for every possible combination of activities. However, this is not the only advantage Petri nets have over finite state automata. Petri nets come with a standardized way to incorporate a time aspect. For finite state automata, several techniques have been developed, but there is no general consensus over what methods are most suitable. The problem becomes even larger when we would like to incorporate both time and non-determinism. Both the aspects of time and non-determinism are easily incorporated in Petri nets with only small modifications, and can even be mixed with traditional Petri nets.

References

[1] Mankyu Sung, Michael Gleicher, and Stephen Chenney. Scalable behaviors for crowd simulation, 2004.