Dynamic object-oriented geospatial modeling

Tomáš Richta, Martin Hrubý
Department of Intelligent Systems
Faculty of Information Technology
Brno University of Technology
xricht16@stud.fit.vutbr.cz, hrubym@fit.vut.cz

Keywords: dynamic, object-oriented, modeling, geospatial, methodology, DEVS, GIS, database

Abstract

Published literature about moving objects (MO) simplifies the problem to the representation and storage of moving points, moving lines, or moving regions. The main insufficiency of this approach is lack of MO inner structure and dynamics modeling – the autonomy of moving agent. This paper describes basics of the object-oriented geospatial methodology for modeling complex systems consisting of agents, which move within spatial environment. The main idea is that during the agent movement, different kinds of connections with other moving or stationary objects are established or disposed, based on some spatial constraint satisfaction or nonfulfilment respectively. The methodology is constructed with regard to following two main conditions – 1) the inner behavior of agents should be represented by any formalism, e.g. Petri net, finite state machine, etc., and 2) the spatial characteristic of environment should be supplied by any information system, that is able to store defined set of spatial types, and support defined set of spatial operations. Finally, the methodology is demonstrated on simple simulation model of tram transportation system.

Motivation

This paper outlines the methodology for modeling objects moving within the spatial environment, while establishing and disposing connections with other moving or stationary objects. Movement of objects is constrained by the environment, and the connection between two objects (moving, or stationary one) is established when some predefined spatial constraint is satisfied, and disposed when the constraint is no longer fulfilled. For example, moving agents could be vehicles, that move within the city. When each vehicle reaches some distance range to the previous one, it must slower its movement, or even stop to avoid the collision. There are also many cameras mounted on street lamps, that monitor the movement of vehicles. This could be modeled as a system where objects move within some predefined path, and

establishes or disposes connections with the other ones, which is triggered by the achievement of some specific distance to it. There also exist connections between stationary objects, and moving ones, triggered by the presence of the moving ones within the visibility range of those stationary.

The idea of this paper is to define the methodology for such a system modeling. In this motivation section this idea is discussed in more detail. In the next related work section, the other important ideas, that are important for the methodology are briefly introduced. In the contribution section the main contribution of this paper is described, and the methodology outlined. In the last section the specific problem is solved using presented methodology.

Spatially embedded networks

The important observation concerning the moving object positions, is that the movement of objects is usually not free, but constrained by some predefined paths (e.g. roads, streets, rails, pavements, passages, stairs, doors, etc.) that connects some places (crossings, stations, buildings, squares, etc.). Both paths and places form some kind of spatially embedded network. This observation allows to abstract form the classical notion of spatial position, defined as a function $P : \Omega \to R^n$, where Ω is universe of objects, and n is the dimension of the space. Position specification becomes even more complicated, concerning the geospatial position, which includes the problem of projecting the Earth surface to some regular grid.

The idea of abstracting the environment to the form of spatially embedded network was first introduced by Wolfson et al. [1], discussed by Jensen [2], and formalized by Guting [3]. Spatial network is modeled as a graph $G = (V, E)$, with set of vertexes V and set of edges $E \subseteq V \times V$. The set of possible positions of moving object within such a graph G is then defined as [3]

$$Pos(G) = V \cup (E \times (0, 1)).$$

It means that the agent is positioned either on some vertex $v \in V$ or on some relative position on the edge defined as tuple (e, x), where $e \in E, x \in (0, 1)$. Position of objects then becomes studied from the topological point of view. But if there is a mapping from V to set of point features, and from E to set of linear features, it is possible to compute the precise geospatial position of objects, using some geographic information system (GIS).

Dynamic geospatial networking

The main idea of this paper is to define the principles and rules of dynamic connection establishing and disposal among moving, and also stationary objects, based on some spatial constraint satisfaction or nonfulfilment – *dynamic geospatial networking*. Following example of networking assumes simple spatial constraint to be the visibility of objects. Based on that, connections are established when objects see each other, and disposed, when they do not. Schema of the example is shown in Fig. 1, where three moving objects (e.g. vehicles) are following predefined path, being observed by two watchtowers A and B. When each vehicle reaches the visibility range of watchtower, the connection between the vehicle and stationary object is established. There are also connections between moving vehicles, that see each other, and also between two watchtowers for the same reason.

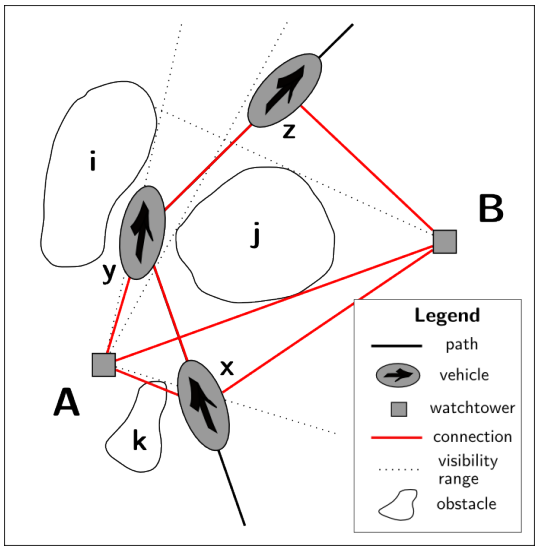

Figure 1: Geospatial networking example

The scene could be interpreted as following. The vehicle x is in the range of watchtower B and right now enters the range of watchtower A, because it gets beyond the obstacle k. It also sees the vehicle y, but does not see the vehicle z, because of the obstacle j. The vehicle y sees both other vehicles, and is monitored by the watchtower A, and no the B, because of j. Finally the vehicle z is monitored by the watchtower B, and sees the vehicle y.

Related work

Geographic information systems

Geographic information system (GIS) is system capable of storing and retrieval of spatially characterized data. The study of GIS cover many disciplines, like spatial data capturing, geospatial analysis, cartographic visualization, etc. For the purposes of discussed methodology, the GIS should provide the spatial data back end for geospatial analysis computations – visibility, signal propagation, etc. From this point of view the following functionality of GIS is important:

- storage of basic spatial data types (points, lines, polygons)
- basic spatial operations (length, distance, ...)
- advanced analytical operations (visibility, signal propagation, ...)
- well defined interface or language for requirements (queries) formulation

Regarding the coupling of different types of models, some ideas concerning interaction between GIS and other models exist. For example Brandmeyer and Karmi isolated five levels of environmental models coupling [4]. Those levels could be thought of as levels of maturity of GIS and other systems integration. The described methodology uses GIS as outer resource of information and defines strict interface for simulation model and GIS communication. This

approach is similar to *loose coupling* level defined in [4]. It means that all necessary spatial queries are immediately sent through the interface to GIS, and resulting data is returned back. There is no integration between those two models, so any GIS satisfying defined criteria could be used.

Moving objects databases

Moving objects represent research area concerned on the computational representation of spatially or geospatially embedded objects, that change their position over time. Moving objects cover broad research area, including popular terms like Global Positioning Systems (GPS), Location Based Services (LBS), and many others. For the purposes of this paper the problem is reduced to a moving objects databases, more precisely to moving objects in networks databases. This research area covers the spatio-temporal modeling and algebras [5], [6], [3], implementation issues of the physical data models [2], efficient querying [7], [8], and indexing techniques [9], [10], [11], and database benchmarking [12].

Guting et al. described the algebra of spatio-temporal predicates [5], [6], including the type system formal specification, definition of spatial and spatio-temporal predicates with temporal aggregation, to the querying the developments in STSQL – spatio-temporal extension of structured query lanuage (SQL). This work laid the basics of spatio-temporal databases, and further was extended by Guting, Almeida, and Ding, to cover the area of moving objects in spatial networks [3].

Discrete event system specification

Research of moving objects databases reduces the moving object to be either moving point, moving line, or moving region. The main idea of this paper is to enhance the model with moving objects autonomous behavior. The important formalism that encapsulates the inner behavior of agents, provides the means for its temporality, and defines clear interface for its incoming and outgoing communication is called Discrete EVent system Specification (DEVS), and was introduced by Ziegler in [13], [14]. A DEVS basic component, also called atomic, is defined as a structure [13]

$$DEVS = \langle X, Y, S, \delta, \lambda, ta \rangle$$

where

X is the set of external input events,

Y is the set of output events,

S is the set of sequential (partial) states,

$\lambda : S \to Y$ is the output function,

$ta : S \to R^{+}_{0,\infty}$ is the time advance function,

δ is a transition function consisting of two parts,

1) the internal transition function

$$\delta_{int} : S \to S,$$

2) and the external transition function

$$\delta_{ext} : Q \times X \to S,$$

where Q is the total state of $DEVS$ given by

$$Q = \{(s, e) | s \in S \wedge 0 \leq e \leq ta(s)\}.$$

Concepcion and Ziegler further extended the model to provide for hierarchical modeling by the possibility to hierarchically combine atomic DEVSes [15], [14]. Coupled DEVS is a structure that consists of a number of other coupled or atomic DEVSes, and formally it is defined as

$$CDEVS = \langle X, Y, D, \{M_d\}, C_{xx}, C_{yx}, C_{yy}, Select \rangle$$

where

X is the set of input events,

Y is the set of output events,

D is set of component references, where

$$\forall d \in D : M_d \in \{DEVS, CDEVS\},$$

$C_{xx} \subseteq X \times \bigcup_{i \in D} X_i$ is the set of external input couplings,

$C_{yx} \subseteq \bigcup_{i \in D} Y_i \times \bigcup_{i \in D} X_i$ is set of internal couplings,

$C_{yy} : \bigcup_{i \in D} Y_i \to Y^{\phi}$ is external output coupling function,

$Select : 2^D \to D$ is the tie-breaking function, that solves the problem of simlutaneous events.

Structure of the input and output sets of DEVS and CDEVS could also be refined to provide the notion of incoming and outgoing ports for events as following [14]

$X = \{(p, v) | p \in InPorts, v \in X_p\}$, where

$InPorts$ is the set of input ports, and $X_p \subseteq X$

$Y = \{(p, v) | p \in OutPorts, v \in Y_p\}$, where

$OutPorts$ is the set of output ports, and $Y_p \subseteq Y$.

Another important extension to DEVS formalism had been done by Barros [16], who extended the definition of DEVS formalism to support changes in structure. Barros defined the DSDN – *dynamic structure DEVS network* as [16], [14]

$$DSDN_{\Delta} = \langle X_{\Delta}, Y_{\Delta}, \chi, M_{\chi} \rangle$$

where

Δ is the DSDN name,

X_{Δ} is the input event set of DSDN,

Y_{Δ} is the output event set of DSDN,

χ is DSDN *network executive* name,

M_χ is model of χ, which is defined as

$$M_\chi = \langle X_\chi, S_\chi, Y_\chi, \delta_{int_\chi}, \delta_{ext_\chi}, \lambda_\chi, ta_\chi \rangle$$

with sequential state $s_\chi \in S_\chi$ defined as

$$s_\chi = \langle D^\chi, \{M_i^\chi\}, \{I_i^\chi\}, \{Z_{i,j}^\chi\}, Select^\chi, V^\chi \rangle$$

where

D^χ is the set of components of network, where $\chi \notin D^\chi$,

$M_i^\chi = \langle X_i, S_i, Y_i, \delta_{int_i}, \delta_{ext_i}, \lambda_i, ta_i \rangle$ is model of component i, for all $i \in D^\chi$,

I_i^χ is the set of influencers of i, for all $i \in D^\chi \cup \{\chi, \Delta\}$, where

$i \notin I_i^\chi$ for all $i \in D \cup \{\chi, \Delta\}$,

$Z_{i,j}^\chi$ is i-to-j translation function, for all $j \in I_i$, and

$Z_{\Delta,j}^\chi : X_\Delta \to X_j$,

$Z_{i,\Delta}^\chi : Y_i \to Y_\Delta$,

$Z_{i,j}^\chi : Y_i \to X_j$,

$Z_{k,\chi}^\chi(y) \neq \phi \Rightarrow Z_{k,j}^\chi(y) = \phi$, for

$k \in D^\chi \cup \{\Delta\}$, and for all

$j \in I_k^\chi - -\{\chi\}$, with

ϕ as null event,

$Select^\chi : \Pi \to D^\chi \cup \{\chi\}$, is select function, where

$\Pi = 2^{D^\chi \cup \{\chi\}} - -\{\}$,

$Select^\chi(A) \in A$,

V^χ are other state variables not defined before.

s_χ is called *structure*, and any change in described variables is called *change in structure*. Filippi and Bisambiglia published a DSDN modification that should provide for modeling spatially embedded network of modified atomic components called *Point-DEVS*, that change their position within a spatial environment [17]. This approach is similar to Celular-DEVS, and is used to simulate fire propagation in natural environment. Atomic DEVS reduced to have the partial state defined as tuple (V_d, E), where V_d is displacement vector, and E is local space property of Point-DEVS. This work is similar to the presented methodology, but not usable as solution, because it defines only the position to be the partial state of DEVS.

Contribution

Moving objects DEVS modification

For the purpose of presented methodology, a simple extension of atomic DEVS component called NDEVS – *Networking-DEVS*, which is defined as structure

$$NDEVS = \langle X, Y, S, \delta, \lambda, ta, \gamma \rangle$$

where

S, δ, λ, ta remains the same as in atomic DEVS,

$X = \{(p, v) | p \in P_{in}, v \in X_p\}$ is the set of input ports and values,

$Y = \{(p, v) | p \in P_{out}, v \in Y_p\}$ is the set of output ports and values,

$\gamma : P_{in} \cup P_{out} \to L \cup \epsilon$ is annotation function, where

L is language for networking rules definition, and

ϵ is empty word.

The extension of coupled component is similar to the atomic one, so the following structure defines NCDEVS – *NetworkingCoupled-DEVS*

$$NCDEVS = \langle X, Y, D, \{M_d\}, C_{xx}, C_{yx}, C_{yy}, Select, \gamma \rangle$$

where

$D, M_d, C_{xx}, C_{yx}, C_{yy}$, and $Select$ are the same as in CDEVS,

$\forall d \in D : M_d \in \{NDEVS, NCDEVS\}$,

$X = \{(p, v) | p \in P_{in}, v \in X_p\}$ is the set of input ports and values,

$Y = \{(p, v) | p \in P_{out}, v \in Y_p\}$ is the set of output ports and values,

$\gamma : P_{in} \cup P_{out} \to L \cup \epsilon$ is annotation function, where

L is language for networking rules definition, and

ϵ is empty word.

Dynamic changes of the structure of network, could be modeled as DSDN, with following modifications

$$M_\chi = \langle X_\chi, S_\chi, Y_\chi, \delta_{int_\chi}, \delta_{ext_\chi}, \lambda_\chi, ta_\chi, \psi_\chi \rangle$$

where

$s_\chi \in S_\chi = \langle D^\chi, \{M_i^\chi\}, \{I_i^\chi\}, \{Z_{i,j}^\chi\}, Select^\chi, V^\chi \rangle$,

$\forall i \in D^\chi : M_i^\chi \in \{NDEVS, NCDEVS\}$, and

ψ_χ is evaluation function defined as

$$\psi_\chi : \bigcup_{i,j \in D^\chi, i \neq j} S_i \times P_i \times L \times S_j \times P_j \times L \to bool$$

where

$\forall x \in D^\chi : P_x = P_{in_x} \cup P_{out_x}$,

L is language for networking rules definition.

During the model evolution, right after each internal transition function δ_{int_χ} execution, the system evaluates the function ψ_χ on all components $i, j \in D^\chi$, where $i \neq j$. This modifies the results given by $Z_{i,j}$ function of s_χ to $Z'_{i,j}$, based on the evaluation of annotations of ports $p_x \in P_x, x \in D^\chi$ given by the function $\gamma_x, x \in D^\chi$, within the context of present state $s_x \in S_x, x \in D^\chi$ of each component. The modification is given by following rule

$$\psi_\chi(s_i, p_i, \gamma_i(p_i), s_j, p_j, \gamma_j(p_j)) \Rightarrow (y_i, x_j) \in \{Z'_{i,j}\}$$

It is necessary to state, that the presented modifications of DEVS does not extend its modeling capabilities, which could be formally proved, but this proof is not part of this paper.

Dynamic geospatial modeling methodology

The main idea of developed methodology is to define basic rules for transforming from the real-world situation to the dynamic geospatial networking system. Because the proper definition of whole methodology is too complex, only the simple outline is introduced. The transformation should consist of following steps:

1. each object of the system is described as NDEVS, or NCDEVS d, using following rules
   (a) the set of states is defined as $S_d = S_{bas} \times P \times R^+_{0,\infty}$, where S_{bas} is the set of basic states of the object, and $W = v_0 e_1 v_1 e_2 v_2 ... e_k v_k, e_i = (v_{i-1}, v_i), 1 \leq i \leq k$ is the path the object is following
   (b) each port of d is annotated with string that specifies the the set of rules R_p of port p_d, defining its ability to connect to the other ports within the system, the rule $r \in R_p = (C_s, C_o)$, where C_s is the set of spatial constraints (e.g. visibility, distance less than, etc.), and C_o is the set of other constraints (e.g. name of the port, etc.)
   (c) the object containing multiple NDEVS, is coupled into NCDEVS, it is necessary to specify the rules on NCDEVS according to the rules on ports, that are connected to them
2. all objects are then supplied to the network executive χ, that solves the problem of dynamic networking by the following sequence
   (a) if the state of any of the NDEVS $d \in D$ changes
   (b) $\forall p_d \in P_{in_d} \cup P_{out_d}, \forall p_x \in \bigcup_{x \in D^\chi} P_{in_x} \cup P_{out_x}$ the ψ_χ function is evaluated
   (c) based on the result of ψ_χ the $Z_{i,j}$ function of χ is modified

Let's consider the tram transportation system as a case study of described methodology. Model of such a system consists of rail network and moving trams. Rail network consists of places (platforms and switches), and transitions (rail segments). Each platform is connected with two rail segments – the one by that it is achievable, and the other by which it could be leaved. Each platform is considered to be a part of some station object. Rail segments connect two places, and each switch has one incoming rail segment and two outgoing rail segments. Spatial environment is then constructed as a network of transitions and places.

Trams are moving agents, that change their position within this network in time, and make decisions at the switches based on the assigned travel schedule. Switches and platforms are stationary objects, that mediate the environment connections. Each place is mapped to the point feature, and each transition is mapped to line feature within some GIS. Let's consider following real-world situation – in Prague, there is station Malostranská, the scheme of this station is shown in Fig. 2.

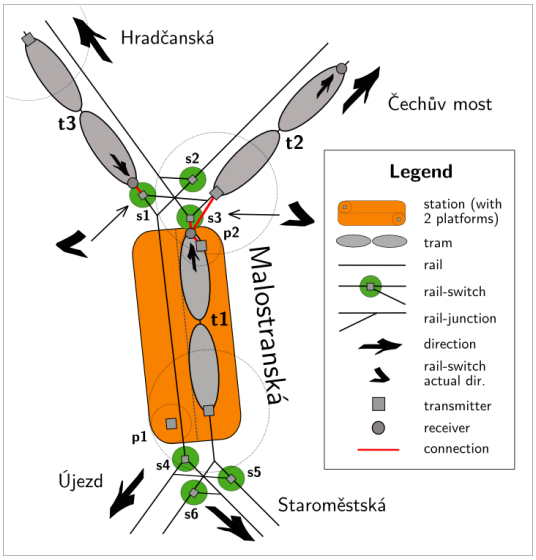

Figure 2: Malostranská station

This schema could be interpreted as following – tram $t1$ stopped on platform $p2$, and the the switch $s3$ indicates, that $t1$ is heading to Čechův most. Tram $t2$ is also heading to Čechův most, and at the moment it blocks $t1$'s movement. Tram $t3$ is coming from Hradčanská, and the switch $s1$ indicates, that it can go further to the $p1$ on Malostranská.

More formally – tram is a vehicle, that moves along the rails within predefined path $W = v_0e_1v_1e_2v_2...e_kv_k, e_i = (v_{i-1}, v_i), 1 \leq i \leq k$, containing sequence of vertexes $v_i \in V$ and edges $e_i \in E$. Actual heading of the tram is given by the next element of P, and actual position as $p \in Pos(G)$, defined previously. Tram is modeled as NDEVS

$$Tram = \langle X, Y, S, \delta, \lambda, ta, \gamma \rangle$$

with following values

$$X = \{(?go, true), (?stop, true)\}$$

$$Y = \{(!next, p \in Pos(G)), (!position, p \in Pos(G))\}$$

$$S = \{(d, \sigma) | d \in W \times \{stopped, running\}, \sigma \in \mathbb{R}^{+}_{0,\infty}\}, s_0 = ((v_0, stopped), \infty)$$

$$\forall s \in S : ta(s) = \sigma$$

$$\delta_{ext}((((v_i, stopped), \sigma), t_e), go) = ((v_i, running), \sigma)$$

$$\delta_{ext}((((v_i, stopped), \sigma), t_e), stop) = ((v_i, stopped), \sigma - -t_e)$$

$$\delta_{ext}((((v_i, running), \sigma), t_e), go) = ((v_i, running), \sigma - -t_e)$$

$\delta_{ext}((((v_i, running), \sigma), t_e), stop) = ((v_i, stopped), \sigma)$

$\delta_{ext}(((((e_i, x), running), \sigma), t_e), go) = (((e_i, x), running), \sigma - -t_e)$

$\delta_{ext}(((((e_i, x), running), \sigma), t_e), stop) = (((e_i, x), stopped), \sigma)$

$\delta_{ext}(((((e_i, x), stopped), \sigma), t_e), go) = (((e_i, x), running), \sigma)$

$\delta_{ext}(((((e_i, x), stopped), \sigma), t_e), stop) = (((e_i, x), stopped), \sigma - -t_e)$

$\delta_{int}((v_i, running), \sigma) = (((e_{i+1}, 0), running), \sigma)$

$\delta_{int}(((e_i, 0 < x < 1), running), \sigma) = (((e_i, x + vt_e), running), \sigma)$

$\delta_{int}(((e_i, x \geq 1), running), \sigma) = ((v_{i+1}, stopped), \sigma)$

$\delta_{int}((v_i, stopped), \sigma) = ((v_i, running), \sigma)$

$\lambda((v_i, stopped), \sigma) = (!position, v_i)$

$\lambda((v_i, stopped), \sigma) = (!next, (e_{i+1}, 0))$

$\lambda((v_i, running), \sigma) = (!position, v_i)$

$\lambda((v_i, running), \sigma) = (!next, (e_{i+1}, 0))$

$\lambda(((e_i, x), stopped, \sigma) = (!position, (e_i, x))$

$\lambda(((e_i, x), stopped, \sigma) = (!next, (v_{i+1}))$

$\lambda(((e_i, x), running, \sigma) = (!position, (e_i, x))$

$\lambda(((e_i, x), running, \sigma) = (!next, (v_{i+1}))$

$\gamma(?stop) =$

```
'select p from allOutPorts where p.name = 'stop''
```

$\gamma(?go) =$

```
'select p from allOutPorts where p.name ='go''
```

$\gamma(!position) =$

```
'select p from allInPorts where p.name = 'position''
```

$\gamma(!next) =$

```
'select p from allInPorts where p.name = 'next''
```

Let's assume that the *Tram* has two sensors (transmitter/receiver), that are able to broadcast and receive information to and from other sensors in the system. The transmitter is on the back of tram, and the receiver in the front of the vehicle. Tram could then indicate a proximity of the other by receiving the signal from broadcaster, resulting in tram stop to avoid the collision. Transmitter could be modeled as following NDEVS component

$$Transmitter = \langle X, Y, S, \delta, \lambda, ta, \gamma \rangle$$

noindent with following values

$X = \{\}$

$Y = \{(!signal, bool)\}$

$S = \{(d, \sigma) | d \in \{v\} \times \{nosignal, signal\}, \sigma \in \mathbb{R}^{+}_{0,\infty}\}, s_0 = (nosignal, 0)$

$\lambda((v, nosignal), \sigma) = (v, signal)$

$\lambda((v, signal), \sigma) = (v, nosignal)$

$\gamma(!signal) =$

```
'select p from allInPorts  where p.name = 'signal' and p.distance(v) < delta'
```

And receiver as following NDEVS component

$$Receiver = \langle X, Y, S, \delta, \lambda, ta, \gamma \rangle$$

with following values

$X = \{(?signal, true)\}$

$Y = \{(!stop, true), (!go, true)\}$

$S = \{(d, \sigma) | d \in \{v\} \times \{waiting, receiving\}, \sigma \in \mathbb{R}^{+}_{0,\infty}\}, s_0 = (waiting, \infty)$

$\delta_{ext}((((v, waiting), \sigma), t_e), signal) = ((v, receiving), \sigma)$

$\delta_{ext}((((v, receiving), \sigma), t_e), signal) = ((v, receiving), \sigma - -t_e)$

$\delta_{ext}((((v, receiving), \sigma), t_e), \phi) = ((v, waiting), \infty)$

$\lambda((v, receiving), \sigma) = (!stop, true)$

$\lambda((v, waiting), \sigma) = (!go, true)$

$\gamma(?signal) =$

```
'select p from allOutPorts where p.name = 'signal' and p.distance(v) < delta'
```

$\gamma(!stop) =$

```
'select p from allInPorts where p.name = 'stop''
```

$\gamma(!go) =$

```
'select p from allInPorts where p.name = 'go''
```

Tram with sensors is then following NCDEVS model

$$TramWithSensors = \langle X, Y, D, \{M_i\}, C_{xx}, C_{yx}, C_{yy}, Select, \gamma \rangle$$

with following values

$X = \{(?signal, true), (?go, true)\},$

$Y = \{(!next, p \in Pos(G)), (!position, p \in Pos(G))\},$

$D = \{tram, transmitter, receiver\},$

$M_{tram} = Tram, M_{transmitter} = Transmitter, M_{receiver} = Receiver$

$C_{xx} = \{(?signal, receiver.?signal), (?go, tram.?go), ()\}$

$C_{yx} = \{(receiver.!go, tram.!go), (receiver.!stop, tram.!stop)\}$

$C_{yy}(transmitter.!signal) = !signal$

$C_{yy}(tram.!position) = !position$

$C_{yy}(tram.!next) = !next$

$\gamma(?signal) =$

```
'select p from allOutPorts where p.name = 'signal' and p.distance(v) < delta'
```

$\gamma(?go) =$

```
'select p from allOutPorts where p.name = 'go' and p.distance(v) < delta'
```

$\gamma(!position) =$

```
'select p from allInPorts where p.name = 'position' and p.distance(v) < delta'
```

$\gamma(!next) =$

```
'select p from allInPorts where p.name = 'next' and p.distance(v) < delta'
```

It could be seen, that γ functions of NCDEVS were defined according to connected NDEVS components. Schematic diagram of described NDEVS and NCDEVS models, and two more, are shown in Fig.3., where the dynamically established and disposed connections are drawn using dashed arrows.

Future work

Spatio-temporal constraint language

For the formalization of ports rules definitions, it is necessary to define the semantics of the spatio-temporal constraint language L, that is used for rules definition. So far the SQL-like notation was used, which probably remains the same, but it is necessary to precisely define the predicates, that might be used within the rules definition.

Agent dimensions

One of the other problems is the size of moving agents. In presented case study, this problem could be easily solved using the space manager, which asks the GIS database for the information about the ordering of vehicles (based on their location and direction). But in future, the size of agents could also play another important roles, so it might be included to the model.

Model persistence

The problem of the model persistence includes the storage of agents inner states, actual connections, and other values, including the history of network evolution. Here it is possible to build on the Moving objects databases fundamentals also briefly introduced in this paper. The area could also introduce the problem of query optimizations, that have not been addressed yet.

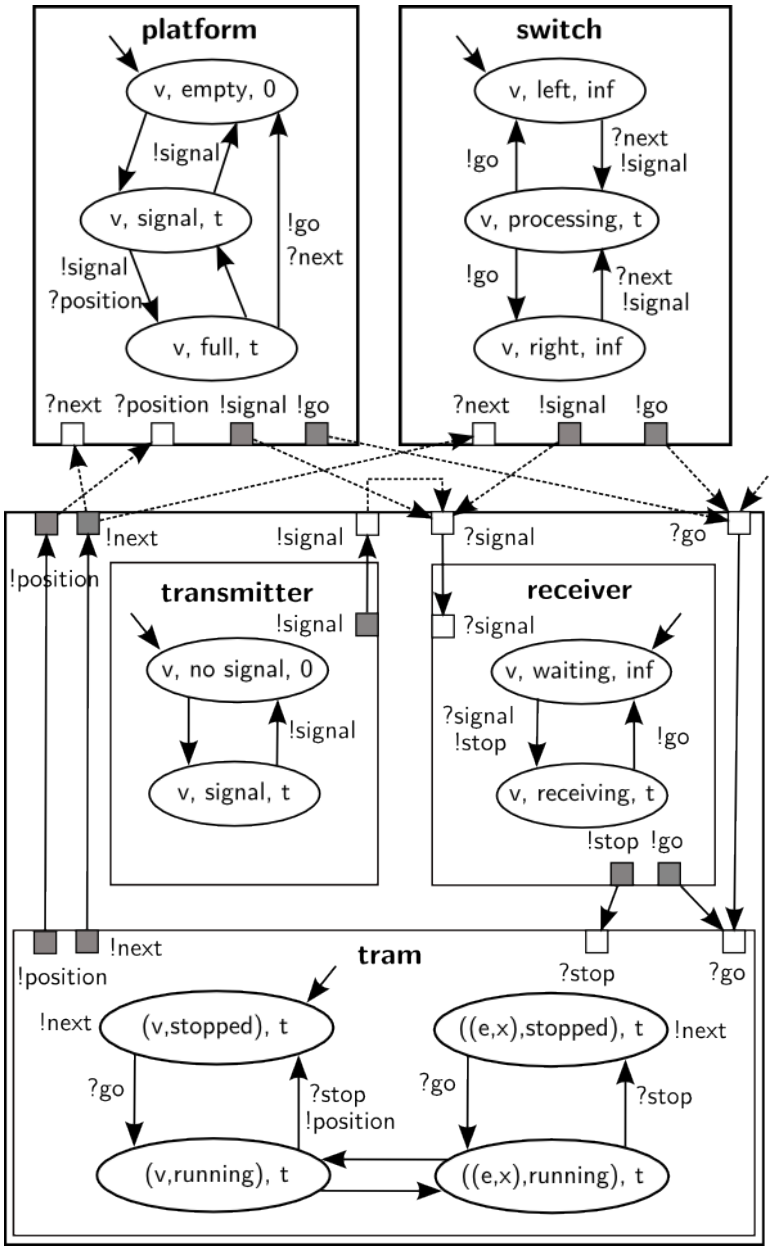

Figure 3: NDEVS and NCDEVS models

Conclusions

This paper introduced the outline of the dynamic geospatial networking methodology, which is based on the DEVS formalism extension, called *Networking-DEVS*, and *NetworkingCoupled-DEVS*. Introduced extensions does not improve the modeling power of classical DEVS formalism, but only defines the different type of its usage. The methodology covers basic rules for the transformation from real-world moving objects, into Networking-DEVS models. It also defines the rules for dynamic connecting of DEVS models, based on some geospatial constraint satisfaction, which is called *geospatial networking*. Connections are established as DEVS ports couplings, based on the rules defined in ports annotations. The dynamic be-

havior of the model is formalized by modified DSDN structure, with *network executive*. The methodology assumes that the network executive is connected with some GIS, that it uses for geospatial analytical computations. Based on the results of those analyzes the connections are established, and disposed. Presented methodology was demonstrated on test case concerning the tram traffic model.

References

1. Ouri Wolfson, Bo Xu, Sam Chamberlain, and Liqin Jiang. Moving objects databases: Issues and solutions. In SSDBM, pages 111-122, 1998.
2. Christian S. Jensen, Torben Bach Pedersen, Laurynas Speicys, and Igor Timko. Data modeling for mobile services in the real world. In SSTD, pages 1-9, 2003.
3. Ralf Hartmut Guting, Victor Teixeira de Almeida, and Zhiming Ding. Modeling and querying moving objects in networks. The VLDB Journal, 15(2):165-190, 2006.
4. Jo Ellen Brandmeyer and Hassan A. Karimi. Coupling methodologies for environmental models. Environmental Modelling and Software, 15(5):479-488, 2000.
5. Ralf Hartmut Guting, Michael H. Boehlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, and Michalis Vazirgiannis. A foundation for representing and querying moving objects. ACM Trans. Database Syst., 25(1):1-42, 2000.
6. Ralf Hartmut Guting and Markus Schneider. Moving Objects Databases. Morgan Kaufmann, 2005.
7. Christian S. Jensen, Jan Kolář, Torben Bach Pedersen, and Igor Timko. Nearest neighbor queries in road networks. In GIS, pages 1-8, 2003.
8. Dimitris Papadias, Jun Zhang, Nikos Mamoulis, and Yufei Tao. Query processing in spatial network databases. In VLDB, pages 802-813, 2003.
9. Su Chen, Christian S. Jensen, and Dan Lin. A benchmark for evaluating moving object indexes. PVLDB, 1(2):1574-1585, 2008.
10. Xiang Li and Hui Lin 0002. Indexing network-constrained trajectories for connectivity-based queries. International Journal of Geographical Information Science, 20(3):303-328, 2006.
11. Talel Abdessalem, Cédric du Mouza, José Moreira, and Philippe Rigaux.Management of large moving objects datasets: Indexing, benchmarking and uncertainty in movement representation. In Yannis Manolopoulos, Apostolos Papadopoulos, and Michael Vassilakopoulos, editors, Spatial Databases, pages 225-249. Idea Group, 2005.
12. Thomas Brinkhoff. A framework for generating network-based moving objects. GeoInformatica, 6(2):153-180, 2002.
13. Bernard P. Zeigler. Multifacetted modelling and discrete event simulation. Academic Press Professional, Inc., San Diego, CA, USA, 1984.

14. Bernard P. Zeigler, Herbert Praehofer, and Tag G. Kim. Theory of Modeling and Simulation, Second Edition. Academic Press, 2 edition, January 2000.

15. Arturo I. Concepcion and Bernard P. Zeigler. Devs formalism: A framework for hierarchical model development. IEEE Trans. Software Eng., 14(2):228-241, 1988.

16. Fernando J. Barros. Dynamic structure discrete event system specification: A new formalism for dynamic structure modeling and simulation. In Winter Simulation Conference, pages 781-785, 1995.

17. Jean-Baptiste Filippi and Paul Bisgambiglia. General methodology 2: enabling large scale and high definition simulation of natural systems with vector models and jdevs. In Winter Simulation Conference, pages 1964-1970, 2002.

Acknowledgement: This work has been supported by the Czech Ministry of Education under the Research Plan No. MSM0021630528 *"Security-Oriented Research in Information Technology"*.