# Transportation
# A Domain Description

**Dines Bjørner**
**DTU Compute, Techn. Univ. of Denmark**
bjorner@gmail.com, www.imm.dtu.dk/~dibj

**January 19, 2013: 20:21**

1

**Abstract**

We give a description of fragments of the transportation domain. We assume familiarity with [1], a base paper for understanding techniques of domain description. 2

# Contents

# 1 Introduction 3

## 1.1 The Problem 4

The problem to be addressed is that of understanding: *What is transportation ?*. What do we mean by *understanding* a particular domain, such as here the, or a transport domain. We shall mean that there is a description of that domain which meets the following criteria:

the description must be accepted by a number of domain stake-holders; and it must be possible to reason about properties of the described domain. 5

Since the domain description conceptually covers also major aspects of railroad nets, shipping nets, and air traffic nets, we shall use such terms as hubs and links to stand for road (or street) intersection and road (or street) segments, train stations and rail lines, harbours and shipping lanes, and airports and air lanes.

### 1.2 Domain Modelling 6

### 1.3 Structure of Paper 7

## 2 Endurants 8

### 2.1 Parts

#### 2.1.1 Root Sorts

The root domain, Δ, the stepwise unfolding of whose description is to be exemplified, is that of a composite traffic system (1a.) with a road net, (1b.) with a fleet of vehicles and (1c.) of whose individual position on the road net we can speak, that is, monitor. 9

1. We analyse the composite traffic system into
   a a composite road net,
   b a composite fleet (of vehicles), and
   c an atomic monitor.

**type**
1. Δ
1a. N
1b. F
1c. M
**value**
1a. <u>**obs**_</u>N: Δ → N
1b. <u>**obs**_</u>F: Δ → F
1c. <u>**obs**_</u>M: Δ → M

#### 2.1.2 Sub-domain Sorts and Types 10

2. From the road net we can observe
   a a composite part, HS, of road (i.e., street) intersections (hubs) and
   b a composite part, LS, of road (i.e., street) segments (links).

**type**
2. HS, LS
**value**
2a. **obs_**HS: N → HS
2b. **obs_**LS: N → LS

11 We analyse the sub-domains of HS and LS.

3. From the hubs aggregate we decide to observe
   a the concrete type of a set of hubs,
   b where hubs are considered atomic; and
4. from the links aggregate we decide to observe
   a the concrete type of a set of links,
   b where links are considered atomic;

12

**type**
3a. Hs = H-**set**
4a. Ls = L-**set**
3b. H
4b. L
**value**
3. **obs_**Hs: HS → H-**set**
4. **obs_**Ls: LS → L-**set**

13

5. From the fleet sub-domain, F, we observe a composite part, VS, of vehicles

**type**
5. VS
**value**
5. **obs_**VS: F → VS

14

6. From the composite sub-domain VS we observe
   a the composite part Vs, which we concretise as a set of vehicles
   b where vehicles, V, are considered atomic.

**type**
6a. Vs = V-**set**
6b. V
**value**
6a. **obs_**Vs: VS → V-**set**

15 The "monitor" is considered atomic. It is an abstraction of the fact that we can speak of the positions of each and every vehicle on the net without assuming that we can indeed pin point these positions by means of, for example, sensors.

## 2.2 Properties 16

Parts are distinguished by their properties: the types and the values of these. We consider three kinds of properties: unique identifiers, mereology and attributes.

### 2.2.1 Unique Identifications 17

There is, for any traffic system, exactly one composite aggregation, HS, of hubs, exactly one composite aggregation, Hs, of hubs, exactly one composite aggregation, LS, of links, exactly one composite aggregation, Ls, of links, exactly one composite aggregation, VS, of vehicles and exactly one composite aggregation, Vs, of vehicles, Therefore we shall not need to associate unique identifiers with any of these.

7. We decide the following:
   - a each hub has a unique hub identifier,
   - b each link has a unique link identifier and
   - c each vehicle has a unique vehicle identifier.

**type**
7a. HI
7b. LI
7c. VI
**value**
7a. **uid_**H: H $\rightarrow$ HI
7b. **uid_**L: L $\rightarrow$ LI
7c. **uid_**V: V $\rightarrow$ VI

### 2.2.2 Mereology 18

**[1] Road Net Mereology** By *mereology* we mean the study, knowledge and practice of understanding parts and part relations.

The mereology of the composite parts of the road net, n:N, is simple: there is one HS part of n:N; there is one Hs part of the only HS part of n:N; there is one LS part of n:N; and there is one Ls part of the only LS part of n:N. Therefore we shall not associate any special mereology based on unique identifiers which we therefore also decided to not express for these composite parts. 19

8. Each link is connected to exactly two hubs, that is,

   a from each link we can observe its mereology, that is, the identities of these two distinct hubs,

   b and these hubs must be of the net of the link;

9. and each hub is connected to zero, one or more links, that is,

   a from each hub we can observe its mereology, that is, the identities of these links,

   b and these links must be of the net of the hub.

20

```
value
8a.  mereo_L: L → Hl-set
axiom
8a.  ∀ l:L•card mereo_L(l)=2,
8b.  ∀ n:N,l:L,hi:Hl •
8b.     l ∈ obs_Ls(obs_LS(n)) ∧ hi ∈ mereo_L(l)
8b.       ⇒ ∃ h:H•h ∈ obs_Hs(obs_HS(n))∧uid_H(h)=hi
value
9a.  mereo_H: H → Ll-set
axiom
9b.  ∀ n:N,h:H,li:Ll •
9b.     h ∈ obs_Hs(obs_HS(n)) ∧ li ∈ mereo_H(h)
9b.       ⇒ ∃ l:L•l ∈ obs_Ls(obs_LS(n))∧uid_L(l)=li
```

21

**[2] Fleet of Vehicles Mereology** In the traffic system that we are building up there are no relations to be expressed between vehicles, only between vehicles and the (single and only) monitor. Thus there is no mereology needed for vehicles.

### 2.2.3 Attributes 22

23 We shall model attributes of links, hubs and vehicles. The composite parts, aggregations of hubs, HS and Hs, aggregations of links, LS and Ls and aggregations of vehicles, VS and Vs, also have attributes, but we shall omit modelling them here.

**[1] Attributes of Links**

10. The following are attributes of links.

   a Link states, l$\sigma$:L$\Sigma$, which we model as possibly empty sets of pairs of distinct identifiers of the connected hubs. A link state expresses the directions that are open to traffic across a link.

b Link state spaces, $l\omega$:L$\Omega$ which we model as the set of link states. A link state space expresses the states that a link may attain across time.

c Further link attributes are length, location, etcetera.

Link states are usually dynamic attributes whereas link state spaces, link length and link location (usually some curvature rendition) are considered static attributes. 24

**type**
10a. $L\Sigma = (HI \times HI)$**-set**
**axiom**
10a. $\forall\ l\sigma{:}L\Sigma \bullet 0 \leq$ **card** $l\sigma \leq 2$
**value**
10a. **attr_**$L\Sigma$: $L \to L\Sigma$
**axiom**
10a. $\forall$ l:L • **let** $\{hi,hi'\}$=**mereo_**L(l) **in** **attr_**$L\Sigma(l) \subseteq \{(hi,hi'),(hi',hi)\}$ **end**
**type**
10b. $L\Omega = L\Sigma$**-set**
**value**
10b. **attr_**$L\Omega$: $L \to L\Omega$
**axiom**
10b. $\forall$ l:L • **let** $\{hi,hi'\}$=**mereo_**L(l) **in** **attr_**$L\Sigma(l) \in$ **attr_**$L\Omega(l)$ **end**
**type**
10c. LOC, LEN, ...
**value**
10c. **attr_**LOC: L → LOC, **attr_**LEN: L → LEN, ...

25

### [2] Attributes of Hubs

11. The following are attributes of hubs:

a Hub states, $h\sigma$:H$\Sigma$, which we model as possibly empty sets of pairs of identifiers of the connected links. A hub state expresses the directions that are open to traffic across a hub.

b Hub state spaces, $h\omega$:H$\Omega$ which we model as the set of hub states. A hub state space expresses the states that a hub may attain across time.

c Further hub attributes are location, etcetera.

Hub states are usually dynamic attributes whereas hub state spaces and hub location are considered static attributes. 26

**type**
11a. $H\Sigma = (LI \times LI)$**-set**

**value**
11a. **attr_**HΣ: H → HΣ
**axiom**
11a. ∀ h:H • **attr_**HΣ(h)⊆{(li,li′)|li,li′:LI•{li,li′}⊆**mereo_**H(h)}
**type**
11b. HΩ = HΣ-**set**
**value**
11b. **attr_**HΩ: H → HΩ
**axiom**
11b. ∀ h:H • **attr_**HΣ(h) ∈ **attr_**HΩ(h)
**type**
11c. LOC, ...
**value**
11c. **attr_**LOC: L → LOC, ...

27

## [3] Attributes of Vehicles

12. Dynamic attributes of vehicles include
    a position
        i. at a hub (about to enter the hub — referred to by the link it is coming from, the hub it is at and the link it is going to, all referred to by their unique identifiers or
        ii. some fraction "down" a link (moving in the direction from a from hub to a to hub — referred to by their unique identifiers)
        iii. where we model fraction as a real between 0 and 1 included.
    b velocity, acceleration, etcetera.
13. All these vehicle attributes can be observed.

28

**type**
12a. VP = atH | onL
12(a)i. atH :: fli:LI × hi:HI × tli:LI
12(a)ii. onL :: fhi:HI × li:LI × frac:FRAC × thi:HI
12(a)iii. FRAC = **Real**, **axiom** ∀ frac:FRAC • 0 ≤ frac ≤ 1
12b. VEL, ACC, ...
**value**
13. **attr_**VP:V→VP,
13. **attr_**onL:V→onL,
13. **attr_**atH:V→atH
13. **attr_**VEL:V→VEL,

```
13.      attr_ACC:V→ACC
13.      ...
```

29

## [4] Vehicle Positions

14. Given a net, n:N, we can define the possibly infinite set of potential vehicle positions on that net, vps(n).
    a vps(n) is expressed in terms of the links and hubs of the net.
    b vps(n) is the
    c union of two sets:
       i. the potentially[1] infinite set of "on link" positions
       ii. for all links of the net

       and

       iii. the finite set of "at hub" positions
       iv. for all hubs in the net.

30

```
value
14.      vps: N → VP-infset
14b.     vps(n) ≡
14a.       let ls=obs_Ls(obs_LS(n)), hs=obs_Hs(obs_HS(n)) in
14(c)i.      { onL(fhi,uid(l),f,thi) | fhi,thi:HI,l:L,f:FRAC •
14(c)ii.           l ∈ ls ∧ {fhi,thi}=mereo_L(l) }
14c.         ∪
14(c)iii.    { atH(fli,uid_H(h),tli) | fli,tli:LI,h:H •
14(c)iv.           h ∈ hs ∧ {fli,tli}⊆mereo_H(h) }
14a.       end
```

31

**[5] Vehicle Assignments** Given a net and a finite set of vehicles we can distribute these vehicles over the net, i.e., assign initial vehicle positions, so that no two vehicles "occupy" the same position, i.e., are "crashed"! Let us call the non-deterministic assignment function vpr.

15. vpm:VPM is a bijective map from vehicle identifiers to (distinct) vehicle positions.

[1]The 'potentiality' arises from the nature of FRAC. If fractions are chosen as, for example, 1/5'th, 2/5'th, ..., 4/5'th, then there are only a finite number of "on link" vehicle positions. If instead fraction are arbitrary infinitesimal quantities, then there are infinitely many such.

16. vpr has the obvious signature.
    a vpr(vs)(n) is defined in terms of
    b a non-deterministic selection, vpa, of vehicle positions, and
    c a non-deterministic assignment of these vehicle positions to vehicle identifiers,
    d being the resulting distribution.

32

**type**
15. $\mathsf{VPM}' = \mathsf{VI} \underset{m}{\rightarrow} \mathsf{VP}$
15. $\mathsf{VPM} = \{|\ \mathsf{vpm{:}VPM}' \bullet \textbf{card dom}\ \mathsf{vpm} = \textbf{card rng}\ \mathsf{vpm}\ |\}$
**value**
16. vpr: V-**set** × N → VMP
16a. vpr(vs)(n) ≡
16b. **let** vpa:VP-**set** • vpa ⊆ vps(vs)(n) ∧ **card** vpa = vard vs **in**
16c. **let** vpm:VPM • **dom** vpm = vps ∧ **rng** vpm = vpa **in**
16d. vpm **end end**

## 2.3 Definitions of Auxiliary Functions 33

17. From a net we can extract all its link identifiers.
18. From a net we can extract all its hub identifiers.

**value**
17. xtr_LIs: N → LI-**set**
17. xtr_LIs(n) ≡ {**uid**_L(l)|l:L•l ∈ **obs**_Ls(**obs**_LS(n))}
18. xtr_HIs: N → HI-**set**
18. xtr_HIs(n) ≡ {**uid**_H(l)|h:H•h ∈ **obs**_Hs(**obs**_HS(n))}

19. Given a link identifier and a net get the link with that identifier in the net.
20. Given a hub identifier and a net get the hub with that identifier in the net.

34

**value**
22. get_H: HI → N $\stackrel{\sim}{\rightarrow}$ H
22. get_H(hi)(n) ≡ ι h:H•h ∈ **obs**_Hs(**obs**_HS(n))∧**uid**_H(h)=hi
22. **pre**: hi ∈ xtr_HIs(n)
22a. get_L: LI → N $\stackrel{\sim}{\rightarrow}$ L
22a. get_L(li)(n) ≡ ι l:L•l ∈ **obs**_Ls(**obs**_LS(n))∧**uid**_L(l)=li
22a. **pre**: hl ∈ xtr_LIs(n)

The ι a:A•$\mathcal{P}$(a) expression yields the unique value a:A which satisfies the predicate $\mathcal{P}$(a). If none, or more than one exists then the function is undefined.

## 2.4 Some Derived Traffic System Concepts 35

### 2.4.1 Maps

21. A road map is an abstraction of a road net. We define one model of maps below.

    a A road map, RM, is a finite definition set function, that is, a specification language map from

    - hub identifiers (the source hub)
    - to finite definition set maps from link identifiers
    - to hub identifiers (the target hub).

```
type
21a.  RM′ = HI →m (LI →m HI)
```

If a hub identifier in the definition set or an rm:RM maps into the empty map then the designated hub is "isolated": has no links emanating from it. 36

22. These road maps are subject to a well-formedness criterion.

    a The target hubs must be defined also as source hubs.

    b If a link is defined from source hub (referred to by its identifier) shi via link li to a target hub thi, then, vice versa, link li is also defined from source thi to target shi.

```
type
22.   RM = {| rm:RM′ • wf_RM(rm) |}
value
22.   wf_RM: RM′ → Bool
22.   wf_RM(rm) ≡
22a.      ∪ { rng(rm(hi))|hi:HI•hi ∈ dom rm } ⊆ dom rm
22b.    ∧ ∀ shi:HI•shi ∈ dom rm ⇒
22b.        ∀ li:LI • li ∈ dom rm(shi) ⇒
22b.          li ∈ dom rm((rm(shi))(li)) ∧ (rm((rm(shi))(li)))(li)=shi
```

37

23. Given a road net, n, one can derive "its" road map.

    a Let hs and ls be the hubs and links, respectively of the net n.

    b Every hub with no links emanating from it is mapped into the empty map.

    c For every link identifier uid_L(l) of links, l, of ls and every hub identifier, hi, in the mereology of l

    d hi is mapped into a map from uid_L(l) into hi' e where hi′ is the other hub identifier of the mereology of l.

38

```
value
23.   derive_RM: N → RM
23.   derive_RM(n) ≡
23a.     let hs = obs_Hs(obs_HS(n)), ls = obs_Ls(obs_LS(n)) in
23b.     [ hi ↦ [ ] | hi:HI • ∃ h:H • h ∈ hs ∧ mereo_H(h) = {} ] ∪
23d.     [ hi ↦ [ uid_L(l) ↦ hi′
23e.                     | hi′:HI • hi′ = mereo_L(l)\{hi} ]
23c.          | l:L,hi:HI • l ∈ ls ∧ hi ∈ mereo_L(l) ] end
```

**Theorem:** If the road net, n, is well-formed then wf_RM(derive_RM(n)).

### 2.4.2 Traffic Routes 39

24. A traffic route, tr, is an alternating sequence of hub and link identifiers such that
    a li:LI is in the mereology of the hub, h:H, identified by hi:HI, the predecessor of li:LI in route r, and
    b hi′:HI, which follows li:LI in route r, is different from hi, and is in the mereology of the link identified by li.

```
type
24.   R′ = (HI|LI)*
24.   R = {| r:R′ • ∃ n:N • wf_R(r)(n) |}
value
24.   wf_R: R′ → N → Bool
24.   wf_R(r)(n) ≡
24.     ∀ i:Nat • {i,i+1}⊆inds r ⇒
24a.        is_HI(r(i)) ⇒ is_LI(r(i+1)) ∧ r(i+1) ∈ mereo_H(get_H(r(i))(n)),
24b.        is_LI(r(i)) ⇒ is_HI(r(i+1)) ∧  r(i+1) ∈ mereo_L(get_L(r(i))(n))
```

40

25. From a well-formed road map (i.e., a road net) we can generate the possibly infinite set of all routes through the net.
    a **Basis Clauses:**
       i. The empty sequence of identifiers is a route.
       ii. The one element sequences of link and hub identifiers of links and hubs of a road map (i.e., a road net) are routes.
       iii. If hi maps into some li in rm then ⟨hi,li⟩ and ⟨li,hi⟩ are routes of the road map (i.e., of the road net).

b **Induction Clause:**

i. Let r⌢⟨i⟩ and ⟨i′⟩⌢r′ be two routes of the road map.

ii. If the identifiers i and i′ are identical, then r⌢⟨i⟩⌢r′ is a route.

c **Extremal Clause:**

i. Only such routes that can be formed from a finite number of applications of the above clauses are routes.

41

```
value
25.      gen_routes: RM → Routes-infset
25.      gen_routes(rm) ≡
25(a)i.      let rs = {⟨⟩}
25(a)ii.         ∪ {⟨li,hi⟩,⟨hi,li⟩|li:LI,hi:HI•hi ∈ dom rm∧rm(hi)=li}
25(b)i.          ∪ {let r⌢⟨li⟩,⟨li′⟩⌢r′:R • {r⌢⟨li⟩,⟨li′⟩⌢r′}⊆rs∧li=li′,
25(b)i.                 r′′⌢⟨hi⟩,⟨hi′⟩⌢r′′′:R • {r′′⌢⟨hi⟩,⟨hi′⟩⌢r′′′}⊆rs∧hi=hi′ in
25(b)ii.           r⌢⟨li⟩⌢r′,r′′⌢⟨hi⟩⌢r′′′ end} in
25(c)i.      rs end
```

42

## [1] Circular Routes

26. A route is circular if the same identifier occurs more than once.

```
value
26. is_circular_route: R → Bool
26. is_circular_route(r) ≡ ∃ i,j:Nat • {i,j}⊆inds r ∧ i≠j ⇒ r(i)=r(j)
```

43

## [2] Connected Road Nets

27. A road net is connected if there is a route from any hub (or any link) to any other hub or link in the net.

```
27.  is_conn_N: N → Bool
27.  is_conn_N(n) ≡
27.      let rm = derive_RM(n) in
27.      let rs = gen_routes(rm) in
27.      ∀ i,i′:(LI|HI)•i≠i′∧{i,i′}⊆xtr_LIs(n)∪ xtr_HIs(n)
27.         ∃ r:R • r ∈ rs ∧ r(1)=i ∧ r(len r)=i′ end end
```

44

## [3] Set of Connected Nets of a Net

28. The set, cns, of connected nets of a net, n, is
    a the smallest set of connected nets, cns,
    b whose hubs and links together "span" those of the net n.

```
value
28.  conn_Ns: N → N-set
28.  conn_Ns(n) as cns
28a.    pre: true
28b.    post: conn_spans_HsLs(n)(cns)
28a.          ∧ ∼∃ kns:N-set • card kns < card cns
28a.              ∧ conn_spans_HsLs(n)(kns)
```

45

```
28b.  conn_spans_HsLs: N → N → Bool
28b.  conn_spans_HsLs(n)(cns) ≡
28b.      ∀ cn:N•cn ∈ cns ⇒ is_connected_N(n)(cn)
28b.    ∧ let (hs,ls) = (obs_Hs(obs_HS(n)),obs_Ls(obs_LS(n))),
28b.            chs = ∪{obs_Hs(obs_HS(cn))|cn ∈ cns},
28b.            cls = ∪{obs_Ls(obs_LS(cn))|cn ∈ cns} in
28b.       hs = chs ∧ ls = cls end
```

46

## [4] Route Length

29. The length attributes of links can be
    a added and subtracted,
    b multiplied by reals to obtain lengths,
    c divided to obtain fractions,
    d compared as to whether one is shorter than another, etc., and
    e there is a "zero length" designator.

```
value
29a.  +,− : LEN × LEN → LEN
29b.  ∗ : LEN × Real → LEN
29c.  / : LEN × LEN → Real
29d.  <,≤,=,≠,≥,> : LEN × LEN → Bool
29e.  ℓ0 : LEN
```

47

30. One can calculate the length of a route.

```
value
30. length: R → N → LEN
30. length(r)(n) ≡
30.     case r of:
30.         ⟨⟩ → ℓ0,
30.         ⟨si⟩⌢r′ →
30.             is_LI(si)→attr_LEN(get_L(si)(n))+length(r′)(n)
30.             is_HI(si)→length(r′)(n)
30.     end
```

48

## [5] Shortest Routes

31. There is a predicate, is_R, which,
    - a given a net and two distinct hub identifiers of the net,
    - b tests whether there is a route between these.

```
value
31. is_R: N → (HI×HI) → Bool
31. is_R(n)(fhi,thi) ≡
31a.     fhi ≠ thi ∧ {fht,thi}⊆xtr_HIs(n)
31b.     ∧ ∃ r:R • r ∈ routes(n) ∧ hd r = fhi ∧ r(len r) = thi
```

49

32. The shortest between two given hub identifiers
    - a is an acyclic route, r,
    - b whose first and last elements are the two given hub identifiers
    - c and such that there is no route, r′ which is shorter.

```
value
32. shortest_route: N → (HI×HI) → R
32a. shortest_route(n)(fhi,thi) as r
32b.     pre: pre_shortest_route(n)(fhi,thi)
32c.     post: pos__shortest_route(n)(r)(fhi,thi)
```

50

```
32b. pre_shortest_route: N → (HI×HI) → Bool
32b. pre_shortest_route(n)(fhi,thi) ≡
32b.    is_R(n)(fhi,thi) ∧ fhi≠thi ∧ {fhi,thi}⊂xtr_HIs(n)

32c. pos_shortest_route: N → R → (HI×HI) → Bool
32c. pos_shortest_route(n)(r)(fhi,thi) ≡
32c.    r ∈ routes(n)
32c.   ∧ ∼∃ r′:R • r′ ∈ routes(n) ∧ length(r′) < length(r)
```

### 2.5 States 51

There are different notions of state. In our example these are some of the states: the road net composition of hubs and links; the state of a link, or a hub; and the vehicle position.

## 3 Perdurants 52

For pragmatic reasons we analyse three kinds of perdurants: actions, events and behaviours.

### 3.1 Actions 53

An action is what happens when a function invocation changes, or potentially changes a state. Examples of traffic system actions are: insertion of hubs, insertion of links, removal of hubs, removal of links, setting of hub state (h$\sigma$), moving a vehicle along a link, stopping a vehicle, starting a vehicle, moving a vehicle from a link to a hub and moving a vehicle from a hub to a link. Here we shalljust illustrate one of these actions. Later, in Sect. 3.3,
54 we shall illustrate the vehicle actions.

33. The insert action applies to a net and a hub and conditionally yields an updated net.
    a The condition is that there must not be a hub in the “argument” net with the same unique hub identifier as that of the hub to be inserted and
    b the hub to be inserted does not initially designate links with which it is to be connected.
    c The updated net contains all the hubs of the initial net “plus” the new hub.
    d and the same links.

55

```
value
33. ins_H: N → H ∼→ N
33. ins_H(n)(h) as n′, pre: pre_ins_H(n)(h), post: post_ins_H(n)(h)
```

33a. pre_ins_H(n)(h) ≡
33a. ∼∃ h′:H • h′ ∈ **obs_**Hs(n) ∧ **uid_**HI(h)=**uid_**HI(h′)
33b. ∧ **mereo_**H(h) = {}

33c. post_ins_H(n)(h)(n′) ≡
33c. **obs_**Hs(n) ∪ {h} = **obs_**Hs(n′)
33d. ∧ **obs_**Ls(n) = **obs_**Ls(n′)

We leave it as exercises to define the other hub and link actions.

## 3.2 Events 56

By an event we understand a state change resulting indirectly from an unexpected application of a function, that is, that function was performed "surreptitiously". Events can be characterised by a pair of (before and after) states, a predicate over these and, optionally, a time or time interval. Events are thus like actions: change states, but are usually either caused by "previous" actions, or caused by "an outside action". 57

34. Link disappearance is expressed as a predicate on the "before" and "after" states of the net. The predicate identifies the "missing" ℓink (!).

34. link_dis: N × N → **Bool**
34. link_dis(n,n′) ≡
34. ∃ ℓ:L • pre_link_dis(n,ℓ) ⇒ post_link_dis(n,ℓ,n′)
35. pre_link_dis: N × L → **Bool**
35. pre_link_dis(n,ℓ) ≡ ℓ ∈ **obs_**Ls(n)

58

35. Before the disappearance of link $\ell$ in net $n$
   a the hubs $h'$ and $h''$ connected to link $\ell$
   b were connected to links identified by $\{l'_1, l'_2, \ldots, l'_p\}$ respectively $\{l''_1, l''_2, \ldots, l''_q\}$
   c where, for example, $l'_i, l''_j$ are the same and equal to uid_Π($\ell$).

59

36. After link $\ell$ disappearance there are instead
   a two separate links, $\ell_i$ and $\ell_j$, "truncations" of $\ell$
   b and two new hubs $h'''$ and $h''''$
   c such that $\ell_i$ connects $h'$ and $h'''$ and
   d $\ell_j$ connects $h''$ and $h''''$.
37. Existing hubs $h'$ and $h''$ now have mereology

a $\{l'_1, l'_2, \ldots, l'_p\} \setminus \{\mathsf{uid\_}\Pi(\ell)\} \cup \{\mathsf{uid\_}\Pi(\ell_i)\}$ respectively

b $\{l''_1, l''_2, \ldots, l''_q\} \setminus \{\mathsf{uid\_}\Pi(\ell)\} \cup \{\mathsf{uid\_}\Pi(\ell_j)\}$

38. All other hubs and links of $n$ are unaffected.

We shall not express the above properties explicitly. Instead we expect such a predicate to
60 hold for the interpretation now give.

39. We shall "explain" *link disappearance* as the combined, instantaneous effect of
    a first a remove link "event" where the removed link connected hubs $\mathsf{hi}_j$ and $\mathsf{hi}_k$;
    b then the insertion of two new, "fresh" hubs, $\mathsf{h}_\alpha$ and $\mathsf{h}_\beta$;
    c "followed" by the insertion of two new, "fresh" links $\mathsf{l}_{j\alpha}$ and $\mathsf{l}_{k\beta}$ such that
        i. $\mathsf{l}_{j\alpha}$ connects $\mathsf{hi}_j$ and $\mathsf{h}_\alpha$ and
        ii. $\mathsf{l}_{k\beta}$ connects $\mathsf{hi}_k$ and $\mathsf{h}_{k\beta}$.

61

```
value
39.      post_link_dis(n,ℓ,n′) ≡
39.          let (h_a,h_b):H×H •
39.              let {li_a,li_b}=mereo_L(ℓ) in
39.              (get_H(li_a)(n),get_H(li_b)(n)) end in
39a.     let n″        = rem_L(n)(uid_L(ℓ)) in
39b.     let hα,hβ:H  •  {hα,hβ}∩obs_Hs(n)={} in
39b.     let n‴        = ins_H(n″)(hα) in
39b.     let n⁗        = ins_H(n‴)(hβ) in
39c.     let ljα,lkβ:L  •  {ljα,lkβ}∩obs_Ls(n)={}
39c.         ∧ mereo_L(ljα) = {uid_H(h_a),uid_H(hα)}
39c.         ∧ mereo_L(lkβ) = {uid_H(h_b),uid_H(hβ)} in
39(c)i.  let n‴‴′      = ins_L(n⁗)(ljα) in
39(c)ii. n′ = ins_L(n‴‴′)(lkβ) end end end end end end end
```

## 3.3 Behaviours

62

### 3.3.1 Traffic

**[1] Continuous Traffic** For the road traffic system perhaps the most significant example of a behaviour is that of its traffic:

40. the continuous time varying discrete positions of vehicles, vp:VP[2],

41. where time is taken as a dense set of points.

[2]For VP see Item 12a on page 8.

**type**
41. $c\mathbb{T}$
40. $cRTF = c\mathbb{T} \rightarrow (V \underset{m}{\rightarrow} VP)$ 63

**[2] Discrete Traffic** We shall model, not continuous time varying traffic, but

42. discrete time varying discrete positions of vehicles,
43. where time can be considered a set of linearly ordered points.

43. $d\mathbb{T}$
42. $dRTF = d\mathbb{T} \underset{m}{\rightarrow} (V \underset{m}{\rightarrow} VP)$

44. The road traffic that we shall model is, however, of vehicles referred to by their unique identifiers.

**type**
44. $RTF = d\mathbb{T} \underset{m}{\rightarrow} (VI \underset{m}{\rightarrow} VP)$ 64

**[3] Time: An Aside** We shall take a rather simplistic view of time [2, 3, 5, 6].

45. We consider $d\mathbb{T}$, or just $\mathbb{T}$, to stand for an ordered set of time points.
46. And we consider $\mathbb{TI}$ to stand for time intervals based on $\mathbb{T}$.
47. We postulate an infinitesimal small time interval $\delta$.
48. $\mathbb{T}$, in our presentation, has lower and upper bounds.
49. We can compare times and we can compare time intervals.
50. And there are a number of "arithmetics-like" operations on times and time intervals. 65

**type**
45. $\mathbb{T}$
46. $\mathbb{TI}$

**value**
47. $\delta$:$\mathbb{TI}$
48. $\mathbb{MIN}, \mathbb{MAX}$: $\mathbb{T} \rightarrow \mathbb{T}$
48. $<,\leq,=,\geq,>$: $(\mathbb{T}\times\mathbb{T})|(\mathbb{TI}\times\mathbb{TI}) \rightarrow$ **Bool**
49. $-$: $\mathbb{T}\times\mathbb{T} \rightarrow \mathbb{TI}$
50. $+$: $\mathbb{T}\times\mathbb{TI},\mathbb{TI}\times\mathbb{T} \rightarrow \mathbb{T}$
50. $-,+$: $\mathbb{TI}\times\mathbb{TI} \rightarrow \mathbb{TI}$
50. $*$: $\mathbb{TI}\times$**Real** $\rightarrow \mathbb{TI}$
50. $/$: $\mathbb{TI}\times\mathbb{TI} \rightarrow$ **Real**

66

### [4] Global Clock

51. We postulate a global clock behaviour which offers the current time.

52. We declare a channel clk_ch.

**value**
51. clock: $\mathbb{T} \rightarrow$ **out** clk_ch **Unit**
51. clock(t) $\equiv$ ... ; clk_ch!t ; ... ; clock(t $\lceil\rceil$ t+$\delta$)
channnel
52. clk_ch:$\mathbb{T}$

### 3.3.2 Globally Observable Parts 67

There is given

53. a net, n:N,

54. a set of vehicles, vs:V-**set**, and

55. a monitor, m:M.

The n:N, vs:V-**set** and m:M are observable from the road traffic system domain.

**value**
53. n:N = **obs_**N($\Delta$)
53. ls:L-**set** = **obs_**Ls(**obs_**LS(n)), hs:H-**set** = **obs_**Hs(**obs_**HS(n)),
53. lis:LI-**set** = {**uid_**L(l)|l:L•l $\in$ ls}, his:HI-**set** = {**uid_**H(h)|h:H•h $\in$ hs}
54. vs:V-**set** = **obs_**Vs(**obs_**VS(**obs_**F($\Delta$))), vis:V-**set** = {**uid_**V(v)|v:V•v $\in$ vs}
55. m:**obs_**M($\Delta$)

### 3.3.3 Road Traffic System Behaviours 68

56. Thus we shall consider our road traffic system, rts, as
    a the concurrent behaviour of a number of vehicles,
    b a monitor behaviour,
    c an initial vehicle position map, and
    d an initial starting time.

69

**value**
56c. vpm:VPM = vpr(vs)(n)
56d. $t_0$:T = clk_ch?

56. rts() =
56a. ∥ {veh(**uid_**V(v))(v)(vpm(**uid_**V(v)))|v:V•v ∈ vs}
56b. ∥ mon(m)([$t_0$ ↦ vpm])

where the "extra" monitor argument, rtf:RTF, records the discrete road traffic initially set to the singleton map from an initial start time, $t_0$ to the initial assignment of vehicle positions.

### 3.3.4 Channels 70

In order for the monitor behaviour to assess the vehicle positions these vehicles communicate their positions to the monitor via a vehicle to monitor channel. In order for the monitor to time-stamp these positions it must be able to "read" a clock.

57. Thus we declare a set of channels indexed by the unique identifiers of vehicles and communicating vehicle positions.

**channel**
57. {vm_ch[vi]|vi:VI•vi ∈ vis}:VP

### 3.3.5 Behaviour Signatures 71

58. The road traffic system behaviour, rts, takes no arguments (hence the first **Unit**); and "behaves", that is, continues, forever (hence the last **Unit**).

59. The vehicle behaviours are indexed by the unique identifier, uid_V(v):VI, the vehicle part, v:V and the vehicle position; offers communication to the monitor behaviour (on channel vm_ch[vi]); and behaves "forever".

60. The monitor behaviour takes the so far unexplained monitor part, m:M, as one argument and the discrete road traffic, drtf:dRTF, being repeatedly "updated" as the result of **in**put communications from (all) vehicles; the behaviour otherwise runs forever.

**value**
58. rts: **Unit** → **Unit**
59. veh: vi:VI → v:V → VP → **out** vm_ch[vi],mi:MI **Unit**
60. mon: m:M → RTF → **in** {vm_ch[vi]|vi:VI•vi ∈ vis},clk_ch **Unit**

### 3.3.6 The Vehicle Behaviour 72

61. A vehicle process is indexed by the unique vehicle identifier vi:VI, the vehicle "as such", v:V and the vehicle position, vp:VPos.

    73 The vehicle process communicates with the monitor process on channel vm[vi] (sends, but receives no messages), and otherwise evolves "in[de]finitely" (hence **Unit**).

62. We describe here an abstraction of the vehicle behaviour at a Hub (hi).
    a Either the vehicle remains at that hub informing the monitor,
    b or, internally non-deterministically,
        i. moves onto a link, tli, whose "next" hub, identified by thi, is obtained from the mereology of the link identified by tli;
        ii. informs the monitor, on channel vm[vi], that it is now on the link identified by tli,
        iii. whereupon the vehicle resumes the vehicle behaviour positioned at the very beginning (0) of that link,
    c or, again internally non-deterministically,
    d the vehicle "disappears — off the radar" !

74

```
62.        veh(vi)(v)(vp:atH(fli,hi,tli)) ≡
62a.           vm_ch[vi]!vp ; veh(vi)(v)(vp)
62b.        ⌈⌉
62(b)i.        let {hi′,thi}=mereo_L(get_L(tli)(n)) in assert: hi′=hi
62(b)ii.       vm_ch[vi]!onL(tli,hi,0,thi) ;
62(b)iii.      veh(vi)(v)(onL(tli,hi,0,thi)) end
62c.        ⌈⌉
62d.           stop
```

75

63. We describe here an abstraction of the vehicle behaviour on a Link (ii).
    Either
    a the vehicle remains at that link position informing the monitor,
    b or, internally non-deterministically,
    c if the vehicle's position on the link has not yet reached the hub,
        i. then the vehicle moves an arbitrary increment $\delta$ along the link informing the monitor of this, or
        ii. else, while obtaining a "next link" from the mereology of the hub (where that next link could very well be the same as the link the vehicle is about to leave),

A. the vehicle informs the monitor that it is now at the hub identified by thi,

B. whereupon the vehicle resumes the vehicle behaviour positioned at that hub.

64. or, internally non-deterministically,

65. the vehicle "disappears — off the radar" !

76

```
61.        veh(vi)(v)(vp:onL(fhi,li,f,thi)) ≡
63a.           vm_ch[ vi ]!vp ; veh(vi)(v)(vp)
63b.       ⌈⌉
63c.           if f + δ<1
63(c)i.            then vm_ch[ vi ]!onL(fhi,li,f+δ,thi) ;
63(c)i.                 veh(vi)(v)(onL(fhi,li,f+δ,thi))
63(c)ii.           else let li′:LI•li′ ∈ mereo_H(get_H(thi)(n)) in
63(c)iiA.               vm_ch[ vi ]!atH(li,thi,li′);
63(c)iiB.               veh(vi)(v)(atH(li,thi,li′)) end end
64.          ⌈⌉
65.             stop
```

### 3.3.7 The Monitor Behaviour 77

66. The monitor behaviour evolves around the attributes of an own "state", m:M, a table of traces of vehicle positions, while accepting messages about vehicle positions and otherwise progressing "in[de]finitely".

67. Either the monitor "does own work"

68. or, internally non-deterministically accepts messages from vehicles.

   a A vehicle position message, vp, may arrive from the vehicle identified by vi.

   b That message is appended to that vehicle's movement trace,

   c whereupon the monitor resumes its behaviour —

   d where the communicating vehicles range over all identified vehicles.

78

```
66.    mon(m)(rtf) ≡
67.        mon(own_mon_work(m))(rtf)
68.    ⌈⌉
68a.     ⌈⌉ { let ((vi,vp),t) = (vm_ch[ vi ]?,clk_ch?) in
68b.          let rtf′ = rtf † [ t ↦ rtf(max dom rtf) † [ vi ↦ vp ] ] in
```

68c. mon(m)(rtf′) **end**
68d. **end** | vi:VI • vi ∈ vis }

67. own_mon_work: M → RTF → M

# 4 Conclusion 79

## 4.1 What Have We Achieved ? 80

## 4.2 Future Work 81

## 4.3 Comparion to Other Work 82

## 4.4 Acknowledgements 83

# 5 Bibliography 84

## 5.1 Bibliographical Notes

## 5.2 References

[1] D. Bjørner. Domain Analysis – Towards a Domain Logic. Research Report 2013-1, DTU Compute, Spring 2013.

[2] W. D. Blizard. A Formal Theory of Objects, Space and Time. *The Journal of Symbolic Logic*, 55(1):74–89, March 1990.

[3] J. M. E. McTaggart. The Unreality of Time. *Mind*, 18(68):457–84, October 1908. New Series. See also: [4].

[4] R. L. Poidevin and M. MacBeath, editors. *The Philosophy of Time*. Oxford University Press, 1993.

[5] A. N. Prior. *Papers on Time and Tense*. Clarendon Press, Oxford, UK, 1968.

[6] J. van Benthem. *The Logic of Time*, volume 156 of *Synthese Library: Studies in Epistemology, Logic, Methhodology, and Philosophy of Science (Editor: Jaakko Hintika)*. Kluwer Academic Publishers, P.O.Box 17, NL 3300 AA Dordrecht, The Netherlands, second edition, 1983, 1991.

# A Illustrations 85