4. Derived Urban Plannings

Third Lecture

- **Agenda:**
  - Preliminaries 66–73
  - The Derived Urban Planning Functions 74–77
  - The Derived Urban Planning Behaviours 78–78
  - The Derived Resumption Repository 79–82
  - A Visual Rendition of Urban Planning 83–87
  - The Urban Planning System 88–93

4.1. Preliminaries
4.1.1. Derived Urban Plan Indices

- We think of *base* urban planning function, modeled by base_up_fct,
  - as being concerned with the overall "division" of the urban space (i.e., geographical area, land and water) into zones for building, recreation, and other (i.e., the *master plan*).
  - Aggregations of these zones, one, more or all (usually several), can then be further "[derive] planned" into
    - (d_1) light, medium and heavy industry zones,
    - (d_2) mixed shopping and residential zones,
    - (d_3) apartment bldg. zones,
    - ..., etc., etc.,
    - (d_{m-1}) villa zones, and
    - (d_m) recreational zones.

- Additional forms of derived plannings are:
  - (d_{m+1}) transport,
  - (d_{m+2}) electricity supply,
  - (d_{m+3}) water supply,
  - (d_{m+4}) waste management,
  - (d_{m+5}) health care,
  - (d_{m+6}) fire brigades,
  - ..., etc., etc.,
  - (d_{m+n}) schools.

- We refer to the d_i's as derived urban plan indices.

42 We think of this variety of "derived" plannings as indexed such as hinted at above,

43 and dups as the set of all indices.

```
type
42   DP == {|d1,d2,...,dn|}
value
43   dups:DP-set = {d1,d2,...,dn}
```

4.1.2. A "Reservoir" of Derived Urban Planning Indices

44 To secure that at most one derived planning, d_i (per i), is initiated

- we introduce a global variable, dps_var,
- initialised to an empty set of derived planning tokens
- and updated with the addition of selected DP tokens.

variable

44 dps_var:DP-**set** := {} **comment** dps_var *denotes a reference*

4.1.3. A Derived Urban Planning Index Selector

45 A function, sel_dps, selects zero, one or more "fresh" DP indices, that is, DP tokens that have not been selected before.

```
value
45      sel_dps: Unit → DP-set
45      sel_dps() ≡
45          let dps:DP-set·dps⊆dups \ c dps_var in
45          dps_var := c dps_var ∪ dps; dps end
comment
44      [ c denotes a contents-taking operator ]
```

- We shall revise the above selector in on Slide 90.

4.1.4. Let us recall:

```
value
31   base_up_beh_1: Unit → in b_tri_ch out b_qui_ch Unit
37   base_up_beh_1() ≡
38       let b_qui = b_base_up_fct(b_tri_ch?)(b_qui_ch?) in b_qui_ch!b_qui ;
39       if b_qui_satisfactory(b_qui)
40           then skip
41           else base_up_beh_1() end
37       end
```

4.1.5. The Derived Urban Plan Generator

46 We therefore edit the base_up_beh_1 behaviour slightly into the revised base_up_beh_2.

- In base_up_beh_2 we insert, "in parallel" (||) with the "resumption" of base_up_beh_2,
- an internal non-deterministic choice behaviour, der_up().
- It specifies the selection of zero, one of more DP tokens,
- and initiates corresponding derived planning behaviours, $\mathsf{der\_up\_beh}_i()$,
- as well as their corresponding "input" triplet oracles, $\mathsf{d\_tri\_beh}_i()$.
- But only at most once.

47 These derived planning behaviours, $\mathsf{der\_up\_beh}_i$, and "input" triplet oracles, $\mathsf{d\_tri\_beh}_i()$

- are like base_up_beh_1, respectively b_tri_beh,
- only now they are "tuned" to the specific derived planning issues (i.e., $_i$).

value

46 der_up: **Unit** → **Unit**

46 der_up() ≡ **let** dps = sel_dps() **in** ||{$\mathsf{der\_up\_beh}_i()$||$\mathsf{d\_tri\_beh}_i()$|i:DP•i ∈ dps} **end**

- We shall introduce the $\mathsf{der\_up\_beh}_i$ and $\mathsf{d\_tri\_beh}_i$ behaviours below.

4.1.6. The Revised Base Urban Planning Behaviour

- We "take over" the basic structure and definition ("contents") of the urban planning function and behaviour from that of the base versions.

48 We think of zero, one or more derived plannings (der_up_beh$_1$, der_up_beh$_2$, ..., der_up_beh$_n$) being initiated after some stage of *base* function, base_up_fct, has concluded.

```
value
37'' base_up_beh_2() ≡
41''     let b_qui=base_up_fct(b_tri_ch?)(b_qui_ch?) in b_qui_ch!b_qui ;
39''     if base_satisfactory(b_qui)
37''        then skip
38''        else
48             der_up() ||
40''           base_up_beh_2() end end
```

4.2. The Derived Urban Planning Functions

- An important form of information for each derived urban planning function
  - is the resumption, i.e., the quintuplet information
  - from the base urban behaviour:
  - bQui.

49 The new forms of information are:

- the derived urban planning auxiliary, dAUX_i,
- the derived urban planning requirements information, dREQ_i,
- as well as the derived urban planning plans, dPLA_i,
- and their ancillary information, dANC_i.

50 The primary arguments for the derived urban planning function, base_up_fct, is therefore a "quintuplet", d_qui:dQUI, of

- a base triplet, b_tri:bTRI, and
- the pair of
  - the derived urban planning auxiliary information, $\mathsf{d\_aux}_i$:dAUX_i,
  - and the derived urban planning requirements, $\mathsf{d\_req}_i$:dREQ_i.

The result of derived urban planning function, der_up_fct, as for the base urban planning function, base_up_fct,

51 is that of a "quintuplet", also a resumption, dQUI_i, of the primary arguments, b_tri:bTRI, and

52 the result, a pair of a derived plan, $\mathsf{d\_pla}_i$, and derived ancillaries, $\mathsf{d\_anc}_i$.

53 As for the base urban planning function, base_up_fct, it has a secondary, derived "quintuplet" argument (which, as for base_up_fct, helps "kick-start" urban planning). This second argument is the result of a previous application of the der_up_fct.

54 The derived urban planning function $\mathsf{der\_up\_fct}_i$ signature is therefore that of a function from a triplet of a most recent base quintuplet, derived urban planning auxiliary and derived urban planning requirements information to functions from derived "quintuplet" arguments to derived "quintuplet" results.

55 The triplet argument, $\mathsf{d\_tri}_i$, and the first part of the result, also a triplet, $\mathsf{d\_tri}'_i$, are the same.

56 The derived urban planning function $\mathsf{der\_up\_fct}_i$ is further characterised by a predicate, $\mathcal{P}_{\mathsf{der}_i}$, which we leave further undefined.

type

49 dAUX$_1$, dAUX$_2$, ..., dAUX$_n$
49 dREQ$_1$, dREQ$_2$, ..., dREQ$_n$
49 dPLA$_1$, dPLA$_2$, ..., dPLA$_n$
49 dANC$_1$, dANC$_2$, ..., dANC$_n$
50 dTRI$_i$ = bQUI×dAUX$_i$×dREQ$_i$ [i:DP·i∈dups]
52 dRES$_i$ = dPLA$_i$ × dANC$_i$ [i:DP·i∈dups]
51 dQUI$_i$ = dTRI$_i$×dRES$_i$ [i:DP·i∈dups]

value

53 der_up_fct$_i$: dTRI$_i$ → dQUI$_i$m → dQUI$_i$ i:DP
54 der_up_fct$_i$(d_tri$_i$)(d_qui$_i$) **as** (d_tri$'_i$,d_res$_i$)
55 d_tri$_i$ = d_tri$'_i$ ∧
56 $\mathcal{P}_{\text{der}_i}$(d_tri$'_i$,d_res$_i$)

56 $\mathcal{P}_{\text{der}_i}$: dTRI$_i$ × dRES$_i$ → **Bool**

4.3. The Derived Urban Planning Behaviour

57 We think of zero, one or more derived plannings
(der_up_beh$_{i_1}$, der_up_beh$_{i_2}$, ..., der_up_beh$_{i_m}$)
being initiated after some stage of the *der_up_fct$_i$* function has concluded.

```
value
37''' der_up_beh_i() ≡
41'''     let d_qui_i = der_up_fct_i(d_tri_ch[i]?)(d_qui_ch[i]?) in
41'''     d_qui_ch[i] ! d_qui_i ;
39'''     if der_satisfactory_i(d_qui_i)
37'''         then skip
38'''         else
57                der_up() ||
40'''             der_up_beh_i() end end
```

4.4. The Derived Resumption Repository

4.4.1. The Consolidated Derived Resumption Map

58 The derived urban planning functions (and thus behaviours) operate, not on simple resumptions, as do the base urban planning functions (and behaviours), but on the aggregation of all derived functions' (etc.) quintuplets, that is, an indexed set of quintuplets – modeled as a derived resumptions map.

type

58 $\mathsf{dQUIm} = \mathsf{DP} \xrightarrow[m]{} \mathsf{dQUI}_i$

4.4.2. The Consolidated Derived Resumption Repository Channel

59 Communications between the individual derived urban planning behaviours and the consolidated derived resumption repository are via an indexed set of channels communicating derived resumptions maps.

channel

59 $\{d\_qui\_ch[\,i\,]:dQUIm|i:DP\cdot i \in dups\}$

4.4.3. The Consolidated Derived Resumption Repository

60 The consolidated derived resumption repository behaviour either (□) updates its state map with received individual derived resumptions, or offers the entire such state maps to whichever derived urban planning behaviour so requests.

```
value
60  d_qui_beh: dQUIm → in,out der_qui_ch[ i ]  Unit  i:DP
60  d_qui_beh(d_qui_m) ≡
60    (d_qui_beh(d_qui_m†[ i↦d_qui_ch[ i ]? ])
60     □
60     d_qui_ch[ i ]!(d_qui_m)) ;
60    d_qui_beh(d_qui_m)
```

4.4.4. Initial Consolidated Derived Urban Plannings

value

$$\mathsf{d\_qui\_m} = [\ \mathsf{d}_1 \mapsto \mathsf{init\_d\_qui}_1,\ ...,\ \mathsf{d}_n \mapsto \mathsf{init\_d\_qui}_n\ ]$$

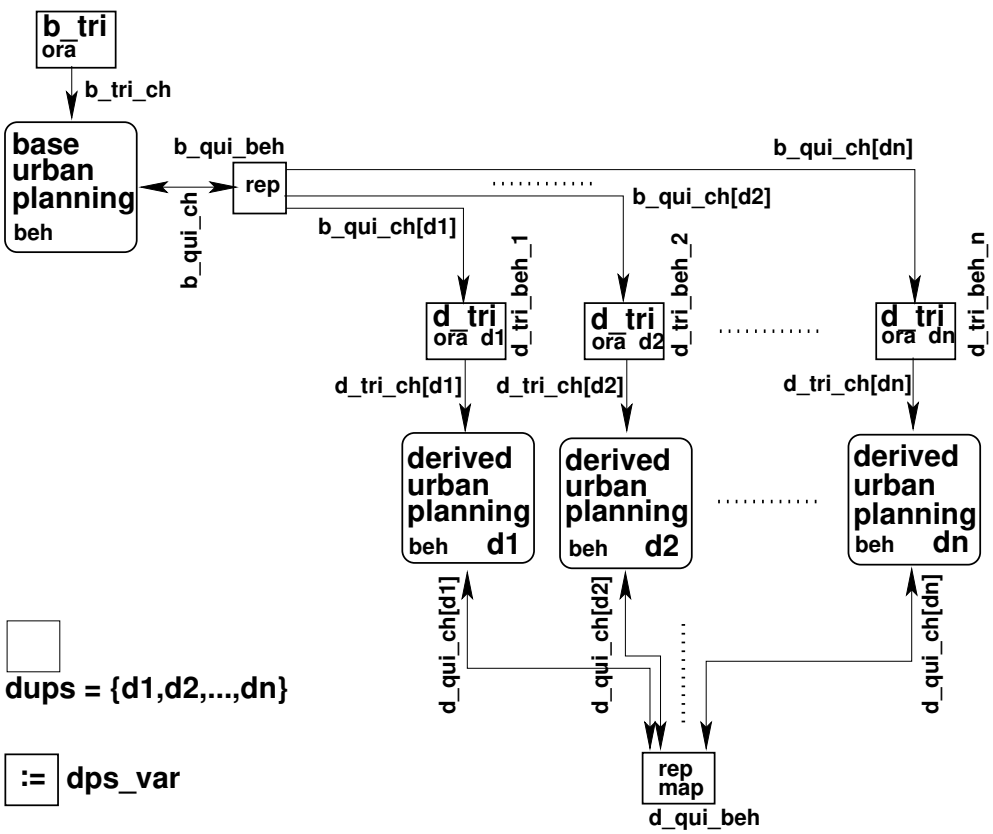
4.4.5. Initialisation of The Derived Quintuplet Oracle

- As for base oracle and repository behaviours we initialise the derived quintuplet oracle:

  der_qui_beh(init_d_qui_m)

4.5. A Visual Rendition of Urban Planning Development

- The urban planning domain, when "operating at full speed", consists of
  - the base urban planning behaviour (i.e., project),
  - zero, one or more derived urban planning behaviours, each of the latter initiated by either
    - the base urban planning project or
    - a derived urban planning project.



Legend:

geo: geodetic, geotechnical, meteorological query
aux: auxiliary information query
req: requirements query
pla: plan
anc: ancillary information
b_, d_: base, derived
b_tri: (b_geo,b_aux,b_req) "triplet"
b_qui: ((b_geo,b_aux,b–req),(b_pla,b_anc)) "quintuplet"
rep: repository
ora: oracle
d_tri: (b_qui,d_aux,d_req) "triplet"
d_qui: (d_tri,(d_pla,d_anc)) "quintuplet"
map: d_qui map
dps_var: derived urban planning index set variable

rep repository behaviour
:= variable
ora oracle behaviour
value
urban planning behaviour
input [query] channel
input/output [query/deposit] channel
beh: behaviour

- The planning behaviours, both the base and the derived, invoke respective urban planning functions, and these produce, such as we have modeled them, **qui**ntuplets of information, which are deposited with respective **qui**ntuplet repository behaviours:
  - the base **qui**ntuplet repository behaviour, and
  - the derived **qui**ntuplet repository behaviour — which maintains these quintuplets for all (invoked and thus ongoing) derived urban planning projects.

- We kindly ask you to review the figure given on Slide 84.
  - All you have to 'master' is the fact that there is
    - one base urban planning project, with its repository of base urban planning "quintuplets", and
    - between 0 and n derived urban planning projects, with their shared, derived urban planning "quintuplets",

- Then there are the channels: the query (input) channels
  - providing auxiliary and requirements information to both the one base urban planning project and
  - the n derived urban planning projects;
- and the query/repository channels
  - providing "quintuplet" aggregated information to the base urban planning project,
  - as well as "quintuplet" aggregated information to the derived urban planning projects.

- Finally there are the "global"
  - value representing the index set of derived urban planning indices, and
  - variable which holds the index set of derived urban planning indices of ongoing derived urban planning projects.

4.6. Revised Selection of Derived Urban Plannings

4.6.1. Review

- The derived urban planning generator function, der_up, cf. Item 46 on Slide 72,

```
value
46   der_up: Unit → Unit
46   der_up() ≡
46       let dps = sel_dps() in
46       ‖ {der_up_beh_i()‖d_tri_beh_i()|i:DP•i ∈ dps} end
```

- was invoked with no arguments, der_up(),
- cf. Item 48 on Slide 73 and Item 57 on Slide 78

```
48              der_up() ‖ [ respectively ]
57              der_up() ‖
```

4.6.2. A Potential Derived Urban Plan Indices Selector

- Selection of potential derived urban planning indices was therefore rather arbitrary.
- We now let the selection depend on the aggregated resumption state of all (ongoing and) derived urban planning behaviours.

61 Function sel_dups examines either the base resumption or the aggregated resumption state of all (ongoing and) derived urban planning behaviours and yields a set of derived urban planning indices.

62 How it does that is, of course, not defined here.

```
value
61  sel_dups: (bQUI|dQUIm) → DP-set
62  sel_dups(dquim) ≡ ...
```

4.6.3. A Revised Derived Urban Plan Index Set Selector

63 We revise the derived urban plan index selector function give earlier, cf. Item 45 on Slide 70. A function, sel_dps, selects zero, one or more DP "fresh" indices, that is, DP tokens that have not been selected before.

```
value
63   sel_dps: DP-set → DP-set
63   sel_dps(dups) ≡
63     let dps:DP-set·dps⊆dups∩dups \ c dps_var in
63     dps_var := c dps_var ∪ dps; dps end
```

4.6.4. Revision of Derived Urban Plan Invocation

- We need to revise the two occurrences of der_up() –
  - in the base urban planning behaviour, and
  - in the (indexed set of) derived urban planning behaviours.
- Thus

```
48          der_up() || [ respectively ]
57          der_up() ||
```

- is to be replaced by:

```
48          der_up(b_qui_ch?) || ... [ respectively ]
57          der_up(d_qui_ch[ i ]?) || ...
```

4.7. The Urban Planning System

64 Finally we can define an urban planning development as a system of concurrent behaviours:

- the base urban planning behaviour,
- the base "quintuplet" repository and
- the derived and consolidate "quintuplet" repository

value

64 up_sys: **Unit** → **Unit**

64 up_sys() ≡ base_up_beh() ‖ b_qui_beh(b_qui_init) ‖ d_qui_beh(d_qui_m)

- Recall that
  - ◈ the derived urban planning behaviours as well as
  - ◈ the derived triplet behaviours

  are started by the base as well as the derived urban planning behaviours.

This was all for today !

- On Tuesday 19 September we shall discuss
  - what has been achieved,
  - what needs to be done

  and I will present
  - the beginnings of a formal model of the information type TUS.