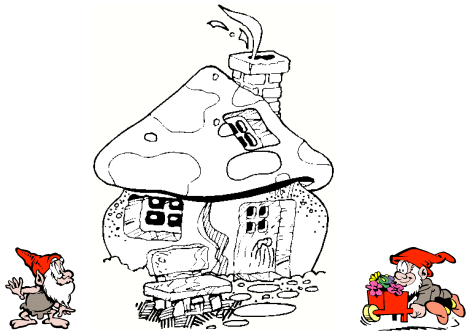
Advanced Design of Information Systems

Lecture 5.2



Prof.Dr.ir. Jan L.G. Dietz

IN3510 (and IN4154) in 2005/2006

TU Delft
Delft University of Technology

Message on a Dutch railway station ...

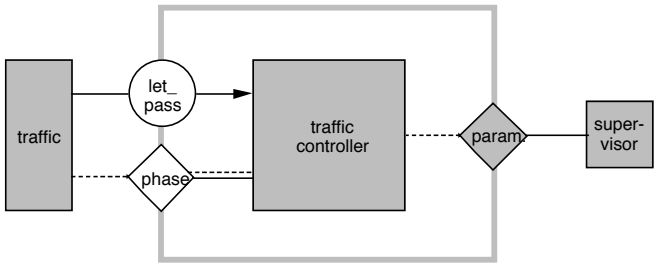
Wegens een storing aan het informatie-systeem kunnen de informatiepanelen op de perrons geen onjuiste informatie weergeven. Voor de juiste informatie raadpleegt u de vertrekstaten.

Making a SMART specification in 10 steps

Step 1 : Draw the global smartienet.
Step 2 : Specify the storage bases of the interface banks.
Step 3 : Specify the emission bases of the interface channels.
Step 4 : Draw the detailed smartienet.
Step 5 : Specify the storage bases of the internal banks.
Step 6 : Specify the emission bases of the internal channels.
*Step 7 : Specify the **S** and **M** of every elementary processor.*
*Step 8 : Specify the **A** and **R** of every elementary processor.*
*Step 9 : Specify the **T** of every elementary processor.*
Step 10 : Validate the resulting ontology.

SMART specification of the TCS (1)

Step 1 : Draw the global smartienet.



SMART specification of the TCS (2)

Step 2 : Specify the storage bases of the interface banks.

The storage base of the bank "param" is:

ext(move_time) [**dom**={1,2}, **rng**=D] ∪
ext(stop_time) [**dom**={1,2}, **rng**=D] ∪
ext(clear_time) [**dom**={1,2}, **rng**=D]

The storage base of the bank "phase" is:

ext(phase) [**dom**={1,2}, **rng**={wait,stop,move}]

SMART specification of the TCS (3)

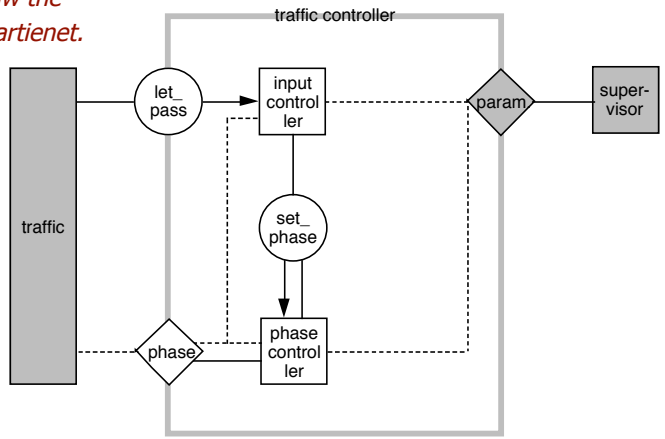
Step 3 : Specify the emission bases of the interface channels.

The emission base of the channel "let_pass" is:

{let_pass(1), let_pass(2)}

SMART specification of the TCS (4)

Step 4 : Draw the detailed smartienet.



SMART specification of the TCS (5)

Step 5 : Specify the storage bases of the internal banks.

< there are no internal banks >

SMART specification of the TCS (6)

Step 6 : Specify the emission bases of the internal channels.

The emission base of the channel "set_phase" is:

ext(set_phase) [**dom**={1,2}, **rng**={wait,stop,move}]

SMART specification of the TCS (7.1)

*Step 7.1 : Specify the **S** and **M** of the processor "input controller".*

S = **ext**(phase) [**dom**={1,2}, **rng**={wait,stop,move}] ∪
ext(move_time) [**dom**={1,2}, **rng**=D] ∪
ext(stop_time) [**dom**={1,2}, **rng**=D] ∪
ext(clear_time) [**dom**={1,2}, **rng**=D]

M = ∅

SMART specification of the TCS (7.2)

*Step 7.2 : Specify the **S** and **M** of the processor "phase controller".*

S = **ext**(phase) [**dom**={1,2}, **rng**={wait,stop,move}] ∪
ext(move_time) [**dom**={1,2}, **rng**=D] ∪
ext(stop_time) [**dom**={1,2}, **rng**=D] ∪
ext(clear_time) [**dom**={1,2}, **rng**=D]

M = **ext**(phase) [**dom**={1,2}, **rng**={wait,stop,move}]

SMART specification of the TCS (8.1)

*Step 8 : Specify the **A** and **R** of the processor "input controller"..*

A = {let_pass(1), let_pass(2)}

R = **ext**(set_phase) [**dom**={1,2}, **rng**={wait,stop,move}]

SMART specification of the TCS (8.2)

*Step 8 : Specify the **A** and **R** of the processor "phase controller"..*

A = **ext**(set_phase) [**dom**={1,2}, **rng**={wait,stop,move}]

R = **ext**(set_phase) [**dom**={1,2}, **rng**={wait,stop,move}]

SMART specification of the TCS (9)

*Step 9 : Specify the **T** of every elementary processor role.*

Take all possible situations of external activation into account.

Determine whether an action is necessary or that the activation can safely be ignored.

If action is needed, specify the transition rule.

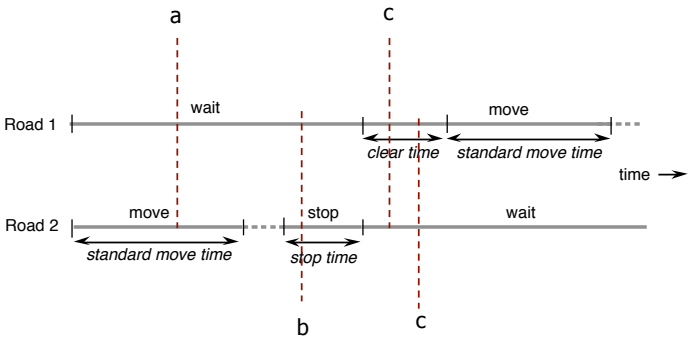
SMART specification of the TCS (9.1)

*Step 9.1 : Specify the **T** of the processor "input controller".*

Situations in which a let_pass(Cycle) can occur:

Cycle	**Crosscycle**		
wait	move	take action	(a)
wait	stop	ignore	
wait	wait	take action	(c)
move	wait	ignore	
stop	wait	take action	(b)
< all other combinations >		raise alarm!	

Conditions for taking action



SMART specification of the TCS (9.1)

*Step 9.1 : Specify the **T** of the processor "input controller".*

Situation a:

```
on let_pass(Cycle) →
    if  phase(Cycle) = wait and phase(Crosscycle) = move
        and not (set_phase(Crosscycle,stop) →
        generate set_phase(Crosscycle,stop) with delay
                max(0,(move_time(Crosscycle) -
                age(phase(Crosscycle) = move)))
    fi
no
```


SMART specification of the TCS (9.1)

*Step 9.1 : Specify the **T** of the processor "input controller".*

Situation b:

```
on let_pass(Cycle) →
    if  phase(Cycle) = stop and phase(Crosscycle) = wait
        and not (set_phase(Crosscycle,stop) →
        generate set_phase(Crosscycle,stop) with delay
                stop_time(Cycle) - age(phase(Cycle) = stop
                + clear_time(Crosscycle) + move_time(Crosscycle)
    fi
no
```


SMART specification of the TCS (9.1)

*Step 9.1 : Specify the **T** of the processor "input controller".*

Situation c:

```
on let_pass(Cycle) →
    if  phase(Cycle) = wait and phase(Crosscycle) = wait
        and not (set_phase(Cycle,move)
        and not (set_phase(Crosscycle,stop) →
        generate set_phase(Crosscycle,stop) with delay
                clear_time(Crosscycle) - age(phase(Cycle) = wait
                + move_time(Crosscycle)
    fi
no
```


SMART specification of the TCS (9.2)

*Step 9.2 : Specify the **T** of the processor "phase controller".*

```
on set_phase(Cycle,stop) →
    if  phase(Cycle) = move →
        generate set_phase(Cycle,wait) with delay stop_time(Cycle);
        phase(Cycle) := stop
    fi
no
```

SMART specification of the TCS (9.2)

*Step 9.2 : Specify the **T** of the processor "phase controller".*

```
on set_phase(Cycle,wait) →
      if  phase(Cycle) = stop →
          generate set_phase(Crosscycle,move) with delay
          clear_time(Crosscycle);
          phase(Cycle) := wait
      fi
no
```


SMART specification of the TCS (9.2)

*Step 9.2 : Specify the **T** of the processor "phase controller".*

```
on set_phase(Cycle,move) →
      if  phase(Cycle) = wait →
                  phase(Cycle) := move
      fi
no
```


SMART specification of the TCS (10)

Step 10 : Validate the resulting ontology.

Validate the operation of every internal processor role. In particular check whether all possible situations are being dealt with.

A convincing demonstration of the correctness of the ontological model could be its implementation as a simulation model, where the let_pass commands are randomly generated.

Assignment 2 (group assignment)

Let op! Afwijkend van eerdere aankondiging in College 1.1!

1. Maak een ontologisch model van het IES-IS, conform het functionele model dat uit de beschrijving van de casus is te halen. Volg daarbij de 10-stappen-procedure. Let vooral op het onderscheid tussen activeren en informeren: iets dat ontologisch gezien informeren is, kan best als trigger of actie zijn geïmplementeerd!

2. Maak een ontologisch model van een systeem naar eigen keuze, bijvoorbeeld een MP3-speler.

Inleverdatum: 27 Maart
Inlevervorm: papieren exemplaar