

DEVELOPMENT OF CORRECT SOFTWARE WITH B-METHOD

Štefan Korečko

Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice

<https://kpi.fei.tuke.sk/en/person/stefan-korecko>

stefan.korecko@tuke.sk



Contents

1. About
 - ▣ DCI FEEI TU Košice
 - ▣ This Tutorial
 - ▣ B-Method
2. TrainDirector/TS2JavaConn Toolset
3. Formal Specification and Verification in B-Method
4. Verified Refinement in B-Method
5. Practice

About

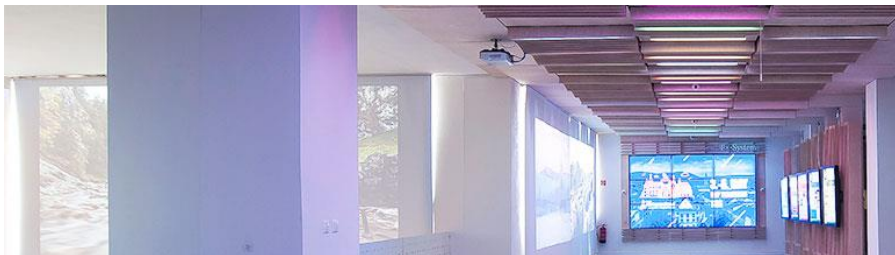
DCI FEEI TU Košice

This Tutorial

B-Method

DCI FEEI TU Košice

- Department of Computers and Informatics
FEEI, Technical University of Košice
- <https://kpi.fei.tuke.sk/en>



This Tutorial

- A practical introduction to the formal software development
 - ▣ specification,
 - ▣ verification,
 - ▣ refinement
- On an example from the railway domain

This Tutorial :: Software Requirements

□ Atelier B

□ <http://www.atelierb.eu/en/download/>

□ Java JDK

□ <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

□ TS2JavaConn/TrainDirector Toolset

□ <http://hron.fei.tuke.sk/~korecko/FMInGamesExp/resources/allInOneTDTS2J.zip>

□ Tutorial package

□ <http://hron.fei.tuke.sk/~korecko/cefp19/cefp19BmethodPack.zip>

B-Method

- Author: Jean-Raymond Abrial
- State based, model-oriented
- For software development
- Based on
 - ▣ Zermelo-Fraenkel set theory
 - ▣ Dijkstra's weakest precondition calculus
- Sophisticated development process
 - ▣ abstract formal specification -> implementation (code)
 - ▣ proof of correctness
 - ▣ Tool: Atelier B (<https://www.atelierb.eu/en>)

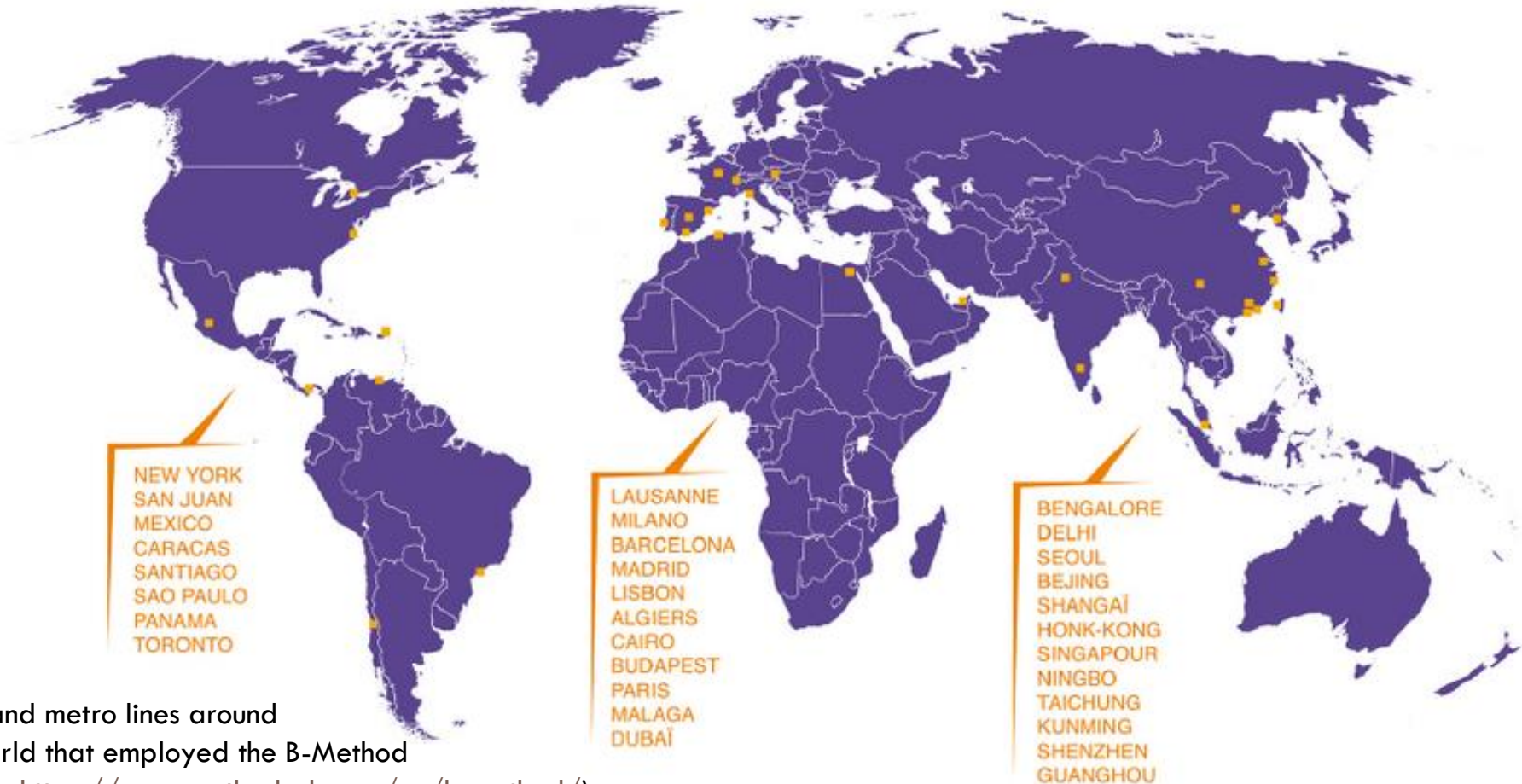
B-Method

- **Specification components = B-machines** (refinements, implementations)
 - ▣ **Concept:** close to class in OOP
 - ▣ **state variables + invariant** (restricts the variables) + **operations** (modify the variables) + **constants, sets ...**

- **Links**
 - ▣ <https://www.methode-b.com/en>
 - ▣ <https://www.clearsy.com/en/>
 - ▣ <https://www.atelierb.eu/en>

B-Method

- Industrial use
 - ▣ Frequently used in the railway domain
 - Including Budapest metro (line M4)



Train and metro lines around the world that employed the B-Method
(source: <https://www.methode-b.com/en/b-method/>)

TD/TS2JC Toolset

Train Director

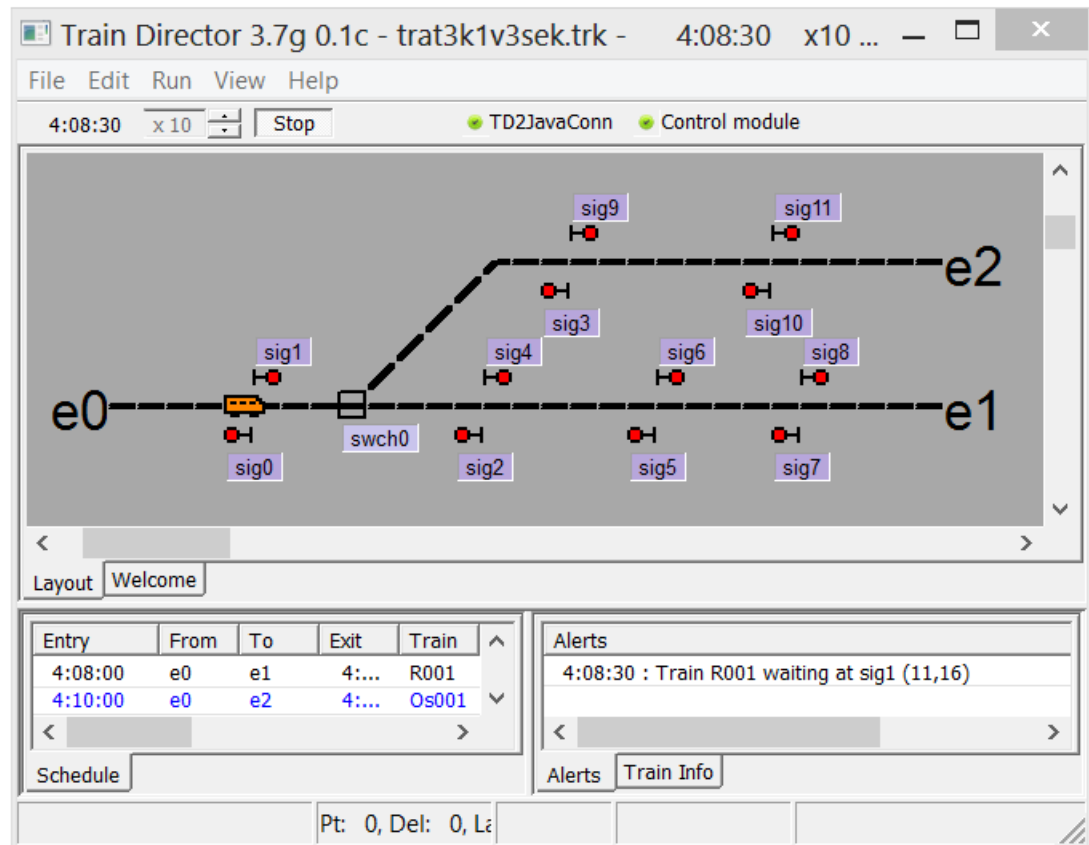
TS2JavaConn

How it works

Scenarios for the tutorial

Train Director

- Centralised traffic control simulator
- Modified by us
 - ▣ Enhanced communication interface
 - ▣ Internal logic disabled
- to create and simulate scenarios
 - ▣ layout + schedule



TS2JavaConn

- Communication between Train Director and control modules
- Generation of control module templates
 - ▣ Java, B language, Perfect

TS2JavaConn v. 1.24

File Setting Help

Train Director 3.7g 0.1c connected Control module connected

Signals	S M		Switches	Element state		Stations	S M		Track Sections
				S	M		S	M	
sig0	▶	▶	swch0	■	■	e0	▶	▶	section(e0, sig0) ●
sig1	▶	▶				e1	▶	▶	section(sig1, swch0) ●
sig2	▶	▶				e2	▶	▶	section(swch0, sig2) ●
sig3	▶	▶							section(swch0, sig3) ●
sig4	▶	▶							section(sig4, sig5) ●
sig5	▶	▶							section(sig9, sig10) ●
sig6	▶	▶							section(sig6, sig7) ●
sig7	▶	▶							section(sig11, e2) ●

Logger :

```
02.09.2014 at 06:43:44:112
-----
in: sectionEnter>s:4:16:10:17,
command recognised: sectionEnter (e0, sig0)
enter(e0_sig0)
result [e0 (RED), ]
out: multiCommand 4 16 C,
02.09.2014 at 06:43:45:200
-----
```

Train Director 3.7g 0.1c connected Control modul connected

Overview Generator

Output path : D:\korecko\projekty\2012_14_KEGA_fm\TD_TS2 change

Project dir : Module open

generate to : B Language

B Language

Content : no data with data

Type : not parametric parametric

MACHINE name : trat3k1v3sek

Config preview Generate module

Values overview

Sections

- e0_sig0
- sig1_swch0
- swch0_sig2
- swch0_sig3
- sig4_sig5
- sig9_sig10
- sig6_sig7
- sig11_e2
- sig8_e1

Control module

- reactive system
- Java application (developed using FM) + configuration file
- getters
 - ▣ signals, switches, (sections)
- methods responding to events from TD
 - requestEnter
 - requestGreen
 - sectionLeave
 - sectionEnter

TD/TS2JC: How it works

Train Director

The screenshot shows the Train Director 3.7g 0.1c interface. The main window displays a track layout with two main tracks, e0 and e1, and a branch track e2. A train (R001) is positioned on track e0, waiting at signal sig1. The interface includes a menu bar (File, Edit, Run, View, Help), a status bar (4:08:30, x10, Stop, TD2JavaConn, Control module), and a bottom panel with a Schedule table and Alerts section.

Entry	From	To	Exit	Train
4:08:00	e0	e1	4:...	R001
4:10:00	e0	e2	4:...	Os001

Alerts: 4:08:30 : Train R001 waiting at sig1 (11,16)

TS2JavaConn

The screenshot shows the TS2JavaConn v. 1.24 interface. It features a menu bar (File, Setting, Help) and a main window with a table of signals and switches. The table has columns for Signals, S, M, Switches, S, M, Stations, S, M, and Track Sections. The table is populated with data for various signals and switches.

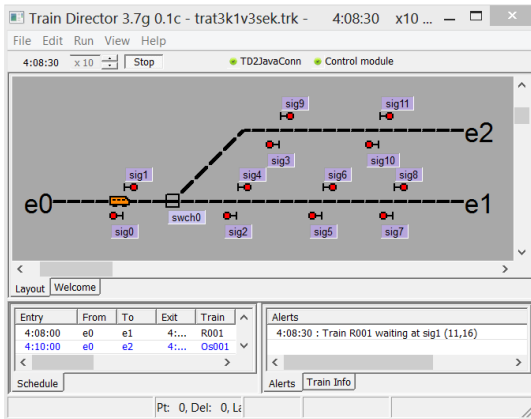
Signals	S	M	Switches	S	M	Stations	S	M	Track Sections
sig1	H	H	swch0	H	H	stn0	S	M	Edinburgh (sig1, swch0)
sig2	H	H	swch1	H	H	stn1	S	M	Edinburgh (sig2, swch1)
sig3	H	H	swch2	H	H	stn2	S	M	Edinburgh (sig3, swch2)
sig4	H	H	swch3	H	H	stn3	S	M	Edinburgh (sig4, swch3)
sig5	H	H	swch4	H	H	stn4	S	M	Edinburgh (sig5, swch4)
sig6	H	H	swch5	H	H	stn5	S	M	Edinburgh (sig6, swch5)
sig7	H	H	swch6	H	H	stn6	S	M	Edinburgh (sig7, swch6)
sig8	H	H	swch7	H	H	stn7	S	M	Edinburgh (sig8, swch7)
sig9	H	H	swch8	H	H	stn8	S	M	Edinburgh (sig9, swch8)
sig10	H	H	swch9	H	H	stn9	S	M	Edinburgh (sig10, swch9)
sig11	H	H	swch10	H	H	stn10	S	M	Edinburgh (sig11, swch10)

Control module

```
public class Controller3way3secTr {  
    ...  
    public Controller3way3secTr() {  
        this.trN = new CntrlSimpleTrack(1);  
        this.trS = new CntrlSimpleTrack(2);  
        this.cntrl3waytrack = new  
        Cntrl3WayTrack();  
    }  
    ...  
}
```

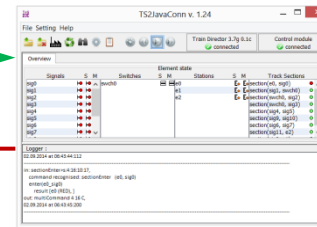
TD/TS2JC: How it works

Train Director



requestGreen
sig1, e1, R001

TS2JavaConn



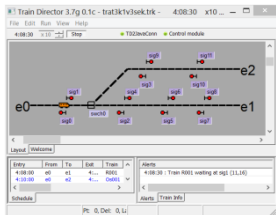
Control module

```
public class Controller3way3secTr {  
    ...  
    public Controller3way3secTr() {  
        this.trN = new CntrlSimpleTrack(1);  
        this.trS = new CntrlSimpleTrack(2);  
        this.cntrl3waytrack = new  
        Cntrl3WayTrack(); }  
    ...  
}
```

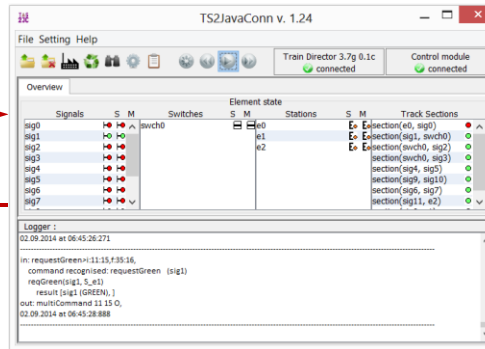
TD/TS2JC: How it works

Train Director

requestGreen
sig1, e1, R001



TS2JavaConn



reqGreen(sig1, S_e1)

Control module

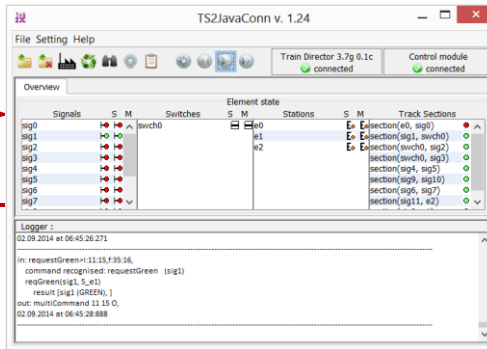
```
public class Controller3way3secTr {
    ...
    public Controller3way3secTr() {
        this.trN = new CntrlSimpleTrack(1);
        this.trS = new CntrlSimpleTrack(2);
        this.cntrl3waytrack = new
        Cntrl3WayTrack(); }
    ...
}
```

entryIndexStationName=S_ %name%
requestGreen=reqGreen
requestGreenParams=%SIGNALS%,%STATIONS.f%

TD/TS2JC: How it works



TS2JavaConn



reqGreen(sig1, S_e1)

1 ← getSig(sig1)

```

sigState=%int%
sigIndex=SIGNALS
getSignalNames=getSig
getSignalParams=%SIGNALS
    
```

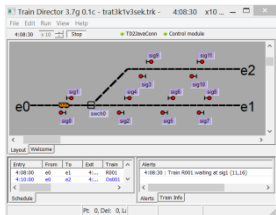
Control module

```

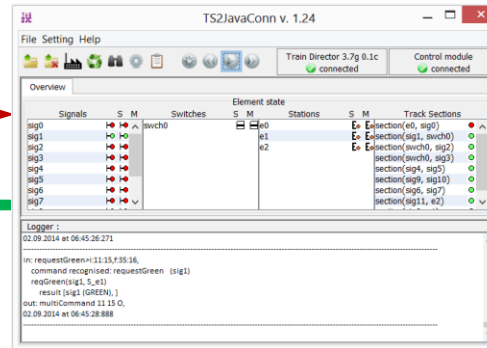
public class Controller3way3secTr {
    ...
    public Integer
    getSig(Controller3way3secTr.SIGNALS sg)
    ...
    public Integer
    getSwch(Controller3way3secTr.SWITCHES sw)
    ...
    public void
    reqGreen(Controller3way3secTr.SIGNALS sg_data,
    Controller3way3secTr.STATIONS st) {
    
```

TD/TS2JC: How it works

Train Director



TS2JavaConn



Control module

```
public class Controller3way3secTr {
    ...
    public Controller3way3secTr() {
        this.trN = new CntrlSimpleTrack(1);
        this.trS = new CntrlSimpleTrack(2);
        this.cntrl3waytrack = new
        Cntrl3WayTrack(); }
    ...
}
```

multiCommand
11 15 0

1 ← getSig(sig1)

sigState=%int%
signalGreenState=1
sigIndex=SIGNALS

TD/TS2JC: How it works

Train Director

Train Director 3.7g 0.1c - trat3k1v3sek.trk - 4:08:40 x10 ...

File Edit Run View Help

4:08:40 x10 Stop TD2JavaConn Control module simulation restarted.

Layout Welcome

Entry	From	To	Exit	Train
4:08:00	e0	e1	4:...	R001
4:10:00	e0	e2	4:...	Os001

Schedule Alerts Train Info

Pt: 0, Del: 0, Lz:

TS2JavaConn

TS2JavaConn v. 1.24

Signal	S	M	Switch	S	M	Station	S	M	Track Section
sig0	H	H	swch0	H	H				E: section(e0, sig0)
sig1	H	H		H	H				E: section(e1, sig1)
sig2	H	H		H	H				E: section(e1, sig2)
sig3	H	H		H	H				E: section(e2, sig3)
sig4	H	H		H	H				E: section(e2, sig4)
sig5	H	H		H	H				E: section(e2, sig5)
sig6	H	H		H	H				E: section(e2, sig6)
sig7	H	H		H	H				E: section(e2, sig7)
sig8	H	H		H	H				E: section(e2, sig8)
sig9	H	H		H	H				E: section(e2, sig9)
sig10	H	H		H	H				E: section(e2, sig10)
sig11	H	H		H	H				E: section(e2, sig11)

Control module

```
public class Controller3way3secTr {  
    ...  
    public Controller3way3secTr() {  
        this.trN = new CntrlSimpleTrack(1);  
        this.trS = new CntrlSimpleTrack(3);  
        this.cntrl3waytrack = new  
            Cntrl3WayTrack();  
    }  
    ...  
}
```

TD/TS2JC: How it works

Train Director

Train Director 3.7g 0.1c - trat3k1v3sek.trk - 4:08:50 x10 ...

File Edit Run View Help

4:08:50 x10 Stop TD2JavaConn Control module Simulation restarted.

Layout Welcome

Entry	From	To	Exit	Train
4:08:00	e0	e1	4:...	R001
4:10:00	e0	e2	4:...	Os001

Schedule Alerts Train Info

Pt: -1, Del: 0, L:

TS2JavaConn

TS2JavaConn v. 1.24

File Setting Help

Train Director 3.7g 0.1c Control module
kg connected kg connected

Overview

Signals	S	M	Switches	S	M	Element state	S	M	Track Sections
sig1	→	→	swch0	→	→	...	E	E	Section(e0, sig1)
sig2	→	→				...	E	E	Section(e0, sig2)
sig3	→	→				...	E	E	Section(e1, sig3)
sig4	→	→				...	E	E	Section(e1, sig4)
sig5	→	→				...	E	E	Section(e1, sig5)
sig6	→	→				...	E	E	Section(e1, sig6)
sig7	→	→				...	E	E	Section(e1, sig7)
sig8	→	→				...	E	E	Section(e1, sig8)
sig9	→	→				...	E	E	Section(e2, sig9)
sig10	→	→				...	E	E	Section(e2, sig10)
sig11	→	→				...	E	E	Section(e2, sig11)

Logfile

Control module

```
public class Controller3way3secTr {  
    ...  
    public Controller3way3secTr() {  
        this.trN = new CntrlSimpleTrack(1);  
        this.trS = new CntrlSimpleTrack(3);  
        this.cntrl3waytrack = new  
            Cntrl3WayTrack();  
    }  
    ...  
}
```

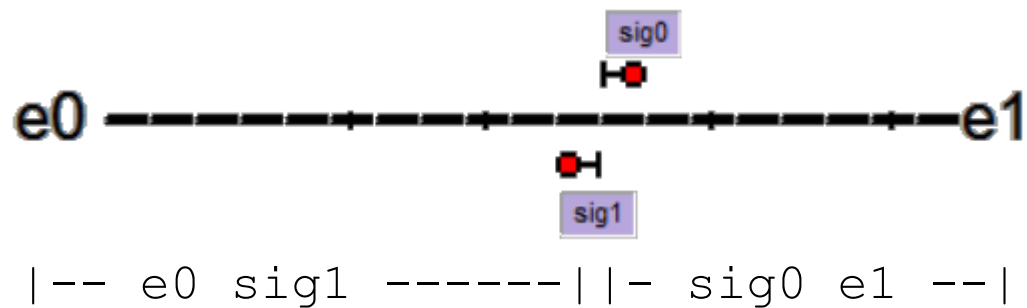
More on TS2JavaConn / TD / OR

- Korečko, Š., Sorád, J. (2014). Using simulation games in teaching formal methods for software development. *Innovative Teaching Strategies and New Learning Paradigms in Computer Programming*, in *Innovative Teaching Strategies and New Learning Paradigms in Computer Programming*, R. Queirós, Ed., IGI Global, 2015, pp. 106–130. (draft version available [here](#)).
- Korečko, Š., Sorád, J., DudlÁková, Z., Sobota, B. (2014, September). A toolset for support of teaching formal software development. In *International Conference on Software Engineering and Formal Methods* (pp. 278-283). Springer
- Korečko, Š., Sobota, B. (2017). Computer Games as Virtual Environments for Safety-Critical Software Validation. *Journal of Information and Organizational Sciences*, 41(2), 197-212.

Our scenarios

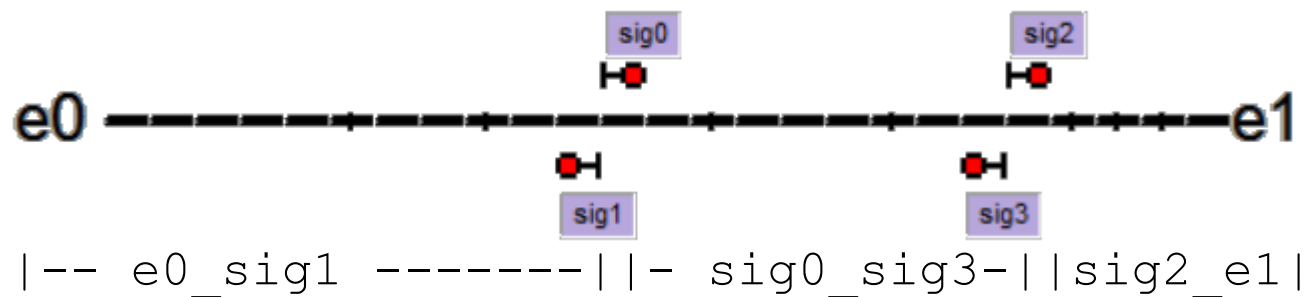
EXAMPLES

- route2sec and route2sec_deadlck



PRACTICE

- route3sec and route3sec_deadlck



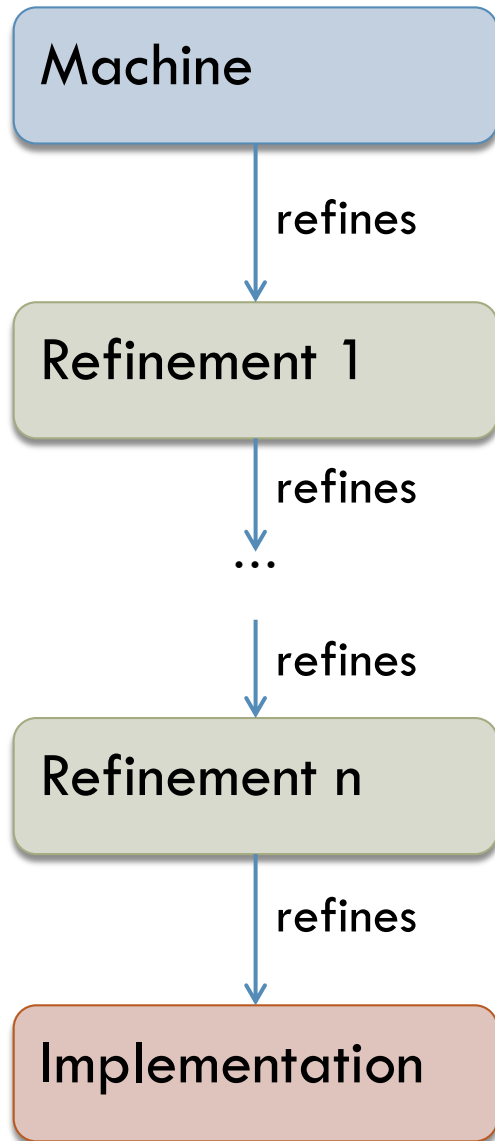
Formal Specification and Verification in B-Method

Development process

B-language

B-machine

Development in B-method



B-language

- Expressions and Predicates
 - ▣ mathematical notation
 - ▣ based on first-order logic and Zermelo-Fraenkel set theory
- Operations
 - ▣ Generalized Substitution Language (GSL)
 - ▣ based on E.W. Dijkstra's Guarded commands and Weakest Precondition Calculus
- Documentation supplied with Atelier-B
 - ▣ **B Language Reference Manual**
 - Detailed description
 - ▣ B Language Keywords and Operators (**B Symbols**)
 - ASCII vs. mathematical notation

GS :: Syntax

GS	meaning of GS
skip	Empty GS (do nothing).
$x := e$	Assignment of value of expression e to variable x .
$S_1 ; S_2$	Sequential composition (do GS S_1 , then GS S_2).
$S_1 \parallel S_2$	Parallel composition (do S_1 and S_2 at once).
PRE E THEN S_1 END	If predicate E holds, do S_1 . Otherwise, do anything.
SELECT E THEN S_1 END	If E holds, do S_1 . Otherwise, do not execute.
IF E THEN S_1 ELSE S_2 END	If E holds, do S_1 . Otherwise, do S_2 .

(selected)

GS :: Semantics

<code>[skip]P</code>	$= P$
<code>[x := e]P</code>	$= P[x:=e]$
<code>[PRE E THEN S END]P</code>	$= E \ \& \ [S]P$
<code>[SELECT E THEN S END]P</code>	$= E \Rightarrow [S]P$
<code>[S₁ ; S₂]P</code>	$= [S_1][S_2]P$
<code>[IF E THEN S₁ ELSE S₂ END]P</code>	$= (E \Rightarrow [S_1]P) \ \& \ (\text{not}(E) \Rightarrow [S_2]P)$

(selected)

GS :: Semantics of Simultaneous Substitution

skip T	= T
x := E y := F	= x, y := E, F
(PRE P THEN S END) T	= PRE P THEN S T END
(SELECT E THEN S END) T	= SELECT E THEN S T END (*)
(IF E THEN S ₁ ELSE S ₂ END) T	= IF E THEN S ₁ T ELSE S ₂ T END (**)

(*) holds if S always terminates.

(*) holds if S₁ and S₂ always terminate.

(selected)

B-machine :: Syntax

```
MACHINE M(p)
CONSTRAINTS C
SETS St
CONSTANTS k
PROPERTIES Bh
VARIABLES v
DEFINITIONS D
INVARIANT I
INITIALISATION T
OPERATIONS
  y ← op(x) =
    PRE P THEN S END
...
END
```

- VARIABLES is
ABSTRACT_VARIABLES
 - ▣ we can also have CONCRETE_VARIABLES
- CONSTANTS is
CONCRETE_CONSTANTS
 - ▣ we can also have ABSTRACT_CONSTANTS

Example 1: B-machine route2sec

MACHINE route2sec

SETS

```
PROP_SIGNAL={green, red};  
PROP_SECTION={free,occupy}
```

CONCRETE_VARIABLES

```
e0, e1, /*entry points*/  
sig0, sig1, /*signals*/  
e0_sig1, sig0_e1 /*track sections*/
```

INVARIANT

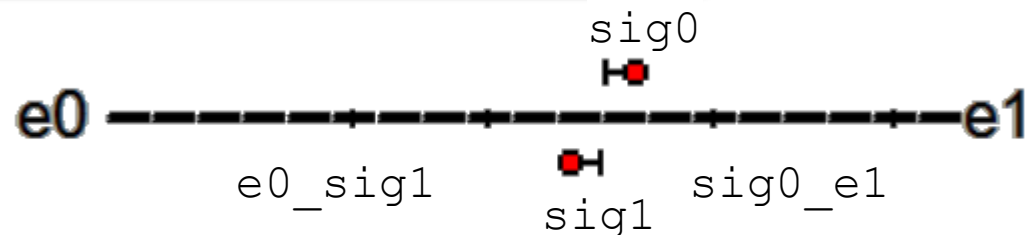
```
e0:PROP_SIGNAL & e1:PROP_SIGNAL &  
...  
& (e0=green => sig1=red)  
& (sig1=green => e0=red)  
& (e1=green => sig0=red)  
& (sig0=green => e1=red)  
& (e0=green => e0_sig1=free)  
...
```

INITIALISATION

```
e0:=red || ... sig1:=red ||...  
e0_sig1:= free || sig0_e1:= free
```

OPERATIONS

```
ss <-- getSig_sig0 = BEGIN ss:=sig0  
END;  
...  
reqGreen_e0 =  
  IF sig1=red & e0_sig1=free  
  THEN e0:=green END;  
enterNI_e0_sig1 =  
  BEGIN e0_sig1:=occupy || e0:=red ||  
  sig1:=red END;  
leaveNI_e0_sig1 =  
  BEGIN e0_sig1:=free END;  
END
```



B-machine :: Proof Obligations

MACHINE $M(p)$
CONSTRAINTS C
SETS S_t
CONSTANTS k
PROPERTIES B_h
VARIABLES v
DEFINITIONS D
INVARIANT I
INITIALISATION T
OPERATIONS
 $y \leftarrow \text{op}(x) =$
 PRE P THEN S END
 ...
END

- Initialisation establishes the invariant
 - $(C \ \& \ B_h) \Rightarrow [T] \ I$
- Each operation preserves the invariant
 - $(C \ \& \ B_h \ \& \ I \ \& \ P) \Rightarrow [S] \ I$

(selected, simplified)

Example 2: Machine Verification

Invariant preservation by **reqGreen_e0**

MACHINE route2sec

...

CONCRETE_VARIABLES e0, sig1, e0_sig1 ...

INVARIANT

...

e0=green => e0_sig1=free

...

INITIALISATION

e0:=red || ... sig1:=red ||...

OPERATIONS

...

reqGreen_e0 =

**IF sig1=red & e0_sig1=free
THEN e0:=green END;**

(demonstrated only for the shown part of the invariant)

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

$(e0=green \Rightarrow e0_sig1=free) \Rightarrow$
[IF sig1=red & e0_sig1=free
THEN e0:=green END]
 $(e0=green \Rightarrow e0_sig1=free)$



$(C \ \& \ Bh \ \& \ I \ \& \ P) \Rightarrow [S] \ I$

$(I \ \& \ P) \Rightarrow [S] \ I$

MACHINE route2sec

...
CONCRETE_VARIABLES e0, sig1, e0_sig1 ...

INVARIANT

...
 $e0=green \Rightarrow e0_sig1=free$

INITIALISATION

e0:=red || ... sig1:=red || ...

OPERATIONS

...
reqGreen_e0 =

IF sig1=red & e0_sig1=free
THEN e0:=green **END**;

(demonstrated only for the shown part of the invariant)

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

```
(e0=green => e0_sig1=free) =>  
[ IF sig1=red & e0_sig1=free  
  THEN e0:=green END ]  
(e0=green => e0_sig1=free)
```



```
(e0=green => e0_sig1=free) =>(   
  ((sig1=red & e0_sig1=free) =>  
    [ e0:=green ] (e0=green => e0_sig1=free))  
  &  
  ((not(sig1=red & e0_sig1=free) =>  
    [skip] (e0=green => e0_sig1=free))  
  )  
)
```

MACHINE route2sec

```
...  
CONCRETE_VARIABLES e0, sig1, e0_sig1 ...
```

INVARIANT

```
...  
  e0=green => e0_sig1=free  
...
```

INITIALISATION

```
e0:=red || ... sig1:=red || ...
```

OPERATIONS

```
...  
reqGreen_e0 =
```

```
  IF sig1=red & e0_sig1=free  
  THEN e0:=green END;
```

```
[IF E THEN S1 ELSE S2 END ]P=  
(E => [S1]P) & (not(E) => [S2]P)
```

(demonstrated only for the shown part of the invariant)

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

$(e0=green \Rightarrow e0_sig1=free) \Rightarrow$
[IF sig1=red & e0_sig1=free
THEN e0:=green END]
 $(e0=green \Rightarrow e0_sig1=free)$



$(e0=green \Rightarrow e0_sig1=free) \Rightarrow$
 $((sig1=red \ \& \ e0_sig1=free) \Rightarrow$
[e0:=green] $(e0=green \Rightarrow e0_sig1=free))$
&
 $((not(sig1=red \ \& \ e0_sig1=free) \Rightarrow$
[skip] $(e0=green \Rightarrow e0_sig1=free))$
)

$[x := e]P = P[x:=e]$
 $[skip]P = P$



$(e0=green \Rightarrow e0_sig1=free) \Rightarrow$
 $((sig1=red \ \& \ e0_sig1=free) \Rightarrow$
 $(green =green \Rightarrow e0_sig1=free))$
&
 $((not(sig1=red \ \& \ e0_sig1=free) \Rightarrow$
 $(e0=green \Rightarrow e0_sig1=free))$
)

(demonstrated only for the shown part of the invariant)

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

$(a \Rightarrow (b \ \& \ c)) \Leftrightarrow ((a \Rightarrow b) \ \& \ (a \Rightarrow c))$
 $(\text{true} \Rightarrow b) \Leftrightarrow b$
 $(a \Rightarrow (b \Rightarrow c)) \Leftrightarrow ((a \ \& \ b) \Rightarrow c)$

```
(e0=green => e0_sig1=free) =>(
  ((sig1=red & e0_sig1=free) =>
    [ e0:=green ] (e0=green => e0_sig1=free))
  &
  ((not(sig1=red & e0_sig1=free) =>
    [skip] (e0=green => e0_sig1=free))
  )
)
```



```
(e0=green => e0_sig1=free) =>(
  ((sig1=red & e0_sig1=free) =>
    (green =green => e0_sig1=free))
  &
  ((not(sig1=red & e0_sig1=free) =>
    (e0=green => e0_sig1=free))
  )
)
```

(demonstrated only for the shown part of the invariant)

```
( (e0=green => e0_sig1=free) =>
  ((sig1=red & e0_sig1=free) =>
    (e0_sig1=free))
)
&
( ((e0=green => e0_sig1=free) &
  not(sig1=red & e0_sig1=free)) =>
  (e0=green => e0_sig1=free)
)
```

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

```
( (e0=green => e0_sig1=free) =>  
  true  
)  
&  
true
```

(a & b) => b



```
( (e0=green => e0_sig1=free) =>  
  ((sig1=red & e0_sig1=free) =>  
    e0_sig1=free))  
)  
&  
( ((e0=green => e0_sig1=free) &  
  not(sig1=red & e0_sig1=free)) =>  
  (e0=green => e0_sig1=free))  
)
```

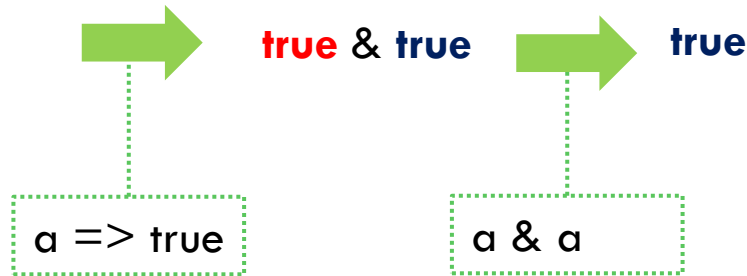
```
(e0=green => e0_sig1=free) =>( (sig1=red & e0_sig1=free) =>  
  (green =green => e0_sig1=free))  
&  
((not(sig1=red & e0_sig1=free) =>  
  (e0=green => e0_sig1=free))  
)
```

(demonstrated only for the shown part of the invariant)

Example 2: Machine Verification

Invariant preservation by reqGreen_e0

```
( (e0=green => e0_sig1=free) =>  
  true  
)  
&  
true
```



```
( (e0=green => e0_sig1=free) =>  
  ((sig1=red & e0_sig1=free) =>  
    e0_sig1=free))  
)  
&  
( ((e0=green => e0_sig1=free) &  
  not(sig1=red & e0_sig1=free)) =>  
  (e0=green => e0_sig1=free)  
)
```



```
(e0=green => e0_sig1=free) =>( (sig1=red & e0_sig1=free) =>  
  (green =green => e0_sig1=free))  
&  
((not(sig1=red & e0_sig1=free) =>  
  (e0=green => e0_sig1=free))  
)
```

(demonstrated only for the shown part of the invariant)

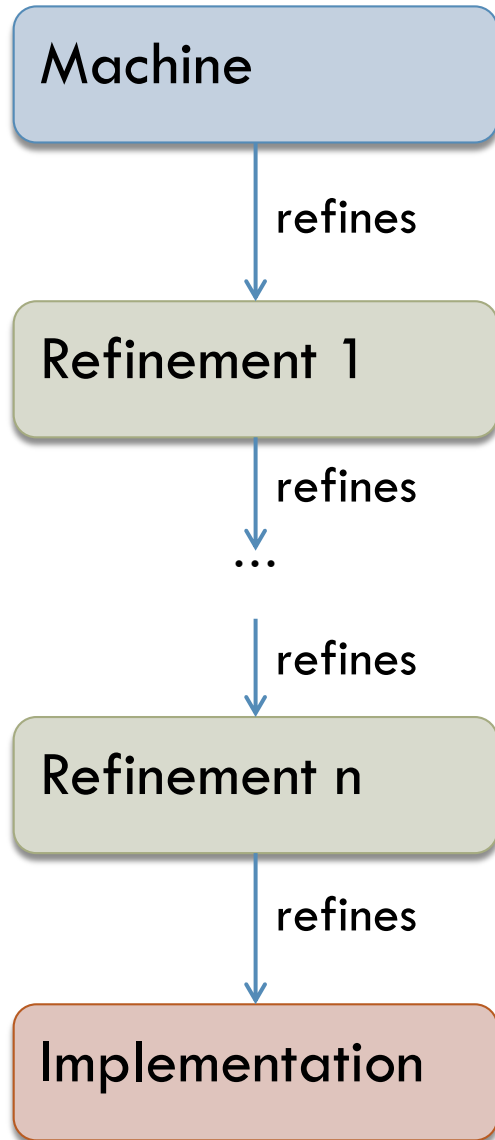
Verified Refinement

Implementation

GS allowed in components

Refinement proof obligations

From machine to Implementation



From machine to Implementation

```
MACHINE M(p)
CONSTRAINTS C
SETS St
CONSTANTS k
PROPERTIES Bh
VARIABLES v
DEFINITIONS D
INVARIANT I
INITIALISATION T
OPERATIONS
  y ← op(x) =
    PRE P THEN S END
  ...
END
```

refines

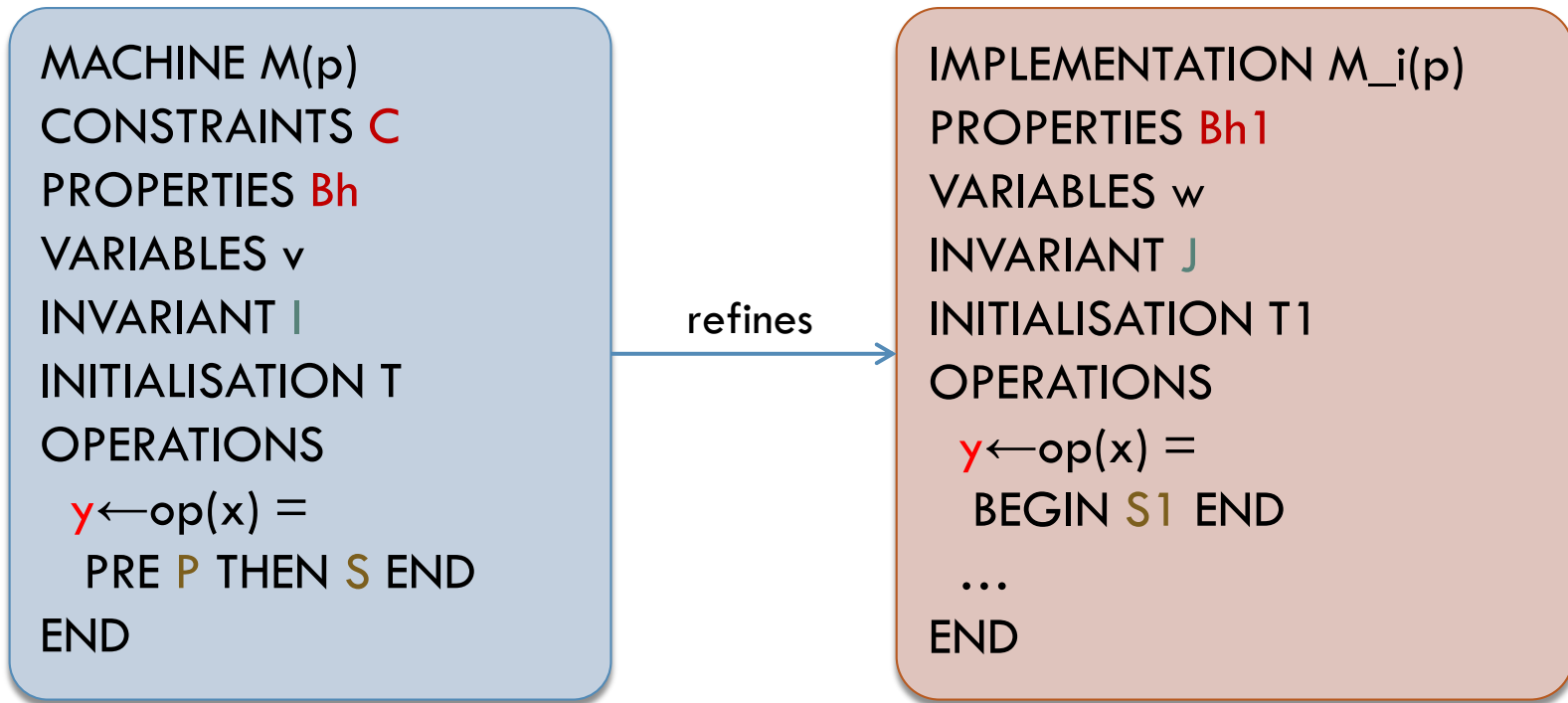
```
IMPLEMENTATION Mi(p)
REFINES M
SETS St1
CONCRETE_CONSTANTS k1
VALUES vk1
PROPERTIES Bh1
VARIABLES w
DEFINITIONS D1
INVARIANT J
INITIALISATION T1
OPERATIONS
  y ← op(x) =
    BEGIN S1 END
  ...
END
```

GS in components

GS	Machine	Refinement	Implementation
skip	yes	yes	yes
$x := e$	yes	yes	yes
$S_1 ; S_2$	no	yes	yes
$S_1 \parallel S_2$	yes	yes	no
PRE E THEN S_1 END	yes	yes	no
SELECT E THEN S_1 END	yes	yes	no
IF E THEN S_1 ELSE S_2 END	yes	yes	yes
WHILE E DO S ... END	no	no	yes

(selected)

Refinement Proof Obligations



- Initialisation establishes the invariant
 - ▣ $(C \ \& \ Bh \ \& \ Bh1) \Rightarrow [T1] (\text{not}([T] \ \text{not}(J)))$
- Each operation preserves the invariant
 - ▣ $(C \ \& \ Bh \ \& \ Bh1 \ \& \ I \ \& \ J \ \& \ P) \Rightarrow [S1'] (\text{not}([S] \ \text{not}(J \ \wedge \ y' = y)))$

(selected, simplified)

Example 3: Machine Refinement

route2sec.mch **refines** to route2sec_i.imp.

MACHINE route2sec

SETS

```
PROP_SIGNAL={green, red};  
PROP_SECTION={free, occup}
```

CONCRETE_VARIABLES

```
e0, e1, ...
```

INVARIANT

```
e0:PROP_SIGNAL & e1:PROP_SIGNAL &
```

```
...
```

INITIALISATION

```
e0:=red || ... sig1:=red || ...
```

OPERATIONS

```
ss <-- getSig_sig0 = BEGIN ss:=sig0
```

```
END;
```

```
...
```

```
reqGreen_e0 =
```

```
IF sig1=red & e0_sig1=free  
THEN e0:=green END;
```

refines



IMPLEMENTATION route2sec_i

REFINES route2sec

INITIALISATION

```
e0:=red ; ... sig1:=red ; ...
```

OPERATIONS

```
ss <-- getSig_sig0 = BEGIN ss:=sig0
```

```
END;
```

```
...
```

```
reqGreen_e0 =
```

```
IF sig1=red & e0_sig1=free  
THEN e0:=green END;
```

Practice

Downloads

Abstract Specification

Refinement

Code generation

Deployment

Downloads

□ Tutorial package

□ <http://hron.fei.tuke.sk/~korecko/cefp19/cefp19BmethodPack.zip>

□ Atelier B

□ <http://www.atelierb.eu/en/download/>

□ Java JDK

□ <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

□ TS2JavaConn/TrainDirector Toolset

□ <http://hron.fei.tuke.sk/~korecko/FMInGamesExp/resources/allInOneTDTS2J.zip>

T.0: Set up Atelier B

1. Unpack the tutorial package to C : /VSD
 - ▣ <http://hron.fei.tuke.sk/~korecko/cefp19/cefp19BmethodPack.zip>
2. Run Atelier B
 - ▣ Don't mind the warning displayed
3. Create a new workspace
 - ▣ *Name:* cefp19
 - ▣ ***Workspace database directory:* C : /VSD/bdb**
4. Set the *Default project directory* to C : /VSD/Bprojects

T.1: Atelier B project for route2sec

1. Run Atelier B.
2. Create a new project called `route2sec`.
 - ▣ type: software development
3. In *Project – Properties – software development* set *Generator to Legacy*.
4. To the project, add the components
 - ▣ `C:/VSD/Bprojects/route2sec.mch` and
 - ▣ `C:/VSD/Bprojects/route2sec_i.imp`
5. Check and prove both components.

T.2: From .imp to .java with BKPICompiler

1. Run BKPICompiler
 - ▣ `C:/VSD/BKPICompiler/BKPIcompiler.jar`
2. Right click on `route2sec` and choose *Generate Java code*
3. Delete `MainClass.java`
4. Compile `route2sec.java`
 - ▣ Just run `compile.bat`
 - ▣ Alternatively, you can use an online compiler
 - e.g. <https://www.compilejava.net/>

T.3: route2sec in TD/TS2JC

1. Run *Train Director*

- ▣ `C:/VSD/TrainDirector/trainindir3.exe`
- ▣ If it doesn't show up, go to its *Properties* and set it to run in the *Windows 7 compatibility mode*

1. In *Train Director*, open the file `C:/VSD/scenarios/route2sec.trk`

2. Run *TS2JavaConn*

- ▣ `C:/VSD/TS2JavaConn/TS2JavaConn.jar`

3. In *TS2JavaConn*, load the module `C:/VSD/Bprojects/route2sec/java/route2sec.class`

4. Start the simulation (in TD or TS2JC).

T.4: Machine Specification for route3sec

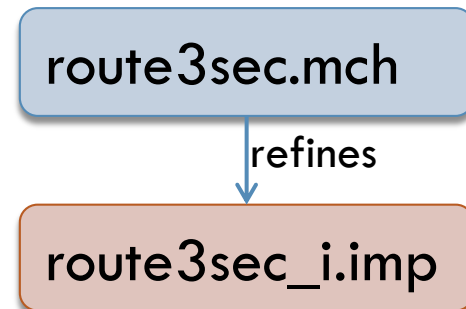
1. In Atelier B, create a new project called `route3sec` and add `route3sec.mch` to it.
 - ▣ In the same way as `route2sec` (T.1)
2. Finish `route3sec.mch` according to the TODO comments inside it.
3. Prove `route3sec.mch`.

route3sec.mch

T.5: route3sec Machine Refinement

1. In the project `route3sec` in Atelier B, create a new implementation component called `route3sec_i.imp`
2. Write code to `route3sec_i.imp`.
 - ▣ Hint: copy the code from `route3sec.mch` and modify it.
3. Verify `route3sec_i.imp`.
4. Translate `route3sec` to Java with the BKPI Compiler
5. Compile `route3sec.java`
6. Try the compiled module with TD/TS2JC
 - ▣ TD scenario is
`C:/VSD/scenarios/route3sec.trk`

(In the same way as `route2sec` (T.2, T.3))



T.6 (optional): Deadlock Free route3sec

1. Specify and verify a deadlock free version of the previous controller for the scenario `route3sec_deadlck`.
 - ▣ Don't forget to add additional safety properties
2. Refine it to an implementation and verify the implementation.
3. Compile the implementation with BKPI Compiler and try with TrainDirector/TS2JavaConn

Thanks ... and fill in the questionnaire

- Thank you for your participation.
 - ▣ I hope you enjoyed it.
- Please, fill in the questionnaire
 - ▣ https://drive.google.com/open?id=19jpdR0PS-Mkq1x5_U1EfVaPGZivjNyBq4t0FVcpWHQ