Dr. Wolfgang Emmerich
Dept. of Computer Science
University College London
Pearson Building Room 402

6-Feb-98

C340 Concurrency

Tutorial Session 2 - Work Sheet

This lab session will give you practical experience with safety analysis and property specification.

Exercise 1:
A small country road goes over a wooden bridge that connects two shores of a creek. The bridge is rather narrow and cars can only go north-bound or south bound. Given that it is constructed out of wood, the bridge can only carry five cars at a time. Model the bridge in FSP.

Exercise 2:
Demonstrate that your bridge model is safe. To do so, formulate the following safety properties in FSP:

- The bridge does not allow more than five cars onto the bridge at any one time
- Cars are either going south or north but there are never two cars on the bridge that go in opposite direction.

Exercise 3:
How do you use the safety properties in an implementation of the bridge model?

Dr. Wolfgang Emmerich
Dept. of Computer Science
University College London
Pearson Building Room 402

6-Feb-98

C340 Concurrency

Tutorial1 - Answer Sheet

Exercise 1:

```
const BRIDGE_CAPACITY=5
range T=0..BRIDGE_CAPACITY

DIRECTION=DIRECTION[0],
DIRECTION[i:T]=(when (i<BRIDGE_CAPACITY) enter->DIRECTION[i+1]
                |when (i>0)leave->DIRECTION[i-1]
                |going[i]->DIRECTION[i]
).

BRIDGE_CONTROLER=(south.going[s:T] ->
                  north.going[n:T] -> (
                      when (n==0) south.enter -> BRIDGE_CONTROLER
                     |when (s==0) north.enter -> BRIDGE_CONTROLER)
                 )+{south.enter,south.going[T],
                    north.enter,north.going[T]}.

||BRIDGE = (north:DIRECTION || south:DIRECTION || BRIDGE_CONTROLER ).
```

Exercise 2:

```
BRIDGE_DIRECTION = (south.going[s:T]-> north.going[n:T] ->
                    (when (s>0 && n>0) unsafe -> ERROR
                    |when (s==0 || n==0) safe ->BRIDGE_DIRECTION)).

BRIDGE_WEIGHT = (south.going[s:T]-> north.going[n:T] ->
                 (when (s+n>BRIDGE_CAPACITY) unsafe -> ERROR
                 |when (s+n<=BRIDGE_CAPACITY) safe ->BRIDGE_WEIGHT)).

||BRIDGE_CHECK = (BRIDGE_DIRECTION || BRIDGE_WEIGHT || BRIDGE).
```

Exercise 3:

The `BRIDGE` would become a monitor class and the safety properties determine the monitor invariants.