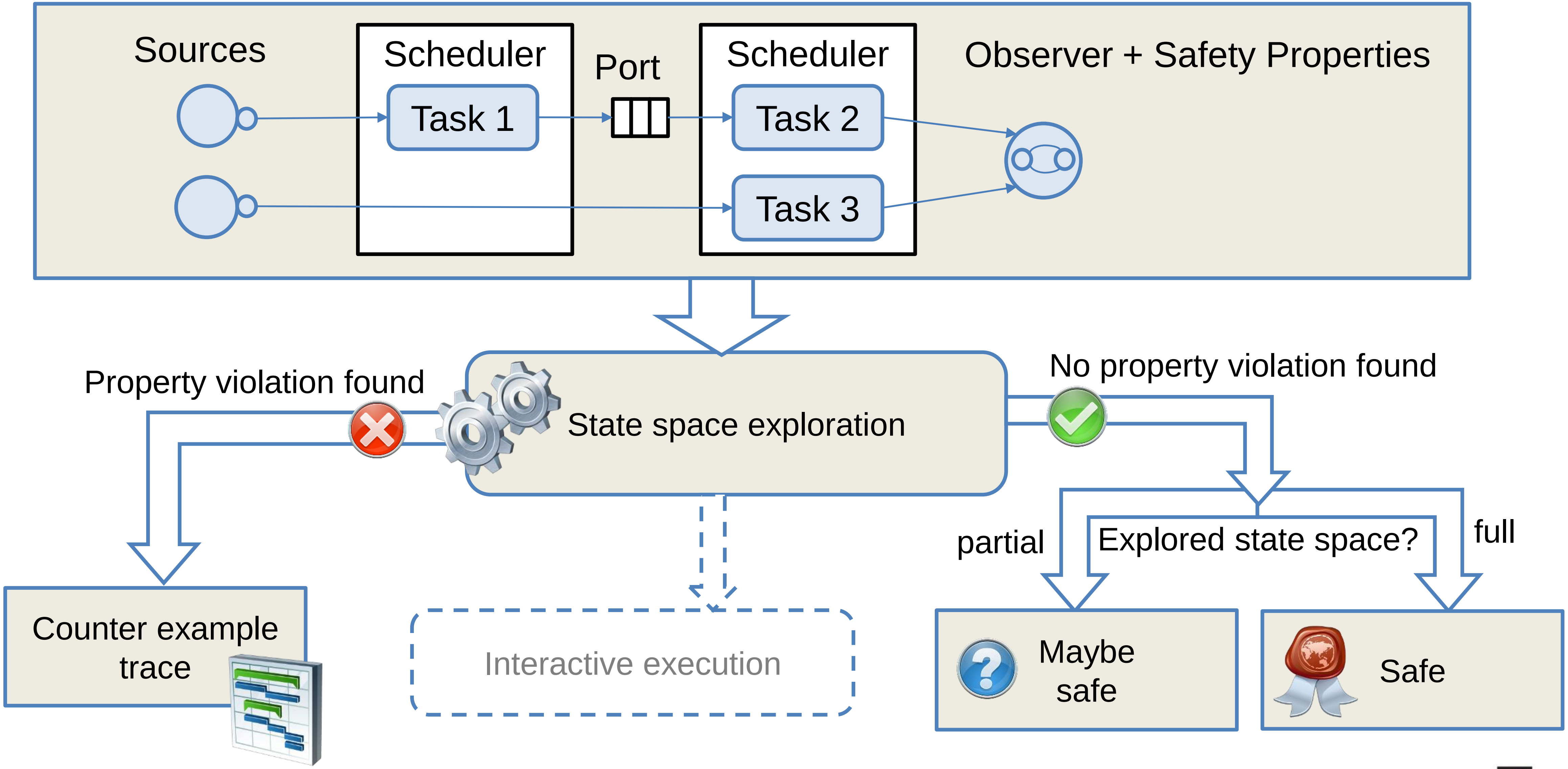
AMALTHEA Timing Analyses with RTana2sim

2021-01-20

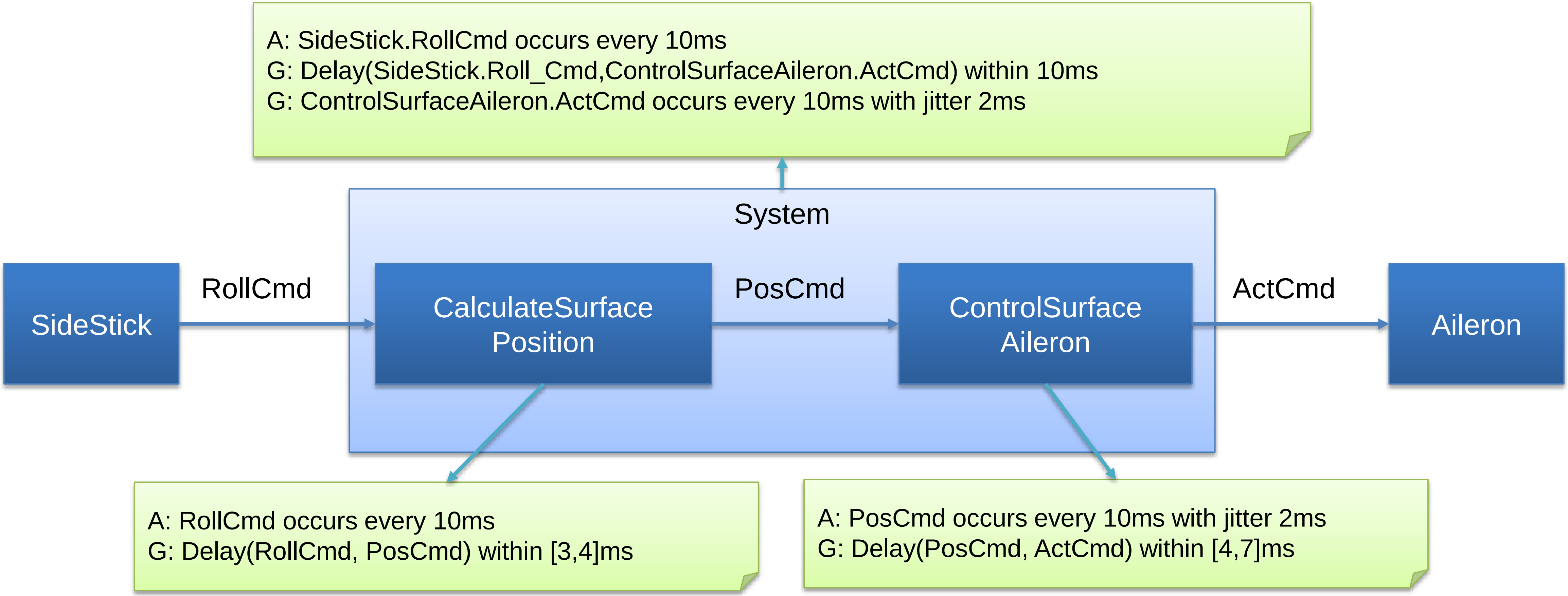
Jan Steffen Becker, Björn Koopmann, Ingo Stierand

OFFIS

RTana2sim Model Checker



Application 1: Virtual Integration



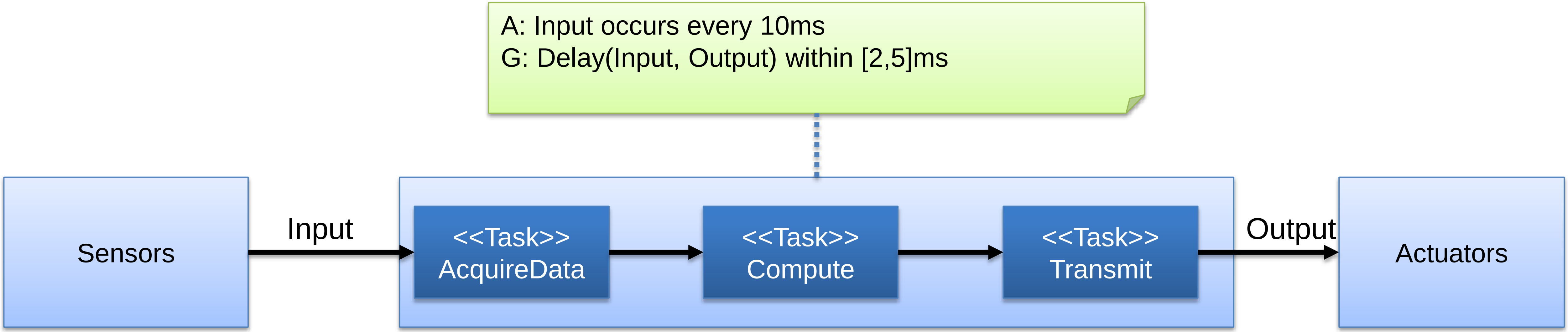
A: SideStick.RollCmd occurs every 10ms
G: Delay(SideStick.Roll_Cmd,ControlSurfaceAileron.ActCmd) within 10ms
G: ControlSurfaceAileron.ActCmd occurs every 10ms with jitter 2ms

A: RollCmd occurs every 10ms
G: Delay(RollCmd, PosCmd) within [3,4]ms

A: PosCmd occurs every 10ms with jitter 2ms
G: Delay(PosCmd, ActCmd) within [4,7]ms

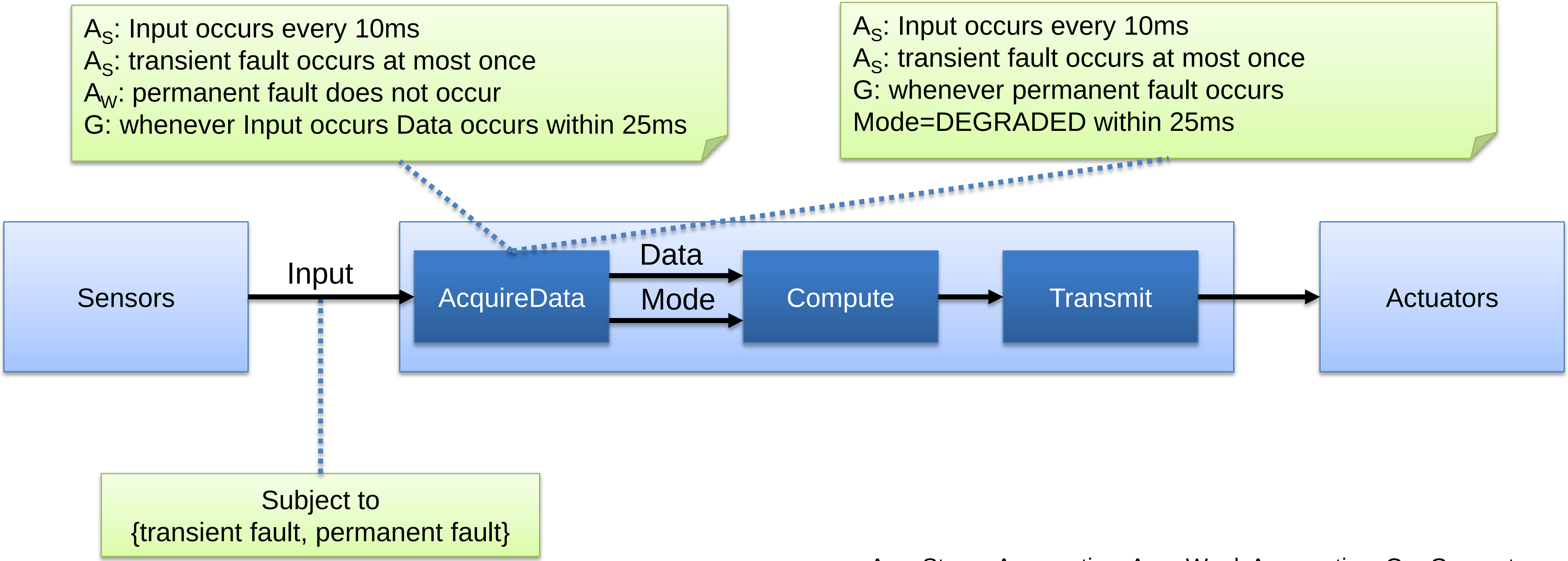
A = Assumption, G = Guarantee

Application 2a: Timing Analysis of Software Tasks (Chains)



A = Assumption, G = Guarantee

Application 2b: Timing Analysis of Software Safety Mechanisms



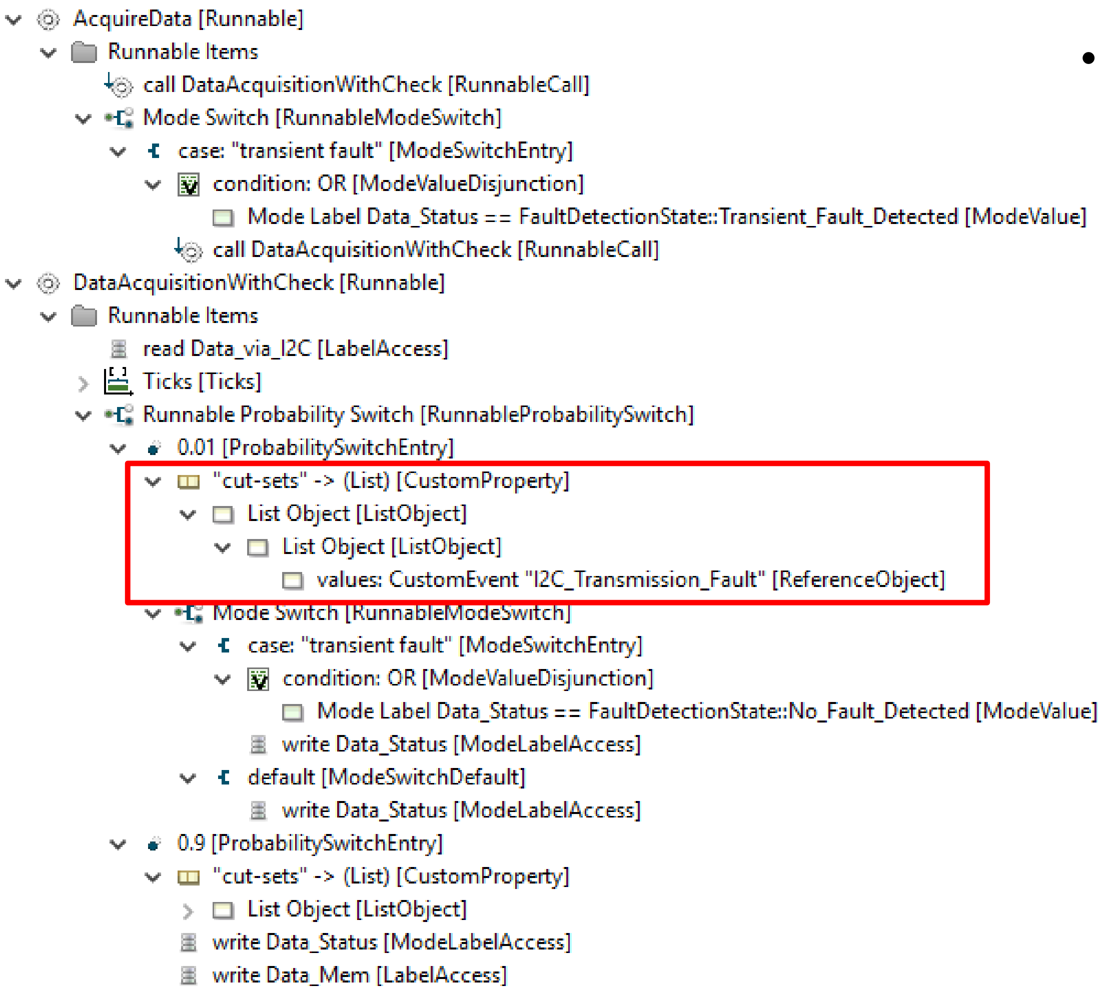
A_S: Input occurs every 10ms
A_S: transient fault occurs at most once
A_W: permanent fault does not occur
G: whenever Input occurs Data occurs within 25ms

A_S: Input occurs every 10ms
A_S: transient fault occurs at most once
G: whenever permanent fault occurs
Mode=DEGRADED within 25ms

Subject to
{transient fault, permanent fault}

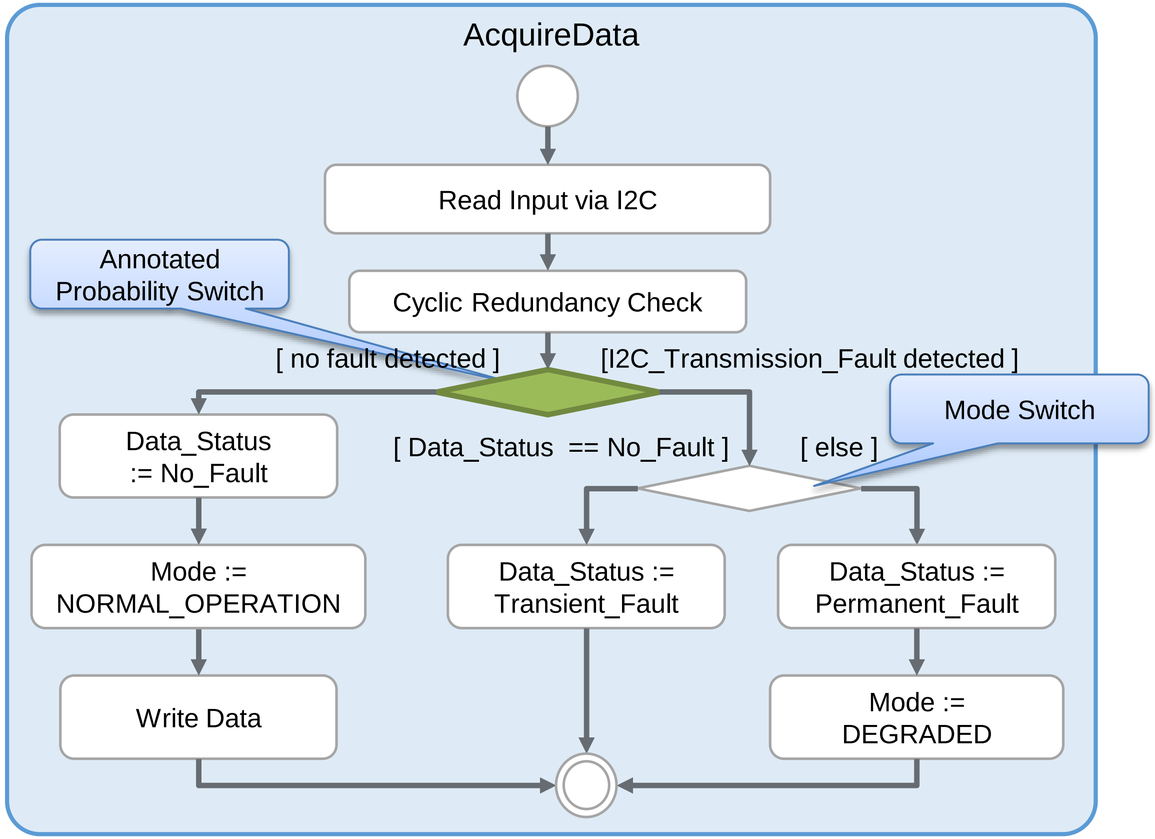
A_S = Strong Assumption, A_W = Weak Assumption, G = Guarantee

Software Safety Mechanism Modeling

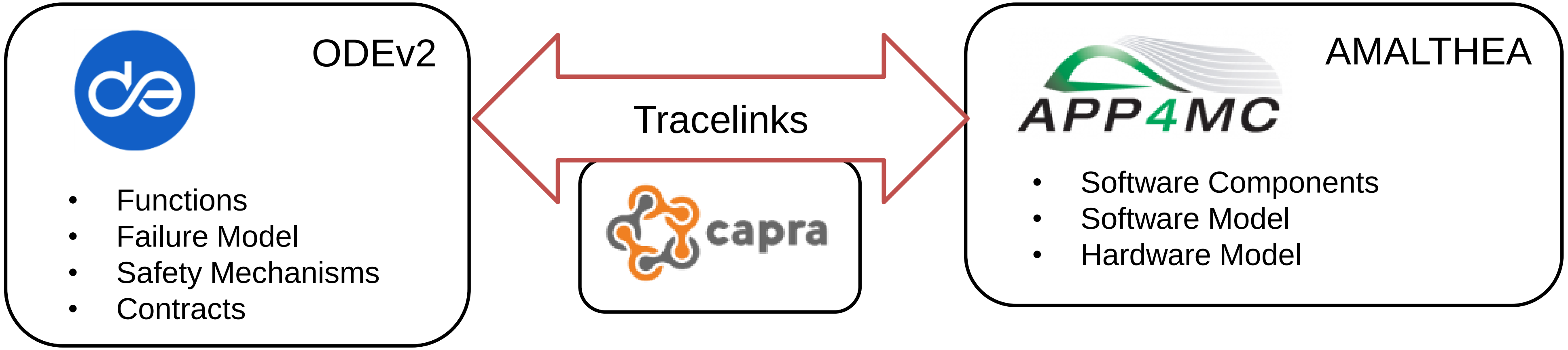


```
AcquireData [Runnable]
  Runnable Items
    call DataAcquisitionWithCheck [RunnableCall]
    Mode Switch [RunnableModeSwitch]
      case: "transient fault" [ModeSwitchEntry]
        condition: OR [ModeValueDisjunction]
          Mode Label Data_Status == FaultDetectionState::Transient_Fault_Detected [ModeValue]
        call DataAcquisitionWithCheck [RunnableCall]
DataAcquisitionWithCheck [Runnable]
  Runnable Items
    read Data_via_I2C [LabelAccess]
    Ticks [Ticks]
    Runnable Probability Switch [RunnableProbabilitySwitch]
      0.01 [ProbabilitySwitchEntry]
        "cut-sets" -> (List) [CustomProperty]
          List Object [ListObject]
            List Object [ListObject]
              values: CustomEvent "I2C_Transmission_Fault" [ReferenceObject]
        Mode Switch [RunnableModeSwitch]
          case: "transient fault" [ModeSwitchEntry]
            condition: OR [ModeValueDisjunction]
              Mode Label Data_Status == FaultDetectionState::No_Fault_Detected [ModeValue]
            write Data_Status [ModeLabelAccess]
          default [ModeSwitchDefault]
            write Data_Status [ModeLabelAccess]
      0.9 [ProbabilitySwitchEntry]
        "cut-sets" -> (List) [CustomProperty]
          List Object [ListObject]
        write Data_Status [ModeLabelAccess]
        write Data_Mem [LabelAccess]
```

- Result of fault detection modeled as probability switch
- Branches annotated with detected faults



Ongoing Work: ODE + AMALTHEA

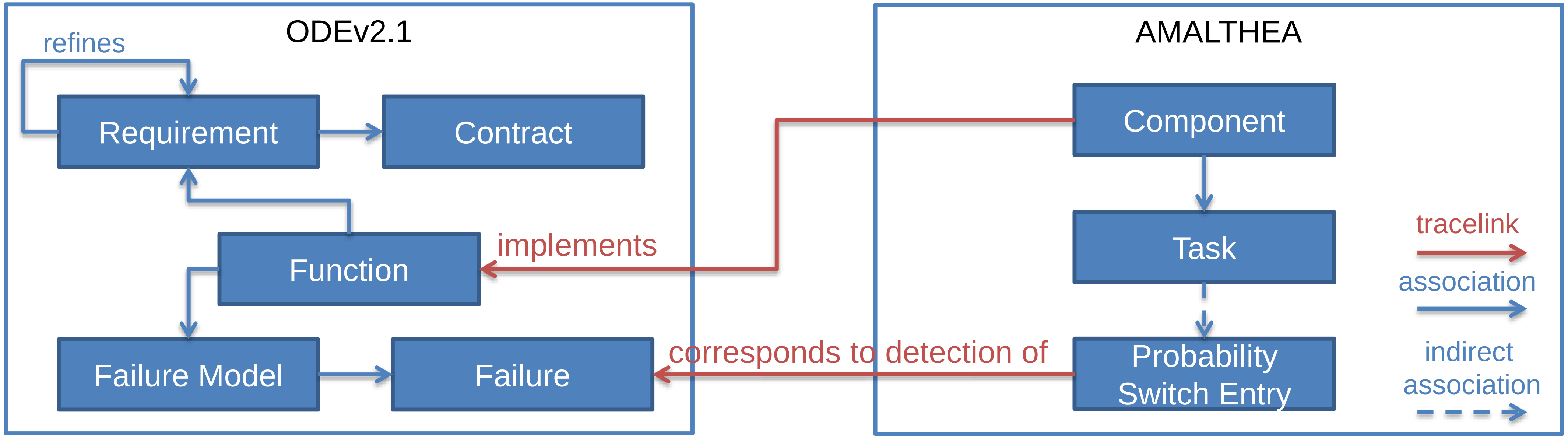


ODEv2

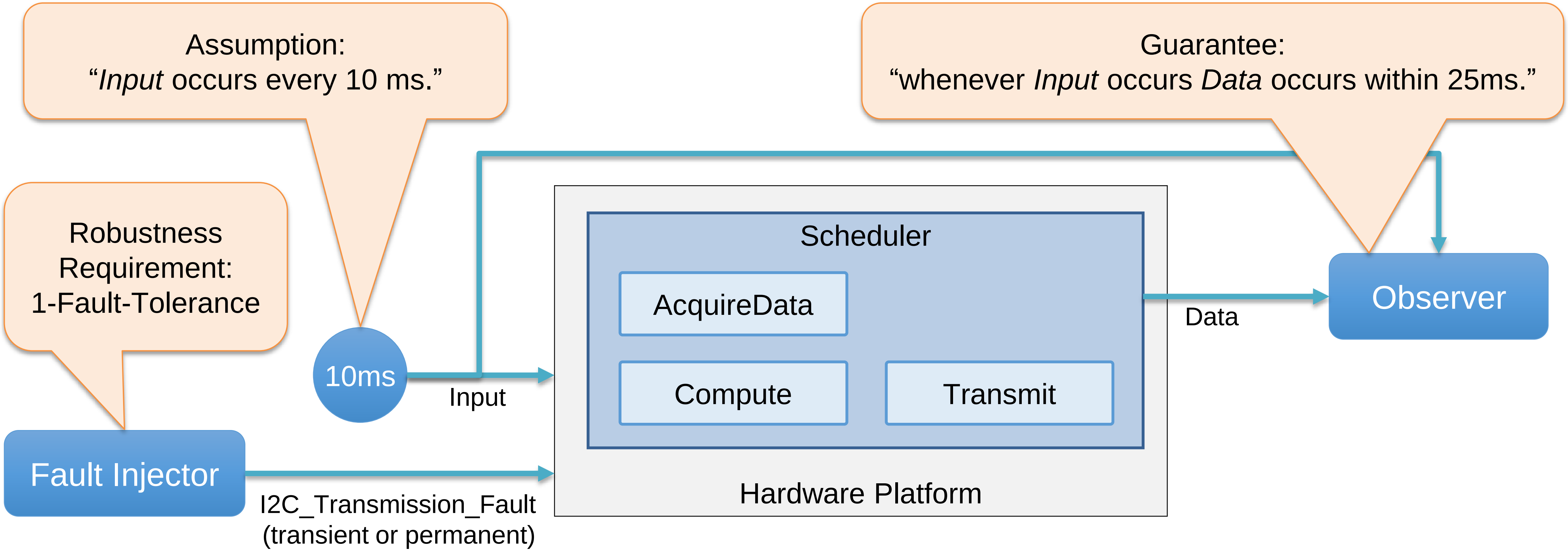
- Functions
- Failure Model
- Safety Mechanisms
- Contracts

AMALTHEA

- Software Components
- Software Model
- Hardware Model



Analysis Model

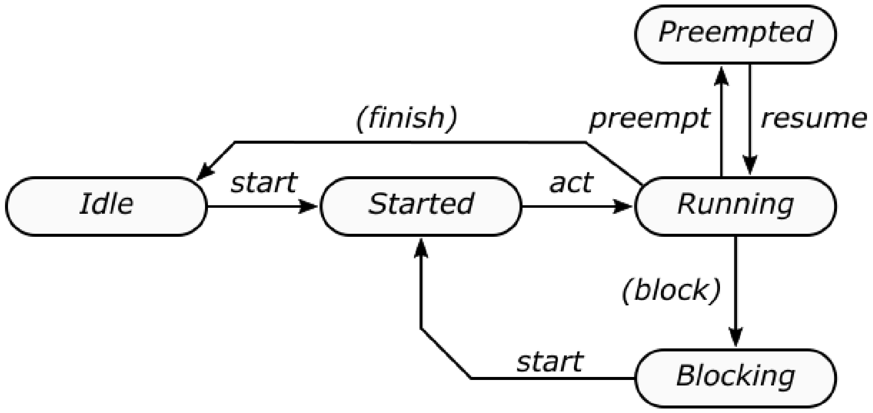


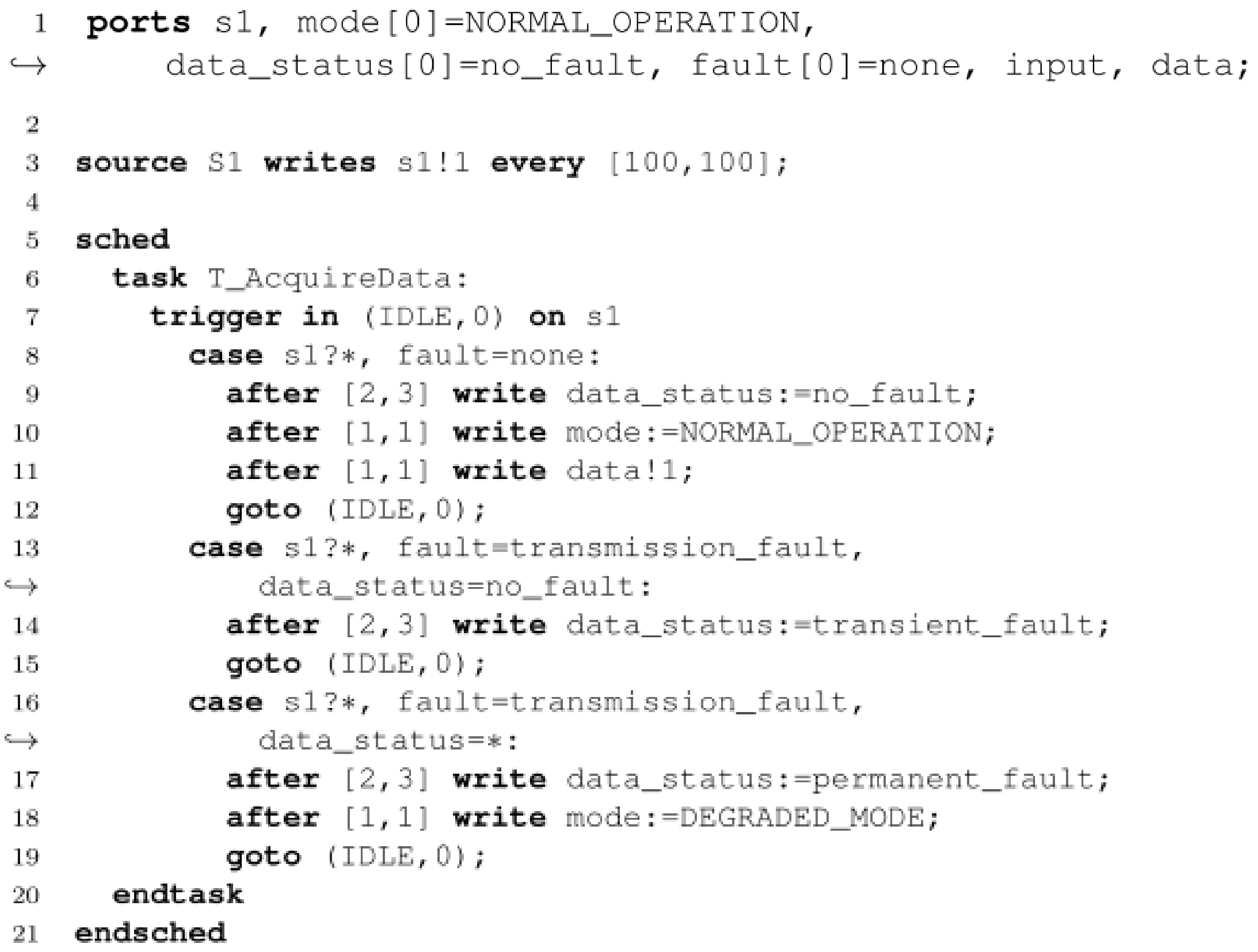
Assumption:
"*Input* occurs every 10 ms."

Guarantee:
"whenever *Input* occurs *Data* occurs within 25ms."

Robustness Requirement:
1-Fault-Tolerance

RTana2sim Model & Syntax

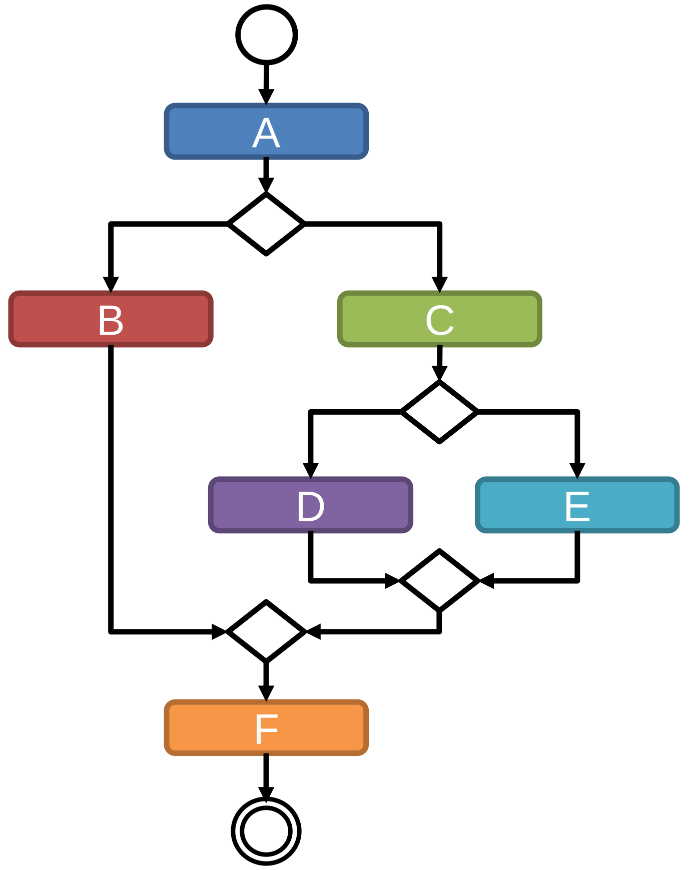



```
ports s1, mode[0]=NORMAL_OPERATION,
    data_status[0]=no_fault, fault[0]=none, input, data;

source S1 writes s1!1 every [100,100];

sched
  task T_AcquireData:
    trigger in (IDLE,0) on s1
      case s1?*, fault=none:
        after [2,3] write data_status:=no_fault;
        after [1,1] write mode:=NORMAL_OPERATION;
        after [1,1] write data!1;
        goto (IDLE,0);
      case s1?*, fault=transmission_fault,
          data_status=no_fault:
        after [2,3] write data_status:=transient_fault;
        goto (IDLE,0);
      case s1?*, fault=transmission_fault,
          data_status=*:
        after [2,3] write data_status:=permanent_fault;
        after [1,1] write mode:=DEGRADED_MODE;
        goto (IDLE,0);
  endtask
endsched
```

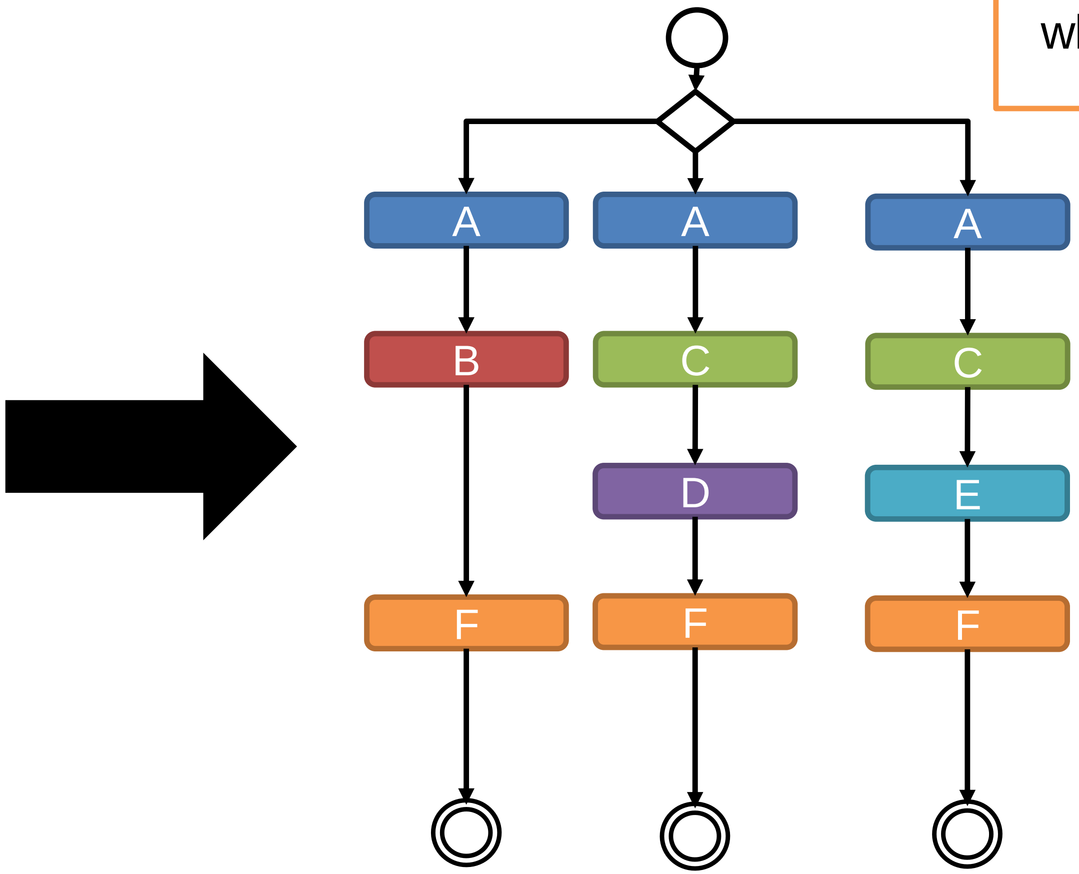
Transformation: Switches

AMALTHEA

Activity Graph



RTana2sim



Assumption:
Read data is stable
while task is running

Transformation: Service Calls

AMALTHEA

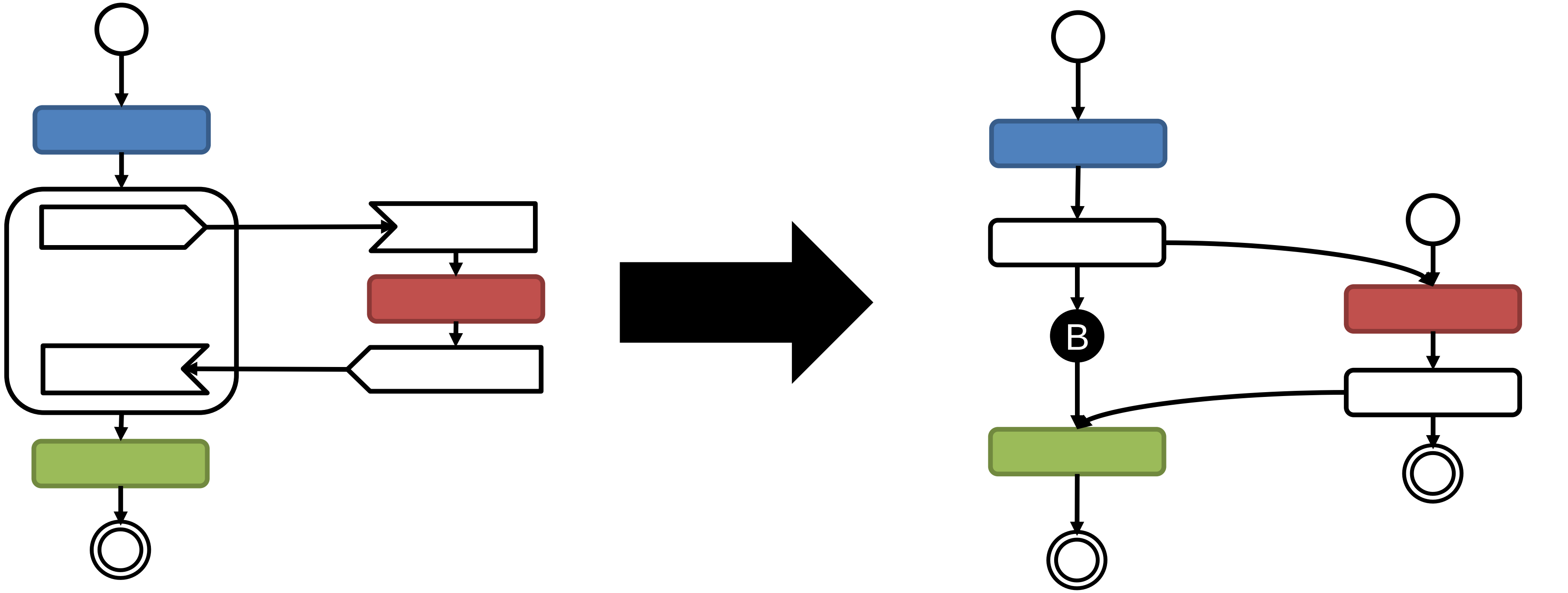
Calling Task

Service Runnable

RTana2sim

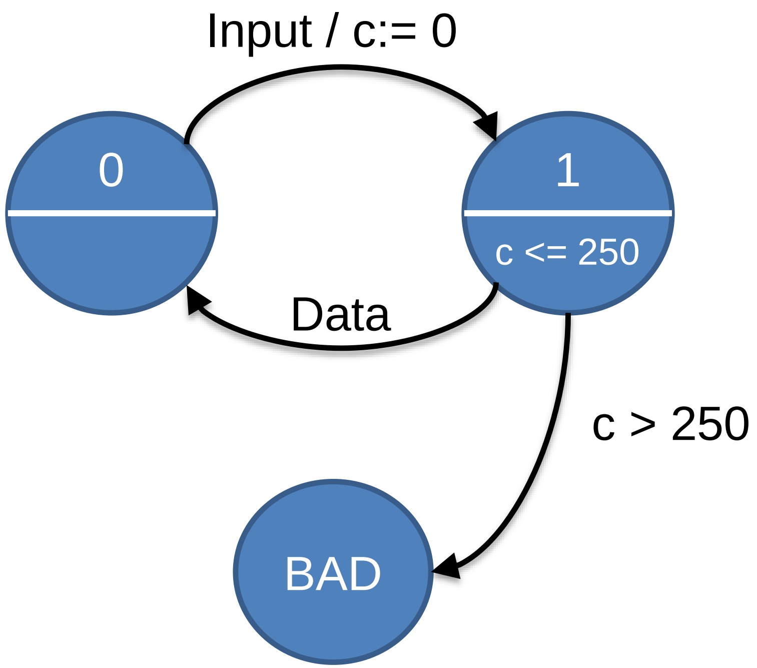
Calling Task

Service Task

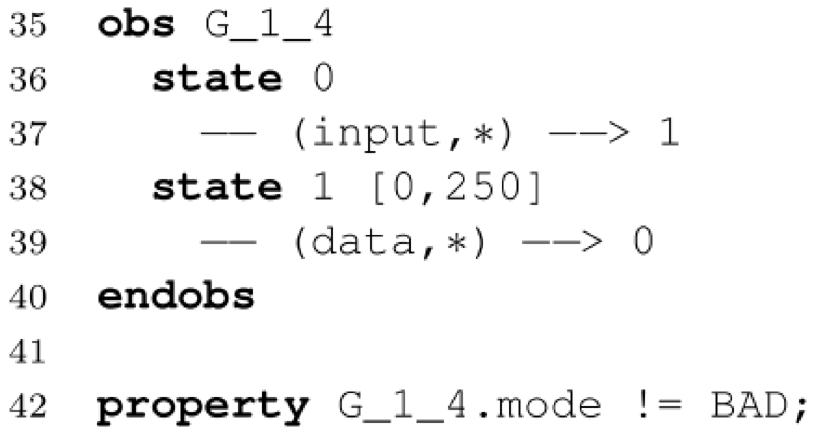


Observers

- Basically a restricted form of one-clock timed automata
- Transitions reset the clock
- Bad state entered when state invariant exceeds



Guarantee:
“whenever *Input* occurs *Data* occurs within 25ms.”



```
obs G_1_4
  state 0
    -- (input,*) --> 1
  state 1 [0,250]
    -- (data,*) --> 0
endobs

property G_1_4.mode != BAD;
```