Politecnico di Milano
Formal Methods for Concurrent and Real-Time Systems

Computer Controller Automatic Transmission

Alessandra Bonetto
Filippo Sironi
Matteo Villa

2009

Contents

List of Figures

Listings

1 Big Picture

Figure 1 shows the *big picture* of the *Computer Controlled Automatic Transmission* we designed.

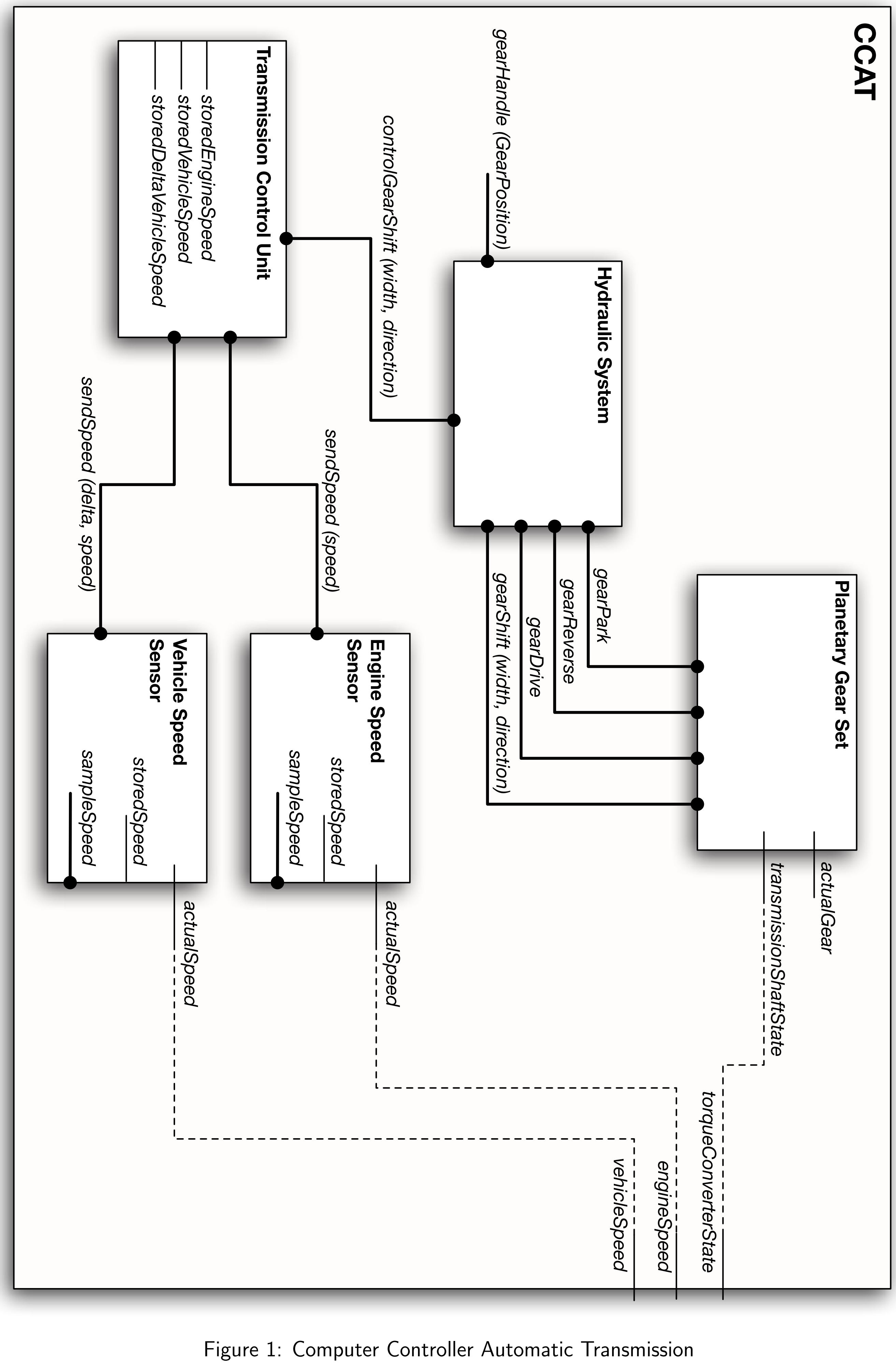

Figure 1: Computer Controller Automatic Transmission

2 Vehicle/EngineSpeedSensor Classes

The *VehicleSpeedSensor* is formalized thanks to the code reported in Listing 1 while the *EngineSpeedSensor* is formalized thanks to the code reported in Listing 2.

During the formalization of sensors we decided to simplify the design assuming that every time a `sampleSpeed` event occurs the state variable `actualSpeed` - which is `time dependent` and `total` - is automatically updated with the actual measured speed. This means we don't provide any axioms formalizing this behavior.

Moreover, we specified the starting point of the constant frequency sample chain saying that sometimes in the past there was a `sampleSpeed` occurrence. Further more, we guarante that `sampleSpeed` events will accure at constant frequency. In addition, if the sensor has memory we imposed that the `storedValue` is equal to 0. These can be consider just like the "initial conditions" of the system.

At the end, we guaranteed a sensor performs the needed action if and only if a sample event occur.

We didn't write any axioms specifying the fact that a `sendSpeed` event is mutually exclusive with itself due to the `total time dependent` parameter it accepts.

Listing 1: VehicleSpeedSensor.trio

```
class VehicleSpeedSensor (const sampleInterval, const
   sampleDelay)

signature:

visible:
    actualSpeed,
    sendSpeed;

temporal domain: real;

items:
    TI sampleInterval: real;
    TI sampleDelay: real;
    TD total storedSpeed: integer;
    TD total actualSpeed: integer;
    event sendSpeed (integer, integer);
    event sampleSpeed;

```

```
axioms:
vars:
    deltaSpeed: integer;
    speed: integer;
formulae:
    SpeedValues:
        actualSpeed >= 0 and storedSpeed >= 0;

    BeginSample:
        SomP (storedSpeed = 0 & sampleSpeed);

    SamplingDefinition:
        sampleSpeed implies Futr (sampleSpeed,
           sampleInterval) and not Lasts (sampleSpeed,
           sampleInterval);

    SamplingAction:
        sampleSpeed implies Futr (deltaSpeed = actualSpeed
           - storedSpeed and speed = actualSpeed and
           sendSpeed (deltaSpeed, speed) and Lasts
           (storedSpeed = actualSpeed, sampleInterval),
           sampleDelay);

    SendSpeed:
        deltaSpeed = actualSpeed - storedSpeed and
           actualSpeed = speed and sendSpeed (deltaSpeed,
           speed) implies Past (sampleSpeed, -sampleDelay);

end
```

Listing 2: EngineSpeedSensor.trio

```
class EngineSpeedSensor (const sampleInterval, const
   sampleDelay)

signature:

visible: actualSpeed, sendSpeed;

temporal domain: real;

items:
    TI sampleInterval: real;
    TI sampleDelay: real;
    TD total actualSpeed: integer;
    event sendSpeed (integer);
    event sampleSpeed;

axioms:
vars:
    speed: integer;
formulae:
    SpeedValues:
        actualSpeed >= 0;

    BeginSample:
        SomP (sampleSpeed);

    SamplingDefinition:
        sampleSpeed implies Futr (sampleSpeed,
            sampleInterval) and not Lasts (sampleSpeed,
            sampleInterval);

    SampleSpeedActions:
        sampleSpeed implies Futr (actualSpeed = speed and
            sendSpeed (speed), sampleDelay);

    SendSpeed:
        actualSpeed = speed and sendSpeed (speed) implies
            Past (sampleSpeed, sampleDelay);

end
```

3 PlanetaryGearSet Class

The *PlanetaryGearSet* class is formalized thanks to the code reported in Listing 3.

The Planetary Gear Set guarantees that every time a gear shift event occurs the `actualGear` will be maintained until the shift is finished.

Inside this component are defined all axioms limiting gear shifts to effective ones only (e.g. it is impossibile to shift down a gear if `actualGear` is `First`.

Moreover, through the formalization of the Planetary Gear Set we impose that we can't receive a gear shift event if we are in the middle of a gear shift. Different gear shifting times are defined for different gears and different steps.

The gears `Drive`, `Park`, and `Reverse` can be selected if and only if the transmission shaft is decoupled from the engine.

The state of the Planetary Gear Set changes if and only if an event occurs.

Listing 3: PlanetaryGearSet.trio

```
class PlanetaryGearSet (const singleGearShiftDelay, const dualGearShiftDelay, const driveGearShiftDelay, const parkGearShiftDelay, const reverseGearShiftDelay)

signature:

visible:
    actualGear,
    transmissionShaftState;
    gearShift,
    gearDrive,
    gearPark,
    gearReverse,

temporal domain: real;

domains:
    Gear: {First, Second, Third, Park, Reverse};
    TransmissionShaftState: {Attached, Detached};
    ShiftWidth: 1..2;
    ShiftDirection: {Up, Down};

items:
    TI singleGearShiftDelay: real;
    TI dualGearShiftDelay: real;
    TI driveGearShiftDelay: real;
    TI parkGearShiftDelay: real;
    TI reverseGearShiftDelay: real;
    TD total actualGear: Gear;
    TD total transmissionShaftState:
        TransmissionShaftState;
    event gearShift (ShiftWidth, ShiftDirection);
    event gearDrive;
    event gearPark;
    event gearReverse;

axioms:
vars:
    gearShiftWidth: ShiftWidth;
    gearShiftWidth2: ShiftWidth;
    gearShiftDirection: ShiftDirection;
    gearShiftDirection2: ShiftDirection;
    gear: Gear;
formulae:
    GearDriveShift:
        (actualGear = Reverse and gearDrive implies (Lasts
            (actualGear = Reverse, driveGearShiftDelay) and
            Futr (actualGear = First,
            driveGearShiftDelay))) and
        (actualGear = Park and gearDrive implies (Lasts
            (actualGear = Park, driveGearShiftDelay) and
            Futr (actualGear = First,
            driveGearShiftDelay))) and
        (actualGear = First or actualGear = Second or
            actualGear = Third implies not gearDrive) and
        (gearDrive iff transmissionShaftState = Detached);

    GearShiftsFirst:
        (actualGear = First implies Alw (not gearDrive and
            not ex gearShiftWidth (gearShiftDirection = Down
             and gearShift (gearShiftWidth,
            gearShiftDirection)))) and
        (actualGear = First and gearShiftWidth = 1 and
            gearShiftDirection = Up and gearShift
            (gearShiftWidth, gearShiftDirection) implies
            Lasts (actualGear = First,
            singleGearShiftDelay) and Futr (actualGear =
            Second, singleGearShiftDelay)) and
        (actualGear = First and gearShiftWidth = 2 and
            gearShiftDirection = Up and gearShift
```

```
                (gearShiftWidth, gearShiftDirection) implies
                Lasts (actualGear = First, dualGearShiftDelay)
                and Futr (actualGear = Third,
                dualGearShiftDelay);

    GearShiftsSecond:
        (actualGear = Second implies Alw (not gearDrive and
                not gearPark and not gearReverse and not ex
                gearShiftDirection (gearShiftWidth = 2 and
                gearShift (gearShiftWidth,
                gearShiftDirection)))) and
        (actualGear = Second and gearShiftWidth = 1 and
                gearShiftDirection = Up and gearShift
                (gearShiftWidth, gearShiftDirection) implies
                Lasts (actualGear = Second,
                singleGearShiftDelay) and Futr (actualGear =
                Third, singleGearShiftDelay)) and
        (actualGear = Second and gearShiftWidth = 1 and
                gearShiftDirection = Down and gearShift
                (gearShiftWidth, gearShiftDirection) implies
                Lasts (actualGear = Second,
                singleGearShiftDelay) and Futr (actualGear =
                First, singleGearShiftDelay));

    GearShiftsThird:
        (actualGear = Third implies Alw (not gearDrive and
                not gearPark and not gearReverse and not ex
                gearShiftWidth (gearShiftDirection = Up and
                gearShift (gearShiftWidth,
                gearShiftDirection)))) and
        (actualGear = Third and gearShiftWidth = 1 and
                gearShiftDirection = Down and gearShift
                (gearShiftWidth, gearShiftDirection) implies
                Lasts (actualGear = Third,
                singleGearShiftDelay) and Futr (actualGear =
                Second, singleGearShiftDelay)) and
        (actualGear = Third and gearShiftWidth = 2 and
                gearShiftDirection = Down and gearShift
                (gearShiftWidth, gearShiftDirection) implies
                Lasts (actualGear = Third, dualGearShiftDelay)
                and Futr (actualGear = First,
                dualGearShiftDelay));

    GearShiftsReverse:
        (actualGear = Reverse implies Alw (not gearReverse
                and all gearShiftWidth, gearShiftDirection (not
                gearShift (gearShiftWidth,
                gearShiftDirection)))) and
        (actualGear = Reverse implies SomF (gearDrive or
                gearPark)) and (actualGear = First and
                gearReverse implies Lasts (actualGear = First,
                reverseGearShiftDelay) and Futr (actualGear =
                Reverse, reverseGearShiftDelay)) and
        (actualGear = Park and gearReverse implies Lasts
                (actualGear = Park, reverseGearShiftDelay) and
                Futr (actualGear = Reverse,
                reverseGearShiftDelay)) and
        (gearReverse iff transmissionShaftState =
                Detached);

    GearShiftsPark:
        (actualGear = Park implies Alw (not gearPark and
                all gearShiftWidth, gearShiftDirection (not
                gearShift (gearShiftWidth,
                gearShiftDirection)))) and
        (actualGear = Park implies SomF (gearDrive or
                gearReverse)) and
        (actualGear = First and gearPark implies Lasts
                (actualGear = First, parkGearShiftDelay) and
                Futr (actualGear = Park, parkGearShiftDelay))
                and
        (actualGear = Reverse and gearPark implies Lasts
                (actualGear = Reverse, parkGearShiftDelay) and
                Futr (actualGear = Park, parkGearShiftDelay)Futr
                (actualGear = Park, parkGearShiftDelay)) and
        (gearPark iff transmissionShaftState = Detached);

    GearShiftsTimings:
        all gearShiftDirection ((actualGear = First or
                actualGear = Second or actualGear = Third) and
                gearShiftWidth = 1 and gearShift
                (gearShiftWidth, gearShiftDirection) implies not
                Lasts (gearDrive or gearPark or gearReverse or
                ex gearShiftWidth2, gearShiftDirection2
                (gearShift (gearShiftWidth2,
                gearShiftDirection2)), singleGearShiftDelay))
```

```
                   and
               all gearShiftDirection ((actualGear = First or
                   actualGear = Third) and gearShiftWidth = 2 and
                   gearShift (gearShiftWidth, gearShiftDirection)
                   implies not Lasts (gearDrive or gearPark or
                   gearReverse or ex gearShiftWidth2,
                   gearShiftDirection2 (gearShift (gearShiftWidth2,
                    gearShiftDirection2)), dualGearShiftDelay)) and
               ((actualGear = Reverse and gearDrive) implies not
                   Lasts (gearDrive or gearPark or gearReverse or
                   ex gearShiftWidth2, gearShiftDirection2
                   (gearShift (gearShiftWidth2,
                   gearShiftDirection2)), driveGearShiftDelay)) and
               ((actualGear = Reverse and gearPark) implies not
                   Lasts (gearDrive or gearPark or gearReverse or
                   ex gearShiftWidth2, gearShiftDirection2
                   (gearShift (gearShiftWidth2,
                   gearShiftDirection2)), parkGearShiftDelay)) and
               ((actualGear = Park and gearDrive) implies not
                   Lasts (gearDriver or gearPark or gearReverse or
                   ex gearShiftWidth2, gearShiftDirection2
                   (gearShift (gearShiftWidth2,
                   gearShiftDirection2)), driveGearShiftDelay)) and
               ((actualGear = Park and gearReverse) implies not
                   Lasts (gearDrive or gearPark or gearReverse or
                   ex gearShiftWidth2, gearShiftDirection2
                   (gearShift (gearShiftWidth2,
                   gearShiftDirection2)), reverseGearShiftDelay));

        Nothing:
               all gear (actualGear = gear and not (all
                   gearShiftWidth, gearShiftDirection (gearShift
                   (gearShiftWidth, gearShiftDirection)) or
                   gearDrive or gearPark or gearReverse) implies
                   UpToNow (actualGear = gear) and NowOn
                   (actualGear = gear));

end
```

4 HydraulicSystem Class

5 TransmissionControlUnit Class