Contract–based design and verification using SPARK 2014

Simon Buist Stuart Matthews Thomas Wilson

12 June, Ada Europe 2019, Warsaw

Agenda

- Introduction
- Context of the system
- Worked example
  - Test–driven development (TDD)
  - Contracts
- How contracts affected design & verification
- Benefits of using contracts

Introduction

- This talk details the practical experience of using SPARK 2014 contracts in the implementation of a critical system.
- It is a high safety-integrity system compliant with UK DEF STAN 00-56.

Embedded Protection Subsystem

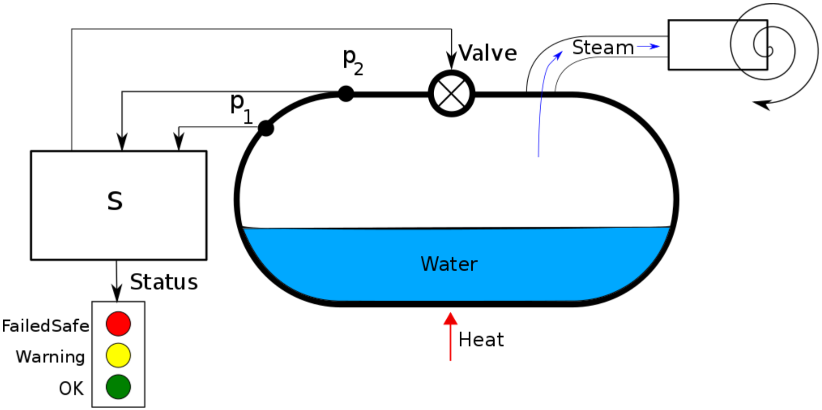


Figure 1: Boiler

Worked example: TDD

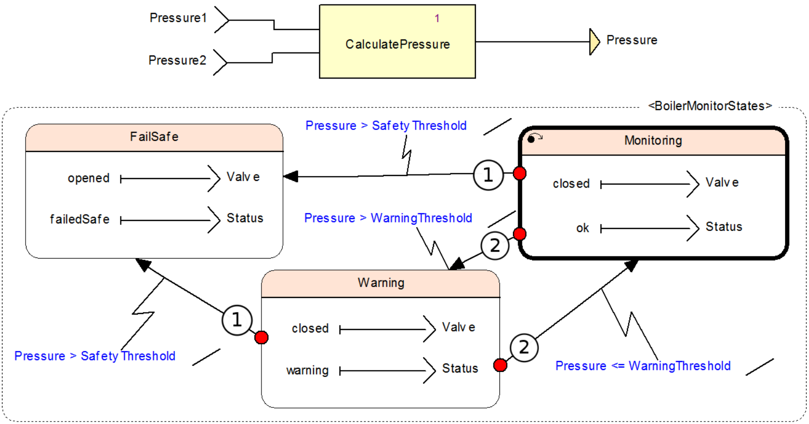


Figure 2: SCADE Requirement

Comparing TDD with Contracts

TDD

SCADE $\rightarrow$ English Specification $\rightarrow$ Test $\rightarrow$ Implementation

Contracts

SCADE $\rightarrow$ Contract $\rightarrow$ Static analysis $\rightarrow$ Implementation

Worked example: TDD

English specification

```
[Pressure is ... of Pressure_1 and Pressure_2]

If [Status = Failed_Safe]
   in any previous cycle then [Status = Failed_Safe]
Otherwise, if [Pressure > Safety_Threshold] then
   [Status = Failed_Safe]
Otherwise, if [Pressure > Warning_Threshold] then
   [Status = Warning]
Otherwise, if [Pressure <= Warning_Threshold] then
   [Status = OK]
Otherwise, [Status = Failed_Safe]
```

Worked example: TDD

API

```
function Update (Old_State : State_T;
                 Pressure_1 : Base_Types.Float64;
                 Pressure_2 : Base_Types.Float64)
   return Result_T;
```

Worked example: TDD

Test

```
procedure Test_Calculate_Pressure
is
   Test_Initialise;
   Test_Step_Covers ("S.Calculate_Pressure.Scenario.1");
   Set_State (Old_State  => State_T'(...),
              Pressure_1 => Base_Types.Float64 (0.0),
              Pressure_2 => Base_Types.Float64 (1.0));
   Check_Result(
      Result_T'(Boiler_Monitor_State => Monitoring;
                Status               => OK;
                Valve                => Closed));
   Test_Initialise; -- another test step...
end Test_Calculate_Pressure;
```

Worked example: Contracts

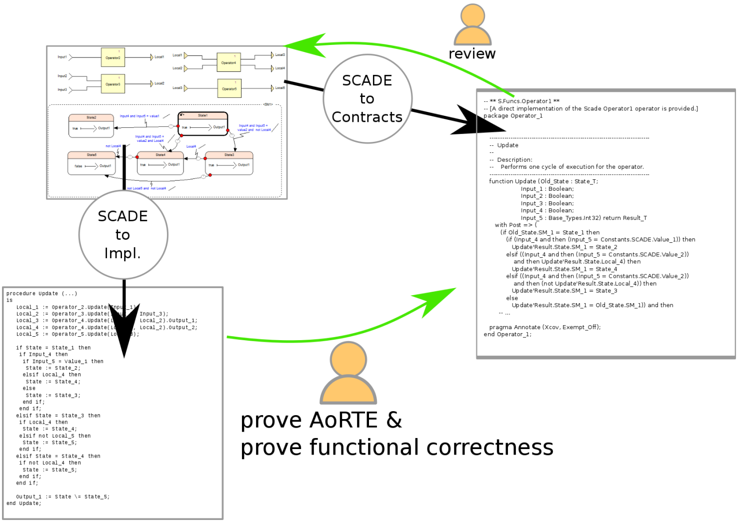


Figure 3: Contracts

altran

Worked example: Contracts

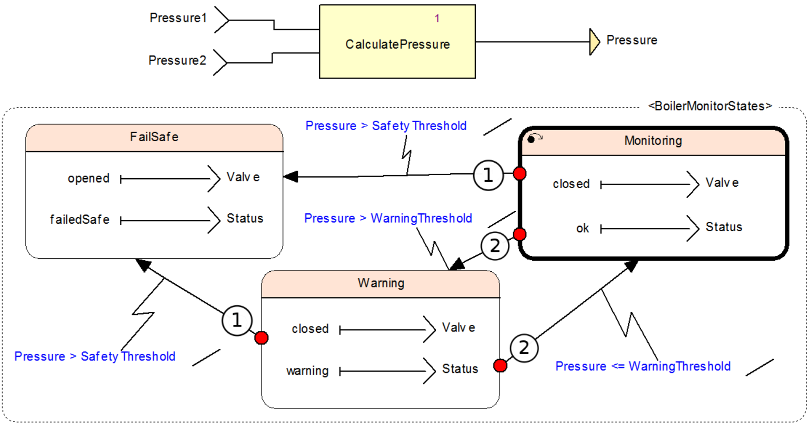


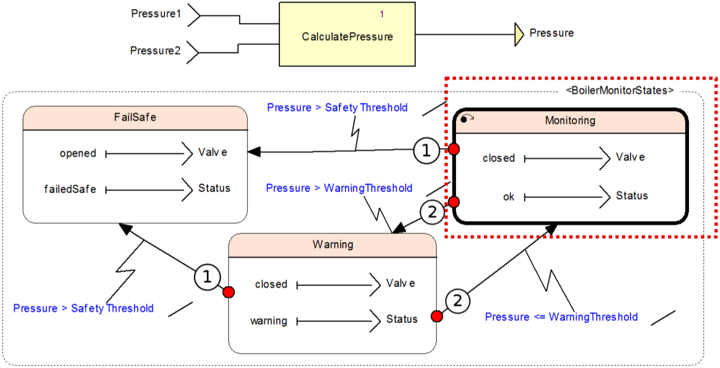
Figure 4: SCADE Requirement

Worked example: Contracts

```
function Update (...) return Result_T
with Post => (
(Calculate_Pressure.Result_T' (
   State => Update'Result.State.Calculate_Pressure_1_State,
   Output_1 => Update'Result.State.Pressure) = Calculate_Pressure.Update (
      Old_State => Old_State.Calculate_Pressure_1_State, Input_1 => Pressure_1,
      Input_2 => Pressure_2)) and then
(if Old_State.Boiler_Monitor_States = Monitoring then
   (if (Update'Result.State.Pressure > Constants.SCADE.Safety_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Fail_Safe
   elsif (Update'Result.State.Pressure > Constants.SCADE.Warning_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Warning
   else Update'Result.State.Boiler_Monitor_States = Old_State.Boiler_Monitor_States))
         and then
(if Old_State.Boiler_Monitor_States = Warning then
   (if (Update'Result.State.Pressure > Constants.SCADE.Safety_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Fail_Safe
   elsif (Update'Result.State.Pressure <= Constants.SCADE.Warning_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Monitoring
   else Update'Result.State.Boiler_Monitor_States = Old_State.Boiler_Monitor_States))
         and then
(if Old_State.Boiler_Monitor_States = Fail_Safe then
   Update'Result.State.Boiler_Monitor_States = Old_State.Boiler_Monitor_States)
      and then
(if Update'Result.State.Boiler_Monitor_States = Monitoring then
   Update'Result.Valve = Update'Result.State.Closed and then
   Update'Result.Status = Update'Result.State.Ok) and then
(if Update'Result.State.Boiler_Monitor_States = Warning then
   Update'Result.Valve = Update'Result.State.Closed and then
   Update'Result.Status = Update'Result.State.Warning) and then
(if Update'Result.State.Boiler_Monitor_States = Fail_Safe then
   Update'Result.Valve = Update'Result.State.Opened and then
   Update'Result.Status = Update'Result.State.Failed_Safe));
```

Worked example: Contracts

```
(if Old_State.Boiler_Monitor_States = Monitoring then
   (if (Update'Result.State.Pressure > Constants.SCADE.Safety_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Fail_Safe
   elsif (Update'Result.State.Pressure > Constants.SCADE.Warning_Threshold) then
      Update'Result.State.Boiler_Monitor_States = Warning
   else Update'Result.State.Boiler_Monitor_States = Old_State.Boiler_Monitor_States))
```



Autogenerated Ada body

```
function Update (Old_State : State_T;
                 Pressure_1 : Base_Types.Float64;
                 Pressure_2 : Base_Types.Float64) return Result_T
is
   Result : Result_T;
begin
   Result.State.Pressure := Calculate_Pressure.Update (
      Old_State => Old_State.Calculate_Pressure_1_State,
      Input_1 => Pressure_1,
      Input_2 => Pressure_2).Output_1;
   Result.State.Calculate_Pressure_1_State := Calculate_Pressure.Update (
      Old_State => Old_State.Calculate_Pressure_1_State,
      Input_1 => Pressure_1,
      Input_2 => Pressure_2).State;
   Result.State.Boiler_Monitor_States := (if
(Old_State.Boiler_Monitor_States = Monitoring) then (if (Result.State.Pressure
> Constants.SCADE.Safety_Threshold) then Fail_Safe else (if
(Result.State.Pressure > Constants.SCADE.Warning_Threshold) then Warning else
Old_State.Boiler_Monitor_States)) else (if (Old_State.Boiler_Monitor_States =
Warning) then (if (Result.State.Pressure > Constants.SCADE.Safety_Threshold)
then Fail_Safe else (if (Result.State.Pressure <=
Constants.SCADE.Warning_Threshold) then Monitoring else
Old_State.Boiler_Monitor_States)) else Old_State.Boiler_Monitor_States));
Result.Valve := (if (Result.State.Boiler_Monitor_States = Monitoring) then
Result.State.Closed else (if (Result.State.Boiler_Monitor_States = Warning)
then Result.State.Closed else Result.State.Opened));
   Result.Status := (if
(Result.State.Boiler_Monitor_States = Monitoring) then Result.State.Ok else (if
(Result.State.Boiler_Monitor_States = Warning) then Result.State.Warning else
Result.State.Failed_Safe));
      return Result;
end Update;
```

Comparing TDD with Contracts

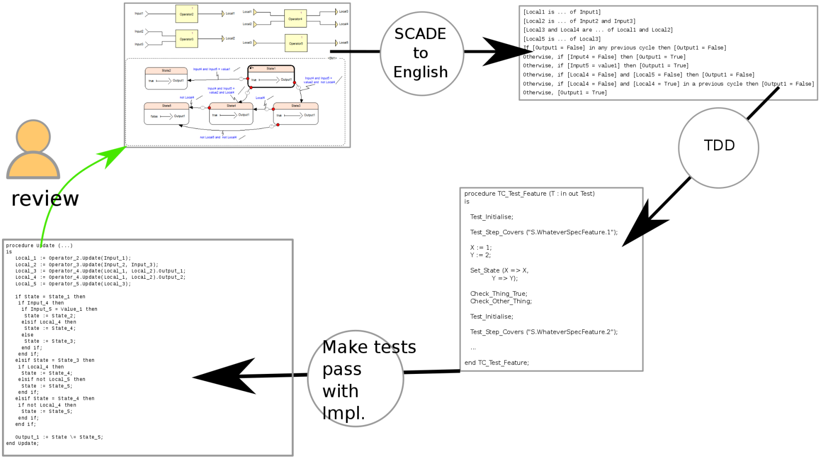


Figure 5: TDD

Comparing TDD with Contracts

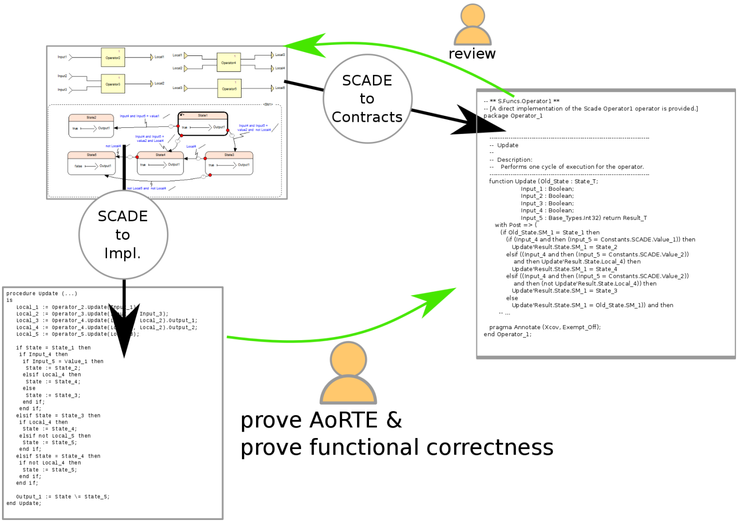


Figure 6: Contracts

altran

How contracts affected verification

Verification of the system takes a hybrid approach, using both proof and test to establish functional correctness of the implementation. The SPARK 2014 contracts play a role in both these verification activities.

How contracts affected verification: dynamic (testing)

- Run-time checking of the contracts ensures they are always met during system testing, because we're using Ada 2012 contracts.
- Even though functional correctness had been proven, the run-time checking found an error in a low-level interrupt handler.

Verification: delivered executable

- Using the flag `--gnata`, we left the contracts built-in to the delivered executable.
- We designed the system so that any failure of such a run-time check will have the effect of putting the system into a safe state.

Run-time checking of contracts

When we used 64-bit floating point operations within interrupt handlers for the first time, if the interrupt handler interrupted a floating point operation then the top 32-bits of the registers could be corrupted

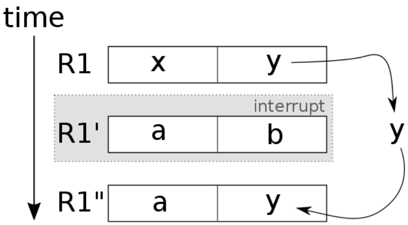


Figure 7: Register corruption

Conclusion

- Zero defects found in code derived from SCADE specifications.
- Leaving run-time checks in found fault on target bootloader.
- We found a viable & practicable technique for proving correctness against the SCADE specification.

Acknowledgement: This work was supported by the SECT-AIR project, funded by the Aerospace Technology Institute and Innovate UK, as project number 113099.

altran