CARL SCHULTZ, STEFAN HALLERSTEDE

STEAM BOILER CASE

AARHUS UNIVERSITY

Copyright © 2021 Carl Schultz, Stefan Hallerstede

PUBLISHED BY AARHUS UNIVERSITY

TUFTE-LATEX.GOOGLECODE.COM

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.apache.org/licenses/LICENSE-2.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

First printing, February 2021

Contents

Introduction

This document describes a model of a steam-boiler.[1] The purpose of the steam-boiler is to drive a turbine by moving it with steam. The steam boiler contains water that is heated to produce the steam. The amount of water contained in the boiler is critical: if there is too much, or too little water, the turbine and boiler itself are at risk of serious failure. The boiler system has a *software controller* that is responsible for ensuring that the quantity of water stays within a safe range.

Figure 1 presents a diagram of the steam-boiler system. The system consists of:

- a water container;
- two pumps that can pump water into the container;
- a valve that can drain water out of the container;
- a steam outlet;
- a software controller responsible for ensuring that the water quantity stays with a safe range.

[1] This is an adapted and highly simplified case based on an original case description written by LtCol. J.C. Bauer for the Institute for Risk Research of the University of Waterloo, Ontario, Canada. Subsequently, an adapted case description was written by Jean-Raymond Abrial which has been used as the main source of details for the case presented here:

Jean-Raymond Abrial, Egon Börger, and Hans Langmaack. The steam boiler case study: Competition of formal program specification and development methods. In *Formal Methods for Industrial Applications*, pages 1–12. Springer, 1996

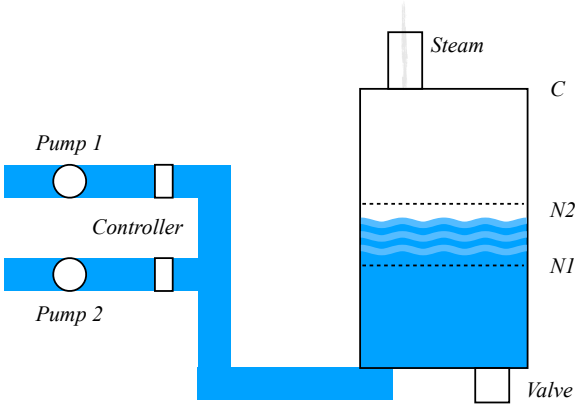

Figure 1: Diagram of the steam-boiler.

Model Variables and Constants

Table 1 lists the variables in the steam-boiler model.

Variable	Type	Units	Description
q_e	real	kg	Water quantity in the boiler
s_e	real	kg/sec	Quantity of steam exiting the boiler
h_e	real	kg/sec^2	Heat factor
v_e	integer	(none)	Status of the valve
p_{1e}	integer	(none)	Status of pump 1
p_{2e}	integer	(none)	Status of pump 2

Table 1: Variables in the steam-boiler model.

Each pump can be *on*, *off*, or *failed*. The status of each pump determines the values of the variables p_{1e} and p_{2e}: if pump i is *on* then $p_{ie} = 1$ otherwise $p_{ie} = 0$. Pumps can be repaired once they have failed.

The valve can be *open* or *closed*. The status of the valve determines the value of the variable v_e: if the valve is open then $v_e = 1$, otherwise $v_e = 0$.

Table 2 lists the constants in the steam-boiler model. All constants are *reals*.

Constant	Units	Description
C	kg	Water capacity of the boiler
W	kg/sec	Capacity of the steam released through the outlet
P	kg/sec	Capacity of the pump
V	kg/sec	Capacity of the valve
N_1	kg	Lower bound of the safe water quantity level
N_2	kg	Upper bound of the safe water quantity level
U_1	kg/sec^2	Maximum rate of *increase* of steam output
U_2	kg/sec^2	Maximum rate of *decrease* of steam output

Table 2: Constants in the steam-boiler model. All constants are *reals*.

The evolution of the quantity of water in the boiler over time is defined by the formula:

$$\frac{dq_e}{dt} = P(p_{1e} + p_{2e}) - Vv_e - s_e.$$

The change in steam output over time is defined by the formula:

$$\frac{ds_e}{dt} = h_e.$$

The boiler has a maximum water capacity, and thus q_e is subject to the following constraints:

$$0 \leq q_e \leq C.$$

The steam outlet has a maximum output capacity, and has restrictions on its rate of change over time. Thus, s_e is subject to the following constraints:

$$0 \leq s_e \leq W,$$

$$-U_2 \leq \frac{ds_e}{dt} \leq U_1.$$

Table 3 presents suggested constant values and initial variable values. Simulation traces may start with different initial values to exercise various scenarios and use cases.

Table 3: Suggested initial variable values, and suggested constant values.

Variable	Suggested Initial Value	Constant	Suggested Value
q_e	500	C	1000
s_e	0	W	25
h_e	0	P	20
v_e	0	V	50
p_{1e}	0	N_1	400
p_{2e}	0	N_2	600
		U_1	1
		U_2	1

Controller Requirements

The controller must ensure that the water quantity q_e stays within the safe range, i.e.

$$N_1 \leq q_e \leq N_2.$$

The steam output constraints are (assumed to be) governed by physical laws, and thus the controller is not responsible for ensuring that steam output constraints are satisfied. The controller is connected to sensors and actuators that enable the controller to perform the following functions:

- monitor the water quantity;
- monitor the steam output quantity;
- monitor the status of each pump;
- monitor the status of the valve;
- turn each pump on and off;
- open and close the valve.

The controller has four different modes that determine its expected behaviour according to the state of the boiler:

1. Initialisation mode
2. Normal mode
3. Degraded mode
4. Emergency stop mode

Initialisation Mode

When the controller is started up it first enters *Initialisation* mode. During initialisation, the controller monitors the water quantity and

utilises the pumps and valve to ensure that the quantity falls within the safe range between N_1 and N_2.

Figure 2 illustrates the controller's required response based on water quantity q_e. If q_e is below N_1 then the controller must react by turning on the pumps and closing the valve. If q_e is above N_2 then the controller must react by turning off the pumps and opening the valve. If q_e is between N_1 and N_2 then the controller turns off the pumps, closes the valve, and enters *Normal mode*. Once the controller has exited *Initialisation* mode it never returns back to it before controller execution is terminated.

The boiler should initially have no heat and therefore no steam output. If the controller detects that the steam output is greater than zero during initialisation then the controller switches into *Emergency Stop* mode.

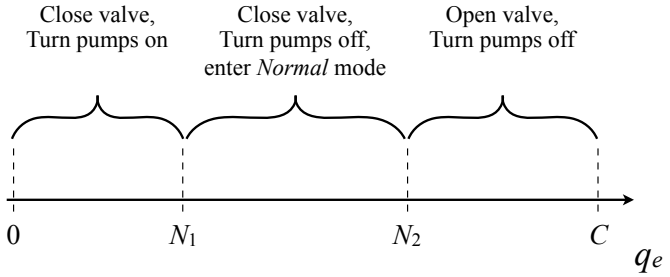

Figure 2: Required response of the controller based on the water quantity level q_e during *Initialisation* mode.

Normal and Degraded Mode

In *Normal* and *Degraded* mode, the controller only uses the pumps to alter the water quantity. The boiler is expected to be producing heat and outputing steam, which continuously lowers the water quantity, q_e. Figure 3 illustrates the controller's required response based on water quantity q_e. If q_e falls below N_1 then the controller must turn on the pumps. If q_e rises above N_2 then the controller must turn off the pumps. If the water quantity is near either limit (0 or C) then the controller must also immediately switch into *Emergency Stop* mode.

If exactly one of the pumps has *failed* then the controller switches into *Degraded* mode. If the controller is in *Degraded* mode and both pumps are no longer failed (i.e. due to being repaired) then the controller switches back to *Normal* mode. If both pumps have *failed* then the controller switches to *Emergency Stop* mode.

Emergency Stop Mode

In this mode, the responsibility for taking appropriate actions is given over to the physical environment. Therefore, the controller

software program terminates.

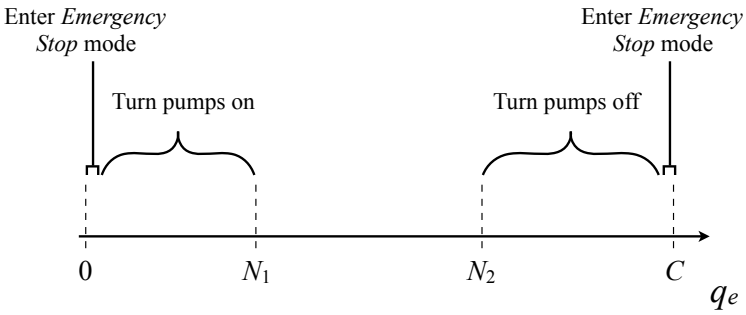

Figure 3: Required response of the controller based on the water quantity level q_e during *Normal* and *Degraded* modes.

Bibliography

Jean-Raymond Abrial, Egon Börger, and Hans Langmaack. The steam boiler case study: Competition of formal program specification and development methods. In *Formal Methods for Industrial Applications*, pages 1–12. Springer, 1996.