ISSN 0280–5316
ISRN LUTFD2/TFRT--5501--SE

Modelling and Simulation of an Industrial Control Loop with Friction

Jonas Eborn

Department of Automatic Control
Lund Institute of Technology
March 1994

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
March 1994

Document Number
ISRN LUTFD2/TFRT--5501--SE

Author(s)
Jonas Eborn

Supervisor
Henrik Olsson

Sponsoring organisation

Title and subtitle
Modelling and Simulation of an Industrial Control Loop with Friction

Abstract

Modelling of an industrial control loop for concentration control using Omola, an object-oriented modelling language, is described. The major modelling effort has been made to make a complete nonlinear model of the pneumatic positioner–actuator of the control valve in the loop. A library of several friction models has also been developed. These models have been used to describe friction in the control valve. The different friction models have been compared to each other and the effects of various frictional phenomena on the control loop behaviour have been studied. The control loop has been simulated using the OmSim working environment. The simulations show that concentration variations observed during operation of the control loop at a paper mill can be caused by friction alone. Possible measures to overcome the control valve problem are studied and a simple way of reducing the problem is proposed.

Key words
friction,modelling,simulation,control valve,process control,pneumatic

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
English

Number of pages
56

Recipient's notes

Security classification

Contents

Preface

About this report

This report is a Master's thesis in Engineering Physics presented at the Department of Automatic Control, Lund Institute of Technology.

I would like to thank everybody who has helped me with this project: Jens Pantzare at Stora Technology; Percy Wangeby at SOMAS; professor Karl Johan Åström for inspiration; Sven Erik Mattsson, Mats Andersson and Tomas Schönthal for explaining Omola and always correcting the bugs in OmSim that I discovered and my supervisor Henrik Olsson for being supportive throughout the course of this project.

Finally, I would like to thank my wife Pernilla and my daughter Elise.

Purpose of the work

The purpose of this work was originally to use object-oriented modelling to describe different friction models and investigate them in a tracking application using a simple linear process model. The simplicity of the modelling language Omola was however greater than expected and we wanted to use a more elaborate process model. The idea of modelling an industrial application with a pneumatic positioner and actuator came up. Tore Hägglund had, as part of his work with control valves, received process data from a control application which we believed to suffer from problems concerning friction. The application is a loop for control of the concentration of fibres in pulp at a paper mill. We decided to incorporate this application in my work and to model in detail the entire control loop, especially the pneumatic positioner and actuator. The model would then be used to study the effects of friction on the performance of the control loop and to see if friction could cause the type of problems that had been observed.

1. Introduction

A paper mill processes pulp into paper. To be able to produce high quality paper there are a number of key points in the process that needs to be controlled well. One of these, as pointed out by many different sources, is the first step in the process where the pulp received from the pulp mill is fed into the paper process. Good concentration control in this step of the process facilitates control in succeeding steps. Variations in the fibre concentration can affect the quality of the paper produced. Efforts have been made to show a correlation between variations in fibre concentration and paper quality [Pantzare, 1993]. Although these efforts have not yet succeeded, it is still felt that fibre concentration control is a very important issue.

The concentration of fibres in pulp is controlled by diluting the pulp with water. The water flow is controlled with a PI-controller and a control valve. This is a fairly simple setup used in many paper mills. The application studied in this report is a loop for concentration control at STORA Billerud Paper in Gruvön, just outside Karlstad, Sweden. In this control loop there is a problem with concentration fluctuations appearing after some period of operation and then increasing very slowly in amplitude until the problem has to be dealt with. The solution so far has been to do maintenance on the control valve. This must be done while the process is stopped and can therefore be costly. Figure 1.1 shows recorded data from the control loop. The output signal, y, is normalised to take values between zero and one and so is the control signal, v.

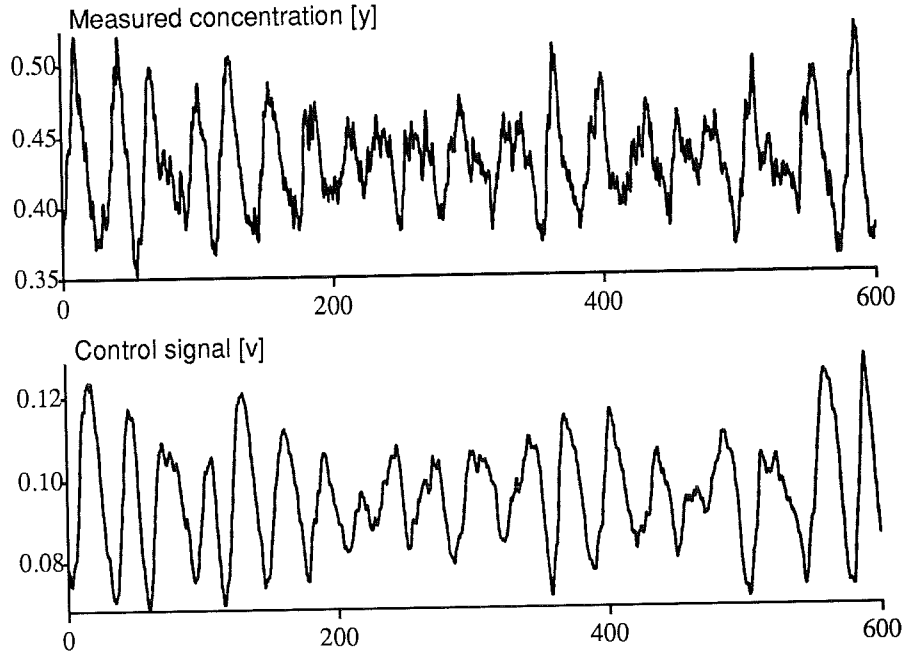

Figure 1.1 Concentration fluctuations during operation of the control loop.

In Figure 1.2 is a diagram of the control loop. Pulp flowing through a pipe is diluted with water at the beginning of an enlargement of the pipe. The pipe enlargement is 70 cm long and has a diameter of 30 cm. The fibre concentration is measured at the downstream end of the enlargement. The sensor estimates the fibre concentration in the fluid by measuring the torque required to drive a propeller at constant speed. The controller in the loop is a pneumatic PI-

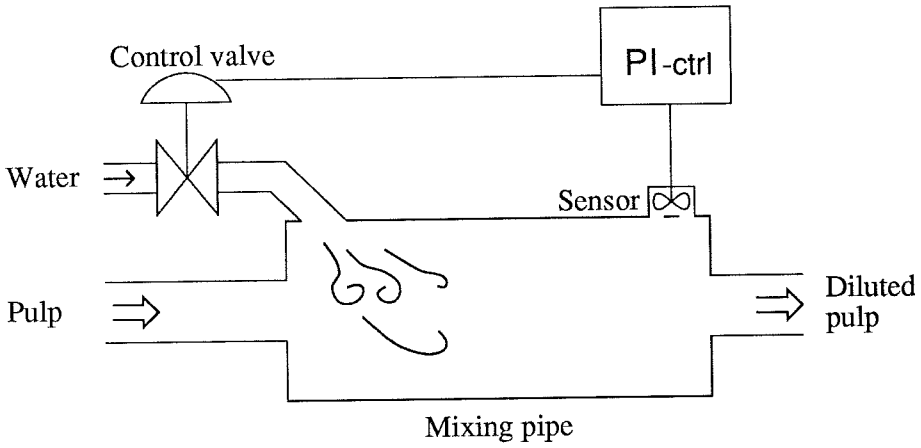

Figure 1.2 A schematic diagram of the setup for pulp fibre concentration control.

controller. The control signal is the reference value to the control valve, the desired valve position. The control valve consists of a pneumatic positioner, a pneumatic actuator and a valve.

The outline of the report is as follows. Chapter 2 is a description of the parts of the control loop studied in this report. Chapter 3 contains a detailed functional description of the positioner–actuator in the loop. The friction models are described in Chapter 4. Simulations and comparisons with real data can be found in Chapter 5 and in Chapter 6 is a summary of the results. Appendix A explains the pneumatic theory used. Appendix B gives a brief explanation of the working environment, Omola and OmSim, and an introduction to the developed models. The Omola code of the models is listed in Appendix C.

2. The Control Loop

This section will give a further description of the parts of the control loop.

2.1 Mixing pipe

The process being controlled is dilution of pulp flowing through a pipe. The diluting point is at the beginning of a 70 cm long enlargement while concentration measurements are made at the end of the enlargement. The concentration in the pipe varies with both position and time. To describe the process in a simplified way, the mixing, which is in reality distributed along the pipe, is considered perfect and instantaneous at the diluting point. The transport delay of less than a second is lumped with the dynamics of the sensor.

For perfect mixing, the steady-state equation of the fibre concentration is simple,

$$C_{out} = C_{in}\frac{Q_{in}}{Q_{in} + Q_{water}}$$

where C_{in} is the fibre concentration before the diluting point, Q_{in} is the volume flow rate of pulp into the pipe and Q_{water} is the volume flow rate of water through the control valve. In the model developed, C_{in} and Q_{in} are considered constant.

The concentration sensor is of type MEK2000 from BTG in Källe. It measures the torque required to drive a propeller at constant speed. From this measurement it estimates the viscosity of the fluid and calculates the fibre concentration. The sensor is placed in a small chamber in the side wall of the pipe. This suggests that there is both a delay and a time constant associated with the concentration measurement. The sensor has been modelled as a first-order lag with time delay, which includes the transport delay of the pulp flow through the mixing pipe.

2.2 PI-controller

The controller in the loop is a VALMET pneumatic PI-controller from 1971. The only information available concerning this device is the controller parameters. The controller is modelled as a standard parallel PI-controller, given by

$$\begin{aligned} e(t) &= y(t) - y_{ref}(t) \\ i(t) &= \tfrac{1}{T_i}\int^t e(\tau)d\tau \\ v(t) &= K(e(t) + i(t)) \end{aligned}$$

where y is the measured signal, e is the control error and i is the integral part of the control action. The control signal, v, is limited to values between 0 and 1. This signal is then transformed into a pressure signal in the range 20-100kPa. Note that this is a direct controller, which is the reason why $e = y - y_{ref}$.

Between the controller and the positioner is a pneumatic transmission line. The exact length of the line has not been measured but the physical distance between controller and positioner is over 20 meters. The length of this line introduces a lag between signal pressure at the controller and the control

pressure at the positioner. As an approximation, a second-order transfer function for transmission lines can be derived (see Appendix A.3). This transfer function has also been implemented in the model of the controller.

2.3 Positioner–actuator

The pneumatic positioner–actuator is a power servo used to control the position of the valve. The positioner–actuator is mounted on the valve as shown in Figure 2.1.

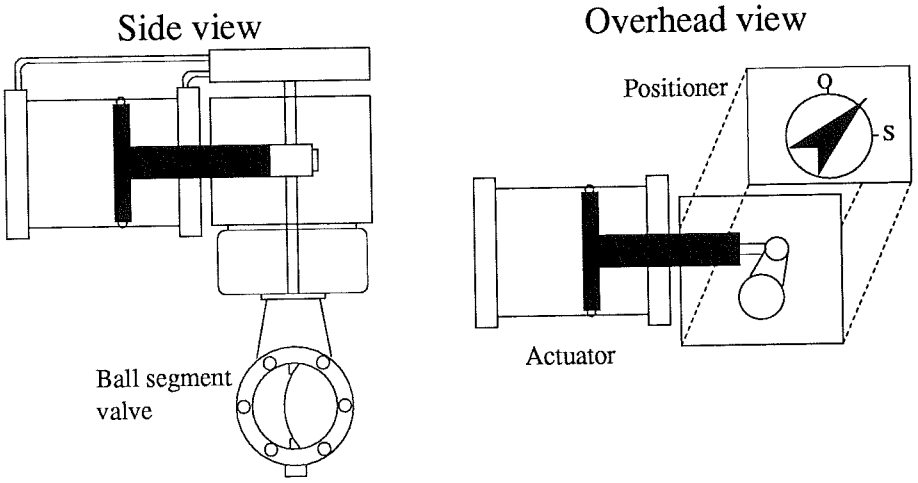

Figure 2.1 The positioner–actuator mounted on the ball segment valve.

The modelled positioner is a SP400 with pneumatic control signal from SOMAS in Säffle. It is a nonlinear control device controlling the flow of compressed air into and out of the cylinder chambers of the actuator. The flow is a nonlinear function of the position error and since the chamber pressures are roughly the integral of the flow the positioner–actuator will act as an integrating controller. The actuator is of type SOMAS S21. This is a double-acting actuator, with pressure on both sides of the piston. The actuator cylinder is mounted on a housing. This housing is mounted on the rotary axis of the valve. A lever in the housing converts the linear motion of the actuator to rotary motion. The positioner is also mounted on this axis, thus measuring valve position.

There is friction present in both the actuator and the valve. Two friction models could be used, one for the actuator friction and one for the valve friction. Using two similar friction models would render twice as many degrees of freedom in the parameters. It would be possible to simulate a more complex friction behaviour, but it would also be more difficult to analyse. Also, there would be problems with the simulation. One problem is events occurring at almost the same time. Another is that having two sets of friction equations would slow down the simulations. Since the actuator and the valve are rigidly coupled, the friction of the valve can be included in the actuator friction model. Only one friction model is used to account for the total friction of the actuator and valve.

The important dynamics of the system are those of the positioner–actuator. Also the major part of the friction in the system is in the actuator or it can be added to the friction of the actuator. The positioner and actuator have

been the main modelling objectives and they are described in more detail in Section 3.

2.4 Valve

The valve is a ball segment valve of type SOMAS KVT1. The characteristics of the valve are somewhere in between linear and equal percentage as can be seen in Figure 2.2. In this application the valve operates very near the closed position. As shown in Figure 1.1 the control signal is 10%±3%. Valve opening never exceeds the control signal with more than a few percent. For such small changes the valve characteristic will be close to linear.

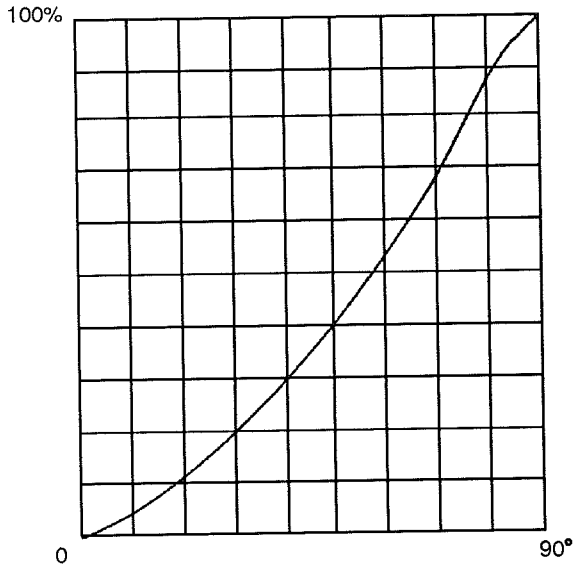

Figure 2.2 Valve characteristic of SOMAS KVT1 ball segment valve.

No measurements are available of pressure in the water or pulp pipe, from which to make water flow calculations. With the small operating range in mind the control valve has been modelled as a linear valve with a parameter Q_{max} determining the flow through a fully open valve. The parameter has been set according to knowledge of the pulp flow and fibre concentrations at normal operation [Pantzare, 1993]. The assumption of a linear valve will of course be valid only in a range around the operating point.

2.5 Discussion

Gain, time constant and dead-time of the process

The models that have been developed in this work are based on product information and fundamental mechanics and thermodynamics. Only a few measurements are available. These are the recordings of measured concentration and control signal during steady-state operation shown in Figure 1.1 and an identification experiment that has been made to determine a process model

from control signal to measured concentration [Pantzare, 1993]. This experiment gave estimates of the parameters of a first-order process model with dead-time. The gain estimate is about three times while the estimates for time constant and dead-time both are approximately 10 seconds.

When all the sources of dynamics in the loop model excluding the concentration sensor are added together, half of the dead-time and a fifth of the time constant of the estimated first-order model can be accounted for. The remaining dynamics are attributed to the concentration sensor. Since the measuring of the fibre concentration actually takes place in a small chamber and not in the main flow it is fair to assume that there is both a dead-time and a time constant associated with the concentration measurement. These parameters are unknown and difficult to determine. They have been set to make the sum of dead-times and time constants match those of the estimated process model.

Extensions of the models

As an extension of the mixing pipe model the parameters C_{in} and Q_{in} could be replaced by an input. This would make it possible to run simulations from a data file with measurements of C_{in} and Q_{in}.

The model of the ball segment valve is not complete. It can only be used for simulations of steady-state behaviour. To make simulations of transients like a start-up sequence a model that is valid throughout the whole operating range is required. In order to develop a complete model a look-up table of Figure 2.2 could be made. Measurements or calculations of Q_{max}, the flow through a fully open valve, would also be necessary.

3. Positioner-Actuator Model

In this section the interior of the positioner–actuator will be described. Also, some remarks on the model implementation will be given.

3.1 SP400 Positioner

A pneumatic positioner is a mechanical control device. In fact it is a nonlinear proportional controller. Together with the actuator it works as an integrating controller though. It is the most complex unit of the ones modelled and the key part in explaining the limit cycle behaviour of the control loop.

Functional description

In Figure 3.1 is a diagram of the main parts of the SP400 positioner. The heart of the construction is the pilot valve with spool (40). The spool is held by the lever (46). At equilibrium the position of the valve corresponds to the pneumatic control signal. The lever is kept in the center position by two forces: the force from control pressure on the diaphragm (41) and the force from the position feedback spring (53). While the spool is held in the center position the pressures in the actuator chambers are maintained.

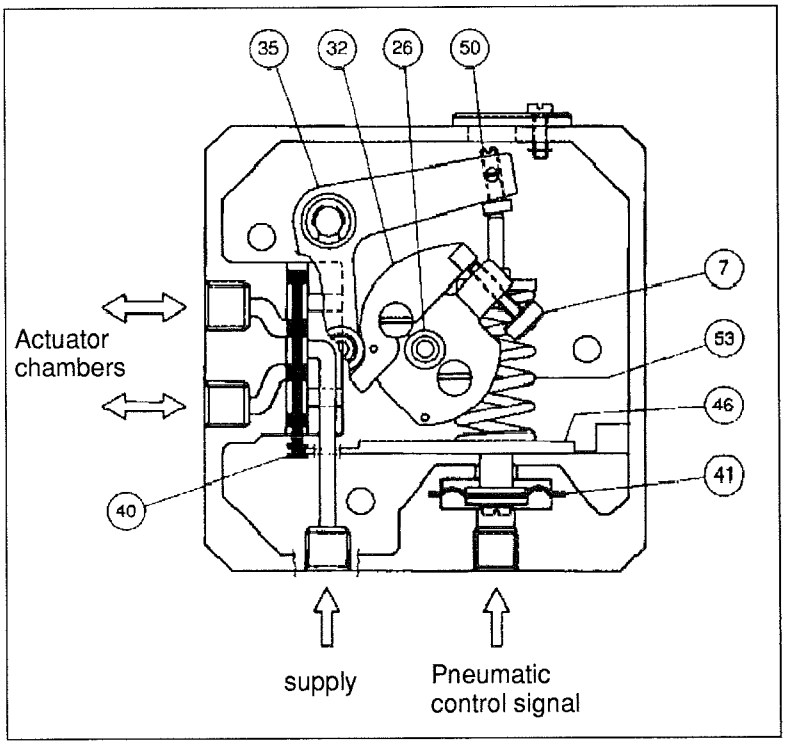

Figure 3.1 Interior of SP400 positioner.

When the control pressure is raised the force on the diaphragm increases and the lever moves the spool. Compressed air flows into one of the actuator chambers and out of the other. The pressure difference in the cylinder gives rise to a force on the piston. When this force becomes big enough the piston moves the valve. The disc (32) is mounted on the valve axis (26). It converts

the rotary motion of the valve axis into linear motion and via the arm (35) this motion is transferred to the spring (53). The spring then presses harder against the lever. As the valve moves to the desired position the lever and the pilot valve spool returns to the center position.

Interface

The interface of the positioner model breaks down into four parts. There is one mechanical coupling to the valve axis and three pneumatic ports: to the actuator chambers and to the controller. The supply port and the exhaust port are not regarded as parts of the interface, but as an infinite source and sink respectively. The supply and atmospheric pressures are regarded as constant.

The mechanical coupling is described by an angle and a torque. The outlet ports to the actuator chambers are described by pressure, temperature and mass flow. The chamber at the diaphragm has almost no volume. Therefore the pneumatic connection to the controller is seen as nonflowing and the control signal interface is modelled with merely a pressure. Interfaces between models in Omola are described in more detail in Appendix B.

Components of the positioner

The important parts of the positioner are: the position feedback spring, the lever, the diaphragm, the disc, the arm and the pilot valve. The spring is modelled as a linear spring with parameters for spring constant and original length. The lever is modelled as a rigid body affected by forces at three points. The relation of these forces are determined by a torque equation with respect to the lever joint. The angular deflection of the lever determines the transversal positions of the points where the lever is connected to the spool, spring and diaphragm.

As stated in the last subsection the air flow into the diaphragm chamber is regarded as zero. This implies that the diaphragm is merely an area converting pressure into force. It has been modelled with an equation with the area as a parameter.

The rotating disc and the arm have a simple task; converting angular position into linear. This task is described by a simple relation and two parameters: one offset position and the gain constant from angle to position.

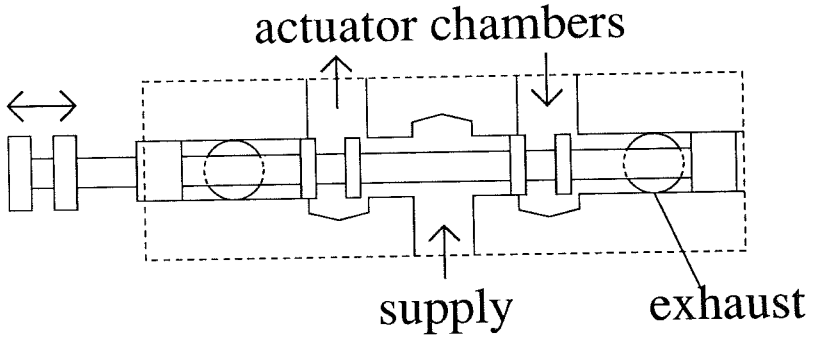

Figure 3.2 Positioner pilot valve in slightly open position.

The pilot valve in Figure 3.2 is the most complicated part of the positioner. It has variable structure. There are four possible ways for air to pass through the valve, but only two of them are active at the same time. Depending on the sign of the spool position the supply port is connected to one of the outlet

ports while the other outlet port is connected to the exhaust port. Each flow way is seen as an orifice, a restriction of air flow, with variable area. The opening area of the orifice is a function of spool position. It is the main factor determining the flow through the valve. The flow equations are described in Appendix A.

Pilot valve characteristics

To calculate the flow through the orifices of the pilot valve, the relation between spool position and flow area has to be determined. The spool has two lands closing the path of the flow when the spool is in the center position. When the spool is moved a little two openings with the shape of circular segments allow for air to pass through the orifice as shown in Figure 3.3; there is another opening on the other side of the spool. An expression of the true flow

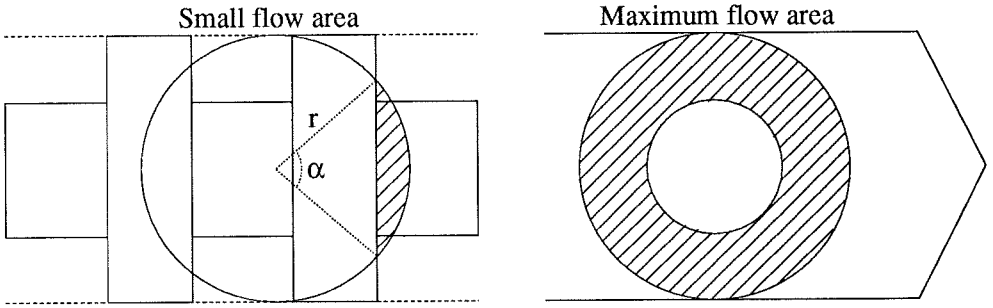

Figure 3.3 Pilot valve flow area in slightly open position and maximum flow area.

area is difficult to derive because of the complex geometry with two circular pipes perpendicular to each other. Ideally measurements would be made to determine the area function. No measurements are available though and an approximation will have to suffice. For small spool displacements the projected area of the opening in Figure 3.3 will be used. At bigger spool displacements the flow area will be limited by the area of the passage between the lands. When the spool has been moved the full pipe width, the flow area will be the cross-section area along the spool as shown to the right in the figure. The spool radius is half the land radius. The maximum flow area is

$$A_{max} = \frac{3\pi r^2}{4}$$

The area of one of the openings can be expressed as the area of a sector of a circle minus the area of a triangle. The expression for the total orifice area is

$$A = 2\left(\pi r^2 \frac{\alpha}{2\pi} - (r - x)\frac{h}{2}\right)$$

where r and α are defined in the figure, x is the spool displacement and h is the height of the opening. By expressing α and h in x, and setting $\xi = x/r$ this expression can be rewritten as

$$A_1 = 2r^2\left(\arctan\frac{\sqrt{1-(1-\xi)^2}}{(1-\xi)} - (1-\xi)\sqrt{1-(1-\xi)^2}\right) = f_1(\xi)r^2 \quad (3.1)$$

where the dimensionless function $f_1(\xi)$ depends only on the geometry. Relation 3.1 will be valid as long as the passage between the lands is larger than the

opening. To describe the transition between (3.1) and the maximum flow area a first-order approximation has been used. Relation 3.1 is used until $\xi = 1/2$ and then the area rises asymptotically towards A_{max} by the expression

$$A_2 = r^2 \left[a + \left(\frac{3\pi}{4} - a \right) \left(1 - e^{-b(\xi - 1/2)} \right) \right] = f_2(\xi) r^2$$

where a and b are chosen to make this expression match the value and derivative of A_1 in the point $\xi = 1/2$. The area function for $\xi > 0$ becomes

$$A = \begin{cases} f_1(\xi) r^2 & , \xi \leq 1/2 \\ f_2(\xi) r^2 & , \xi > 1/2 \end{cases} \tag{3.2}$$

This area function is shown in Figure 3.4.

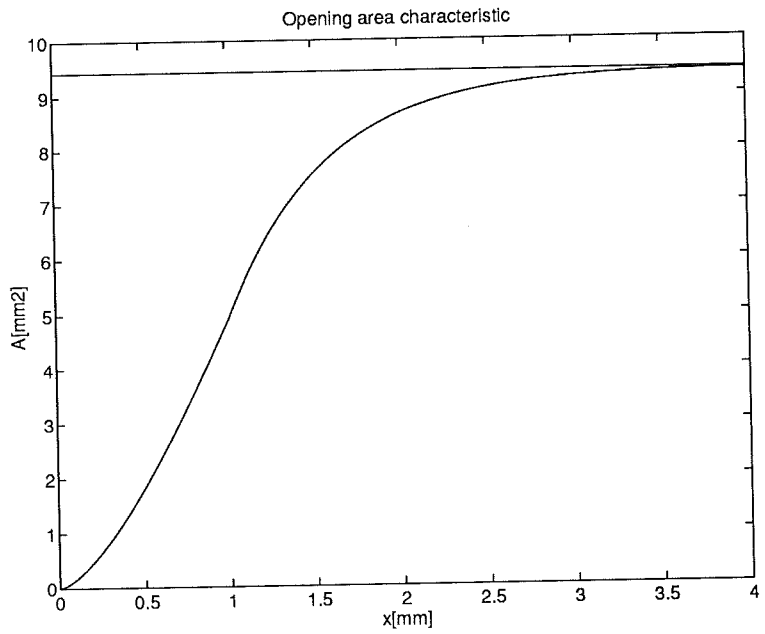

Figure 3.4 Flow area characteristic of the pilot valve. ($r = 2$mm)

SP300 Positioner

The positioner modelled in this work is the newly developed SP400 while the positioner in the application is the older SP300. The reasons for this are not dealt with here.

The function of the two positioners is of course the same. The main difference is that, while the SP400 responds very fast, almost instantly, to a change in the control signal, the SP300 is much slower. The apparent dead-time is several seconds. The reasons are difficult to understand. As shown in Figure 3.5 the interior is similar. The major difference in the construction is that the force from the diaphragm (1) does not act on the lever (3) directly, but via another lever (2). Another difference is that the force from this other lever and the force from the spring (7) are not acting on the same point of the lever. These two factors would not be important if the levers were rigid, but they are not. While the SP400 lever is made of stainless steel, the levers of the SP300 are made of aluminium, a rather soft metal. The forces from the diaphragm and the spring are relatively large and bend the levers instead of deflecting

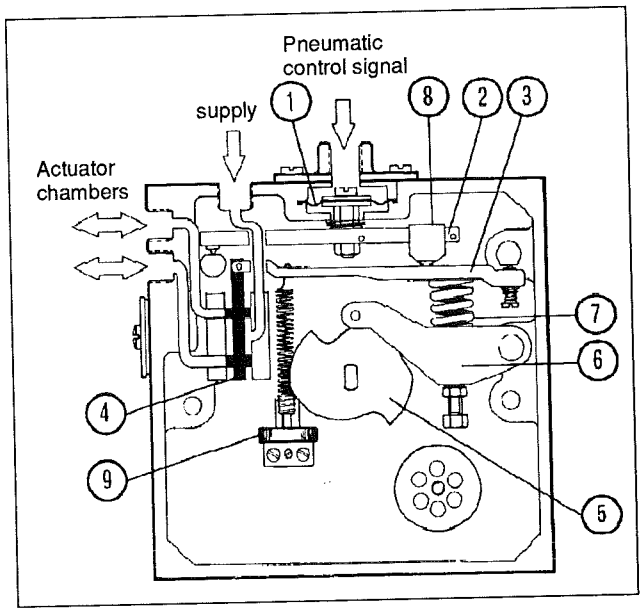

Figure 3.5 Interior of SP300 positioner.

them. The result is that the spool is moved less. This is the sole explanation to the slow dynamics of the SP300 [Wangeby, 1993].

The elastic property of the levers is difficult to model. To be able to use the model of the SP400 and still get the slow response of the SP300 a bending factor has been included in the model. This bending factor divides the spool position so the spool displacement will be smaller.

3.2 S21 Actuator

An actuator is a power element delivering force to move for example a valve. The actuator model has almost the same interface as the positioner: one mechanical coupling to the valve axis and two pneumatic ports, one for each chamber.

Components of the actuator

There are very few parts in the actuator in Figure 3.6. There is the piston and the lever. Also, there are a few abstract "parts", which are regarded as parts in a modelling sense. These are the friction and the two chambers, the volumes of air on each side of the piston.

The piston is modelled as an inertial mass; the acceleration is determined by Newton's equation. Lumped with this mass is also other masses and moments of inertia of the actuator–valve system. The lever relates position of the piston to angular position of the valve. No concern has been taken to the angle of the piston arm; the position of the piston is the same as the position of the endpoint of the lever, see Figure 3.6. In reality there will be a small difference between the two displacements in the figure when the piston is close to one of the end positions. Regarding the displacements as equal makes the conversion to an angle straightforward. Since the valve friction and inertia was lumped with those of the actuator there will be no torques with respect to the valve axis.

The chambers are modelled as variable volumes of air. The states of each chamber are: pressure, temperature, air mass and volume. The variable volume

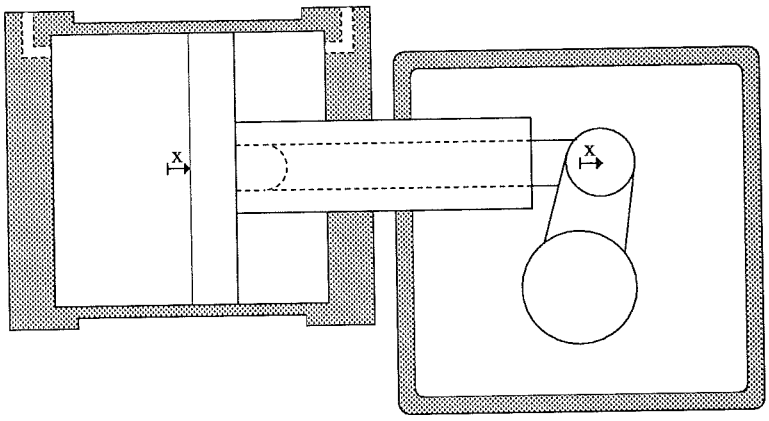

Figure 3.6 A schematic diagram of the S21 actuator.

is caused by one moving wall, called the diaphragm. In the S21 actuator it is a piston and not a diaphragm, but this fact is not important. The relations describing volume change and force will be

$$\dot{V} = -A\dot{x} \qquad\qquad F_x = pA$$

where A is the area perpendicular to the x-direction. The area is taken with sign according to the orientation of the area. The mass of air in the chamber is determined by the mass flow through the inlet. The states are also related by the thermodynamical laws described in Appendix A.

4. Modelling of Friction

The classical descriptions of friction include Coulomb friction, static friction and viscous friction. The friction force is always opposed to the direction of motion. Coulomb friction assumes that the friction force is constant, whereas viscous friction is proportional to the velocity of the motion and static friction is a larger force occurring only during sticking. In all combinations of these friction models the friction force is given as a static function of the velocity although static friction needs to be handled in a special way.

When studying the more detailed behaviour of friction a few observations can be made. The transition between the friction force at rest and during motion is not discontinuous. The force decreases continuously with increased velocity. This is called the Stribeck effect. Another observation is that a small displacement occurs during sticking. The behaviour resembles that of a stiff spring with damping. This is usually referred to as the Dahl effect. When studying friction force as a function of velocity for varying velocities during unidirectional motion a hysteretic effect can be noticed. This phenomenon is sometimes attributed to frictional lag. Furthermore friction shows a random behaviour due to the irregularities of the surfaces in contact.

This work will concentrate on five models of friction with slightly different features

- A classical friction model.
- The Dahl model [Dahl, 1977].
- An extended Dahl model [Canudas *et al.*, 1994].
- The Bristle model [Haessig and Friedland, 1990].
- The Seven parameter model [Armstrong-Hélouvry, 1991].

The classical model is a static model. The other four models are all dynamic and contain, to various extents, the phenomena described previously. An overview of the features of the models is shown in Table 1.

4.1 A model shell

Friction is a nonlinear function of velocity. It is present between surfaces in contact moving relative to each other. The motion can be linear or angular. To be able to use different models of friction behaviour a superclass holding the common denominators of the friction models have been defined

```
FrictionModel ISA Model WITH
parameters:
  Fc, Fs, Fv, Kc, d, vStrib ISA Parameter;
variables:
  F, v ISA Variable;
END;
```

This superclass defines the variables involved, F and v, and the important parameters. The model language, Omola, is described further in Appendix B.

Also, a model shell has been developed. It defines the interface of the friction model and serves to separate the inside, the model of friction behaviour, from the outside, the friction force acting on a surface.

```
FrictionForce ISA Model WITH
  Friction ISA FrictionModel;
  C ISA MechCut;
equations:
  Friction.v = C.x';
  C.Fx = Friction.F;
END;
```

The interface, C, is a mechanical cut of the class MechCut with the components x and Fx. The model shell relates these to the variables of the friction model. The apostrophe in the code denotes a derivative.

4.2 The classical model

The classical model of friction consists of the three frictional phenomena Coulomb, viscous and static friction. In the implementation of the model it has been slightly modified to allow for well-behaved simulations. The relation describing friction force is

$$F = \begin{cases} u & \text{, while Stuck} \\ F_c \operatorname{sign}(v) + F_v v & \text{, while NOT Stuck} \end{cases}$$

where F_c is the Coulomb friction level and F_v is the viscous friction constant. The driving force, u, is the sum of all other forces acting on the body subjected to friction. The boolean variable Stuck is used to describe which mode of friction is used. Switching between the two modes is accomplished by discrete events. Sticking begins when the velocity falls below a certain zero-velocity tolerance. Slipping begins when the driving force becomes greater than F_s, the maximum static friction. The model is similar to the model proposed in [Karnopp, 1985]. There are some major differences though. One important part of the Karnopp model is a dead-zone in the relation between momentum and velocity, which makes the velocity exactly zero during sticking. This dead-zone also defines the zero-velocity region where the body is stuck. In my implementation of the classical friction model the velocity is set to zero when sticking begins.

To achieve absolute sticking the model requires knowledge of the driving force. This limits the use of the model. It will not be easily interchangeable with the other models and in complex situations with multiple forces applied to the object the driving force may be difficult to derive.

4.3 The Dahl model

The original Dahl model is a simple model which contains Coulomb friction and spring-like behaviour at rest. Friction is a continuous function of velocity given by

$$\dot{F} = K_c v \left(1 - \operatorname{sign}(v)\frac{F}{F_c}\right)$$

where K_c is a stiffness parameter.

This model is interesting mainly because it is dynamic and described by one single equation. There is no need for discrete events or mode switching, which complicates the use of other models. The drawback is that many frictional phenomena can not be described by the model.

Feature	Classical	Dahl	Ext. Dahl	Bristle	7-param
Static friction	F_s		F_s	(F_s)	F_s
Coulomb friction	F_c	F_c	F_c	(F_c)	F_c
Viscous friction	F_v		F_v		F_v
Stribeck effect			v_S		v_S
Spring-like behaviour		K_c	K_c, d	K_c	K_c
Frictional lag			*implicitly*		τ_L
Varying break-away force			*implicitly*		γ
Randomness				Δ	

Table 1 Features of the different friction models and the associated parameters.

4.4 An extended Dahl model

In the extended Dahl model, friction force is given by

$$\begin{cases} \dot{z} = v\left(1 - \text{sign}(v)\dfrac{z}{g(v)}\right) \\ F = K_c z + d\dot{z} + F_v v \end{cases}$$

where d is a damping coefficient and $g(v)$ is a function determining the steady-state relationship between velocity and friction force. The function can for example be chosen as

$$g(v) = F_c + (F_s - F_c)e^{-(v/v_S)^2}$$

where v_S is the Stribeck velocity. If $g(v) = F_c$ and $d = F_v = 0$ then this model reduces to the Dahl model as described above.

The model contains all the important frictional phenomena: spring-like behaviour, arbitrary steady-state characteristics, frictional lag and varying break-away force. Frictional lag and varying break-away force are implicit qualities of the model dynamics, they do not depend explicitly on any parameters.

4.5 The Bristle model

The Bristle model is designed to emulate the randomness of friction. The frictional force is summed from spring action of minute bristles, or bonds, which break at a certain deflection and then become reestablished. Static friction is attained through a number of extra bristles which are active near zero velocity. The magnitude of friction cannot be determined directly by the parameters F_s and F_c. Instead the mean Coulomb friction and an approximate magnitude of static friction is determined indirectly via a number of parameters associated with the bristles.

The model is laborious to use in simulations due to the discontinuous behaviour and as a result of this, the large number of discrete events needed.

4.6 The Seven parameter model

The Seven parameter model differs from the others in that it really consists of two models, one active only during sticking and the other only during sliding.

During sticking the model is a very stiff spring

$$F = K_c x \tag{4.1}$$

where x is the displacement. During sliding, it consists of Coulomb, viscous and Stribeck friction with frictional lag

$$F = \text{sign}(v)\left[F_c + (F_{s,act}(t_2) - F_c)\left(\frac{1}{1 + (\frac{v(t-\tau_L)}{v_S})^2}\right)\right] + F_v v$$

where τ_L is the time delay of the frictional lag. The actual static friction, $F_{s,act}$, depends on the duration of the last sticking period. This makes it possible for the model to incorporate time dependent static friction. During sticking, the static part of friction rises asymptotically towards the value F_s by the relation

$$F_{s,act}(t_2) = F_{s,last} + (F_s - F_{s,last})\frac{t_2}{t_2 + \gamma}$$

where $F_{s,last}$ is the magnitude of friction when sticking started, t_2 is the dwell-time at zero velocity and γ is the time constant of rising static friction.

A major drawback of this model is that it is not specified when switching between the two modes should be done and how x should be chosen when sticking begins.

5. Simulations

In order to study the effects of friction on the performance of the control loop simulations have been made in the OmSim simulation environment.

5.1 Preliminaries

Integration method

The model is written on the form of differential algebraic equations, DAEs, with differential equations describing the dynamics of the system and algebraic equations for constraints. The simulator supplies two DAE-solvers, DASRT, which is an improved version of DASSL equipped with a root-finding algorithm, and Radau5, an implicit Runge-Kutta algorithm. For simulations of models with events DASRT is the only possible choice since the root-finder is necessary for computing the time when events occur. To be consequent DASRT has been used for all the simulations.

Initial values

To start a simulation the state variables of the model must fulfill the constraints given by the algebraic equations. Initial values of the state variables must be precalculated to make the residuals of the equations sufficiently small. When the simulation is started an algebraic equation solver is invoked to calculate the initial values of all the variables in the model with the desired accuracy.

The model of the control loop has up to 22 state variables depending on the friction model used. These are chosen by OmSim from the DAEs given by the model code and are for example the states of the actuator chambers, control pressure, positions, and velocities of the positioner–actuator and the integral value of the controller. The simulations are started close to a stationary point. Then all velocities will be zero and other state variables will be easy to calculate. The operating point of the system is defined by the mean values of the signals in Figure 5.1. The normalised output is $y = 0.43$ which corresponds to a fibre concentration of 3%. The normalised control signal is $v = 0.1$. From these values many of the initial values can be calculated. It is more difficult to determine the pressures in the actuator chambers. Test simulations showed that after initial transients had disappeared the mean pressure in the chambers approached $p \approx 600\,\text{kPa}$. The temperature in the chambers is taken as 303 K. To disturb the system slightly the initial value of the measured output is set to $y = 0.4$.

Parameter values

The parameter values of the friction models cannot be determined without detailed experimental studies. Since the application is not a lab setup this is not possible. Reasonable parameter values have been chosen as nominal values to be used in the simulations. Friction in a pneumatic actuator amounts to about 20% of the maximum load of the actuator. The maximum load of the S21 actuator is 5-6 kN depending on the working direction. Therefore the

Coulomb friction parameter is set to 1000 N. Static friction is chosen to be 50% higher than Coulomb friction. The nominal parameters of the other friction models are chosen somewhat arbitrarily according to recommendations in the references, mainly [Armstrong-Hélouvry, 1991]. The parameter values of the different friction models are listed in Table 2. The parenthesis around the parameters for the Bristle model indicate that these are not explicit parameters.

Model	F_s N	F_c N	F_v Ns/m	v_S m/s	K_c N/m	others
Classical	1500	1000	0	–	–	–
Dahl	–	1000	–	–	10^8	–
ext. Dahl	1500	1000	0	0.0002	10^8	$d = 9000\,\mathrm{Ns/m}$
Bristle	(1500)	(1000)	–	–	10^7	$\Delta = 7.5 \cdot 10^{-5}\,\mathrm{m}$
Seven par.	1500	1000	0	0.0002	10^8	$\tau_L = 0.02\,\mathrm{s}$, $\gamma = 1\,\mathrm{s}$

Table 2 Nominal parameter values used in the simulations.

5.2 Results

The simulations of the control loop are made using the different friction models and the parameter values of Table 2 if no other value is specified. The simulations will make it possible to make comparisons of the friction models and study the effects of different frictional phenomena on the performance of the control loop. The positioner model used is the SP300. This makes it possible to compare the model behaviour with the recorded data in Figure 5.1.

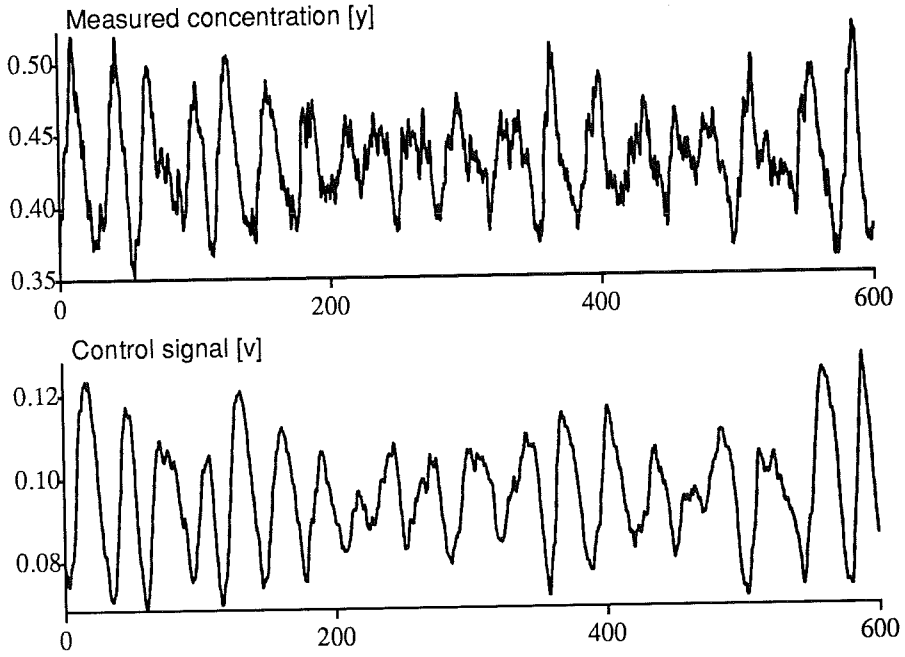

Figure 5.1 Concentration fluctuations during operation of the control loop.

The classical friction model

As a starting point a simulation of the system using the classical friction model with the nominal parameter values will be described in detail. Then the other simulations will be compared to this. The simulation recordings are taken after 100-200 seconds when the initial transient has disappeared. The variables plotted are: the normalised output of the concentration measurement, the normalised valve opening, the normalised control signal from the PI-controller and the friction force. The control signal can be seen as a reference to the positioner and is plotted together with the valve position.

The result of the simulation is shown in Figure 5.2. The output looks like a series of step responses and this is indeed the case. On this time-scale the valve moves almost instantly between different positions. The detail of the friction force shows that the slipping period is approximately a hundredth of a second. Since friction force changes instantly from F_s to F_c the accelerating force on the actuator piston will be very big compared to the inertia. But as soon as the piston begins to move the driving force decreases because of an equalisation

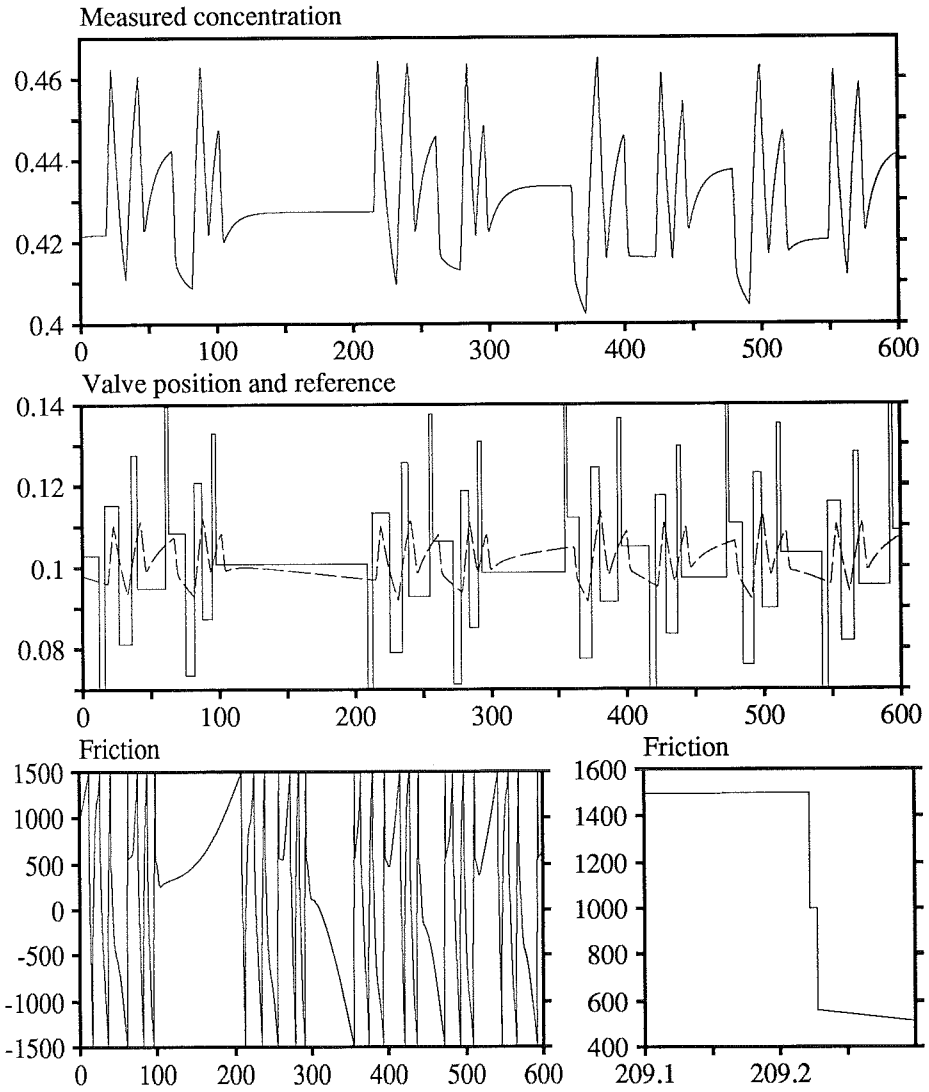

Figure 5.2 Typical result of a simulation run. In this simulation the classical friction model was used. The detail on the right shows the friction force during a slipping period.

of the pressures in the actuator chambers. When the driving force becomes smaller than F_c the piston slows down until it sticks. The friction force then becomes equal to the driving force. The behaviour is similar to the one seen in Figure 5.1 although the amplitude of the fluctuations is smaller. Another thing noted are the long, calm stretches at $t = 100\,\mathrm{s}$ and $t = 300\,\mathrm{s}$. When the valve sticks close to the desired position it will remain there for a longer period of time while actuator force builds up. This is a very slow process since the position error, which determines the pilot valve opening in the positioner, is small. The driving force is counteracted by static friction until it reaches F_s when the valve moves and the rapid fluctuations restart.

The magnitude of friction in the actuator and the control valve is uncertain. But the behaviour will be similar even if the friction levels are halved, see Figure 5.3. The fluctuations are a little slower and the amplitude is smaller.

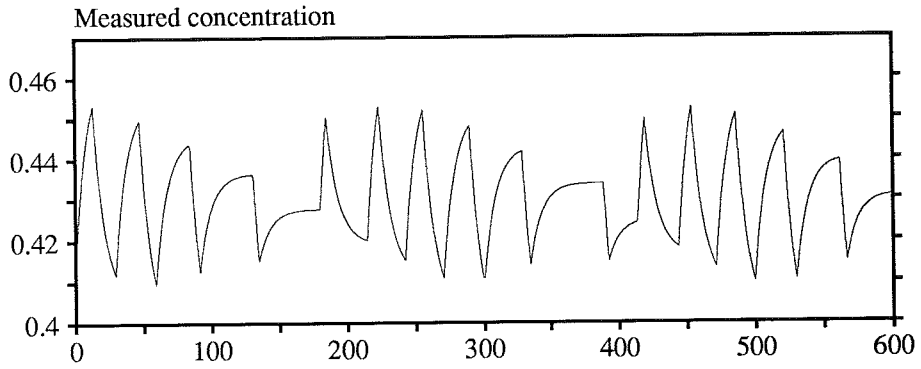

Figure 5.3 The classical friction model with $F_c = 500\,\mathrm{N}$, $F_s = 750\,\mathrm{N}$.

A different behaviour is seen when the friction parameters are equal, $F_c = F_s = 1000\,\mathrm{N}$, Figure 5.4. The concentration fluctuations are smoother,

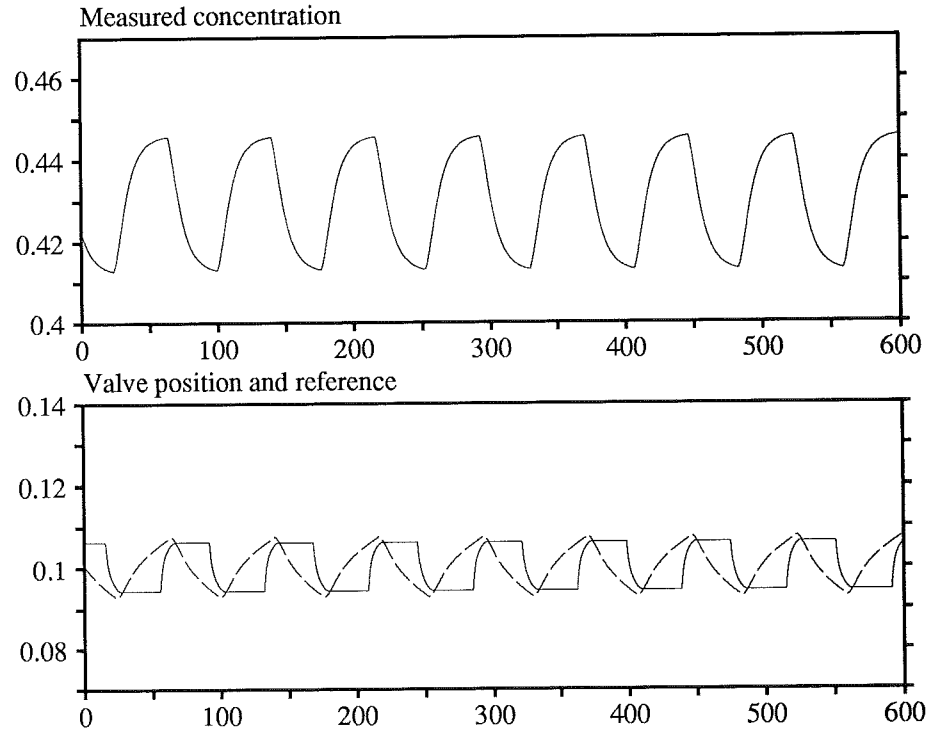

Figure 5.4 The classical friction model with $F_c = F_s = 1000\,\mathrm{N}$.

much slower and not irregular. The motion of the valve is slower because the accelerating force is smaller. When slipping starts the friction force does not decrease. It remains the same until slipping stops and the valve sticks. It is interesting to note that the presence of higher static friction is not necessary for the fluctuations to continue. Merely Coulomb friction is enough.

The addition of viscous friction will also have a smoothing effect like the one seen when $F_s = F_c$. This is due to the fact that the mean friction during slipping will be larger, closer to F_s. But the magnitude of viscous friction has to be large for the effects to be visible, as large as F_v =1000 Ns/m. The simulation with viscous friction is shown in Figure 5.5. When viscous friction is even larger, $F_v \geq$2000 Ns/m, the behaviour will approach that of Figure 5.4.

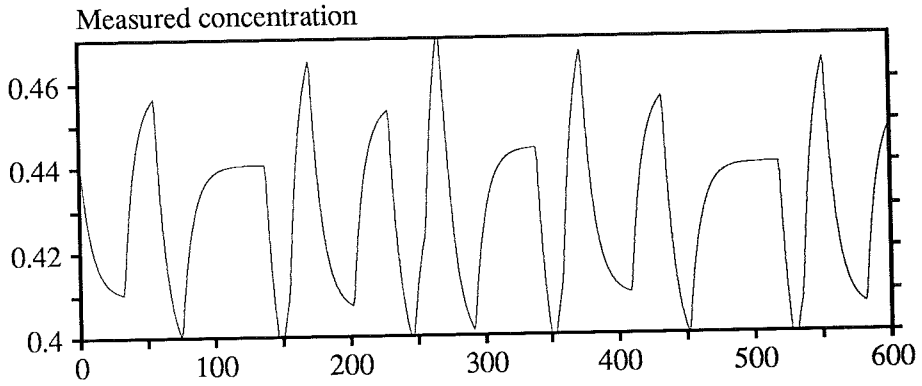

Figure 5.5 The classical friction model with F_v =1000 Ns/m.

As implemented here the classical friction model is very well-suited for simulation. This requires that the transition from sticking to slipping is handled by events. The simulation will be efficient since the DAE-solver can use large time steps for the solution between events, the length of the steps will be dictated mostly by the plotting accuracy wanted.

The Dahl model

The original Dahl model only contains Coulomb friction. When this model is simulated with the nominal spring constant, $K_c = 10^8$ N/m, the behaviour resembles that of the classical model with $F_s = F_c$, see Figure 5.6. This choice of spring constant corresponds in a linear approximation to a displacement of $K_c/F_c = 10\,\mu$m, which is a reasonable value. Because of the dynamics of the

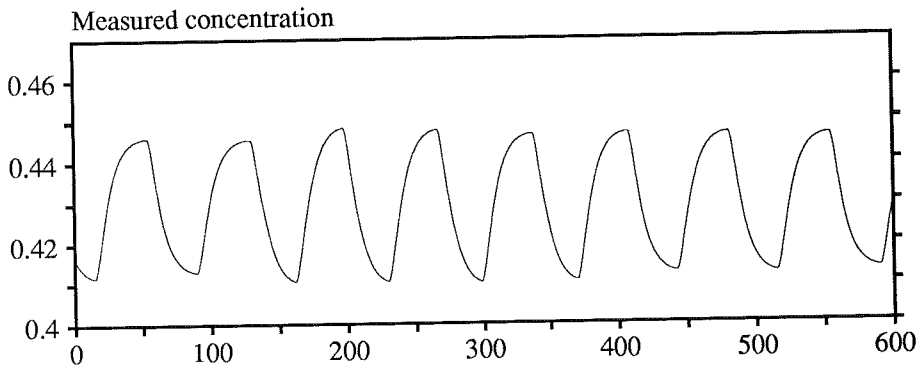

Figure 5.6 The Dahl model with $K_c = 10^8$ N/m.

model, the displacement can be up to five times this value. If the model is simulated with a softer spring, $K_c = 10^7\,\mathrm{N/m}$, the fluctuations disappear almost completely after 600 seconds, see Figure 5.7 where the initial transient is included. The displacement in this case is 0.1-0.5 mm, a rather large displacement. The full stroke of the actuator piston is 8 cm and the typical movement here is 1-2% of the stroke, i.e. ≈1mm. This means that nearly all the motion is non-sliding.

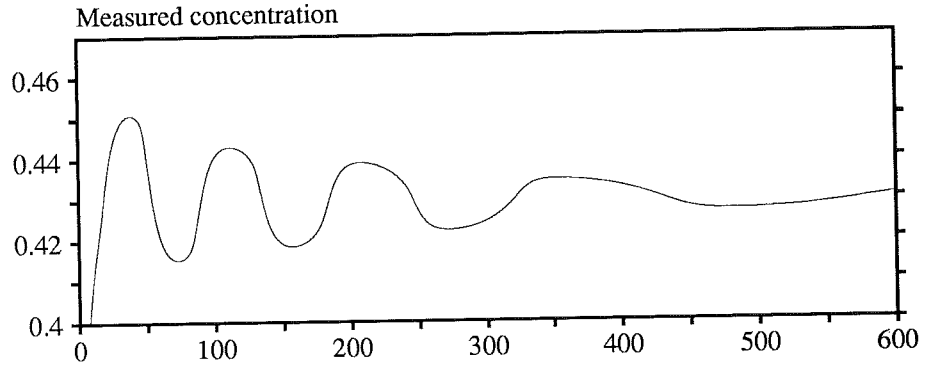

Figure 5.7 The Dahl model with $K_c = 10^7\,\mathrm{N/m}$.

The Dahl model with a large spring constant is a stiff problem and can be difficult to handle for the simulator. The solution to the DAE-system changes very slowly most of the time, but when the valve moves it requires very short time steps to give the desired accuracy. For the simulations with DASRT to work the plotting interval has to be set very small, which makes the simulation inefficient. There are however other integration methods for stiff problems available which perform better and since the Dahl model requires no event handling the choice of DASRT is not necessary.

The extended Dahl model

A simulation with the extended Dahl model with nominal parameter values shows a behaviour resembling very much that of the classical friction model, see Figure 5.8. The fluctuations are perhaps a little slower due to the damping included through the parameter d. The addition of Stribeck friction also has a damping effect although this is hardly visible when v_S is as small as 0.0002 m/s. When the Stribeck velocity is higher the effects are clearly visible, see Figure 5.9. The fluctuations are alternatingly slow and fast. When the difference between valve position and reference is large the valve still jumps

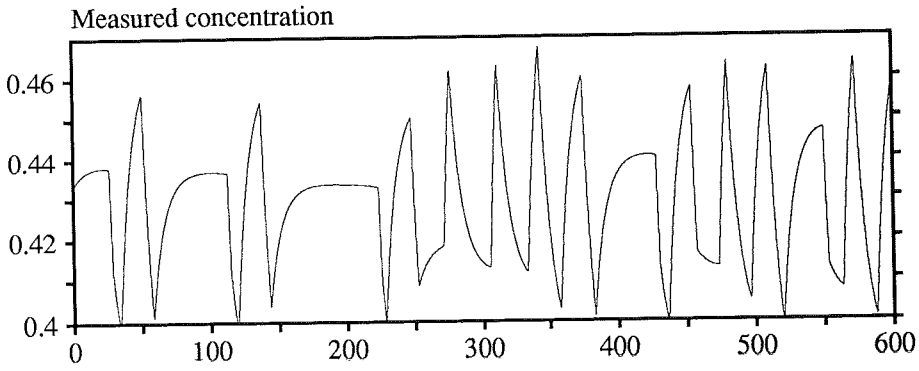

Figure 5.8 The extended Dahl model, $F_c = 1000\,\mathrm{N}$, $F_s = 1500\,\mathrm{N}$.

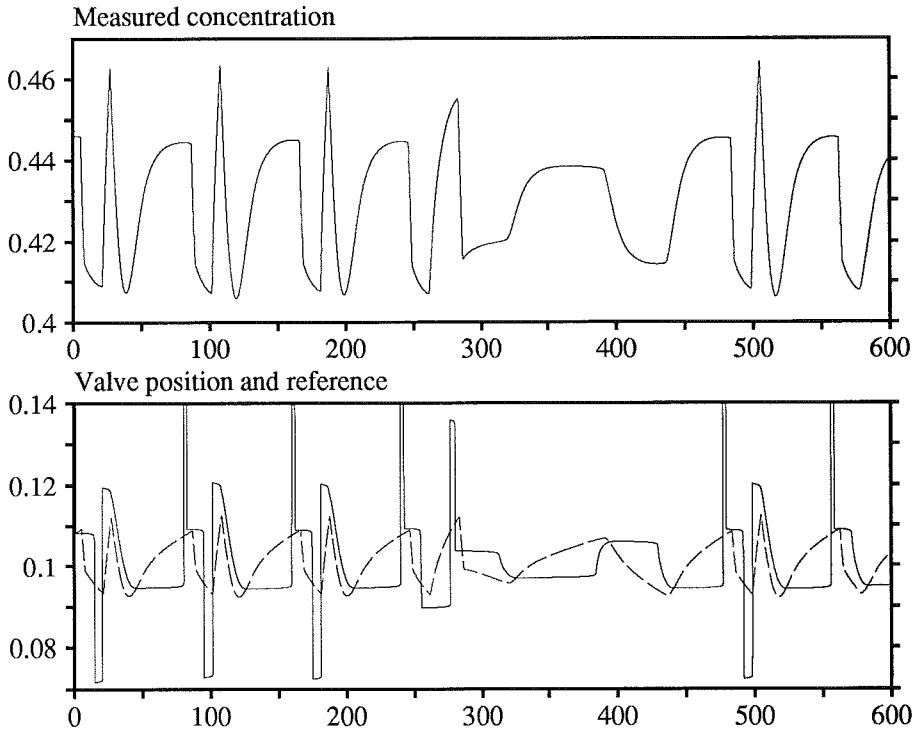

Figure 5.9 The extended Dahl model, $v_S = 0.005\,\mathrm{m/s}$.

quickly to a new position. The difference from the case of no or little Stribeck friction is that the valve does not need to come as close to the desired position for the movement to be smooth. When the Stribeck velocity is very high, $v_S \geq 0.05\,\mathrm{m/s}$, the smoothing effect will cause a behaviour like that of the Dahl model. The velocity of the actuator piston will then never exceed the Stribeck velocity.

To use the extended Dahl model in simulation is similar to the Dahl model. The damping in the model makes it a little easier to use though.

The Bristle model

Simulations with the Bristle model yields results similar to the ones observed before. It can give results slightly more irregular than the other models but there are no fundamental differences. The simulation in Figure 5.10 is made with a total spring constant $K_c = 10^7\,\mathrm{N/m}$ (The spring constant of each bristle is $10^6\,\mathrm{N/m}$). Using $K_c = 10^8\,\mathrm{N/m}$ as in the simulations of the other models would be desirable but the simulation time would be more than tenfold.

From the plot of the friction force during slipping in Figure 5.10 it is not hard to see why this model is laborious to use in simulations. During a slipping period the model will produce several hundred events a few microseconds apart. This will slow down the simulation enormously. There is also a problem when motion stops. The spring-like bristles will generate undamped oscillations in the friction force unless the bristle positions are chosen to make the sum of bristle forces equal the driving force. This makes it necessary to have the driving force, u, as an input to the friction model, as was the case with the classical friction model.

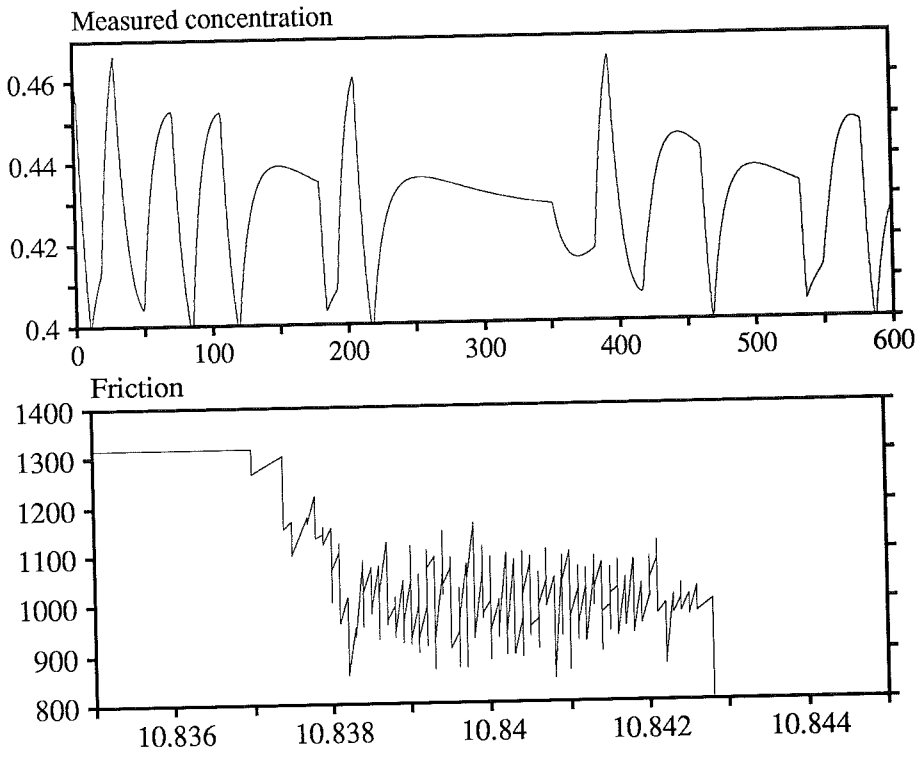

Figure 5.10 The Bristle model with $K_c = 10^7$ N/m.

The Seven parameter model

The simulation of the Seven parameter model gives a surprisingly regular result as seen in Figure 5.11. It looks like the system approaches a stable limit cycle.

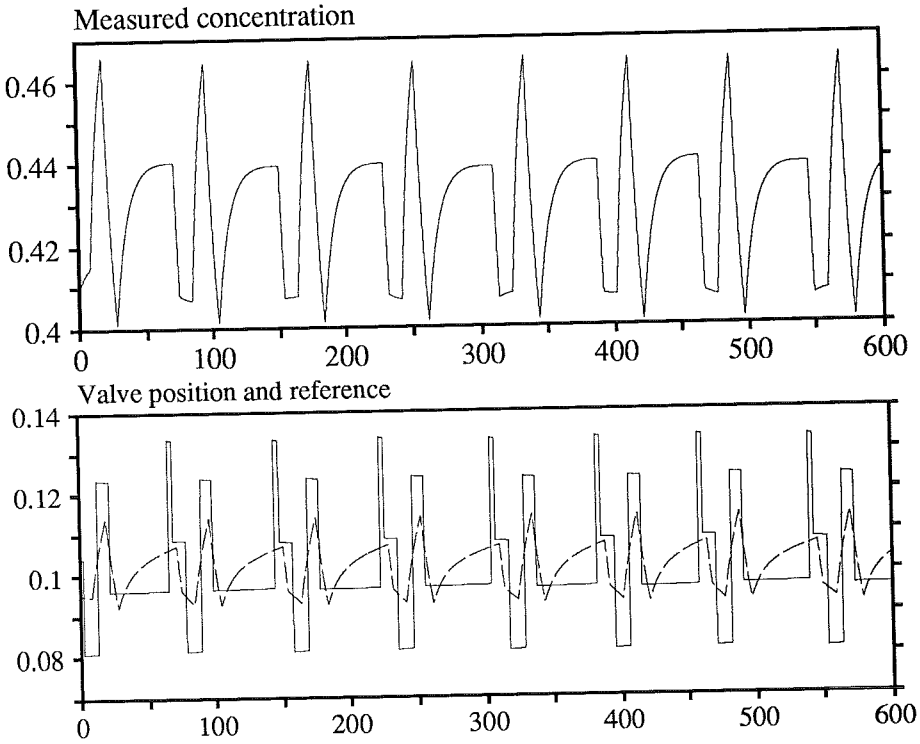

Figure 5.11 The Seven parameter model with $K_c = 10^8$ N/m.

Besides this the differences from the previous simulations are small.

The model is not at all suited for simulation. It has two modes, sticking and sliding. The switching between these modes is not specified. In this implementation sticking occurs when the velocity falls below a certain zero-velocity. The model in sticking is an undamped spring. The displacement, x, in (4.1) must be chosen as

$$x = \frac{u}{K_c}$$

otherwise there will be undamped oscillations in the friction force. This model too requires knowledge of the driving force, u. Also, the use of a physical delay, τ_L, makes the system difficult to simulate. In OmSim, the delay function uses interpolation between values stored for plotting. To be able to catch the velocity peaks when the position changes quickly the plotting interval has to be smaller than $\tau_L/2 = 0.01\,\mathrm{s}$. Since the DAE-solver stops at every plotting interval the simulation will be very inefficient. The simulation in Figure 5.11 took four hours to perform on a SUN SPARCstation 2. A simulation with the classical model takes about fifteen minutes to perform.

Effect of varying process gain

As stated in Section 2 the gain of the process is about three times. This is of course not a very exact figure. There are a lot of uncertainties when making identification experiments on an industrial application. Too see how the gain affects the observed concentration fluctuations, simulations were made with the extended Dahl model and 60% higher process gain. The Stribeck velocity is $v_S = 0.005\,\mathrm{m/s}$. In Figure 5.12 the results are shown. It can be seen that the amplitude is bigger as can be expected, but the fluctuations are also more irregular and of higher frequency. The behaviour is very similar to the observed

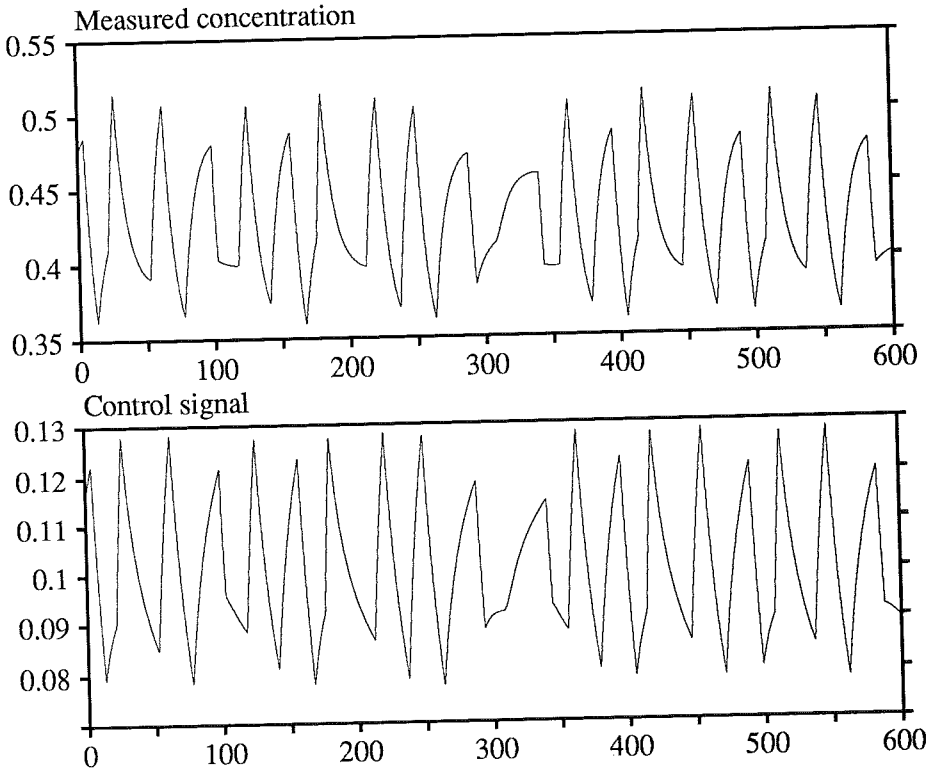

Figure 5.12 The system simulated with higher gain.

behaviour of Figure 5.1.

The process gain is high mainly because the control valve is much larger than necessary. If a smaller valve was used which could operate at 50% valve opening the gain would decrease with 60-80%. Then the amplitude of the concentration fluctuations would be smaller and since the gain in the controller could be increased the positioner–actuator would be less sensitive to friction and backlash. Figure 5.13 shows a simulation where the gain has been reduced with 80%. The controller parameters have also been altered to match the new process parameters. The problem with the valve moving back and forth remains but the lower gain makes the concentration fluctuations a lot smaller. The simulation indicate that if a smaller valve could be used in practice, the problem would be reduced to a minimum. To stop the valve chattering the positioner need to be redesigned with a deadzone. A deadzone in the pilot valve would stop the actuating force from growing when the position error is small. The PI-controller in the loop also has to be changed to allow for small concentration errors without increasing the control signal.

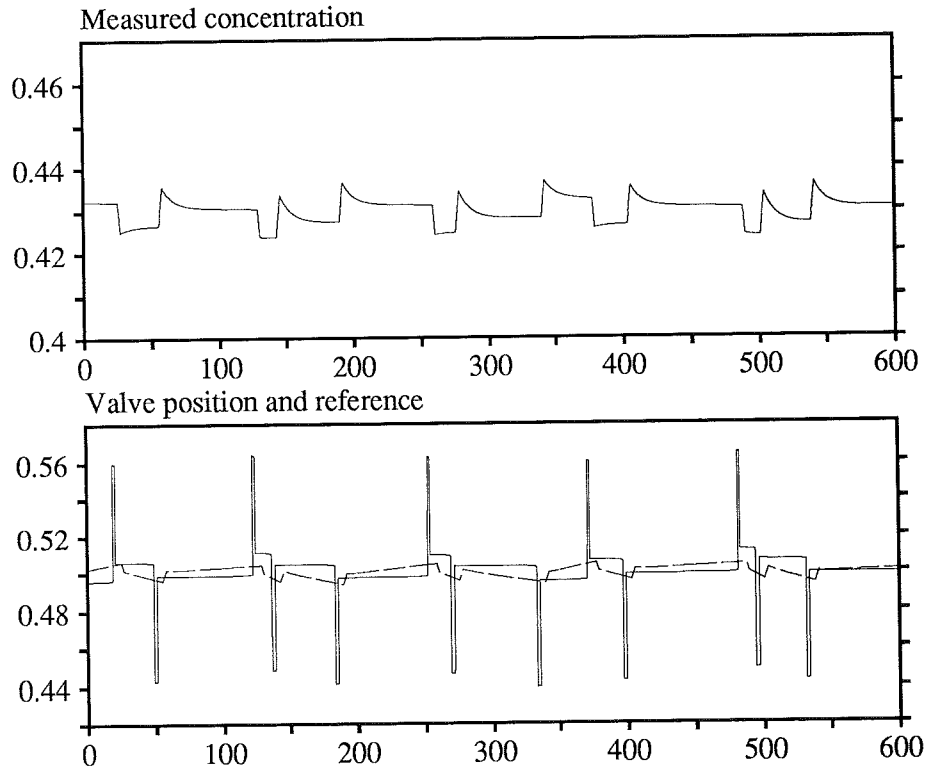

Figure 5.13 The system simulated with a smaller valve and hence lower gain.

Simulations of the SP400 model

An interesting point with the simulations would be to see if the concentration fluctuations would be smaller if the newer positioner, SP400, is used instead of the SP300. A simulation of the control loop with the SP400 positioner using the extended Dahl model and the nominal parameters used before show that the concentration fluctuations become very small, see Figure 5.14. They would hardly be visible if measurement noise was added. But the valve is still moving back and forth, even more violently since the positioner is faster. Valve chattering like this could seriously damage the equipment in the long

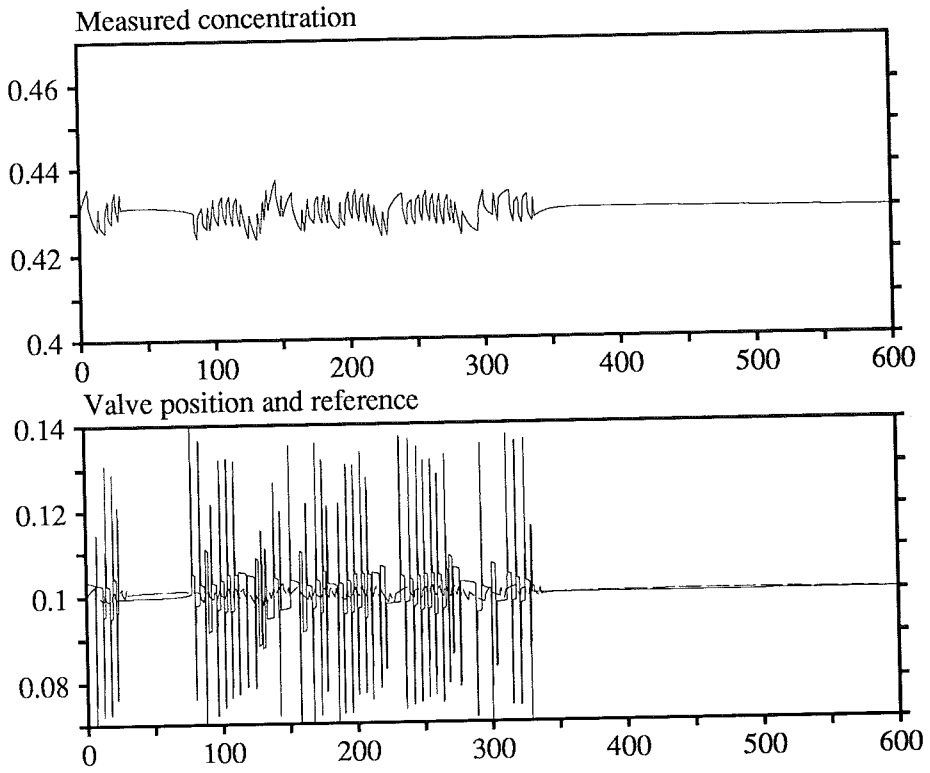

Figure 5.14 The system simulated with the SP400 model and extended Dahl friction.

run. Large alternating forces on the parts could result in wear and backlash and this could also cause a behaviour resembling the observed.

At t =340 s the valve sticks at a position very close to the desired one. The corresponding distance of the piston position from the desired piston position is smaller than the possible displacement, $F_s/K_c = 15\,\mu$m. In the simulation the valve will then remain stuck in this position. In reality this would hardly be the case though, load disturbances and measurement noise would make this operating point unstable and stop the valve from remaining there for any longer periods of time. This indicates that using the SP400 positioner would not solve the problem.

5.3 Experiences using OmSim

The working environment Omola/OmSim is easy to learn. Developing models and simulating them has worked smoothly and quickly most of the time. Large and complex models are easily handled with the object-oriented structure. The simulator in OmSim has been very easy to use. Minor flaws in OmSim encountered while doing this work have been corrected most swiftly. However, a few problems could be mentioned.

Firstly, when large models are assembled it is common that a redundant equation is specified by mistake by the user. The model compiler then informs that an equation system is overdetermined and presents a list of all the system equations of which only one should be deleted. Because of the large number of equations in the list it is of almost no help in determining which equation is redundant.

Secondly, initial values have to be specified to be able to start a simulation. In some cases this means that the user has to solve a set of nonlinear equations. If very high accuracy in the simulation is required the simulation can not be started with this accuracy. An initial step has to be made first with lower accuracy and then the accuracy can be raised one step at a time.

Lastly, when the state variables are chosen this is done by the model compiler from the equations of the model. Even a minor modification in the model can however change the set of state variables. This means that different initial values have to be specified. The values used before the modification may be partly useless and even overridden by incorrect default values.

6. Conclusions

This report has described the modelling of a concentration control loop in a paper mill. The major effort was made to model the nonlinear behaviour of the pneumatic positioner–actuator. A library of several friction model implementations have also been developed. The models have been developed using Omola, an object-oriented modelling language. The friction models have been used in combination with the control loop model in simulations to study the effects of friction on the control loop behaviour. The simulations have been made in the OmSim working environment and show that the observed behaviour of the control loop with large variations in concentration can be reproduced by the model. It is noteworthy that friction alone can cause this behaviour. A common opinion in the industry is that backlash has to be present to create the observed oscillations. Even the use of a simple friction model with only Coulomb friction causes an oscillating behaviour.

The effect of the process gain on the behaviour of the control loop has also been studied. The large gain caused by using a big valve at small valve openings is one part of the explanation to the large concentration variations. The simulations indicate that using a smaller valve operating at larger openings would reduce the amplitude of the variations. Combining this with deadzones in the positioner and PI-controller would probably solve the control problem at the cost of a small constant concentration error.

Two different positioner models have been simulated. Besides the positioner used in the control loop today a model of a newer positioner has been used. The simulations indicate that even though changing positioner would reduce the concentration variations this is not a good solution. Valve chattering would create new problems.

The friction models are to varying extents suited for simulation. The classical friction model in its original formulation is difficult to use. However, an implementation with events is efficient and fast to use in simulations. This requires that an integration method capable of handling events is available. The requirement to have the driving force as an input to the model besides velocity is an additional drawback. The Dahl model has the appealing quality of being simple due to the fact that it is dynamic. It requires no event-handling or mode-switching and the code becomes very simple. The same holds for the extended Dahl model which is more usable as it contains all the important frictional phenomena. The only drawback is that simulating the dynamic friction behaviour requires a DAE-solver that can handle stiff problems without using unnecessarily short time steps in the solution. The Bristle model and the Seven parameter model are not at all suited for simulation. They both require long simulation times and the driving force as a model input.

References

ANDERSEN, B. W. (1967): *The Analysis and Design of Pneumatic Systems.* John Wiley & Sons.

ARMSTRONG-HÉLOUVRY, B. (1991): *Control of Machines with Friction.* Kluwer Academic Publishers, Boston, Ma.

BACKÉ, W. and O. OHLIGSCHLÄGER (1989): "A Model of Heat Transfer in Pneumatic Chambers." *The Journal of Fluid Control*, **20:1**, pp. 61–78.

CANUDAS, C., H. OLSSON, K. J. ÅSTRÖM, and P. LISCHINSKY (1994): "A New Model for Control of Systems with Friction." *IEEE Transactions on Automatic Control.* To appear.

DAHL, P. (1977): "Measurement of solid friction parameters of ball bearings." In *Proc. of the 6th Annual Symposium on Incremental Motion, Control Systems and Devices.* University of Illinois.

HAESSIG, D. and B. FRIEDLAND (1990): "On the Modeling and Simulation of Friction." In *Proceedings of the 1990 American Control Conference,San Diego*, pp. 1256–61.

KARNOPP, D. (1985): "Computer Simulation of Stick-Slip Friction in Mechanical Dynamic Systems." *Journal of Dynamic Systems, Measurement, and Control*, **107**, pp. 100–103.

MATTSSON, S. E., M. ANDERSSON, and K. J. ÅSTRÖM (1993): "Object-oriented modelling and simulation." In LINKENS, Ed., *CAD for Control Systems*, chapter 2, pp. 31–69. Marcel Dekker Inc, New York.

PANTZARE, J. (1993): STORA Technology. Private communication.

WANGEBY, P. (1993): SOMAS. Private communication.

A. Pneumatic theory

Taking the step into the world of pneumatics I have realised that it really consists of two separate worlds; the theorist's world, where from Navier-Stokes equation and general thermodynamics fluid flow calculations are made that are very difficult to follow; and the practicalist's world, that is interested mainly in pneumatic machinery and where all equations are linearised as soon as possible. This has given me some difficulties since my area of interest was a little of both. To simulate pneumatic machinery I needed fairly simple equations for small scale components, e.g., pipes and volumes of air, but since the equation solver of the simulator works just as well with nonlinear equations, linearising would be to oversimplify the problem.

A.1 Flow equations

Air at normal temperature and moderate pressure can be well approximated as an ideal gas. The equation for mass flow, W, of an ideal gas is, [Andersen, 1967],

$$W = Ap_1 \left\{ \frac{2\gamma M}{(\gamma - 1)RT_1} \left[\left(\frac{p_2}{p_1}\right)^{\frac{2}{\gamma}} - \left(\frac{p_2}{p_1}\right)^{\frac{\gamma+1}{\gamma}} \right] \right\}^{\frac{1}{2}} \tag{A.1}$$

where A is the cross-section area, γ is the ratio of specific heats, M is the molar mass, R is the gas constant, p_1 and T_1 is total pressure and absolute temperature respectively and p_2 is the local pressure of the flowing gas. This relation is valid at any point in a flowing gas, but it is not practically useful since the total and local pressures are difficult to work with. According to Andersen some assumptions can be made that makes it possible to use (A.1) for flow calculations at restrictions.

- Total pressure, p_1, can be taken as upstream static pressure if the approach velocity is small. The error is less than one percent if the ratio of upstream cross-section area to restriction area is more than five.
- Total absolute temperature, T_1, may be taken as upstream static temperature under the same condition as the above.
- Local pressure, p_2, is downstream static pressure if recovery of dynamic pressure is negligible. This means that all kinetic energy is lost in the expansion after the restriction. This is usually the case if cross-section area increases abruptly.
- Because the flow through a restriction contracts further after passing the restriction, A in (A.1) has to be replaced by effective area, $A_e = C_d A$. The discharge coefficient, C_d, is the ratio between minimum flow area and restriction area.

Under these assumptions Relation A.1 with A replaced by $C_d A$ is valid for calculations of subsonic flow through the restriction. When the ratio between p_2, the pressure downstream from the restriction, and p_1, the pressure upstream, falls below a certain critical pressure ratio the flow becomes sonic, choked. This means that the flow velocity at the minimum flow area has reached the speed of sound. Lowering p_2 further will not result in a larger mass flow. The mass flow remains constant at constant upstream pressure and the flow equation

becomes

$$W_{cr} = C_d A p_1 \left\{ \frac{\gamma M}{RT_1} \left(\frac{2}{\gamma+1} \right)^{\frac{\gamma+1}{\gamma-1}} \right\}^{\frac{1}{2}}$$

The critical pressure ratio is a function of γ given by

$$\left(\frac{p_2}{p_1} \right)_{cr} = \left(\frac{2}{\gamma+1} \right)^{\frac{\gamma}{\gamma-1}}$$

For air the critical pressure ratio is 0.53 which corresponds to $\gamma = 1.4$.

A.2 Air expansion

Expansion of air in a chamber is another topic in pneumatics. Different cases exist, depending on if volume and/or mass is variable. For a chamber in the actuator studied here the most general case holds: variable volume and variable mass. A common assumption made for expansion of an ideal gas is then that it follows the polytropic equation

$$p \left(\frac{V}{m} \right)^n = constant$$

where n is the polytropic exponent. The value of n can be chosen between $n = 1$ for isothermal expansion and $n = \gamma$ for adiabatic expansion. The true expansion function falls somewhere in between these extremes. A heat transfer model exists that describe measurements of charging and discharging a pneumatic chamber well [Backé and Ohligschläger, 1989]. This model requires knowledge of several parameters that are not available here, some of them which have to be measured. Since the thermal energy content of air is low compared to that of the cylinder and piston, it is reasonable to assume that the cylinder remains at constant temperature. Thus, as the results of Backé and Ohligschläger indicate, the transient behaviour during charging or discharging of a volume is close to adiabatic while the steady-state behaviour is isothermal. Only isothermal expansion is used throughout the simulations. There are three reasons for this.

- Charging and discharging is slower in this application than in the article by Backé and Ohligschläger.
- The actuator is working exclusively near the closed position. This causes the bigger changes in pressure in the chamber with the smallest volume, hence the smallest volume to area ratio and the fastest heat transfer.
- According to the manufacturer, there is dead-time in the positioner. As can be seen in Figure A.1 the choice of expansion behaviour will mainly affect the dead-time of the positioner. Using isothermal expansion results in a behaviour closer to the measured data.

A.3 Transmission lines

As was explained in Subsection 2.2 there is a lag in the transmission line between control signal and the pressure signal to the positioner. A relation describing this lag has been derived [Andersen, 1967]. The flow equation of the transmission line can be solved under a few assumptions.

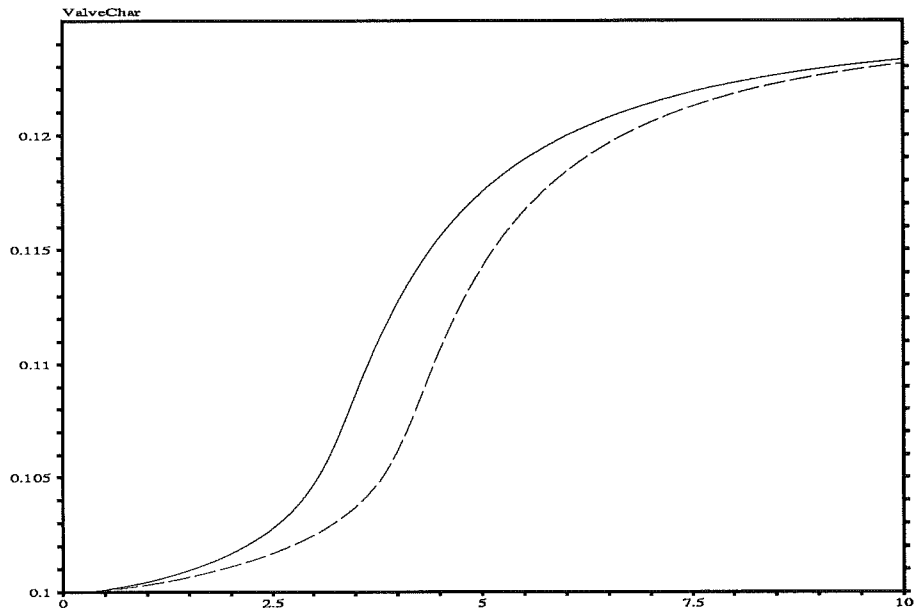

Figure A.1 Step response of positioner–actuator under the assumption of adiabatic (—) and isothermal (- -) expansion.

- Flow in the line is assumed to be laminar. This gives a parabolic velocity profile.
- A one-dimensional flow equation is used.
- The polytropic gas equation is used to account for heat transfer.

To solve the flow equation and derive an expression for the lag in the transmission line the pressure function is assumed to have an exponential time dependence

$$p(y,t) = p(y)e^{st} + p_0$$

where p_0 is the mean pressure in the line and y is the position along the line. The solution can be expressed in hyperbolic functions of a function $\alpha(s)$. For small s, i.e. at low frequencies, power series expansions of the hyperbolic functions can be used. Truncating these series for powers of s higher than two renders us the second-order transfer function

$$\frac{p_2}{p_1} = \frac{\omega_n^2}{s^2 + 2\zeta_n\omega_n s + \omega_n^2} \tag{A.2}$$

with

$$\omega_n = \frac{\beta}{\sqrt{\frac{1}{2} + \frac{\overline{V}}{AL} + \frac{E_2^2}{12}\left(\frac{1}{2} + 2\frac{\overline{V}}{AL}\right)}}$$

$$\zeta_n = \frac{1}{2}\left(\frac{1}{2} + \frac{\overline{V}}{AL}\right)\frac{E_2}{\beta}\omega_n$$

where $\overline{V}$ is the effective terminal volume and β and E_2 are calculated parameters. In this application the terminal volume is very small compared to the line volume AL. Neglecting this term gives the simplified expressions for the natural frequency and damping

$$\omega_n = \frac{\beta}{\sqrt{\frac{1}{2} + \frac{E_2^2}{24}}} \qquad \zeta_n = \frac{E_2}{4\beta}\omega_n$$

with the parameters

$$\beta = \frac{1}{L}\sqrt{\frac{nRT}{M}} \qquad E_2 = 8\pi\frac{\eta\beta L^2}{p_0 nA}$$

where L is line length, η is the viscosity of air at temperature T, n is the polytropic exponent above and A is cross-section area. All of the equations used here are of course originally expressed in pounds, inches etc. Translating these to SI-units was tedious work and for readers interested in American pneumatic literature it could be noted that

$$R_a T_a = \frac{RTg_a}{Mg^2}$$

where subscript a denotes values expressed in American units, e.g

$$[R_a] = \frac{\text{in.lb}}{\text{lbm.}^\circ\text{R}}$$

The step response of the transfer function (A.2) to a unit step is plotted in Figure A.2. The parameters have been evaluated for the conditions of the application.

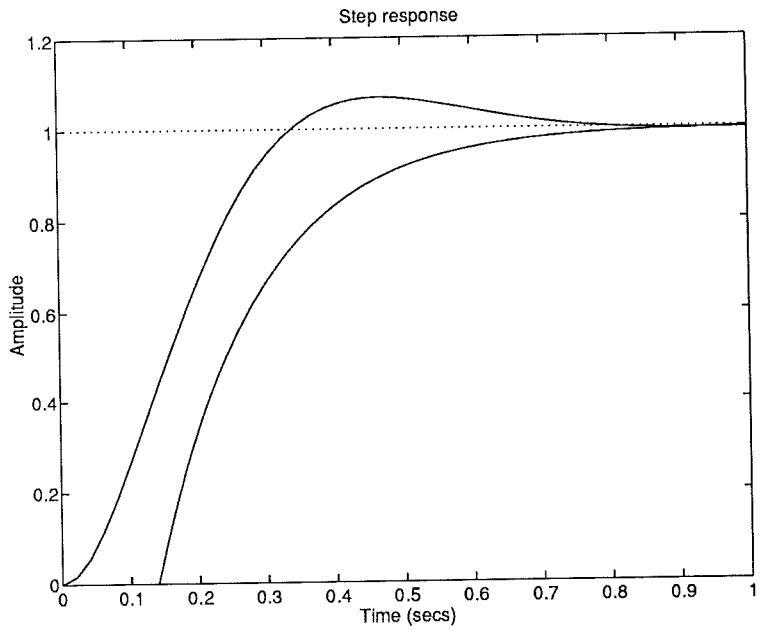

Figure A.2 Step responses of possible transfer functions for the transmission line.

Because of the length of the line, a first-order approximation could also be considered. Andersen gives expressions for the coefficients of a first-order lag with a time delay. They are

$$D_L = \beta^{-1} \qquad T_L = (\frac{1}{2} + \frac{\overline{V}}{AL})\frac{E_2}{\beta}$$

The step response of this transfer-function is also plotted in the figure. Using a first-order lag is not recommended by Andersen when $\zeta_n < 1.2$. In this case $\zeta_n = 0.65$ and the figure shows that correspondence between the two functions is poor. The delay D_L is simply the physical delay, the time it takes a wave to pass through the line travelling at the speed of sound.

B. Modelling in Omola

This appendix will give you a brief explanation of the modelling language Omola and the simulation environment OmSim. The basic structure of the Omola models and the model interfaces are also explained.

B.1 Object-oriented modelling and simulation

The Department of Automatic Control runs a research program on Computer Aided Control Engineering. Within this program an object-oriented modelling language, Omola, and a simulation environment, OmSim, has been developed [Mattsson *et al.*, 1993]. Omola contains object-oriented features such as

- modularisation
- abstraction
- inheritance and specialisation

to support model development as well as reuse of models.

What is especially appealing with Omola and object-oriented modelling in general is the intuitive approach. Components are defined, with state variables, equations and terminals; terminals are interfaces describing interaction between components. The components are then assembled to complex models. By inheritance, properties common to a group of related classes can be collected in a superclass. Doing so, three things are achieved.

- The reusability is increased when more abstract classes are used.
- Maintenance is simplified since code does not have to be repeated. This also reduces the amount of code.
- By arranging the model library in a tree structure it becomes more legible and easier to use.

The OmSim environment contains several parts besides the simulator. It includes a user friendly browser for access to the model database and a graphical model editor. The editor allows you to edit existing model classes and create new classes from submodels in your data base. In Figure B.1 is the graphical model of the control loop. The symbols in the blocks are made with a bitmap editor. Bitmaps can be included in a predefined library. This enables the user to easily develop models looking like diagrams. Compare the model with the diagram in Figure 1.2.

B.2 Omola classes

A class in Omola is created from a more generic superclass, the most generic class being `Class` in the predefined `Base` library. This class is not to be used directly. A number of classes to be used as superclasses are derived from `Base::Class`: e.g. `Base::Model` for model classes and `Base::Terminal` for terminal classes.

A model can be a complex model, which is composed of submodels, or a basic component, which is so simple that no further decomposition is possible or necessary, see Listing B.1. The behaviour of a component is described

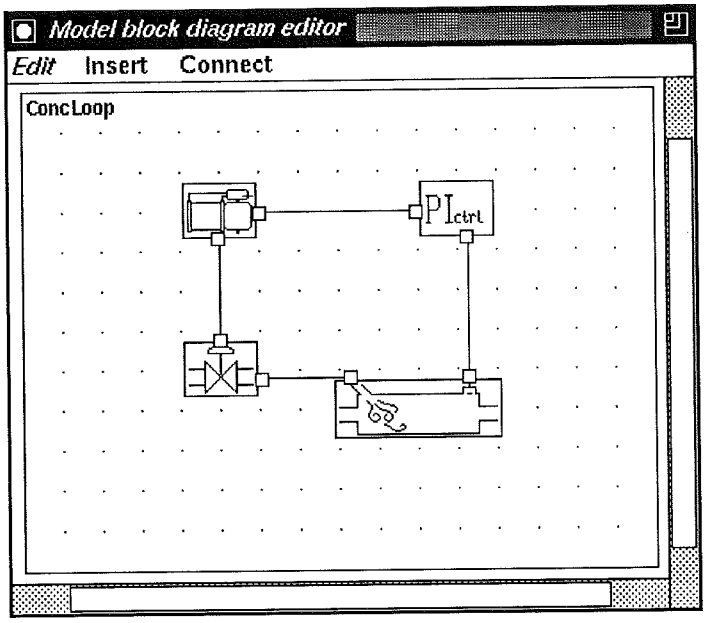

Figure B.1 The control loop model as it appears in the graphical model editor.

by algebraic and differential equations. Omola also supports discrete-event handling and difference equations. The distinction between components and complex models is of course not absolute. A submodel does not necessarily have to be a component as indicated in Listing B.1. The submodel can also be composed of other submodels and it can also have terminals, variables and equations.

```
Component ISA Superclass
WITH
    terminals
    parameters
    variables
    equations
END;
```

```
Complexmodel ISA Base::Model
WITH
    submodels
    parameters
    connections

END;
```

Listing B.1 Class definitions in Omola.

For structuring purposes a class

```
PneumaticsModel ISA Base::Model;
```

has been defined. It will be used as a superclass to all models in the libraries `Pneumatics` and `Gruvon`. Besides showing that all the models are components of the same application, they will also belong to one common inheritance tree, with `PneumaticsModel` as the root of this tree.

B.3 Omola terminals

Interaction between submodels is achieved by connecting terminals. Terminals are interfaces of one or several variables. The connection can be done graphically with the `Connect` button in Figure B.1 or directly in the class definition with the connection statement

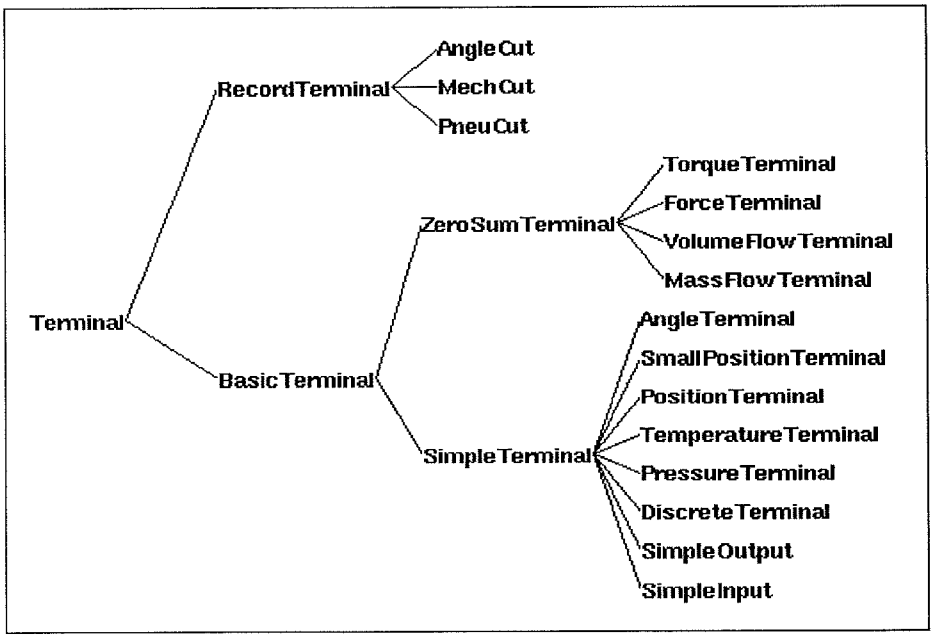

Figure B.2 Terminal structure of the `Pneumatics` library. (The classes `DiscreteTerminal`, `SimpleOutput` and `SimpleInput` are predefined in the `Base` library and not used in this report.)

`Submodel1.Terminal AT Submodel2.Terminal`

When modelling a new application and building a new model library, it is often natural to start by defining the terminal types. When the terminals needed have been defined, compatibility between submodels is easily handled. The terminal types needed in this application are pneumatic terminals for air flow, mechanical terminals describing interaction between rigid bodies and fluid flow terminals. The result will be the terminal class tree, which is shown in Figure B.2.

Mass flow rate of an ideal gas is characterised by pressure and absolute temperature. Thus, a pneumatic terminal class with three components is defined as

```
PneuCut ISA Base::RecordTerminal WITH
  p ISA PressureTerminal;
  T ISA TemperatureTerminal;
  W ISA MassFlowTerminal;
END;
```

The superclass `Base::RecordTerminal` is a predefined class with special use. A connection of terminals of any class derived from `RecordTerminal` implies that the components of the terminals should be connected. The three components of `PneuCut` are all scalar terminals.

The main terminal classes of Omola are `SimpleTerminal` and `ZeroSumTerminal`. Simple terminals are used for potential variables like electric potential, position, pressure and temperature. Zero-sum terminals are used to connect flow variables like electric current, force and flow rate. A connection of simple terminals generates an equality in the instantiated model while a connection of zero-sum terminals generates a zero-sum equation. These equations as well as equations and equalities written in the class definition do not imply any causality. Causality is determined when the model is instantiated,

compiled, into simulation code.

The components of PneuCut are defined as

```
PressureTerminal ISA SimpleTerminal WITH
  quantity := "pressure";
  unit     := "Pa";
END;

TemperatureTerminal ISA SimpleTerminal WITH
  quantity := "temperature";
  unit     := "K";
END;

MassFlowTerminal ISA ZeroSumTerminal WITH
  quantity := "mass.flow.rate";
  unit     := "g/s";
END;
```

The attributes unit and quantity are used to check consistency of connections. You can also prefix units and in a connection with different prefixes, scaling factors are automatically introduced.

To fully specify a three-dimensional mechanical cut twelve components would be needed, three positions, angles, forces and torques. For a two-dimensional cut six components are needed, two positions and forces, one torque and one angle. In a cut between rigid bodies there is however seldom full mobility. Restrictions in the mechanical construction impose constraints on the motion. In this application there are no single cuts with two degrees of freedom. Efforts to use a generic mechanical library in two dimensions did not turn out as expected. As a result of this only one-dimensional cuts are used in the models. This decision was caused mainly by index reduction problems. The restraining equations imposed by the problem to reduce the degrees of freedom created large difficulties for OmSim's index reduction procedure.

Using only one-dimensional cuts has one drawback, the distinction between linear and rotary motion. Therefore, two mechanical terminal types have been defined: `MechCut` for linear cuts with position `x` and force `Fx` and `AngleCut` for rotary cuts with angle `fi` and torque `T`. The single direction of the axis in each cut is only locally defined and not fixed globally.

The three main terminal types of this application have now been defined. They are used in the internal description of the positioner–actuator and in two of the connections in Figure B.1; controller–positioner and positioner–valve. For the connection concentration-sensor–controller a simple terminal type is used. For the measured concentration a scalar value in the range 0–1 has been used.

Only one more important terminal type remains to be described: a water flow terminal for the connection valve–mixing pipe. Water flow can thermodynamically be described by pressure and temperature in the same way as air flow, but as stated in Subsection 2.4 the water pressure is neither known nor measured. Since a linear model of the valve was made, only a scalar terminal is needed to describe water flow. As seen in Figure B.2 of the complete terminal structure a basic zero-sum terminal has been used for the terminal type `VolumeFlowTerminal`.

C. Omola listings

When the physical system described in this report is modelled it can be decomposed into three parts. There are the complex models on the higher levels in the control loop model (The three levels to the left in Figure C.1). There are also friction models and basic components (The two levels to the right in the figure). The components are pneumatic and mechanical, but since some of them are both pneumatic and mechanical, no distinction has been made between these two types. The terminal classes described in Appendix B and pneumatic and mechanical component classes belong to the library `Pneumatics`. The friction models are in the library `Friction` and the complex models on the top-most level of the control loop model belong to the library `Gruvon`.

These three model libraries are listed in C.1 through C.3.

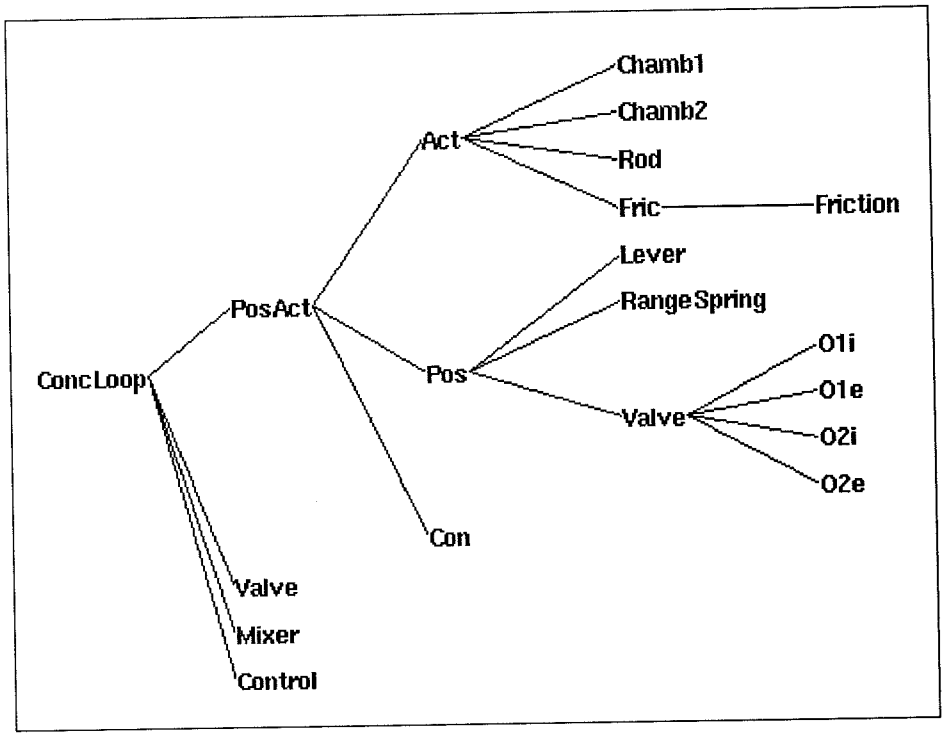

Figure C.1 The model structure of the control loop model.

C.1 Pneumatics library

```
LIBRARY Pneumatics;

USES Base;

% global constants:
gamma TYPE Real := 1.4;                    % Cv/Cp
R TYPE Real := 8314.3;                     % Gas constant J/kmolK
M TYPE Real := 29;                         % Molar mass   kg/kmol
p0 TYPE Real := 1e5;                       % norm air pressure
ps TYPE Real := 7e5;                       % supply air pressure
T0 TYPE Real := 303;                       % norm temp in Kelvin
Pi TYPE Real := 2.0*acos(0);
eps TYPE Real := 1e-6;

PressureTerminal ISA SimpleTerminal WITH
  quantity := "pressure";
  unit     := "Pa";
END;

TemperatureTerminal ISA SimpleTerminal WITH
  quantity := "thermodynamic.temperature";
  unit     := "K";
END;

MassFlowTerminal ISA ZeroSumTerminal WITH
  quantity := "mass.flow.rate";
  unit     := "g/s";
END;

VolumeFlowTerminal ISA ZeroSumTerminal WITH
  quantity := "volume.flow.rate";
  unit     := "m3/s";
END;

PneuCut ISA RecordTerminal WITH
  p ISA PressureTerminal;
  T ISA TemperatureTerminal;
  W ISA MassFlowTerminal;
END;

PositionTerminal ISA SimpleTerminal WITH
  quantity := "length";
  unit     := "m";
END;

SmallPositionTerminal ISA SimpleTerminal WITH
  quantity := "length";
  unit     := "mm";
END;
```

```
AngleTerminal ISA SimpleTerminal WITH
  quantity := "angle";
  unit     := "rad";
END;

ForceTerminal ISA ZeroSumTerminal WITH
  quantity := "force";
  unit     := "N";
END;

TorqueTerminal ISA ZeroSumTerminal WITH
  quantity := "torque";
  unit     := "N.m";
END;

MechCut ISA RecordTerminal WITH
  Fx ISA ForceTerminal;
  x  ISA PositionTerminal;
END;

AngleCut ISA RecordTerminal WITH
  T ISA TorqueTerminal;
  fi ISA AngleTerminal;
END;

PneumaticsModel ISA Model;

PointMass ISA PneumaticsModel WITH
  C ISA MechCut;
parameters:
  m ISA Parameter WITH default := 1.0;END;
equations:
  m*dot(C.x,2) = C.Fx;
END;

Beam ISA PneumaticsModel WITH
  C1, C2 ISA MechCut;
variables:
  L TYPE Real;
equations:
  C1.Fx + C2.Fx = 0;
  C2.x - C1.x = L;
END;

Link ISA Beam WITH
parameters:
  L ISA Parameter WITH default := 1.0;END;
END;

Spring ISA Beam WITH
parameters:
  k  ISA Parameter WITH default := 1.0;END;
```

```
  L0 ISA Parameter WITH default := 1.0;END;
equations:
  C2.Fx = k*(L-L0);
END;

Flapper ISA PneumaticsModel WITH
% Simplified model for small angles, angular acceleration
% neglected and sin(angle)=angle
  C1, C2, C3 ISA MechCut WITH
    x ISA SmallPositionTerminal;
  END;
parameters:
  y1, y2, y3 ISA Parameter WITH default := 0.1;END;
variables:
  angle TYPE Real;                          % Deflection in mrads
equations:
  0 = C1.Fx*y1+C2.Fx*y2+C3.Fx*y3;
% Torque balance at joint
  C1.x = y1*angle;
  C2.x = y2*angle;
  C3.x = y3*angle;
END;

Chamber ISA PneumaticsModel WITH
  C1, C2 ISA PneuCut;
  Init ISAN Event;
parameters:
  A ISA Parameter;
% Signed area taken with respect to the normal direction. +/-
  n ISA Parameter WITH default := 1.4;END;
% polytropic exponent; 1.0 = isothermal, 1.4 = isentropical
variables:
  konst TYPE Discrete Real;
  p, m, V, T ISA Variable;
equations:
  OnEvent Init DO
    new(konst) := p*(V/m)^n;
  END;
  C1.p = C2.p;
  C1.T = C2.T;
  p = C1.p;
  T = C1.T;
  m' = (C1.W + C2.W)/1000;                  % mass in kg
  p*V = m/::M*::R*T;
  p*(V^n) = (m^n)*konst;
END;

DeadEndChamber ISA Chamber WITH
  C2.W := 0;
END;

DEDiaphragm ISA DeadEndChamber WITH
```

```
  CM ISA MechCut;
equations:
  V' = -A*CM.x';
  CM.Fx = (p-::p0)*A;                          % the normal force
END;

Pipe ISA PneumaticsModel WITH
% General pipe model
  C1, C2 ISA PneuCut;
variables:
  A TYPE Real;                                 % Pipe area
equations:
  C1.W + C2.W = 0;
END;

Orifice ISA Pipe WITH
variables:
  T, Cd TYPE Real;                             % Discharge coeff.
  Wcr, Wcdel TYPE Real;                        % Critical flow
  Wsub , Wsdel TYPE Real;                      % Subcritical flow
  kvot, negkvot TYPE Real;
equations:
  kvot = C2.p/C1.p;
  T = IF kvot<1 THEN C1.T ELSE C2.T;
  negkvot = IF kvot<1 THEN kvot ELSE 1/kvot;
  Wcdel = 1000*Cd*A*sqrt(::gamma*::M/T/::R*(2/(::gamma+1))^
          ((::gamma+1)/(::gamma-1)));
  Wcr = IF kvot<1 THEN C1.p*Wcdel ELSE -C2.p*Wcdel;
  Wsdel = IF (1.0-negkvot)<::eps
          THEN 0.0
          ELSE 1000*Cd*A*sqrt(2*::gamma*::M/(::gamma-1)/::R/T
               *(negkvot^(2/::gamma)-negkvot^
               ((::gamma+1)/::gamma)));
  Wsub = IF kvot<1 THEN C1.p*Wsdel ELSE -C2.p*Wsdel;
  C1.W = IF negkvot<((2/(::gamma+1))^(::gamma/(::gamma-1)))
         THEN Wcr
         ELSE Wsub;
END;

ServoValve ISA PneumaticsModel WITH
  C1, C2 ISA PneuCut;
  O1i, O1e, O2i, O2e ISAN Orifice WITH
    Cd ISA Parameter WITH default := 0.8;END;
% 0.8 is an approximate value for sharp-edged orifices
  END;
  Supply ISA PneuCut WITH
    p := ::ps;
    T := ::T0;
  END;
  Ambient ISA PneuCut WITH
    p := ::p0;
    T := ::T0;
```

```
  END;
  CM ISA MechCut;
parameters:
  r ISA Parameter WITH
    default := 0.002;                    % land and port radius
  END;
  Kflow ISA Parameter WITH
    default := 1000;                     % flow force constant
  END;
  BendFac ISA Parameter;                 % Bend.factor of SP300
variables:
  Ao, A1, A2 TYPE Real;
  xabs TYPE Real;
equations:
  xabs = abs(CM.x)/r/BendFac;            % normed pos of valve
  Ao = IF xabs<0.5
       THEN 2*r*r*(atan(sqrt(1-sqr(1-xabs))
            /(1-xabs))-(1-xabs)*sqrt(1-sqr(1-xabs)))
       ELSE sqr(r)*(1.228+(3*::Pi/4-1.228)
            *(1-exp(-3.5*(xabs-0.5))));
  A1 = IF CM.x>0 THEN Ao ELSE 0;
  A2 = IF CM.x<0 THEN Ao ELSE 0;
  CM.Fx = Kflow*CM.x/BendFac;
  O1i.A = A1;
  O2i.A = A2;
  O1e.A = A2;
  O2e.A = A1;
connections:
  Supply AT O1i.C1;
  Supply AT O2i.C1;
  Ambient AT O1e.C2;
  Ambient AT O2e.C2;
  C1 AT O1i.C2;
  C1 AT O1e.C1;
  C2 AT O2i.C2;
  C2 AT O2e.C1;
END;
```

C.2 Friction library

```
LIBRARY Friction;

USES Base;

% global constants:
Pi TYPE Real := 3.141592654;
eps TYPE Real := 1e-6;
False TYPE DISCRETE Integer := 0;
True TYPE DISCRETE Integer := 1;

FrictionModel ISA Model WITH
  u ISA SimpleInput WITH
    default := 0;
  END;
% The driving force u is needed for the classical, bristle and
% seven parameter models
  Stick ISAN EventOutput;
parameters:
  Fc, Fs, Fv, Kc, d, vStrib ISA Parameter;
% Coulomb, static and viscous friction; stiffness, damping
% and Stribeck velocity
variables:
  F, v ISA Variable;
END;

FrictionForce ISA Model WITH
  Friction ISA FrictionModel;
  C ISA Pneumatics::MechCut;
  StickOut ISAN EventOutput;
  StickIn ISAN EventInput;
equations:
  Friction.v = C.x';
  C.Fx = Friction.F;
  OnEvent StickIn CAUSE StickOut;
connections:
  Friction.Stick AT StickIn;
END;

EventFric ISA FrictionModel WITH
  UnStick ISAN Event;
variables:
  Stuck TYPE DISCRETE Integer;
  Fsign TYPE DISCRETE Real;
equations:
  OnEvent v<::eps CAUSE Stick;
  OnEvent v>-::eps CAUSE Stick;
  OnEvent Stick DO
    new(Stuck) := ::True;
  END;
  OnEvent Stuck*u>Fs CAUSE UnStick;
```

```
  OnEvent Stuck*u<-Fs CAUSE UnStick;
  OnEvent UnStick DO
    new(Stuck) := ::False;
    new(Fsign) := sign(u);
  END;
  F = IF Stuck
      THEN u
      ELSE Fsign*Fc+Fv*v;
END;

Bristle ISA Model WITH
  dsnap, Kc, delta, vslide ISA Parameter;
% Snapping point, spring rate , new range and sling velocity
  x ISA SimpleTerminal;                    % global position
  dF ISA ZeroSumTerminal;                  % Friction contribut.
  Init ISAN Event;
  Snap ISAN Event;
variables:
  xb TYPE DISCRETE Real;                   % Bristle pos
  dx TYPE Real;
equations:
  OnEvent (abs(dx)>dsnap) CAUSE Snap;
  OnEvent Snap DO
    new(xb):=xb+sign(dx)*rect()*delta;
  END;
END;

BigBristle ISA FrictionModel WITH
  x ISA SimpleTerminal;                    % local position
  Fl ISA ZeroSumTerminal;                  % total force
  BSlide ISA Bristle WITH
  equations:
    dsnap := OUTER::dsnap;
    Kc := OUTER::Kc;
    delta := OUTER::delta;
    dx = x-xb;
    dF=Kc*dx;
  END;
  B1 ISA THIS::BSlide;
  B2 ISA THIS::BSlide;
  B3 ISA THIS::BSlide;
  B4 ISA THIS::BSlide;
  B5 ISA THIS::BSlide;
  B6 ISA THIS::BSlide;
  B7 ISA THIS::BSlide;
  BStick ISA Bristle WITH
  variables:
    Stuck TYPE DISCRETE Integer;
  equations:
    dsnap := OUTER::dsnap;
    Kc := OUTER::Kc;
    delta := OUTER::delta;
```

```
    OnEvent Snap DO
      new(Stuck) := IF abs(v)>vslide THEN ::False ELSE ::True;
    END;
    dx = IF Stuck THEN x-xb ELSE 0;
    dF = Kc*dx;
  END;
  B9 ISA THIS::BStick;
  Init ISAN Event;
parameters:
  dsnap ISA Parameter WITH
    default := 1.5e-4;                    % Snapping point
  END;
  Kc ISA Parameter WITH
    default := 1e6;                       % Bristle spring rate
  END;
  delta ISA Parameter WITH
    default := 0.75e-4;                   % New bristle rge
  END;
  vslide ISA Parameter WITH
    default := 0.001;                     % Min. slide velocity
  END;
equations:
  OnEvent Init DO
    new(BStick.Stuck) := ::True;
    new(B9.Stuck) := ::True;
    new(BSlide.xb) := (rect()-0.5)*delta/2;
    new(B2.xb) := (rect()-0.5)*delta/2;
    new(B4.xb) := (rect()-0.5)*delta/2;
    new(B6.xb) := (rect()-0.5)*delta/2;
    new(B7.xb) := (rect()-0.5)*delta/2;
    new(B1.xb) := -new(BSlide.xb);
    new(B3.xb) := -new(B2.xb);
    new(B5.xb) := -new(B4.xb);
    new(BStick.xb) := -new(B6.xb);
    new(B9.xb) := -new(B7.xb);
  END;
  OnEvent v<::eps CAUSE Stick;
  OnEvent v>-::eps CAUSE Stick;
  OnEvent Stick DO
    new(BStick.Stuck) := ::True;
    new(B9.Stuck) := ::True;
    new(x) := u/10/Kc;
    new(BSlide.xb) := (rect()-0.5)*delta/2;
    new(B2.xb) := (rect()-0.5)*delta/2;
    new(B4.xb) := (rect()-0.5)*delta/2;
    new(B6.xb) := (rect()-0.5)*delta/2;
    new(B7.xb) := (rect()-0.5)*delta/2;
    new(B1.xb) := -new(BSlide.xb);
    new(B3.xb) := -new(B2.xb);
    new(B5.xb) := -new(B4.xb);
    new(BStick.xb) := -new(B6.xb);
    new(B9.xb) := -new(B7.xb);
```

```
  END;
  x'=v;
  F = Fl;
  x AT BSlide.x;
  x AT B1.x;
  x AT B2.x;
  x AT B3.x;
  x AT B4.x;
  x AT B5.x;
  x AT BStick.x;
  x AT B6.x;
  x AT B7.x;
  x AT B9.x;
  Fl AT BSlide.dF;
  Fl AT B1.dF;
  Fl AT B2.dF;
  Fl AT B3.dF;
  Fl AT B4.dF;
  Fl AT B5.dF;
  Fl AT BStick.dF;
  Fl AT B6.dF;
  Fl AT B7.dF;
  Fl AT B9.dF;
END;

DahlFric ISA FrictionModel WITH
variables:
  Fx TYPE Real;                                % dF/dx
equations:
  Fx = Kc*(1-sign(v)*F/Fc);
  F' = Fx*v;
END;

HenFric ISA FrictionModel WITH
variables:
  z, g TYPE Real;                              % defl. & Strib funct.
equations:
  g = (Fc+(Fs-Fc)*exp(-sqr(v/vStrib)))/Kc;
  F = Kc*z+d*z'+Fv*v;
  z' = v-abs(v)*z/g;
END;

Armstrong ISA FrictionModel WITH
  UnStick ISAN Event;
  Stuck ISA Variable WITH
    value, initial TYPE DISCRETE Integer;
    initial := ::True;
  END;
  Fslast ISA Variable WITH
    value TYPE DISCRETE Real;
    initial := Fs;
  END;
```

```
  Fsact ISA Variable WITH
    initial := Fs;
  END;
parameters:
  tau ISA Parameter WITH
    default:=0.02;                          % Frict. memory lag
  END;
  gamma ISA Parameter WITH
    default:=1.0;                           % time constant of Fs
  END;
variables:
  Trest TYPE DISCRETE Real;
  Ff,F2,x,dv TYPE Real;
  Fsign TYPE DISCRETE Real;
equations:
  dv=delay(v,tau);
  F2 = ((Fsact-Fc)/(1+sqr(dv/vStrib))+Fc)*Fsign+Fv*v;
  Ff = Kc*x;
  F = IF Stuck THEN Ff ELSE F2;
  x' = IF Stuck THEN v ELSE 0;
  Fsact' = IF Stuck
              THEN (Fs-Fslast)*gamma/sqr(Base::Time-Trest+gamma)
              ELSE 0;
  OnEvent (10*Stuck+v)<::eps CAUSE Stick;
  OnEvent (-10*Stuck+v)>-::eps CAUSE Stick;
  OnEvent Stick DO
    new(Stuck) := ::True;
    new(Fsact) := abs(F2);
    new(Fslast) := abs(F2);
    new(Trest) := Base::Time;
    new(x) := u/Kc;
  END;
  OnEvent Ff>Fsact CAUSE UnStick;
  OnEvent Ff<-Fsact CAUSE UnStick;
  OnEvent UnStick DO
    new(Stuck) := ::False;
    new(x) := 0;
    new(Fsign) := sign(x);
  END;
END;
```

C.3 Gruvon library

```
LIBRARY Gruvon;

USES Pneumatics, Base;

S21 ISA PneumaticsModel WITH
  C1, C2 ISA PneuCut;
  Chamb1, Chamb2 ISA DEDiaphragm;
  Rod ISA PointMass WITH
    Stick ISA EventInput;
    OnEvent Stick DO
      new(C.x') := 0;
    END;
  END;
  Fric ISA Friction::FrictionForce WITH
    Friction ISA Friction::HenFric;
  END;
  CM ISA MechCut;
equations:
  Fric.Friction.u := -(Chamb1.CM.Fx+Chamb2.CM.Fx+CM.Fx);
connections:
  C1 AT Chamb1.C1;
  C2 AT Chamb2.C1;
  Chamb1.CM AT CM;
  Chamb2.CM AT CM;
  Fric.C AT CM;
  CM AT Rod.C;
END;

CenterPart ISA PneumaticsModel WITH
  C ISA MechCut;
  CA ISA AngleCut;
parameters:
  R ISA Parameter WITH
    default := 0.06;                          % Lever length
  END;
  stroke ISA Parameter;
equations:
  stroke := R/sqrt(2);
  CA.T = C.Fx*R;
  CA.fi = (stroke - C.x)*Pneumatics::Pi/(4*stroke);
END;

SP400 ISA PneumaticsModel WITH
  C1, C2 ISA PneuCut;
  input ISA PressureTerminal;
  CA ISAN AngleCut;
  Lever ISA Flapper;
  RangeSpring ISA Spring;
  Valve ISA ServoValve;
  MechIn ISA MechCut;                         % Force from input pr.
```

```
parameters:
  CamConst ISA Parameter;
% Gain from angle to spring end position
  Zeropos ISA Parameter;
% Spring end position at zero angular position
  Ain ISA Parameter;                          % Input diaphragm area
equations:
  RangeSpring.C2.x = Zeropos - CamConst*CA.fi/Pneumatics::Pi*2;
  MechIn.Fx = Ain*input;
connections:
  MechIn AT Lever.C1;
  RangeSpring.C1 AT Lever.C2;
  Valve.CM AT Lever.C3;
  C1 AT Valve.C1;
  C2 AT Valve.C2;
END;

SOMAS ISA PneumaticsModel WITH
  input ISA PressureTerminal;
  output ISA AngleCut;
  Act ISA S21;
  Pos ISA SP400;
  Con ISA CenterPart;
equations:
  Pos.CA.T = 0;                               % No moment from cam
connections:
  input AT Pos.input;
  Pos.C1 AT Act.C1;
  Pos.C2 AT Act.C2;
  Pos.CA AT Con.CA;
  Act.CM AT Con.C;
  Con.CA AT output;
END;

PropControl ISA PneumaticsModel WITH
  y ISA SimpleTerminal;
  u ISA PressureTerminal;
  Init ISA Event;
  Sample ISA Event;
parameters:
  K, Ti, SetPoint ISA Parameter;
% PI-gain, integral time and concentration reference
  Amp ISA Parameter WITH
    default := 80000.0;                       % Pressure signal span
  END;
  Bias ISA Base::Parameter WITH
    default := 20000.0;                       % Pr.sign offset in Pa
  END;
  omega, zeta ISA Parameter;
% Second order parameters of the transmission line
variables:
  e, i, v, pv TYPE Real;
```

```
equations:
  e = (y - SetPoint);
  i'= e/Ti;
  v = if K*(e+i)>1 then 1 else if e+i<0 then 0 else K*(e+i);
  pv = Bias + Amp*V;
%Convert the control signal to a pressure signal
  u''+2*zeta*omega*u' = omega*omega*(pv-u);
%Transm. line lag, u is the pressure signal at the positioner
END;

PulpPipe ISA PneumaticsModel WITH
  input ISA VolumeFlowTerminal;
  output ISA SimpleTerminal;
  Init ISA Event;
parameters:
  L, T ISA Parameter;
% Dead time and time constant of the concentration sensor
  ConcIn, Win ISA Parameter;
% Fibre concentration and volume flow of pulp input
  Ky, Offset ISA Parameter;
% The gain and offset from concentration to the measured output
variables:
  Conc TYPE Real;
  Normy TYPE Real;                          % Normed output
equations:
  Conc = ConcIn*Win/(Win + input);
  Normy = Ky*(Conc+Offset);
  T*output' + output = delay(Normy, L, 0.4);
%First-order lag and delay of the measured output
END;

BallValve ISA PneumaticsModel WITH
  input ISA AngleCut;
  output ISA VolumeFlowTerminal;
parameters:
  MaxFlow ISA Parameter WITH
    default := 0.2;
  END;
variables:
  perc_open TYPE Real;                      % Valve opening
equations:
  input.T = 0;
  perc_open = input.fi/Pneumatics::Pi*2;
  output = -MaxFlow*perc_open;
END;

ConcLoop ISA PneumaticsModel WITH
  PosAct ISA SOMAS;
  Valve ISA BallValve;
  Mixer ISA PulpPipe;
  Control ISA PropControl;
connections:
```

```
  Mixer.output AT Control.y;
  Control.u AT PosAct.input;
  PosAct.output AT Valve.input;
  Valve.output AT Mixer.input;
  Graphic ISA super::Graphic;
END;
```