A closed loop arterial pressure controller and an infusion toolbox for anaesthesia

Meijers, Roger H.A.M.

Award date: 1996
A Closed Loop Arterial Pressure Controller 
and an Infusion Toolbox for Anaesthesia

Integration and Clinical Evaluation

By Roger H.A.M. Meijers

Master Thesis.
Submitted in partial fulfillment of the requirements for 
the Master's degree in Electrical Engineering (Ir.).

Supervisors: Prof. dr. ir. J.E.W. Beneken, 
Dr. ir. J.A. Blom and 
Prof. dr. A.A. d'Hollander.

May 1995 - January 1996.

The faculty of Electrical Engineering of the Eindhoven University of Technology 
declines all responsibility for the contents of graduation or training reports.
Abstract

During surgery it is the task of the anaesthetist to bring the patient into an optimal condition which includes tasks like creating a state of unconsciousness and suppressing pain sensation and muscle reactions. Therefore various drugs and devices need to be controlled and monitored. Computers can be of great help to support the anaesthetist through retrieving information, giving a neat overview of the data at a central location and automation of certain tasks. The final goal would be a central terminal to control and monitor all functions and devices for anaesthesia.

In this report a closed loop blood pressure controller is described which is based on a simple and robust PI-controller and a supervising expert system. Adaptive control is necessary because the sensitivity of the patients to the used drug, i.e. sodium nitroprusside (SNP), varies over a wide range. The drug SNP decreases the mean arterial pressure (MAP) through dilation of the smaller arteries. 33 clinical tests during cardiac surgery have been performed and the evaluation, which was one of the objectives of this research, shows good and safe performance. The controller was on average in automatic mode for more than 90 % of the time of an operation and the performance during effective control showed the MAP to be within a distance of 10 mmHg to the setpoint for 89 % of the time. The average distance to the setpoint during effective control was 4.5 mmHg.

Automatic delivery of various other drugs was during these tests performed using an Infusion Toolbox. This Infusion Toolbox is a complete guiding system to assist the anaesthetist to deliver simultaneously many drugs through computer controlled infusion and is also described in this report. The second objective of this research was to integrate the blood pressure controller and the Toolbox. The created design has a modular structure and is based on a client-server model. The communication between both applications is accomplished through usage of the serial network of the Toolbox which is build around a universal device communication driver (UDCC) and a bedside communication controller (BCC) staying in the PC. This network provides a kind of device communication controller (DCC) for the various commercially available infusion devices which can be controlled.

The communication network and protocol were implemented and a first prototype of the controller with a new user-interface in the Infusion Toolbox application is build. This prototype now has to be tested extensively through simulations before it can be used in a clinical environment. The resulting system will then be another step in the direction of a machine for total intravenous anaesthesia (TIVA).
Acknowledgements

Research is never the result of the effort of one single person. This project would not have been possible without the support of many people. Therefore, I would like to thank everybody who in any way contributed to the final result. In particular I thank Prof. dr. A. d'Hollander of the Département d' Anaesthesiologie, Hôpital Erasme (Brussels) for giving me the opportunity to work in an international and medical environment and providing the necessary means and contacts. Moreover, it was very pleasant to work with him and the anaesthetists: Dr. Luc Barvais and Dr. Denis Schmartz, who were very interested in the blood pressure controller and very cooperative in performing the clinical tests with great accuracy.

I also thank Prof. F. Cantraine of the Département de Calcul Scientifique, Faculté de Médecine, Université Libre de Bruxelles for his participation in the project and the critical comments and educational discussions. Special acknowledgements go to Eddy Cousseart for sharing his experience, knowledge and many hours of collaboration, and Didier Jaumot for his attention and social engagement.

Dr. ir. J.A. Blom deserves a lot of credit for his guidance, positive attitude and help in creating this report. All other employees and students of the Department of Medical Electrical Engineering I thank for their help and sympathy during my pleasant stay there.

Finally I am very grateful to my parents, Annette, family and friends for their love, patience, and support during all the years. A special thought of remembrance goes to my kind grandparents, who unfortunately did not live to see the day of my graduation.

Roger.
Contents

1. Introduction ..................................................................................................................1

2. A Blood Pressure Controller ..........................................................................................3

   2.1. Introduction .............................................................................................................3
   2.2. Goal of the Application .........................................................................................4
   2.3. The Total System ..................................................................................................5
       2.3.1. The Patient's Response Model ....................................................................6
       2.3.2. PI(D) Control ..............................................................................................7
       2.3.3. Arterial Pressure Measurement, Conversion and Validation ......................9
   2.4. SIMPLEXYS Real Time Expert Systems ...............................................................11
       2.4.1. The Programming Language ......................................................................11
       2.4.2. The Rule Base ..............................................................................................14
       2.4.3. The Toolbox ..................................................................................................15
       2.4.4. Inference ........................................................................................................16
   2.5. Implemented Mechanisms .....................................................................................17
       2.5.1. Gain Adaptations ..........................................................................................18
       2.5.2. Oscillation Detection ....................................................................................20
       2.5.3. Transient Detection ......................................................................................22
   2.6. Other Aspects of the Design ..................................................................................23
       2.6.1. Knowledge Acquisition ................................................................................23
       2.6.2. The User Interface .........................................................................................23
       2.6.3. Previous Tests ...............................................................................................24

3. An Infusion Toolbox .........................................................................................................25

   3.1. The System Setup ..................................................................................................25
       3.1.1. The Hardware ...............................................................................................26
       3.1.2. The Software ................................................................................................27
   3.3. The Tools ................................................................................................................28
       3.3.1. Simulating the Pharmacokinetics for INfusion of Anaesthetic drugs (SPINA) 28
       3.3.2. Composition of INfusion sheets for Anaesthesia (CINA) ............................30
       3.3.3. Monitoring of INfusion for Anaesthesia (MINA) .........................................31
   3.4. Conclusions ............................................................................................................31
4. Extension of the Infusion Toolbox
with the Blood Pressure Controller  

4.1. Object Oriented Analysis of the BP-controller ........................................... 33
4.2. Flow Diagram ................................................................................................. 38
4.3. The Communication Model ........................................................................... 41
   4.3.1. The ISO/OSI Reference Model ................................................................. 41
   4.3.2. System Design ......................................................................................... 44
   4.3.3. OO-representation of the Transport Service ........................................... 46
4.4. The Communication Protocol ...................................................................... 47
   4.4.1. Implemented Features ............................................................................ 47
   4.4.2. Error Handling ....................................................................................... 49
4.5. Other Aspects of the Design ....................................................................... 50
   4.5.1. The New User-Interface ......................................................................... 51
   4.5.2. Extra Facilities ....................................................................................... 52
4.6. Conclusions .................................................................................................. 52
4.7. Future Developments and Tests ................................................................... 52

5. Clinical Evaluation of the stand-alone Blood Pressure Controller  

5.1. Test Environment and Trial Setup ............................................................... 53
5.2. Patients Data and Surgery Type .................................................................. 55
5.3. Statistical Analysis ....................................................................................... 56
   5.3.1. Average Values ...................................................................................... 56
   5.3.2. Histograms ............................................................................................ 59
   5.3.3. Comparison with Previous Tests ........................................................... 61
5.4. Typical Examples of the Performance ........................................................... 62
   5.4.1. Normal Good Behaviour ....................................................................... 62
   5.4.2. Typical Cases Needing Improvement ................................................... 65
   5.4.3. An Extreme Situation ........................................................................... 67
5.5. Gain Adaptations .......................................................................................... 68
5.6. Opinion of the Anaesthetists ....................................................................... 70
5.7. Conclusions .................................................................................................. 72

6. Future Developments ..................................................................................... 73

6.1. The Blood Pressure Controller ..................................................................... 73
6.2. The Infusion Toolbox .................................................................................. 74

7. Conclusions and Recommendations .................................................................. 75

References .......................................................................................................... 77

Appendices ........................................................................................................... 81
Appendix A: Tables with the Complete Statistical Data Extracted from the Clinical Tests ................................................................. 83
Appendix B: Notations used in Flow Charts ......................................................... 90
Appendix C: Instruction Manual for the Anaesthetists ........................................ 91
Chapter 1

Introduction

Anaesthesia is a complex process in which the anaesthetist evaluates many clinical signs to form a judgement on the adequacy of anaesthesia. The use of computers to assist the anaesthetist to monitor and control the delivery of drugs can offer considerable advantages. As yet no single intravenous anaesthetic drug can effectively and safely provide hypnosis, analgesia and amnesia. Thus intelligent combinations of hypnotics and opioids are necessary, especially for total intravenous anaesthesia (TIVA). Total intravenous anaesthesia is becoming increasingly popular among anaesthetists. It has several advantages, namely each component of the anaesthetic protocol can be controlled independently and the operating room remains unpolluted with nitrous oxide or volatile anaesthetic agents. TIVA aims to maintain a constant blood concentration of each anaesthetic agent. To achieve this infusion rates need to be repeatedly altered.

There are several syringe and volumetric pumps which are accurate enough for use in TIVA and which may be controlled by a computer. Computer controlled infusion of drugs can be designed as either 'open' loop or 'closed' loop systems. The computer can calculate theoretical blood concentrations of a drug according to a pharmacokinetic model and drive an infusion device; this is called open loop. A closed loop control system uses an input signal, such as blood pressure (BP), to alter the infusion of some vasodilator drug to provide a stable blood pressure for a patient by automatically altering the vasodilator infusion rate. Closed loop systems therefore improve the delivery of drugs by relating the infusion directly to the response of the patient. A good overview of computer controlled infusion for anaesthesia is given by O'Hara et al. [O'Hara, 1992]; this paper also reviews the history of computer controlled delivery systems.

A lot of research is done on controllers for regulating infusion of one single agent, but only a few research teams have been working on systems to accomplish total intravenous anaesthesia. Certainly no commercial device is available yet. However, some initial clinical tests confirm the expected benefits of TIVA: greater haemodynamic stability, decreased drug consumption, more rapid recovery and a lesser need for postoperative ventilatory support. Unfortunately it is an expensive technique. Moreover, there is considerable interpatient variability of the drug concentration required for a similar clinical effect. Various methods are proposed to decrease this variability: population pharmacokinetic models, Bayesian forecasting or closed loop systems; the latter are still research tools.

The Medical Electrical Engineering Department of Eindhoven University of Technology has studied automatic blood pressure control for several years. Various prototypes of a closed loop system have been developed and tested, most of them using Sodium Nitroprusside (SNP), see [Geene, 1993], [Lammers, 1990] and [Zwart, 1990]. But also a controller using Nitroglycerine (NTG), based on an equivalent setup, has been developed and tested [Hoeksel, 1992]. Arterial blood pressure control using SNP has been the objective of many research projects. Most of them
focused on treatment of post-operative hypertension after cardiac surgery, see for example [Chaudri, 1992], [Chitwood, 1992], [Delapasse, 1994], [Pajunen, 1990] and [Ying, 1992]. Blood pressure regulation during cardiac surgery provides a more complex environment though. Thus most controllers are designed for use in the intensive care environment and not for use in the operating room. One example of control during cardiac surgery is described by Martin et al. [Martin, 1992(a&b)]. The controller developed in Eindhoven is also designed for usage during cardiac surgery.

Literature contains a large variety of control strategies for closed loop control of blood pressure. Theoretical developed strategies are: PID control and its variants, optimal control, adaptive control, rule-based control, fuzzy control and neural network control. In a review article by Isaka and Sebald [Isaka, 1993] the various strategies are classified and described shortly, while also a large number of references is listed. The prototypes of the controllers developed at the Medical Electrical Engineering Department use the strategy of PI-control supervised by a real-time expert system. The expert system is rule based and built using the expert system shell SIMPLEXYS which is developed in particular for applications in patient monitoring [Blom, 1990]. Difficulties in controlling blood pressure during cardiac surgery, the controller system setup and its various mechanisms are described in chapter 2.

One of the objectives of this research has been the evaluation of the most recent prototype of this blood pressure controller in a clinical situation. Previous clinical tests have shown already that it can be a useful and successful tool in clinical practice. In order to confirm this and pinpoint further necessary improvements a new series of clinical tests was performed in collaboration with the Department of Anaesthesiology of the Erasme Hospital (Brussels, Belgium)*. In chapter 5 an overview of the test environment, the results and the statistical analysis is given. The conclusions from this analysis and the opinions of the anaesthetist who performed the tests led to some recommendations for further enhancement of the controller. These future developments are described in chapter 6.

The anaesthetists in Brussels also use a system which provides tools to deliver automatically and simultaneously various drugs by controlling commercially available infusion devices. This system, called Infusion Toolbox, has been developed together with the Department of Data Processing at the Faculty of Medicine, Free University of Brussels (ULB)**. The Infusion Toolbox is based on a personal computer, a serial communication network and various data bases which contain information about infusion devices, drug packaging and pharmacokinetic models. A description of this system is given in chapter 3.

The second objective of this research was to study the Infusion Toolbox and create and implement a design for integration of the blood pressure controller and the Infusion Toolbox. The integration is accomplished through connection of the BP-controller to the serial network of the Toolbox and implementation of a communication protocol based on a client-server model. In fact the blood pressure controller became a black box providing services to the Toolbox. The exact design for extension of the Infusion Toolbox is discussed in chapter 4. This is partially done through showing the object oriented analysis. Finally in chapter 7 of this report conclusions and recommendations are noted which form a basis for further research on extension of the total system in the direction of a machine for control of total intravenous anaesthesia.

* Département d' Anaesthésiologie, Hôpital Erasme, Bruxelles, Belgique.

** Département de Calcul Scientifique, Faculté de Médecine, Université Libre de Bruxelles, Bruxelles, Belgique.
Chapter 2

A Blood Pressure Controller

The prototype of the blood pressure controller, which evolved from long-term research done at the Medical Electrical Engineering Department of Eindhoven University of Technology, will be globally described in this chapter. After stating the goal of the application and a global description of the system setup, the important aspects of PI(D) control and the retrieval of the blood pressure signal are noted. Although in order to understand the principle of control of this system there is no need to know a lot about SIMPLEXY, a brief description of this expert system shell, which is used to built the supervising expert system, is given for completeness. To understand the behaviour of the controller better the safety mechanisms for transient and oscillation detection as well as gain adaptation, which are implemented in the expert system, are explained next. Finally various other aspects of the design, for example the user-interface, are shown.

2.1. Introduction

During and (shortly) after some types of surgical procedures the patient's blood pressure needs to be regulated, in particular it needs to be kept in a controlled hypotensive state. This is only a small part of the task of the anaesthetist. The patient needs to be brought into an optimal condition for surgery which also includes tasks like creating a state of unconsciousness and suppressing pain sensation and muscle reactions. Many of the drugs used for this already lower the blood pressure, but during and after cardiac surgery the pressure is frequently lowered additionally.

A lowered blood pressure has a number of medical benefits. The excessive bleeding caused by hypertension, which limits the view of the surgeon and increases the amount of blood lost during the operation, is reduced by the induced hypotension. Furthermore extra complications which slow down the patient's recovery are avoided. The anaesthetists use vasodilating drugs to lower the arterial pressure, usually Sodium Nitroprusside (SNP) or Nitroglycerine (NTG). The patient's response to these drugs is fast so regulation and stabilization is possible by varying the flow rate of the drug.

The major advantages of SNP are rapid onset of action, short duration, relatively minor direct cardiac effects and effective afterload reduction by vasodilation of both venous capacitance and arteriolar resistance beds. Blood pressure (BP) is, roughly speaking, a product of cardiac output and vascular resistance; infusion of SNP will cause a reduction in BP due to a decrease of the resistance. However, many of the characteristics that make nitroprusside effective and attractive also can be disadvantageous. Severe afterload reduction may decrease diastolic perfusion pressure below physiologic ranges which can be life threatening. As the drug is short acting, the
rate of infusion must be adjusted frequently, causing manual blood pressure control to be very
time-consuming. Further the cyanide toxicity of SNP demands cautious usage and limitation of
the total doses over a certain time period. An exhaustive discussion of various aspects regarding
SNP can be found in a recent publication by Friederich and Butterworth [Friederich, 1995] which
reviews the usage of Sodium Nitroprusside over the last twenty years.

So manually controlling the infusion rate requires continuous monitoring of the pressure and
adjustment of the flow rate. To relieve the anaesthetist from this often difficult and certainly time
consuming task an automatic controller is developed. In the first instance this was done for SNP
[Geene, 1993; Lammers, 1990; Zwart, 1990] but at a later stage the same structure was also used
for a NTG-controller [Hoeksel, 1992]. In the next section we will discuss the major problems for
automatic control and the goal of the application.

2.2. Goal of the Application

There are some restrictions that prevent the use of a classic controller. First the sensitivity to the
drug varies over a wide range between the different patients and is unknown prior to the
operation. Secondly, during the operation the pressure and the sensitivity can change due to
many other causes. The patients response to SNP is complicated. It dilates the small arterioles
but also influences the baroreceptor action. At large flow rates the dilation often reaches
saturation and as a result it is a non-linear process [Blom, 1990]. Cardiac surgery in particular
may be the most hostile environment for closed-loop regulation of arterial pressure using SNP.

During cardiac surgery, blood pressure control is subject to a variety of disturbances. Vasoactive
and cardioactive drugs are often administered, by either bolus or infusion, introducing severe
disturbances to a controller. The arterial line may be flushed and/or blood drawn, invalidating the
measured pressure. Low anaesthetic levels may allow the patient to react to painful stimuli,
changing patient response characteristics. Blood volume variations and patient position changes
will also change patients response characteristics. Cooling and warming of the patient as he or
she goes on or of cardiopulmonary bypass also challenges the controller. In fact there is an
extensive list of events that can occur and have impact on the mean arterial pressure (MAP). A
reasonably complete list can be found in the paper by Martin et al. [Martin, 1992(b)] in which a
list of events is presented in Table I.

To overcome these restrictions and control the arterial pressure safely, specific medical and
control knowledge is incorporated in an expert system which supervises a simple and robust PI-
controller. The total system is shown in Figure 2.1 and will be described extensively in the next
sections. However, before going into more detail the main goals will be described.

First and above all, the controller must respond safely in all circumstances. To ensure safety at
least two conditions must be satisfied [O'Hara, 1992]; analysis of the raw blood pressure signal is
necessary to validate the measurements, and to avoid overreaction of the controller to rapid
changes in the arterial pressure due to surgical actions, these so called transients need to be
detected. Secondly, the controller must be applicable for all kind of patients and therefore be able
to deal with the large range of the patient's sensitivity. The various mechanisms which are
implemented to ensure safe and stable control are discussed in the following sections. Further,
the control speed, this is the time to bring the pressure towards the desired level, should
approximate the control speed of manual regulation by the anaesthetist.
To be able to verify whether the designed system could reach its goal, more specific performance characteristics have been established. The most recent ones used by Geene [Geene, 1993] are:

1. The mean arterial pressure (MAP) should be brought towards the setpoint within 10 minutes after selection of a new setpoint; this is known as the settling time.
2. The maximum overshoot of the setpoint should be less than 10 mmHg.
3. In steady-state the error, i.e. distance between the MAP and the setpoint, should be within 10 mmHg.

Figure 2.1: The functional units of the complete blood pressure control system.

2.3 The Total System

The developed automatic controller consists of a number of functional units which are shown in Figure 2.1. As stated before, a PI-controller is used to control the pressure to go to and stay close to a given setpoint and an expert system performs gain adaptations to cope with all types of patients. The gain is selected depending on the sensitivity of the patient. The expert system incorporates a knowledge base containing all rules on which decisions of the SIMPLEXYS inference engine are based. The exact aspects of the expert system will be discussed in section 2.4, while the various mechanisms are reviewed in section 2.5. Further the system consists of a unit to sample and validate the blood pressure signal, an infusion pump and a user-interface.
The principal behaviour of the system is that at the start of each 5-seconds-period information coming from the pump, the AD-converter, the pressure validation unit and the keyboard is analyzed. The inference engine evaluates this information and draws conclusions. These conclusions are for example about whether the new infusion flow rate, determined by the PI-controller, can be used; whether the gain should be changed; whether some kind of artefact occurred in the pressure signal, etc. A change of the gain, for example, can be necessary due to a change in the drug sensitivity of the patient. In the ideal situation the gain should be inversely proportional to the patient's sensitivity. Finally, at the end of the 5-seconds-period, the graphical display is refreshed.

### 2.3.1. The Patient's Response Model

The design of the controller is based on a simplified model of the patient's reaction to sodium nitroprusside in reality. A first order model with time delay, of which the mathematical description is given in formula 2.1, was found to be accurate enough.

\[
Y(s) = e^{-\tau} \cdot \frac{-K}{s\tau + 1} \cdot U(s) \tag{2.1.a}
\]

\[
U(s) = \mathcal{L}(u(t)) \tag{2.1.b}
\]

\[
Y(s) = \mathcal{L}(y(t) - y(0)) \tag{2.1.c}
\]

In these equations and in Figure 2.2 the step response to a unit dose of SNP is shown. In the equations the symbol \(\mathcal{L}\) denotes the Laplace operator and the symbols \(y\) and \(u\) denote, respectively, the pressure and the infusion flow rate. The model is determined by three parameters. The static behaviour is described by the sensitivity \(K\), which is the final change of pressure after a unit flow change. The dynamic behaviour is described by the delay time \(T\) and the time constant \(\tau\). The delay time is the time before the pressure shows any change after the flow has been changed, while \(\tau\) is the time it takes, after the delay time, until 63% of the complete effect of a flow change is realized.

![Figure 2.2: Step response to nitroprusside of a first order patient model with time delay [Lammers, 1990].](image-url)
All parameters vary between patients. It is already stated that the patient's sensitivity range is expected to be large. Therefore K is assumed to vary from 1/9 to 9 times normal, with a nominal value of 25 [mmHg/µg/kg/min] for a person of 80 kg. Moreover, sensitivity changes in time and decreases at higher flow rates due to the earlier mentioned non-linearity. The dynamics of the system also change during the operation due to variations in blood volume, supply of other drugs, cardiopulmonary bypass, hypothermia, etc. The nominal values of respectively T and τ were estimated to be 50 and 60 seconds [Lammers, 1990].

To overcome these parameter deviations, adaptive control is necessary. This is accomplished by continuously monitoring the patient's response with an expert system. This expert system adapts the gain of the robust PI-controller in order to ensure safe and fast enough control for all patients.

2.3.2. PI(D) Control

Because a PID-controller is simple and robust, it was chosen as the basic controller. PID-controllers regulate a variable to a target. The difference between the target and the variable is defined as the error. In order to decrease the error the output, \( F_k \), is changed according to equation 2.2. In this equation \( F_k \) denotes the output at time \( k \), \( e \) the error, \( \Delta t \) the time between two measurements and \( G \) the gain which is a scaling factor. \( K_p, K_i \) and \( K_d \) are constants.

\[
F_k = G(K_p e + K_i \sum \Delta e \Delta t + K_d \frac{\Delta e}{\Delta t}) \tag{2.2}
\]

The right-hand side of the equation consists of three components. The first component is proportional to the error and therefore called P-term. The second term is the accumulated error over time (integral of error), which is referred to as I-term. The last part, D-term, is the rate of change or derivative of this error.

In the blood pressure control application the output of the PID-controller is the flow rate of nitroprusside and the error is the distance between the MAP and the setpoint. Now the P-term works against any change of pressure and is used for anticipating a changing pressure before it has crossed the setpoint. It has a stabilizing task by increasing the flow rate when the pressure increases and vice versa. The integration term (I-term) has a regulative task and adjusts the flow rate as long as the pressure is not at the target level through increasing the flow rate when the pressure is above the setpoint and decreasing it when the pressure is below.

The D-term is normally applied to limit an overshoot. However, in this application there is too much measurement noise and artificial fluctuations in the MAP which causes this term not to improve the control performance. Therefore it is removed and only PI-control is applied.

The period between two measurements is 5 seconds. A slightly changed equation is used to calculate the necessary flow change over this 5-second period. This, so called dog lead principle, prevents unlimited growth of the integration term and makes it possible to start with an arbitrary flow rate. As can be seen in equation 2.3, the flow rate is now calculated as a change of the flow rate at the previous sample moment. Automatic control can now start from any manual flow rate set by the anaesthetist.
A Blood Pressure Controller

\[ F_k = F_{k-1} + G(K_p(y_k - y_{k-1}) + K_i(y_k - r_k)) \]  (2.3)

In this equation \( F_k \) and \( F_{k-1} \) denote, respectively, the current and previous flow rate and \( y_k \) and \( y_{k-1} \) the current and previous pressure. The complete list of parameters is:

- \( y \): Mean Arterial Pressure (MAP) [mmHg].
- \( r \): target / setpoint [mmHg].
- \( F \): flow rate [ml/h] or [mg/h].
- \( G \): overall control gain [\( \frac{1}{2} \) ... 9].
- \( K_p \): P-parameter, \( K_p = 0.145 \) [ml/h/mmHg].
- \( K_i \): I-Parameter, \( K_i = \frac{K_p}{20} \) [ml/h/mmHg].
- \( K_i = \frac{K_p}{32} \) [ml/h/mmHg].

The task of the controller is to regulate the pressure towards the setpoint and stabilize it at this target level. These tasks are generally conflicting which means that in order to perform each task optimal, different \( K_p : K_i \) ratios are necessary. The performance of the controller is optimized by tuning two PI-controllers, one for regulation (1) and one for stabilization (2). The expert system decides which control mode is suitable at each moment, mainly dependent on whether the MAP is within a certain distance range of the setpoint. The exact criteria are well described by Geene [Geene, 1993]. For regulation mode the \( K_p : K_i \) ratio is 20:1 while for stabilization mode the ratio is 32:1.

The PI-controller can now cope with the expected range of the dynamic patient characteristics. Together with the gain adaptation mechanism this ensures stability and robustness of the control system. However, to ensure safety for all patients, validation of the blood pressure measurements and detection of transients is necessary. The mechanisms through which this is obtained are discussed in the next sections. Before doing so, two effects which additionally influence PI-control will be mentioned.

The first effect is limitation of the flow rate change. When the MAP is far above the setpoint, the flow usually changes fast. If after a while the high MAP turns out to be temporary, for example caused by other drugs or pain, the controller needs to recover back to normal. This phenomenon is called integral-wind up. A solution is to limit the flow rate change. To prevent the I-term limitation \( (\Delta F_{i,\text{max}}) \) to be influenced by noise which influences the P-term limitation \( (\Delta F_{p,\text{max}}) \) both limitations should be looked at independently [Geene, 1993]. The effect is reached through implementation of the following equations:

\[ \Delta F_{\text{max}} = \Delta F_{i,\text{max}} + \Delta F_{p,\text{max}} \]  (2.4)
A Blood Pressure Controller

\[ \Delta F_{\text{max}} = 30 \cdot K_i \cdot \text{Gain} \quad (2.5\text{.a}) \]
\[ \Delta F_{p,\text{max}} = 2 \cdot K_p \cdot \text{Gain} \quad (2.5\text{.b}) \]

This results in more robust control in case of transients. The second effect that influences PI-control is calculation of a virtual negative flow. The influence of the P-term of the controller in time becomes linearly dependent on the distance of the MAP to the setpoint. However, due to the computation method this influence of the P-term is not represented in the flow anymore if the flow becomes zero. A decrease in pressure should then result in a negative flow which is impossible. In order to overcome overreaction of the controller to a fast pressure change below the setpoint if the pressure has been zero, a virtual negative flow is used in the computations when the flow is zero. Control is started again when the virtual flow exceeds zero or the pressure rises above a limit, i.e. 3 mmHg under the setpoint.

2.3.3. Arterial Pressure Measurement, Conversion and Validation

The feedback signal used for the purpose of closed loop control is the 5 second averaged MAP. Only valid heart cycles contribute to this average which implies that the average can only be computed reliable if there are at least two valid cycles within the 5 second period. Otherwise no new MAP can be computed and the controller must take appropriate actions.

The arterial pressure signal is obtained through sampling the analog signal which is taken directly from the analog output of the monitoring equipment in the operating room. The signal is sampled at a sample rate of 50 Hz using a Lab Master card from Scientific Solutions Inc. for the AD-conversion. To reduce the influence of measurement noise the signal is filtered before sampling, with a low pass anti-aliasing filter with cut off frequency of 25 Hz.

Sometimes the pressure measurements are disturbed in which case they do not reflect the real pressure. This can be caused by flushing of the arterial line, sampling of blood, electrocautery, etc. Incorrect measurements cause unreliable control and should be prevented. In this application the shape of the arterial pressure curve in each heart cycle is checked and compared with previous heart cycles. It is assumed that the shape of the curve of heart cycles following each other is almost identical. The comparison is based on several features which are extracted from each pulse period, see Figure 2.3.

The extracted features are:
- the diastolic pressure (Mi),
- the systolic pressure (Ma),
- the upslope pulse pressure (D1),
- the downslope pulse pressure (D2),
- the systolic pressure slope (H1),
- the pulse period or heart rate (HP),
- the average of the blood pressure over a full period (PG).

The changes in those parameters define the validity of a heart cycle [Lammers, 1990 and Zwart, 1992]. Only when they have physiological values and differ little from the parameters from previous cycles, the heart cycle is considered valid.
When invalid measurements occur the controller first continues using the last valid MAP. If within a short period valid measurements are available again, control continues with the new valid MAP. A longer period of invalid measurements forces the controller to switch to manual mode, giving an alarm to get the anaesthetist's attention. Simulations and practice show that the controller can work accurately for 2 minutes if the MAP is close to the setpoint, and one minute otherwise. For safety reasons the system returns to manual after 1 minute if the MAP is near the setpoint and after 30 seconds otherwise. The anaesthetist can switch back to automatic control as soon as the measurement is valid again.

The expert system which supervises the PI-controller cannot take reliable decisions based on only one MAP measurement. The MAP is influenced by fluctuations with temporary and permanent causes. More measurements must be taken into account in order to reduce random influences such as noise. All through the rule base of the expert system moving average filters are used for filtering. A moving average filter implements a low pass filter and is described by the following formula:

\[ x_k = f \cdot x_{k-1} + (1-f) \cdot y_k \]  

Output \( x_k \) is the moving average of \( y_k \). The characteristics of the filter are determined by the filter parameter \( f \) (between 0 and 1). The main reason for filtering is the reduction of meaningless signal variations. A filter introduces a time delay, but this is often acceptable for decisions taken by the expert system because incorrect decisions based on noisy signals are in general unacceptable. Besides a filtered MAP value, the distance between the MAP and the setpoint is presented by the filtered variable (both \( f = 0.8 \)). Further the flow rate is filtered in order to detect large flow changes (\( f = 0.98 \)) and calculate the pre-transient flow rate (\( f = 0.93 \)).

Before discussing some other safety mechanisms, the expert system will be described first in the next section.
2.4. SIMPLEXYS Real Time Expert Systems

The SIMPLEXYS programming language and toolbox to create real time expert systems has been developed especially for applications like the closed loop automatic blood pressure controller. The features these applications demand are the capability of operating in real time, the ease of use and understanding, compactness in order to run on a PC, efficiency, speed and above all reliability. Sufficient potentials to check correctness must also be provided.

To accomplish this a special toolbox was created which is written in the efficient programming language Pascal (a version written in the C programming language is also available). Advantages that made Pascal (or C) more preferable, than for example LISP, were its speed and the capability to run on small and cheap computers. In fact the toolbox is a sophisticated expert system shell as described by Jackson [Jackson, 1990].

2.4.1 The Programming Language

The most important tool of SIMPLEXYS is the programming language. It allows certain types of human knowledge to be described in a formal but still natural way and the translation of this description of a body of knowledge, called the knowledge base, into an internal representation that can be easily manipulated by the computer. This translation is performed by another tool, called Rule Compiler, and is described in §2.4.3.

Only the important concepts of the language will be described here, for more details is referred to [Blom, 1990]. SIMPLEXYS uses rules for its Knowledge Representation. A rule's conclusion can either have the value TR (true), PO (possible or unknown), or FA (false). This is called three-valued logic and allows a very simple style of inexact reasoning.

Each rule consists of two to four parts, namely:
- a rule header,
- a rule type,
- an initial value,
- and 'thelsen'.

The rule header contains the name of the rule and an explanatory text-string. See for example the first line of Xmpl_1.

```
Xmpl_1:
MaxDosisReached: 'Maximum cumulative dose is almost reached'
btest TotalFlow > 0.90 * MaxDosis
```

There are two rule classes to be distinguished in the language:

(1) Primitive rules.
These rules are independent of other rules and get their value by some sort of direct assignment. There are 5 different primitive rules which are explained shortly here.

[a] Fact rules (FACT), which have a constant and unchanging value (TR,PO,FA).
[b] Ask rules (ASK), which are given a value by asking the user (TR,PO,FA). The answer is entered by using the keyboard.

In the blood pressure control application these first two primitive rule types are not used.
A Blood Pressure Controller

[c] Test rules (TEST), used for testing data through Pascal interfacing. The result of the test is either TR, PO or FA. Test rules are useful in control applications. A special test rule is the binary test (BTEST), which can only result in TR or FA (e.g. see Xmpl_1).

[d] Memo rules, used as memory, can only be given a value by other rules, e.g. through a then tr. Memo rules can either be given TR, PO or FA.

[e] State rules, denoting a context, assigned a value initially or via the 'protocol' (a set of ON statements that describe context switches). State rules are either TR or FA, but never PO. For example:

\begin{verbatim}
Xmpl_2:
Stabilizing: 'the flow is stabilized near the setpoint'
state
then goal: GainUpFlow2
\end{verbatim}

(2) Evaluation rules. These rules operate on a higher level and are dependent on other rules (either primitive or other evaluation rules). This is illustrated in the following example:

\begin{verbatim}
Xmpl_3:
OvershootDown: 'the pressure is decreasing too fast'
OvershootEnable and UnderSp and StateInSp < (50) and AboveBrd
then do write_debug('gain down request: overshoot down')
then tr: GnDnReq
\end{verbatim}

N.B. all examples are taken from the real rule base of the controller and therefore more complex than would be necessary to illustrate the exact feature of a rule type. So in the last example in the second line there are stated 4 names of other rules, already showing some of the logic operations possible with rules which will be discussed later in this section. Also the two last lines already show some features of the Thelses part of the rules.

In the initial value section of a rule, which is not mandatory, the rule can be given a value (TR, PO or FA) for the first run.

Finally the thelses section, which is not mandatory either, allows multiple consequences from a single evaluation. There are three types of thelses: THENs, ELSEs and IFPOs. THENs are used to allow consequences if the result of the rule evaluation is TRue. ELSEs are used to allow consequences if the result is FAlse and IFPOs are used if the result of the evaluation was POssible. A thelse must be followed by one of the next three possibilities:
- a value (TR, PO, FA).
  These thelses allow multiple conclusions from just one evaluation. Under condition that the evaluated rule has a certain value other rules will be assigned to certain values too (see Xmpl_3).
- a goal.
  The argument, which must be a rule, is evaluated immediately (see Xmpl_2).
- a Pascal section.
  This kind of thelses provide a "hook" to Pascal to manipulate data, to print or to display, etc. The Pascal statements are executed if the rule was evaluated to TR (THEN DO, see Xmpl_3), FA (ELSE DO) or PO (IFPO DO).
As already noted before it is possible to perform various logic operations with rules. The Logic used in SIMPLEXYS is very much like Boolean logic. Boolean logic is easy to use and fast, which makes it suitable for real time applications. Unlike Boolean logic, which uses two values (TR and FA), three valued logic is used instead. This means a rule's conclusion is considered to have the value 'possible' if it cannot be proved to be 'true' or 'false'. The reason for this is that it agrees better with human reasoning, because we don't see everything just in black and white. Fuzzy logic might be even closer to human reasoning but, considering the required speed and reliability of this kind of medical applications, this is probably the better alternative.

An expression consist of two entities, propositions (also called variables) and operators. In SIMPLEXYS three types of operators can be used: monadic, dyadic and history operators. They will be briefly described below; truth tables and more extensive examples of their usage can be found in [Blom, 1990].

The monadic operators (one argument) are:

- NOT R : the negation of R,
- MUST R : R is guaranteed to be true,
- POSS R : no definitive value can be determined for R,

in which R represents a rule of any type. Besides the familiar NOT two more operators are defined which assure that by combining them all monadic logic operations can be formed.

Next to the dyadic operators AND and OR, already known from Boolean logic, there are three other dyadic operators. Namely UCAND, UCOR and ALT. The ALT operand is new and stands for 'logically equivalent alternative'. In the expression: x ALT y, y is used as an alternative for x whenever the value of x cannot be determined (value of x is PO). This construct has the benefit that knowledge can be kept together in the knowledge base. It also introduces the possibility of a conflict in the case that the arguments take on opposite values in which case there is an inconsistency in the knowledge base.

UCAND and UCOR are logically equivalent with respectively AND and OR. The difference is that in contrast to the latter they are evaluated unconditionally which means that both propositions will be evaluated regardless the fact that the result of the expression already is determined by the value of the first variable. This is very useful in the inference if we think about the case that besides the value of the expression also evaluation of both argument rules is wanted because each rule could trigger other rules that need to be evaluated.

The third types of operators are history operators. Each rule has a history counter containing the period, in seconds, during which the rule remained unchanged. By performing a numerical comparison it is now possible to determine for how long a rule has had a certain value without changing. The history operators are:

- = equal
- <> not equal
- > greater than
- >= greater than or equal
- < less than
- <= less than or equal

In an expression with a history operator the left argument is a rule and the argument to the right side is a numerical expression surrounded by parentheses denoting a time in seconds, an example
is already shown in Xmpl_3. One application of history operators is, that rules can be used to
detect stable situations. The history operators are frequently used in the blood pressure control
application in rules for detection of transients, oscillation, necessity of gain adaptation because of
too fast or too slow response, etc.

To complete the brief description of the programming language the priority of the operators is
noted. In SIMPLEXYS the history operators have the highest priority, followed by the monadic
operators. The dyadic operators have a lower priority than the monadic operators again.
Operators of the same type all have an equal priority level. Like in other programming languages
priority can be forced by using parentheses.

2.4.2. The Rule Base

In this section a description of the knowledge base of the expert system will be given. In
SIMPLEXYS applications it is also called mIes base, because the knowledge is represented by
rules constructed according to the syntax described in the previous section. The rule base
however does not only contain rules but is composed of up to 8 sections which each starts with a
certain keyword. The first 6 sections are optional and contain Pascal code to create an interface
for the rules. The Inference Engine, which is described later in this report, will include these
sections without any changes. The last two sections are mandatory. The exact sections and their
keywords are:

1] USES : Libraries to be included
2] DECLS : Declarations
3] INITG : Global initializations
4] INITR : Run initializations
5] EXITR : Run exit code
6] EXITG : Global exit code
7] RULES : The rules
8] PROCESS : The protocol.

The uses section lists the Pascal units to be included. The declarations section contains the Pascal
declarations of all variables, procedures and functions that will be used by initializations, exit
codes, TEST rules and THEELSE DO's. The difference between the two initialization sections is
that the statements of the INITG section will only be executed immediately after the system
startup and the statements of the INITR section will be executed at the start of each new mn. The
concept of internal runs will become more clear in the section describing the inference.

About the exit code sections similar remarks can be made. The EXITG section will only be
executed at the end of the last run while the EXITR section will be executed immediately after
each run. The rules section must contain all the descriptions and definitions of all the rules
according to the syntax prescribed by the programming language.

Finally the rule base ends with the process section. The protocol stated in this section describes
the dynamics of the process. The total process can be divided in a number of runs, during which
the rule values do not change. So we can distinguish a static environment during which the rules
are evaluated at most once and a dynamic environment during which the rule values can change.
What Goals will be evaluated during one run mostly depends on what states are active (true) in
that run. The process section contains all state transitions or ON statements. These have the following format:

ON Trigger FROM FromList TO ToList

A state transition causes a change in active state(s); some states can become active (true) while others can become inactive (false). The transition takes place if the trigger evaluates to TRUE and if all the states in the fromlist are active. The process will end if no states are active anymore.

2.4.3. The Toolbox

Once the rule base has been created, it has to be linked with the Inference Engine to obtain a working expert system. Conversion from rule base to expert system is done by the SIMPLEXYS Toolbox.

The three main functions of the toolbox are:

a) Conversion of the rule base into an appropriate form for the Inference Engine.
   This is done by the Rule Compiler.

b) Checking the rule base for correctness and completeness. This is of course a very important aspect, for errors in the rule base will lead to erroneous behaviour of the expert system which is unacceptable.

c) Building the expert system by linking the converted and checked rule base with the Inference Engine.

This functionality is accomplished by the incorporation of the six tools shown in Figure 2.4.

Figure 2.4: The SIMPLEXYS Toolbox with the various tools and their file generation or inclusion.
The Rule Compiler translates the rule base into an internal representation which is saved in six files. Each file contains a specific part of the knowledge as denoted below.

1) rinfo.qqq : all the arrays and tables used for representing the rules and their mutual connectivity.
2) rtest.qqq : all the test sections defined in the test rules.
3) rhist.qqq : the information about the history sections.
4) rdodo.qqq : the collection of DO sections used in the rule base.
5) rinex.qqq : the initialization and exit sections.
6) ruses.qqq : the Turbo Pascal units used by the rule base.

The compiler also checks for some syntax errors and some very simple semantic errors. The latter is done more extensively by the Semantic Checker which performs several checks and generates the appropriate messages if errors are detected, providing a very powerful tool for (partially) proving correctness of the rule base.

Another checking can be performed using the Protocol checker. This tool is used for the detection of errors in the process description part (protocol). This is done quite extensively and covers syntax, topology, as well as dynamic errors. A small but useful tool during the design stage is the Option generator. This can be used to select several run-time options for the Inference Engine (e.g. the choice between real- and simulated-time or to display and save extra debugging information).

The Inference Engine tool actually builds the expert system by combining the output of the Rule Compiler with inference processes into one program. The exact inference processes of the expert system will be described in the next section. The Pascal sections are now checked for correctness by the Pascal compiler after which the expert system is ready to run. The last tool, the Tracer / Debugger, can be used after the expert system has run. It is a tool to examine the inferencing process which took place in the expert system while it processed symbolic information.

2.4.4. Inference

A main characteristic of SIMPLEXYS is that compilation of the knowledge base into a correct internal representation is necessary before the Inference Engine can use the knowledge. After binding this representation to the Inference Engine the rules can be evaluated with high speed which is of course of great importance for real time applications. The internal representation is a set of tables in which pointers denote the relations between rules.

So searching and matching is only done once. This is a good approach because all problems, if specified precisely enough, can be reduced to search problems and therefore knowledge representation is just as crucial to solving hard problems as inference techniques. In common expert systems often a huge amount of searching and matching is done by the inference engine.

In this application in general the data, offered to the system, changes over time and therefore the necessary conclusions have to be derived repeatedly, probably many times with different goals. A single run is now defined as one evaluation of all the rules. In the blood pressure control application a run lasts 5 seconds, including data acquisition, displaying the results and control of
the infusion device. The inference performed in one run is called a single run inference. All primary goals of a certain context are evaluated and the next context is determined.

More specific the following is done:

- Update time.
- Execute INITR section.
- Execute matching THELSEs of all FACT, MEMO and finally of all STATE rules.
- Try to perform context switch.
- Execute EXITR section.
- Undefine all ASK, TEST and EVAL rules for next run.

Besides single run inference, global inference is defined as the sequence of runs. The global inference process has the following structure:

- Initialize the conclusions and history values of all rules.
- Obtain conclusions of FACT rules not initialized directly (by INITIALLYs).
- Initialize the time.
- Execute INITG section.
- Repeat a single run inference until no STATE rule has conclusion TRUE anymore.
- Execute EXITG section.

At the top level, SIMPLEXYS reasoning is backward or goal oriented. In fact the pattern of reasoning is a combination of both forward and backward chaining. During a single run inference all primary goals of the given context are evaluated (i.e. backward chaining). Primary goals are usually EVAL rules, but they can also be ASK or TEST rules and through the THELSEs this can result in evaluation of secondary goals. This is a kind of forward chaining, because the result of the evaluation of the primary goals determines whether or not certain secondary goals have to evaluated.

Further STATE rules and the ON statements in the process section (protocol) determine the next context and thus which goals are to be evaluated during the next run. To decide whether or not a context switch should be made, the trigger rules are evaluated, which again is backward chaining. So the inferencing is rather complex. But together with the internal knowledge representation this makes the inference of the expert system very efficient and well suited for real time applications.

2.5. Implemented Mechanisms

The supervisory expert system is used to implement a number of mechanisms which are necessary to improve controller performance and safety. The expert system analyzes a specific situation or process state and, according to the medical knowledge incorporated in the rules, the controller is adapted or overruled. In this section the transient detection, the oscillation detection and the various mechanisms for gain adaptation will be briefly described. For the exact details is once more referred to previous reports [Hoeksel, 1992; Geene, 1993; Lammers, 1990 and Zwart, 1990]. The prototype of the controller used in the clinical evaluation, which is reported on in chapter 5 of this report, contained the same rule base as the controller described by Geene and therefore the mechanisms are equivalent too.
2.5.1. Gain Adaptations

Requirements for the PI-controller are: a smooth change of the flow after a setpoint change, a smooth change of the MAP to that target level, and safe behaviour for patients whose characteristics are far from nominal. For safety reasons the controller initially assumes a very sensitive patient, and thus control gain is low. However, in many cases the patient's sensitivity will be less. To cope with the complete range of the patient's sensitivity the gain of the PI-controller must be adapted. The gain is selected out of 9 classes which vary between a value of 1/9 and 9, each differing a factor $\sqrt{3}$. A gain of 1 belongs to a patient with nominal sensitivity of 25 [mmHg/µg/kg/min]. In the ideal case the control gain should be inversely proportional to the sensitivity (see Table 2.1), then controller adaptation is correct. Control remains good when the gain differs at most 2 classes from optimal. The MAP may overshoot the setpoint more than 10 mmHg or oscillate if the gain differs more than 2 classes from optimal.

Table 2.1: The control gain is adapted until it equals the inverse of the standardized sensitivity. A sensitivity of 25 mmHg/µg/kg/min is standardized on the value 1.

<table>
<thead>
<tr>
<th>Sensitivity of the patient</th>
<th>very insensitive</th>
<th>....</th>
<th>normal</th>
<th>....</th>
<th>very sensitive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain of the controller</td>
<td>1/9</td>
<td>1/(3$\sqrt{3}$)</td>
<td>1/3</td>
<td>$\sqrt{3}$/3</td>
<td>1</td>
</tr>
</tbody>
</table>

The gain selected by the expert system is determined by various mechanisms, comparing the MAP change with a minimum or maximum which is allowed in that context. A measure of progress is necessary to determine this MAP change. Because the sensitivity of a patient is also variable in time, gain adaptation must be a continuous process. The controller regulates the MAP towards the setpoint along an ideal curve in time, see Figure 2.5(a). The general gain adaptation mechanism performs gain adaptation when the real curve deviates from the ideal curve.

Figure 2.5: Gain adaptation mechanism; (a) ideal curve towards setpoint, (b) ideal derivative and allowed deviation bound.
The derivative of the pressure curve is a measure of progress. A minimum and maximum derivative is tabulated for each discrete distance to the setpoint. Figure 2.5(b) shows the expected derivative as function of the distance to the setpoint, the bound around this ideal derivative is the allowed deviation. At error rates larger than 30 mmHg the I-term limitation (Eq. 2.5.a, § 2.3.2) prevents further acceleration and the maximal flow rate is reached.

To prevent the adaptation to be influenced by noise and transients, the progress over a specific period is used instead of the derivative. This progress is defined as the difference between the minimum and the maximum distance to the setpoint during a number of samples. The number of samples taken into account is different for gain up and gain down adaptation. For general gain up adaptation a window of 30 samples, i.e. 150 seconds, is used. The general gain down mechanism only uses 7 samples (35 seconds) which ensures a fast reaction in case of overestimation.

Gain adaptation always consists of two steps. First a change of the gain is requested, depending on certain conditions which must all be satisfied. Secondly, a number of constraints must be fulfilled, otherwise the request will not be acknowledged and no change appears. So the gain is only adapted after a request has been acknowledged. For the general adaptation mechanism the conditions which initiate a request are:

(1) Gain up:  
- The progress is too slow (smaller than a minimum derivative).
- The MAP is moving towards the setpoint. This to avoid adaptation due to pressure changes not caused by SNP. In engineering terms: the current MAP is near 2 mmHg of the minimum distance to the setpoint measured during the last 30 samples of the pressure.
- The sampled pressure values are out of the 6 mmHg bound around the setpoint. This to distinguish changes from noise and transients.

(2) Gain down:  
- The progress is too fast (larger than a maximum derivative).
- The MAP is changing towards the setpoint. Implemented by the request that the current MAP should be at most 2 mmHg away from the minimum distance to the setpoint measured during the last 7 samples.
- The smallest distance is out of the 4 mmHg bound.

Correct decisions are drawn when the control flow is adjusted in one direction for some time. To ensure this the following five constraints must always be satisfied, if a request is to be acknowledged:

(i) The MAP is above or below the setpoint for more than 4 minutes.
(ii) The controller is in automatic mode for more than 4 minutes.
(iii) No transients were detected for 3 minutes.
(iv) The last 3 minutes the setpoint was constant.
(v) No gain change appeared for 3 minutes.

Further an extra constraint for acknowledgement of a gain down request is that the flow was above its minimum 2 or 3 minutes ago, otherwise the pressure change was probably not caused by a flow change. Gain up adaptation also has extra constraints. These can be summarized with the term: "saturation". Three types of saturation are distinguished: Control is not effective because there is almost no progress towards the setpoint measured over the last 50 samples, the flow arrives at its minimum or maximum within 2 minutes, or the flow change exceeded 50%
during the last minute. Finally the gain can of course not be increased or decreased more than, respectively, the maximum (9) or minimum (1/9) level.

Due to false interpretation of changes in the mean arterial pressure, the general adaptation mechanisms can fail. To ensure robust and safe control, several other mechanisms are implemented, which also request a gain change. The first mechanism detects an overshoot of the setpoint, which may occur when the gain is too large. The general gain down mechanism is not active around the setpoint. Therefore, this overshoot detection mechanism generates a gain down request when the right conditions are satisfied.

Another situation, which is not taken into account in the general gain down mechanism, is a large setpoint change up. Small setpoint changes are accumulated. This mechanism should act if, due to a low target level, the flow rate is high and the patient becomes less sensitive, caused by the non-linear response to SNP. The gain can be adapted upwards in such a case. A setpoint increase will most probably undo the effect and therefore the gain should be decreased. This mechanism will barely trigger because large setpoint changes upwards almost never occur.

The third and final gain down mechanism is initiated by detection of a pressure oscillation, which is described in the next section. There is also an extra gain up mechanism implemented. When the flow rate is large and the gain is small the controller cannot act fast enough during hypotensive periods. To limit this effect, a minimum gain is defined. The gain noted not realistic when the current gain multiplied by the pressure decrease, belonging to a patient with optimal gain 1 and a nominal sensitivity of 25 mmHg, is larger than 80 mmHg, see equation 2.7.

\[
\text{Gain} \left( \frac{25 [\text{mmHg} / \mu\text{g} / \text{kg} / \text{min}]}{\text{flow [\mu\text{g} / \text{kg} / \text{min}]} > 80[\text{mmHg}]} \right)
\] (2.7)

A negative filtered error (distance to the setpoint) and a non-realistic gain generate a gain up request. This mechanism is active when artifacts, noise and false oscillation detections mislead the other gain adaptation mechanisms.

2.5.2. Oscillation Detection

A too high gain in combination with an unexpected large time delay between drug delivery and patient response can result in an oscillation in the pressure signal. In most cases, such an oscillation is triggered by events not related to SNP infusion. This can be the infusion of other drugs or for example switching to cardiopulmonary bypass (CPB). The latter introduces an additional delay time up to 45 seconds, dependent on the extra blood volume in the perfusion machine. The nitroprusside has to enter this heart-lung machine first because infusion is only allowed in the venous system to protect the patient against small air bubbles. Instead of changing the PI-parameters, which would be the obvious thing to do because the patient model is changed now, the controller tries to cope with this extra delay time through gain adaptation if necessary. Determination of an optimal set of PI-parameters for the bypass phase should be considered as an option to improve controller performance though.

The controller has some mechanisms to prevent oscillation with large amplitude. Pressure and flow oscillation are detected separately because noise and artifacts in the MAP destroy the interesting correlation between flow and pressure measurement. When the system is oscillating, the pressure signal is alternately higher and lower than the setpoint. Therefore the number of
setpoint crossings per unit time is used as a Figure of merit. Influence of random fluctuations around the setpoint is limited by hysteresis boundaries, see Figure 2.6.

A counter is incremented by each sequential crossing of these borders. If the counter reaches a value of four an oscillation is detected. Because only too many border crossings within a short time period are regarded as an oscillation, the counter is decremented every 250 seconds.

![Figure 2.6: Pressure oscillation detection.](image)

The minimum value of the counter is zero. The counter is also decremented after a setpoint change, and reset to zero when a transient has been detected, because these events can cause incorrect increments. The oscillation boundary above the setpoint is smaller, so the mechanism is sensitive to a MAP oscillation with an average slightly below the setpoint. A gain down request is made if an oscillation in the pressure is detected.

There are patients whose pressure is inherently unstable: the MAP shows spontaneous fluctuations comparable to oscillations. The best action in such cases is also to decrease the gain, in order to stabilize the control. So the oscillation detection mechanism acts safely in these cases too.

Besides pressure oscillation detection, there is also a mechanism implemented for detection of a flow oscillation. The reason for this is that flow changes are not very effective if the pressure is near the setpoint but fluctuating fast. Such a varying flow can in combination with transients lead to pressure oscillation. Due to the response time of the drug, fast small pressure oscillations cannot be reduced. In fact the MAP is stable near the setpoint than, and the flow rate has probably almost reached the optimum. So a better strategy is to change the control mode to stabilizing if this kind of oscillation is detected.

The flow oscillation detection differs from the pressure oscillation detection. This mechanism searches for extrema in the flow which deviate at least an amplitude A from the previous maximum or minimum. The amplitude A is given in units of $\mu g/kg/min$ and defined according to equation 2.8. This ensures equal safety for all patients because it is dependent on the gain and corrected for the patient weight.

$$A = Gain \cdot \Delta F_{\text{max}} \cdot 60[s]$$

A flow oscillation is detected when six sequential deviations of this amplitude in opposite directions have occurred within a short time period, see Figure 2.7.
Equivalent to the pressure detection mechanism, the counter is decremented every 250 seconds. When a flow oscillation is detected and the MAP is near the setpoint, the PI-controller is switched to stabilizing mode.

To prevent interference between pressure and flow oscillation detection, the flow oscillation counter is reset when the pressure oscillation counter becomes two, and the pressure oscillation counter is reset 30 seconds after a flow oscillation is detected.

2.5.3. Transient Detection

Control of the closed loop system is based on no other information than the pressure signal. An unexpected, temporary, large pressure increase or decrease can occur in this signal due to external causes. Such a pressure change is defined as a transient. They can be caused by pain, surgical actions, infusion of other drugs, a state of shock, etc. The controller should not attempt to regulate the pressure towards the setpoint during a transient. This would require a large change in the flow rate which the controller is not able to achieve fast and accurately enough. Besides the system must recover to the pre-transient value again when the transient is over.

Further it is undesirable to try to suppress a transient because a fast changing pressure is a diagnostic aid for the anaesthetist. Compensation with SNP for a fast rising pressure that is caused by pain should be avoided.

Thus, transients need to be detected. To accomplish this, the new MAP is compared with previous measurements each sample time; when the difference exceeds a certain limit, a transient is detected. The slower the MAP changes, the more it must rise before a transient is detected. The system distinguishes between up and down transients, because a different strategy is necessary to act safely in each case. When an unexpected large increase in pressure is detected, the flow rate is only allowed to vary below the pre-transient rate and not to exceed it. An up transient is detected when the MAP rises more than 20 mmHg in less than 15 seconds, or 25 mmHg in 30 seconds, or 40 mmHg in 150 seconds.

In case of a down transient, limitation of the flow rate below a certain level is not the safest strategy. A low pressure can be dangerous, so it is assumed that the quick decrease is permanent (worst case) and the flow is shut off completely first. After this it is only allowed to vary below the pre-transient value. If the decrease turns out to be temporary after all the flow rate is resumed at the pre-transient level.
The end of a transient is detected when the MAP is almost back at the pre-transient value or if a certain time has past. In the latter case the change turns out to be permanent and normal control is continued. This allows short periods of hypertension in case of incorrect detection of an up transient which cannot be avoided because it is unknown in advance whether a large pressure change will be temporary or permanent.

When a transient is detected, the expert system overrules the PI-controller and restricts the change of the flow rate. This causes the pressure and the flow to deviate from normal. As a result other decisions, e.g. about gain adaptation and oscillation detection, are disabled. Due to this, unnecessary detections generally are disadvantageous, because control is less effective for some minutes. To prevent incorrect transient detections, three additional mechanisms are implemented which determine whether a transient detection should be acknowledged or not. These take into account abrupt manual adjustments of the flow rate, changes of the setpoint and small successive pressure changes in opposite directions.

2.6. Other aspects of the Design

2.6.1. Knowledge Acquisition

The elicitation of knowledge is done through interviews with 'domain experts', reviewing literature and attending operations during clinical tests at various hospitals. Also experiments with Yorkshire pigs were done to gain a better understanding of the variability of SNP and construct a model for simulation purposes. After this the knowledge was implemented in the knowledge base and a running program was built.

Imperfections and errors are corrected by repeating the process of knowledge acquisition, implementation in an improved running version and performing simulations and tests again. This procedure is called 'Rapid Prototyping'. SIMPLEXYS provides an expert system shell and the choice to use it eliminates the conceptualization and formalization stages of the knowledge acquisition procedure. The concepts to represent and the structure to organize knowledge are already determined then.

The three stages of the procedure left now are: identification, implementation and testing. These were and still are repeatedly gone through in order to improve the performance of the expert system continuously. More about the practical performance will be said in a later section. First some short remarks about the user interface will be made.

2.6.2. The User Interface

There are two different classes of users for this expert system application: clinicians and knowledge engineers. Because they have different requirements there are in fact two user interfaces. First there is a simple clinically acceptable interface. Giving a good numerical and graphical overview of all the important parameters (flow rate, pressure, setpoint, etc.) and showing the possible commands with the appropriate function-keys. An example is shown in Figure 2.8. This user interface is very easy to use.
The user interface for the knowledge engineers is more complicated because it must allow a complete reconstruction of the internal (inferencing) and external (control) performance. Various information is stored in files and some tools exist to review and analyze the performance. The Tracer / Debugger, already mentioned together with the other tools of the SIMPLEXYS toolbox, can, for instance, detect which mechanisms have been involved at run time at which times.

2.6.3. Previous Tests

As stated at the beginning of this chapter the described prototype evolved over the years. Besides simulations, a number of real tests during cardiac surgery were performed. The first tests were done at the Catherina Hospital in Eindhoven (NL) [Zwart, 1990]. Although this showed the system to be practical useful, various improvements resulted from the test. At a later stage a similar expert system application, using Nitroglycerine as the pressure lowering drug, was built [Hoeksel, 1992] and used at the Academic Hospital of Maastricht (NL), where it is still being tested. The clinical evaluation described in this report is a follow-up of initial tests performed at the Erasme Hospital in Brussels (B) [Geene, 1993].
Chapter 3

An Infusion Toolbox

Total intravenous anaesthesia requires an intelligent combination of various drugs, which should be continuously administrated in order to establish and maintain desired plasma concentrations with great precision. In this chapter a complete guiding system to assist the anaesthetist to deliver simultaneously many drugs to a patient is described. This system, called Infusion Toolbox, is studied and described in order to create a design for integration of the blood pressure controller and the Toolbox. First the system setup is shown through a global survey of the hardware, the communication network and the software. Secondly, an outline of the incorporated modules and the specific tools they provide is given.

3.1. The System Setup

The Infusion Toolbox is a system implemented in an Object Oriented Programming (OOP) environment (Smalltalk) and runs on a PC connected to a Universal Device Communication Controller (UDCC) through a serial line (RS232-link). The total system is shown in Figure 3.1. The devices connected to the UDCC in the figure are just shown as examples.

This system evolved from earlier systems and software programs, see [Tavernier, 1987], [Cantraine, 1988 and 1989], [d'Hollander, 1989] and [Barvais, 1989 and 1991], which have been...
An Infusion Toolbox

embedded into an OOP-environment and whose features have been improved. In the design for integration of the blood pressure controller and this system, which will be extensively described in the next chapter, the controller will in fact be seen as a special peripheral device. The Infusion Toolbox is the server of the controller (client) and according to the received information controls one of the infusion devices.

3.1.1. The Hardware

Many of the variety of infusion devices available today are equipped with a serial (RS232) port through which they can be controlled by a computer. Due to the lack of standardization all of them unfortunately have their own command language and communication protocols. There is no pump with a Device Communication Controller (DCC) interface, conform the Medical Information Bus (MIB) standards (Table 3.1), so far. Nevertheless the design of the communication network of the Infusion Toolbox is close to these MIB standards.

Table 3.1: Medical Information Bus (MIB) standards.

| P1073.1 | Overall architecture, defines high level interface between device and host, and language by which they communicate (MDDL). |
| P1073.2 | Defines physical and data link connection (through layer 4 of ISO-OSI model) between BCC and DCC. |
| P1073.3 | Defines connection to remotely located hosts. |

The network based on RS232 has a star topology and a kind of Bedside Communication Controller (BCC) in the PC. This BCC works as a usual DOS peripheral device driver and is called INFU. Data between programs and devices is exchanged via the Universal Device Communication Controller (UDCC) by creating handles referring to INFU.

The RS232 serial communication network implementation consists of two parts:

1. A data network of 8 channels multiplexed on the PC serial line which is used to exchange data between the computer and devices.

2. A control network of two kinds of connections:
   - logical lines from UDCC to devices (DTR) and vice versa (DSR), and
   - a high priority channel on the data network from UDCC to PC transmitting device connection or disconnection and control messages that check if the computer is really working.

The UDCC is a piece of hardware that provides up to 8 medical devices with a DCC-like interface. The purpose of the UDCC is to supervise the peripheral connections, to check that the computer is working, to provide the pump devices with a DCC-like controller and to provide extra services in order to keep the pumps standby or recover failures.

To be precise, the UDCC hardware consists of: a Motorola MC68HC11 2.0 MHz CPU with timer (4.1 ms), 8 bits A/D converter, 4 logical I/O ports and a serial port; a memory of 16K External Ram, 0.5K EEPROM and 8K ROM; further as peripherals 4 DUART MC68681 with 2 serial ports, 1 parallel output (8 bits), 1 parallel input (6 bits) and timer (3.6864 MHz).
3.1.2. The Software

A small but crucial software program in the total design of the Infusion toolbox is the INFU driver. This driver (gginn.sys) is loaded into the higher memory at startup of the PC. One part of this driver is programmed in assembler and is based upon the common DOS tty-driver. The other part of the driver, which implements procedures with higher level functionality, is implemented in C. The driver INFU layers can be divided in two parts: a Logical Unit (LU) and a Slot.

The LU is the master of the Slot it is connected to and runs in the background on the computer. It provides the system with an abstract view of the devices which enables programming any kind of process independently of the device’s hardware characteristics. Further the Logical Unit manages time scheduling of requests to be sent, stores input messages and signals the status of the Slot. The Slot carries out the communication protocol conversion for the device it is connected to and extracts the data requested by the LU from the monitoring devices.

In order to accomplish the communication between devices and the main programs independently of the device, a virtual device command syntax has been developed. In this syntax a command has a header and a tail part. The header switches the transmission to the command queue or register, while the tail forms the command for the device. Some examples of the syntax, in this case directly used in DOS to communicate with the INFU driver, are shown in Table 3.2.

<table>
<thead>
<tr>
<th>Table 3.2: examples of the command syntax.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Start up the Transport Service (Ts))</td>
</tr>
<tr>
<td>echo off</td>
</tr>
<tr>
<td>echo &quot;&gt;0 G&quot; &gt;infu</td>
</tr>
<tr>
<td>echo &quot;&gt;1 I Ts:1:1&quot; &gt;infu</td>
</tr>
<tr>
<td>Link Transport service for device 1 to Slot 1 to serial port 1</td>
</tr>
<tr>
<td>(request status)</td>
</tr>
<tr>
<td>echo &quot;&gt;1:E&quot; &gt;infu</td>
</tr>
<tr>
<td>copy infu con</td>
</tr>
<tr>
<td>Request status of transport service</td>
</tr>
<tr>
<td>Copy the answer to the screen</td>
</tr>
<tr>
<td>(start data sent)</td>
</tr>
<tr>
<td>echo &quot;&gt;1:D&quot; &gt;infu</td>
</tr>
<tr>
<td>copy infu con</td>
</tr>
<tr>
<td>copy mess infu</td>
</tr>
<tr>
<td>Indicate that the next message is data to be sent to device 1</td>
</tr>
<tr>
<td>read from the driver buffer</td>
</tr>
<tr>
<td>send the message</td>
</tr>
<tr>
<td>(set infusion rate)</td>
</tr>
<tr>
<td>echo &quot;&gt;1 500 300&quot; &gt;infu</td>
</tr>
<tr>
<td>Put an infusion of 500 ml/h during 300 seconds in queue of device 1</td>
</tr>
<tr>
<td>(reset)</td>
</tr>
<tr>
<td>echo &quot;&gt;1!&quot; &gt;infu</td>
</tr>
<tr>
<td>reset the transport service</td>
</tr>
<tr>
<td>(N.B. Dos commands and comments are noted in this font.)</td>
</tr>
</tbody>
</table>

The various software modules of the Infusion Toolbox, which will be discussed more extensively in the next section, are implemented in the object oriented programming environment: Smalltalk. In fact Digitalk's widely used implementation, Smalltalk/V 286, is used. However, most of the concepts and techniques can easily be transferred to other versions of the language. Even the transfer to a version for another operating system instead of DOS, for example Unix or Apple systems, will cause minor problems because Smalltalk creates a virtual machine.

The power of OOP-techniques lies in the ability it gives the designer to manage the complex, rapid changing applications, which have become the catchword in modern software. This is important because medicine is complex and we should be able to build quickly and safely new
prototypes of infusion tools based on theoretical assumptions, which then can be tested. Moreover, in this application in particular, abstract objects can be defined which are close to the anaesthetist's world and involve the anaesthetist vocabulary to express what it has to do and has to know. This makes collaboration between the anaesthetist and the computer scientist much easier.

Some class and subclass examples out of the universe of classes which define the modularity of the system are: a class `DeviceController`, whose responsibility it is to supervise the device connections, data transmissions and protocol conversions, and a subclass `Pump` which controls the real pump device and adds extra services the hardware doesn't offer to the pump controller. Another example of the class hierarchy is the class `DataManager`, which provides the system with data it has to know, for example through the subclass `Patient`, which knows the patient characteristics, or subclass `Drug`, which contains all information about drug usage.

Before the designed infusion tools and their features are discussed, it should be noted that a uniform user-interface is created, based on various elements which are used in different combinations for each tool. These elements or objects are called WinTools in order to make clear that they don't have the behaviour of their corresponding Widget (X Toolkit, OSF/Motif).

3.2. The Tools

The Infusion Toolbox system consists of three modules. In this section the structure and functionality of these modules will be described. The first module SPINA is used to select the most appropriate drug and evaluate the pharmacokinetics. The second module CINA is used to prepare the infusion sheet, and the third module MINA provides the means to start up and control drug delivery and synchronize the infusion pumps.

3.2.1. Simulating the Pharmacokinetics for INfusion of Anaesthetic drugs (SPINA)

Knowledge of the manner and speed of resorption, distribution and excretion of drugs concerning their pharmacological, therapeutical and toxic behaviour is referred to as the pharmacokinetics of drugs. The anaesthetist has to use this knowledge to determine the posology to be applied during each specific operation. The term posology is used to identify an infusion regimen, possibly containing several successive steps. To determine the posological scheme best suited for a patient with given age, weight, biometry, pathology and surgical indication is a complex matter. The SPINA module is developed to help the anaesthetist to evaluate a chosen posology before, during and after surgery.

SPINA is a powerful tool for simulation of the theoretical drug distribution and infusion rate according to bi- or tri-compartment pharmacokinetic models. An example of a general tri-compartment model is given in Table 3.3. To permit the selection of the most appropriate model for a specific patient, SPINA contains a database of pharmacokinetic model records retrieved from the literature. The records are classified according to four criteria: age, biometry, pathology and surgical indication. The corresponding keywords help the anaesthetist to select a model. A model record consists of four main sections: parameters, keywords, bibliographical references and comments.

Once a pharmacokinetic model has been chosen a pharmacokinetic simulation can easily and rapidly be performed after introducing the target plasma drug concentration. After the
anaesthetist has composed a posological scheme SPINA provides the resulting theoretical drug distribution between the plasma and peripheral compartment(s) and the cumulated dose evolution. Pre-, per- and post-operational graph presentation of the distribution can be requested.

Table 3.3: Open tri-compartment model [Tavernier, 1987].

<table>
<thead>
<tr>
<th>Equations:</th>
<th>$rac{dC_1}{dt} = -k_{11}C_1 + k_{21}C_2 + k_{31}C_3 + u(t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$rac{dC_2}{dt} = k_{12}C_1 - k_{22}C_2$</td>
</tr>
<tr>
<td></td>
<td>$rac{dC_3}{dt} = k_{13}C_1 - k_{33}C_3$</td>
</tr>
</tbody>
</table>

where: $C_n(t) = \text{drug concentration in compartment } n [\text{mg/l}]$.

$k_{ij} = \text{transfer rate constant of drug from compartment } i \text{ to compartment } j$.

$k_{10} = \text{elimination rate constant of drug from compartment } n$.

$k_{11} = k_{10} + k_{12} + k_{13}$

$u(t) = \text{function of flow of drug injected into compartment } n [\text{mg/l/sec}]$

The posological scheme normally starts with an initial injection to load the system. Ordinarily this primary injection is of very short time duration to avoid any overloading of the plasma concentration or to reach the desired concentration as fast as possible. SPINA can compute an optimal injection for this. An example of an infusion sequence and the resulting plasma concentration in time is given in Table 3.4 and Figure 3.2.

Table 3.4: Five-step infusion sequence of alfentanil deduced from the theoretical optimal infusion rate calculated from the pharmacokinetic parameters of Shafer's model (A) to generate a constant plasma concentration of 0.2 µg/ml [Barvais, 1989].

<table>
<thead>
<tr>
<th>Step</th>
<th>Dose [µg/kg]</th>
<th>Flow rate [µg/kg/min]</th>
<th>Duration [min]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>40.8</td>
<td>-</td>
<td>Loading dose</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>2.8</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>1.8</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>1.0</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>0.88</td>
<td>80</td>
</tr>
</tbody>
</table>
When an appropriate posology has been determined the next step is to examine whether the prepared infusion scheme can effectively be executed during the surgical procedure with the infusion devices available in the operating room (OR) or intensive care unit (ICU). To support the anaesthetist with this task the module described in the next section was developed.

3.2.2. Composition of INfusion sheets for Anaesthesia (CINA)

Drug packaging, reservoir sizes and infusion device features, such as minimal and maximal flow rate and accuracy, can constrain the application of the optimal infusion scheme determined with SPINA. Furthermore, the amount of drug to be infused is preferably (by the anaesthetists) noted in units of $\mu$g/kg/min, but the flow rate units of the infusion devices, on the opposite, are given in volume per time units, e.g. ml/hour. Due to this, computation of the optimal drug concentration and conversion of the prescribed infusion flow rates into device units can be tedious and often confusing arithmetic problems. For this purpose the CINA module has been developed.

CINA is used to prepare the infusion sheet and can further be used to verify the feasibility of a posology, to optimize the drug concentration and, in case of multistep infusion, to check the precision which can be obtained by usage of a certain infusion device. CINA is based on the same architecture as SPINA and contains three databases: one for the infusion devices, one for the drugs packaging and one for the standard posologic scheme. Records in these databases can of course be modified, deleted or added. They are classified according to keywords for five selection criteria concerning the patients of the reference study. The criteria are similar to the four used by SPINA (age, biometry, pathology, indication) plus a fifth criterion called 'user', which enables the user to give each posology a specific identification.

CINA augments the flexibility in preparation of solutions by proposing different drug concentrations of the reservoir. It preferably determines the drug concentration and total amount of drug needed which requires the minimum drug package units provided by the hospital pharmacy. All the information available is used to compose and save an infusion sheet. This infusion sheet then serves as the basis for control of infusion during the operation.
3.2.3. Monitoring of Infusion for Anaesthesia (MINA)

This last module controls the communication network as described in the first section of this chapter. MINA starts up and controls drug delivery and synchronizes the infusion pumps. It ensures accurate drug titration, allows the coordination of the various infusion devices according to time and external events, informs the user about the current and future status of a drug infusion scheme and records the delivered flow rates. It is also possible to interact on the preprogrammed infusion sequences.

All this is realized by linking the various objects, containing the information about a specific drug, infusion device, patient and chosen posology, together in an object called ControlPanel. The ControlPanel is also linked to a Slot of the transport service (INFU-driver). Once a relation between an object and a ControlPanel is defined this is 'known' all through the system. This ensures safety, because a combination of certain equivalent objects cannot be made. A specific drug, pump and INFU-Slot can only be linked one single time. This assignment is always carefully checked for security. Moreover, a change made in the settings of the ControlPanel, or an object assigned to it, is also immediately 'known' by all objects directly related to each other via the ControlPanel. The inheritance principle of the OOP-language shows its advantage here.

Several of these ControlPanels, for different drugs and devices, can be defined and displayed. This way many drugs and infusion devices can be controlled and monitored simultaneously from a central location. The number of devices which can be controlled simultaneously is limited to 4 at the moment, because the anaesthetists do not perform computer controlled infusion of more than four drugs simultaneously during the clinical evaluation of the system in cardiac surgery. Therefore the used UDCC is not equipped for more. At the moment MINA provides different communication interfaces for up to 12 commercially available infusion devices (Imed, Ivac, Braun etc.).

Other practical features of MINA are the ability to retrieve any preprogrammed infusion sheet for standard cases and the on-line possibilities to edit the infusion sheet, which gives extra flexibility. Further the MINA module generates warning messages to notify, for example, infusion completion or to request syringe replacement. Also audible alarms and explaining messages signal problems like, for instance, occlusion. Finally the possibility to enter comments during surgery should be noted. Together with the command actions performed by the anaesthetist, these comments are saved in an archive. The archives can be analyzed afterwards.

3.3. Conclusions

The Infusion Toolbox is a sophisticated system which provides the anaesthetist with an aid for selection of an adequate infusion scheme and preparation of the infusion sheet for a given patient. Moreover, simultaneous delivery of various drugs can be controlled and monitored safely from a central location. Therefore it will be a good basis for a machine for total intravenous anaesthesia.
Chapter 4

Extension of the Infusion Toolbox with the Blood Pressure Controller

In the previous chapters the blood pressure control system and the Infusion Toolbox have been described. In this chapter a design for integration of both systems is presented. In order to control the SNP controller and retrieve information from it through the serial line of the PC, we will examine the user requirements and define a communication structure to support it. To make it possible to adapt to future developments and to make the structure clear to everybody who wants to use it, the start is from a general point of view with an object oriented analysis. A flow diagram of the program and a communication model show the general implementation of the design. Further the transport service, the communication protocol and some other aspects are discussed in more detail. Finally some conclusions are drawn and recommendations for the future are made.

4.1. Object Oriented Analysis

The design process was started with a thorough analysis. First the information that could possibly be transferred from or to the controller is derived. Secondly an object oriented analysis of the SNP controller is made and a flow-chart of the communication process is generated. Finally, to complete the analysis a short description of the communication model is made.

Let us start by considering the data flows.

From controller: pressure (mean arterial, diastolic and systolic pressure), SNP-flow rate and units, setpoint, messages (alarms, control requests), gain, status (auto, manual, in or out of bypass). In the future e.g.: temperature, statistical data, heart rate,...

To controller: patient weight, SNP concentration, control information (from keyboard), flow rate, events (e.g. skin incision), evaluation data (names, comments, etc.), used site (OR, ICU, etc.).

To clarify the view and find out what actually is necessary to fulfill the goal an object oriented analysis is made. Each object represents a certain functionality of the existing controller with its attributes and services. Because the goal is to accomplish communication with the Infusion Toolbox, in order to make remote control of the SNP-controller possible, we focused on the...
objects related to the desired tasks. Figure 4.1 shows the graphical presentation of the objects and their relationships.

The basic system can be divided into 8 objects, namely: SNP_Controller, PI_Controller, ExpertSystem, DisplayServer, PumpServer, SignalServer, PatientModel and Archive. Although the names almost speak for themselves the functionality of each object will be described in general before we will focus on their attributes and services in particular.

![Object Oriented presentation of the Blood Pressure Controller.](image)

**Figure 4.1:** Object Oriented presentation of the Blood Pressure Controller.

The centre of the network is formed by the **SNP_Controller** object which represents the main frame of the system. The main parameters guiding the total process are collected here. All the
Extension of the Infusion Toolbox

Tasks concerning the normal pressure control are embedded in the object **PI_Control** while the supervising expert system is denoted by the object **ExpertSystem**. This object is linked to a number of other objects which are shown for completeness and to show that it is in fact also a complete toolbox, which is called SIMPLEXYs, see Figure 4.2. Further there is an object **DisplayServer**, which handles all the screen, keyboard and mouse tasks, an object **PumpServer**, which provides the necessary items to control an infusion pump and an object **SignalServer**, which contains all attributes and services dealing with the blood pressure signal in real time. For simulation purposes an object **PatientModel** exists to generate a pressure according to the infusion flow rate. Finally there is an object **Archive** for handling of the data that needs to be stored. As shown in Figure 4.1 there is also an object **Analysis** that is linked to the Archive object. This object denotes the tools the user has to replay the stored data or perform statistics off-line.

Now we will go into more detail and look at the attributes and services of each object.

**SNP_Controller**: PI-controller supervised by an expert system which, based on medical knowledge, determines the necessary flow rate of SNP to control the Mean Arterial Pressure (MAP) to a specified target level (setpoint).

**Attributes**:
- **alarm**: Denotes if there is an alarm and the type of that alarm.
- **controlFlowRate**: The current flow rate used by the system.
- **controlMode**: The current Mode of the system, either Manual or Automatic.
- **map**: The current Mean Arterial Pressure used by the system.
- **quality**: Validity of the MAP (valid / invalid).
- **setpoint**: Current setpoint.
- **surgeryState**: State of surgery at the moment, e.g. normal or bypass.
- **usage**: Denotes if the system is used for real-time or simulation purposes.

**Services**:
- **startAlarm**: Sets alarm of certain type.
- **stopAlarm**: Resets the alarm.
- **changeFlowRate**: Updates the controlFlowRate to the value determined by last inference.
- **changeMode**: Change Mode from Auto to Manual and vice versa.
- **beginTime**: Sets begin time of event or state.

**PI_Controller**: Simple robust controller which tries to minimize the difference between the MAP and the setpoint by increasing or decreasing the flow.

**Attributes**:
- **gain**: The current gain of the PI-controller.
- **parametersPI**: P and I term values used by the controller at the moment.

**Services**:
- **changeGain**: Increase or decrease the gain, determined by the expert system according to sensitivity of the patient.
- **changePI**: Change P and I parameters to other values dependent on current control mode (stabilizing or regulating).
- **flowrate**: Return new flow rate, calculated from current MAP and setpoint, to expert system.
ExpertSystem: Real time, rule based expert system, in our case built with the expert system programming language and toolbox SIMPLEXYs. Supervises the PI-controller based on medical knowledge.

At the moment this and related objects are only listed below but not described in the analysis. The objects, as shown in Figure 4.2, are: ExpertSystem, KnowledgeBase, Rules, Theses, InferenceEngine, Process, RuleCompiler, OptionsBuilder, ProtocolChecker, PetriNetChecker and SemanticsChecker.

Figure 4.2: Object Oriented presentation of a SIMPLEXYs expert system and the toolbox.
DisplayServer: Interface with keyboard and screen.

**Attributes:**
- **controlPanel:** All about how to display the control information.
- **event:** Event that occurred due to user action through keyboard or mouse.
- **formular:** What to display in a formular structure where the user has to fill in a lot of data (medical or evaluation).
- **menu:** What to display in a certain menu structure.
- **message:** What to display to show a message.
- **timeOut:** Time after which a null event has to be sent if nothing happened.
- **trend:** What to display to show the trend of a signal (e.g. ratio, mean or moving average of pressure or flow).

**Services:**
- **display:** Display the given information using the requested attribute.
- **setTimeOut:** Set the time delay for an event to be returned after a request.

PumpServer: Controls the pump, gets desired flow rate of drug, and transforms it to the right pump units.

**Attributes:**
- **pump:** Type and address of the pump
- **controlStatus:** Status of the pumpServer, e.g. busy, waiting, alarm.

**Services:**
- **start:** Starts infusion
- **stop:** Stops infusion
- **delay:** Returns delay of the pump (time requested to perform one action).
- **resolution:** Returns resolution (smallest flow increment).
- **pumpStatus:** Returns the status of pump, (e.g. alarm type: occlusion).
- **changeFlowRate:** Increases or decreases flow rate according to request.
- **currentFlowRate:** Returns current flow rate infused by pump.

SignalServer: Validates the blood pressure signal and calculates the MAP, systolic and diastolic pressure, etc. Also performs calibration for a new configuration.

**Attributes:**
- **quality:** Validity of the measurement.
- **signal:** 50 Hz sample of the arterial pressure signal during 5 seconds (i.e. 250 samples).

**Services:**
- **calibrate:** Determines transformation-parameters in order to let the values of the signal equal clinical values.
- **characteristics:** Returns 7 characteristic values of the signal, i.e. mean, maximum, minimum, systolic-slope, pulse period, distance_up and distance_down.
PatientModel: Calculates a new pressure according to the past pressure values and infusion flow rates using a simple first order patient model. It is used for simulation purposes if no real pressure signal is accessible.

Attribute: - simPressure: Contains during simulation the calculated MAP dependent on the previous MAP and flow, instead of real pressure.

Archive: Performs file handling and storage of data.

Attributes: - fileType: Filename and extension.
- medium: Device and path.

Services: - fromFile: Read information from a file.
- mergeFiles: Merge data from two files into a new file.
- toFile: Write data to file.

Analysis: Off-line examination of the data.

Attributes: - statistics: Determines statistical information from data.

Services: - avrDistanceSetp: Returns average distance to setpoint for certain period.
- avrFlow: Returns average flow during certain period.
- avrPressure: Returns average MAP during certain period.
- distanceStd: Returns standard deviation of distance to setpoint.
- flowStd: Returns standard deviation of flow.
- pressureStd: Returns standard deviation of MAP.
- replay: Request to replay data from a file on the screen.

Besides defining objects, their attributes and services, a flow-diagram showing the various states the controller passes through, is a good way to analyze the problem and to prevent making design errors.

4.2. Flow Diagram

In order to be able to adjust or extend the software of the blood pressure controller and the Infusion Toolbox correctly, a good view of the exact states and possible 'routes' the software program can take is necessary. Therefore, a flow diagram of the design is made. In Figure 4.3 the flow-diagram for the communication of the SNP-controller is shown. For an explanation of the graphical notations used see Appendix B.

From the startup state, where the initialization is done, the system proceeds to the display medical menu state. There a request to the DisplayServer is made to display the medical menu and send back the desired parameters if acknowledged by the anaesthetist. At the moment the parameters are the weight of the patient, the SNP-concentration and the operating room number. In the future other environments could be indicated, e.g. ICU, and the name of the patient for evaluation purposes. The medical parameters are evaluated. When outside a given range or when a communication error has occurred an error-message is sent according to the type of error and finally the system returns to the display medical menu state. If the medical parameters are OK, the next state will be main menu. A request is made to the DisplayServer to show the main
menu and send back the desired command. This command can be either one of the five shown in Figure 4.3 or an error. In the following sections the various actions for each command will be described.

![Flow diagram of the program for integration of the BP-Controller and the Infusion Toolbox.](image)

First the most simple and shortest actions are reviewed. In case of an erroneous command a message will be sent. This message is dependent on the kind of error (wrong command or communication error). Next the system returns to the main menu state. In the case of the exit (5) command the END state is entered in which all files are closed and the system is shut down. If the user wants to change the medical parameters (4) a return to the display medical menu state is made.

The calibration (3) command makes the system switch to the state calibration instruction in which a request to the DisplayServer is made to show an instruction of how to perform the calibration on the screen and return a start command in case the anaesthetist indicates everything
to be ready. If a communication error appears here a message will be sent to the DisplayServer and a return to the main menu state is made. When the command arrives uncorrupted the **calibrate** routine from the SignalServer is started. If no problem occurs during this calibration the main menu state is entered again. If for some reason the calibrate routine could not determine a correct set of transformation parameters the **display error** state is entered. This requests the DisplayServer to show a message and ask the user what the next step should be. When the **answer** is to stop the calibration we return to the main menu. Normally the user will want to try calibration again and we return to the display instruction state. A communication error could occur at this moment in which case an error message will be sent to the DisplayServer, after which the system will return to the main menu anyway.

Before and after calibration the anaesthetist has to **look at the pressure** (2). After this command the system enters the **display pressure** state. Here a request to the DisplayServer is made to show the complete (50 Hz sampled) blood pressure signal in a trend display, updated every second, until the user wants to stop looking at the pressure or a communication error occurs. This will trigger the system to go back to the main menu state.

The last command from the main menu to discuss now is the **start controller** (1) command. This starts up the real control loop of the system. The first state **process** is just an intermediate state from where the control loop is re-entered in the flow diagram. From there a **get signal** routine is started which requests the characteristics and the validity of the signal from the SignalServer. With the results obtained we go into the **displaySNPstate**. In this state the DisplayServer is requested to show the last relevant information (e.g. MAP, setpoint, flow rate, mode, etc.) available to the SNP_Controller. During the first pass this is only the pressure just obtained and all the initial values, like manual mode, zero flow and setpoint 80 mmHg, which are set in STARTUP. The displaySNPstate is followed by a start of the **Inference process**. This is the ExpertSystem inferencing all the information available and supervising the PI_Controller in order to obtain the next flow to be infused and what settings to change or events to trigger. When the goals are reached and the new values are stored a **transport check** is performed. In fact we do not explicitly check if the communication facility is still prohibited, this is done at a lower level. The transport service, an INFU like DOS driver, returns a disconnection state and a communication error message. In that case we enter the **communication error** state which starts an **alarm** routine. The alarm routine has to notify the user and set the event before returning to the process state in order to let the expert system infer with this information and decide what the appropriate actions will be.

When the communication check succeeds the next state will be **controlPump**. In this state the PumpServer will be requested for the status of the pump, the current flow infused and to change the flow, dependent on the result of the Inference process. The controlPump state is followed by a **readEvents** routine that retrieves all the events coming from the DisplayServer. These events are requests from the user to change the settings of the controller, e.g. go to automatic mode, change setpoint, increase flow or surgery from Bypass, etc. Unless there is a communication error or the event is to quit control the **process** state will be re-entered with all this information and the whole process starts again. Getting the new signal parameters and displaying the new information, etc.

If reading the events or controlling the pump is not possible due to a communication error the communication error state will be entered and the alarm routine is run before processing again. Finally of course the event could be a request to quit control. Then we enter the **display**
evaluation state which requests the DisplayServer to display all the evaluation questions and return the answers the anaesthetist gives, after which we return to the main menu again.

This completes the analysis of the global system and general entities performing the various tasks. Because we want to change the controller in such a way that we can use some of the entities of the Infusion Toolbox, namely the DisplayServer and PumpServer, we need to design a communication link between the two systems.

4.3. The Communication Model

A general analysis of a communication design is made. To keep the design compatible with a wide range of communication device designs it will be described according to the Open Systems Interconnection (OSI) Reference Model for computer networks. Not all of the 7 layers of the model will be necessary in the communication needed at the moment but by keeping the design modular it will be easy to add the layers in a later stage when necessary. Therefore all the layers will be mentioned, to show their place in the design which is shown next.

4.3.1. The ISO/OSI Reference Model

The Open System Interconnection (OSI) reference model was produced by the International Standards Organization (ISO) and is concerned with the exchange of information between application processes. The aim is to enable application processes to cooperate in carrying out a particular (distributed) information processing task irrespective of the computers on which they are running. The overall structure of the model is shown in Figure 4.4.

Figure 4.4: Overall structure of the ISO/OSI reference model [Halsall, 1993].
General description of the 7 layers:

**Physical layer**: performs the transmission of bits on a communication channel.

**Data link layer**: transforms the transmission facility into a channel which is free of transmission errors to the upper layers. Outgoing data frames are built. Begin, end and acknowledgments in incoming frames are recognized here.

**Network layer**: controls the performance of the subnet. Determines the route along which the message is sent from source to destination.

**Transport layer**: performs multiplexing of various messages on one channel and makes connections and disconnections. It is a real end-to-end-layer.

**Session layer**: controls dialogue between different entities, token control and synchronization.

**Presentation layer**: implements functions that are used often by various applications, so finding a general solution is worthwhile. For example coding of data.

**Application layer**: serves as the window between the corresponding application processes that are exchanging information.

Each of these layers can be seen as an object. Every object has to provide various services, called primitives, to the objects (layers) which it directly interacts with. So the next step in the design process is to define the necessary primitives for each object. In the following part these primitives will be described as generally as possible.

**Application layer**

The services in this layer can, for example, be designed according to the architecture described in the "service model for communication systems" and the "domain information model" (CEN/TC251/PT5-021/N95-21 and -23). However because this is a quite recent standard proposal which has not been generally accepted and the SNP-controller application is not built object oriented, we restrict ourselves strictly to the communication primitives of this layer for the moment. These primitives are collected in an object for control of connections.

This is called ACSE (Association Control Service Element) in the OSI-model and the primitives are:

- **A_associate**: requesting and accepting an association.
- **A_release**: releasing an association and accepting the release.
- **A_U_abort**: association abort by user(client) in case of a failure.
- **A_P_abort**: association abort by server.

**Presentation layer**

This layer is at the moment superfluous. Only the absolutely necessary primitives are stated here.

- **P_connect**: create a presentation connection.
- **P_release**: end gracefully.
- **P_U_abort**: abrupt end by user.
- **P_P_abort**: abrupt end by server.

If in the future extra services like coding or encryption have to be implemented more primitives should be added here in order to make those possible.
Session layer

For the moment communication will involve two entities only so token control and synchronization are superfluous. Further the dialogue is very simple. This layer again doesn't have to be implemented, only the simplest primitives are stated.

- **S_connect**: create a session.
- **S_release**: end session gracefully.
- **S_U_abort**: abrupt end by user.
- **S_P_abort**: abrupt end by server.

Transport layer

This is a more extended object in the design. Because there are more primitives to be defined they are divided into several groups to improve readability.

Local management:
- **T_alloc**: allocates transport interface data structures.
- **T_free**: frees structures allocated with **T_alloc**.
- **T_bind**: binds a transport address to a transport endpoint.
- **T_unbind**: unbinds transport address from transport endpoint.
- **T_open**: establishes a transport endpoint connected to provider.
- **T_close**: close a transport endpoint.
- **T_getstate**: returns the state of a transport endpoint.
- **T_look**: returns the current event on a transport endpoint.
- **T_error**: transport interface error message.

Connection establishment:
- **T_connect**: requests a connection with destination.
- **T_accept**: accepts a request for a transport connection.
- **T_listen**: retrieve an indication of a connect request from other transport user.

Connection mode data transfer:
- **T_send**: send the data over an established connection.
- **T_receive**: retrieve the data that has arrived over a connection.

Connection release:
- **T_abort**: aborts a connection or rejects a connection request.
- **T_disconnect**: requests a release of a connection.
- **T_rcvabort**: returns an indication of an aborted connection, including a reason code and user data.
- **T_rcvdiscon**: returns indication that remote user has requested a release of a connection.

Network layer

At this moment our communication will only be point to point as stated before. Therefore routes between and addresses of entities do not have to be specified. Because of the strict timing
required for the SNP-controller application it will also be very difficult to include this application in a larger network where timing cannot be guaranteed. No primitives are defined here.

**Data Link layer**

A lot of work is also done in this layer.

- **DL_wait**: wait for an event and determine event type.
- **DL_totransp**: deliver information of incoming frame to transport layer.
- **DL_fromtransp**: get information from transport layer for transmission on channel.
- **DL_tophysic**: give frame to physical layer for transmission.
- **DL_fromphysic**: get incoming frame from physical layer and copy information.
- **DL_starttimer**: start clock and put time-out event on.
- **DL_stoptimer**: stop clock and put time-out event off.
- **DL_startacktimer**: start extra timer for transmission of separate acknowledge.
- **DL_stopacktimer**: stop extra timer.
- **DL_enabletransp**: allow transport layer ready event.
- **DL_disabletransp**: forbid transport layer ready events.

**Physical layer**

- **F_transmit**: transmit a bit.
- **F_receive**: receive a bit.
- **F_request**: request to send.
- **F_clear**: clear to send.
- **F_data**: data set ready.
- **F_ready**: terminal ready.

So far the abstract description of the OSI model. In the next section the design for communication between the blood pressure controller and the Infusion Toolbox will be shown. It will be noted where each specific layer of this model is represented in the design.

**4.3.2. System Design**

The design for integration of both systems is based upon the communication structure of the Infusion Toolbox as noted in chapter 3. The serial port, which was used to control the infusion pump in the stand-alone blood pressure controller, is now used to connect the controller to the serial network of the Infusion Toolbox. The pump will be addressed by the Infusion toolbox which is the server of the controller. In order to exchange the necessary information between both applications the blood pressure controller application is extended with a communication module. This module retrieves data from or sends data to the Toolbox application via the INFU driver and Universal Device Communication controller (UDCC) using the special command syntax on which the communication of the Infusion Toolbox network is based. A scheme of the design is shown in Figure 4.4.
Extension of the Infusion Toolbox

In this design we can distinguish the various layers of the OSI reference model. The data network is formed by the serial network with a star topology around the UDCC. The physical layer is formed by the ttya-driver. The INFU driver is the implementation of the data link layer and the transport layer. At the moment we cannot speak of a network layer, which normally is included between these two layers, because there is only point-to-point communication between two computers in this design. Also the session and presentation layer are not implemented yet because they are superfluous at the moment. They can however be added between the transport layer and the application layer in the future if necessary. The application layer is formed by the communication module in both the blood pressure controller and the Infusion Toolbox application software.

To accomplish communication using this structure and to ensure safety the INFU driver had to be adjusted. An object oriented presentation of the transport model of the INFU driver and some aspects of the protocol are shown in the next sections. The design setup for integration of both systems according to this structure is chosen because its modularity. The Infusion Toolbox controls the information exchange between the various devices. The blood pressure controller can now be seen as a black box giving information about the blood pressure and the amount of nitroprusside which should be infused in order to regulate the pressure safe and stable towards a specified target level. Development of the Infusion Toolbox through extension with various 'black boxes' according to this structure can result in a central terminal for anaesthesia. This way the central computer doesn't become overloaded with all sorts of programs requiring CPU time. Several applications which are more or less sovereign can be added in the future. The limiting
factor will then most probably be the communication network which cannot transfer all the command and display information fast enough. Upgrading the communication network by for example usage of an internal bus structure directly coupled to a communication processor is probably the next step then. This can only be accomplished if all applications and devices are equipped with an equal communication interface which now is formed by the INFU driver and its command syntax.

4.3.3. OO-Representation of the Transport Service

In this section an object oriented representation of three lower layers in the OSI model, which form the bases of the design is shown, see Figure 4.6. In this representation the transport layer is implemented by an object TransportService which is responsible for sending and receiving messages and forms the interface between the application layer and the data link layer (DataLayer). The application can for example request the state of the transport service (TS).

Figure 4.6: OO-representation of the transport service implemented in the INFU driver.
The state of the TS can be ready, i.e. ready to send and receive a message, or listening, which is waiting for a connection to a remote TS. Also the type of the TS can be noted. This determines the format of the messages read or written by the application layer. The messages can either be ASCII text ended by a newline symbol (type 1) or binary messages (type 2).

By using an Emitter and a Receiver the DataLayer converts the messages into frames which are sent to the PhysicalLayer, or vice versa in case frames are received from the PhysicalLayer. The DataLayer is also responsible for recovering transmission errors. The structure of the messages and the frames, already shown in the figure, will be discussed in the next section together with the communication protocol. The PhysicalLayer then transmits these frames byte by byte.

The Emitter sends the frames which must be acknowledged. It contains, for example, the frame to send, the sequence number of the last frame successfully sent, emitter states (ready or busy) and a queue of frames which couldn't be sent yet. The Receiver verifies whether a received frame is not corrupted and keeps track of the last data frame successfully received in order to perform correct acknowledgement. Further the state of the receiver (waiting or reading) and the sequence number of the Ack-frame are noted by this object.

Finally there is an object TimeScheduler which uses the system clock interrupt to schedule certain events, e.g. whether the acknowledge time limit or the polling time limit passed by. This is necessary in the used protocol to ensure safety by detecting loss of data or a broken connection.

4.4. The Communication Protocol

The communication between the blood pressure controller and the Infusion Toolbox uses a protocol based upon acknowledgement and re-transmission of data. Also a Cyclic Redundancy Checksum (CRC) is used to detect errors in the transmitted frames. Polling ensures that a disconnection of the communication link between the two entities is detected and appropriate actions can be taken. These features of the protocol will be discussed more extensively in this section together with the error handling which is dependent on the moment the connection, due to some reason, does not function well enough or is lost completely.

4.4.1. Implemented Features

The messages used by the transport service consist of a head part, containing the size of the message, and a tail part which contains the contents of the message. At the moment the maximum length of a message is limited to 256 characters which is conform the maximum length of strings used in Pascal. According to their size these messages are split into a number of frames by the DataLayer. A frame consists of three parts: a head, the data and a CRC. The head contains the kind of frame, the sequence number given to the frame by its sender and the total length of the frame (head and CRC included). There are four kind of frames: Data-, EoDt-, Polling- and Ack-frames.

The Data-frame is the normal numbered frame containing the data. The EoDt-frame is a numbered frame containing the last bytes of a message (End-Of-Data). A Polling-frame is an unnumbered control frame used to test if a remote DataLayer is active. Finally the Ack-frame is
a numbered control frame again which denotes that a Data-frame has been successfully received (acknowledgement).

Globally the communication protocol has the following concept. The Application can request the status of the transport service (TS), reset the TS if necessary and read messages from or copy messages to the TS. A number of messages can be kept in the two different buffers of the TS, so the application does not have to wait for an answer. The transport service handles the transfer of the data autonomously controlled by interrupts. This is very useful for the blood pressure controller application because it can now send its messages at any time and only read all the incoming messages from the Infusion Toolbox once every 5 seconds, conform the 5-seconds-period. At the moment the sizes of the send and receive buffer are limited to 4 Kb each, which is enough to prevent any overload, during the 5 seconds. The Infusion Toolbox application seems to be more then fast enough and reads a message from the buffer almost immediately. This mechanism of course has to be tested more extensively for absolute safety and in case of problems the buffers can easily be extended.

As long as the TS is ready and in the output buffer are files to be sent, the DataLayer will split these messages into frames and transfer them via the physical layer to the remote TS. If a frame is received correctly it will be acknowledged. In case a frame is not acknowledged within a certain time the frame will be sent again. This re-transmission is performed three times, after which an erroneous connection is detected and reported to the application by the TS. The communication is full duplex, i.e. it can send and receive at the same time, with a signaling rate of 9600 Baud. During the initial tests 3 re-transmissions seemed to be enough at this baud rate. If, due to external influences, the transmissions contain too many errors the number of re-transmissions can be increased or the baud rate decreased.

Another safety mechanism in the protocol is Polling. When the transport service is ready but there are no messages to be sent or received for a certain time period, the TS will automatically send a Polling-frame in order to check if there still is a connection. If there is no answer from the remote TS to this Polling-frame the connection will be noted as broken and the status of the TS will become disconnected. The application using the TS can then take appropriate actions. The blood pressure controller, for example, knows the last acknowledged flow rate of nitroprusside and has to decide whether this is safe at the moment considering the current pressure and setpoint. It can try to reset the TS and establish connection again; when this is successful within the safe control period, control can be continued. If the connection cannot be re-established within a certain time period the Infusion Toolbox application will generate an alarm and request manual control by the anaesthetist. After this time period the blood pressure controller also assumes manual control.

Due to all kind of disturbances, e.g. electrocautery, the transmitted frame can contain errors. It can be a single random error but mostly the errors come in groups (bursts). Of course such errors need to be detected. The common method for detection of error bursts is based on the use of polynomial codes. A single set of check digits is generated (computed) for each transmitted frame, based on the contents of the frame, and appended by the transmitter to the tail of the frame. The receiver then performs a similar computation on the complete frame and check digits. If no errors have been induced, a known result should always be obtained; if a different answer is found, this indicates an error. The exact binary calculations will not be shown here. They can be found in any book on data communication, for example [Halsall, 1993]. At the moment the mechanism is implemented and the space reserved in the frame is used for transmission of the number computed in the DataLayer of the INFU-driver software. The type of errors that are
detected is determined by the generator polynomial. The standard CRC-16 or CRC-CCITT polynomial is probably sufficient.

In case of error detection in a frame by the receiver no acknowledge will be sent for the frame and the re-transmission mechanism acts after time out at the emitter. Too many error detections can again lead to disconnection of the TS. The loss of the connection with the other application can in fact occur at any stage of the process as shown in the flow diagram (Figure 4.3, § 4.2). The appropriate reaction of the application is not necessarily the same at each stage, therefore a number of stages have been defined which are known to both applications, Toolbox and BP-controller. This error handling is described in the next section.

4.4.2. Error Handling

To make sure each application knows exactly at what stage a communication problem occurred and what action the other application expects, the program states shown in Figure 4.7 have been defined.

![Program State Diagram](Figure 4.7: program state diagram for communication protocol between Toolbox and Controller.)
50 Extension of the Infusion Toolbox

A transfer to another state is determined by the blood pressure controller. If a switch is made, triggered by a certain command, the new state number is transmitted to the Infusion Toolbox application. A request to go the next state often comes from the Toolbox. When a disconnection occurs, new information exchange starts from the last successfully finished state after the connection is re-established. The various stages will be shortly described next.

At start up both applications are in state 0; after the connection is established and confirmed the systems are ready and a switch to state 1 is made. In this state the medical data, e.g. patient weight and SNP concentration, have to be entered at the user-interface of the Infusion toolbox. These are transferred to the blood pressure controller when the anaesthetist confirms them. Then state 2 can be entered in which the controller is waiting for a request for the next action. Normally usage of the system will be continued and medical data is correct. So a switch to state 3 (setup) is made. From here the anaesthetist can decide to view the raw pressure signal or calibrate the A/D-conversion. In both cases the applications are noted to be at state 4. At the end of these actions always a return to state 3 occurs. The end of the setup actually ends the initialization phase of the controller. A disconnection during these stages is not dangerous, because there is no patient control by the system yet.

The applications are in state 5 now. This is the main menu from which all options can be chosen. The difference with state 2 is the possibility to start closed loop control of the MAP from state 5. Also signal setup or change of the medical data can be performed again. In the latter case a state 6 is entered which only differs from state 1 by the fact that a return to state 5, instead of state 2, is made if the action is finished. The main objective is of course to start closed loop control (state 7). A disconnection of the transport service in this state requires a different approach. The expert system of the controller needs to decide then whether the system can stay in automatic mode assuming the last acknowledged flow rate or has to go to manual mode waiting for a connection and new information from the Infusion Toolbox. The TS needs to be reset in order to try and connect again. In case of disconnection in state 7 the appropriate actions need to be taken by the Infusion Toolbox application. An audible alarm needs to notify the anaesthetist and a switch of the ControlPanel to manual control needs to be made.

These actions are quite similar to the actions of the controller in case of invalid pressure measurements. As soon as the connection is established again, control can be continued by going to automatic mode again. Ending closed loop control is automatically followed by an evaluation questionnaire after which the systems return to state 5. From there a new control session can be started or the application can be shut down.

This completes the description of the program states. A disconnection of the transport service should never occur though. However, to ensure the absolute safety which is obliged by medical systems this error handling needs to be exactly defined.

4.5. Other Aspects of the Design

To end this chapter about the design for integration of the Toolbox and the controller application some aspects which have not been noted will be reviewed. The user-interface had to be re-constructed adapting to the windows oriented environment of Smalltalk which the Infusion Toolbox uses. Remote control from another PC also gives some extra features, some already present in the Toolbox application, which can be used off-line.
4.5.1. The New User-Interface

As stated in chapter 3 the Infusion Toolbox is implemented in Smalltalk and therefore uses the Smalltalk environment to display all information and interact with the user. This environment is based on graphical windows and some special elements, so called Wintools. The anaesthetists in Brussels are used to this user-interface. In order to control the blood pressure controller remotely a user-interface needed to be created in the Smalltalk environment too. A ControlPanel object, similar to the ones to control the various infusion devices, is created based on the same elements but showing all the information shown by the user-interface of the stand-alone application.

Figure 4.8: Example of the initial user-interface of the blood pressure controller in the Smalltalk environment; main window at the left and the ControlPanel at the right.

The new user-interface consists of mainly two windows shown in Figure 4.8. Primary, the main window used to setup up the controller, e.g. choose pump, drug or calibrate A/D-conversion, and to start closed loop control. Secondary, a ControlPanel window, which should be permanently visible during control, showing the command buttons and the numerical data, and buttons to open windows displaying the trend of the MAP and the SNP flow. Extra windows will be opened for entering medical data, perform the calibration, view the raw pressure signal or add events and comments to the archive file.
4.5.2. Extra Facilities

The presented design creates some extra facilities due to the fact that the CPU of the central PC, which remotely controls the infusion devices and the blood pressure controller, is not permanently occupied by these processes. Due to this the anaesthetist can perform some off-line tasks. First he can review the MAP and SNP flow of the total operation at any time. The stand-alone controller only shows the last 30 minutes. Secondly, events and comments can be noted off-line in the archive file which is very useful for evaluation purposes.

Due to the fact that the Infusion Toolbox can control up to 12 commercially available pumps, the blood pressure controller can address them too now. Some devices have a different accuracy at a different range. For example steps of 0.1 ml/hr below 10 ml/hr and steps of 1 ml/hr above. This needs to be taken into account by the controller in order to determine the exact flow rate and rest to be used in the next 5-seconds period. Also the MAP is not available anymore as a vital sign for the anaesthetist if the controller works properly. To restore this, extra data processing algorithms could be implemented to extract more characteristics of the MAP and flow.

4.6. Conclusions

A design for integration of the Infusion Toolbox is presented which is based on the idea of building a modular system for central control and monitoring of all devices and functions during anaesthesia. The blood pressure controller becomes a black box in this system providing information about the blood pressure and the necessary flow rate of nitroprusside to be infused in order to regulate the MAP towards a specified setpoint. The design is based upon the communication structure of the Infusion toolbox. Therefore a communication protocol and various safety mechanisms have been defined and implemented. A first version of the new controller has been built together with its corresponding ControlPanel in the Infusion Toolbox. This version is not completely stable yet though. Extensive tests have to be performed and improvement or generation of some rules might be necessary.

4.7. Future Developments and Tests

The integrated system as a whole has not been tested. The communication between the two applications was tested and was found to satisfy the goal. The transport service can be controlled and connection can be established. Information can be exchanged through transferring messages and both applications keep track of the state of the program. Of course very extensive simulations have to be performed to detect any small overseen bugs and ensure safety of the system. Simulations have to be performed in real time and not in the current simulation time. The controller shortens the '5-second period' during simulations because conversion and validation of the blood pressure is replaced by data from a file. To many messages might overload the communication network buffers then. So simulations need to run in real time as well.

There is one part of the design which has to be implemented still. Extra rules have to be added to the rulebase of the expert system, similar to the rules for invalid pressure measurements, to handle disconnection of the transport service during closed loop control.
Chapter 5

Clinical Evaluation of the stand-alone Blood Pressure Controller

In this chapter the results of the clinical evaluation will be presented and discussed. First the conditions under which the tests have been performed, and some medical information will be described. Secondly, the results obtained from a thorough statistical analysis of the performance of the blood pressure controller are shown. Next typical examples of the behaviour during various stages of the operation or behaviour in a specific situation are displayed. Before drawing conclusions the general opinion of the anaesthetists, resulting from a small questionnaire after each test, is stated.

5.1. Test Environment and Trial Setup

The clinical tests were performed at the academic hospital: ‘Hôpital Erasme’, which is associated with Université Libre de Bruxelles. During these tests the system setup was similar to the initial tests performed by van Geene [Geene, 1993]. According to the system setup described in chapter 2, a Compaq AT386-20 PC with an EGA monochrome plasma display, an IMED 929 infusion pump and a LabMaster card were placed on a trolley. This trolley was placed next to the monitoring equipment, from which the analog blood pressure signal was retrieved. Since the initial tests the Hewlett Packard monitoring equipment has changed to an equivalent monitoring system from Datex though. Besides a slight change in the calibration routine this did not influence the basic system.

Sodium nitroprusside (SNP) was infused through a separate infusion line connected to a catheter. The concentration of the infusion fluid was equal in all cases, namely 0.2 mg/ml. The IMED 929 infusion pump has a resolution of 1 ml/hr, which results in an accuracy of 0.04 μg/kg/min for a person of 80 kilograms.

Cardiac surgery, during which the controller was tested, is a very complex environment for controlling blood pressure. Besides the various artefacts, which for example can occur due to interaction of the surgeon or influence of other drugs, an operation consists of several stages. After all necessary preparations are made and the patient is under complete narcosis, the surgeon starts with the skin incision and opening of the chest in order to gain access to the heart. This is the first stage which can cause a severe hypertensive period just after the skin incision.
During the second stage the heart is not operating. Respiration and circulation are maintained by the heart-lung machine, also called perfusion machine. In this phase there is no heart cycle and therefore pressure validation needs to be switched to perfusion mode, otherwise the signal will be marked invalid most of the time. For obvious reasons this phase is often denoted by cardiopulmonary bypass or shortly bypass. This should, however, not be confused with what often is the cause for cardiac surgery, namely creating one or more bypasses around the heart in order to prevent a lack of oxygen at certain parts of the muscle. During perfusion (bypass) the heart is first cooled down which stops the periodical contraction so the surgeon can perform his work. After re-warming and aortic declamping the heart is reactivated again, possibly with the help of some external triggering with an electrical pulse. This ends the bypass stage during which the blood pressure is quite stable, except at the start and around declamping.

The third and final stage of the operation is the closure of the chest and preparation of the patient for recovery and intensive care. Before closure of the chest this phase often shows severe hypotension which the controller can not counteract. The operation is almost finished and automatic blood pressure control is stopped in most cases. Sometimes the controller was used to regulate a part of the, often occurring, post-operative hypertension before the patient was moved to the recovery or intensive care unit.

After the system had been connected and the calibration of the parameters for AD-conversion of the blood pressure signal had been performed, automatic control was in most cases started before skin incision. In general the anaesthetists used the following setpoint values for the MAP: 80, 70 and 80 mmHg during, respectively, the pre-, during- and post-bypass phase. Of course deviation of these values took place if necessary for a specific patient. During all tests, except one, the closed loop blood pressure controller was used together with computer controlled infusion of the drugs Midazolam (a benzodiazepine) and Sufentanil (a morphinomimetic). The Infusion Toolbox was used in order to achieve a plasma concentration of 100 ng/ml for midazolam and 2 ng/ml for sufentanil. Muscle relaxation was achieved by pancuronium, with no pharmacokinetic profile. No inhalation anaesthetics were used.

Three rare patients received a bolus injection of droperidol, because the blood pressure increased rapidly and was not controlled fast enough by the SNP infusion at the start of bypass. In a few exceptional patient schedules, for fast-track recovery and early detubation (same day), one of the anaesthetists used propofol in replacement of midazolam. Propofol is a hypnotic drug with a short half-cycle which gives a very rapid recovery. However none of the patients in the test series used for the evaluation of the SNP controller received propofol. Further the procedure included ventilation of the patients with air and oxygen, a bypass with a membrane oxygenator and a flow of 2.4 L/min/m². Also a moderate hypothermia of 30 °C was achieved.

After simulations and a number of open loop tests did show the system to be safe, a trial of 34 tests was performed. During these 34 tests the controller was not changed. In the first 9 tests the control system was under supervision of the anaesthetist and the author of this report, but after this introduction period the anaesthetists were able to perform the tests autonomous using the short instruction manual shown in Appendix C. More explicit information about the tests is shown in the next section.
5.2. Patients Data and Surgery Type

Some information about the patients and the surgery is shown in Table 5.1. The tests were performed in the period from the 27th of July to the 10th of October 1995.

<table>
<thead>
<tr>
<th>test no.</th>
<th>weight [kg]</th>
<th>gender [male/female]</th>
<th>surgery type</th>
<th>total time [minutes]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>64</td>
<td>m</td>
<td>bypass</td>
<td>165</td>
</tr>
<tr>
<td>2</td>
<td>89</td>
<td>m</td>
<td>bypass</td>
<td>202</td>
</tr>
<tr>
<td>3</td>
<td>93</td>
<td>m</td>
<td>aortic valve replacement</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>83</td>
<td>m</td>
<td>bypass</td>
<td>244</td>
</tr>
<tr>
<td>5</td>
<td>75</td>
<td>f</td>
<td>mitral valve replacement</td>
<td>154</td>
</tr>
<tr>
<td>6</td>
<td>94</td>
<td>m</td>
<td>bypass</td>
<td>328</td>
</tr>
<tr>
<td>7</td>
<td>70</td>
<td>m</td>
<td>bypass</td>
<td>272</td>
</tr>
<tr>
<td>10</td>
<td>70</td>
<td>f</td>
<td>bypass</td>
<td>222</td>
</tr>
<tr>
<td>11</td>
<td>84</td>
<td>m</td>
<td>bypass</td>
<td>225</td>
</tr>
<tr>
<td>12</td>
<td>51</td>
<td>f</td>
<td>bypass, mitral valve and tricuspid</td>
<td>246</td>
</tr>
<tr>
<td>14</td>
<td>55</td>
<td>f</td>
<td>mitral valve replacement</td>
<td>168</td>
</tr>
<tr>
<td>16</td>
<td>53</td>
<td>f</td>
<td>mitral valve replacement</td>
<td>179</td>
</tr>
<tr>
<td>17</td>
<td>76</td>
<td>m</td>
<td>bypass</td>
<td>336</td>
</tr>
<tr>
<td>18</td>
<td>86</td>
<td>f</td>
<td>aortic valve replacement</td>
<td>191</td>
</tr>
<tr>
<td>19</td>
<td>75</td>
<td>m</td>
<td>heart transplant</td>
<td>327</td>
</tr>
<tr>
<td>20</td>
<td>85</td>
<td>m</td>
<td>bypass</td>
<td>246</td>
</tr>
<tr>
<td>21</td>
<td>73</td>
<td>m</td>
<td>bypass</td>
<td>215</td>
</tr>
<tr>
<td>22</td>
<td>61</td>
<td>m</td>
<td>bypass</td>
<td>259</td>
</tr>
<tr>
<td>23</td>
<td>79</td>
<td>m</td>
<td>bypass</td>
<td>235</td>
</tr>
<tr>
<td>24</td>
<td>60</td>
<td>m</td>
<td>aortic valve replacement</td>
<td>216</td>
</tr>
<tr>
<td>26</td>
<td>61</td>
<td>m</td>
<td>mitral valve replacement</td>
<td>192</td>
</tr>
<tr>
<td>27</td>
<td>75</td>
<td>m</td>
<td>aortic valve replacement</td>
<td>198</td>
</tr>
<tr>
<td>28</td>
<td>79</td>
<td>m</td>
<td>bypass</td>
<td>278</td>
</tr>
<tr>
<td>29</td>
<td>74</td>
<td>m</td>
<td>mitral valve replacement</td>
<td>166</td>
</tr>
<tr>
<td>30</td>
<td>82</td>
<td>m</td>
<td>bypass</td>
<td>217</td>
</tr>
<tr>
<td>31</td>
<td>65</td>
<td>m</td>
<td>bypass</td>
<td>311</td>
</tr>
<tr>
<td>32</td>
<td>72</td>
<td>m</td>
<td>bypass and aortic valve repl.</td>
<td>261</td>
</tr>
<tr>
<td>34</td>
<td>70</td>
<td>m</td>
<td>bypass</td>
<td>135</td>
</tr>
<tr>
<td>35</td>
<td>70</td>
<td>m</td>
<td>bypass</td>
<td>261</td>
</tr>
<tr>
<td>36</td>
<td>71</td>
<td>m</td>
<td>bypass and aortic valve repl.</td>
<td>272</td>
</tr>
<tr>
<td>37</td>
<td>83</td>
<td>m</td>
<td>(redo) bypass</td>
<td>304</td>
</tr>
<tr>
<td>38</td>
<td>66</td>
<td>m</td>
<td>mitral valve replacement</td>
<td>263</td>
</tr>
<tr>
<td>39</td>
<td>80</td>
<td>f</td>
<td>bypass</td>
<td>333</td>
</tr>
<tr>
<td>40</td>
<td>88</td>
<td>m</td>
<td>bypass</td>
<td>257</td>
</tr>
</tbody>
</table>

The test numbers 8, 9, 13, 15, 25 and 33 are not used because due to an erroneous calibration the test had to be restarted and the program automatically selects the next test number. Thus the total test series includes 34 test. However, test 19 is not included in the statistical analysis because the system was only used a single time during the special case of a heart transplant. Therefore 33 tests are used for clinical evaluation.
Although four anaesthetists have used the system most tests were performed under supervision of Dr. L. Barvais (21) and Dr. D. Schmartz (10). Most patients in the test series were male (27 out of 34). The average weight of the patients was 74 kg (51 to 94 kg). The surgery type was mainly coronary artery bypass (20) and either mitral (6) or aortic (4) valve replacement. On the average a total operation lasted 236 minutes. During the tests the controller was in automatic mode on average 90.6% of the total time and thus 9.4% in manual mode with a standard deviation of 9.6%. The period of automatic and manual control for each test is shown in Table A.7 of Appendix A.

A large part of the manual control period is formed by a period at the start of a surgery when the patient is being prepared and a period at the end of the post-bypass phase when the system is stand-by in manual mode, with zero flow, waiting until the anaesthetist has the time to shut the system off. Further the system sometimes switches automatically back to manual mode due to a too long period of invalid measurements of the pressure caused by blood sampling performed by the anaesthetist. Manual control periods due to intervention of the anaesthetist were very rare and limited to a short period if necessary. The anaesthetists necessarily switched back to manual four times, three times because of to slow control at the start of bypass, due to a too low gain, and once because of an error caused by an extreme situation, which is shown in section 5.4.3. This already indicates safe behaviour of the controller in almost all situations during the tests.

The performance of the controller will be examined more extensively in the next sections through a statistical analysis of the stored data and examples of the behaviour in normal and specific situations. Before showing the results it should be noted that the effect of the SNP infusion is not always obviously distinguishable from other influences, e.g. infusion of other drugs or surgical actions. The numeric results always have to be interpreted bearing this in mind.

5.3. Statistical Analysis

5.3.1. Average Values

As a measure of the performance of the controller the average MAP, the average distance to the setpoint and the average flow rate of each test is computed. All results of these computations can be found in Table A.1 of Appendix A. In Table 5.2 below only the minimum and maximum average and the average over all tests of these averages is shown.

<table>
<thead>
<tr>
<th>parameter</th>
<th>total run avr. (std.)</th>
<th>auto control avr. (std)</th>
<th>min.</th>
<th>max.</th>
<th>std. auto control avr. (std.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP [mmHg]</td>
<td>74.1 (4.0)</td>
<td>74.3 (4.3)</td>
<td>66.4</td>
<td>84.5</td>
<td>7.2 (2.4)</td>
</tr>
<tr>
<td></td>
<td>MAP - setpoint</td>
<td>[mmHg]</td>
<td>8.3 (3.3)</td>
<td>8.1 (3.2)</td>
<td>3.0</td>
</tr>
<tr>
<td>flow rate [mg/hr]</td>
<td>3.1 (2.8)</td>
<td>3.4 (2.9)</td>
<td>0.0</td>
<td>11.1</td>
<td>4.5 (3.4)</td>
</tr>
</tbody>
</table>

The average MAP was 74 mmHg which is according to the expectations if we take into account the general setpoint settings of 80, 70 and 80 mmHg during the various stages of the operation. There is no significant difference between the values of the total test and automatic control due to the fact that manual control periods were very short. A more interesting and commonly used
figure to judge controller performance is the distance to the setpoint, i.e. $I \text{MAP} - \text{setpoint} I$. On average the distance was 8 mmHg with a minimum of 3 mmHg and a maximum of 17 mmHg. The latter occurred during an operation of a patient with a low blood pressure, in which case infusion of nitroprusside was barely necessary. This is also related to the minimum average flow rate of 0 mg/hr noted. Elimination of the influence of hypotensive patients is correct because the controller cannot counteract a pressure beneath the setpoint. Therefore, the averages during effective control, which is defined as control at the time the flow of SNP is larger than zero, are computed also and the main results are shown in Table 5.3.

<table>
<thead>
<tr>
<th>parameter</th>
<th>auto control</th>
<th>eff. control</th>
<th>min.</th>
<th>max.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{MAP} [\text{mmHg}]$</td>
<td>74.3 (4.3)</td>
<td>78.2 (4.2)</td>
<td>71.0</td>
<td>85.7</td>
</tr>
<tr>
<td>$</td>
<td>\text{MAP} - \text{setpoint} I</td>
<td>[\text{mmHg}]$</td>
<td>8.1 (3.2)</td>
<td>4.5 (1.6)</td>
</tr>
<tr>
<td>flow rate [mg/hr]</td>
<td>3.4 (2.9)</td>
<td>4.8 (3.7)</td>
<td>0.1</td>
<td>13.6</td>
</tr>
</tbody>
</table>

There is a significant difference with the values for automatic control as can be seen. On average the average MAP and flow rate are a little higher which can be explained by the obvious correlation between the need of SNP infusion and a pressure above or close to the setpoint. Very meaningful, however, is the decrease of the average distance to the setpoint to an average value of 4.5 mmHg with a small standard deviation over all tests. This clearly shows that the controller in general is able to suppress hypertension effectively and keep the MAP close to the desired target level.

As already noted in the first section of this chapter cardiac surgery consists of three phases. To examine the performance of the controller during each of these stages the same figures also have been computed for, respectively, the pre-, during- and post-bypass phase. In Table 5.4 and Table 5.5 the main results for automatic control are given, the results for each individual test can be found in Table A.2 of Appendix A.

| surgery phase | average MAP [mmHg] | average $| \text{MAP} - \text{setpoint} I | [\text{mmHg}]$ |
|---------------|---------------------|---------------------|
| pre-bypass    | 76.9 (7.1) 59.4 90.7 | 9.6 (7.3) 2.5 23.7 |
| during bypass | 72.4 (4.6) 65.0 83.2 | 7.0 (2.7) 2.7 14.5 |
| post-bypass   | 64.7 (21.6) 51.9 81.3 | 10.6 (8.4) 2.7 37.0 |

There are clear differences between the various stages which can be explained by the typical characteristics of each phase. Before bypass the setpoint is at 80 mmHg and the blood pressure is not really influenced yet and often varies below the setpoint. So the distance to the setpoint is a little larger and there is a limited amount of SNP infused on average, mainly due to the hypertensive period caused by the skin incision. During the bypass phase the setpoint is lowered to 70 mmHg to induce hypotension. The controller is active and the MAP is decreased and kept at a closer distance to the setpoint due to effective control. In order to accomplish this a significantly larger flow of SNP needs to be infused. The standard deviation between the various
Clinical Evaluation

tests of the average MAP and average distance is also significantly lower during this phase which shows that the MAP is more stable. Finally during the post-bypass phase initially hypotension causes the average MAP to decrease, while the setpoint is raised again. Therefore the average distance to the setpoint is larger compared to the other phases. However this is strongly dependent on the patient as can be seen by the large standard deviation of the average MAP during the various tests.

Table 5.5: Average, minimum and maximum of average flow rate during automatic control taken over all 33 tests for each of the three surgery phases.

<table>
<thead>
<tr>
<th>surgery phase</th>
<th>average flow rate [mg/hr]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>average (std.)</td>
</tr>
<tr>
<td>pre-bypass</td>
<td>0.8 (1.6)</td>
</tr>
<tr>
<td>during bypass</td>
<td>5.4 (4.4)</td>
</tr>
<tr>
<td>post-bypass</td>
<td>1.1 (1.8)</td>
</tr>
</tbody>
</table>

Besides dividing cardiac surgery into the three mentioned phases the bypass phase can be split into two periods: a cooling period at the start of this phase, during which a moderate hypothermia of 30 °C is achieved, and a re-warming period, during which the temperature is increased to normal body-temperature again. To see whether this influences the controller performance or the patient's sensitivity to nitroprusside the various averages have also been computed separately for these two periods. The main results are denoted in Table 5.6.

Table 5.6: Average of the average pressure, the average distance to the setpoint and the average flow rate during the cooling and warming period of the perfusion (bypass) phase. In this case taken over 27 tests, because the start of a period in some tests was not indicated. Again the values surrounded by parentheses denote the standard deviation.

<table>
<thead>
<tr>
<th>Period</th>
<th>average MAP [mmHg]</th>
<th>average</th>
<th>MAP - setpoint</th>
<th>average flow rate [mg/hr]</th>
</tr>
</thead>
<tbody>
<tr>
<td>cooling</td>
<td>74.2 (5.1)</td>
<td>5.2</td>
<td>(2.0)</td>
<td>6.1 (6.8)</td>
</tr>
<tr>
<td>(re-) warming</td>
<td>70.0 (6.5)</td>
<td>8.4</td>
<td>(4.3)</td>
<td>4.7 (4.4)</td>
</tr>
</tbody>
</table>

From these results it can be seen that generally the average MAP during cooling is above the setpoint of 70 mmHg. The average distance to the setpoint however is in general only 5 mmHg with a low standard deviation of 2 mmHg. So the MAP alters slightly above the setpoint and the average flow of SNP is quite high. During the warming period the average MAP is on average equal to the setpoint but fluctuating more. The average distance to the setpoint is 8 mmHg. These fluctuations can be explained by the surgical actions, e.g. aortic declamping, at this stage of the operation. On average the average flow is lower than during cooling but still high.

All this is an indication for decreased sensitivity of the patient during cooling. However whether this is caused by the decreased temperature cannot be concluded. Other factors caused by perfusion could be of influence too, e.g. larger blood volume, larger delay time before the SNP reaches the smaller arteries and different baroreceptor action. Most probably it is a combination of factors. Controller performance is however good and stable although nitroprusside consumption is quite high. All in all the conclusion can be that the MAP in general is within the 10 mmHg bound to the setpoint, which is required for acceptable control. A better view of the performance of the controller is acquired by examination of the percentage of time the blood pressure was within certain boundaries to the setpoint during the tests.
5.3.2. Histograms

The percentage of time the MAP was at a certain distance to the setpoint during each test has been computed. The area around the setpoint, from 30 mmHg below to 30 mmHg above, was divided in 12 regions of 5 mmHg each. An example of histograms displaying the results of these calculations is shown in Figure 5.1.

Figure 5.1: Histograms showing the percentage of time the MAP was in a certain distance region to the setpoint during test number 18. A histogram is shown for: (a) the total operation (auto. & manual control), (b) automatic control, (c) effective control (flow > 0) and (d) control when the pressure was above the setpoint.

The example of Figure 5.1 shows that viewing only the automatic control period does not completely justify the performance of the controller. Long hypotensive periods deteriorate this
view of the performance because they cannot be compensated for by the SNP controller. Therefore also the histograms for effective control, i.e. flow larger than zero and not at its maximum, and for control when the MAP is above the setpoint are shown. These results show very good performance with the MAP for 90% of the time within the 5 mmHg bound to the setpoint. This is however just one test case and the results for other tests are not all as perfect as for this case.

As stated before, these histograms have been computed for all the tests. The numeric results from this can be found in tables A.3 up to A.6 of Appendix A. In order to show the general performance, the average values over the total test series of 33 tests have been determined. The histograms in Figure 5.2 and Figure 5.3 show the results.

![Histograms](image)

Figure 5.2: Histograms showing the percentage of time the MAP was in a certain distance region to the setpoint on average for all 33 tests; (a) during automatic control, (b) during effective control (i.e. flow > 0).

Again the histogram for automatic control also shows the hypotensive periods which occur during the operations. Still, on average 50% of the time the MAP is within the 5 mmHg bound to the setpoint and 71% of the time within the 10 mmHg bound. Judgement of the controller performance using the histogram for effective control is more appropriate however. Then the MAP is on average 71% of the time within the 5 mmHg bound and 89% of the time within the 10 mmHg bound. The latter resembles the results of the initial tests by Geene [Geene, 1993], which gave an average of 90% of the time within the 10 mmHg bound. Performance during
control when the MAP was above the setpoint shows a similar result, an average of 87% of that
time the MAP does not rise more than 10 mmHg above the setpoint.

![Histogram](image)

**Figure 5.3:** Histogram showing the percentage of time the MAP was in a
certain distance region to the setpoint on average for all 33 tests, during the
period of control when the MAP was above the setpoint.

### 5.3.3. Comparison with Previous Tests

A comparison of the results during automatic control of this prototype of the controller with the
results obtained with an earlier prototype used at the Catharina Hospital in Eindhoven [Zwart,
1990] and a prototype of a controller, based on the same structure, using nitroglycerin (NTG)
tested at the Academic Hospital of Maastricht [Hoeksel, 1992] is shown in Table 5.7.

**Table 5.7:** Percentages of the offset over time in zero-centered bands in automatic
control for an old and the new prototype of the SNP-controller and the NTG-controller.

<table>
<thead>
<tr>
<th>offset band</th>
<th>SNP 1990 (Eindhoven)</th>
<th>NTG 1992 (Maastricht)</th>
<th>SNP 1995 (Brussels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>± 5 mmHg</td>
<td>33 %</td>
<td>52 %</td>
<td>50 %</td>
</tr>
<tr>
<td>± 10 mmHg</td>
<td>61 %</td>
<td>79 %</td>
<td>71 %</td>
</tr>
<tr>
<td>± 15 mmHg</td>
<td>78 %</td>
<td>90 %</td>
<td>83 %</td>
</tr>
<tr>
<td>± 20 mmHg</td>
<td>88 %</td>
<td>95 %</td>
<td>90 %</td>
</tr>
</tbody>
</table>

This comparison shows a clear improvement of the new prototype of the SNP-controller
compared to the old prototype. The results are comparable with the results obtained with the
NTG-controller. The results from Brussels are computed over 33 tests which is almost twice as many as the 18 tests used for calculation of the results from Maastricht.

Interpretation of the results shown in this section is very difficult. Judgement of the actual performance of the controller should be based on a comparison with manual control performed by the anaesthetist under similar circumstances. This is to some extent possible by asking the opinion of the anaesthetists about the performance of the controller during each test. The results of this are shown in section 5.6. But the performance of the controller is not only based on the speed and accuracy of bringing the MAP towards a specified target level. A stable blood pressure and safe behaviour of the controller under all circumstances is even more important. Therefore the next section will review the behaviour of the controller during the various surgery stages and during typical events.

5.4. Typical Examples of the Performance

The goal of the designed controller is to act safe during all stages of cardiac surgery. To verify whether this goal is reached, the behaviour of the controller during specific control problems, caused by surgical events, will be examined. Problems can for example be control during severe hyper- or hypotension, large pressure fluctuations or temporary changes in the patient's sensitivity. First an example of a complete operation with good controller performance will be shown and some typical events, which occur during cardiac surgery, will be noted.

5.4.1. Normal Good Behaviour

During an operation typical events occur which show more or less similar response of the MAP. These events can clearly be distinguished in the next example, together with the corresponding reaction of the controller. In all the following figures, showing the real time behaviour of the controller, the upper window denotes the MAP and the setpoint in mmHg while the lower window displays the infused amount of drug in mg/hr.

Figure 5.4: MAP, setpoint and flow during the first part of a surgery (test 18). The following events occurred: skin incision (9:45), sternal spread (9:55), invalid measurements due to blood sampling (10:11), start Bypass and switch of pressure validation to perfusion mode (10:20), decrease of setpoint (10:21).
In Figure 5.4 the first part of surgery is shown. It can be seen that the hypertension caused by skin incision (9:45) is suppressed well by the controller. After sternal spread (9:55) the pressure drops further below the setpoint, which the controller cannot, and in this case should not, counteract. From 10:11 to 10:14 a long period of invalid measurements is shown caused by the anaesthetist taking a blood sample. This period was larger than the safety mechanism permits and thus an automatic switch back to manual control mode was made. As soon as pressure measurements were valid again the anaesthetist switched back to automatic control.

The start of the bypass phase (10:20) always shows a short decrease of the pressure, caused by clamping of the aorta, followed by a hypotensive period. For this sensitive patient the controller performs well. It was already stated before that at the start of bypass the pressure validation has to be put in perfusion mode (no heart cycle) and the setpoint is lowered from 80 to 70 mmHg (10:21).

During the cooling period of the bypass phase the MAP is very stable and the pressure is maintained at the desired hypotensive level with an almost constant flow of SNP, as can be seen in Figure 5.5. Due to warming (11:36) the blood pressure increases which results in the demand of a higher flow rate in order to keep the MAP close to the setpoint (Figure 5.6).
The end of the bypass phase and a part of the post-bypass phase are shown in Figure 5.7. Aortic declamping (11:51) causes a hypotensive period followed by strong fluctuations. In this case of a stable and sensitive patient these fluctuations are not very extreme and controller performance is good. This operation was relatively short and the gain was not increased too much by adaptation.

In a lot of cases, however, the gain has to be adapted upwards much more in order to counteract the temporary decrease of sensitivity and keep the MAP stable around the target level, see Figure 5.8. Although controller performance at this stage is still good, the high gain can cause problems after declamping. Besides gain adaptation, which will be discussed more extensively in later sections of this chapter, the performance of the mechanisms to ensure extra safety will be reviewed first.

Validation of the blood pressure signal showed good performance during the complete test series. No unnecessary change to manual occurred. Long periods of invalid measurements only occurred due to blood sampling by the anaesthetist. If this takes longer than 90 seconds a return to manual cannot be prevented out of precaution. Probably the most difficult for MAP validation is a patient with mitral valve problems. They often have a blood pressure curve which shows large variations between successive heart cycles. Due to this, measurements are marked not valid incorrectly, which is inherent to the used validation algorithm because a comparison between successive heart cycles is made. An example of this is shown in Figure 5.9 from 9:45 to 10:56 with a period
of blood sampling from 9:55 to 9:57. The validation mechanism does not have to be changed though because there are still enough measurements valid to ensure safe control.

Another safety mechanism is detection of transients, i.e. large, unexpected pressure changes. Transient detection is very complex. There is, however, a lot of expertise incorporated in the controller, which makes this mechanism one of the key-properties. Erroneous detections very seldomly occur and their influence on control is limited to a short period of time. Besides the examples of transient detections in Figure 5.9, detection of two up-transients with in between a down transient is shown in Figure 5.11.

5.4.2. Typical Cases Needing Improvement

Oscillation detection is also a safety mechanism incorporated in the controller. Oscillations were detected several times during the test series, mostly during the bypass phase. In Figure 5.10 an example of an oscillation, caused by too high gain and induced by an action of the perfusionist, is shown.

Figure 5.9: MAP, setpoint and flow at the start of surgery, showing a lot of invalid pressure measurements (9:45-10:56) due to the specific mitral valve problems of this patient. Other events are: start bypass (10:06) and detection of down transients (10:06 and 10:09:50).

Figure 5.10: MAP, setpoint and flow during the bypass phase of an operation showing an oscillation detection at 11:45:30.
Clinical Evaluation

The oscillation mostly occurs in the flow and much less in the pressure. The controller performance is safe and the oscillation is detected before the MAP becomes unstable. After detection a gain down adaptation is performed, resulting in a very stable MAP again. So the mechanism works correctly and the resulting action has the desired effect. However, detection is rather slow and could be faster. This is caused by the fact that the detection mechanism is based on the normal patient response. During bypass the patient response changes, due to: a larger blood volume, extra delay time because the drug has to pass through the perfusion machine first, hypothermia, etc. The response becomes slower and therefore the period over which the oscillation can be detected should be enlarged too. In other words, during bypass the oscillation detection mechanism can be improved by changing the time period, after which the oscillation counter is decremented by one, from 250 seconds to 350 seconds.

The change in the patient response also has its effect on the performance of the controller at the start and end of the bypass stage. Although the gain always is adapted correctly during pre-bypass, the controller often acts to slowly to suppress the severe hypertension period at the start of bypass. This can be seen from Figure 5.11. A solution for this would be to determine the response of the patient and perfusion machine together and compute an extra set of optimal PI-parameters for the bypass stage. During bypass the dynamic response of the patient changes, which could be noted in the patient response model by an increase of the time delay T and the time constant τ.

![Figure 5.11: MAP, setpoint and flow at startup of the bypass phase, showing slow reaction of the controller to the severe hypertension starting at 11:33; Further up-transients (11:34 and 11:35:50) and a down transient (11:34:50) are detected.](image-url)

Another method to cope with the changed response is to increase the control gain at the start of bypass. It was already stated before that the gain adaptation mechanism increases the gain towards the end of the bypass stage in order to cope with the changed response and the temporary decrease of sensitivity. This however results in too aggressive control after aortic declamping when the patient response and the sensitivity return to normal again. An example of this is shown in Figure 5.12.

A solution for these problems at the start and end of bypass would therefore be to increase the gain at the start of bypass and decrease the gain to the pre-bypass level again at or just before aortic declamping. The controller has no knowledge about when these events occur. The start of bypass, however, already has to be marked by the anaesthetist in order to switch the MAP-validation to perfusion mode. At that moment also the gain level could be stored and increased
Clinical Evaluation

by one. Aortic declamping is only noted as comment for evaluation purposes at the moment. An extra command key could be added to switch the gain level back to pre-bypass, which ideally should be just before declamping. This demands an extra action of the anaesthetist though, which has to be remembered.

As an acceptable solution and safe mechanism, aortic declamping could be detected automatically. This is possible if both MAP-validation algorithms, for normal and perfusion mode, are used in parallel. When the normal algorithm detects a series of valid heart cycles, the aorta must have been declamped and the heart operating again. MAP-validation could also be switched back to normal mode automatically then. Detection of declamping should be safe and no incorrect detections should occur. Then automatic declamping detection can be used to adapt the gain back to the pre-bypass level.

Detection of the start of the bypass can also be considered, using the same principle of the two validation algorithms in parallel. This would give the system more information about the stage of the surgery with less interaction of the anaesthetist. To ensure safe detection of the start of the bypass phase is a little more complex. It has to be examined whether it is possible to detect a slow convergence of the systolic and diastolic pressure towards the MAP during successive heart cycles, using the normal validation algorithm, in combination with an increasing number of valid measurements using the validation algorithm for perfusion phase.

To verify whether the previously described mechanism could be applied, the gain adaptations during the various stages are examined in section 5.5. Before, an extreme situation which occurred during one of the tests will be discussed.

5.4.3. An Extreme Situation

An exceptional case during one of the clinical tests appeared when the anaesthetist, after succeeding to raise the blood pressure in an extreme hypotensive period with several infusions of a vaso-active drug, took a sample of the blood pressure, see Figure 5.13. During blood sampling the MAP measurements are invalid but a spike in the pressure signal, caused by the performed actions, was erroneously validated to be correct. At the end of bypass the gain was high and due to the previous invalid MAP values, the transient detection mechanism didn't act either. Thus the controller generated a high flow rate. The large amount of drug infused in the short period, before the next valid pressure showed the real MAP, caused a severe drop of the blood pressure again.
The erroneous performance of the controller caused by this rare combination of events, i.e. long hypotensive period, blood sampling, high gain, a signal spike and incorrect validation, is difficult to prevent. The risk of such a situation can be reduced by demanding valid measurements during two or more 5-second periods after a longer period (> 30 seconds) of invalid measurements. This however, slows down controller performance in the normal situations with invalid measurements. Another method to use can be limitation of the effect of such an event through usage of an adaptive maximum flow rate, dependent on previous flow rates, instead of a constant maximum.

The examples of the performance reviewed in this section, except the last one, show the robustness and safety of the controller. The MAP is in general stable and the various events are handled well. The MAP-validation and the transient detection mechanism perform very well. The oscillation detection mechanism acts appropriately and stabilizes control. During bypass, oscillation detection appears rather late due to the changed patient response and this can easily be improved. The changed patient response during bypass demands some changes to improve controller performance at the start and end of this stage. The gain adaptation, which performs well and ensures safety for all patients and their wide range of sensitivities, already copes with a part of this changed response. Adding extra knowledge to the system about how to adjust the gain before the patient characteristics change at the start of bypass and at declamping, as described, will improve performance.

5.5. Gain Adaptations

In order to evaluate the gain adaptation and extract knowledge about how gain adaptation can be extended to create better control at the start and end of bypass, the gain adaptations during all tests are examined. In Figure 5.14 the gain changes in time during all 33 tests are reflected for the three stages of cardiac surgery.

The gain can vary between 9 gain classes according to the values shown in Table 2.1 of chapter 2. These 9 classes are denoted by 0 for the lowest gain up to 8 for the highest gain. Out of precaution a surgery always starts with a low gain of class 2 because the sensitivity of the patient is unknown prior to the operation. This can clearly be seen in Figure 5.14(a). During the pre-bypass phase, which starts with the initial learning phase, the gain is adapted to become inversely proportional to the patient's sensitivity. The need for adaptive control is obvious from this figure, the used gain ranges widely between patients. However, the most extreme values were never
reached in this phase and most patients appeared to be rather sensitive or nominal, for 21 patients gain ended in class 2 and for 7 patients in class 4.

Figure 5.14: Gain adaptations in time during all 33 clinical tests. NB. each line connects the gain adaptations of one test in time; the numbers between parentheses at the end of the lines indicate the number of tests which ended that phase at the corresponding gain level.
During the bypass phase, Figure 5.14(b), many more adaptations are performed due to the already mentioned causes. In almost all clinical test cases the gain is eventually adapted upwards and reaching a new appropriate level within the first hour of the bypass. The range of gain classes used is large and only the lowest gain class was never used during the 33 tests. The number of tests ending at a certain gain level is much more distributed across the various classes for this stage, with the main centre above level 4, which corresponds with nominal.

The gain levels, reached at the end of bypass for the various tests, barely change anymore during the post-bypass. This is shown in Figure 5.14(c). This phase is often very short and post-operative hypertension is mostly treated in recovery, so control is not often required.

Gain adaptation is the key-quality of the controller. It ensures safe and robust control for all patients. The performance of the controller and the gain adaptation mechanism is, however, troubled at the start and end of bypass. At clamping and declamping of the aorta sudden changes in the patient's response characteristics appear, which the adaptation mechanism cannot foresee and react on fast enough. This results in too slow or too aggressive control in certain cases, which could be prevented by implementation of the mechanism described in the previous section. The gain adaptations during bypass show an eventual increase of the gain in almost all test cases. So incrementing the gain one class at the start of bypass seems to be safe, also because the gain down adaptation mechanisms act fast enough in case of a very sensitive patient.

5.6. Opinion of the Anaesthetists

The performance of the controller should be as good and safe as manual blood pressure regulation by the anaesthetists. Preferably even better of course but the fact that it relieves them for a great part from this time consuming task is already an advantage of computer controlled infusion. Because of the different circumstances it is not really possible to compare manual control to automatic control of the closed loop blood pressure controller. However, the experienced anaesthetists must be able to judge the performance of the controller related to their normal manual blood pressure control.

A small questionnaire was made and implemented in a small computer program. This program was automatically run after each clinical test, requesting a short and fast evaluation of the controller performance. The questionnaire contained a few questions to state information about the patient, surgery and anaesthesia in general, a number of questions to evaluate the performance of the controller during the specific case, and also a number of questions to evaluate the user interface. The questions about the surgery, anaesthesia and the performance evaluation, together with the possible answers are shown in Table 5.8.

The possible answers are given a corresponding value, which is also noted in the table. The number of times an answer was given during the total test series can now easily be evaluated. The results from this is given in Table 5.9.
Table 5.8: Questions the anaesthetists were asked to answer by a computer program after an automatic control session.

<table>
<thead>
<tr>
<th>Asked Question</th>
<th>Corresponding value:</th>
<th>Possible Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: The number of hemodynamic events related to surgery was?</td>
<td>1: small</td>
<td>2: medium</td>
</tr>
<tr>
<td>2: For this type of surgery anaesthesia was?</td>
<td>1: very easy</td>
<td>2: normal</td>
</tr>
<tr>
<td>Was the flow rate change slow compared to your normal manual SNP infusion?</td>
<td>1: never</td>
<td>2: almost</td>
</tr>
<tr>
<td>3: During Hypertensive periods</td>
<td>1: very stable</td>
<td>2: more</td>
</tr>
<tr>
<td>4: During Hypotensive periods</td>
<td>1: almost none</td>
<td>2: less</td>
</tr>
<tr>
<td>5: In what way did the automatic delivery change pressure stability?</td>
<td>1: sure</td>
<td>2: probably</td>
</tr>
<tr>
<td>7: If this application is available in the future, would you use it during this type of operation?</td>
<td>1: not</td>
<td>2: sure</td>
</tr>
</tbody>
</table>

The results of the evaluation show that the anaesthetists regarded the number of hemodynamic events and anaesthesia for the tests in the test series as normal, certainly not difficult. The speed of the controller in changing the flow rate, during both hypertensive or hypotensive periods, compared to manual control was almost never considered to be slow. It is the anaesthetists opinion that the automatic delivery, using this controller, increases pressure stability. So according to this evaluation the performance of the controller is satisfactory. It is obvious that the time spent on SNP delivery is less compared to manual control, but what is a sure sign that the controller's performance is adequate is the indication of the anaesthetists that they almost certainly would use the controller if available in the future.

Table 5.9: Results obtained from evaluation by the anaesthetists (Brussels '95) of 33 clinical tests. The questions and possible answer are presented in Table 5.8.

<table>
<thead>
<tr>
<th>Question no.</th>
<th>&quot;1&quot;</th>
<th>&quot;2&quot;</th>
<th>&quot;3&quot;</th>
<th>&quot;4&quot;</th>
<th>&quot;5&quot;</th>
<th>avr. 'answer'</th>
<th>std.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>4</td>
<td>17</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>2.4</td>
<td>0.9</td>
</tr>
<tr>
<td>(2)</td>
<td>6</td>
<td>13</td>
<td>8</td>
<td>6</td>
<td>0</td>
<td>2.4</td>
<td>1.0</td>
</tr>
<tr>
<td>(3)</td>
<td>7</td>
<td>14</td>
<td>9</td>
<td>3</td>
<td>0</td>
<td>2.2</td>
<td>0.9</td>
</tr>
<tr>
<td>(4)</td>
<td>8</td>
<td>11</td>
<td>12</td>
<td>1</td>
<td>1</td>
<td>2.3</td>
<td>1.0</td>
</tr>
<tr>
<td>(5)</td>
<td>6</td>
<td>15</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2.2</td>
<td>0.8</td>
</tr>
<tr>
<td>(6)</td>
<td>7</td>
<td>17</td>
<td>8</td>
<td>1</td>
<td>0</td>
<td>2.1</td>
<td>0.8</td>
</tr>
<tr>
<td>(7)</td>
<td>19</td>
<td>9</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1.6</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Besides evaluation of the performance of the controller, also the user-interface was evaluated in order to determine which items should definitely be in the new user-interface to be developed for the integrated system of Infusion Toolbox and controller. The items and the rating of how much the anaesthetist indicated to use them are given in Table 5.10.
Table 5.10: Results of the evaluation of the user interface by the anaesthetists (Brussels '95). They had to respond to the question: "How often did you use the following items on the screen during the test?" on a scale of 5.

<table>
<thead>
<tr>
<th>Item displayed</th>
<th>not</th>
<th>&quot;1&quot;</th>
<th>&quot;2&quot;</th>
<th>&quot;3&quot;</th>
<th>&quot;4&quot;</th>
<th>&quot;5&quot;</th>
<th>avr.</th>
<th>std.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setpoint</td>
<td>0</td>
<td>8</td>
<td>17</td>
<td>8</td>
<td>0</td>
<td>3.0</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Flow rate</td>
<td>4</td>
<td>10</td>
<td>11</td>
<td>7</td>
<td>1</td>
<td>2.7</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>Total SNP delivered</td>
<td>13</td>
<td>12</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2.1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>Message window</td>
<td>0</td>
<td>7</td>
<td>14</td>
<td>11</td>
<td>1</td>
<td>3.2</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>Graphical pressure window</td>
<td>0</td>
<td>2</td>
<td>11</td>
<td>18</td>
<td>2</td>
<td>3.6</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Graphical flow rate window</td>
<td>0</td>
<td>4</td>
<td>8</td>
<td>17</td>
<td>4</td>
<td>3.6</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>Graphical setpoint in window</td>
<td>1</td>
<td>15</td>
<td>14</td>
<td>3</td>
<td>0</td>
<td>2.6</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>Mean Arterial Pressure</td>
<td>0</td>
<td>2</td>
<td>15</td>
<td>15</td>
<td>1</td>
<td>3.5</td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>

The evaluation of the usage of the various items in the user-interface doesn't show much variation between the items. This could be an indication that the items have been chosen very well and all are seen as necessary or useful. Only the graphical displays showing the trend of the pressure and the flow and the numeric MAP are used a little more frequently and appreciated a lot.

Finally to end this short evaluation and put it into the right perspective, it should be noted that most tests and thus evaluations have been performed by two anaesthetists.

5.7. Conclusions

The overall performance of the controller can be noted as very satisfactory. The MAP was in general very stable and close to the desired target level. The various mechanisms performed adequate and control was safe for all patients during all the clinical tests. Only in one very extreme situation safety was not absolutely guaranteed. The gain adaptation mechanism determines the quality of this adaptive controller and was found to perform well. However, the performance of the controller can still be improved during the bypass stage of cardiac surgery, by taking into account the changed patient response during this phase.
Chapter 6

**Future Developments**

The previous chapters discussed the blood pressure controller and the Infusion toolbox system, presented a design for integration of both applications and showed the results of the clinical test series performed with the stand-alone blood pressure controller. This showed both sophisticated applications to be very practical and safe, even for the hostile environment of cardiac surgery. These or similar systems are going to be common practice in the O.R. and improve the quality of patient care. Further enhancement and development still remains. Some of the improvements or next developments which can be expected in the near future will be mentioned in this chapter. First, for the blood pressure controller and secondly for the complete system of Infusion Toolbox and controller.

6.1. The Blood Pressure Controller

The clinical evaluation shows that the MAP can robustly and safely be regulated towards and maintained at a specified target level by computer controlled infusion of nitroprusside using this system. Nevertheless improvement of the performance of the controller is still possible and required in some cases. In particular at the start and end of bypass the performance must be enhanced by taking into account the sudden changes of the patient's response and use this knowledge to adapt the control gain in advance.

To perform this automatically and ensure it cannot be forgotten, detection of the start of bypass and of aortic declamping should be implemented. The MAP-validation algorithms for normal and perfusion mode can be used in parallel to detect the disappearance or the return of valid heart cycle curves of the pressure. The MAP-validation has shown to be safe enough. Therefore it can also be used to automatically switch back the controller to automatic control mode in the special case when a return to manual mode was made due to invalid pressure measurements. The latter normally only occurs due to the taking of blood samples by the anaesthetist.

After such a switch to manual the flow of SNP is now kept constant at the last computed level and an audible alarm warns the anaesthetist, who then has to take action or push a button in order to inform the system it is safe to go to automatic control again. Returning to automatic mode as soon as a number of valid MAP-samples is detected again is more safe because the real MAP might deviate too much during the manual control period while the flow is not adapted. So if the anaesthetist doesn't react within a short time, for example due to one of the many other activities, MAP control is less stable and safe.

An automatic return to manual mode of the controller should, anyway, always generate an alarm message and an audible alarm. The alarm should be loud enough and clearly perceptible from
other alarms. The anaesthetists requested, besides of course to be able to stop the alarm and handle the event, the possibility to snooze an alarm temporarily. This would be a button to indicate that the alarm is noticed but a more important activity has to be performed or finished first, so the audible alarm should be silenced but sound again if within a certain safe time period no appropriate action is taken. The Infusion Toolbox application already contains standard alarms like this in order to indicate important events needing adequate action of the anaesthetist. Therefore, this is already accomplished in the integrated system of the Toolbox and the controller, because in that case the controller requests the central computer to generate the alarm.

6.2 The Infusion Toolbox

The final goal for development of the Infusion Toolbox is central monitoring and control of anaesthesia. The assistance of the anaesthetist with the preparation and evaluation of infusion schemes and with automatic control of simultaneous infusion of various drugs is already accomplished. Due to integration with the blood pressure controller, closed loop control of the MAP can be performed and acquisition of the pressure signal becomes possible for the Toolbox. This is another step in the direction of a machine for total intravenous anaesthesia (TIVA), which also should include connection to the Hospital Information System (HIS) and other intelligent systems and data acquisition devices, e.g. for temperature, for ECG, for Auditory Evoked Potentials (AEPs), etc.

The integrated system of the Infusion Toolbox and the controller creates the opportunity to increase control of the patient's blood pressure even more. The clinical evaluation in the previous chapter showed that long hypotensive periods during cardiac surgery can occur. The anaesthetists suggested to develop an equivalent controller to suppress these hypotensive periods and control the MAP by automatic infusion of a vaso-active drug, which increases the pressure. Because this opposes the effect of MAP-control using nitroprusside, which decreases the pressure, both controllers should be aware of each others actions and be complementary. This can now be accomplished by expanding the controller using SNP with a similar sub-system which is activated if certain conditions, requiring an increase of the pressure, become true. Then another ControlPanel in the Infusion Toolbox environment could be activated to infuse the computed flow of the vaso-active drug, e.g. nor-adrenalin, through another infusion device. As soon as the MAP approaches the desired target level, the main controller based on nitroprusside can assume control again.

A system like this, using nor-adrenalin, is already being developed by Ir. B. Hoeksel at the Academic Hospital of Maastricht. At the moment initial tests are performed and the results will be presented in a publication at the end of this year. This system uses three infusion devices because the pulmonary arterial pressure is also controlled automatically using the drug nitroglycerine (NTG). So the expert system is also extended for combined usage of a NTG-controller and a SNP-controller, which is even more complex because NTG also influences the mean arterial pressure.
Chapter 7

Conclusions and Recommendations

The blood pressure controller was used in 33 test cases during cardiac surgery, which is probably the most hostile environment for a blood pressure controller due to the influences of surgical actions and other infused drugs on the mean arterial pressure (MAP). This clinical evaluation showed the controller to perform well and safe. The controller was in automatic mode for more than 90% of the time of the operations and a necessary switch to manual control only occurred four times during the total test series. During automatic control the MAP was within 10 mmHg distance of the setpoint for 71% of the time with an average distance of 8 mmHg.

The pressure can only be decreased because the controller uses the drug sodium nitroprusside (SNP). Therefore judgement of the performance during effective control, i.e. SNP flow larger than zero, is more correct. During effective control the MAP was within 10 mmHg distance of the setpoint for 89% of the time and the average distance to the setpoint was 4.5 mmHg, which is very good performance.

The gain adaptation mechanism worked well and is the key quality of the controller. It copes with the wide range of the patient’s sensitivity and ensures safety for all patients. At the start of bypass and after aortic declamping control is sometimes, respectively, too slow or too aggressive. This is caused by the temporary change in patient’s response and sensitivity. The performance of the controller can easily be improved by adapting the gain one or two classes upwards at the start of bypass and adapting it back to the pre-bypass value at declamping. This can be accomplished automatically by using the two pressure validation algorithms in parallel, in order to detect these events.

The various safety mechanisms implemented, e.g. oscillation detection, transient detection and pressure measurement validation, also showed good performance during the total test series. Oscillation detection acted safely and stabilised the MAP when necessary. During the bypass phase of the operation, oscillation detection could act faster through usage of a larger time period before the counter is decreased. The transient detection and pressure validation, except in one extreme situation, ensured absolute safety.

The second objective of this research was to integrate the blood pressure controller and the Infusion Toolbox. A design, based on the communication structure of the Toolbox, was made and implemented. The design is modular to make further extension of the Toolbox possible, which is necessary in order to be able to reach the final goal: a central terminal to control and monitor all functions and devices during anaesthesia. Therefore, a client-server relation exists in which the Infusion Toolbox sends commands to the controller and one of the infusion devices
according to the information obtained from the blood pressure controller, e.g. the MAP and the flow needed to regulate the pressure towards a desired target level.

The first prototype of the integrated design is not completely stable yet and has to be tested extensively through simulations first. In particular rules concerning disconnection of the transport service have to be implemented in the expert system and tested. The communication network and protocol are provided now and their robustness can be tested, which is absolutely necessary for medical applications.

Although the blood pressure controller was used as stand-alone application during all the test, infusion of other drugs was computer controlled using the Infusion Toolbox. A long term evaluation is necessary and performance with and without automatic delivery of other drugs should be compared. Through this it should be verified whether automatic delivery of other drugs affects the performance of the controller.
References

Barvais, L., E. Coussaert, F. Cantraine and A. d'Hollander.
*The Pharmacokinetics of Intravenous Anaesthetic Drugs Given by Infusion: SPINA - A Software Program.*

Barvais, L., F. Cantraine, E. Coussaert and A. d'Hollander.
*CINA: a software to Compose Infusion sheets for IV Anesthetic drugs.*

*Use of a Computerized Closed-Loop Sodium Nitroprusside Titration System for Antihypertensive Treatment After Open Heart Surgery.*
Critical Care Medicine, vol. 18; no. 10: pp. 1061-1065, 1990.

Blom, J.A.
*The SIMPLEXYS Experiment: Real Time Expert Systems in Patient Monitoring.*

Cantraine, F.

Cantraine, F., E. Coussaert, A. d'Hollander and L. Barvais.
*MINA: Monitoring of Intravenous Anaesthesia.*

*Sedation After Cardiac Bypass Surgery: Comparison of Propofol and Midazolam in the Presence of a Computerized Closed Loop Arterial Pressure Controller.*

*Evaluation of Closed Loop Control of Arterial Pressure During Hypotensive Anaesthesia for Local Resection of Intraocular Melanoma.*

*Multicenter Trial of Automated Nitroprusside Infusion for Postoperative Hypertension.*

Delapasse, J.S., K. Behbehani, K. Tsui and K.W. Klein.
*Accomodation of Time Delay Variations in Automatic Infusion of Sodium Nitroprusside.*

Duncan, R.
*Advanced MSDOS; The Microsoft Guide for Assembly Language and C Programmers.*
Friederich, J.A. and J.F. Butterworth IV.  
*Sodium Nitroprusside: Twenty Years and Counting.*  

Geene, W.J. van  
*Clinical Evaluation of a Blood Pressure Controller in Anaesthesia.*  
M.Sc. thesis, Faculty of Electrical Engineering, Eindhoven University of Technology,  
The Netherlands, 1993.

*Pilot Study of an Expert System Adviser for Controlling General Anaesthesia.*  

Halsall, F.  
*Data Communications, Computer Networks and Open Systems.*  

O'Hara, D.A., D.K. Bogen and A. Noordergraaf.  
*The Use of Computers for Controlling the Delivery of Anesthesia.*  

Hoeksel, S.A.A.P.  
*Adaptive Blood Pressure Control Using Nitroglycerin.*  
M.Sc. thesis, Faculty of Electrical Engineering, Eindhoven University of Technology,  
The Netherlands, 1992.

d'Hollander, A.A., F. Cantraine, E. Coussaert and L. Barvais.  
*PC-Based Multistep Infusion Regimens and On-line Control of Four Infusion Devices.*  
Conf. Paper. Knowledge Based Systems in Medicine: Methods, Applications and  
Evaluation. Proceedings of the Workshop "System Engineering in Medicine". Springer-  

Isaka, S. and A.V. Sebald.  
*Control Strategies for Arterial Blood Pressure Regulation.*  

Jackson, P.  
*Introduction to Expert Systems.*  
2nd ed., Addison-Wesley (UK), 1990.

*Evaluation of a Long-Range Adaptive Predictive Controller for Computerized Drug  
Delivery Systems.*  

Lammers, J.O.  
*Knowledge Based Adaptive Blood Pressure Control: A SIMPLEXYS Expert System  
Application.*  
AIO thesis, Faculty of Electrical Engineering, Eindhoven University of Technology,  
The Netherlands, 1990.

Lutgens, J.M.A.  
*Knowledge Base Correctness Checking for SIMPLEXYS Expert Systems.*  
M.Sc. thesis, Faculty of Electrical Engineering, Eindhoven University of Technology,  
The Netherlands, 1990.

*Improved Safety and Efficacy in Adaptive Control of Arterial Pressure through the use of  
a Supervisor.*  
References

Supervisory Adaptive Control of Arterial Pressure During Cardiac Surgery.

McMenemin, I.M., J.A. Church and G.N.C. Kenny.
Sedation Following Cardiac Surgery: Evaluation of Alfentanil and Morphine in the
Presence of a Computerized Closed Loop Arterial Pressure Controller.

Model Reference Adaptive Control with Constraints for Postoperative Blood Pressure
Management.

Philippens, E.H.J.
M.Sc. thesis, Faculty of Electrical Engineering, Eindhoven University of Technology,
The Netherlands, 1990.

Scientific Solutions Inc.

Scientific Solutions Inc.
Lab Master DMA: Handbook.

Shafer, D. and D.A. Ritz.
Practical Smalltalk: Using Smalltalk/V.

Tavernier, A., E. Coussaert, A. d'Hollander and F. Cantraine.
Model-based Pharmacokinetic Regulation in Computer-assisted Anesthesia.
An Interactive System: CARIN.

Designing Object-Oriented Software.

Fuzzy Control of Mean Arterial Pressure in Postsurgical Patients with Sodium
Nitroprusside Infusion.

Zwart, R.
Implementatie en Evaluatie van een Robuuste Adaptieve Bloeddrukregelaar.
M.Sc. thesis (in Dutch), Faculty of Electrical Engineering, Eindhoven University of
Appendices

Appendix A: Tables with the complete statistical data extracted from the clinical tests.

Appendix B: Notations used in the flow diagram.

Appendix C: Instruction Manual for the Anaesthetists.
Appendix A: Tables with the complete statistical data from the clinical tests.

Table A.1: Average MAP, average distance to the setpoint and average flow rate during the total operation, during automatic control and during effective control (flow rate > 0) for all tests performed in Brussels (1995).

<table>
<thead>
<tr>
<th>test no.</th>
<th>average MAP [mmHg]</th>
<th>average MAP - setpoint</th>
<th>average flow rate [mg/hr]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>total</td>
<td>automatic</td>
<td>eff.</td>
</tr>
<tr>
<td>1</td>
<td>76.6</td>
<td>76.4 (5.4)</td>
<td>78.5</td>
</tr>
<tr>
<td>2</td>
<td>75.4</td>
<td>75.4 (5.0)</td>
<td>77.6</td>
</tr>
<tr>
<td>3</td>
<td>73.1</td>
<td>73.0 (4.1)</td>
<td>74.4</td>
</tr>
<tr>
<td>4</td>
<td>73.3</td>
<td>72.9 (12.0)</td>
<td>77.3</td>
</tr>
<tr>
<td>5</td>
<td>68.5</td>
<td>66.4 (6.6)</td>
<td>74.8</td>
</tr>
<tr>
<td>6</td>
<td>74.2</td>
<td>74.2 (3.2)</td>
<td>74.4</td>
</tr>
<tr>
<td>7</td>
<td>73.7</td>
<td>73.7 (2.9)</td>
<td>74.2</td>
</tr>
<tr>
<td>10</td>
<td>67.8</td>
<td>67.9 (6.0)</td>
<td>71.0</td>
</tr>
<tr>
<td>11</td>
<td>69.2</td>
<td>69.2 (7.6)</td>
<td>72.5</td>
</tr>
<tr>
<td>12</td>
<td>73.6</td>
<td>73.5 (5.1)</td>
<td>74.3</td>
</tr>
<tr>
<td>14</td>
<td>77.3</td>
<td>77.4 (7.1)</td>
<td>78.5</td>
</tr>
<tr>
<td>16</td>
<td>75.6</td>
<td>75.7 (4.6)</td>
<td>76.5</td>
</tr>
<tr>
<td>17</td>
<td>81.4</td>
<td>80.5 (7.7)</td>
<td>82.3</td>
</tr>
<tr>
<td>18</td>
<td>69.7</td>
<td>69.8 (5.2)</td>
<td>72.6</td>
</tr>
<tr>
<td>19*</td>
<td>67.6</td>
<td>66.9 (10.4)</td>
<td>72.1</td>
</tr>
<tr>
<td>20</td>
<td>75.8</td>
<td>76.0 (4.5)</td>
<td>77.5</td>
</tr>
<tr>
<td>21</td>
<td>78.9</td>
<td>82.7 (10.1)</td>
<td>85.0</td>
</tr>
<tr>
<td>22</td>
<td>72.0</td>
<td>72.6 (11.1)</td>
<td>81.6</td>
</tr>
<tr>
<td>23</td>
<td>72.5</td>
<td>72.8 (8.3)</td>
<td>82.9</td>
</tr>
<tr>
<td>24</td>
<td>75.6</td>
<td>78.1 (8.8)</td>
<td>84.2</td>
</tr>
<tr>
<td>26</td>
<td>69.6</td>
<td>69.9 (12.0)</td>
<td>76.1</td>
</tr>
<tr>
<td>27</td>
<td>78.5</td>
<td>78.5 (5.8)</td>
<td>80.9</td>
</tr>
<tr>
<td>28</td>
<td>77.9</td>
<td>77.7 (5.8)</td>
<td>76.9</td>
</tr>
<tr>
<td>29</td>
<td>70.1</td>
<td>70.3 (6.6)</td>
<td>72.2</td>
</tr>
<tr>
<td>30</td>
<td>72.5</td>
<td>72.5 (5.9)</td>
<td>75.8</td>
</tr>
<tr>
<td>31</td>
<td>79.8</td>
<td>80.1 (7.5)</td>
<td>82.7</td>
</tr>
<tr>
<td>32</td>
<td>77.0</td>
<td>76.7 (10.0)</td>
<td>77.8</td>
</tr>
<tr>
<td>34</td>
<td>70.7</td>
<td>71.0 (5.5)</td>
<td>74.4</td>
</tr>
<tr>
<td>35</td>
<td>74.2</td>
<td>74.3 (7.0)</td>
<td>77.1</td>
</tr>
<tr>
<td>36</td>
<td>77.8</td>
<td>77.7 (10.4)</td>
<td>84.5</td>
</tr>
<tr>
<td>37</td>
<td>75.5</td>
<td>74.5 (10.2)</td>
<td>85.5</td>
</tr>
<tr>
<td>38</td>
<td>83.8</td>
<td>84.5 (6.0)</td>
<td>85.7</td>
</tr>
<tr>
<td>39</td>
<td>71.4</td>
<td>73.1 (8.0)</td>
<td>80.0</td>
</tr>
<tr>
<td>40</td>
<td>68.9</td>
<td>69.1 (9.5)</td>
<td>79.3</td>
</tr>
<tr>
<td>avg</td>
<td>74.3</td>
<td>74.5 (7.1)</td>
<td>78.2</td>
</tr>
<tr>
<td>std</td>
<td>3.9</td>
<td>4.1 (2.4)</td>
<td>4.2</td>
</tr>
</tbody>
</table>

* In all tables test 19 is not included in the total average because it was the only heart transplant in the test series.
Table A.2: Average MAP, average distance to the setpoint and the average flow rate, all during automatic control, for the pre-, during- and post-bypass phase of all operations.

<table>
<thead>
<tr>
<th>test no.</th>
<th>average MAP [mmHg]</th>
<th>avg I MAP - setpoint I [mmHg]</th>
<th>average flow rate [mg/hr]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>pre</td>
<td>during</td>
<td>post</td>
</tr>
<tr>
<td>1</td>
<td>82.1</td>
<td>70.5</td>
<td>78.2</td>
</tr>
<tr>
<td>2</td>
<td>84.4</td>
<td>69.5</td>
<td>73.7</td>
</tr>
<tr>
<td>3</td>
<td>77.2</td>
<td>70.0</td>
<td>75.6</td>
</tr>
<tr>
<td>4</td>
<td>81.4</td>
<td>67.9</td>
<td>71.3</td>
</tr>
<tr>
<td>5</td>
<td>67.3</td>
<td>68.5</td>
<td>59.6</td>
</tr>
<tr>
<td>6</td>
<td>77.9</td>
<td>69.9</td>
<td>77.8</td>
</tr>
<tr>
<td>7</td>
<td>77.3</td>
<td>69.9</td>
<td>75.8</td>
</tr>
<tr>
<td>8</td>
<td>65.3</td>
<td>68.9</td>
<td>68.2</td>
</tr>
<tr>
<td>9</td>
<td>75.5</td>
<td>65.0</td>
<td>66.7</td>
</tr>
<tr>
<td>10</td>
<td>79.4</td>
<td>71.5</td>
<td>74.0</td>
</tr>
<tr>
<td>11</td>
<td>81.5</td>
<td>75.8</td>
<td>-</td>
</tr>
<tr>
<td>12</td>
<td>75.1</td>
<td>74.4</td>
<td>79.3</td>
</tr>
<tr>
<td>13</td>
<td>86.7</td>
<td>72.4</td>
<td>81.3</td>
</tr>
<tr>
<td>14</td>
<td>74.0</td>
<td>67.6</td>
<td>72.0</td>
</tr>
<tr>
<td>15</td>
<td>71.3</td>
<td>67.0</td>
<td>67.0</td>
</tr>
<tr>
<td>16</td>
<td>84.3</td>
<td>78.0</td>
<td>73.6</td>
</tr>
<tr>
<td>17</td>
<td>83.8</td>
<td>73.6</td>
<td>77.6</td>
</tr>
<tr>
<td>18</td>
<td>71.3</td>
<td>68.8</td>
<td>71.9</td>
</tr>
<tr>
<td>19*</td>
<td>59.4</td>
<td>68.2</td>
<td>66.6</td>
</tr>
<tr>
<td>20</td>
<td>78.0</td>
<td>71.0</td>
<td>80.0</td>
</tr>
<tr>
<td>21</td>
<td>87.4</td>
<td>82.1</td>
<td>51.9</td>
</tr>
<tr>
<td>22</td>
<td>71.7</td>
<td>78.2</td>
<td>55.9</td>
</tr>
<tr>
<td>23</td>
<td>70.0</td>
<td>77.2</td>
<td>60.0</td>
</tr>
<tr>
<td>24</td>
<td>68.0</td>
<td>81.8</td>
<td>79.1</td>
</tr>
<tr>
<td>25</td>
<td>86.1</td>
<td>71.3</td>
<td>67.0</td>
</tr>
<tr>
<td>26</td>
<td>84.3</td>
<td>78.0</td>
<td>73.6</td>
</tr>
<tr>
<td>27</td>
<td>83.8</td>
<td>73.6</td>
<td>77.6</td>
</tr>
<tr>
<td>28</td>
<td>71.3</td>
<td>68.8</td>
<td>71.9</td>
</tr>
<tr>
<td>29</td>
<td>77.8</td>
<td>68.4</td>
<td>69.7</td>
</tr>
<tr>
<td>30</td>
<td>83.1</td>
<td>77.8</td>
<td>-</td>
</tr>
<tr>
<td>31</td>
<td>80.3</td>
<td>71.7</td>
<td>77.7</td>
</tr>
<tr>
<td>32</td>
<td>68.7</td>
<td>70.1</td>
<td>73.2</td>
</tr>
<tr>
<td>33</td>
<td>77.7</td>
<td>68.1</td>
<td>79.9</td>
</tr>
<tr>
<td>34</td>
<td>83.1</td>
<td>78.5</td>
<td>59.9</td>
</tr>
<tr>
<td>35</td>
<td>79.0</td>
<td>72.5</td>
<td>63.7</td>
</tr>
<tr>
<td>36</td>
<td>90.7</td>
<td>83.2</td>
<td>81.1</td>
</tr>
<tr>
<td>37</td>
<td>78.2</td>
<td>70.5</td>
<td>-</td>
</tr>
<tr>
<td>38</td>
<td>70.0</td>
<td>68.7</td>
<td>57.3</td>
</tr>
<tr>
<td>avg</td>
<td>77.4</td>
<td>72.5</td>
<td>71.1</td>
</tr>
<tr>
<td>std</td>
<td>6.5</td>
<td>4.6</td>
<td>8.2</td>
</tr>
</tbody>
</table>

In the tests 14, 31 and 39 the controller was switched off at the end of bypass, therefore no post-bypass value is listed in these cases.
Table A.3: Percentage of time the MAP was maintained within a certain distance range from the setpoint during the total operation.

<table>
<thead>
<tr>
<th>test no.</th>
<th>percentage of total time the MAP was in the given distance range to the setpoint</th>
<th>total time [min.]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[distance range in mmHg]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-30 -25 -20 -15 -10 -5 0 5 10 15 20 25 30</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0 0 1 6 14 23 28 13 6 6 2 0</td>
<td>165</td>
</tr>
<tr>
<td>2</td>
<td>0 0 2 8 9 28 28 16 5 2 1 1</td>
<td>202</td>
</tr>
<tr>
<td>3</td>
<td>0 0 2 7 13 29 41 7 1 0 0</td>
<td>220</td>
</tr>
<tr>
<td>4</td>
<td>2 4 8 6 26 31 5 2 3 1</td>
<td>244</td>
</tr>
<tr>
<td>5</td>
<td>0 6 6 24 23 11 3 7 3</td>
<td>154</td>
</tr>
<tr>
<td>6</td>
<td>0 0 1 2 9 41 41 4 1 0</td>
<td>328</td>
</tr>
<tr>
<td>7</td>
<td>0 0 1 1 7 37 40 12 2 0</td>
<td>272</td>
</tr>
<tr>
<td>10</td>
<td>0 5 2 13 21 24 28 4 1 0</td>
<td>222</td>
</tr>
<tr>
<td>11</td>
<td>2 4 9 11 15 31 23 4 1 0</td>
<td>225</td>
</tr>
<tr>
<td>12</td>
<td>0 0 2 5 11 33 31 12 4 0</td>
<td>246</td>
</tr>
<tr>
<td>14</td>
<td>1 1 3 5 10 24 26 14 7 4</td>
<td>168</td>
</tr>
<tr>
<td>16</td>
<td>0 0 1 2 16 24 36 13 5 2</td>
<td>179</td>
</tr>
<tr>
<td>17</td>
<td>1 3 6 5 18 26 25 10 2 1</td>
<td>336</td>
</tr>
<tr>
<td>18</td>
<td>0 0 5 16 18 24 34 3 1 0</td>
<td>191</td>
</tr>
<tr>
<td>19*</td>
<td>3 5 8 8 15 24 21 5 5 0</td>
<td>327</td>
</tr>
<tr>
<td>20</td>
<td>0 0 2 3 10 38 31 10 3 0</td>
<td>246</td>
</tr>
<tr>
<td>21</td>
<td>9 3 2 7 10 21 12 8 7 4 2</td>
<td>215</td>
</tr>
<tr>
<td>22</td>
<td>12 11 5 23 13 9 14 0 0 0</td>
<td>259</td>
</tr>
<tr>
<td>23</td>
<td>11 26 17 10 25 5 3 1 0</td>
<td>235</td>
</tr>
<tr>
<td>24</td>
<td>7 18 19 10 8 9 16 7 1 1 1</td>
<td>216</td>
</tr>
<tr>
<td>26</td>
<td>6 9 8 8 8 12 25 8 1 2 1</td>
<td>192</td>
</tr>
<tr>
<td>27</td>
<td>2 1 2 5 11 23 34 18 4 0</td>
<td>198</td>
</tr>
<tr>
<td>28</td>
<td>0 3 6 18 22 20 21 7 2 0 1</td>
<td>278</td>
</tr>
<tr>
<td>29</td>
<td>2 3 4 11 22 19 35 4 0 0 0</td>
<td>166</td>
</tr>
<tr>
<td>30</td>
<td>0 4 5 10 13 33 19 11 3 1 0</td>
<td>217</td>
</tr>
<tr>
<td>31</td>
<td>1 1 3 6 12 28 23 12 6 3 2</td>
<td>311</td>
</tr>
<tr>
<td>32</td>
<td>3 5 6 8 8 12 25 8 5 3 1</td>
<td>261</td>
</tr>
<tr>
<td>34</td>
<td>0 0 9 19 23 23 16 6 2 0</td>
<td>135</td>
</tr>
<tr>
<td>35</td>
<td>2 4 5 9 16 28 22 8 3 2 0</td>
<td>261</td>
</tr>
<tr>
<td>36</td>
<td>7 3 7 12 10 13 24 5 5 4 1</td>
<td>272</td>
</tr>
<tr>
<td>37</td>
<td>5 9 9 17 24 15 5 2 0 0 0</td>
<td>304</td>
</tr>
<tr>
<td>38</td>
<td>1 1 4 5 11 28 24 15 4 2 2</td>
<td>263</td>
</tr>
<tr>
<td>39</td>
<td>2 8 14 17 17 20 17 2 1 0</td>
<td>333</td>
</tr>
<tr>
<td>40</td>
<td>8 17 12 9 15 18 13 5 1 1 0</td>
<td>257</td>
</tr>
<tr>
<td>avg</td>
<td>2.5 4.5 5.7 9.3 14.2 23.5 24.3 7.8 2.8 1.3 0.5 0.3 235</td>
<td></td>
</tr>
<tr>
<td>std</td>
<td>3.4 5.9 4.5 5.4 5.5 8.2 9.6 4.6 2.2 1.6 0.8 0.6 52</td>
<td></td>
</tr>
</tbody>
</table>
Table A.4: Percentage of time the MAP was maintained within a certain distance range from the setpoint during automatic control.

<table>
<thead>
<tr>
<th>test no.</th>
<th>percentage of total time the MAP was in the given distance range to the setpoint</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[distance range in mmHg]</td>
<td>[min.]</td>
</tr>
<tr>
<td>-30</td>
<td>-25</td>
<td>-20</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19*</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>22</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>23</td>
<td>11</td>
<td>26</td>
</tr>
<tr>
<td>24</td>
<td>6</td>
<td>17</td>
</tr>
<tr>
<td>26</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>27</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>29</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>31</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>34</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>35</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>36</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>37</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>38</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>39</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>40</td>
<td>9</td>
<td>17</td>
</tr>
<tr>
<td>avg</td>
<td>2.4</td>
<td>4.6</td>
</tr>
<tr>
<td>std</td>
<td>3.6</td>
<td>6.2</td>
</tr>
</tbody>
</table>
Table A.5: Percentage of time the MAP was maintained within a certain distance range from the setpoint during effective control (flow rate > 0).

<table>
<thead>
<tr>
<th>test no.</th>
<th>percentage of total time the MAP was in the given distance range to the setpoint</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[distance range in mmHg]</td>
<td></td>
</tr>
</tbody>
</table>
|          | -30 | -25 | -20 | -15 | -10 | -5 | 0 | 5 | 10 | 15 | 20 | 25 | 30 |[
| 1        | 0   | 0   | 0   | 2   | 1   | 25 | 33 | 14 | 6  | 6  | 3  | 0  | 134 |
| 2        | 0   | 0   | 0   | 2   | 8   | 25 | 35 | 20 | 7  | 2  | 1  | 1  | 160 |
| 3        | 0   | 0   | 0   | 0   | 3   | 25 | 59 | 11 | 1  | 0  | 0  | 0  | 147 |
| 4        | 1   | 1   | 2   | 2   | 3   | 33 | 42 | 7  | 2  | 3  | 1  | 1  | 176 |
| 5        | 0   | 0   | 0   | 3   | 5   | 33 | 18 | 8  | 22 | 11 | 0  | 0  | 42  |
| 6        | 0   | 0   | 0   | 0   | 4   | 40 | 47 | 5  | 1  | 0  | 0  | 0  | 281 |
| 7        | 0   | 0   | 0   | 0   | 5   | 36 | 43 | 13 | 2  | 0  | 0  | 0  | 246 |
| 8        | 0   | 0   | 0   | 2   | 6   | 29 | 52 | 8  | 2  | 0  | 0  | 0  | 121 |
| 9        | 0   | 1   | 5   | 3   | 3   | 36 | 44 | 8  | 1  | 0  | 0  | 0  | 114 |
| 10       | 0   | 0   | 0   | 1   | 6   | 35 | 38 | 14 | 4  | 0  | 0  | 0  | 1  | 200 |
| 11       | 0   | 0   | 0   | 1   | 7   | 22 | 44 | 16 | 6  | 2  | 0  | 0  | 1  | 143 |
| 12       | 0   | 0   | 0   | 2   | 6   | 44 | 36 | 10 | 0  | 1  | 1  | 1  | 138 |
| 13       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 14       | 1   | 0   | 2   | 8   | 23 | 29 | 16 | 8  | 4  | 3  | 1  | 1  | 150 |
| 15       | 0   | 0   | 0   | 1   | 7   | 22 | 44 | 16 | 6  | 2  | 0  | 0  | 1  | 143 |
| 16       | 0   | 0   | 0   | 2   | 6   | 44 | 36 | 10 | 0  | 1  | 1  | 1  | 138 |
| 17       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 18       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 19*      | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 20       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 21       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 22       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 23       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 24       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 25       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 26       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 27       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 28       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 29       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 30       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 31       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 32       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 33       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 34       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 35       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 36       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 37       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 38       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 39       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| 40       | 0   | 0   | 0   | 0   | 3   | 29 | 61 | 6  | 1  | 0  | 0  | 0  | 105 |
| avg      | 0.2 | 0.2 | 0.8 | 1.7 | 5.8 | 30.1 | 40.4 | 12.3 | 4.4 | 2.0 | 0.8 | 0.5 | 135 |
| std      | 0.4 | 0.4 | 1.1 | 1.1 | 4.4 | 7.3 | 11.0 | 4.6 | 4.2 | 2.5 | 1.1 | 0.7 | 57  |
Table A.6: Percentage of time the MAP was maintained within a certain distance range from the setpoint during the time the pressure was above the setpoint (‘hypertension’).

<table>
<thead>
<tr>
<th>test no.</th>
<th>percentage of total time the MAP was in the given distance range to the setpoint</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[distance range in mmHg]</td>
<td>[0..5]</td>
</tr>
<tr>
<td>1</td>
<td>53</td>
<td>23</td>
</tr>
<tr>
<td>2</td>
<td>53</td>
<td>31</td>
</tr>
<tr>
<td>3</td>
<td>82</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>72</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>87</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>74</td>
<td>23</td>
</tr>
<tr>
<td>10</td>
<td>84</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>83</td>
<td>15</td>
</tr>
<tr>
<td>12</td>
<td>64</td>
<td>24</td>
</tr>
<tr>
<td>14</td>
<td>46</td>
<td>25</td>
</tr>
<tr>
<td>16</td>
<td>62</td>
<td>23</td>
</tr>
<tr>
<td>17</td>
<td>74</td>
<td>21</td>
</tr>
<tr>
<td>18</td>
<td>90</td>
<td>8</td>
</tr>
<tr>
<td>19*</td>
<td>63</td>
<td>18</td>
</tr>
<tr>
<td>20</td>
<td>69</td>
<td>21</td>
</tr>
<tr>
<td>21</td>
<td>34</td>
<td>22</td>
</tr>
<tr>
<td>22</td>
<td>74</td>
<td>23</td>
</tr>
<tr>
<td>23</td>
<td>91</td>
<td>9</td>
</tr>
<tr>
<td>24</td>
<td>64</td>
<td>24</td>
</tr>
<tr>
<td>26</td>
<td>67</td>
<td>22</td>
</tr>
<tr>
<td>27</td>
<td>61</td>
<td>31</td>
</tr>
<tr>
<td>28</td>
<td>67</td>
<td>23</td>
</tr>
<tr>
<td>29</td>
<td>89</td>
<td>11</td>
</tr>
<tr>
<td>30</td>
<td>54</td>
<td>32</td>
</tr>
<tr>
<td>31</td>
<td>51</td>
<td>24</td>
</tr>
<tr>
<td>32</td>
<td>59</td>
<td>19</td>
</tr>
<tr>
<td>34</td>
<td>63</td>
<td>25</td>
</tr>
<tr>
<td>35</td>
<td>65</td>
<td>22</td>
</tr>
<tr>
<td>36</td>
<td>64</td>
<td>11</td>
</tr>
<tr>
<td>37</td>
<td>65</td>
<td>21</td>
</tr>
<tr>
<td>38</td>
<td>50</td>
<td>31</td>
</tr>
<tr>
<td>39</td>
<td>84</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>65</td>
<td>27</td>
</tr>
<tr>
<td>avg</td>
<td>66.4</td>
<td>20.2</td>
</tr>
<tr>
<td>std</td>
<td>14.9</td>
<td>6.8</td>
</tr>
</tbody>
</table>
Table A.7: The total time of the operations and the time the controller was in manual mode or automatic mode during each operation.

<table>
<thead>
<tr>
<th>test no.</th>
<th>Total time [min.]</th>
<th>Automatic control [min.] [%]</th>
<th>Manual control [min] [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>165</td>
<td>157</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>202</td>
<td>199</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>220</td>
<td>200</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>244</td>
<td>233</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>154</td>
<td>117</td>
<td>37</td>
</tr>
<tr>
<td>6</td>
<td>328</td>
<td>327</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>272</td>
<td>267</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>222</td>
<td>220</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>225</td>
<td>215</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>246</td>
<td>242</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>168</td>
<td>165</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>179</td>
<td>178</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>336</td>
<td>263</td>
<td>73</td>
</tr>
<tr>
<td>14</td>
<td>191</td>
<td>188</td>
<td>3</td>
</tr>
<tr>
<td>15</td>
<td>327</td>
<td>198</td>
<td>129</td>
</tr>
<tr>
<td>16</td>
<td>246</td>
<td>220</td>
<td>26</td>
</tr>
<tr>
<td>17</td>
<td>215</td>
<td>148</td>
<td>67</td>
</tr>
<tr>
<td>18</td>
<td>259</td>
<td>189</td>
<td>70</td>
</tr>
<tr>
<td>19</td>
<td>235</td>
<td>219</td>
<td>16</td>
</tr>
<tr>
<td>20</td>
<td>216</td>
<td>177</td>
<td>39</td>
</tr>
<tr>
<td>21</td>
<td>192</td>
<td>177</td>
<td>15</td>
</tr>
<tr>
<td>22</td>
<td>198</td>
<td>184</td>
<td>14</td>
</tr>
<tr>
<td>23</td>
<td>278</td>
<td>260</td>
<td>18</td>
</tr>
<tr>
<td>24</td>
<td>166</td>
<td>165</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>207</td>
<td>207</td>
<td>10</td>
</tr>
<tr>
<td>26</td>
<td>311</td>
<td>280</td>
<td>31</td>
</tr>
<tr>
<td>27</td>
<td>261</td>
<td>249</td>
<td>12</td>
</tr>
<tr>
<td>28</td>
<td>135</td>
<td>131</td>
<td>4</td>
</tr>
<tr>
<td>29</td>
<td>261</td>
<td>255</td>
<td>6</td>
</tr>
<tr>
<td>30</td>
<td>272</td>
<td>238</td>
<td>34</td>
</tr>
<tr>
<td>31</td>
<td>304</td>
<td>238</td>
<td>66</td>
</tr>
<tr>
<td>32</td>
<td>263</td>
<td>234</td>
<td>29</td>
</tr>
<tr>
<td>33</td>
<td>333</td>
<td>208</td>
<td>125</td>
</tr>
<tr>
<td>34</td>
<td>257</td>
<td>234</td>
<td>23</td>
</tr>
</tbody>
</table>

| avg      | 235.5             | 211.6                        | 90.6                    |
| std      | 51.9              | 44.7                         | 9.6                     |

Appendices
Appendix B: Notations used in the flow diagram.

The FSM/SDL Dialect and Notation from the program OOTher is used.

The graphic language used is a simplified (standard) SDL.
Meaning of symbols follows:

**State**
The state symbol denotes a stable and reproducible situation in the finite automata.

**Received message**
A received message corresponds to a received information or signal. In OO languages it corresponds with a call to a method/service/member function, or reception of a letter.

**Sent message**
A sent message corresponds in OO languages to invoking a method/service/member function in a class, or sending of a letter.

**Synchronous message sending + reception**
It is an aggregation of send and a receive symbol, however, it is performed immediately without involving the state handling mechanisms, i.e. a synchronous inquiry (asking).

**Action/Processing**
The action symbol describes that some kind of processing is performed synchronously.

**Condition/Selection**
The condition symbol allows selection of alternative processing (if then/else or a switch/case block). Symbols following the condition symbol must be completed by a condition text meaning true/false.

**Iteration**
The iteration symbol denotes a repetitive processing. You may specify loop condition within the box. Everything to the corresponding next/label symbol is performed.

**Next iteration/label**
Denotes end of an iteration block. It may also be used as a label.
Appendix C: Instruction Manual for the Anaesthetists.

Manual for Nitroprusside bloodpressure controller

How to connect and startup:

1. Connect BNC-plug of Datex-monitoring system with arterial pressure to labmaster board.
2. Connect Imed 929 pump to PC through the serial port and put **pump on computer control**.
3. Switch on computer.
4. If the system does not start up automatically, go to C:\SNP_CTRL directory, type MAIN and use the <return> key afterwards.
5. Enter weight (1), SNP concentration (2), operating room (3) and go to main menu by typing (4).

Now first **CALIBRATE** the bloodpressure signal:

1. Look at the arterial pressure (2). Even if it seems OK **calibration is necessary**!
   So go out of the pressure window (any key).
2. **ZERO bloodpressure signal** on Datex-monitoring system.
3. Make pressure exactly 180 mmHg using the pneumatic pressure simulator.
4. While signal is 180 mmHg go to Calibration (3) and type "C". If counter is running for a few seconds release pressure and let it go to zero again at least once!
   Then the real pressure signal can be connected again.
5. When calibration is finished look at arterial pressure again (2).
6. If it looks OK switch to controller (1).
   **Check** in numerical pressure window if MAP, systolic and diastolic pressure are equal to the pressure values on the monitoring system (Remember that the controller system has a delay of 5 seconds!).

Now the system is ready to be used.

Although the function keys in the display are self explanatory they are listed below:

- **F1** Go TO **AUTO**matic control.
- **F2** Go TO **MANUAL** control.
- **F3** Increase SETPOINT for Mean Arterial Pressure (MAP).
- **F4** Decrease SETPOINT.
- **F6** Increase FLOW of Sodium NitroPrusside (SNP).
- **F7** Decrease FLOW.
- **F8** Make FLOW zero (in case of emergency).
- **F9** Go TO or come FROM Cardiac Pulmonary Bypass (CPB).
- <ctrl> **F10** Quit

In order to evaluate the controller there are also some keys to mark certain events which occur during the operation, they are listed below:

- [1] Skin incision.
How to use the MAP controller:

From startup you are in MANUAL mode.

In MANUAL mode the pressure is shown but no SNP flow is calculated.

How to go to AUTOMATIC Control:

1. First flush the line by manually inducing a slight flow on the infusion device.
2. The pressure can be controlled manually now by using the < F6 > and < F7 > keys.
3. The flow can be shut down completely with < F8 >, this is not necessary though.

Now the system is ready to go to AUTOMATIC Control with the < F1 > key.

The FLOW can NOT be adjusted in AUTO mode.

You can always RETURN TO MANUAL mode though (if necessary) by typing < F2 > and adjust the flow there.

Of course the SETPOINT can be adjusted in AUTOMATIC mode.

Cardiac Pulmonary Bypass (CPB):

When the operation is going into the perfusion state, this change has to be indicated to the controller with the < F9 > key. This is absolutely necessary to make a good blood pressure validation possible.

Normally the SETPOINT is lowered then too. At the end of perfusion this has to be undone again with < F9 > in order to return to normal validation.

ALARMS:

The system can generate some alarms. They are not documented here because that would be too extended for this short manual and they are always clearly explained in the message window together with the time of occurrence. One in particular will be mentioned though:

During the automatic run the system can automatically switch back to MANUAL control because of invalid measurement of the bloodpressure signal for a too long period (90 seconds). If this happens because of the taking of blood samples at that moment you can switch to AUTOMATIC mode again as soon as the measurements are valid again.

Finally the system can be SHUT DOWN by using the < CTRL > and < F10 > keys together.

NB. if the system is in AUTO mode it will first switch to MANUAL for safety reasons.

Then push < CTRL > < F10 > again if you really want to Quit.

To END the SESSION exit from the main menu (5).

P.S.: For evaluation purposes we ask you to indicate the different stages of the operation to the system using the numbers [ 1 ] to [ 6 ] as stated before and on the LABMASTER box. Please also answer the questions of the evaluation program at the end of the control session.