DISCRETE EVENT MODELING AND CONTROL OF THE NEONATAL RESUSCITATION PROGRAM ALGORITHM

ROSALIE M. WILSON

Discrete Event Modeling and Control of the Neonatal Resuscitation Program Algorithm

by

©Rosalie M Wilson

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of
Master of Engineering

Faculty of Engineering and Applied Science
MEMORIAL UNIVERSITY OF NEWFOUNDLAND

May 2006

ST.JOHN'S NEWFOUNDLAND

Your file Votre référence
ISBN: 978-0-494-30518-8
Our file Notre référence
ISBN: 978-0-494-30518-8

Discrete Event Modeling and Control of the Neonatal Resuscitation Program Algorithm

by

Rosalie M Wilson

Abstract

The Neonatal Resuscitation Program (NRP) is the accepted standard for training caregivers in the resuscitation of compromised newborn infants. The NRP introduces a need for resuscitation training and evaluation. ANAKIN is a simulation tool in which caregivers can practice their NRP skills, while allowing trainers to evaluate the student's NRP skills in a formalised manner.

This work models the NRP algorithm as well as a baby's vital signs, so that birth scenarios can be explored. Although the NRP is the accepted standard, modeling methods such as discrete event systems (DES) allow other practices to be explored, without causing harm to an infant.

This work is but a start in this area of research which brings together medical science and engineering. Some areas for future work are presented.

Table of Contents

List of Figures

List of Tables

Nomenclature

AAP American Academy of Pediatrics

ACoRN Acute Care of the Resuscitated Newborn

Bpm Beats per minute

CHD Coronary Heart Disease

CPR Cardiopulmonary Resuscitation

DES Discrete Event Systems

ECG Electrocardiogram

FSM Finite State Machine

GUI Graphical User Interface

INCA Instrumentation, Control and Automation

METI Medical Education Technologies Incorporated

NRP Neonatal Resuscitation Algorithm

OSV Observable State Variable

PDA Personal Digital Assistant

PDA Personal Digital Assistant

TETRA Telehealth and Educational Technology Resources Agency

TTM Timed Transition Machine

Chapter 1

Introduction

1.1 Anakin

Anakin is a joint project between Memorial University of Newfoundland and Labrador's Faculty of Medicine, Instrumentation, Control and Automation (INCA) Lab at the Faculty of Engineering and Applied Science, and TETRA (Telehealth and Educational Technology Resources Agency). The aim of the Anakin project is twofold.

1. To provide a user-friendly and portable learning tool for health care providers (called trainees), in particular those living in remote regions.

2. An evaluation tool enabling trainers to evaluate the resuscitation skills of their students.

The Anakin project has several components: there is a mannequin of a neonate, equipped with actuators that simulate the vital signs of an infant in need of resuscitation, including heartbeat and skin tone. The mannequin is also equipped with sensors whose purpose it is to detect the level of care that is being administered to the mannequin by the trainee. For example, sensors detect the degree of neck tilt, chest compression depth, and intubation depth.

Another component of the Anakin is the software that handles data received from the sensors, and allows the trainer to send information to the mannequin, such as

Figure 1-1: Anakin: Chest Compression Sensor Source: Aziz, K. (2002)

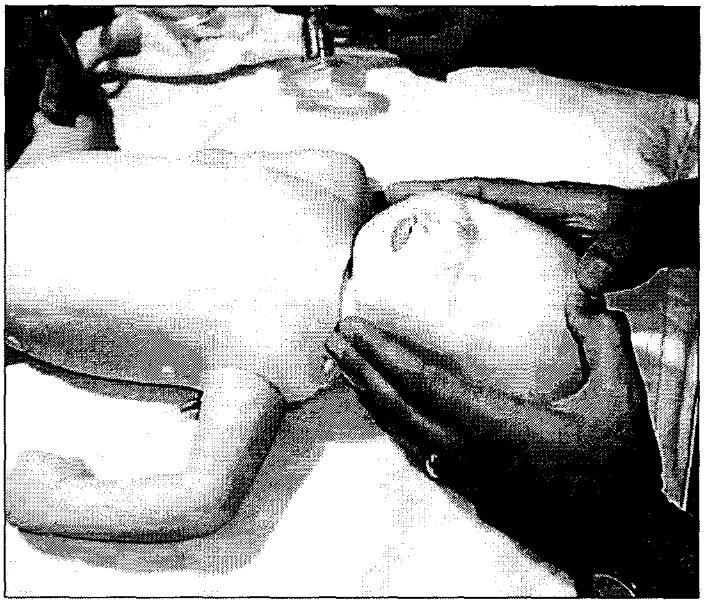

Figure 1-2: Anakin: Neck Tilt Sensor Source: Aziz, K. (2002)

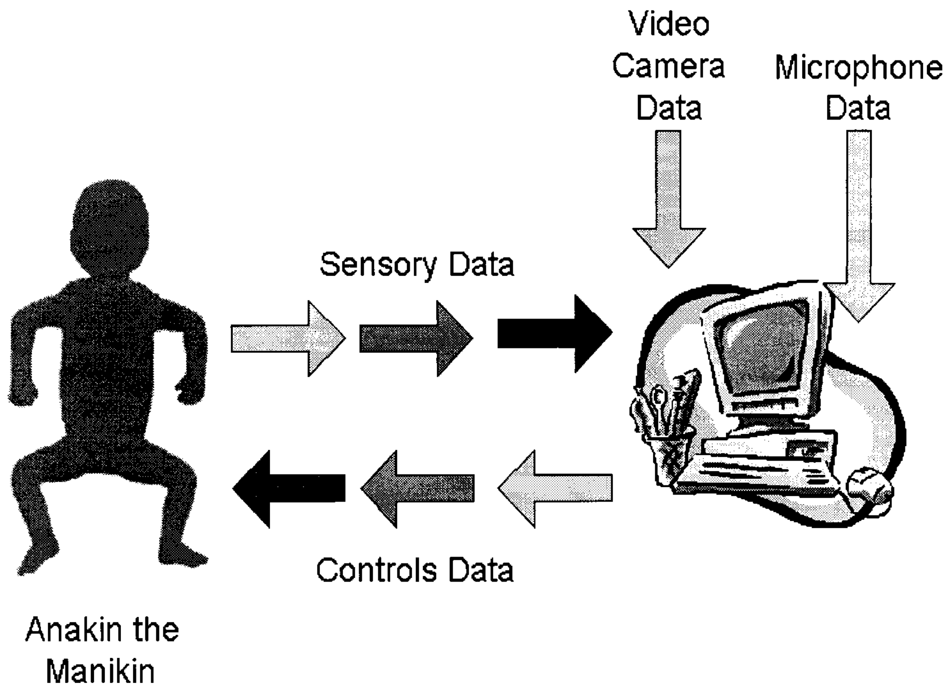

Figure 1-3: The Anakin System

increase heartbeat, or change skin tone. The flow of information to and from the Anakin is shown in figure 1-3. The trainer's workstation has a graphical user interface (GUI) with the Anakin's sensors and actuators. The GUI shows the trainee's interventions, according to the nationally accepted guidelines for neonatal resuscitation, and a video screen that allows the trainer to visually confirm the trainee's actions. Using an on-screen checklist the trainer (or trainee using playback) can objectively score the NRP provider's performance. The goal for the future software version is to be a self-evaluating tool, with the use of a Knowledge Based Engine.

The Anakin will be pre-programmed with a number of scenarios of increasing difficulty to test the providers' abilities under a variety of circumstances. This involves changing the Anakin's vital signs and response to resuscitation, which in real life would vary from case to case. Clinical tests have demonstrated that each scenario should not exceed 15 minutes. The Anakin responds to the correct (or in-correct) trainee responses, by changing its vital signs such as heartbeat, or by giving an audiovisual prompt, allowing an opportunity for the trainee to give corrective intervention. This method of learning is in keeping with most critical care programs.

The unique feature of this method of learning/training is that it simulates a real-life situation, while allowing trainees to learn from mistakes and reinforce skills. Performing simulated resuscitations can be more enjoyable than traditional classroom training, particularly when trainees can score themselves. Trainees can receive expert opinion immediately from trainers, or at a later stage. This is a powerful tool in giving neonatal resuscitation providers confidence in this life-saving skill, not to mention proficiency and skill retention.

As a part of the Anakin's trial usage in Newfoundland, NRP training sessions using Anakin simulators will occur 4 to 6 times in two years for providers in selected centres (conditional on funding). The NRP provider/trainee will be booked to attend a one-hour session with the simulator, either on-line or off-line, during which 2 or 3 scenarios will be presented. The provider's skills will be assessed by a distant instructor, and the score will be logged in their personal file. Performance logging may later be used to register providers who complete prerequisite hours.

1.1.1 The Portable Training Device

The trainer's GUI can reside on either a stationary PC, or a PDA such as a Palm or a Pocket PC. The latter is portable, thus enabling the trainer to move freely about the training room while the resuscitation procedure is being performed, whilst the former forces the trainer to remain relatively stationary behind a workstation. The PDA allows the trainer to get a better view of the neonate and trainee, and encourages personal interaction with the students. This "closeness" may result in a more accurate evaluation of the student, though this must be studied in trials before one can be certain. The author has created a PDA application for student and/or self evaluation in 2003 [41]. This application runs on Palm OS devices. Work is currently underway to create a similar application for Pocket PC devices, which use the Windows Mobile operating system.

Users of the portable training device do not require any specialized computer training, other than familiarity with operating a PDA. The Palm version of the NRP checklists is launched on the PDA with a single click, similar to any other Palm OS application. The PDA is Bluetooth enabled with a Red-M Blade Handspring Module that plugs into the Handspring's Springboard Expansion Slot. The PDA connects via Bluetooth to a laptop or PC running an application that receives and parses bytes from the communication port. The computer may be Bluetooth enabled with the 3Com USB Bluetooth Adapter. This application can in turn send messages over RS232 to the Anakin. All messages are one byte in length. The Palm OS application is interrupt driven, "waking up" whenever one or more bytes are at the Bluetooth Virtual Serial port. The PC application polls the Bluetooth Virtual Serial port every 50 ms.

The Palm OS application replicates the checklists that are used by trainers during a neonatal resuscitation training procedure. It has a total of 25 forms. The user can navigate through the forms by selecting 'Previous' and 'Next' buttons, or by selecting items from the menu bar. The trainer makes selections in the checklists by clicking on popup triggers, which display popup lists. The trainer's selections are stored to global variables. When the procedure is complete, a record is created that preserves the state of all the global variables, and the record is stored to a database. Each session creates a new record that is stored to the database.

1.2 Neonatal Resuscitation Program

The NRP is an education program that helps health care providers to learn the skills necessary to resuscitate newborns. The NRP was developed in 1987 by the American Academy of Pediatrics (AAP) and the American Heart Association to help reduce the

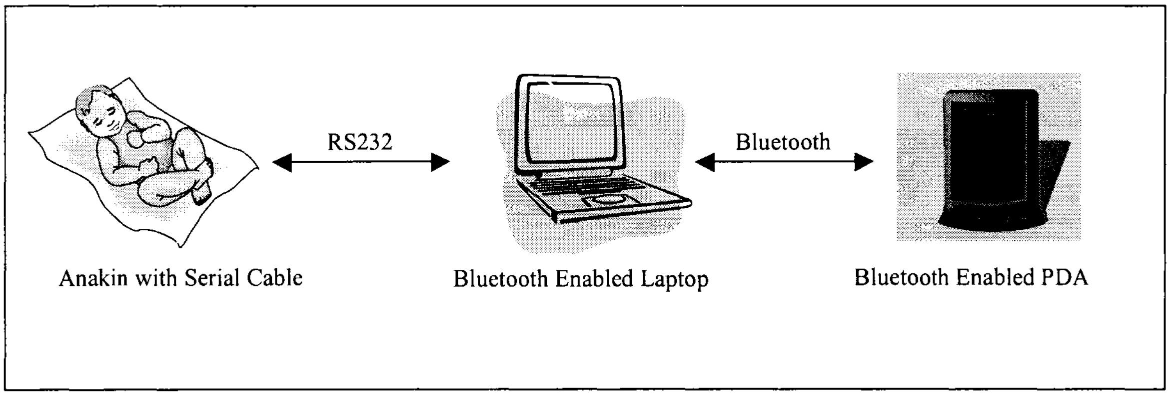

Figure 1-4: Network Configuration for the Portable GUI

incidence of neonatal asphyxia and death [1]. It is endorsed by the Canadian Pediatric Society, Health Canada, and a number of professional organizations in Canada. It is also distributed in every province by the national and provincial Heart and Stroke Foundations or outreach educator programs. (In British Columbia, Ontario and Nova Scotia, The Heart and Stroke Foundations are not NRP organizations.) Canada has a well-established network of regional and institutional NRP instructors, with complete coverage in Newfoundland and Labrador. Today, the NRP is internationally recognized, with over 25, 000 instructors and over 1.5 million health care providers [19]. In Canada there are over 10,000 NRP healthcare providers distributed throughout approximately 500 institutions [10].

More than 90% of all newborns require little to no aid during the transition from the uterus to the outside world. The remaining 5% to 10% of newborns require some form of resuscitation at birth [1]. This could be as simple as tactile stimulation such as rubbing the feet, or as involved as positive pressure ventilation and the administration of epinephrine. Within minutes, a newborn can suffer irreparable damage or even death. It is believed that annually over 900,000 infant lives worldwide could be saved from death due to asphyxia with the administration of simple airway or breathing interventions [19]. NRP targets this critical window of opportunity, when every second is vital. It details specific steps and timing that a caregiver should follow to achieve maximum health benefit. The resuscitation of a newborn infant comes with challenges that are unique, compared to the resuscitation of an adult, or child. This uniqueness arises from several factors, including the petite size of the infant, its delicate state, and its abrupt departure from the liquid-filled environment of the uterus to the gas-filled environment on earth where breathing air is suddenly required.

The NRP algorithm has undergone changes over the years, and is constantly being evaluated and updated when new information arises. The NRP guidelines referred to in this work are the result of recommendations from the International Guidelines 2000 Conference on Cardiopulmonary Resuscitation and Emergency Cardiovascular Care [19]. (The newest NRP guidelines have been published in December of 2005.) This conference assembled professionals from a variety of relevant fields, including neonatal resuscitation. Through a process of evidence evaluation and using the most current available scientific data, the guidelines for NRP were updated. The evidence came from a variety of study types, including controlled trials, prospective and retrospective observational studies, randomized trials, animal studies, etc. Recommendations were classified in a range from Class I (Always acceptable, proven safe, definitely useful) to Class III (Unacceptable, no documented benefit, may be harmful).

Although the NRP guidelines are recommended, they are not actually forced upon any individual or organization. Its use is voluntary, and is thought to provide the best practices for neonatal resuscitation [1]. In Canada, NRP is the accepted standard for educating neonatal caregivers.

1.2.1 Anakin as an NRP Training/Evaluation Tool

As stated earlier, one of the aims of the Anakin project is the development of an NRP training/evaluation tool. This aim can be further subdivided into three categories:

1. Face to face, trainer led NRP skills training. This could take the form of a

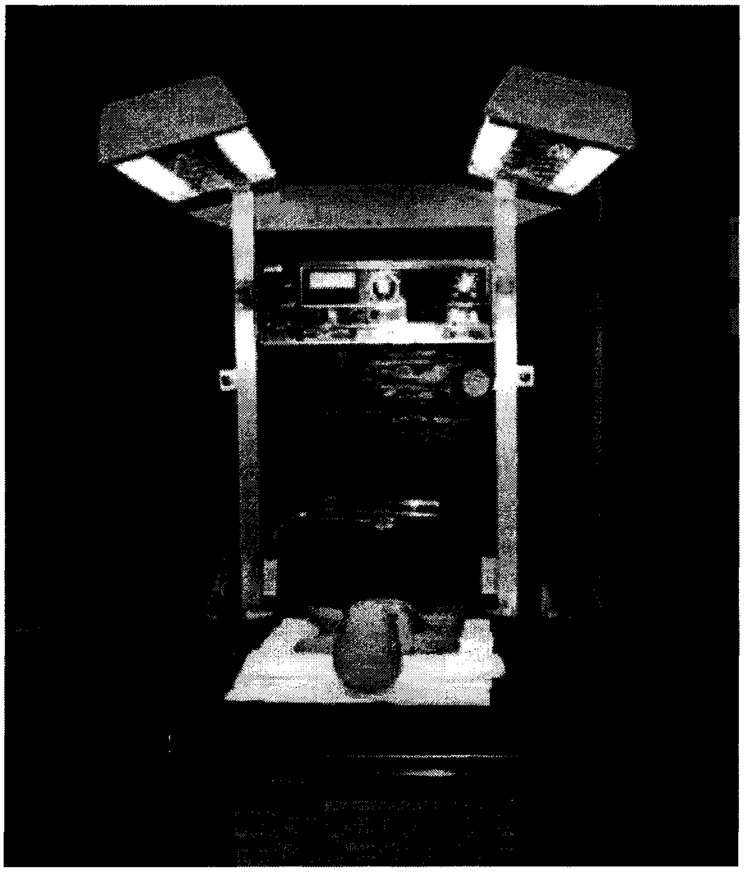

Figure 1-5: Anakin on a Neonatal Workstation

classroom setting. Anakin software can run on a portable Personal Digital Assistant (PDA), equipped with wireless Bluetooth technology. Wireless technology gives the trainer greater freedom to move about the classroom setting without the limitation of wires or the need for a direct line of site between devices. One possible wireless configuration enables the trainer to communicate between a PDA and a PC using Bluetooth. The PC is connected via serial cable to the mannequin. For a truly wireless experience, a Bluetooth chip can be directly embedded into the mannequin allowing direct communication with the PDA. A wireless solution has yet to be realized.

2. Remote NRP skills training. As a remote distributed training system, a trainer provides training, skills-assessment and feedback to a trainee situated at a remote site. Using the Anakin's software and a network link, the trainer can control the mannequin's actuators while having access to its sensory data.

3. Self-directed learning. As a self-directed NRP training tool, the Anakin will function as a stand-alone system. Trainees will perform the NRP on the mannequin, which will simulate pre-programmed scenarios. The Anakin system will store the trainee's actions in a database, which can be viewed and scored against standardized NRP performance criteria. The Anakin can be used at anytime, and as often as the user wishes.

1.3 How Simulation Training Meets Today's Challenges

One of the challenges with the NRP is acutely felt in rural areas. It is recommended that neonatal workers have their NRP skills updated at least once every two years, and every hospital or health care corporation is responsible for determining the competence and qualifications of its own neonatal caregivers. NRP practitioners must be tested by qualified NRP trainers. Quite often, NRP trainers are located in larger centers, such as St. John's. As an example, a trainee working in a region where there is no NRP trainer must travel to St. John's to undergo testing, which causes numerous logistical problems, and heavy financial costs to the healthcare system. The trainee must find a suitable replacement for the duration of their training in St. John's. In addition, in rural areas where birth rates are low, the NRP is seldom used and hence skill retention suffers [37]. A portable tool that can be shipped periodically to various regions in the province would alleviate these difficulties, by allowing the trainee to either practice NRP on their own, or to connect with an experienced trainer via a communications network.

More generally, medical education today has its challenges, and these challenges may be better met with the enhanced training provided by medical simulators such as the Anakin [18]. One of the challenges is making the jump from textbook to practice in a safe manner[20]. When direct experimentation is not possible, due to economic, ethical or other reasons, simulation training can be an excellent alternative to the real thing. It allows the user to study cause and effect relationships in a safe environment where there is no risk of injury [17]. One of the benefits of simulation training is that time can be expanded or sped up[39]. The former allows for a more thorough examination of the task at hand, while the latter allows one to analyze a process that may take hours or days in real life in only a few seconds.

Another advantage of simulation training is that it allows one to exhaustively test changes to a system before implementing them in real life [38]. This will be referred to as validation. It is useful because in many instances, once changes are implemented to a real world system, reversing or correcting these changes can prove to be very costly in terms of money, and even perhaps human life. In some cases, disrupting a real world system to test changes may not even be possible.

Skills such as handling a newborn infant, and providing effective chest compressions at the correct depth cannot be learned from a textbook alone, but require hands-on training. Medical simulators help bridge the gap between the textbook and actual practice[18]. Some other industries that have successfully used simulation training include the airline industry (i.e. flight simulation) and the military (i.e. training tank personnel, infantrymen, etc.). The lure of simulation training is that users can practice over and over, until they feel comfortable with a task. Simulators can also simulate scenarios of increasing complexity as the user's skills progress[16]. It is interesting to note that among the earliest patient simulators were animal models. Even to some degree today, animals are used as patient simulators[14].

1.4 ACoRN

Another potential use of the Anakin system would the extension of its role from teaching newborn resuscitation to teaching stabilization (also known as "post-resuscitation"). Stabilization requires a greater variety of treatment algorithms than resuscitation, with greater complexity of diagnoses, investigations and therapies. ACoRN (Acute Care of at-Risk Newborns) is a new neonatal program, sponsored (in part) by Health Canada, and was unveiled at the Canadian Pediatric Society annual meeting in Calgary in June 2003. The ACoRN algorithms follow those in NRP and may be ideally suited to seamless adaptation into Anakin software.

The ACoRN algorithms have been available since February 2005. The following flowchart gives an example of a stabilization algorithm that details the management of an infant who appears shocked after resuscitation (NRP) has been completed.

1. Does the neonate appear shocked? (if yes, go to question 2; if no, go to next algorithm)

2. What is the blood pressure? (if it is low go to question 3; if not, go to the "causes of shock" subroutine)

3. Do you have intravenous access? (if yes, go to question 4; if no, go to the "no intravenous access" sub-routine)

4. Initiate treatment with a volume expander, given over 15 minutes

5. Is the blood pressure now normal (if yes, go to "further investigations for

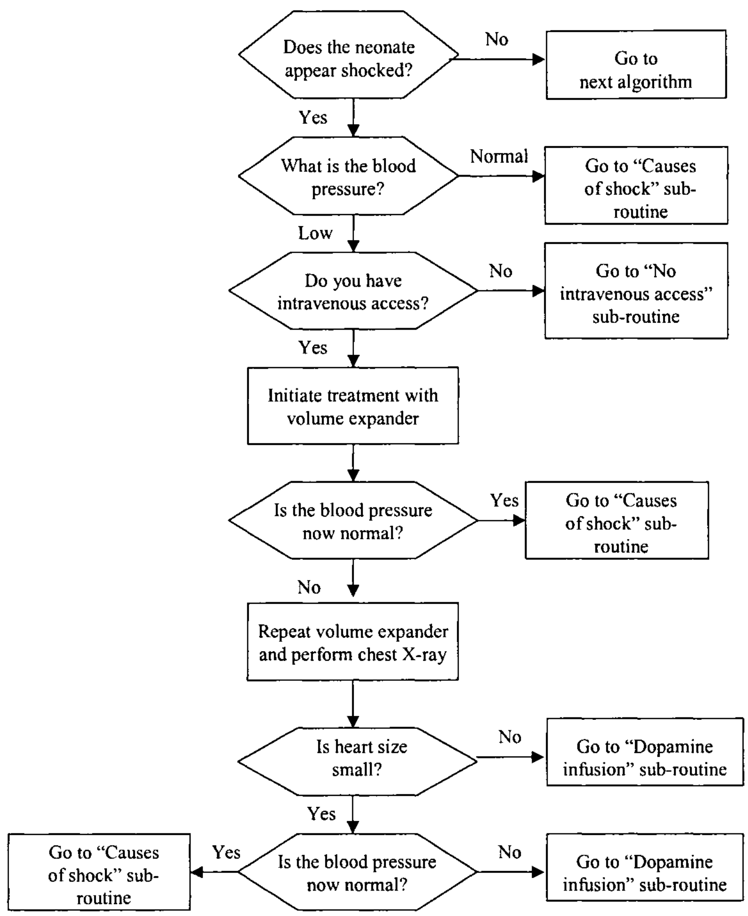

Figure 1-6: The ACoRN Algorithm for an Infant Who Appears Shocked

shock" subroutine; if no, got to question 6).

6. Repeat volume expander over 15 minutes and perform chest X-ray.

7. Is the heart size abnormal on chest X-ray? (if small, go to question 8; if normal or enlarged, go to question 10)

8. Is the blood pressure now normal (if yes, go to the "causes of shock" subroutine; if no, go to #10).

9. The heart is small on chest X-ray and the blood pressure is low in a shocked neonate: this suggests low circulating blood volume so consider alternative volume expander and go to question 10

10. Start medical treatment for shock (dopamine) according to "dopamine infusion" sub-routine.

11. Consult specialist and attend to other unstable states (e.g. breathing difficulties, low blood sugar,. . . . . .)

Thus a provider can train in the management of an individual or multiple conditions that might arise following satisfactory completion of an NRP training exercise, more closely mimicking the complexity of real life and neonatal disease.

1.5 The NRP Algorithm (MEGACODE)

The American Academy of Pediatrics has created resuscitation flow diagrams that describes the neonatal resuscitation procedures, which are recreated in figures 1-7 and 1-8 [1]. For evaluation purposes, a number of checklists have been created, which allow the trainer to assess the student's resuscitation capabilities. The checklists correspond to the main categories of neonatal resuscitation. The neonatal resuscitation checklists used for the Anakin are separated into two main scenarios, which will determine the care that needs to be administered:

1. The neonate is preterm and no meconium is present.
2. The neonate is preterm and meconium is present.

The checklists for the "preterm, no meconium" scenario are divided into the headings as follows:

1. Initial Steps
2. Provides and Verifies Adequate Ventilation
3. Chest Compressions
4. Administers Epinephrine

The meconium scenario has its own unique checklist, which is completed in addition to the preterm checklists above. Each of these six main headings is further subdivided into individual checklists, or forms. Individual questions in each checklist are answered with a "Yes", "No" or "OK" response. "Yes" is selected when the action has been correctly completed in a timely manner. "No" is selected when the trainee failed to complete the action, or the action was not done correctly. "OK" is selected when the action was completed satisfactorily, out of sequence, or only after being prompted.

From the point of view of the NRP provider, their assessment of the situation must be timely. Some of the things they must be watchful for include the presence of meconium, breathing, heart rate, skin tone and muscle tone. If all of these factors are deemed normal, then the infant is provided with routine care, which involves placing the baby on the mother's chest and drying with cloths. If the rapid assessment has found something abnormal, such as a low heart rate, then the initial steps are commenced. Following the initial steps, the need for subsequent intervention is based upon the following three vital signs.

1. Breathing. Is it spontaneous or not spontaneous?

2. Heart rate. Is it greater than 100 bpm, between 60 bpm and 100 bpm or is it less than 60 bpm?

3. Skin tone. Is it blue, pink, or is there evidence of peripheral cyanosis?

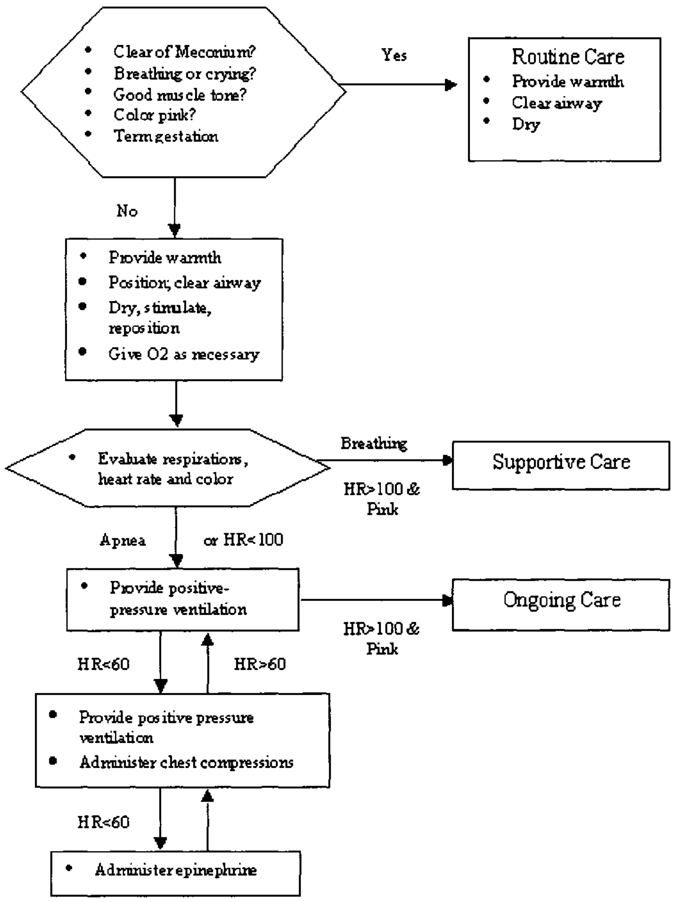

Figure 1-7: The NRP Flowchart

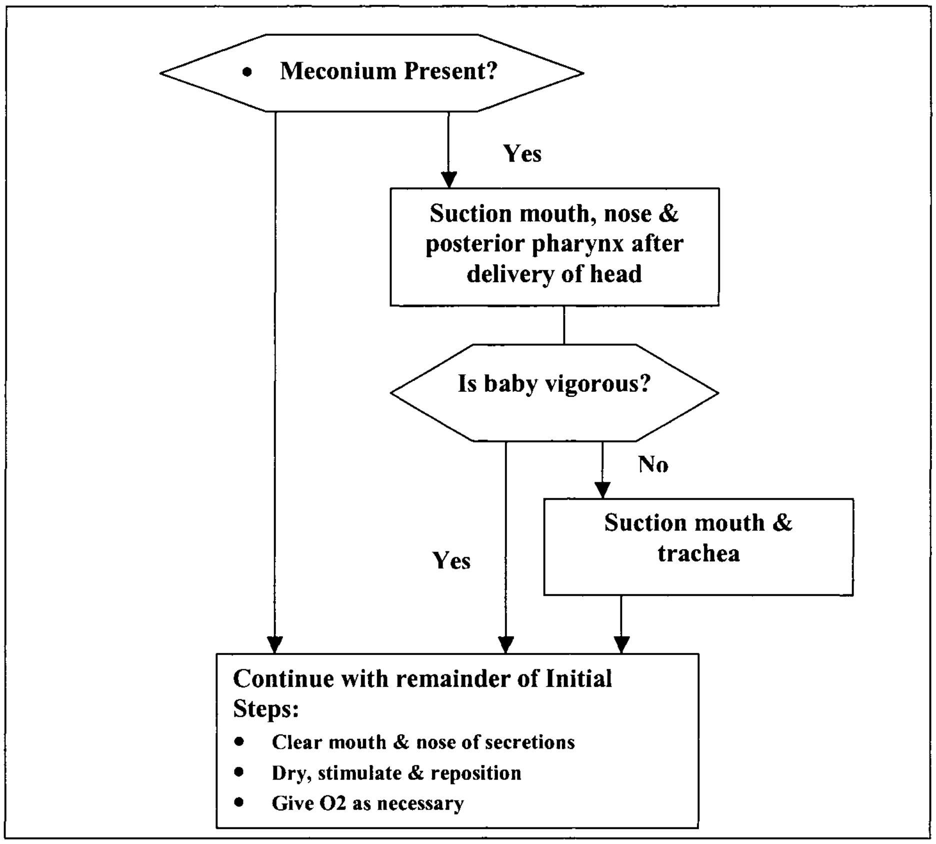

Figure 1-8: Flowchart for Meconium

1.6 Problem Statement

The intent of this work is to develop a series of models (both non-timed and timed state machines) that can be used to provide verification of the neonatal resuscitation procedure. The models will be divided into two main areas:

1. The neonate's vital signs and statistics related to resuscitation.
2. The resuscitation procedure itself, which can be subdivided into smaller procedures for ease of use and computation.

Once a set of models have been developed, the theory of DES will be used to prove or verify the safety of several resuscitation scenarios.

Chapter 2

Literature Review

A review of the literature is presented in the following three areas:

1. Discrete event modeling and simulation in the health care industry.
2. Simulation tools used for resuscitation training.
3. An evaluation of the NRP and student skill retention.

The purpose of examining the use of discrete event modeling and simulation in the overall health care industry is to get an idea of the inroads this theory has made in healthcare. It was also to see if anything similar has been done in the area of neonatal resuscitation. Other methods of simulation modeling used in health care, such as Markov chains are also examined briefly.

Various simulation tools that are currently used in the area of resuscitation training and evaluation are examined, with a focus on those specific to newborns and infants. Lastly, the NRP training program is examined in an attempt to compare the effectiveness of simulation training to traditional lectures when measuring skill acquisition, retention, and student proficiency.

The use of models for the purposes of testing or validation is frequent in the health care system. The model can take many forms, from a discrete event model to a mannequin. The problem with this form of validation is that it does not provide verification. Certification usually involves running through various scenarios on the model, but one cannot conclude with 100% certainty that every possible scenario has been tested. If for example, one is testing for safety, the only way to conclude with absolute certainty that the system is safe is through verification. Verification is a formal proof of a certain property. It can be compared to a mathematical proof, which in a formal way, proves with 100% certainty that a mathematical theorem holds true. Depending upon the system, whether it be continuous, discrete or a combination of the two (i.e. hybrid), one can use the theories of continuous time systems, discrete event systems or hybrid systems respectively to perform verification. The use of hybrid systems for verification is still very new and requires further research. The work presented here uses the theory of discrete event systems, described in chapter 3, to perform verification on a small part of the neonatal resuscitation procedure (chapter 5).

2.1 Simulation Modeling in the Health Care System

The healthcare industry has used discrete event modeling as an analytical tool. There are several examples in the literature where a discrete event simulation model of the physician clinic environment has been used to provide a risk free method of evaluating current practices. "Risk free" are the key words. The model allows one to analyze the system over the long term, without putting people's lives at risk, nor expending enormous financial costs, etc. Some areas of interest would be patient waiting time, admissions policies, patient scheduling and admissions, and physician workload, to name a few. By evaluating the clinical environment, recommendations can be made to provide higher quality care, in a cost effective manner. This was explored in [39], using an object oriented, discrete-event visual simulation environment of a physician clinic environment. The object oriented nature of the model allows objects to be reused, and easily maintained. Some of the objects that are modeled include the physical layout of the clinic (i.e. waiting room, registration area, physician office area, etc.), the clinic's human resources (i.e. physicians, nurses, assistants, lab technicians, etc.), and patients (subdivided into different patient levels, depending upon the amount of physician time each requires). It is noted in [39] that since the model is only an abstraction of a real world system, it cannot be deemed 100% accurate or realistic, and as a result, cannot be verified or absolutely validated. The model may however be sufficiently accurate for its intended purpose.

In [38], a discrete event simulation model of the hospital-based teaching ambulatory clinic setting was created using computer simulation. These clinics provide care to patients while at the same time provide training to new physicians. This simulation model takes into account waiting time, flow time, and competition for resources, in an effort to optimize clinic management and financial performance. Discrete event modeling was chosen as the simulation method as it lends itself well to capturing process flow data and competition for resources. The model was developed using a discrete event simulation language called SIMAN. The outcome of the simulation was a trainee-teacher ratio that optimizes clinic operation while minimizing costs.

In a more specific application, discrete event simulation has been used to model coronary heart disease in [2]. The model is based on an individual's risk factors, and on probability distribution data from a coronary heart disease study. Risk factors include age, sex, and medical history. By changing an individual's risk factors, one can analyze the disease's effect on health care resources. A similar study was described in [8].

Discrete event simulation was used to model a liver allocation system in [36]. By simulating this system, analysts gain information about the number of organs that are wasted, patient survival rates, financial costs, etc.

It is suggested in [3] that discrete event modeling is well suited to modeling the interaction between individuals. This need arises when modeling infectious diseases, where a person's risk of infection is based on the number of people who are already infected, or when the course of treatment for a patient depends upon the number of patients already receiving treatment. The latter example arises when resources are limited. In [3], it is explained that discrete event modeling allows for a patient's past history to be accounted for (unlike the Markov models), and it can model the interaction between specific patients. The downfall of this method, as cited by [3] is that discrete event modeling requires specific software to cope with the complexity of the models, and that analysis of these models is labour intensive and time consuming. Thus, discrete event modeling is richer than Markov models, yet requires longer running times and more computing power.

In [2], a "discrete-event micro-similation model" is being developed to determine when an individual may succumb to coronary heart disease (CHD). In the model, individuals have a set of attributes that define their risk profile and disease status. Examples of risk factors include smoking, high blood pressure, cholesterol, physical inactivity, etc. Events in the model could be changes in risk factors, or changes in disease status. Based on probability distributions, a set of times to CHD events can be determined. If risk factors should change, then a new probability distribution is used and new times to CHD events are generated, while the old ones are removed. To run the model, users use a set of PASCAL simulation routines that were developed specifically for health care systems. The simulation runs until the patient's death, or when the patient reaches the age of 85, whichever occurs first. This work is useful not only in projecting when a patient may succumb to CHD given his or her risk factors, but can also be used as a basis for estimating health care costs associated with CHD.

Markov models have been used in [7] to model the progress of HIV infection and AIDS. In this article, a finite number of states is used to describe the varying levels of disease (two states signifying two different levels of HIV infection, one state for AIDS, and one state for death). Patients are in one of these states at any one time, and they may remain there or move to a different state. Clearly, once a patient reaches the dead state, they must remain in that state. Probabilities are assigned to each ordered pair of states, signifying the chance that the patient will be in the second state at the end of one time cycle. A fixed time cycle is used, which is one year. It is interesting to note that the probability of transition depends exclusively upon the state that the patient is in at the beginning of the year (i.e. the initial condition), rather than taking into account the amount of time a patient has been in a particular state or the previous states of the patient in previous time cycles. Thus, it does not account for the patient's past history. This is known as the Markov assumption, or homogeneity assumption, and illustrates one of the limitations of using a Markov model.

2.2 A Review of Current Neonatal Simulation Tools

Since neonatal resuscitation is such a critical process, the simulation training tools ought to be as realistic as possible. Ease of use and transportability are two other factors that may be of importance when choosing a medical simulation tool. When the Anakin project commenced in 2001, the premise behind the project was a novel one. There were no similar newborn-sized mannequins in the marketplace that were instrumented with sensors and actuators for the purpose of NRP training and evaluation. At the time, a life like neonate mannequin that would automatically react to the level of care being provided did not exist, nor did a real life training tool for NRP trainers that offered an in depth method of student performance evaluation.

A search at the time in the medical literature and a survey of the leading medical equipment providers did not reveal a comparable NRP training tool to be in existence. There were however, several companies producing full size instrumented mannequins. One of these companies is the Florida based firm, Medical Education Technologies Incorporated (METI). It produces the Human Patient Simulator, which is driven by a computer model, and exhibits lifelike qualities such as blinking, breathing, and speech. In addition, it has a heartbeat and pulse, and mimics a human's response to cardiopulmonary resuscitation (CPR), medication, ventilation, intubation, etc. Therefore, it was clear that there were very sophisticated medical training mannequins in existence, but a similarly sophisticated neonate mannequin had yet to be produced. Instead, medical schools were training their students using a passive plastic doll that has no ability to react to a trainee's performance, nor to exhibit realistic vital signs. A leading manufacturer of the passive doll used in medical training is Laerdal Medical Corporation [23]. This doll is in use even today in many institutions.

Currently, METI has addressed the lacking neonate mannequin market, and was slated to release BabySim, in the first quarter of 2005. It boasts realistic touch and feel, clenched fists, and a layer of baby fat for life likeness, while students can practice CPR, defibrillation, the administration of drugs, etc. Previous to this, METI had released PediaSim, which replicates a 7 year old boy, and was the first portable simulated pediatric patient in the United States. PediaSim has a life like airway, pulmonary, cardiovascular, metabolic and neurological systems, in addition to visual and audible breathing, heart rate, dilating pupils, a tongue that can swell, etc. In short, it is a very sophisticated child like mannequin. PediaSim is driven by a computer with the use of mathematical models of a child's physiology and pharmacology. Using these sophisticated models, the system is able to react to critical injuries and the level of care being provided automatically [27].

A neonatal simulator currently on the market is PEDI Blue by Gaumard. PEDI Blue is the size of a newborn during its first 28 days of life. Medical trainers can practice intubation and CPR on the doll, while the doll can exhibit both peripheral or central cyanosis, depending on the setting chosen by the instructor. PEDI Blue is unique in its ability to exhibit a realistic skin color, ranging from pink for a lively, healthy neonate, to blue for a neonate who is in need of oxygen. The color can be set in advance, but will also change depending on the level of care being provided. This is detected by a Code Blue monitor, which has a small computer. It detects ventilations and compressions, and can determine whether these meet CPR standards [15].

In [22], a prototype system is described for neonatal resuscitation training that uses a virtual environment built with VRML and Java. There is a virtual newborn infant that displays several vital signs including heart rate, respiration rate, skin tone, and movement. Each of these vital signs is assigned its own observable state variable (OSV), each with its own range of values. The trainer is able to control all of the OSVs using a Java dialog box with sliders. The trainer can also load predefined discrete event scenarios, or scenarios can load automatically in reaction to a trainee's actions. This is to make the job of the trainer less labour intensive.

A simulation-based training program in neonatal resuscitation called NeoSim is described in [18]. It is used to train and assess NRP providers. The components of the NeoSim simulator include a neonatal mannequin and a computer program that allows the mannequin's heart rate and skin color to be changed remotely. The mannequin is life like, possessing an airway that can be intubated, inflatable lungs, as well as an umbilical cord.

A patent database search conducted by the Genesis Group at Memorial University has uncovered two relevant patents, including US patents 4611998 and 5509810 [32] [34]. The former dates back to September 16, 1986 and is described as a "Simulator for Teaching Neonatal Resuscitation", while the latter dates back to April 23, 1996 and is described as an "Interactive Neonatal Resuscitation Training Simulator".

The "Interactive Neonatal Resuscitation Training Simulator" is an interactive training simulator that uses a life-like mannequin that looks and responds like a neonate. It can simulate the condition of needing resuscitation and detects, evaluates and responds to a caregiver's activity. It is described as having an "intelligent interactive control mechanism" that responds to the level of care being provided based on predefined scenarios. As part of this simulator, there is a neonatal workstation that uses virtual reality to simulate a real life hospital setting.

The "Simulator for Teaching Neonatal Resuscitation" has a life-like mannequin of an infant that is equipped with a lung simulator, and a pressure switch transducer that produces an electrical signal, which can be used to determine the rate and pressure of ventilations. In addition, it has a plunger mechanism and a potentiometer that work together to determine the rate and depth of chest compressions. The data is sent to a recorder so that it may analyzed, thus evaluating a trainee's performance.

In [35], a portable patient simulator for resuscitation training is described. This tool is not directed at neonates, but it is interesting to note as it uses a mobile, PDA-based ECG simulator. One of its features is that it can detect events, and automatically reacts according to preprogrammed scenarios.

It is interesting to note that in all of the above training tools, there is no mention of the NRP, and therefore there is still an aspect of novelty to the Anakin, however slight, since the anticipated software system for Anakin is based specifically on the NRP guidelines. Since NRP is the widely accepted standard, the others make use of it indirectly .

2.3 Evaluation of NRP Skills Retention

The neonatal care provided in the first few minutes of a neonate's life can have lifelong consequences to the health of the neonate [33]. The percentage of newborns requiring resuscitation is approximately 6 percent. For newborns weighing less than 1500 grams, this percentage dramatically increases to 80 percent. Studies have shown that NRP training is effective in improving the immediate post-training resuscitation knowledge and skills of trainees [37][13][11]. While is has been shown that most trainees can learn the skills required to perform NRP, knowledge retention and practical skill retention have been shown to be two very different things. Knowledge retention declines at a much slower rate than practical skill retention, which tends to decrease substantially faster according to [13] and [24]. It is often assumed incorrectly that the frequent use of NRP skills in real-life emergency situations improves skill retention, however the existing evidence shows that this is not necessarily the case. It is believed that this is not the case because performing the NRP procedure without feedback and analysis of performance does not help skills retention. In fact, one could surmise that without proper training and feedback, "bad" skills may be reinforced and retained. It has also been shown that the completion of a certified NRP training course is not a guarantee that protocols will be followed as they were taught. In [5] it was found that as the resuscitation procedure increased in complexity, it was more likely that the NRP protocol was not performed correctly.

Poor skills retention may be due to inadequate instructional and practice time, thus trainees never effectively learn the proper skills in the first place. Patterns that are complex, or that offer the learner many choices for action, such as the NRP, require more time to learn than may be given in training programs. One of the strategies identified by skills learning theory to improve the retention of resuscitation skills is practice. In order to master a skill and retain it, practice is essential. Practice should take place during initial learning as well as afterwards in order to keep the skill. Another important factor in skill learning is that the learner needs sensory input from the task and feedback about their performance to reinforce meaningful skills. Some types of feedback are believed to be more effective than others in resuscitation skills training. One of these is simultaneous feedback, which does not wait until the completion of a task before feedback is given. The use of a simulated resuscitation mannequin such as the Anakin can provide simultaneous feedback to the user through visual or audio cues, or ongoing interaction with an instructor [10].

The use of resuscitation training mannequins also makes NRP training more realistic. The more realistic the mannequin, the more realistic the training. By practicing on a mannequin, there is absolutely no risk or health threat, which [17] found to be one of the most positive aspects of real life patient simulators. In [18], simulation-based training is twofold. It requires both a realistic scenario and a realistic environment for the trainee. The environment is provided by the simulator itself. Simulation training provides the option for trainees to practice as often as they like (repetition), whenever they like. This allows them to experience numerous lifelike clinical situations in a short period of time, which would otherwise not be possible.

In [9], the use of videoconferencing technology as a tool to assess the skills of rural NRP providers is examined, which has parallels to the Anakin project's use in rural Newfoundland. Results showed that there was no significant difference between the performance of those who were assessed locally or remotely, and that the remote students enjoyed the experience of having a remote instructor who carried out the assessment.

In another study, the effectiveness of training using a computer-based simulator is compared to the effectiveness of training in a traditional seminar style[16]. Both the simulator and seminar approaches were found to result in better student performance, and interestingly, there was no significant difference found between the two groups. This is encouraging in that at a minimum, the newer simulation based training is as good as traditional training.

A preliminary study at Memorial University of Newfoundland and Labrador involving the beta version of the Anakin was conducted to investigate its effectiveness for enhancing and the retention of neonatal resuscitation skills [10]. The Anakin was found to empower students by providing an independent way of learning, which in turn increased the student's confidence levels in their resuscitation abilities.

Chapter 3

Discrete Event Systems

3.1 Background

Simulation modeling using discrete event systems allows one to test and explore a system. This form of testing is known as validation. It is a good alternative to direct experimentation, particularly where such experimentation would be difficult to perform, or not feasible due to practical, financial or ethical concerns. Time can be "compressed", allowing analysis of the effect of events on the system that in reality would occur over long periods of time, but in the computer simulation can be done in a matter of minutes. This is particularly useful in large systems. Discrete event simulation helps diagnose problems, and allows the study of cause and effect relationships. Discrete event simulation is an ideal tool in NRP training, where trainees can learn from their mistakes in a safe environment where the lives of infants are not threatened. It also allows one to test new procedures risk free, at a low cost and with no disruption to the real life system. These are just a few of the benefits offered by discrete event simulation.

This work takes validation one step further, to verification. Verification is achieved through the use of DES, the basis of which follows.

3.1.1 Finite State Machines

A model of the DES will consist of a finite state machine (FSM). Other methods of representing DES include Petri Nets [28] and Markov Chains [12]. These methods will not be explored in this work because Petri Nets can be modeled as FSM, and Markov Chains provide only statistical properties, and not exact past histories. The following definitions are based on [6]. A deterministic FSM model:

$G = (X, \Sigma, \Delta, x_0, X_m)$

where:

X is a finite set of states,

Σ is a set of events,

Δ is the set of labeled transitions,

x_0 is the initial state, and

X_m is the set of marker states.

A control technology can be defined by controllable events, and uncontrollable events, i.e. $\Sigma = \Sigma_{uc} \cup \Sigma_c$. Controllable events can be prevented from occurring at any time, but one has no control over uncontrollable events, hence the name.

Let Σ^* be the set of all finite strings in Σ . A labeled transition is a triplet (x, σ, x') with $x, x' \in X$ and $\sigma \in \Sigma$.

Definition 1 *The symbol $f(x, \sigma)$ is defined if $\exists\ x' \in X$ and $(x, \sigma, x') \in \Delta$.*

Let $s = \sigma_1, \sigma_2, ..., \sigma_n \in \Sigma^*$.

Definition 2 *The symbol $f(x, s)$ is defined for a string s if $\exists\ x_1, x_2, ..., x_n \in X$ and $(x, \sigma_1, x_1), (x_1, \sigma_2, x2), ..., (x_{n-1}, \sigma_n, x_n) \in \Delta$.*

Definition 3 *Languages generated. The language generated by $G = (X, \Sigma, \Delta, x_0, X_m)$ is $L(G) = \{s \in \Sigma^* : f(x_0, s)\}$ .The language is a set of symbols and strings that are generated when, starting from the initial state x_0, all transitions are traversed across the automaton's state space.*

Definition 4 *Languages marked. The language marked by* $G = (X, \Sigma, \Delta, x_0, X_m)$ *is* $L_m(G) := \{s \in L(G) : f(x_0, s) \in X_m\}$.

Marked states signifies the completion of a task. I.e. If the task is to go from state A to state B, the task is complete when state B is reached. Thus, state B is a marked state. The string "s" is able to take the system from the initial state to the marked state. The event generation does not necessarily need to stop at the completion of a marked sequence, since marked states are not necessarily final states.

$L_m(G) \subseteq L(G)$ is always true. For nonblocking, it is required that every string in $L(G)$ be a prefix of a string in $L_m(G)$. If this is true then every event sequence will eventually lead to a marked state.

The FSM model does not allow for timing. There are many instances in which the time when events occur may be unknown, unpredictable or simply may not be of importance to the model. In these cases, FSMs are suitable. Since events may occur at irregular intervals, time spent in states will also vary. This cannot be accurately modelled using FSMs. The FSM model assumes that events occur between [0, inf], i.e. at any time. State transitions, or events, often represent a physical, real life event that changes the physical state of what is being modeled. For example, the real life action of administering epinephrine to a neonate will change the physical state of the infant. (The state change can be positive or negative depending on how the epinephrine is administered). In the case of the NRP, events carried out by the health care provider should occur at particular points in time. The delay of certain events from occurring could cause irreparable damage to the neonate. Hence, timing is crucial to the NRP, since the longer a neonate is in a compromised state, the more damage that can occur, or even death. Timing issues are addressed with the use of timed transition machines, which are described in section 3.4. In addition to timing, event ordering is also crucial to the NRP.

3.1.2 Specifications

In the language $L(G)$, there may be strings that are unacceptable, because they are in violation of a condition. For example, any string that leads to the death of the neonate would be unacceptable. There may also be substrings that are unacceptable, because they represent an inadmissible ordering of events. For example, any string that does not follow the NRP algorithm. Hence, there will be a sublanguage of $L(G)$ that represents admissible behavior for the system. This is represented as a range of admissible behavior: $Lr \subset La \subseteq L(G)$ where Lr is the minimal required behavior, and La is the maximal required behavior.

3.2 Operations on Automata

3.2.1 Accessibility/ Reachability

Upon observation of $L(G)$ and $L_m(G)$, there may be states that are not accessible (reachable) from x_0 by some string in $L(G)$. These states can be removed from G, as well as all transitions that are attached to these states. This operation is denoted $Ac(G)$.

3.2.2 Coaccessiblity and Blocking

A state x of G is coassessible if there is a string in $L_m(G)$ that passes through x. Thus, there is a path in G from state x to a marked state in G. All states that are not coassessable can be removed from the automaton, and the operation is denoted $CoAc(G)$. Clearly this operation may affect $L(G)$ as states are deleted, but there would be no affect on $L_m(G)$.

Related to the concept of coaccessiblity is blocking. One form of blocking is deadlock. This occurs when the system enters a deadlock state prior to finishing the task. A deadlock occurs when no further event can be executed, and the task at hand has not yet finished. Another form of blocking is livelock. This occurs when a set of unmarked states forms a strongly connected component, yet have no transitions out of the set. Thus, events will continue to execute, hence the word "live", however, the task can never complete, because no state in the set is marked, and there are no transitions out of the set.

3.2.3 The Trim operation

A trim automaton is one that is both accessible and coassessible, i.e. $Trim(G) := CoAc[Ac(G)] = Ac[CoAc(G)]$.

3.2.4 Composition without Timing

Composition operations combine the behavior of two or more automata operating concurrently into a single joint automata. It is a useful tool in order to analyze or simulate a system modeled by a collection of automata. There exist two types of composition operations. One is the product operation, and the second is the parallel composition. The parallel composition (also known as synchronous product) is examined in more detail as follows.

Synchronous Product

The synchronous product of two automaton is denoted $G = G_1 \parallel G_2$, where $G, G_1$and G_2 are all FSMs, and $\parallel$ is the synchronous product operator. If $G_1 := (X_1, \Sigma_1, \Delta_1, x_{0,1}, X_{m,1})$ and $G_2 := (X_2, \Sigma_2, \Delta_2, x_{0,2}, X_{m,2})$ then $G_1 \parallel G_2 := (X, \Sigma, \Delta, x_0, X_m)$. Events common to both $\Sigma_1$and Σ_2 must be executed concurrently, if possible. The language of the product, denoted $L(G_1 \parallel G_2)$ is equal to the intersection of $L(G_1)$ and $L(G_2)$ if $\Sigma_1 = \Sigma_2$. In addition, $L_m(G_1 \times G_2) = L_m(G_1) \cap L_m(G_2)$.

Events that are unique to each automaton can occur at any time, and need not be "synchronized".

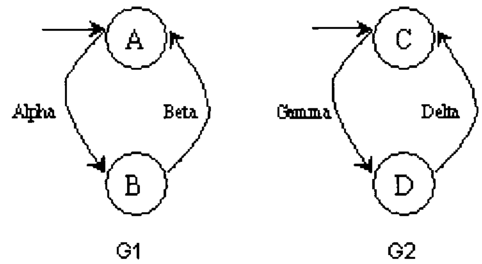

Figure 3-1: Two simple automata (G_1 and G_2) to demonstrate synchronous product where $X_1 = \{A, B\}$, $X_2 = \{C, D\}$, $\Sigma_1 = \{alpha, beta\}$, $\Sigma_2 = \{gamma, delta\}$, $x_{0,1} = A$, and $x_{0,2} = C$.

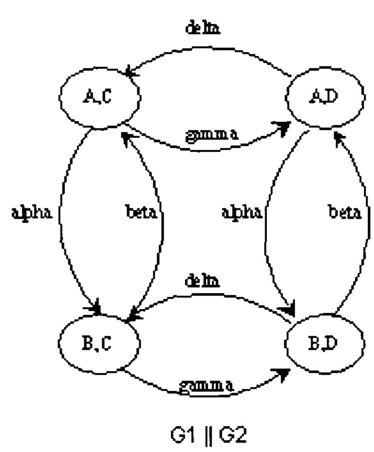

Figure 3-2: The synchronous product of two simple automata.

In the synchronous product, state names are derived from the state names from the original automata. Thus, in figure 3-2, state (A, C) derives its name from both G_1 and G_2 where A is the current state in G_1 and C is the current state in G_2. If (X, Y) is an ordered pair of G, where X is a state in G_1 and Y is a state in G_2 then the transitions of G would either be $(X, Y) \rightarrow (X', Y)$ or (X, Y') where the former is a transition in G_1 and the latter is a transition in G_2. The example shown in figure 3-2 is in fact a special case of synchronous product called the shuffle product, where $\Sigma_1 \cap \Sigma_2 = \emptyset$. The two automata do not have any events in common, resulting in a product possessing all possible shuffles of Σ_1 and Σ_2. It should be noted that the synchronous product operation can result in extremely large graphs, since the number of states in the product increases exponentially with the number of states in the originating models. This is called state explosion.

3.2.5 Supervisory Control

Control and coordination can ensure the proper flow of events. Some other properties that can be addressed through control are stability, and deadlock or livelock.

In controlling the DES or timed transition model (TTM), we assume there are some events that are controllable. In the visual graphic, a controllable event is often represented by a line crossing through the event arrow. When an event is controllable, it can be disabled or prevented from occurring when needed. Thus, the event set Σ must be partitioned into controllable and uncontrollable events, i.e. $\Sigma = \Sigma_u \cup \Sigma_c$. The controllable events in Σ_c can be prevented from occurring at any point in time. Of course, not all events in a system will necessarily be controllable. In some cases, despite the best efforts of the NRP provider, the neonate's state may not improve, and in fact could deteriorate, depending on the childbirth scenario. Though we would like to be able to control such an event that would cause the neonate's health to deteriorate, this will not always be possible in real life.

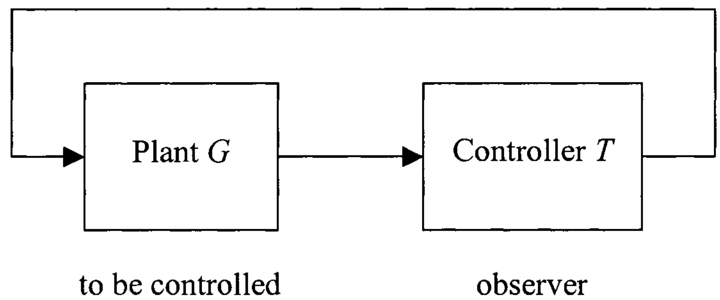

Figure 3-3: Closed Loop Controller

There will be a set of control inputs $\Gamma \subseteq \Sigma$ that will describe permissible events. Clearly $\Sigma_u \subseteq \Gamma$ since there is no way of disabling uncontrollable events. For any $\delta \in \Gamma$ then δ is enabled or permitted to occur. The distinguishing feature of this set is that Γ is not defined in the model, but is selected externally by the controller. It is the objective of supervisory control. Control of a DES involves continually switching control input in reaction to the string of previously generated events and allowing all uncontrollable transitions to occur. This type of control is called Supervisory Control, which involves feedback, i.e. a closed loop. See figure 3-3.

The main goal from the supervisory controller is to ensure that bad things don't happen, and good things do, i.e. undesirable sequences of events will not be allowed to occur while desirable sequences are. While administering the NRP, the goal is to keep the neonate alive and healthy, therefore any sequences of events that would lead to the death of the neonate must be disabled, if possible. By the same analogy, sequences of events that lead to a healthy neonate would be enabled. In essence, the open loop behavior of the plant G, needs to be modified so that it performs according to the childbirth scenario. For this to happen, the modeler needs to specify how she wants G to act. There are several methods available to do this. One is to specify the exact closed-loop behavior. Another is to specify the limits of behavior within which

G must be contained. In designing supervisory control strategies, one would like to ensure that the plant behaves both optimally and legally, i.e. the least restrictive while staying within the given specification.

3.3 Safety

There are several safety properties of an automaton that can be considered. In the case of undesirable or "dangerous" states in the automaton, there is concern about their reachablity from $x_{0.}$ Another concern is the presence of strings or substrings in $L_m(G)$ that are either illegal or undesirable. In other words, we ask is $L_m(G)$ a subset of an admissible or legal language?

3.4 Timed Discrete Event Systems

Finite state automata with the addition of timing is referred to here as the TTM [4]. Timing is an important factor in the NRP, since the longer the infant goes without breathing, the more likely it is that the infant will suffer irreparable damage, even death. When timing is introduced, events are constrained to occur within timebounds. Each transition will have its own upper and lower timebound and timebounds are relative to the clock associated with that particular transition. The lower timebound can be any natural number, while the upper timebound can be any natural number up to infinity. An instantaneous transition occurs when a timebound's constraint is satisfied by its clock value. A lower timebound of zero means "right now, without delay". It is important to note that timing occurs on events, not states. A timing of [0, inf] means the event can take place at any time. This is equivalent to the untimed case. A timing of [2, inf] indicates the event cannot take place before the passing of two time units. The event can occur anytime after two ticks of the clock. A timing of [2, 4] indicates the event cannot take place before two ticks of the clock, but it *must*

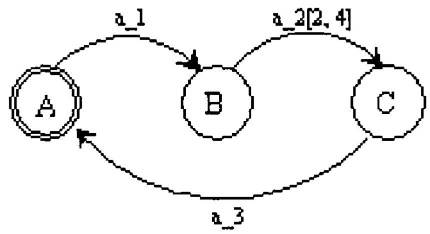

Figure 3-4: A simple TTM model.

occur by four ticks of the clock, i.e. it can go at t = 2, t = 3, and t = 4, and no other times. A tick event represents the passage of one unit of clock time. TTMs can in fact be modeled as FSMs, simply by adding extra states to the FSM model. These extra states will delay the event from occurring by the appropriate number of ticks. In the case when the upper timebound equals infinity, it is not feasible to model an infinite number of ticks. Instead, a self loop is added, indicating it can loop on itself until infinity. This ensures a finite model. The addition of timing is inherently more complex than a simple FSM, and often leads to much larger models than when no timing is incorporated into the model. The timed model is particularly sensitive to state explosion since there is an additional state for every clock tick. See figures 3-4 and 3-5.

Synchronization takes place much like in the untimed case. When a clock's value is equal to the timebound's upper time, the transition must take place, as long as the rules of synchronization are met, and this transition takes precedence over all other transitions of the composed system.

With the enhancement of timing, the concept of forcible events arises. These are events that preempt the tick of the global clock. An event can be both forcible and controllable [4].

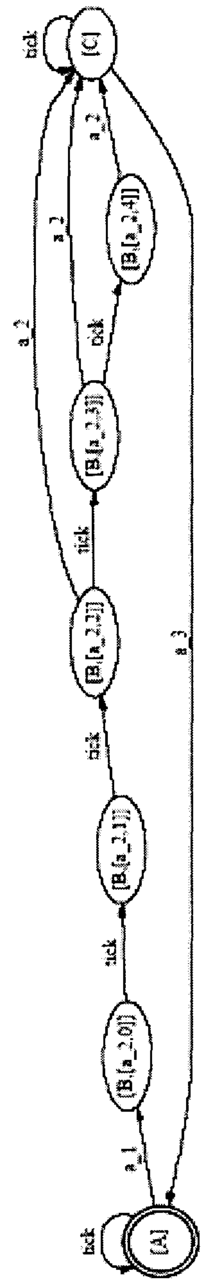

Figure 3-5: FSM equivalent of a simple TTM

3.5 The OTCT Software

To synthesize the discrete event systems, there are several software suites available. OTCT was originally created at the University of Toronto by Dr. Siu O'Young, and has continued to develop over last ten years [29]. OTCT synthesizes supervisory controllers for DES based on the work by Ramadge and Wonham [31]. In lieu of a GUI, the user writes scripts in .txt files to define the plant, using finite state machines, or timed transition machines, and specifications. Specifications can be both FSMs or TTMs. A DOS batch file is created, containing OTCT functions such as fsm, which creates an finite state machine object, or ttm, which creates a timed transition machine object. These objects, once created, can be manipulated by other OTCT functions as defined in the batch file. When the batch file has completed running, results can be viewed graphically in .pdf files and the control data appears as a .txt file. The OTCT functions that will be used in this work are:

1. fsm(object.fsm): This creates an FSM from the file object.fsm.

2. ttm(object.ttm): This creates a TTM from the file object.ttm.

3. sync(object1, object2): This forms the synchronous product of objects 1 and 2, which are either TTMs or FSMs. The result is either a TTM or FSM..

4. supfcBySync(plant, specification): This computes the maximally permissive controller given the plant and the specification. It enforces nonblocking, such that all states are reachable and coreachible.

5. trim(object1): The trim operation removes the illegal states from G , and all transitions attached to them, which enforces nonblocking.

6. condat(c): This computes the controller state feedback map for the controller c. It identifies where forcing or disabling must occur.

Chapter 4

Discrete Event Modeling of the NRP

4.1 Introduction

In this chapter, FSMs and TTMs are developed to model the neonatal resuscitation procedure, and the vital signs of a neonate that are related to resuscitation. The resuscitation procedure has been divided into smaller, individual procedures, which could be put together to form the full neonatal resuscitation procedure. It is more practical to examine the individual procedures, as they are manageable in size, and maintain a certain degree of complexity. If one were to model the entire resuscitation procedure in one structure, it would either be too dense for human analysis, or the modeler would have to abstract many of the procedure's subtleties in an effort to create a model of manageable size.

Supervisory control can be applied to these models with the use of a suitable software suite, such as OTCT. In Chapter 5, verification of a small part of the neonatal resuscitation procedure is examined by creating several resuscitation scenarios.

4.2 NRP Specifications

In keeping with the NRP algorithm, there will be certain strings or substrings that violate legal behavior. Some informal examples of behavior that can be enforced and that are relevant to the NRP include:

1. Avoiding a list of illegal states of G. An example of an illegal state when modeling the health of a neonate would be the dead state. Illegal states should be removed from G, and all transitions attached to them.

2. Executing a set of events in certain priority order. For example, the NRP algorithm insists the neonate should be dried before providing oxygen.

3. Not allowing event X to occur more than N times between two occurrences of event Y. For example, after checking the neonate's heart rate, do not ventilate more than 15 times before checking its heart rate again.

4. Execute event Z only if event X precedes event Y in the string generated so far. This requires remembering how a particular state of G was reached in order to determine the future admissible behavior. In this case, state splitting should be utilized, in which a state is split into as many states as necessary, and the event set is adjusted accordingly.

5. Alternating between the two events, positive pressure ventilation and chest compressions, with positive pressure ventilation occurring first.

4.3 The Neonate's Vital Signs and Statistics

The infant's heart rate, skin tone, and breathing are all vital signs of interest in the NRP, and can be modeled as automata. Other physical properties of the infant that are of interest are depth of chest compression, and neck tilt, which are both direct results of the trainee's actions. The Anakin mannequin is currently equipped with sensors that detect chest compression depth, neck tilt, and depth of the endotracheal tube (see Intubation in the next section). It also has actuators that reveal heart rate, and skin tone. The feedback from the sensors is a direct result of trainee intervention. With the current system, the mannequin is connected either to a PC, or a PDA, whose GUI displays the sensor's feedback. Thus, information that would normally prove difficult to ascertain by simple visual means, such as depth of endotracheal tube, is available to the trainee or trainer for evaluation.

4.3.1 Heart rate

A heart rate of 100 beats per minute (bpm) or greater is considered in the normal, healthy range. A heart rate in the range of 60 to 99 bpm is low, and an indication of trouble. This requires positive pressure ventilation, possibly combined with chest compressions if positive pressure ventilation alone is not effective. When the heart rate falls within this range, the distinction between whether it is increasing or decreasing/no change is a consideration. Below 60 bpm, the infant is in serious danger. In modeling the heart rate as an automaton, it is not necessary to model each individual beat per minute as a separate state, since doctors have distinguished the three ranges mentioned above, and the 60-99 bpm range can be separated into two automata for increasing or decreasing/no change. Thus, the heart rate automaton has a state representing each of the five possible ranges (greater than 100 bpm, between 60 and 100 bpm increasing, between 60 and 100 bpm decreasing/no change, and less than 60 bpm). This automaton provides sufficient distinction between the range of possible heart rates. Note that in figure 4-1, "<60 bpm" is indicated as the initial state, however, any of the four states could be the initial state depending on the particular childbirth scenario. ">100 bpm" is the marked state. Table 4.1 lists the event labels and their meanings.

4.3.2 Skin tone

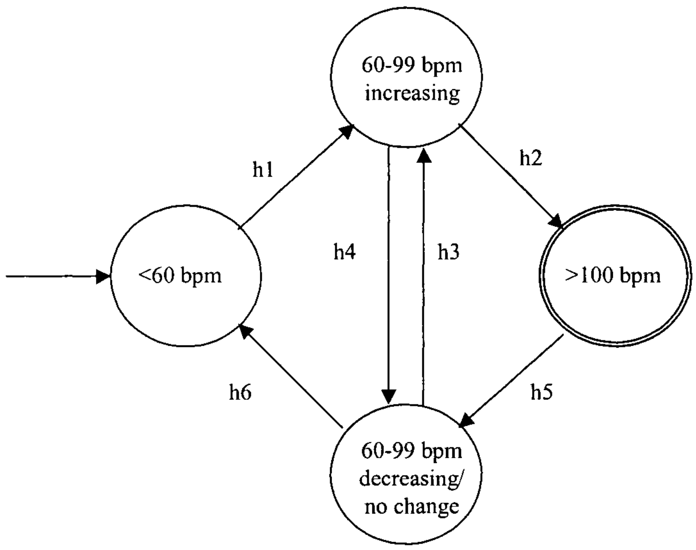

Figure 4-1: Heart Rate State Machine

Event Label	Description
h1	Heart rate increases above 60 bpm.
h2	Heart rate increases above 100 bpm.
h3	Heart rate increases and is within 60-99 bpm.
h4	Heart rate decreases (or stops increasing), and is within 60-99 bpm.
h5	Heart rate decreases below 100 bpm.
h6	Heart rate decreases below 60 bpm.

Table 4.1: Heart Rate Event Labels

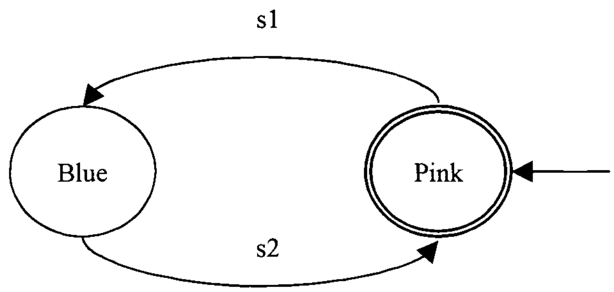

Figure 4-2: Skintone State Machine

Event Label	Description
s1	Skin tone deteriorating to blue
s2	Skin tone improving to pink

Table 4.2: Skin Tone Event Labels

The infant's skin tone is a visible sign of its state of health. Skin tone generally takes on two tones: pink or peripheral cyanosis, for a healthy infant, or blue for an infant lacking oxygen. Thus, the skin tone automaton is a simple two state automaton. In figure 4-2, the initial state is indicated as "Pink", however, the initial state could equally be "Blue" depending on the particular childbirth scenario. "Pink" is the marked state. Table 4.2 lists the event labels and their meanings.

4.3.3 Breathing

Breathing is the most important factor in the NRP, and the first vital sign that NRP providers should look for. It takes precedence over heart rate and skin tone. If the infant is not breathing spontaneously, then positive pressure ventilation is commenced immediately. Only after evaluating the infant's breathing, and finding it to be spontaneous does the NRP provider evaluate the heart rate. If the heart rate is less than 100 bpm, positive pressure ventilation is started immediately. Finally, if both the breathing and heart rate are satisfactory, then the skin tone is evaluated. If the infant's skin tone is blue, yet exhibits spontaneous breathing and a heart rate

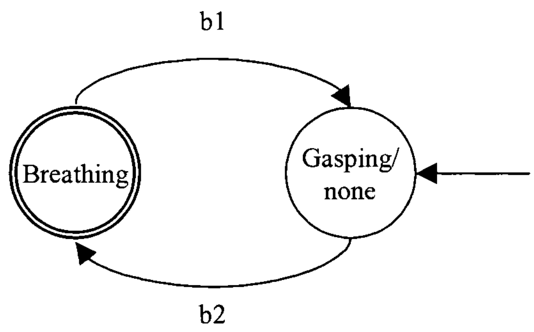

Figure 4-3: Breathing State Machine

Event Label	Description
b1	Breathing deteriorating
b2	Breathing improving

Table 4.3: Breathing Event Labels

greater than 100 bpm, then the infant is administered 100% O_2. Positive pressure ventilation in this latter case would not be required. It is likely that in the case of "no breathing present", and/or the heart rate is less than 100 bpm, the neonate's skin tone will be blue. However, this is considered to be "not of interest", as the other vital signs take precedence over skin tone.

Breathing can be divided into three categories: breathing without difficulty (i.e. spontaneous), breathing with difficulty (i.e. gasping) , and not breathing. The breathing automaton thus consists of three states. However, both gasping and not breathing are both serious concerns and would require intervention, and therefore these two states can be modeled as one. Thus, the automaton has only two states. In figure 4-3, "Gasping/none" is indicated as the initial state, however either state could be the initial state depending on the childbirth scenario. Breathing is the marked state. Table 4.3 lists the event labels and their meanings.

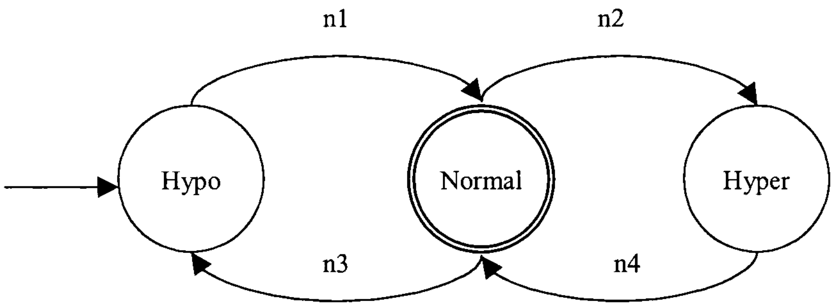

Figure 4-4: Neck Tilt State Machine

4.3.4 Neck Tilt

An important consideration in NRP training is how to properly position the infant's neck. Neck position can be divided into three categories: hyperextended, normal, or hypoextended. The trainee, when administering positive pressure ventilation, must position the infant's neck in a slightly extended manner, such that the airways are open and thus receptive to positive pressure ventilation. If the neck is hyperextended, the neonate's neck is subject to injury, and the airways may be cut off, thus rendering positive pressure ventilation less effective or ineffective. If the neck is hypoextended, this indicates the neck is not sufficiently extended to maintain a clear airway, thus rendering positive pressure ventilation ineffective. A "normal" neck position provides the correct neck extension such that the airways are clear, and it is the position required for proper administration of positive pressure ventilation. Thus, a three state automaton is sufficient to model the states of the neck (see figure 4-4). "Normal" is the marked state, while "Hypo" is indicated as the initial state. However, the latter could change depending on the particular childbirth scenario. Table 4.4 lists the event labels and their meanings.

Event Label	Description
n1	Extending the neck to a normal position
n2	Not extending the neck sufficiently
n3	Extending the neck too much
n4	Extending the neck to a normal position

Table 4.4: Neck Tilt Event Labels

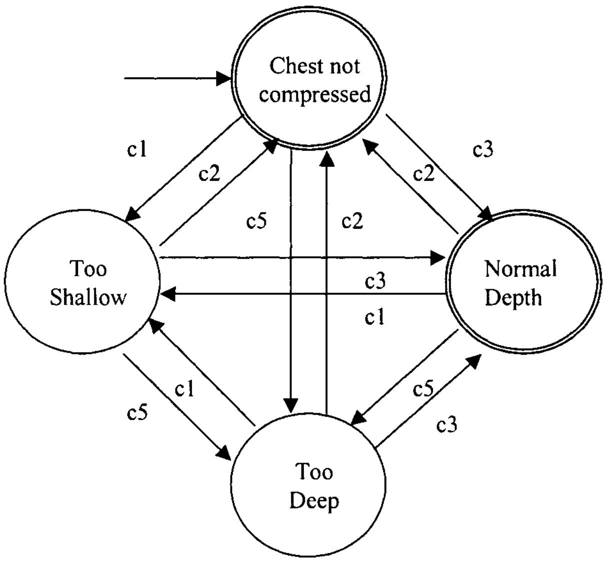

Figure 4-5: Chest Compression Depth State Machine

4.3.5 Chest Compression Depth

The depth of the chest compression given by the trainee is another important consideration to the NRP. The guidelines recommend a relative chest compression depth of one third to one half of the anterior-posterior dimension of the chest. If the chest compression depth is too shallow, it will be ineffective; too deep and the infant's chest cavity could be damaged. Thus, the chest compression automaton is modeled with three states: too shallow, too deep, and normal. A fourth state is introduced, representing no chest compression (see 4-5). This is selected as the initial and marked state. Another marked state is "Normal Depth". Table 4.5 lists the event labels and their meanings.

Event Label	Description
c1	Compress chest too shallow
c2	Stop chest compressions
c3	Compress chest at proper depth
c4	Stop chest compressions
c5	Compress chest too deeply

Table 4.5: Chest Compression Event Labels

4.4 The Trainee's Actions

The trainee's actions can also be modeled as automata. Actions are grouped into the following procedures.

1. Initial steps
2. Chest Compression
3. Ventilation
4. Intubation
5. Epinephrine

Within each of these individual procedures there is a specific ordering of events, and frequently there is timing associated with these events. Yet these procedures are also linked at a higher level. Taken as a whole, and performed in a specific order, they form the NRP. Depending upon the health of the neonate, not all procedures may be required. In the worst case scenario, all of the procedures, from initial steps to administering epinephrine inclusive would be needed. The state machines that follow show the correct way to perform the individual procedures. In practice, the trainee can make mistakes in the ordering and/or timing of events. The neonate's health will reflect the quality of care being provided. Thus, if the procedures are carried out in an effective and timely manner, the neonate's health will ultimately improve. Otherwise, the neonate's vital signs will indicate a worsening condition.

4.4.1 Initial Steps

Upon the arrival of a newborn baby, the medical practitioner must quickly determine the need for resuscitation. There are several key indicators that will cause the practitioner to immediately commence resuscitation. These indicators include: the presence of meconium, absence of respiration or crying, poor muscle tone, bluish skin tone, and preterm gestation. These are evaluated concurrently via visual inspection. If any of these indicators are present, the initial steps of resuscitation are commenced. The initial steps address the following needs.

- providing warmth,
- establishing a clear airway,
- tactile stimulation,
- administering free flow oxygen.

Providing warmth to the newborn infant is considered vital, since stress due to cold can increase oxygen consumption and be an impedance to resuscitation [19]. In order to prevent heat loss, the infant is placed immediately on a radiant warmer, its skin dried, and placed in warmed blankets. Alternatively, the dried neonate can be placed on the mother as a source of heat.

Clearing the airway is achieved by properly positioning the infant's neck, and removing secretions as necessary. Tactile stimulation is used to induce spontaneous respirations. Drying and suctioning the infant are commonly enough stimulation in most cases, though stimulation of the feet and the back may also be used. Free flow oxygen is given to achieve a pink skin tone. It may be given through an oxygen mask or oxygen tubing.

According to the NRP guidelines, the initial steps are performed in a specific order and consist of placing the neonate on a radiant warmer, removing wet linen and

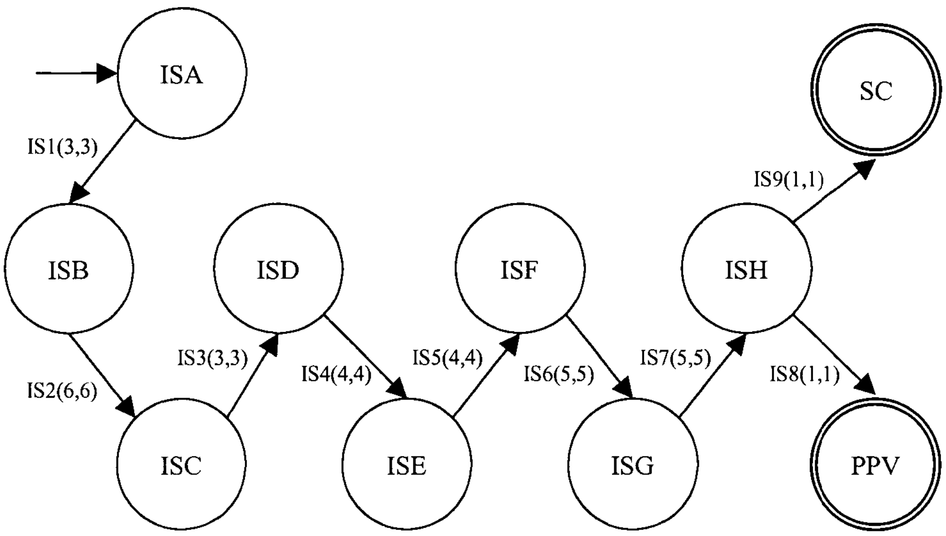

Figure 4-6: Initial Steps State Machine with Timing

drying, properly positioning the neck, placing a shoulder roll (optional), considering intubation if meconium is present (see Meconium), suctioning mouth and nose, providing tactile stimulation by slapping soles of feet or firmly rubbing spine if apneic, and administering free flow oxygen.

Upon completion of these initial steps, assessment of the newborn's respiration, heart rate and color will determine whether continued resuscitation efforts are needed. The initial steps are completed in 30 seconds. If heart rate, breathing and skin tone are all satisfactory, then supportive care is provided. However, if either the breathing or heart rate are unsatisfactory, (i.e. heart rate is less than 100 bpm, and the neonate is gasping or not breathing), positive pressure ventilation is commenced immediately. In the case that the neonate is breathing spontaneously, and its heart rate is above 100, yet its skin tone is blue, 100% oxygen is provided.

Timing has been added to the transitions, indicating the upper and lower timebounds for each step. Referring to figure 4-6, Table 4.6 lists the states and their descriptions.

State	Description
ISA	Start initial steps; place neonate on radiant warmer
ISB	Remove cloth and dry
ISC	Extend neck
ISD	Suction mouth
ISE	Suction nose
ISF	Provide tactile stimulation
ISG	Administer free flow oxygen
ISH	Evaluation
SC	Start supportive care
PPV	Start positive pressure ventilation

Table 4.6: Initial Steps State Descriptions

4.4.2 Positive Pressure Ventilation (Bag and Mask Ventilation)

Positive pressure ventilation has a specific ordering and timing of events. Ventilation is often provided with a bag and mask. The correct sized mask for the infant must be used, and it must be positioned properly around the neonate's mouth and nose. For positive pressure ventilation to be effective, the neonate's neck must be positioned properly. Positive pressure ventilation begins with an initial pressure of 30-40 cm/H_2O. Subsequently, a pressure of 15-20 cm/H_2O is given, and repeated at a delivery rate of 40-60 per minute for 30 seconds. This is equivalent to 20-30 events. Throughout this procedure the trainee must observe for chest rise. If the chest is not rising, then the neck may not be positioned properly, the mask may need to be reapplied, or perhaps there are secretions in the airway that must be removed. Using a higher pressure to ventilate may also be a consideration. Once the first round of positive pressure ventilation has been provided, the heart rate is reevaluated. If the heart rate is below 60 bpm, positive pressure ventilation with chest compressions is commenced. If the heart rate is between 60 bpm and 100 bpm and increasing, positive pressure ventilation continues. In the case that the heart rate is not increasing, positive pressure ventilation with chest compressions is commenced. In the case that

Heart rate (bpm)	Action
<60	Continue PPV with chest compressions.
60-100 increasing	Continue PPV.
60-100 not increasing	Continue PPV.
>100	Discontinue PPV. Watch for breathing. Provide ongoing care.

Table 4.7: Actions Based on Heart Rate

the heart rate is above 100 bpm, and breathing is spontaneous, positive pressure ventilation is discontinued, and ongoing care is provided. Table 4.7 lists the trainee's actions based on a baby's heart rate in bpm.

In figure 4-7, all of the ventilation events following the initial pressure are combined into one event, and labeled "bb". When "bb" is successfully completed, the state changes to "Ventilated for 30 seconds". The neonate's heart rate and breathing states are added to the figure to make the decision process explicit, however this decision process would not normally be found in a finite state machine. One can add more detail to this state machine by making every ventilation an event. Since 20-30 ventilation events are expected, from 20 events onward, it is acceptable to check the neonate's heart rate and breathing. This more detailed state machine is shown in figure 4-8, with the neonate's states removed. In both figures, the states "Ongoing Care" and "PPV & Chest Compressions" are marked states, since both indicate a successful end to the ventilation procedure. Tables 4.8 and 4.9 list the event labels and states for figures 4-7 and 4-8 respectively.

Figure 4-9 shows a simple positive pressure ventilation model with timing. The initial compression takes place in state A and takes one time unit. Event "a" signifies the initial compression has been delivered, at which time the automata moves to state B. Ventilation takes place in state B and takes 30 time units. Upon completion of ventilation, the automata will move to state C for ongoing care, state D for positive pressure ventilation and chest compressions, or self loop on state B to repeat the ventilation procedure. The choice will depend upon the neonate's heart rate, respiration

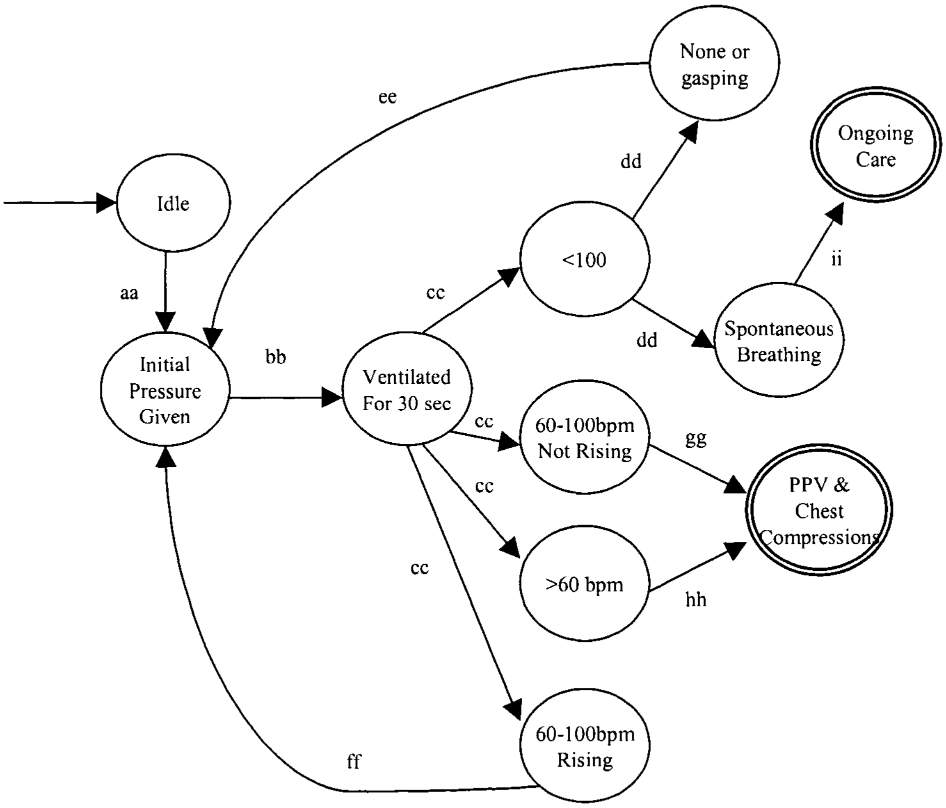

Figure 4-7: Positive Pressure Ventilation (PPV) Diagram

Event Label	Description
aa	Provide initial pressure of 30-40 cm H_2O
bb	Ventilates for 30 seconds using 15-20 cm H_2O
cc	Evaluates heart rate
dd	Evaluates breathing
ee	Provide initial pressure of 30-40 cm H_2O
ff	Provide initial pressure of 30-40 cm H_2O
gg	Start PPV with Chest Compressions
hh	Start PPV with Chest Compressions
ii	Provide Ongoing Care

Table 4.8: PPV Event Labels

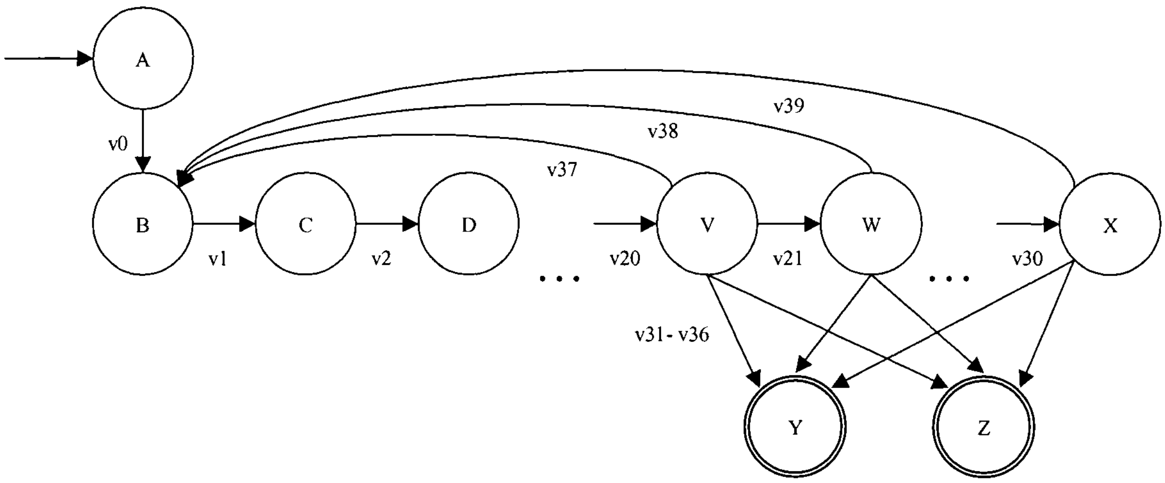

Figure 4-8: Expanded PPV State Machine

State	Description
A	Starts initial compression
B, C, D	Starts ventilation events 1, 2, 3.
V, W, X	Starts ventilation events 20, 21, 30
Y	Start Ongoing Care
Z	Start PPV & Chest Compressions

Table 4.9: PPV State Descriptions

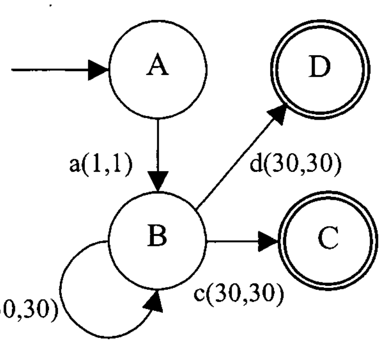

Figure 4-9: PPV Timed Transition Model (TTM)

and skin tone.

4.4.3 Chest Compressions

Chest compressions are always administered in conjunction with positive pressure ventilation. Therefore, one automata representing the combined behavior of the ventilation procedure and chest compressions can be created. To properly administer chest compressions, the trainee must correctly locate the lower third of the sternum, and use the finger tips of the middle and index fingers or the distal portion of both thumbs, as to not injure the infant. The neonate's back must always be firmly supported throughout the procedure. As the chest compressions are given, the trainee repeats, "one and two and three and breath and..."), ensuring ventilation after every third chest compression. The trainee delivers compressions and breaths at a rate of 120 events per minute (90 compressions and 30 breaths), continuing for 30 seconds, at which time the heart rate is reevaluated. In practice, this is equal to approximately 60 events. The exit or marked states for the chest compression automaton in figure 4-10 are "Intubation", "PPV", and "Epinephrine". The choice of action following the chest compression procedure depends on the triad of vital signs: heart rate, respiration and skin tone. If the heart rate is above 60, then chest compressions are discontinued, and ventilation alone is administered. However, if the heart rate is less than 60 bpm, epinephrine may be considered. The administering of epinephrine may require intubation and is usually a last attempt when other resuscitation methods have proved ineffective.

Figure 4-10 groups essentially four events into one. The three compressions followed by one ventilation ("one and two and three and breath..." are shown as one event, after which time the state changes. Since there are 60 individual events in total, there will be 15 of these grouped events. This state machine could be expanded into a much larger one by having a new state after each of the 60 events. In contrast, figure 4-11 shows a simplified four state TTM where the entire chest compression and

Event Label	Description
cc0...cc13	3 chest compressions followed by one ventilation
cc14	Administer epinephrine
cc15	Intubate infant
cc16	Return to PPV procedure

Table 4.10: Chest Compression Event Labels

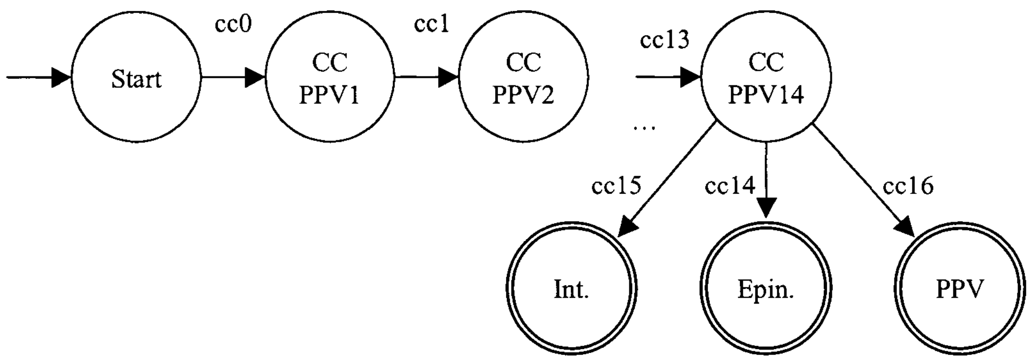

Figure 4-10: Chest Compression State Machine

ventilation routine is grouped into one event.

4.4.4 Intubation

Endotracheal intubation is a procedure by which a tube is inserted through the mouth down into the trachea. The tube that is used is often a flexible plastic tube called an endotracheal tube. This tube is inserted using a laryngoscope, an instrument that allows the doctor to see down into the airway. The endotracheal tube serves as an open passage through the upper airway. The purpose of endotracheal intubation is to permit air to pass freely to and from the lungs in order to ventilate the lungs. Endotracheal tubes can be connected to ventilator machines to provide artificial respiration. Intubation is an advanced airway management protocol, and is administered if the combined positive pressure ventilation and chest compressions have failed to increase the heart rate above 60 bpm.

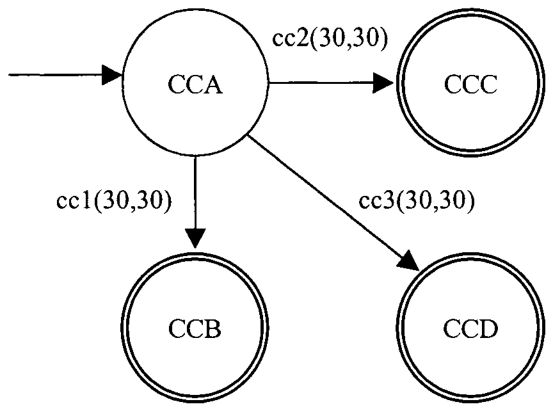

Figure 4-11: Chest Compression State Machine with Timing

Event Label	Description
in1	Insert laryngoscope
in2	Insert endotracheal tube
in3	Remove laryngoscope
in4	Connect bag to tube
in5	Ventilate
in6	Cannot place (or find) endotracheal tube
in7	Abort intubation attempt

Table 4.11: Intubation Event Labels

Some considerations during this procedure are the correct positioning of the laryngoscope, inserting the breathing tube the correct distance into the trachea (neither too shallow nor too deep), confirming air entry into both sides of the chest, confirming absence of air into the stomach, securing the endotracheal tube with tape, and connecting the bag and ventilating with an appropriate rate and pressure. The entire procedure is performed within 20 seconds. Figure 4-12 shows the state machine for the endotracheal tube depth, which has three states. The meanings of the states are self explanatory. Table 4.11 describes the events in figure 4-13.

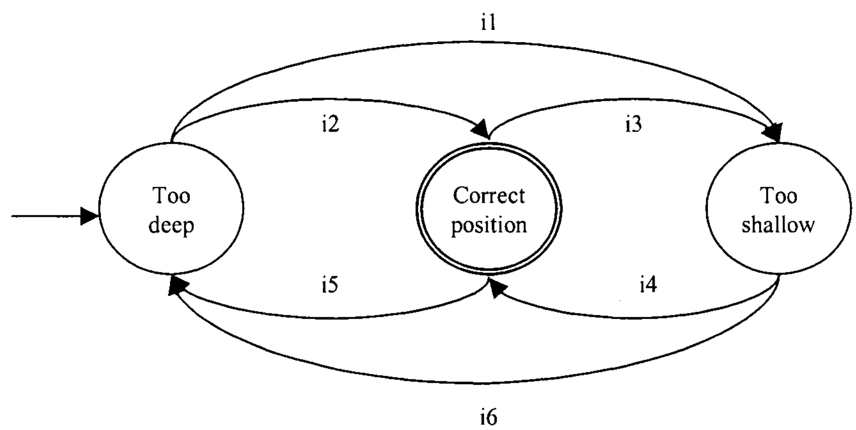

Figure 4-12: Endotracheal Tube Depth State Machine

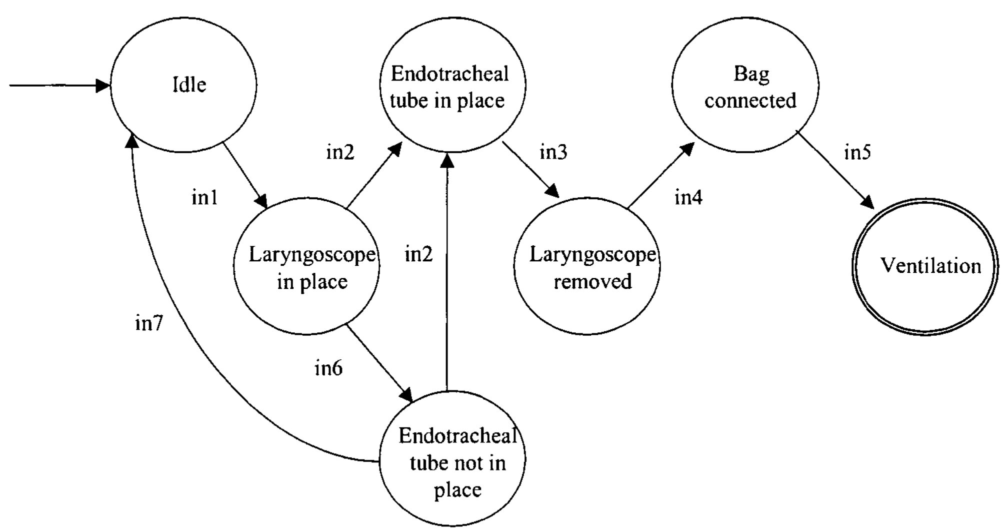

Figure 4-13: Intubation State Machine

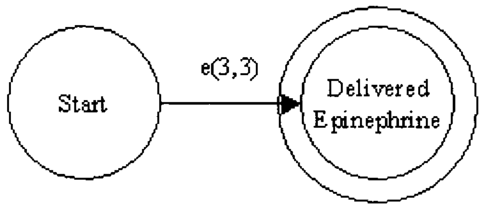

Figure 4-14: Epinephrine State Machine with Timing

4.4.5 Administer Epinephrine

The drug epinephrine is administered if all other attempts to resuscitate the infant have failed. The strength of epinephrine is 1:10,000, and the dosage is 0.1-0.3 mg./kg. The route of delivery can either be the umbilical vein or endotracheal tube. The rate of infusion is rapid, and can be repeated after 3 to 5 minutes if necessary. The automaton for this procedure can be a simple 2 state machine, where event "e" occurs after the correct strength and dosage of epinephrine has been administered .

4.4.6 Meconium

Meconium is the greenish black stool passed by some newborns. Occasionally, the neonate passes this stool during delivery, while still in the womb. This is known as meconium staining of the amniotic fluid, and occurs in approximately 5% to 10% of births. The amniotic fluid and meconium mix to form a green stained fluid of various viscosity. There is the possibility that the neonate will aspirate while still in the uterus, or while still covered in this meconium stained fluid after birth, in which case the meconium can be inhaled into the lungs. This can cause a partial or complete blockage of the airways. It is also an irritant, causing inflammation of the airways, and possibly chemical pneumonia. Meconium aspiration occurs frequently in post term fetuses (i.e. more than 40 weeks gestation), and in fetuses that are stressed during labor. This condition is serious, and is a leading cause of severe illness and death in the newborn. About one third of infants with meconium aspiration will require some type of assisted breathing.

In the presence of meconium, the trainee must first address this problem even before performing the initial steps of the NRP. If the neonate is not vigorous, tracheal suction is required. To do this, the trainee must first correctly insert a laryngoscope into the mouth. The trainee may optionally suction the oropharynx and request the cricoid pressure. The endotracheal tube is inserted into the trachea and the laryngoscope is removed. The meconium aspirator is then connected. The endotracheal tube is withdrawn while suction is being applied. These steps must be performed within 20 seconds. The procedure needs to be repeated until such time that there is no more meconium in the trachea, or if the neonate's heart rate is greater than 100 bpm. If the heart rate is less than 100 bpm, then ventilation is required. Once the problem of meconium aspiration has been addressed, the trainee continues on with the initial steps.

Chapter 5

Using Supervisory Control with the NRP

It is up to the instructor to create childbirth scenarios which will determine the level of care needed in order to restore the health of the neonate. A spectrum of dynamic scenarios can be created. For example, in some cases a neonate would only require the initial steps, while in more severe cases, the neonate would require extensive care from positive pressure ventilation to epinephrine. Prior to each practice session with the Anakin, the instructor (or user) would create a birth scenario, which will determine the course of action that must follow, in order to restore the neonate's health.

To demonstrate supervisory control of the NRP using OTCT, several scenarios have been created. These scenarios have been devised purely to show supervisory control at work, and may or may not reflect a real life situation. For example, one of the scenarios involves one NRP provider and two neonates in need of positive pressure ventilation. Ideally in real life, there would be two or more NRP providers to handle such a scenario. However, in this case, the NRP provider's actions can be coordinated using supervisory control such that both neonates survive.

In these scenarios, liberty has been taken with the NRP algorithm, particularly in terms of timing. Timing has been altered to keep the models small, and to make the analysis easier. In addition, these examples focus on a small part of the NRP rather than the entire algorithm, in an attempt to derive easy-to-follow models. The OTCT code for the following scenarios is found in Appendix A. Appendix A also contains the text versions of the state/transition maps of the plants and supervisory controllers, as a supplement to the visual maps shown here.

5.1 Scenario 1a: One NRP Provider and One Neonate

Scenario one involves an NRP provider and a neonate. The health of the neonate has been simplified to three basic states. The "normal" state represents the case where all vital signs are within the normal, healthy range. The "abnormal" state represents a neonate in need of medical attention, because one or more of its vital signs are not within the healthy range. The final state is "dead". If the neonate is in the "abnormal" state, it can transition back to the "normal" state if proper medical care is provided, but once the neonate is in the "dead" state, nothing more can be done. "Normal" is a marked state. The transition between the "abnormal" and "dead" states has timing associated with it, signifying that the neonate will not immediately die if one or more of its vital signs are compromised. Rather, as in real life, it will die after a period of time if proper medical attention is not provided. In this example a time bound of [3, infinity] has been assigned. This indicates that after three clicks of the clock, and thereafter, the neonate can die, but not before. See figure 5-1.

The positive pressure ventilation procedure is modeled very simply by two states: "XPPV" and "PPV", where "XPPV" is the marked state. The "PPV" state represents the caregiver actively performing positive pressure ventilation while "XPPV" means that no treatment is being given. See figure 5-2. The event "a0" is expeditible (also called forcible), so that the supervisor can force the start of positive pressure ventilation. There is a timebound of [1,1] on the "b0" transition, meaning that the positive pressure ventilation procedure takes one unit of time. Following treatment,

PLANT	[[abnormal1,[c2,1]],[XPPV]]
SUPER	[[[abnormal1,[c2,1]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a0

Table 5.1: Scenario 1a Control Data

the model returns to the marked state, "XPPV".

The models for positive pressure ventilation and the neonate share a common event, "b0". In the timed transition model for the neonate, "b0" is the one way transition from the "abnormal" to "normal" state, which signifies that the neonate's health has improved to that of a healthy neonate. In the positive pressure ventilation timed transition model, "b0" occurs when the positive pressure ventilation procedure has ended. Since the label on these two transitions is the same, they must be synchronized in the plant.

The specification for this example is simple. The "c2" transition will not be allowed, where "c2" is the transition from the "abnormal" to "dead" state. This is because the "dead" state is not marked and has no exit. The specification is seen in figure 5-3.

When the synchronous product is taken of the positive pressure ventilation and neonate model, the result is the plant, which has 15 states and 22 transitions. Using the plant and the specification, the maximally permissive controller is computed, as seen in figure 5-5. This also enforces non-blocking so all states are reachable and coreachable. The circles areas in the plant diagram show the states and transitions that are "trimmed" in the controller. Table 5.1 displays the control data, by listing the plant state followed by the supervisor state where disabling and/or forcing occur, together with the transitions which must be disabled or forced there.

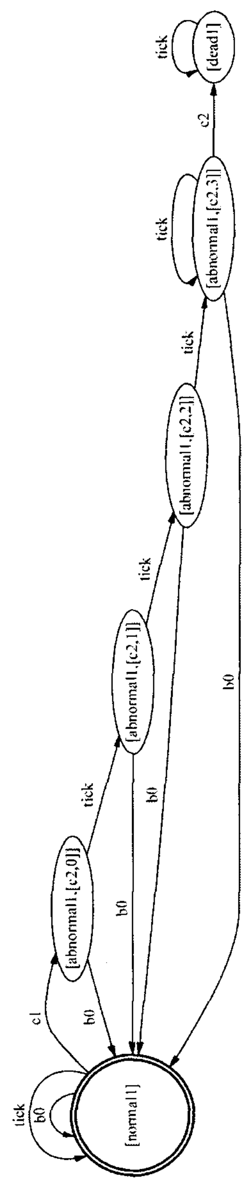

Figure 5-1: Scenario 1a Neonate Timed Transition Machine

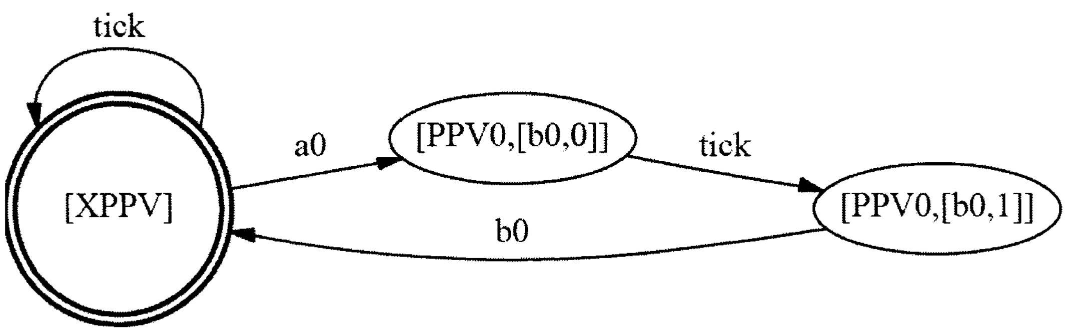

Figure 5-2: Scenario 1a Positive Pressure Ventilation Timed Transition Machine

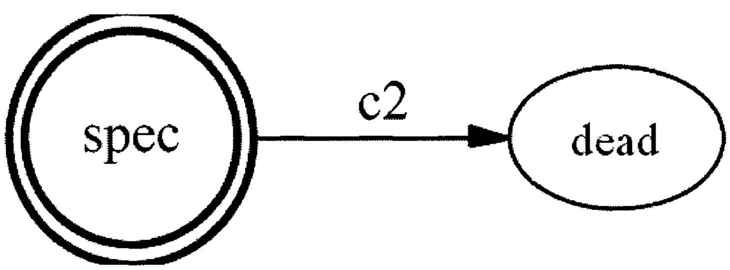

Figure 5-3: Scenario 1a Specification State Machine

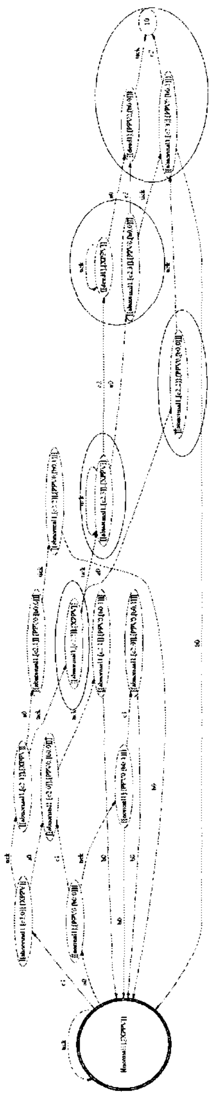

Figure 5-4: Scenario 1a Plant (no control)

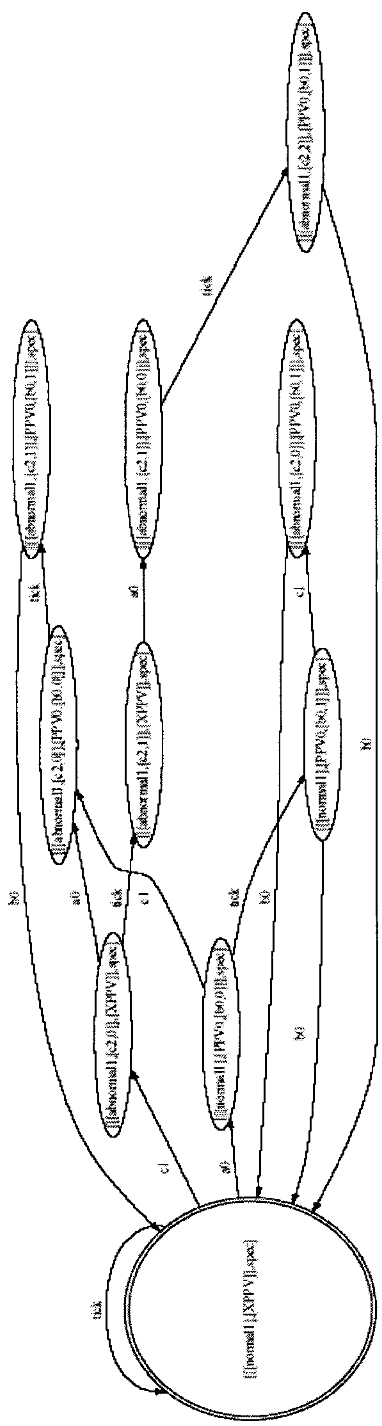

Figure 5-5: Scenario 1a Supervisory Controller

5.2 Scenario 1b: One NRP Provider and One Neonate Revised

It can be seen in scenario 1a that the model allows the neonate's state to be "normal" while positive pressure ventilation is being performed. This would not be acceptable in real life, since performing positive pressure ventilation on a healthy neonate could actually cause harm. Scenario 1b rectifies this situation. To specify in the model that positive pressure ventilation should not be performed on a healthy neonate, a self loop is added to the "abnormal" state labeled "a0". This is shown in figure 5-6. Now "a0" is common to both the neonate and positive pressure ventilation models, and therefore will occur simultaneously in the synchronous product. This guarantees that this situation never arises, as shown in the supervisory controller in figure 5-8. The circles areas in the plant diagram (5-7) show the states and transitions that are "trimmed" in the controller, which enforces reachablity and coreachability. The specification and neonate models are unchanged from scenario 1a, and the control data results are the same.

5.3 Scenario 2: One NRP Provider and Two Neonates

We can add to scenario 1b by introducing a second neonate. The task is to coordinate the NRP provider's actions such that both neonates survive. A new automaton representing the second neonate is added. It is similar to the neonate automaton described in scenarios 1a and 1b in both the states and transitions. However, the states have different names and the transitions have different labels. Theses two models are synchronized together to form one model, which is too large to show here graphically, having 36 states and 156 transitions.

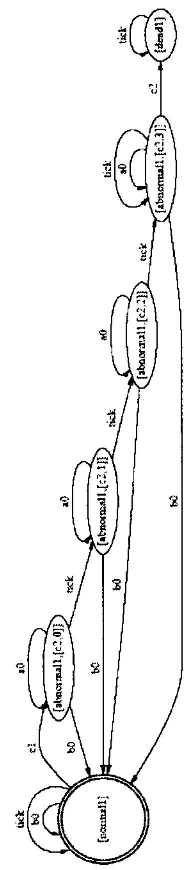

Figure 5-6: Scenario 1b Neonate Timed Transition Machine

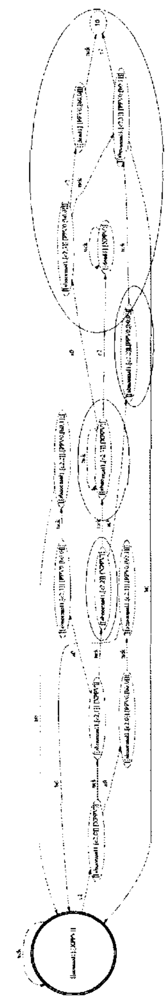

Figure 5-7: Scenario 1b Plant (no control)

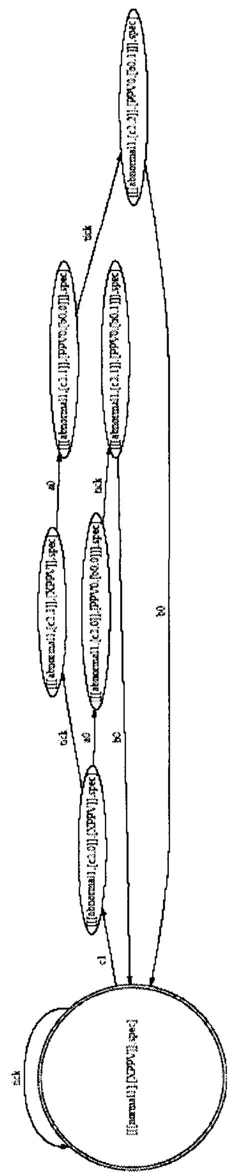

Figure 5-8: Scenario 1b Supervisory Controller

PLANT	[[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[XPPV]]
SUPER	[[[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a0, a1
PLANT	[[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[XPPV]]
SUPER	[[[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a1 Delay a0
PLANT	[[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[XPPV]]
SUPER	[[[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a0 Delay a1
PLANT	[[[abnormal1,[c2,1]],[normal2]],[XPPV]]
SUPER	[[[[abnormal1,[c2,1]],[normal2]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a0
PLANT	[[[normal1],[abnormal2,[c4,1]]],[XPPV]]
SUPER	[[[[normal1],[abnormal2,[c4,1]]],[XPPV]],spec]
EXPEDITE / DELAY	Expedite a1

Table 5.2: Scenario 2 Control Data

The positive pressure ventilation automaton (see figure 5-11) expands to take into account the second neonate. The states "PPV0" and "PPV1" represent the treatment of two separate neonates. Its transitions labeled "a0" and "a1" are both expeditible and controllable. This model is synchronized with neonate model to form the plant.

The specification is similar to scenarios 1a and 1b, except for the addition of the "c4" transition (see figure 5-12). As before, the function supfcBySync is used with the plant and the specification, resulting in the maximally permissive controller for scenario 2. The plant and the controller are too large to be seen properly in this thesis format, however they are included in figures 5-9 and 5-10 respectively so the reader can appreciate the complexity involved even with the simple models used here. Table 5.2 displays the control data, by listing the plant state followed by the supervisor state where disabling and/or forcing occur, together with the transitions which must be disabled or forced there.

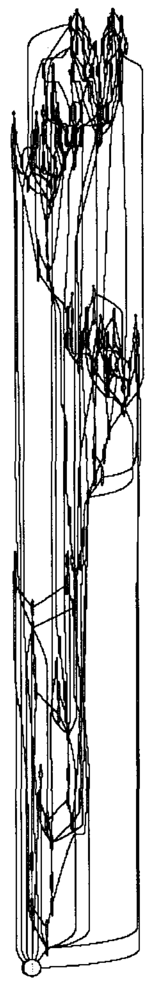

Figure 5-9: Scenario 2 Plant

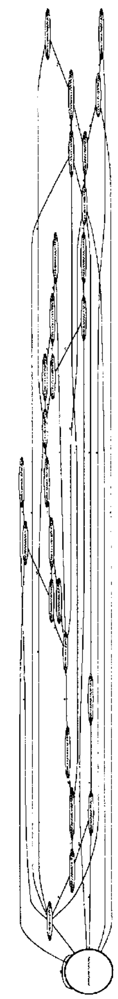

Figure 5-10: Scenario 2 Supervisory Controller

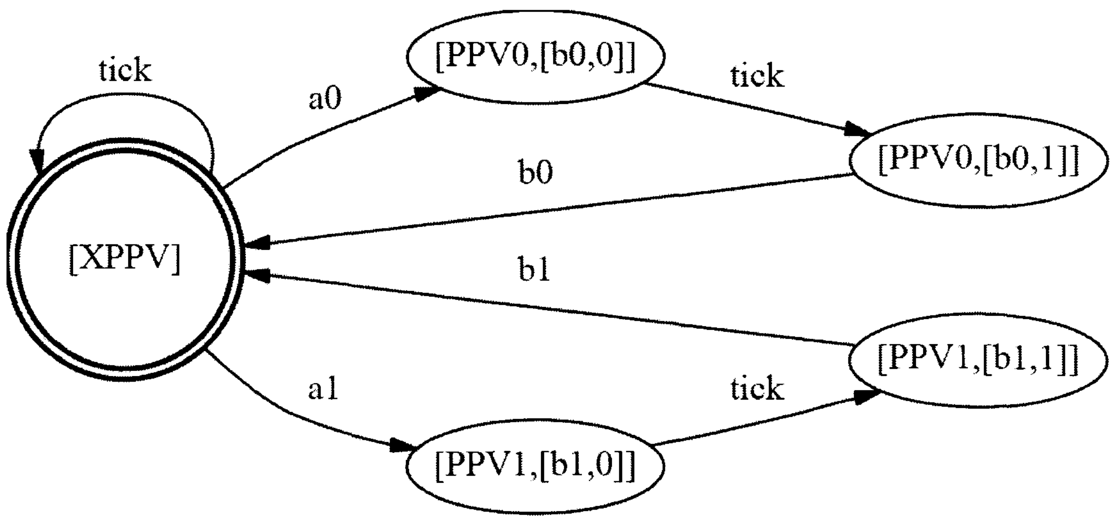

Figure 5-11: Scenario 2 Positive Pressure Ventilation Timed Transition Machine

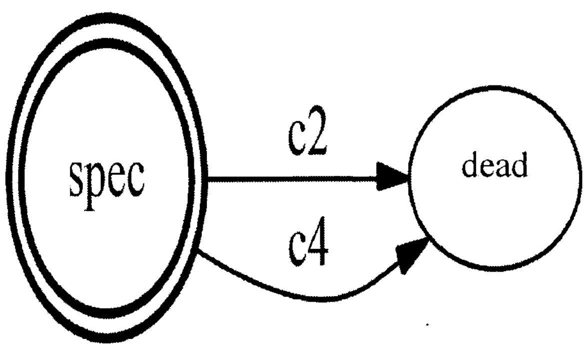

Figure 5-12: Scenario 2 Specification State Machine

5.4 Scenario 3: No Supervisory Controller

In scenario 2, the time bounds on the transitions from the neonate's abnormal to dead states were [3, inf], meaning the neonates will die after 3 or more ticks of the clock in the abnormal state when no positive pressure ventilation treatment is given. With time bounds of [1,1] in the positive pressure ventilation model, both neonates were able to be saved with the appropriate forcing and delaying of transitions. If one (or both) of the neonate's timebounds was changed to [2, inf], there is no supervisory controller that can guarantee the safety of both neonates. Although it would seem that both babies can be saved since positive pressure ventilation takes only 1 unit of time, this is not the case, because at the same time that ventilation is finished, the baby can die. This is too fine a point for a controller to guarantee safety.

The control data output from OTCT for this scenario is "The initial state of the supervisor is not defined" and the supervisory controller is empty, i.e. it has no transitions and no states. The supervisory controller state/transition map is as follows:

transition table: empty

state map: empty

Chapter 6

Conclusion and Future Work

6.1 Conclusion

When it is necessary to investigate the inner workings of certain processes, (i.e. to test for safety, efficiency, cost, etc.) it may be useful to model the processes as a discrete event simulation model and perform tests using the models rather than the real thing. Using a modeling technique such as discrete event simulation has many advantages, particularly when testing on the real world system is not possible, or not feasible from a practical or financial point of view. To provide verification of a system, discrete event systems can be used. This research represents the first steps in how one could model the NRP algorithm and a neonate's vital signs as a discrete event system, and how these models can be used to explore different childbirth scenarios. The models can be very simple or in-depth. This research has strived to keep the models as simple (i.e. small) as possible to avoid complex simulations, and to simply illustrate the wide range of possibilities that this modeling and verification technique possess.

This research discusses the use of simulation environments as both a training and a teaching (evaluation) tool. The Anakin is a life-like neonatal simulation tool that helps caregivers learn lifesaving skills, while allowing the trainer to evaluate a trainee's skills with the use of specialized software containing checklists, that can be reviewed and scored after a training session. It is unique in its application of the NRP algorithm.

6.2 Contributions to Research

The goals of this research as outlined in the Problem Statement have been reached. Specifically, the following contributions were made by the author:

1. The development of finite state machines and timed transition models to describe a neonate's vital signs and statistics in the following areas:

    (a) heart rate

    (b) skin tone

    (c) breathing

    (d) neck tilt

    (e) depth of chest compression

2. The development of finite state machines and timed transition models to describe the neonatal resuscitation procedure, which includes the following subprocedures:

    (a) initial steps

    (b) positive pressure ventilation

    (c) chest compressions

    (d) intubation

    (e) administering epinephrine

3. Verification of neonatal resuscitation scenarios, using the supervisory control provided by the OTCT software. This is the first time a "formal method" has been used to provide verification of an aspect of the NRP algorithm, where verification ensures 100% validation.

6.3 Future Work

In the discussion, discrete event systems were used to model the NRP and the neonate's vital signs. DES modeling appears to suit the NRP algorithm with its well defined state jumps. However, there are many aspects of a neonate's vital signs that would fall under the category of a continuous time system, or indeed a hybrid system with both continuous and discrete parts (see below for definitions). The lungs, metabolism, kidney functioning, etc. are all continuous time systems. In continuous systems theory, models of the dynamic system are typically described using differential and algebraic equations. There are many texts available on mathematical physiology, in which different physical systems of the body are described as mathematical models, i.e. differential and algebraic equations [21][40]. As future work, one may consider birth scenarios as continuous time systems, or as a combination of both continuous and discrete time systems. The following gives a brief description of continuous and hybrid systems.

6.3.1 The Continuous Time System

A continuous time system can be described by some differential equation

$$\dot{x} = f(x(t), u(t), t),\ x(0) = x_0$$
$$y(t) = g(x(t), u(t), t)$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the input vector and $y \in \mathbb{R}^r$ is the output vector. x_0 is the initial state.

An assumption of continuous time systems theory is that the functions f and g are Lipschitz continuous, or is said to satisfy a Lipschitz condition. That is, it moves Lipschitz-continuously from one real valued state to the next. Lipschitz continuity is an important condition in the existence and uniqueness theorem for ordinary differential equations. Mathematically, with respect to the state vector x, a function f satisfies the Lipschitz condition if a constant L exists such that

$$\|f(x,u,t) - f(\hat{x},u,t)\| \leq L \cdot \|x - \hat{x}\|$$

holds true for all x, $\hat{x}, u$ and t.

In contrast , the signals in discrete event systems have a discrete range. Signals may be binary values, or come from an finite or infinite value set. In DES, there are abrupt jumps (called events) leading from one state to another, whereas a continuous value system evolves through continuous movements. In many cases a continuous time system can be approximated as a discrete event system. (See below for details.)

6.3.2 The Hybrid System

Hybrid dynamical systems are systems that cannot be dealt with in terms of purely continuous variables or purely discrete events. Neither system's continuous movement nor its abrupt state jumps can be abstracted to either purely continuous or purely discrete variables. To do so would amount to the loss of important aspects of the system. Often times hybrid systems are dealt with by decomposing the system into its constituent parts. That is, dealing with the continuous subsystem separately from the discrete subsystem. Figure 6-1 shows a decomposed hybrid system where u is the continuous input, y is the continuous output, v is the discrete valued input and w is the discrete valued output. The quantizer and injector are the interfaces between the two subsystems.

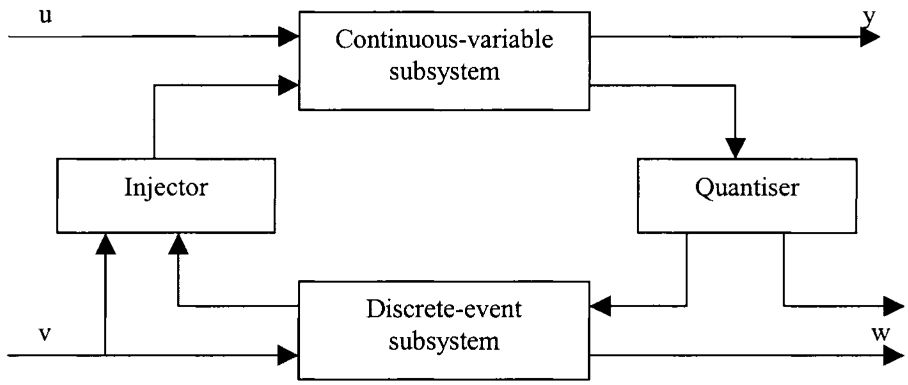

Figure 6-1: A Hybrid System Decomposed

6.3.3 Discrete Approximation of Continuous Systems

In some cases when dealing with hybrid systems, one can in fact represent the continuous system as an FSM. This could help reduce the complexity of the analysis or control problem, and allow one to use discrete event simulation and control software. Of course, this discrete model will not be an accurate representation of the true system, since it ignores its continuous aspects, however, it is up to the modeler to determine if the approximation fits his or her needs. In many cases, a discrete event approximation is perfectly valid. Similarly, one could approach the problem from a purely continuous point of view by reconstructing its discrete components. In [30], it states that the decision of whether to represent a hybrid system as wholly discrete or continuous depends upon its specifications: if the specifications are given in continuous variables, then system should be converted to a continuous system. If the specifications are in discrete terms, then the system should be represented as a discrete system. For this approach to be valid, it is required that the discrete time behavior of the continuous system (B_c) is contained in the behavior of the discrete model (B_d). I.e. B_c is a proper subset of B_d. According to [30], the smaller the difference between B_c and B_d, the more accurate the discrete approximation. If $B_c \subseteq B_d$ were not true, then the continuous system could respond to an input signal in such a way that would not be predicted by the discrete approximation. From a control point of view, this is clearly unacceptable.

The issue of approximating a continuous system in the discrete world is of special interest, since the software being used here for simulation and control is based on discrete event systems. Once a suitable discrete event representation has been made of the continuous system, it can be manipulated using the DES tools described in Chapter 3, such as the composition and trim operations.

The Continuous-Discrete World Interface

In order to represent a continuous system as a DES, there must be a clearly defined interface between the two. Given the function for a continuous time system, $\dot{x} = f(x(t), u(t), t)$, we see that there are three things required for the input interface (i.e. discrete to continuous world):

1. There must be a suitable mapping of the states, i.e. the discrete states, x_d must be mapped to continuous states, x_c.
2. The discrete inputs or events must be mapped to continuous inputs, u.
3. The continuous function f depends on time t, whereas a DES is asynchronous. Clocks generate continuous time. If more than one continuous time model is involved, their clocks must be synchronized.

There must also be an interface describing the output interface (i.e. continuous to discrete world):

1. Again, there must be a suitable mapping of the states, this time from continuous states to discrete states. This can be accomplished by dividing the state space into regions, which will define the discrete states. This may require a large number of regions, depending on the model and depending on the level of accuracy required. There must be enough regions such that events are generated in a manner that will allow the supervisory controller to make safe decisions.

2. The output from the continuous system needs to be modeled as discrete behavior (i.e. events). This would occur when the continuous model crosses from one of the regions described above into another region.

3. Continuous time is translated into the discrete world as a tick.

The details of a continuous-discrete interface is a research challenge that can be investigated in the future.

Appendix A

OTCT Code

A.1 Scenario 1a

A.1.1 baby.ttm

initial.

[normal1]. % The state "normal1" is the initial state.

marker.

[normal1]. % The state "normal1" is the marked state.

tran.[% Explanation: ex. [A, a, B]. In this example, there is a transition from state A to state B, called "a".

[normal1, c1, abnormal1],

[abnormal1, c2, dead1],

[abnormal1, b0, normal1],

[normal1, b0, normal1]].

timer. % The transition "c2" is timed, with a lower bound of 3 and an upper bound of infinity.

[c2, 3,inf].

A.1.2 ppv.ttm

```
initial.
    [XPPV].
    marker.
    [XPPV].
    tran.[
    [XPPV, a0, PPV0],
    [PPV0, b0, XPPV]].
    expeditible.
    [a0].     % The transition "a0" is expeditible, or forcible.
    timer.
    [b0, 1, 1].
```

A.1.3 spec.fsm

```
initial.
    [spec].
    marker.
    [spec].
    tran.[
    [spec, c2, dead]].
```

A.1.4 run.txt

```
ppv = ttm(ppv.ttm)
    printWithMap(ppv,ppv.pri)
    baby = ttm(baby.ttm)
    printWithMap(baby,baby.pri)
    plant = sync(baby, ppv)
```

```
printWithMap(plant,plant.pri)
spec = fsm(spec.fsm)
printWithMap(spec,spec.pri)
sup = supfcBySync(plant,spec)
printWithMap(sup,sup.pri)
condat(plant,sup,control.txt)
```

A.1.5 command.bat

```
otct run.txt
otct2dot plant.pri plant.dot
otct2dot ppv.pri ppv.dot
otct2dot baby.pri baby.dot
otct2dot spec.pri spec.dot
otct2dot sup.pri sup.dot
dot -Tps spec.dot -o spec.ps
dot -Tps sup.dot -o sup.ps
dot -Tps plant.dot -o plantt.ps
dot -Tps ppv.dot -o ppv.ps
dot -Tps baby.dot -o baby.ps
```

A.1.6 Plant State/Transition Map

transition table:

[0, 4, 15] [0, 8, 1] [0, 0, 0] [1, 4, 2] [1, 0, 4][2, 0, 3] [3, 2, 0] [4, 4, 5] [4, 0, 7] [5, 0, 6][6, 2, 0] [7, 4, 8] [7, 0, 11] [8, 0, 9] [9, 2, 0][9, 6, 10] [11, 4, 12] [11, 6, 14]

[11, 0, 11] [12, 6, 13][12, 0, 9] [13, 0, 10] [14, 4, 13] [14, 0, 14] [15, 8, 2][15, 0, 16] [16, 2, 0] [16, 8, 17] [17, 2, 0]

state map:

0 [[normal1],[XPPV]]

1 [[abnormal1,[c2,0]],[XPPV]]

2 [[abnormal1,[c2,0]],[PPV0,[b0,0]]]

3 [[abnormal1,[c2,1]],[PPV0,[b0,1]]]

4 [[abnormal1,[c2,1]],[XPPV]]

5 [[abnormal1,[c2,1]],[PPV0,[b0,0]]]

6 [[abnormal1,[c2,2]],[PPV0,[b0,1]]]

7 [[abnormal1,[c2,2]],[XPPV]]

8 [[abnormal1,[c2,2]],[PPV0,[b0,0]]]

9 [[abnormal1,[c2,3]],[PPV0,[b0,1]]]

10 [[dead1],[PPV0,[b0,1]]]

11 [[abnormal1,[c2,3]],[XPPV]]

12 [[abnormal1,[c2,3]],[PPV0,[b0,0]]]

13 [[dead1],[PPV0,[b0,0]]]

14 [[dead1],[XPPV]]

15 [[normal1],[PPV0,[b0,0]]]

16 [[normal1],[PPV0,[b0,1]]]

17 [[abnormal1,[c2,0]],[PPV0,[b0,1]]]

label map:

0 tick

2 b0

4 a0

6 c2

8 c1

A.1.7 Supervisory Controller State/Transition Map

transition table:

% Lists all of the transitions in the controller, using the same notation as above.

[0, 4, 7] [0, 6, 1] [0, 0, 0] [1, 4, 2] [1, 0, 4][2, 0, 3] [3, 2, 0] [4, 4, 5] [5, 0, 6] [6, 2, 0][7, 6, 2] [7, 0, 8] [8, 2, 0] [8, 6, 9] [9, 2, 0]

state map:

% These states are the result of the function supfcBySync(plant, spec) where plant is the synchronous product of the PPV and Baby TTMs.

0 [[[normal1],[XPPV]],spec]

1 [[[abnormal1,[c2,0]],[XPPV]],spec]

2 [[[abnormal1,[c2,0]],[PPV0,[b0,0]]],spec]

3 [[[abnormal1,[c2,1]],[PPV0,[b0,1]]],spec]

4 [[[abnormal1,[c2,1]],[XPPV]],spec]

5 [[[abnormal1,[c2,1]],[PPV0,[b0,0]]],spec]

6 [[[abnormal1,[c2,2]],[PPV0,[b0,1]]],spec]

7 [[[normal1],[PPV0,[b0,0]]],spec]

8 [[[normal1],[PPV0,[b0,1]]],spec]

9 [[[abnormal1,[c2,0]],[PPV0,[b0,1]]],spec]

label map:

0 tick

2 b0

4 a0

6 c1

A.2 Scenario 1b

A.2.1 baby.ttm

initial.

[normal1].

marker.

[normal1].

tran.[

[normal1, c1, abnormal1],

[abnormal1, c2, dead1],

[abnormal1, b0, normal1],

[abnormal1, a0, abnormal1],

[normal1, b0, normal1]].

timer.

[c2, 3,inf].

A.2.2 Plant State/Transition Map

transition table:

[0, 8, 1] [0, 0, 0] [1, 2, 2] [1, 0, 4] [2, 0, 3][3, 4, 0] [4, 2, 5] [4, 0, 7] [5, 0, 6] [6, 4, 0][7, 2, 8] [7, 0, 11] [8, 0, 9] [9, 4, 0][9, 6, 10][11, 2, 12] [11, 6, 14] [11, 0, 11]

[12, 6, 13] [12, 0, 9][13, 0, 10] [14, 0, 14]

state map:

0 [[normal1],[XPPV]]

% the first bracket represents the neonate, i.e. normal1

%the second bracket represents the PPV procedure, i.e. XPPV

1 [[abnormal1,[c2,0]],[XPPV]] % [c2,0] represents the clock associated with c2 while in the abnormal1 state.

2 [[abnormal1,[c2,0]],[PPV0,[b0,0]]]

3 [[abnormal1,[c2,1]],[PPV0,[b0,1]]]

4 [[abnormal1,[c2,1]],[XPPV]]

5 [[abnormal1,[c2,1]],[PPV0,[b0,0]]]

6 [[abnormal1,[c2,2]],[PPV0,[b0,1]]]

7 [[abnormal1,[c2,2]],[XPPV]]

8 [[abnormal1,[c2,2]],[PPV0,[b0,0]]]

9 [[abnormal1,[c2,3]],[PPV0,[b0,1]]]

% after 3 ticks of the clock in the abnormal1 state, the neonate will either die, or recover, but it cannot stay in the abnormal1 state.

10 [[dead1],[PPV0,[b0,1]]]

11 [[abnormal1,[c2,3]],[XPPV]]

12 [[abnormal1,[c2,3]],[PPV0,[b0,0]]]

13 [[dead1],[PPV0,[b0,0]]]

14 [[dead1],[XPPV]]

label map:

%these represent the transitions. Refer to the diagrams in Chapter 5.

0 tick % tick of the clock

2 a0 % The transition from XPPV to PPV0, i.e. starting the PPV routine.

4 b0 % The transition from PPV0 to XPPV after 1sec, i.e. stopping the PPV routine. Also the transition from the neonate's abnormal1 to normal1 state.

6 c2 % The transition between the abnormal1 state and the dead state (after 3 time units) in the neonate TTM

8 c1 % The transition from the normal1 to abnormal1 state.

A.2.3 Supervisory Controller State/Transition Map

transition table:

[0, 6, 1] [0, 0, 0] [1, 2, 2] [1, 0, 4] [2, 0, 3][3, 4, 0] [4, 2, 5] [5, 0, 6] [6, 4, 0]

state map:

%the first bracket is the neonate, the second represents PPV, and the third is the specification.

%refer to diagrams in Chapter 5

0 [[[normal1],[XPPV]],spec]

1 [[[abnormal1,[c2,0]],[XPPV]],spec]

2 [[[abnormal1,[c2,0]],[PPV0,[b0,0]]],spec]

3 [[[abnormal1,[c2,1]],[PPV0,[b0,1]]],spec]

4 [[[abnormal1,[c2,1]],[XPPV]],spec]

5 [[[abnormal1,[c2,1]],[PPV0,[b0,0]]],spec]

6 [[[abnormal1,[c2,2]],[PPV0,[b0,1]]],spec]

label map:

0 tick

2 a0

4 b0

6 c1

A.3 Scenario 2

A.3.1 baby1.ttm

initial.

[normal1].

marker.

[normal1].

tran.[

[normal1, c1, abnormal1],

[abnormal1, c2, dead1],

[abnormal1, b0, normal1],

[abnormal1, a0, abnormal1]].

timer.

[c2, 3,inf].

controllable.

[a0].

A.3.2 baby2.ttm

initial.

[normal2].

marker.

[normal2].

tran.[

[normal2, c3, abnormal2],

[abnormal2, c4, dead2],

[abnormal2, b1, normal2],

[abnormal2, a1, abnormal2]].

timer.

[c4, 3,inf].

controllable.

[a1].

A.3.3 ppv.ttm

marker.

[XPPV].

tran.[

[XPPV, a0, PPV0],

[PPV0, b0, XPPV],

[XPPV, a1, PPV1],

[PPV1,b1, XPPV]].

expeditible.

[a0, a1].

controllable.

[a0,a1].

timer.
[[b0, 1, 1],
[b1, 1, 1]].

A.3.4 spec.fsm

initial.
[spec].
marker.
[spec].
tran.
[[spec, c2, dead],
[spec, c4, dead]].

A.3.5 run.txt

ppv = ttm(ppv.ttm)
printWithMap(ppv,ppv.pri)
baby1 = ttm(baby1.ttm)
printWithMap(baby1,baby1.pri)
baby2 = ttm(baby2.ttm)
printWithMap(baby2,baby2.pri)
babysync = sync(baby1, baby2)
printWithMap(babysync,babysync.pri)
plant = sync(babysync, ppv)
printWithMap(plant,plant.pri)
spec = fsm(spec.fsm)
printWithMap(spec,spec.pri)
sup = supfcBySync(plant,spec)
printWithMap(sup,sup.pri)

condat(plant,sup,control.txt)

A.3.6 command.bat

otct run.txt

otct2dot plant.pri plant.dot

otct2dot ppv.pri ppv.dot

otct2dot baby1.pri baby1.dot

otct2dot baby2.pri baby2.dot

otct2dot babysync.pri babysync.dot

otct2dot spec.pri spec.dot

otct2dot sup.pri sup.dot

dot -Tps spec.dot -o spec.ps

dot -Tps sup.dot -o sup.ps

dot -Tps plant.dot -o plantt.ps

dot -Tps ppv.dot -o ppv.ps

dot -Tps baby1.dot -o baby1.ps

dot -Tps baby2.dot -o baby2.ps

dot -Tps babysync.dot -o babysync.ps

A.3.7 Plant State/Transition Map

transition table:

[0, 12, 1] [0, 6, 91] [0, 0, 0] [1, 1,141] [1, 6, 2][1, 0, 35] [2, 1, 3] [2, 3, 93] [2, 0,133] [3, 0, 4][4, 8, 5] [5, 3,130] [5, 12, 6][5, 0, 9] [6, 1, 7][6, 3,125] [6, 0,127]

[7, 0, 8] [8, 8, 9] [9, 3,124][9, 12, 10] [9, 0, 28] [10, 1, 11] [10, 3,123] [10, 0,106][11, 0, 12] [12, 8, 28] [12, 4, 13] [13, 8, 14] [14, 12, 15][14, 0, 14] [15, 1, 16]

[15, 0, 17] [16, 0, 13] [17, 1, 18][17, 0, 20][18, 0, 19] [19, 8, 14] [20, 1, 21] [20, 0, 24][21, 0, 22] [22, 8, 14] [22, 10, 23] [24, 1, 25] [24, 10, 27][24, 0, 24] [25, 10, 26]

[25, 0, 22] [26, 0, 23] [27, 0, 27][28, 3,117] [28, 12, 29] [28, 4, 14] [28, 0, 28] [29, 1, 30][29, 3, 31] [29, 4, 15] [29, 0,106][30, 4, 16] [30, 0, 12][31, 4, 32] [31, 0, 34]

[32, 0, 33] [34, 2, 35] [34, 4, 33][35, 1,103] [35, 6, 36] [35, 0, 41] [36, 1, 37][36, 3, 39][36, 0,100] [37, 0, 38] [38, 8, 5] [39, 0, 40] [40, 2, 41][41, 1, 99] [41, 6, 42]

[41, 0, 48] [42, 1, 43] [42, 3, 46][42, 0, 67] [43, 0, 44] [44, 8, 5] [44, 10, 45] [46, 0, 47][47, 2, 48] [47, 10, 54] [48, 1, 87] [48, 10, 55] [48, 6, 49][48, 0, 48][49, 1, 50]

[49, 3, 52] [49, 10, 56] [49, 0, 67][50, 10, 51] [50, 0, 44] [51, 0, 45] [52, 10, 53] [52, 0, 47][53, 0, 54] [54, 2, 55][55, 6, 56] [55, 0, 55] [56, 3, 53][56, 0, 57] [57, 3, 58]

[57, 0, 60] [58, 0, 59] [59, 2, 55][60, 3, 61] [60, 0, 64] [61, 0, 62][62, 2, 55] [62, 4, 63][64, 3, 65] [64, 4, 27] [64, 0, 64] [65, 4, 66] [65, 0, 62][66, 0, 63] [67, 1, 68]

[67, 3, 72] [67, 10, 57][67, 0, 74][68, 10, 71] [68, 0, 69] [69, 8, 9] [69, 10, 70] [71, 0, 70][72, 10, 58] [72, 0, 73] [73, 2, 48] [73, 10, 59] [74, 1, 75][74, 3, 79]

[74, 10, 60] [74, 0, 82] [75, 10, 78] [75, 0, 76][76, 8, 28] [76, 10, 77] [76, 4, 22] [77, 4, 23] [78, 0, 77][79, 10, 61][79, 0, 80] [80, 2, 48] [80, 10, 62] [80, 4, 81]

[81, 10, 63] [82, 1, 83] [82, 3, 85] [82, 10, 64] [82, 4, 24][82, 0, 82] [83, 10, 84][83, 4, 25] [83, 0, 76] [84, 4, 26][84, 0, 77] [85, 10, 65] [85, 4, 86] [85, 0, 80]

[86, 10, 66][86, 0, 81] [87, 10, 98] [87, 6, 50][87, 0, 88] [88, 8, 0][88, 10, 97] [88, 6, 89] [89, 8, 91] [89, 10, 90] [91, 3, 92][91, 12, 2] [91, 0, 5] [92, 12, 93] [92, 0, 95]

[93, 0, 94][94, 2, 35] [95, 2, 0] [95, 12, 96] [96, 2, 1] [97, 6, 90][98, 6, 51] [98, 0, 97] [99, 6, 43] [99, 0, 88] [100, 1,101][100, 3,102] [100, 0, 74] [101, 0, 69]

[102, 0, 73][103, 6, 37][103, 0,104] [104, 8, 0] [104, 6,105] [105, 8, 91] [106, 1,107][106, 3,109] [106, 4, 17] [106, 0,113] [107, 4, 18] [107, 0,108][108, 8, 28] [108, 4, 19]

[109, 4,110][109, 0,112] [110, 0,111][112, 2, 41] [112, 4,111] [113, 1,114] [113, 3,115] [113, 4, 20][113, 0, 82] [114, 4, 21] [114, 0, 76] [115, 4,116] [115, 0, 80]

[116, 0, 81] [117, 12, 31] [117, 4,122] [117, 0,118] [118, 2, 0][118, 12,119] [118, 4,121] [119, 2, 1] [119, 4,120] [121, 12,120][122, 12, 32] [122, 0,121] [123, 0, 34]

[124, 12,123] [124, 0,118][125, 0,126] [126, 2, 35] [127, 1,128] [127, 3,129] [127, 0,113][128, 0,108] [129, 0,112] [130, 12,125] [130, 0,131] [131, 2, 0][131, 12,132]

[132, 2, 1] [133, 1,134] [133, 3,136] [133, 0,138][134, 0,135] [135, 8, 9] [136, 0,137] [137, 2, 41] [138, 1,139][138, 3,140] [138, 0, 82] [139, 0, 76] [140, 0, 80] [141, 6, 3]

[141, 0,142] [142, 8, 0] [142, 6,143] [143, 8, 91]

state map:

% the first bracket is the combined behavior of the two neonates, the second bracket represents PPV.

0 [[[normal1],[normal2]],[XPPV]]

1 [[[abnormal1,[c2,0]],[normal2]],[XPPV]]

2 [[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[XPPV]]

3 [[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]]

4 [[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV0,[b0,1]]]

5 [[[normal1],[abnormal2,[c4,1]]],[XPPV]]

6 [[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[XPPV]]

7 [[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[PPV0,[b0,0]]]

8 [[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[PPV0,[b0,1]]]

9 [[[normal1],[abnormal2,[c4,2]]],[XPPV]]

10 [[[abnormal1,[c2,0]],[abnormal2,[c4,2]]],[XPPV]]

11 [[[abnormal1,[c2,0]],[abnormal2,[c4,2]]],[PPV0,[b0,0]]]

12 [[[abnormal1,[c2,1]],[abnormal2,[c4,3]]],[PPV0,[b0,1]]]

13 [[[abnormal1,[c2,1]],[dead2]],[PPV0,[b0,1]]]

14 [[[normal1],[dead2]],[XPPV]]

15 [[[abnormal1,[c2,0]],[dead2]],[XPPV]]

16 [[[abnormal1,[c2,0]],[dead2]],[PPV0,[b0,0]]]

17 [[[abnormal1,[c2,1]],[dead2]],[XPPV]]

18 [[[abnormal1,[c2,1]],[dead2]],[PPV0,[b0,0]]]

19 [[[abnormal1,[c2,2]],[dead2]],[PPV0,[b0,1]]]

20 [[[abnormal1,[c2,2]],[dead2]],[XPPV]]

21 [[[abnormal1,[c2,2]],[dead2]],[PPV0,[b0,0]]]

22 [[[abnormal1,[c2,3]],[dead2]],[PPV0,[b0,1]]]

23 [[[dead1],[dead2]],[PPV0,[b0,1]]]

24 [[[abnormal1,[c2,3]],[dead2]],[XPPV]]

25 [[[abnormal1,[c2,3]],[dead2]],[PPV0,[b0,0]]]

26 [[[dead1],[dead2]],[PPV0,[b0,0]]]

27 [[[dead1],[dead2]],[XPPV]]

28 [[[normal1],[abnormal2,[c4,3]]],[XPPV]]

29 [[[abnormal1,[c2,0]],[abnormal2,[c4,3]]],[XPPV]]

30 [[[abnormal1,[c2,0]],[abnormal2,[c4,3]]],[PPV0,[b0,0]]]

31 [[[abnormal1,[c2,0]],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

32 [[[abnormal1,[c2,0]],[dead2]],[PPV1,[b1,0]]]

33 [[[abnormal1,[c2,1]],[dead2]],[PPV1,[b1,1]]]

34 [[[abnormal1,[c2,1]],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

35 [[[abnormal1,[c2,1]],[normal2]],[XPPV]]

36 [[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[XPPV]]

37 [[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]]

38 [[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[PPV0,[b0,1]]]

39 [[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

40 [[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

41 [[[abnormal1,[c2,2]],[normal2]],[XPPV]]

42 [[[abnormal1,[c2,2]],[abnormal2,[c4,0]]],[XPPV]]

43 [[[abnormal1,[c2,2]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]]

44 [[[abnormal1,[c2,3]],[abnormal2,[c4,1]]],[PPV0,[b0,1]]]

45 [[[dead1],[abnormal2,[c4,1]]],[PPV0,[b0,1]]]

46 [[[abnormal1,[c2,2]],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

47 [[[abnormal1,[c2,3]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

48 [[[abnormal1,[c2,3]],[normal2]],[XPPV]]

49 [[[abnormal1,[c2,3]],[abnormal2,[c4,0]]],[XPPV]]

50 [[[abnormal1,[c2,3]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]]

51 [[[dead1],[abnormal2,[c4,0]]],[PPV0,[b0,0]]]

52 [[[abnormal1,[c2,3]],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

53 [[[dead1],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

54 [[[dead1],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

55 [[[dead1],[normal2]],[XPPV]]

56 [[[dead1],[abnormal2,[c4,0]]],[XPPV]]

57 [[[dead1],[abnormal2,[c4,1]]],[XPPV]]

58 [[[dead1],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

59 [[[dead1],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

60 [[[dead1],[abnormal2,[c4,2]]],[XPPV]]

61 [[[dead1],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

62 [[[dead1],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

63 [[[dead1],[dead2]],[PPV1,[b1,1]]]

64 [[[dead1],[abnormal2,[c4,3]]],[XPPV]]

65 [[[dead1],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

66 [[[dead1],[dead2]],[PPV1,[b1,0]]]

67 [[[abnormal1,[c2,3]],[abnormal2,[c4,1]]],[XPPV]]

68 [[[abnormal1,[c2,3]],[abnormal2,[c4,1]]],[PPV0,[b0,0]]]

69 [[[abnormal1,[c2,3]],[abnormal2,[c4,2]]],[PPV0,[b0,1]]]

70 [[[dead1],[abnormal2,[c4,2]]],[PPV0,[b0,1]]]

71 [[[dead1],[abnormal2,[c4,1]]],[PPV0,[b0,0]]]

72 [[[abnormal1,[c2,3]],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

73 [[[abnormal1,[c2,3]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

74 [[[abnormal1,[c2,3]],[abnormal2,[c4,2]]],[XPPV]]

75 [[[abnormal1,[c2,3]],[abnormal2,[c4,2]]],[PPV0,[b0,0]]]

76 [[[abnormal1,[c2,3]],[abnormal2,[c4,3]]],[PPV0,[b0,1]]]

77 [[[dead1],[abnormal2,[c4,3]]],[PPV0,[b0,1]]]

78 [[[dead1],[abnormal2,[c4,2]]],[PPV0,[b0,0]]]

79 [[[abnormal1,[c2,3]],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

80 [[[abnormal1,[c2,3]],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

81 [[[abnormal1,[c2,3]],[dead2]],[PPV1,[b1,1]]]

82 [[[abnormal1,[c2,3]],[abnormal2,[c4,3]]],[XPPV]]

83 [[[abnormal1,[c2,3]],[abnormal2,[c4,3]]],[PPV0,[b0,0]]]

84 [[[dead1],[abnormal2,[c4,3]]],[PPV0,[b0,0]]]

85 [[[abnormal1,[c2,3]],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

86 [[[abnormal1,[c2,3]],[dead2]],[PPV1,[b1,0]]]

87 [[[abnormal1,[c2,3]],[normal2]],[PPV0,[b0,0]]]

88 [[[abnormal1,[c2,3]],[normal2]],[PPV0,[b0,1]]]

89 [[[abnormal1,[c2,3]],[abnormal2,[c4,0]]],[PPV0,[b0,1]]]

90 [[[dead1],[abnormal2,[c4,0]]],[PPV0,[b0,1]]]

91 [[[normal1],[abnormal2,[c4,0]]],[XPPV]]

92 [[[normal1],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

93 [[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[PPV1,[b1,0]]]

94 [[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

95 [[[normal1],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

96 [[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]]

97 [[[dead1],[normal2]],[PPV0,[b0,1]]]

98 [[[dead1],[normal2]],[PPV0,[b0,0]]]

99 [[[abnormal1,[c2,2]],[normal2]],[PPV0,[b0,0]]]

100 [[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[XPPV]]

101 [[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[PPV0,[b0,0]]]

102 [[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

103 [[[abnormal1,[c2,1]],[normal2]],[PPV0,[b0,0]]]

104 [[[abnormal1,[c2,2]],[normal2]],[PPV0,[b0,1]]]

105 [[[abnormal1,[c2,2]],[abnormal2,[c4,0]]],[PPV0,[b0,1]]]

106 [[[abnormal1,[c2,1]],[abnormal2,[c4,3]]],[XPPV]]

107 [[[abnormal1,[c2,1]],[abnormal2,[c4,3]]],[PPV0,[b0,0]]]

108 [[[abnormal1,[c2,2]],[abnormal2,[c4,3]]],[PPV0,[b0,1]]]

109 [[[abnormal1,[c2,1]],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

110 [[[abnormal1,[c2,1]],[dead2]],[PPV1,[b1,0]]]

111 [[[abnormal1,[c2,2]],[dead2]],[PPV1,[b1,1]]]

112 [[[abnormal1,[c2,2]],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

113 [[[abnormal1,[c2,2]],[abnormal2,[c4,3]]],[XPPV]]

114 [[[abnormal1,[c2,2]],[abnormal2,[c4,3]]],[PPV0,[b0,0]]]

115 [[[abnormal1,[c2,2]],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

116 [[[abnormal1,[c2,2]],[dead2]],[PPV1,[b1,0]]]

117 [[[normal1],[abnormal2,[c4,3]]],[PPV1,[b1,0]]]

118 [[[normal1],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

119 [[[abnormal1,[c2,0]],[abnormal2,[c4,3]]],[PPV1,[b1,1]]]

120 [[[abnormal1,[c2,0]],[dead2]],[PPV1,[b1,1]]]

121 [[[normal1],[dead2]],[PPV1,[b1,1]]]

122 [[[normal1],[dead2]],[PPV1,[b1,0]]]

123 [[[abnormal1,[c2,0]],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

124 [[[normal1],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

125 [[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

126 [[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

127 [[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[XPPV]]

128 [[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[PPV0,[b0,0]]]

129 [[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

130 [[[normal1],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

131 [[[normal1],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

132 [[[abnormal1,[c2,0]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

133 [[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[XPPV]]

134 [[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV0,[b0,0]]]

135 [[[abnormal1,[c2,2]],[abnormal2,[c4,2]]],[PPV0,[b0,1]]]

136 [[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV1,[b1,0]]]

137 [[[abnormal1,[c2,2]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]]

138 [[[abnormal1,[c2,2]],[abnormal2,[c4,2]]],[XPPV]]

139 [[[abnormal1,[c2,2]],[abnormal2,[c4,2]]],[PPV0,[b0,0]]]

140 [[[abnormal1,[c2,2]],[abnormal2,[c4,2]]],[PPV1,[b1,0]]]

141 [[[abnormal1,[c2,0]],[normal2]],[PPV0,[b0,0]]]

142 [[[abnormal1,[c2,1]],[normal2]],[PPV0,[b0,1]]]

143 [[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[PPV0,[b0,1]]]

label map:

0 tick

1 a0

2 b1

3 a1

4 c4

6 c3

8 b0

10 c2

12 c1

A.3.8 Supervisory Controller State/Transition Map

transition table:

[0, 8, 1] [0, 4, 16] [0, 0, 0] [1, 1, 25] [1, 4, 2][1, 0, 9] [2, 1, 3] [2, 3, 18] [3, 0, 4] [4, 6, 5][5, 3, 22] [5, 8, 6] [6, 3, 7] [7, 0, 8] [8, 2, 9][9, 1, 13] [9, 4, 10] [10, 1, 11]

[11, 0, 12] [12, 6, 5][13, 4, 11] [13, 0, 14] [14, 6, 0] [14, 4, 15] [15, 6, 16][16, 3, 17] [16, 8, 2] [16, 0, 5] [17, 8, 18] [17, 0, 20][18, 0, 19] [19, 2, 9] [20, 2, 0] [20, 8, 21]

[21, 2, 1][22, 8, 7] [22, 0, 23] [23, 2, 0] [23, 8, 24] [24, 2, 1][25, 4, 3] [25, 0, 26] [26, 6, 0] [26, 4, 27] [27, 6, 16]

state map:

0 [[[[normal1],[normal2]],[XPPV]],spec]

1 [[[[abnormal1,[c2,0]],[normal2]],[XPPV]],spec]

2 [[[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[XPPV]],spec]

3 [[[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]],spec]

4 [[[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV0,[b0,1]]],spec]

5 [[[[normal1],[abnormal2,[c4,1]]],[XPPV]],spec]

6 [[[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[XPPV]],spec]

7 [[[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[PPV1,[b1,0]]],spec]

8 [[[[abnormal1,[c2,1]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]],spec]

9 [[[[abnormal1,[c2,1]],[normal2]],[XPPV]],spec]

10 [[[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[XPPV]],spec]

11 [[[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[PPV0,[b0,0]]],spec]

12 [[[[abnormal1,[c2,2]],[abnormal2,[c4,1]]],[PPV0,[b0,1]]],spec]

13 [[[[abnormal1,[c2,1]],[normal2]],[PPV0,[b0,0]]],spec]

14 [[[[abnormal1,[c2,2]],[normal2]],[PPV0,[b0,1]]],spec]

15 [[[[abnormal1,[c2,2]],[abnormal2,[c4,0]]],[PPV0,[b0,1]]],spec]

16 [[[[normal1],[abnormal2,[c4,0]]],[XPPV]],spec]

17 [[[[normal1],[abnormal2,[c4,0]]],[PPV1,[b1,0]]],spec]

18 [[[[abnormal1,[c2,0]],[abnormal2,[c4,0]]],[PPV1,[b1,0]]],spec]

19 [[[[abnormal1,[c2,1]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]],spec]

20 [[[[normal1],[abnormal2,[c4,1]]],[PPV1,[b1,1]]],spec]

21 [[[[abnormal1,[c2,0]],[abnormal2,[c4,1]]],[PPV1,[b1,1]]],spec]

22 [[[[normal1],[abnormal2,[c4,1]]],[PPV1,[b1,0]]],spec]

23 [[[[normal1],[abnormal2,[c4,2]]],[PPV1,[b1,1]]],spec]

24 [[[[abnormal1,[c2,0]],[abnormal2,[c4,2]]],[PPV1,[b1,1]]],spec]

25 [[[[abnormal1,[c2,0]],[normal2]],[PPV0,[b0,0]]],spec]

26 [[[[abnormal1,[c2,1]],[normal2]],[PPV0,[b0,1]]],spec]

27 [[[[abnormal1,[c2,1]],[abnormal2,[c4,0]]],[PPV0,[b0,1]]],spec]

label map:

0 tick

1 a0

2 b1

3 a1

A.4 Scenario 3

A.4.1 baby1.ttm

initial.

[normal1].

marker.

[normal1].

tran.

[

[normal1, c1, abnormal1],

[abnormal1, c2, dead1],

[abnormal1, b0, normal1],

[abnormal1, a0, abnormal1]

].

timer.

[c2, 2,inf].

controllable.

[a0].

A.4.2 Supervisory Controller State/Transition Map

transition table: empty

state map: empty

References

[1] American Academy of Pediatrics/ American Heart Association: Textbook of Neonatal Resuscitation. American Heart Association 2000.

[2] H. Babad, C. Sanderson, B. Naidoo, I. White, D. Wang. The Development of a Simulation Model of Primary Prevention Strategies for Coronary Heart Disease. *Health Care Management Sciences*, Vol. 5, pp. 269-274, 2002.

[3] P. Barton, S. Bryan, S. Robinson. Modelling in the Economic Evaluation of Health Care: Selecting the Appropriate Approach. *Journal of Health Services Research and Policy*, Vol. 9, No. 2, pp. 110-118, April 2004.

[4] B. A. Brandin, W. M. Wonham. The Supervisory Control of Timed Discrete-Event Systes. *Proceedings of the 31st Conference on Decision and Control*, Tucson, Arizona, pp. 3357-3362, December 1992.

[5] D. Carbine, N. Finer, E. Knodel, W. Rich. Video recording as a means of evaluating neonatal resuscitation performance. *Pediatrics*, Vol. 106, No. 4, pp. 654-658, 2000.

[6] C. Cassandras, S. Lafortune. Introduction to Discrete Event Systems. Springer, 1999.

[7] J.V. Chancellor, A. M. Hill, C.A. Sabin, K.N. Simpson,M. Youle M. Modelling the cost effectiveness of lamivudine/zidovudine combination therapy in HIV infection. *Pharmacoeconom,* Vol. 12, pp. 54-66, 1997.

[8] K Cooper, R Davies, P. Roderick, D Chase and J Raftery. The development of a simulation model for the treatment of coronary artery disease. *Health Care Management Science*, Vol. 5, pp. 259-267, 2002.

[9] C. Cronin, S. Cheang, D. Hlynka, E. Adair, S. Roberts. Videoconferencing can be used to assess neonatal resuscitation skills. *Medical Education*, Vol. 35, No. 11, pp. 1013-1023, November 2001.

[10] V. Curran, K. Aziz, S. O'Young, C. Bessell. Evaluation of the Effect of a Computerized Training Simulator (Anakin) on the Retention of Neonatal Resuscitation Skills. *Teaching and Learning in Medicine*, Vol. 16, No. 2, pp. 157-164, 2004.

[11] V. Curran, K. Aziz, S. O'Young, C. Bessell, H. Schulz. A comparison of face-to-face versus remote assessment of neonatal resuscitation skills. *Journal of Telemedicine and Telecare*, Vol. 11, pp. 97-102, 2005.

[12] J.L. Doob. Stochastic Processes. John Wiley and Sons, New York, N.Y., 1953

[13] S. Dunne, P. Niday, N.E. Watters, P. MacGrath, D. Alcock. The provision and evaluation of a neonatal resuscitation program. *Journal of Continuing Education for Nurses,* Vol. 23, No. 3, pp. 118-126, May-June 1992.

[14] B. Eaton, D. Messent, I. Haywood. Animal cadaveric models for advanced trauma life support training. *Annals of the Royal College of Surgeons of England*, Vol. 72, pp. 135-139, 1990.

[15] Gaumard. *PEDI Blue Neonatal Simulator with SmartSkin Technology.* (n.d.). Retrieved Dec. 01, 2005 from www.gaumard.com.

[16] M. K. Gilbart, C. R. Hutchison, M. D. Cusimano, G. Regehr. A computer-based trauma simulator for teaching trauma management skills. *The American Journal of Surgery*, Vol. 179, pp. 223-228, 2000.

[17] J.A. Gordon, W. M. Wilkerson, D. W. Shaffer, E. G. Armstrong. Practicing medicine without risk: students and educators response to high-fidelity patient simulation. *Academic Medicine*, Vol. 76, No. 5, pp. 469-472, 2001.

[18] L. Halamek, D. Kaegi, D. Gaba, Y. Sowb, B. Smith, S. Howard. Time for a new paradigm in pediatric medical education: Teaching neonatal resuscitation in a simulated delivery room environment. *Pediatrics*, Vol. 106, No. 4, pp. 1-6, 2000.

[19] International Guidelines for Neonatal Resuscitation: An Excerpt From the Guidelines 2000 for Cardiopulmonary Resuscitation and Emergency Cardiovascular Care: International Consensus on Science. *Pediatrics,* Vol 106, No. 3, September 2000.

[20] C. R. Kaufmann. Computers in Surgical Education and the Operating Room. *Annales Chirugiae et Gynaecologiae,* Vol. 90, pp. 141-146, 2001

[21] J. Keener, J. Sneyd. Mathematical Physiology. Springer, New York, 1998.

[22] D. Korosec, L.P. Halamek, D. Zazula. Making Virtual Baby Alive. *Proceedings of the 23rd Annual EMBS International Conference*, Istanbul, Turkey, pp. 4052-4055, 2001.

[23] Laerdal Medical. *Laerdal Yesterday and Today.* (n.d.). Retrieved Dec. 02, 2005 from www.laerdal.com.

[24] C. Levitt, J. Kaczorowski, E. Outerbridge, V. Jiminez, B. Connolly, B. Slapcoff. Knowledge gained following neonatal resuscitation program courses. *Family Medicine*, Vol. 26, No. 6, pp. 403-406, 1996.

[25] J. Lunze. What is a Hybrid System? *Modelling, Analysis and Design of Hybrid Systems Lecture Notes in Control and Information Sciences*, Vol. 279, pp. 3-14, 2002.

[26] J. Lunze, J. Raisch. Discrete Models for Hybrid Systems. *Modelling, Analysis and Design of Hybrid Systems Lecture Notes in Control and Information Sciences*, Vol. 279, pp. 67-80, 2002.

[27] Medical Education Laboratories Inc. *PediaSIM Pediatric Simulator Clinical Features.* (n.d.). Retrieved Dec. 02, 2005 from www.meti.com.

[28] T. Murata. Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, 77(4), pp. 541-579, 1989.

[29] Object TCT: User's Guide. Condata Technologies Ltd. February 1996.

[30] J. Raisch, S. D. O'Young. Discrete Approximation and Supervisory Control of Continuous Systems. *IEEE Transactions on Automatic Control*, Vol. 43, No. 4, pp. 569-573, April 1998.

[31] P. Ramadge, W. M. Wonham. The Control of Discrete Event Systems. *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 81-97, January 1989.

[32] Ramamurthy. 1986. *Simulator for Teaching Neonatal Resuscitation.* U.S. Patent # 434/265. 4,611,998.

[33] C.A. Ryan, L.M. Clark, A. Malone, S. Ahmed. The effect of a structured neonatal resuscitation program on delivery room practices. *Neonatal Network*, Vol. 18, No. 1, pp. 25-30, February 1999.

[34] Schertz et al. 1996. *Interactive Neonatal Resuscitation Training Simulator and Method.* U.S. Patent # 432/262. 5,509,810.

[35] M. Schlindwein, G. v. Wagner, M. Kirst, M. Rajeweiz, F. Karl, J. Schochlin, A. Bolz. Mobile Patient Simulator for Resuscitation Training with Automatic External Defibrillators. *Biosignale*, Vol. 47, pp. 559-560, 2002.

[36] S. Shechter, J. Kreke, A. Schaefer, D. Angus, C. Bryce, M. Roberts. A Clinically Based Discrete Event Simulation of End Stage Liver Disease and the Liver Allocation Process. *Medical Decision Making*, Vol 25, No. 2, pp. 199-209, 2004.

[37] M. Skidmore, H. Urquhart. Retention of skills in neonatal resuscitation. *Pediatric Child Health,* Vol. 6, No. 1, pp. 31-35, 2001.

[38] J.E. Stahl, M.S. Roberts, S. Gazelle. Optimizing Management and Financial Performance of the Teaching Ambulatory Care Clinic. *Journal of General Internal Medicine*, Vol 18, pp. 266-274, April 2003.

[39] J. R. Swisher, S. H. Jacobson, J. B. Jun, O. Balci. Modeling and Analyzing a Physician Clinic Environment Using Discrete-Event (Visual) Simulation. *Computers & Operations Research*, Vol. 28, pp. 105-125, 2001.

[40] J. B. West. Respiratory Physiology 3rd Edition. Williams and Wilkins, Baltimore, 1985.

[41] R. Wilson. Neonatal Resuscitation Software for PDA. Engr. 8800 Project Report, Memorial University, May 2003.