Testing of Safety-Critical Software Embedded in an Artificial Heart

Sungdeok Cha[1], Sehun Jeong[1], Junbeom Yoo[2] and Young-Gab Kim[1]

[1]Korea University, Seoul, Korea

[2]Konkuk University, Seoul, Korea

Abstract Software is being used more frequently to control medical devices such as artificial heart or robotic surgery system. While much of software safety issues in such systems are similar to other safety-critical systems (e.g., nuclear power plants), domain-specific properties may warrant development of customized techniques to demonstrate fitness of the system on patients. In this paper, we report results of a preliminary analysis done on software controlling a Hybrid Ventricular Assist Device (H-VAD) developed by Korea Artificial Organ Centre (KAOC). It is a state-of-the-art artificial heart which completed animal testing phase. We performed software testing in in-vitro experiments and animal experiments. An abnormal behaviour, never detected during extensive in-vitro analysis and animal testing, was found.

1 Introduction

The number of patients who suffer from heart-related disease is increasing rapidly. The American Heart Association estimates that about 800,000 American deaths were due to heart-related disease in 2006, exceeding casualties caused by cancer and accidents (AHA 2010). Organ transplant is only a partially effective solution because a patient's immune system may reject a transplanted heart, and because the number of heart donations simply cannot satisfy the demand. According to the Organ Procurement and Transplantation Network and Scientific Registry of Transplant Recipients, only 2,277 heart donations were made in the US in 2006 (HHS 2008, 2010).

To the vast majority of cardiac patients, an artificial heart is the only practical alternative to extend precious life. In 2004, the US Food and Drug Administration (FDA) approved clinical use of artificial hearts (FDA 2004), and nearly 850 patients received artificial hearts manufactured by SynCardia (SynCardia Systems 2010). However, to the best of our knowledge, there have been no attempts to validate the safety of software embedded in an artificial heart in a clinical environment. With active cooperation by biomedical researchers, we performed validation of software embedded in the H-VAD system (Jeong et al. 2009, Lee et al.

C. Dale, T. Anderson (eds.), *Advances in Systems Safety*, DOI 10.1007/978-0-85729-133-2_9,

2009). After gaining an in-depth understanding of source code including its architecture, we measured code coverage using probing statements, performed in-vitro[1] testing, and discovered two abnormal behaviours previously unknown to KAOC staff despite extensive 'in-vitro system validation' and animal experiments. We recently completed the initial phase of animal testing[2] whose primary objective was to replicate newly discovered abnormal behaviours.

This paper is organized as follows. In Section 2, we briefly explain the H-VAD system with emphasis on software architecture. In Sections 3, we report results observed in software testing of artificial heart software. We describe both in-vitro and animal testing. Section 4 concludes the paper and suggests future work.

2 H-VAD artificial heart

H-VAD is a portable artificial heart which set the record of 183 days of successful operation on a calf. This record is sufficient to satisfy FDA regulations on long-term experiments. Primary features of KAOC H-VAD are:

- It is easier to carry than other portable artificial heart systems. It combines two traditional methods to drive pump: air-driven and electric motor-driven. The motor-driven actuator generates air pressure necessary to pump blood without any additional air compressor or vacuum pump.
- The system has two control parameters, pumping rates (PRs) and stroke lengths (SLs), whose values can be changed by pressing buttons. The former sets the heart beat rate while the latter controls the volume of ejected blood.

The system consists of one or two blood pumps and a control device. As depicted in Figures 1 and 2, the control device consists of the following components:

Pneumatic pump. The electric motor moves pusher plates back and forth to produce air flow to the blood pump via a closed-loop air tube. Subsequent valve movement in the blood pump complements a less than fully functional heart. Three motor hall sensors provide the input necessary to compute the current direction and speed of pusher plate movement. Another pump hall sensor captures crossing of the pusher plates over the centre of the pump.

Display. Essential information (e.g. operation mode, pump status) is shown on the LCD display.

Control panel. Six buttons, including start and stop, allow users to change the pump setting.

[1] A procedure performed not in a living organism but in a controlled environment.

[2] The animals were treated in accordance with *The Guide for the Care and Use of Laboratory Animals* issued by Korea University School of Medicine.

CPU. A TMS320 F2810 microprocessor, a Texas Instrument chip, executes the embedded software.

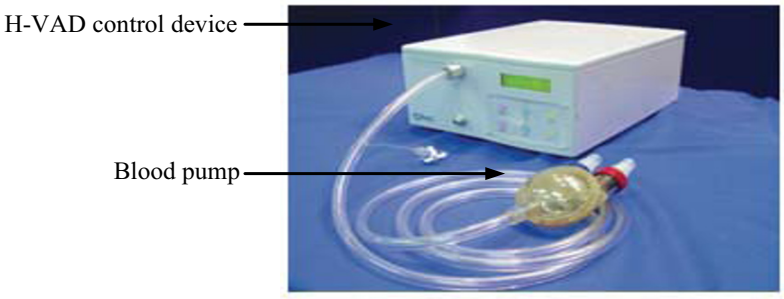

Fig. 1. KAOC H-VAD

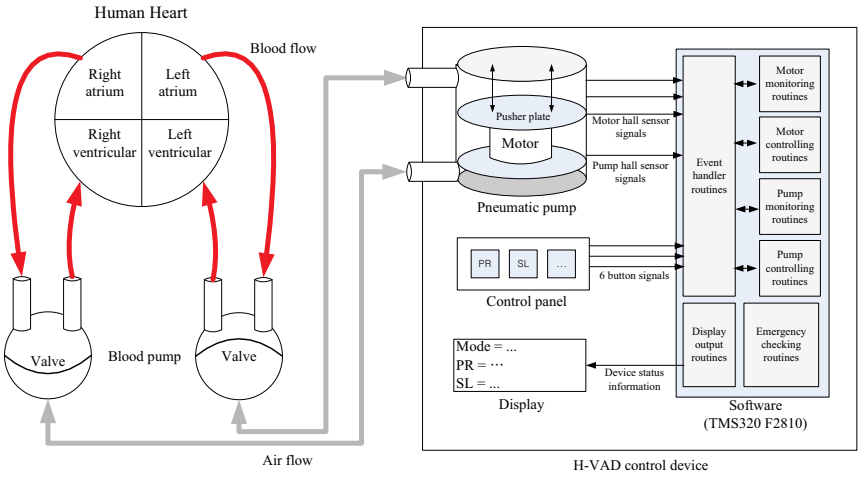

Fig. 2. Hardware context diagram of H-VAD

H-VAD software monitors and controls the motor to keep the pump's movement stable according to the current PR and SL setting. PR is calculated based on the number of pusher plate movements per minute, and SL is calculated on the distance between the two end points.

There are about 9,000 lines of C code, including vendor-supplied skeleton code, scattered in over 20 files. Logic unique to KAOC H-VAD is implemented in about 3,800 lines of code organized in an event-driven architecture, with four interrupt handlers tracking motor speed and plate movements, six routines processing button inputs, and a timer routine.

Two emergency modes of operation are built into the system. One is triggered if there is no motor hall sensor signal input for 500 milliseconds, and the other occurs when the pump actuator does not pass the centre position in 3,000 milliseconds.

3 Testing the H-VAD software

Our quality assurance effort on H-VAD software consists of two parts: code coverage measurement in in-vitro environment using blood-like liquid and 'live' animal testing conducted in close consultation with KAOC staff and in compliance to the relevant guideline.

3.1 In-Vitro environment: testing and code coverage measurement

When our project began, the H-VAD system had already been fully developed, and animal testing successfully kept a calf alive for 183 days. However, no analysis had been conducted with a particular emphasis on software. Our first task was to understand the system requirements and develop in-depth knowledge of the software logic through code review. We monitored execution, using 211 probing statements we added, in an in-vitro environment to determine which program paths have been taken. Because all branch conditions are relatively simple, no differences existed among several coverage criteria (e.g. branch or condition coverage) defined in the software testing literature. We made sure that probing statements did not alter program semantics.

While recording execution traces, we manipulated the H-VAD setting by repeatedly pressing buttons so that the software would exercise as many branches as possible. Figure 3 illustrates how we systematically prepared test cases based on the following guidelines:

- Each button, the only medium through which users may influence system behaviour, is pressed at least once.
- All permitted parameter values are covered, and an attempt was made to set the value outside the possible range. For example, stroke length may vary from 30 to 90 while the 'typical' setting is 60.
- The stop button was pressed at arbitrary and random moments to simulate as many exceptional situations as practicable and at various pump positions.
- We tried to force the system to engage in both predefined emergency modes.

Analysis of execution traces revealed that 170 of 211 probe statements (80.6%) had been executed. Further analysis of uncovered probes revealed the following:

- Some statements are logically unreachable. For example, the switch statement had true and false paths to follow, but the programmer, for some unknown reason, had implemented yet another default path to follow in case neither path is taken. Statements in the default clause could never be executed in the absence of abnormal control errors.
- The possibility of executing some statements is permanently determined by the hardware configuration. For example, execution of some *switch-case* statement is based on a control variable indicating deviation of the centre hall sensor installation position from the actual centre position of the pump. Such deviation never occurs unless one performs physical reconstruction of the pneumatic pump.

ID	Test inputs			Precondition				Related Guide line	Description
		PR	SL	Control mode	Running mode	Pump position	Pump direction		
TC1	START	50	60	PR	STOP	Center	N/A	1	Baseline
TC2	Plus (10 times)	50	60	PR	RUNNING	N/A	N/A	2	Increasing pump rate
TC3	Plus	150	60	PR	RUNNING	N/A	N/A	3	Upper boundary checking for pump rate
TC4	Minus (10 times)	50	60	PR	RUNNING	N/A	N/A	2	Decreasing pump rate
TC5	Minus	5	60	PR	RUNNING	N/A	N/A	3	Lower boundary checking for pump rate
TC6	SL	50	60	PR	RUNNING	N/A	N/A	1	Changing the control mode
TC7	Plus (10 times)	50	60	SL	RUNNING	N/A	N/A	2	Increasing stroke length
TC8	Plus	50	80	SL	RUNNING	N/A	N/A	3	Upper boundary checking for stroke length
TC9	Minus (10 times)	50	60	SL	RUNNING	N/A	N/A	2	Decreasing stroke length
TC10	Minus	50	30	SL	RUNNING	N/A	N/A	3	Lower boundary checking for stroke length
TC11	PR	50	60	SL	RUNNING	N/A	N/A	1	Changing the control mode
TC12	Plus (10 times)	50	60	PR	STOP	N/A	N/A	2	Same with TC2 but in stopped mode
TC13	Plus	150	60	PR	STOP	N/A	N/A	3	Same with TC3 but in stopped mode
TC14	Minus (10 times)	50	60	PR	STOP	N/A	N/A	2	Same with TC4 but in stopped mode
TC15	Minus	5	60	PR	STOP	N/A	N/A	3	Same with TC5 but in stopped mode
TC16	SL	50	60	PR	STOP	N/A	N/A	1	Same with TC6 but in stopped mode
TC17	Plus (10 times)	50	60	SL	STOP	N/A	N/A	2	Same with TC7 but in stopped mode
TC18	Plus	50	80	SL	STOP	N/A	N/A	3	Same with TC8 but in stopped mode
TC19	Minus (10 times)	50	60	SL	STOP	N/A	N/A	2	Same with TC9 but in stopped mode
TC20	Minus	50	30	SL	STOP	N/A	N/A	3	Same with TC10 but in stopped mode
TC21	PR	50	60	SL	STOP	N/A	N/A	1	Same with TC11 but in stopped mode
TC22	STOP	N/A	N/A	N/A	RUNNING	Top	Up	4	Pressing STOP button when the pump is going up from the center position
TC23	STOP	N/A	N/A	N/A	RUNNING	Center	Up	4	Pressing STOP button when the pump is passing the center position and the direction is up.
TC24	STOP	N/A	N/A	N/A	RUNNING	Bottom	Up	4	Pressing STOP button when the pump is going up from the bottom
TC25	STOP	N/A	N/A	N/A	RUNNING	Top	Down	4	Pressing STOP button when the pump is going down from the top
TC26	STOP	N/A	N/A	N/A	RUNNING	Center	Down	4	Pressing STOP button when the pump is passing the center position and the direction is down.
TC27	STOP	N/A	N/A	N/A	RUNNING	Bottom	Down	4	Pressing STOP button when the pump is going down from the center position
TC28	Disable center hall sensor	N/A	N/A	N/A	RUNNING	N/A	N/A	5	Triggering emergency mode 1
TC29	Block air valve	N/A	N/A	N/A	RUNNING	N/A	N/A	5	Triggering emergency mode 2

Fig. 3. Test cases for the code coverage test

While measuring code coverage in the in-vitro environment, we found two abnormal behaviours of the H-VAD system while executing the test cases highlighted in Figure 3. Our collaborators, biomedical engineering research staff at KAOC who developed the system and conducted 'system-level testing' as well as animal experiments, were unaware of such behaviour. They agreed that the H-VAD system appeared to be malfunctioning.

First, we noticed an unintended excessive pumping, as illustrated in Figure 4, while executing the test case TC9 shown in Figure 3. The monitored PR value is abruptly increased when Ref_Left_MAX_Velocity, one of several variables used in computing motor speed, is set to a negative value. This change flipped the shape of the curve used in motor speed decision. Sudden and excessive pumping may yield critical, if not fatal, consequences to a ventricular system in clinical use due to the increased blood flow and tension.

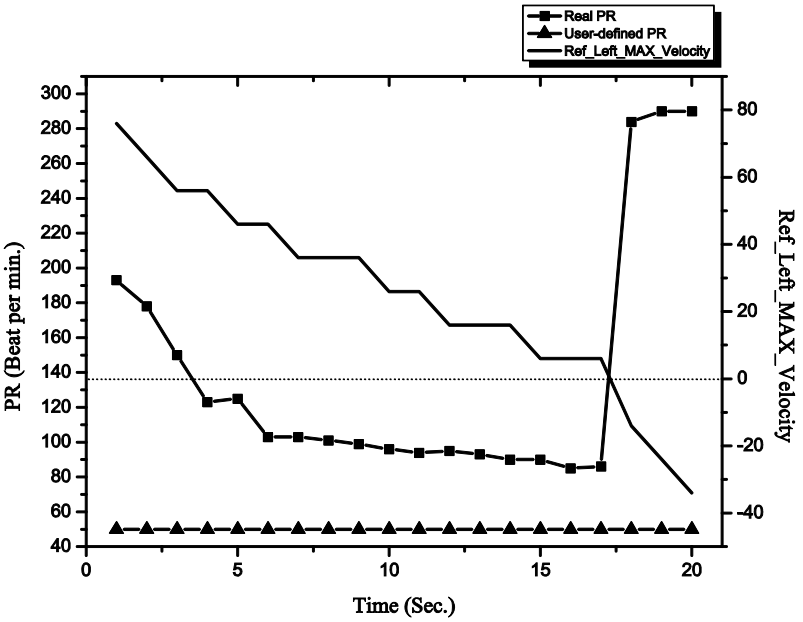

Fig. 4. Unintended, sudden, and excessive pumping

The other anomaly, shown in Figure 5, involves recurrent oscillation of pump speed where actual pump rate is simply unable to match the specified rate. Test case TC2 from Figure 3 uncovered this anomaly. This pattern was 'audibly' apparent as the pump kept trying to meet but ended up exceeding the target value and repeated a similar effort in the reverse direction. It appears that the H-VAD control algorithms failed properly to deal with subtle and exceptional scenarios in sufficient detail.

However, we must not forget that these anomalies took place in a realistic but not actual environment. While the in-vitro circulatory system mimics the human body (e.g. fluid characteristics closely match those of human blood), there are

fundamental gaps between the two. It is difficult, if not impossible, to accurately reproduce the dynamic and continuous physical changes occurring inside a human body using a static in-vitro system. Still, the anomaly we discovered was serious enough to warrant further investigation in a more realistic environment, and an animal test was scheduled.

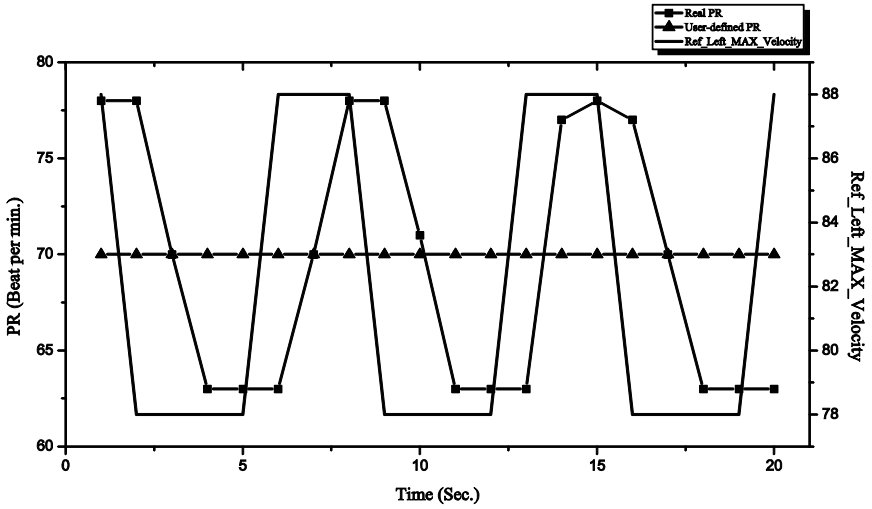

Fig. 5. Recurrent oscillation of pump rate

3.2 H-VAD software testing in an animal experiment

The primary objectives of animal testing were to compare code coverage in the clinical environment against that recorded in the in-vitro setting, and to determine if the two abnormal behaviours we uncovered in the in-vitro setting would occur in the clinical environment. Prior to the animal experiment, in addition to obtaining proper approval, we carefully developed and documented scenarios to repeat the anomalies. In addition to logging the values of critical variables used in the software, we also logged all the critical information, including vital biometric signals such as blood pressure and flows, using sensors attached to an animal. Furthermore, we used a video camera to record all relevant information during an animal experiment to enable accurate post-mortem analysis. See Figure 6.

When measuring code coverage, we tried to the best of our ability to maximize coverage while not endangering the life of the testing subject (a piglet). In the in-vitro setting, we could arbitrarily change the PR and SL values, but in the animal experiment updates made to the PR and SL values could result in rapid and potentially fatal changes in the blood flow and pressure of the animal. Such constraints limited the feasible test cases we could execute in the animal experiment. Furthermore, behaviour of the H-VAD system, even in the identical setting, is differ-

ent between animal and in-vitro environments. For example, an attempt to set the SL value to the permitted minimum failed in the animal experiment, as it kept triggering an emergency mode; this had not previously happened.

Fig. 6. Animal experiment setting

In the animal experiment, we observed that 166 out of 211probing statements (78.7%) were covered. This small difference in coverage was caused by our inability to repeat certain test cases so as not to endanger the animal's life. We therefore concluded that there was virtually no difference in code coverage between the two experimental settings.

In the animal experiment, we were unable to recreate situations where excessive pumping occurred despite our complete understanding of the software logic and the test sequences which led to the anomaly. We knew that Ref_Left_MAX_ Velocity would assume a negative value, in the software configuration we tested, if the actual pump rate was much larger than the user specified PR value. While we were able to increase the real PR value fairly rapidly in the in-vitro environment with no difficulty, such a phenomenon did not occur in the animal experiment. Figure 7 illustrates our effort. Our domain experts explained that the root cause of the difference lies in the strong blood tension in the live circulatory system and the subsequent decrease in the pump rate.

One must remember that our inability to reproduce this anomaly does not necessarily guarantee that the system is completely free from such an anomaly. However, considering the physical and fundamental differences in operational environments, it appears fairly safe to claim that such an anomaly would not occur in clinical usage.

On the other hand, we were able to recreate the other anomaly in which the H-VAD system was unable to converge to the specified setting. Figure 8 illustrates how this anomaly is revealed in the blood pressure graph. Partial success of H-VAD software validation in the animal experiment clearly reveals that further analysis of software correctness is absolutely necessary prior to clinical use.

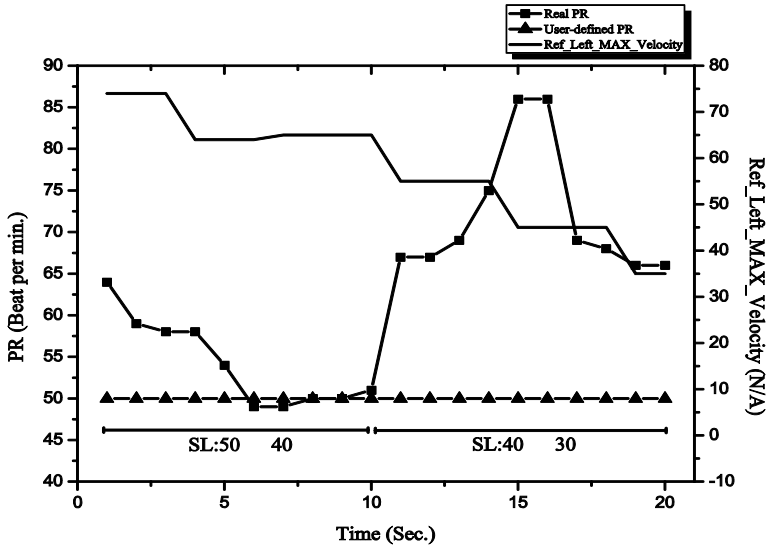

Fig. 7. An attempt to recreate the excessive pumping anomaly

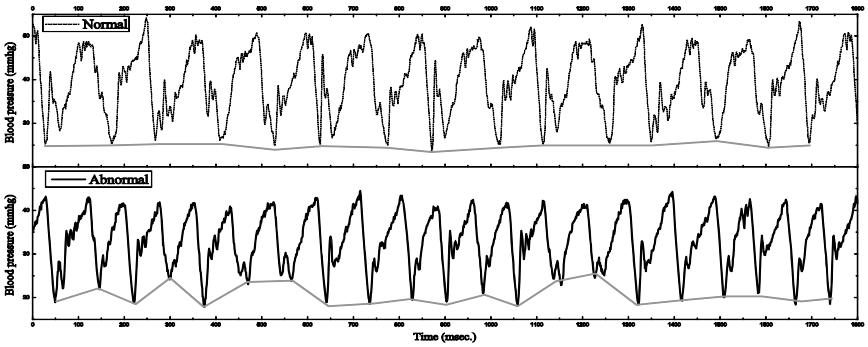

Fig. 8. Oscillation of pump rate revealed in irregular blood pressure

4 Conclusions

Use of software in medical devices, especially in life-critical and clinical settings, is beginning and will increase in the future. In practice, as well as artificial hearts, a robotic surgery system is another notable example where software has direct and immediate impact on a patient's life. Such trends, however, will surely increase and appear to be irreversible. In order to ensure the safe operation of such devices, researchers in the software engineering and software safety communities need to work more closely with domain experts. The research reported in this paper could

not have been carried out without the active participation and enthusiastic support of physicians and biomedical engineers.

One must develop systematic approaches to safety-critical software validation while minimizing casualties, including animals used in clinical testing. Real world validation through deployment, like command and control systems in military war game systems, is often impossible. Yet, one must develop approaches to develop credible safety assurance.

While it is absolutely necessary, animal testing is labour-intensive and costly. In order to conduct the animal testing reported in this paper, three MDs, including a surgeon, and several assistants had to offer their expertise. The preparatory surgical operation alone took nearly six hours. Each animal used in an experiment has different and unique vital characteristics. Therefore, one must be careful not to overly generalize results from animal experiments.

Our experience proves once again how difficult it is to make a credible claim on software quality especially in a safety-critical setting. Although highly useful and essential, testing of safety-critical software embedded in medical devices in an in-vitro setting has limitations.

We are planning various activities to continue our research on software safety issues in medical systems. One possibility includes model-driven development (MDD) of software. Once a formal model is developed, test cases might be generated automatically, and model checking techniques could be applied. One might able to synthesize source code as has been accomplished in other domains (e.g. the nuclear industry).

Numerous significant challenges must be resolved to achieve technical breakthrough. Faithful and accurate modelling of the environment seems to be one of the most difficult challenges. In the case of an artificial heart, one must mathematically model all the essential details of the human circulatory system.

Acknowledgments This research was supported by the National IT Industry Promotion Agency (NIPA) under the program of Software Engineering Technologies Development. The authors would like to express thanks to research staff at the Korea Artificial Organ Center. Mr Chi-bum Ahn provided extraordinary support in our quest to understand the H-VAD system and validate its software safety.

References

AHA (2010) Heart disease and stroke statistics 2010 update at-a-glance. American Heart Association. http://www.americanheart.org/downloadable/heart/1265665152970DS-3241%20HeartStrokeUpdate _2010.pdf. Accessed 25 August 2010

Desikan S, Ramesh G (2006) Software testing: principles and practice. Pearson Education

FDA (2004) Recently-approved devices 2004 device approvals. U.S. Food and Drug Administration. http://www.fda.gov/MedicalDevices/ProductsandMedicalProcedures/DeviceApprovalsandClearances/RecentlyApprovedDevices/ucm073321.htm. Accessed 25 August 2010

HHS (2008) The 2008 annual report of the OPTN and SRTR: heart transplantation. US Department of Health and Human Services. http://www.ustransplant.org/annual_reports/current/. Accessed 25 August 2010

HHS (2010) Organ procurement and transplantation network national data. US Department of Health and Human Services. http://optn.transplant.hrsa.gov/latestData/step2.asp. Accessed 25 August 2010

Jeong G, Hwang C, Nam K, Ahn C, H Kim, Lee J, Choi J, Son H et al (2009) Development of a closed air loop electropneumatic actuator for driving a pneumatic blood pump. Artif Organs 33:657-662

Lee J, Kim B, Choi J, Choi H, Ahn C et al (2009) Optimal pressure regulation of the pneumatic ventricular assist device with bellows-type driver. Artif Organs 33:627-633

SynCardia Systems (2010) About SynCardia Systems, Inc. http://www.syncardia.com/SynCardia/about-syncardia.html. Accessed 25 August 2010