A prototype expert system for analysis and diagnosis of single chamber paced ECGs

By M.P.T. Chin Kwie Joe

M.Sc. thesis
Carried out from August 1996 to June 1997

In order of:
Under supervision of: Dr. Ir. P. Cluitmans
Advised by: Dr. Ir. J.A. Blom
Dr. L.M. Van Gelder

The department of Electrical Engineering of the Eindhoven University of Technology accepts no responsibility for the contents of M.Sc. theses or reports on practical training periods
Abstract

A rule-based expert system has been proposed that will assist clinicians in the often difficult task of analyzing multi-channel pacemaker electrocardiograms (ECGs). Analysis of the pacemaker ECGs is important to the follow-up evaluation of patients with implanted pacemakers. Because of the complexity and variability of pacemaker algorithms, diagnosis of pacemaker ECGs is often considerably more difficult than the interpretation of usual ECGs. However, comparatively little work has been done in this area, mainly since the diversity and complexity of pacemaker logic makes interpretation a difficult task.

The proposed prototype expert system on interpretation of the pacemaker ECG can provide great clinical benefit because few clinicians are adequately trained in the diagnosis of such ECGs for the interpretation of pacemaker functionality. The expert system guides the clinicians through the analysis of the pacemaker ECGs during a follow-up. The program bases its conclusion on information it receives from the user's response to questions the system poses regarding specific characteristic of the ECG waveform during multiple cycles. The clinician simply answers the question with Yes or No.

The system uses a top-down method to analyze the ECG information. The system evaluates two domain specific tasks during the interaction with the user. First, the system starts with analyzing the pacemaker ECGs for pacemaker malfunctions (no output, intermittent output, noncapture, intermittent capture, oversensing or undersensing). When the malfunction type is identified, the system starts the second task: diagnosis of the pacemaker ECGs. During this task the expert system tries to determine the cause of the pacemaker malfunction (e.g., lead-electrode fracture, pacemaker configuration setting, pacemaker malfunction etc.). The expert system gives a solution for the pacemaker malfunction when the cause of malfunction is found.

Evaluation of this system by medical experts demonstrates that it mimics an instructional assistant in a consistent and reliable manner. Although the system has not been tested extensively, preliminarily tests show that the prototype could identify all the presented test cases.

The expert system described in this report is still a prototype. The prototype contains the domain specific knowledge of the single chamber pacemaker. The domain specific knowledge of dual chamber pacemakers is not yet implemented in the system. Thus, further development on the expert system is necessary.
Contents

1 Introduction .................................................. 5
   1.1 Status of the ‘pacemaker expert system project’ 5
   1.2 The objectives of this study 6
   1.3 The outline of this report 8

2 Expert systems .................................................. 9
   2.1 Why an expert system 9
   2.2 The Structure of an expert system 10
   2.3 Production rules 12
   2.4 Designing an expert system 13

3 Heart physiology and stimulation .................................. 15
   3.1 The electrocardiogram 15
      3.1.1 Anatomy of the heart 15
      3.1.2 Origin of the electrocardiogram 16
      3.1.3 Indication for pacing 17
      3.1.4 ECG Measurements 18
      3.1.5 Nomenclature ECG 19
      3.1.6 The paced ECG 20
   3.2 The pacemaker 21
      3.2.1 The pacemaker lead 22
      3.2.2 Pacemaker programming modes 23
      3.2.3 Pacemaker malfunctions 24

4 Applied knowledge acquisition methods .......................... 27
   4.1 Project setup 27
   4.2 The knowledge acquisition process 28
      4.2.1 Project management 28
      4.2.2 KADS 29
      4.2.3 Rapid Prototyping 30
      4.2.4 Heuristic matching 31

5 The analyzed domain knowledge .................................... 33
   5.1 Decomposition of the knowledge domain 34
   5.2 Decomposition of the problem-solving strategy of the domain expert 35
   5.3 The reasoning strategy of the prototype expert system 39
   5.4 The Push-Pop function 41
Development of the third prototype expert system

6.1 Development tools
   6.1.1 Simplexys
   6.1.2 Delphi

6.2 The prototype expert system
   6.2.1 The user interface

6.3 System design of the prototype expert system

6.4 The current status of the project

Test and evaluation of the prototype expert system

Conclusions and Recommendations

Glossary

References

Appendix A: The help function for the questions.


Appendix D: The Push-Pop function.

Appendix E: The implemented knowledge and strategy used in the first and second prototype.
1
Introduction

The human heart is the natural pump of the human body. The heart pumps blood to all organs of the body and provides oxygen and nutrients to them. Due to heart diseases the pumping abilities of the heart may decline. People with symptomatic bradycardia, where the cardiac output does not meet physiologic demands, are often candidates for cardiac pacing.

The chance for pacemaker malfunctions in modern pacing devices is very slim, but there is always a potential risk for the patient. Therefore, pacemaker function should be verified regularly during a pacemaker follow-up. The difficult verification of the pacemaker function is largely based on the analysis of the electrocardiogram (ECG) of the patient.

The interpretation of a pacemaker ECG is, particularly for beginning diagnosticians, a difficult task because interpretation of pacemaker ECGs is mainly based on empirical and associative knowledge. This type of knowledge is not structured and consists mostly of rules of thumb.

To solve this problem the section Medical Engineering of the University of Technology Eindhoven and the department for Cardiac Catheterization and Pacemaker Clinic of the Catharina Hospital in Eindhoven have developed a prototype expert system. This prototype expert system guides a diagnostician during a pacemaker follow-up. The functionality of the current prototype is limited to single chamber pacemakers.

1.1 Status of the ‘pacemaker expert system project’

The initial goal of the construction of an expert system for analysis and diagnosis of the function of permanent pacemakers was to check how systematical and consistent Dr. L. van Gelder’s approach was in detecting a faulty pacemaker function. The development of the first prototype expert system (1994), by Ir. R. Bourgonje, proves that it is possible to capture the domain specific knowledge described by Dr. L. van Gelder in his Ph.D. thesis (1995): ‘The ECG in the evaluation of pacemaker function and diagnosis of malfunction.’

The aim of this project is to use the expert system as an interactive educational learning tool. The ‘pacemaker expert system’ project is finished when it contains all the domain knowledge from Van Gelder’s thesis. After completion, the most likely course is integration of the expert system in pacemaker programming equipment. This integration enables the expert system to collect pacemaker specifications and telemetry data directly. This data can be used in the

---

1 The domain expert: Dr. L. van Gelder. Head of the Department for Cardiac Catheterization and Pacemaker clinic in the Catharina hospital, board member of the Dutch Pacemaker Registry and consultant of the Dutch Heart Foundation.
reasoning process, which can be shortened thereby and lead the ECG interpreter to the cause of malfunction with minimal interaction with the user [Van Gelder, 1995].

A realization of the above-mentioned utilization of the expert system will take many years of research and development. Thus it is expected that many graduation students from the University of technology Eindhoven will work on this major project. I am already the third graduation student in a row working on this project.

**The first prototype**

According Van Gelder the first prototype performed at a very acceptable level. Despite the quality of the prototype, many cases presented to the expert system could not be identified. This was because only part of Van Gelder's thesis had been implemented by Bourgonje. The first prototype contains only parts of the domain knowledge about the analysis and diagnosis of the function of single chamber pacemakers. Therefore, the next step was to complete the prototype expert system for detecting faulty single chamber pacemaker functions. For a description of the first prototype see [Bourgonje, 1994].

**The second prototype**

Ir. P. Wouters was the second student who worked on this project (1997). He developed a second prototype expert system. This prototype is not an extension on the first prototype, but it is another expert system. It used a completely different approach. The second prototype contains only the analysis pacemaker ECG interpretation. The first prototype however was more elaborate than the second prototype.

The justification for this new approach is that, according Wouters, the first prototype is difficult to maintain, update and expand. Thus, a proposal was made to develop an easily maintainable expert system. Consistent with this proposal a new prototype expert system was developed. The second prototype had one major drawback: Van Gelder's reasoning strategy is not recognizable in the second prototype. According to Van Gelder the second prototype is impractical. So it is not recommended to follow this different approach. For a description of the second prototype see [Wouters, 1997].

**1.2 The objectives of this study**

By studying both prototype expert systems the following question arises: What are the objectives of the expert system? In other words, what should the expert system do and which conditions have to be satisfied?

Bourgonje stated that an expert system for analysis and diagnosis of the function of permanent pacemakers should satisfy the following three goals [Bourgonje, 1994]:

1. The expert system should assist the diagnosticians in the evaluation of pacemaker functions.
2. The expert system has to be educational, so that diagnosticians can use the system as an interactive learning tool.

3. The expert system’s answers/conclusions must be unambiguous.

On top of those goals Wouter’s formulated three other goals:

4. The ‘break in’ of the knowledge domain has to be cut down. It takes too much time to understand the basics of pacemaker evaluation.

5. The expert system should be easy to maintain and update. Both the first and second prototypes are difficult to maintain and update.

6. During development of the expert system the user-requests (requests from the domain expert) on the user-friendliness of the system should not be discarded.

The goals formulated by Bourgonje are related to the potential users of the expert system. Those goals are the objectives for the finished pacemaker expert system. The three goals devised by Wouters focus on the technical structure of the expert system. Those goals are related to the knowledge engineer.

The six goals of my predecessors are useful directives for further work on the prototype expert system.

For the third prototype expert system the following goals are added to the list of goals:

7. The third prototype expert system should contain the decision-making strategy of the domain expert. The first prototype is able to simulate the domain expert very well, but the prototype is not able to switch from one task to another the way a physician does. The second prototype fails to reproduce the reasoning strategy of the domain expert. In the third prototype that problem should be solved.

8. The acquired domain knowledge in the first and second prototypes has to be used in the third prototype expert system. Otherwise, the development of the expert system will not progress. This does not mean that the knowledge should be formulated in the same fashion as in the first two prototypes.

9. The objectives for the third prototype expert system are that it has to be a user-friendly and educational system. This prototype should contain the domain knowledge for the analysis and diagnosis of the function of single chamber pacemakers, described in Van Gelder’s thesis.
1.3 The outline of this report

The introduction of the pacemaker project is explained in this chapter. The second chapter explains the basic components of an expert system and why an expert system is used for this project. Background knowledge on heart physiology and pacemaker basics are explained in the subsequent chapter. Chapter 4 describes the techniques used for acquiring the knowledge from the domain expert. How the domain knowledge and reasoning strategy of the domain expert are structured is explained in chapter 5. Chapter 6 shows the user-interface of the third prototype expert system. The evaluation of the prototype expert system is described in chapter 7. The conclusions on the system are noted in chapter 8. The recommendations for the prototype expert system are also described in chapter 8. The recommendations are important for the development of a new prototype expert system.
Expert systems are a product of artificial intelligence, the branch of computer science concerned with developing programs that exhibit intelligent behavior.

An expert system is a computer program that relies on knowledge and reasoning to perform a difficult task usually undertaken only by a human expert. Just as a human expert has knowledge of a specific field, say, interpretation of ECGs, an expert system has a knowledge base consisting of knowledge relating to a specific field. Human experts reason and arrive at conclusions based on their knowledge; expert systems reason and arrive at conclusions based on the knowledge they possess.

Jackson [Jackson, 1990] stated the following definition for expert systems:

- An expert system is a computer program that represents and reasons with knowledge of some specialist subject with a view to solving problems or giving advice.

### 2.1 Why an expert system

According to Durking [Durking, 1994] a major part of applications for expert systems is in the field of medicine. Expert systems are particularly suitable in this field, because of the empirical associative knowledge used in medical science. This type of knowledge can relatively easily be implemented in an expert system.

Assisting a human expert is the most commonly found application of expert systems. In this type of application, the system aids the expert in a routine or difficult task. For example, a physician may have knowledge of pacemaker malfunctions, but, due to the rare occurrence of pacemaker malfunctions and the extensive number of possible malfunctions, it would be a difficult task to identify the source of the malfunction. The physician could benefit from the support provided by an expert system, by guiding the physician quickly to isolate the source of the malfunction.

Some principal reasons expert systems are developed to assist diagnosticians are:

- If existing experts are expensive and scarce
- For aiding diagnosticians in some routine task to improve productivity
- For aiding diagnosticians in some difficult task to effectively manage the complexities
Another application of expert systems is to replace a human expert. Some of the principal reasons expert systems are developed to replace an expert are:

- Automate a routine task requiring an expert
- No expert available
- Expert is too expensive

The ‘pacemaker expert system’ project focuses on the first application described in this section. Due to the complexity, size and fuzziness of the knowledge domain developing an expert system for the second application is very difficult. Although, when the ‘pacemaker expert system’ project is completed, the expert system can then be used for developing a new (real-time) expert system for automated pacemaker ECG interpretation.

For a study of applications on expert systems in medicine see [Witte, 1987]. Witte discusses in his report some successful expert systems in the field of medicine. For an analysis on clinical reasoning see [Wullf, 1980]. Wullf’s report outlines how medical problems are solved by physicians.

2.2 The Structure of an expert system

The expert system used in this project is composed of a user interface, a database, a knowledge base, an explanation facility and an interference engine. For the development of the expert system the ‘expert system building tool Simplexys’ is used. The Simplexys tools have been developed at the University of Technology Eindhoven. Simplexys is a collection of tools to design real time expert systems [Blom, 1990]. This project for pacemaker ECG interpretation does not develop a real time expert system but Bourgonje has proven, with his first prototype that Simplexys can be used successfully in this project.
The user-interface
The user interface provides for the communication exchange between the user and the expert system. Through the user interface, the user can enter facts about specific pacemaker specifications and the answers to the questions given by the expert system. The interface is an important factor for the acceptance of the expert system by the user. User friendliness is achieved by a simple Windows interface. Bourgonje developed a window-based interface under the DOS operating system. For the first prototype the DOS interface was adequate. However for the third prototype, which should be a system for demonstration purposes, the DOS interface has too many limitations. Two versions of the third prototype were developed. A DOS version and a Windows 3.11 version. The DOS version has the interface as developed by Bourgonje. The Windows version has the same functionality as the DOS version but extra tools are added to the interface. See paragraph 6.2 for a description of the Windows interface. For an explanation on the DOS interface see [Bourgonje, 1994].

The knowledge base
The expert system knowledge base contains the expert-level knowledge on pacemaker ECG interpretation. This knowledge is obtained from the domain expert Van Gelder and is stored in a knowledge-representational form that is inherent to the expert system design. The domain knowledge is represented with a Simplexys rule, a sort of production rule. See paragraph 2.3 for an explanation on production rules.

The inference engine
The inference engine performs the reasoning tasks for the expert system. The inference engine uses the knowledge in the expert system’s knowledge base and information provide by the user to infer new knowledge.

The knowledge base and the inference engine are the core of every expert system. The sharp distinction between domain knowledge and problem-solving methodology (the inference engine) is fundamental to the nature of every expert system.

The explanation facility
The explanation function allows the user to ask the expert system to justify the answer or the advice the system has given. The expert system justifies its answers or advice by explaining the reasoning. The system provides an ordered list of rules and facts it used to formulate its answer.

The database
The database normally contains two types of data:
- Static data: Facts and relations found in the problem domain; e.g., the pacemaker specification data.
- Dynamic data: Data, collected during a session; e.g., the pacemaker functions (for the pacemaker functions see chapter 5).
2.3 Production rules

After acquiring the knowledge from the expert on some well-focused domain, the knowledge has to be encoded in the expert system. To do this, we will need to find a way of structuring the knowledge to allow the system to solve a problem in a manner similar to problem solving strategy of the expert. This is also called knowledge representation. A way to represent the domain knowledge is with production rules.

Rule-based problem-solving systems are built using the following rule syntax:

\[ P_1, \ldots, P_m \rightarrow Q_1, \ldots, Q_n. \]

with the reading

If premises \( P_1 \) and \( \ldots \) and \( P_m \) are true then perform actions \( Q_1 \) and \( \ldots \) and \( Q_n \).

The premises are sometimes called 'conditions', and the actions 'conclusions'.

Forward-chaining/data-driven inference is an inference strategy that begins with a set of known facts (conditions), derives new facts (conclusions) using rules whose premise matches the known facts, and continues this process until a goal state is reached or until no further rules have premises that match the known or derived facts.

Forward-chaining is a good inference technique if we are working with a problem that requires us to begin with information and then derive logical conclusions. In other problems, we begin with a hypothesis and then attempt to prove it by gathering supporting information. This is called backward-chaining or goal driven search.

**example: backward-chaining**

If a physician identifies a varying prolongation of the escape interval during a pacemaker follow-up on a patient with a VVI pacemaker, then the physician has to check two possible hypotheses which can cause a varying prolongation. He attempts to prove one of those hypotheses (and disprove the other one) by looking for certain symptoms related to the hypothesis.

The majority of the rules in the knowledge base of the prototype expert system are backward-chaining rules. Approximately one of every ten rules in the knowledge base uses forward chaining. The reason for this is that backward-chaining remains focused on a given goal. This produces a series of related topics, a situation that is comfortable for the user. Whereas a forward-chaining rule attempts to infer everything possible from the available information, backward-chaining system searches only that part of the knowledge base that is relevant to the current problem.

---

2 A varying prolongation of the escape interval with an implanted VVI pacemaker is an indication for P wave oversensing or lead malfunction.
Designing an expert system

The building of an expert system may encounter several problems, which can result in an expert system that will never be used. Selecting a domain or problem that is suitable for expert system support is important. According to Martin [Martin,1988] the following tasks should be considered for expert system development:

- The proposed expert system will save time or money or will enable a task to be performed in a substantially better way; the system will truly make a difference to the organization.
- An expert is available.
- Nonexperts require the expertise.
- The task is fully described with facts, rules, and algorithms.
- The expertise to be modeled is narrowly focused.

If there is no expert available then there is no knowledge available from which the expert system can copy its reasoning strategy. Van Gelder is a renowned specialist in the field of pacemaker ECG interpretation. His knowledge respectively his reasoning strategy of the knowledge domain is accepted by most other specialists in this field of medicine.

I gave a demonstration of the third prototype expert system at a Symposium for pacemaker technicians, where 90 percent of the Dutch pacemaker clinics were represented. This resulted in a positive response by the pacemaker technicians. Some of the technicians even volunteer to help with the development. A literature study on this subject shows that since 1992 no new developments took place on pacemaker ECG diagnostics. A literature study done by Bourgonje [Bourgonje, 1994] confirms this. The response at the symposium and the conclusion of the literature studies indicate that there is demand for such systems.

The domain knowledge has to be well defined. Only then can the knowledge be captured in a knowledge-base. The first and second prototypes prove that it is possible to encode most of the knowledge of 'pacemaker ECG interpretation' in facts and rules.

To achieve success in performing difficult tasks, we need to focus on narrow domains and use knowledge relevant to these domains. The technique of focusing on narrow problems is helpful for two reasons; first it reduces the overall complexity of the situation, and secondly it allows the use of domain-specific knowledge.

A survey on successful expert system development done by 'Werkgroep Projectselectie van Expertsystemen' (WPE) in 1989 among 128 Dutch organizations showed that only a third of

---


4 Dr. SE Greenhut is the most important scientist in this field. After 1992 he did not publish any more articles on interpretation on paced ECG [Greenhut, 1991], [Greenhut, 1992].
the developed expert systems will be operational. The WPE-report concluded that the following points, described below, will contribute to the development of a successful expert system.

users of the expert system
- the users should be involved in the development of the system
- the users should have a positive attitude toward the development of the system
- the system should enhance the quality of work
- do not forget the needs of the users

project approach
- team work
- the management should support the development of the system
- the development of the system should be done in phases
- a good knowledge acquisition method is crucial

The feedback by potential users of the expert system is important. During validation and testing of the third prototype by some potential users, questions were asked about the user-friendliness of the system. The user response on the Windows 3.1 interface for the third prototype was good. See chapter 7 for more information on the evaluation of the system.

The pacemaker expert system should be an educational interactive computer program for starting diagnosticians. Nowadays there are only study books and practicals to learn about pacemaker ECG interpretation. Both study methods are important but there is a big gap between both study approaches. The expert system can be used for self-education and for practicals where the system can advise the student. The system will enhance the work for nonspecialist in this field.

WPE concluded in its report that it is not important which knowledge acquisition method is used. It is only important that a method is used. Chapter 4 describes the methodologies used during the development of the third prototype expert system.

communication with the domain expert
The communication with the domain expert is one of the major ‘bottlenecks’ in acquiring knowledge for the expert system. For instance, much of the early interaction between the expert and the knowledge engineer may be spent in educating each other about their separate kinds of expertise. To facilitate the communication with the domain expert, the expert should know about the basics of the technologies and methodologies of expert systems and the knowledge engineer has to use a methodology for dealing with knowledge. The building of an expert system will then be smoother and more efficient.

For an elaborated discussion of the above-mentioned guidelines for building an expert system see [WPE,1989].
3
Heart physiology and stimulation

3.1 The electrocardiogram

Perhaps the electromyogram which is most often measured, and which has the greatest clinical relevance, is the potential produced by the most powerful muscle in the human body; the heart. This potential is called the electrocardiogram, or ECG, and is frequently studied because it tells the cardiologist a great deal about the function (and malfunction) of the heart. The ECG is a very complex signal that has many components. Before we try to understand where these components are produced, we must have a basic understanding of the structure of the human heart.

3.1.1 Anatomy of the heart

The heart pumps blood to all the organs of the body. The blood carries oxygen and nutrients to these organs and removes waste products from them. The heart muscle is a specialized type of muscle (called cardiac muscle), which has interesting electrical properties that will be discussed subsequently. The heart is a four-chambered organ that acts as two pumps tied in parallel; the two pumps are called the left and right hearts. The right heart is composed of the right atrium and right ventricle and pumps blood to the lungs to exchange carbon dioxide in the blood with oxygen in the lungs. The left heart consists of the left atrium and left ventricle and pumps oxygenated blood returning from the lungs to the remaining organs. The heart, and the veins and arteries connected to the heart are shown in diagrammatic form in Figure 3.1. The flows of blood in the heart are summarized next. Blood that has been used by the organs contains carbon dioxide and is returned to the heart through the two great veins, the superior and the inferior vena cava. This spent blood enters the right atrium, flows through the tricuspid valve, and enters the right ventricle, which pumps the blood out the pulmonary semilunar valve through the pulmonary artery to the lungs. Oxygenated blood returns from the lungs and enters the left atrium through the pulmonary veins. It then enters the left ventricle via the mitral valve and is pumped by the ventricle out the aortic semilunar valve into the aorta where it flows to the remaining organs. The pumping of blood is associated with two phases of heart contraction; diastole and systole. Diastole is the resting phase of the heart-contraction; blood filling the heart occurs during this phase. Systole is the active (or contractile) phase of the cardiac cycle; blood pumping occurs during this phase.
3.1.2 Origin of the electrocardiogram

As described earlier, the pumping action of the heart is associated with electrical potentials that can be picked up on the surface of the body with ECG electrodes. The electrical pathways associated with this potential can best be understood with reference to the schematic heart shown in Figure 3.2. The following sequence of events underlies the ECG (and cardiac contraction):

- Excitation starts at the sino/atrial node (S/A node). The S/A node is a piece of cardiac tissue that is super-excitable and serves as the pacemaker of the heart. The series of electrical events that lead to cardiac contraction are initiated by the pacemaker activity of...
the S/A node.

- Atrial muscle contracts. Since cardiac tissue is electrically excitable, when the S/A node depolarizes, a wave of excitation spreads over the atrial muscle and it contracts, forcing blood into the ventricles.
- Atrio/ventricular node (A/V node) excitation. At the bottom of the right atrium is another piece of cardiac tissue that is super excitable, the A/V node. When the wave of excitation in the atria reaches the A/V node, it becomes depolarized and after a short delay this depolarization is spread to the bundle of His.
- Propagation down the bundle of His. This is the main conduit of excitation from the A/V node to the ventricles and delivers cardiac excitation down to the bottom of the ventricles. Thus, ventricular excitation starts not at the top of the ventricles but at the bottom.
- Purkinje system fibers. This is the link between the propagation of excitation down the bundle of His and ventricular contraction.
- Ventricular contraction. This is the last phase of the cardiac cycle.

### 3.1.3 Indication for pacing

Damage or illness may develop in the cardiac conduction system, and the rest of the heart as well, leading to various functional defects. Symptomatic bradycardia is a term used to identify clinical manifestations associated with heart rate that does not allow output to meet physiological demands. Patients exhibiting such symptoms due to slow heart rate generally are candidates for cardiac pacing. A cardiac pacing system, a pacemaker, stimulates the heart by generating a electrical pulse which leads to a cardiac contraction.

Symptomatic bradycardia may developed as the result of:

- Sinus arrest
- Sinus bradycardia
- S-A block [1]
- Bundle of His block [3]

In sinus arrest, there is no spontaneous depolarization of the sinus node. Sinus bradycardia is said to exist when the sinus node operates at a slower than normal rate. A block is said to exist when an impulse is unable to pass a given section of the conduction system or its passage is slower than usual. A block can arise at any level in the conduction system and be temporary or permanent in nature. For a more detailed study on indication for cardiac pacing see [Weston, 1991] or [Lindgren, 1992] (see also paragraph 3.2 for pacemaker functions).
3.1.4 ECG Measurements

For the registration of the electrical activity of the heart standard measurement techniques are used. Standard recording sites must be used to compare the ECG’s from a variety of patients. The Standard Limb Leads, Augmented Limb Leads, and Unipolar Electrode Configuration are the most used ECG recording standards.

**Standard Limb Leads (Einthoven’s Triangle)**

The standard limb locations are used to record the ECG from the patient's limbs; four silver-silver chloride electrodes are used, one on each wrist and on each ankle. Electrodes are placed on both wrists and the left ankle because the heart produces a net potential not only across the arms but also between the arms and the leg. Thus, these limb locations approximately form the apices of an equilateral triangle, with the heart located in the middle of this triangle. This triangle is called Einthoven’s triangle. The situation is shown in Figure 3.4. Once we have placed the electrodes in the correct positions, we must decide which pair of electrodes to put into the inputs of our differential amplifiers. This also has been standardized as the following standard limb leads:

- Lead I \( V_L^+ - V_R \)
- Lead II \( V_F^+ - V_R \)
- Lead III \( V_F^+ - V_L \)

where the superscript (+) denotes which electrode is to go into the noninverting input on the ECG amplifier.

**Augmented Limb Leads**

Even using good laboratory techniques the signal-to-noise ratio of ECG recordings can be further improved by using a different electrode configuration. The augmented limb leads use the same standard limb electrode locations.

- \( aV_R = V_R - \frac{(V_L + V_F)}{2} \)
- \( aV_L = V_L - \frac{(V_R + V_F)}{2} \)
- \( aV_F = V_F - \frac{(V_R + V_L)}{2} \)

where e.g. \( aV_R = \frac{3}{2} V_R \). Thus, the augmented limb leads can produce a reasonable increase in the signal-to-noise ratio (see [Greenhut, 1991]).
Unipolar Electrode Configuration
In many clinical diagnostic measurements, the cardiac vector as revealed by the standard limb leads is not regarded as providing sufficient clinical information. Thus, the unipolar electrode configuration is used. This configuration uses the standard limb locations plus a set of six chest leads ($V_1...V_6$) located from the mid-sternum to under the arm. These leads allow the diagnostician to obtain a three-dimensional cardiac vector for a more detailed determination of the cardiac problem of the patient.

For detailed information of ECG recordings see [Boutkan,1969] and [Phillips,1973].

3.1.5 Nomenclature ECG

Electrical events generated by the heart can be picked up almost anywhere on the surface of the body. Shown in Figure 3.5 is a diagram of the surface potentials that are associated with each phase in the cardiac cycle. The bottom portion of this figure is the sum of all these potentials and represents what we measure on the surface of our body with ECG electrodes. The names of the components of the ECG are the letters P through U.

![Figure 3.5 The components of the (ideal) electrocardiogram (ECG). The intrinsic beat.](image)

The signal amplitude is only a few millivolt. The signal, formed through atrial depolarization, is called the P wave. The corresponding signal from the ventricles is called the QRS complex. Because the muscular mass of the ventricles is far greater than the muscular mass of the atria, the amplitude of the QRS complex is much higher than the amplitude of the P wave. The atrial repolarization is not visible in a surface ECG, as this phase coincides with the ventricular depolarization phase: atrial repolarization is overwhelmed by the much larger QRS complex signal. However, the ventricular repolarization can be recorded and is referred to as the T-wave. The Q and S are the negative peaks immediately preceding and following the R wave. By definition they are always negative. The R wave, by definition, is always positive. P and T can be positive or negative, depending on electrode positions and metabolic changes.
As a result of damage or illness in the cardiac conduction system the shape of the ECG may change. A clinician is able to recognize ECG abnormalities. A specific change in the ECG recordings is directly related to a specific damage or illness in the cardiac conduction. It should be noted that the ECG signal in Figure 3.5 is an ideal ECG signal. In a real ECG recording the basic features of the ECG may not be as clearly visible as in this ideal ECG.

3.1.6 The paced ECG

The ECG recorded from a patient with an implanted pacemaker is different from the ideal ECG signal shown in the previous paragraph (Figure 3.5). The electrical stimulus given by the pacing system is usually clearly visible in the recorded ECGs. Also the location of electrical (atrial/ventricular) stimulation in the heart will influence the ECG signal. Effective ventricular stimulation is easily recognized from the ECG. Effectiveness of the atrial stimulation can often only be confirmed by a 1:1 ventricular response. In Figure 3.6 all possible (ideal) paced ECG are represented.

- **Paced atrial beat**
The atrial paced beat is almost identical to the intrinsic beat. The only difference is the atrial pacemaker stimulus preceeding the P wave.

- **Paced QRS complex**
The (ventricular) paced QRS complex is completely different from the intrinsic beat. The paced QRS complex looks like a left bundle branch block\(^5\).

- **Fusion QRS complex**
A fusion beat is a combination of the patient’s intrinsic beat and a ventricular paced beat.

- **Pseudofusion QRS complex**
A pseudofusion beat is a QRS complex caused by an intrinsic beat that is distorted by the ventricular pacemaker spike.

\(^5\) For more information on ECG interpretation see [Gelder, 1995] or [Weston, 1991].
As a result of damage or malfunction in the implanted cardiac pacing system the shape of the ECG may change or the relation between subsequent paced ECG period may change. A clinician is able to recognize the paced ECG abnormalities. A specific change in the paced ECG recordings is directly related to a specific malfunction of the implanted pacemaker.

For detecting implanted pacemaker malfunction the expert looks at the shape of the paced/intrinsic ECGs, at the time interval between different (paced/intrinsic) beats, at the differences between the time interval between different beats. Before he can do that, he should know the location of stimulation and also the specific pacemaker type and the pacemaker parameter setting, because a different pacing system configuration will change the characteristics of the paced ECG.

### 3.2 The pacemaker

The cardiac pacemaker is an electrical circuit in which a battery provides electricity that travels through a conducting wire to the myocardium, through the myocardium stimulating the heart to beat ("capturing" the heart), and back to the battery, thus completing the circuit. [Weston Moses, 1991].

The part of the pacemaker system which emits (generates) the stimulation pulses is called the 'pulse generator'. The 'electrode' is the part which transmits the stimulation pulses to the heart tissue. The pulse generator and electrode are interconnected by an insulated conductor. The electrode is affixed to one end of the conductor so as to ensure safe electrical function. An electrode plus an insulated electric conductor is called a 'lead'. A pulse generator and lead are referred to as a 'pacemaker'. Figure 3.7 shows a schematic of a ventricular implanted single chamber pacemaker.

![Figure 3.7 The components of a single chamber pacemaker. An example of a ventricular pacing system.](image-url)
The pulse generator consists of the following basic features:

- **Timing and control circuit**
  The timing and control circuit decides when a stimulation has to be generated. This depends on the pacemaker programming (mode) and on the information from the sensing circuit.

- **Battery**
  The pacemaker battery is the power source that generates the electrical stimulation for the depolarization of the myocardium. The life span of a battery\(^6\) is approximately 4 to 10 years. It should be emphasized that battery life does not equal pacemaker life, because the circuitry of the lead system may cause pacemaker failure despite a functioning battery.

- **Stimulation circuit**
  This circuit generates the programmed characteristic of the stimulation pulse (voltage height and duration)

- **Sensing circuit**
  The pacemaker is able to sense an intrinsic beat through the lead connected to the pulse generator. This function is controlled by a sensing circuit. The sensitivity of the pacemaker depends on the programmed sensitivity parameters.

- **Communication circuit**
  Through this circuit the clinician can communicate, through a radio wave, with the pulse generator. He can reprogram the setting or can gather information about the implanted pacing system regarding the status of the pacemaker. This is called pacemaker telemetry. The most important parameters in the diagnosis of the pacemaker function of malfunction obtained by telemetry are: lead impedance, marker channel\(^\text{TM}\) and/or intracardiac ECG, and battery status. The telemetry parameters will not be discussed in the report, because not all pacemakers have these telemetry features. For a detailed description and importance of pacemaker parameters see [Van Gelder, 1995].

### 3.2.1 The pacemaker lead

Each pacing lead has a cathodal electrode at the tip. Leads may come in unipolar or bipolar configurations. Unipolar electrodes incorporate the pulse generator case as the anode with the body fluid serving as the return pathway. Bipolar leads incorporate an anodal electrode along the shaft of the lead which ends 1 cm proximal to the tip. Therefore, although the lead is referred to as unipolar, it is actually a widely spaced bipolar lead. Each lead type has its own advantages and disadvantages; also the paced ECGs generated from both leads are different. For the knowledge about the leads in the third prototype expert system again see Van Gelder's thesis.

---

\(^6\) The lithium battery has become the most commonly used pacing power source.
3.2.2 Pacemaker programming modes

As mentioned before, the single chamber pacemaker can stimulate the atrium or the ventricle. During stimulation the pacemaker can be programmed in three different modes: the inhibited, the triggered, or the asynchronous mode.

- **Inhibited mode**
  During the inhibited mode the pacemaker responds with stimulus suppression in response to a sensed event (intrinsic beat). This mode is nowadays the most used single chamber pacemaker program setting.

- **Triggered mode**
  During the triggered mode the pacemaker responds with a stimulus generated in response to a sensed event (intrinsic beat). When intrinsic activity is not sensed during the sensing interval, a pace response is generated at the end of the programmed pacing interval. This mode is seldom used. It is mostly used as a diagnostic tool to detect pacemaker malfunction or to prevent undesired inhibition.

- **Asynchronous mode**
  During the asynchronous mode the pacemaker will generate a pacing spike at constant time intervals. Most pacemakers switch automatically to this mode when the pacing frequency surpasses a predefined threshold frequency.

The prototype expert system expects that the pacemaker is programmed in the inhibited mode. During the follow-up the expert system may instruct the clinician to reprogram the pacemaker into a triggered mode. As mentioned before this mode is mostly used as a diagnostic tool. Detecting the automatic switch from triggered or inhibited mode to asynchronous mode by the expert system will indicate a particular malfunctions of the pacemaker. So the asynchronous mode switching can be used as a diagnostic tool. For every programming setting and place of stimulation there is a standard code. The first letter means the stimulation location. The second letter is the sensing location and the third letter is the pacemaker mode. The pacemaker modes used by the expert system are:

- **AAI**
  The code for atrial inhibited pacing. Pacing and sensing occur in the atrium (AA); the mode of response is inhibited (I). AAI pacing is also called atrial demand pacing.

- **AAT**
  The code for atrial triggered pacing. Pacing and sensing occurs in the atrium (AA); the mode of response is triggered (T).

- **AOO**
  The code for atrial-asynchronous pacing. Pacing occurs in the atrium (A); the mode of response is fixed or asynchronous at the programmed pacing interval. There is no sensing of spontaneous atrial activity (O).

---

7 The NBG code of the Intersociety Commission on Heart Disease Resources.
A PROTOTYPE EXPERT SYSTEM

- **VVI**
The code for ventricular inhibited pacing. Pacing and sensing occur in the ventricle (VV); the mode of response is inhibited (I). VVI pacing is also called ventricle demand pacing.

- **VVT**
The code for ventricular triggered pacing. Pacing and sensing occur in the ventricle (V); the mode of response is triggered (T).

- **VOO**
The code for asynchronous pacing in the ventricle. Also called fixed rate pacing. Pacing occurs in the ventricle (V); the mode response is fixed or asynchronous (O) at the programmed pacing interval. There is no sensing of spontaneous ventricular activity (O).

### 3.2.3 Pacemaker malfunctions

There are various reason why an implanted pacing system can fail. Fortunately malfunctions rarely occur. One of every twenty-five pacemaker patients of the Cardiac Catheterisation and Pacemaker clinic in the Catharina Hospital has a benign pacemaker malfunction which can easily be fixed. A smaller percentage of the patients have a malign pacemaker malfunction.

In this paragraph a list of some possible pacemaker malfunctions will be mentioned. Van Gelder is able to recognize most of those problems by studying the recorded ECG, the pacemaker telemetry data, and X-RAYS. For solving the problems he mostly uses his intuition. In chapter 4 some techniques are explained for capturing the reasoning strategy of a domain expert.

The possible causes for pacemaker malfunction can be grouped in:

- Pulse generator failure
- Problems with the connection between the pulse generator and the lead
- Problems with the lead-connector, conductor, and electrode
- Problems with the electrode and myocardium interface
- Problems with the myocardium
The following list is from [Greenhut, 1991]. This list shows the potential problems that can occur during a pacemaker follow-up. The third prototype expert system is able to solve problems shown in this list.

- External Interference. Sensing interference (EMI) from other devices can affect pulse generator output and sensing.
- Pacemaker sensing of skeletal myopotentials.
- Perforation of a lead in the right ventricle.
- Lead dislodgment.
- Lead malposition.
- Exit block, defined as an abnormal chronic rise in pacing threshold; occurs following the development of excessive scar tissue around the electrode.
- Wire fracture; the breakage of the conductive element of the lead.
- Insulation fracture. Lead insulation breakdown may occur with or without wire fracture.
- Set screw problems (extremely rare). The set screw is the attachment between pacemaker lead and pulse generator.
- Premature battery or circuit failure.

---

8 This list is not complete. Every day more causes for pacemaker malfunction are found.
Building an expert system involves eliciting, analyzing, and interpreting the knowledge that a human expert uses when solving problems. Experience has shown that this process of knowledge acquisition is both difficult and time consuming and is often a major bottleneck in the production of an expert system [Kidd, 1987]. Some techniques have been developed to structure the knowledge acquisition process. However, an adequate theoretical basis for knowledge acquisition has not yet been established. This means that the developed techniques do not give a solution how to solve the knowledge acquisition problem. They should be used as a guideline. For every knowledge domain and project setup a different approach for knowledge acquisition should be used.

4.1 Project setup

As mentioned in the introduction I am already the third student working on this project. Probably more students will work on this project. Every student has to add his share, in the short period the student is working on the project, to the existing prototype otherwise the development of the expert system will not progress. To attain that goal, the development of the expert system should be done in phases. Currently the project is in its first phase: implementation of the domain knowledge of the single chamber pacemaker. The domain knowledge of the single chamber pacemaker consists of two system tasks. The first task is analyzing the pacemaker ECGs for abnormalities. The second task is diagnosing the ECGs in order to find the cause of the abnormalities. The second phase will be to develop a prototype expert system for single and dual chamber pacemakers. The knowledge implemented in the first phase can be reused in the second phase. Knowledge on the interaction between the atrium and ventricle (crosstalk) must be added to the knowledge base. The third phase is the implementation of the expert system in a pacemaker programmer. To achieve that, the system should be transformed into a real-time expert system. This integration will enable the expert system to collect pacemaker
specification, telemetry data, and intracardiac and normal ECG signals directly. This is only possible if model-based (deep) knowledge of the domain knowledge is added to the knowledge base. For model-based knowledge on pacemaker ECGs interpretation see [Smith, 1987], [Greenhut, 1991] and [Bernstein, 1983].

This report discusses only the first phase of this project. The second and third phases are not yet implemented. Those phases are challenges for my successors.

4.2 The knowledge acquisition process

The domain knowledge of the third prototype originates from two sources. To get a global understanding of the domain knowledge, books were used. However, the most dominant source during the knowledge acquisition process is the domain expert. Acquiring knowledge from an expert is distinguished from the more general knowledge acquisition term and is called knowledge elicitation. Eliciting knowledge is done in long sessions between the knowledge engineer and the expert. A session is a discussion that involves an exchange of ideas about the problem. Two styles of acquiring the domain knowledge were used. The first method is interviewing the expert about the domain knowledge. The second method is to present the expert with a case problem and try to uncover the knowledge by watching the expert solve the problem. The objectives of both methods are to uncover the expert's knowledge and problem-solving skills.

4.2.1 Project management

The acquisition of the domain knowledge was done with a top-down approach. Figure 4.2 shows the flow diagram used during the knowledge acquisition process. The flow diagram is a combination of the incremental design model and the prototyping design model described in [Baars, 1991].

The first phase is to acquire a global understanding of the domain knowledge by identifying the tasks used by the expert during

---

pacemaker ECGs interpretation. Also, the order and the importance of all the tasks should be examined. During the second phase a subtask is selected. The knowledge of the subtask is roughly analyzed and then implemented in the knowledge-base. Immediately after the implementation the subtask is evaluated. If necessary the knowledge base is adjusted or domain knowledge is added to the system. When a subtask has been correctly implemented, the prototype expert system is evaluated. The interaction between the subtasks should then be validated. If all the subtasks are implemented and evaluated the prototype expert system is finished.

During development of the third prototype expert system two different knowledge acquisition methods were used, the KADS and the Rapid prototyping methodology. The KADS method has been used for the first three phases: Global analysis domain knowledge, Selection subtask, and Analysis subtask. The rapid prototyping method was used during the phases: Evaluation subtask and Adjusting knowledge-base.

The KADS method structured the domain knowledge by dividing the knowledge into smaller pieces. This method reduces the complexity of domain knowledge. Rapid prototyping gives the knowledge engineer more feedback about the functionality of the system.

4.2.2 KADS

KADS (Knowledge Acquisition and Document Structuring) is a principled and systematic methodology for expert system development. KADS consists of three cycles of knowledge analysis and elicitation aiming at the description of the basic architecture of a prospective expert system.

- **Orientation.** The major goals of this stage are the acquisition of a vocabulary of the target domain to identify domain concepts and to communicate with experts, and an assessment of characteristics of the domain, the types of problems and tasks of the expert.

- **Problem identification.** The subtasks here are the uncovering of the structures of the domain concepts, a functional analysis of the prospective expert system, and a task analysis. The structure of domain concepts is the static description of the knowledge in the domain.

- **Problem analysis.** This is the most complex stage in knowledge acquisition. An analysis of the user and the operational environment of the prospective system is followed by a very detailed analysis of the domain expert in action.

An advantage of the KADS method is that it is able to partition the domain knowledge into smaller pieces. It uses the ‘divide and conquer’ principle. Complex problems are easier to solve if partitioned into decomposable parts and strategically spaced in time.

A disadvantage of the KADS method is that the problem should first be analyzed completely before solution methods are selected and applied. This will take a lot of time. The knowledge engineer, who initially plays the role of a student in a new domain, may find it hard to cope
with the large amount of new data. It is possible that the interpretation of the expert’s knowledge by the knowledge engineer is different from the domain expert’s interpretation. A wrong interpretation will surface at a later stage of the development of the expert system. Only the first two KADS cycles were used during the development of the third prototype expert system. Those cycles structure the knowledge and give a global interpretation model. The ‘problem analysis’ cycle has been replaced by the Rapid Prototyping method. This method is easier and quicker to implement than the proposed third KADS cycle.

4.2.3 Rapid Prototyping

The Rapid prototyping approach is a trial and error method and thus very unstructured compared to the KADS methodology. Figure 4.3 displays a flow diagram of the Rapid Prototyping method. This knowledge acquisition method is very simple. It consists of three stages. The knowledge elicited by the knowledge engineer is immediately implemented in the (prototype) expert system. The system is then evaluated. After the evaluation the elicitation process starts again, completing the circle. Rapid prototyping will capture only the heuristic knowledge of the knowledge domain. Heuristic knowledge can easily be implemented as 'IF.. THEN -rules'. Those rules can easily be understood by the domain expert. Thus the validating and adjusting the faulty rules is relatively simple.

The advantage of this method is that testing the system on faulty or missing knowledge is easy. By demonstrating the prototype expert system, the domain expert gets more involved in the project. This will motivate the expert more on ‘giving’ his knowledge away and ultimately the acceptance of the finished expert system will be easier. Another advantage, from the view of the knowledge engineer, is that is not necessary to have a throughout understanding of the domain knowledge.

The disadvantage of this method is that it will result in an unstructured expert system. The larger the knowledge base, the more difficult it will be to maintain and update the expert system.

The KADS method is used for identifying higher level concepts of the domain knowledge. The Rapid prototyping method is used to get the expert closer to the ‘pacemaker’ project and to capture the heuristic knowledge of the expert. By using a combination of the KADS and Rapid prototyping methodology the knowledge base of the third prototype expert system is easily maintainable and the elicited knowledge can immediately be implemented without analyzing it throughly. This will speedup the development of the ‘pacemaker expert system’.
4.2.4 Heuristic matching

Expert behavior that is seemingly domain-specific may originate from higher level problem-solving methods, which is well structured and has some degree of domain independence. The heuristic classification method devised by Clancey [Clancey, 1985] is used for the problem-solving strategy. Clancey calls heuristic classification:

- A non-hierarchal association between data and category requiring intermediate inferences, possibly involving concepts in another taxonomy.

The problem and solutions in a diagnostic task are linked with each other through a heuristic match concerning relations between the problem and solutions. Heuristic classification is best understood schematically, as in Figure 4.4, which show the basic steps in heuristic classification.

![Figure 4.4 Inference structure of heuristic classification.](image)

- Data abstraction is used to abstract the pacemaker data and pacemaker ECGs data for identifying the global problem e.g., oversensing, no output, noncapture etc.
- Heuristic match. Although the match between the raw data of a particular case and the final diagnosis is hard to perform, it is often easier to perform a match between a global problem and broad classes of malfunctions. This matching process is heuristic because the map from data to hypothesis may not be one to one at any level.
- Solution abstraction is used to represent a class of solutions.
- Solution refinement. Having identified a global problem and narrowed the solution space with heuristic matching, we still need to identity the candidate in that space. Thus, we need to find the cause of pacemaker malfunction induced by a specific global problem. This may require the gathering of further data.
**Example:**

During a pacemaker ECG interpretation, the domain expert first tries to identify the problem type by gathering information about the particular case (data abstraction). Let us say the expert finds that the pacemaker stimulus shows noncapture. The next task is to find the cause of noncapture. The expert will then only examine the hypotheses that cause noncapture. This class of hypotheses is found through a heuristic match.

Heuristic classification simulates the (high level) problem solving strategy of the domain expert. This method can be used in many different domains but it is closely linked with the knowledge acquisition methods (KADS and Rapid prototyping). The tasks, classes of hypotheses and relations between hypotheses acquired through knowledge acquisition should be tuned with the heuristic classification method.

By using this method the expert's reasoning strategy is modeled. This will give a better structured knowledge base which leads to a better maintainable expert system.
The analyzed domain knowledge

Expert systems are designed to support activities that are not well defined. Often these activities are very fluid by their nature. The needs and requirements of these activities are constantly changing. This is also true for the pacemaker ECG interpretation knowledge. First the domain is very unstructured, if compared with exact sciences, and secondly new methods and problems on pacemaker ECG interpretation are discovered every day. Also, the pacemaker technology advances year after year. Thus it is practically impossible to develop an expert system for diagnosis of pacemaker malfunction that can diagnose all types of pacemakers. However, expert systems are practically suitable for this constantly evolving process. In contrast with traditional computer programs new domain knowledge can easily be appended to expert systems.

One of the goals of this project was that all the acquired knowledge of the first and second prototype expert system should be reused by the third prototype. The third prototype contains the knowledge of both systems and also new knowledge is added to the system. Another goal was that the prototype system should contain the decision-making strategy of the domain expert.

The second prototype fails to reproduce the human reasoning of the domain expert. The first prototype however is able to stimulate the decision-making strategy of the domain expert very well. Nevertheless, according to Bourgonje I quote [Bourgonje, 1994] (translated from Dutch):

- ‘The difference between the decision strategy of the expert is that during a pacemaker follow-up the expert, after identifying the cause of pacemaker malfunction and solving the problem, is able to validate the result of his action and he is also able to check all the other pacemaker functions. The first prototype is not flexible enough to do that. After giving the solution the first prototype stops with the follow-up’.

To solve that problem a sort of high level conceptual ‘push-pop’ function has been applied in the third prototype’s software.

---

10 This does not mean that the second prototype does not work. The system can recognize the paced ECG very well, but the system fails to simulate the human reasoning. It is considered, by the domain expert, as not user friendly system [Wouters,1997].
5.1 Decomposition of the knowledge domain

A pacemaker follow-up consists of two tasks. The first task, the analysis, is for detecting if there is a pacemaker function failure. If a malfunction is identified the clinician starts with the second task: the diagnosis. During the pacemaker diagnosis process the clinician tries to find the cause of malfunction and, if possible, a solution for the pacemaker problem. The analysis of the pacemaker function consists of an output, capture, and sensing function. The pacing system can be diagnosed for (intermittent) no output, (intermittent) noncapture, oversensing, or undersensing. Figure 5.1 displays a visual representation of the tasks and functions used during a follow-up.

Figure 5.1 Pacemaker functions and potential pacemaker function failures (illustration from [Bourgonje, 1994]).

Analysis of pacemaker functionality

- Output function:
  This function checks for a correct pulse generator output.

- Capture function:
  The electrical discharge of the pacemaker causes stimulation of the myocardium.

- Sensing function:
  The ability of the pacemaker to sense and respond to an intrinsic beat of the heart.

The pacemaker diagnosis task will identify the causes of the following problems, detected during the pacemaker ECG analysis.
Diagnosis of a pacemaker malfunction

- **(intermittent) No output:** The pacemaker does not generate an electrical stimulus.
- **(intermittent) Noncapture:** No depolarization in response to an electrical stimulus emitted by the pacemaker.
- **Oversensing:** The sensing of undesired electrical signals by the pacemaker amplifier.
- **Undersensing:** The pacemaker does not adequately sense the electrical activity of the heart.

For a detailed decomposition of the analysis and diagnosis tasks see [Bourgonje, 1994]. He describes the tasks with hierarchical and causal graphs. Those graphs give a well-defined structure of the knowledge representation in the knowledge base used in the third prototype. It should be noted that the graphs used in Bourgonje’s report are not a one to one representation of the knowledge base.

### 5.2 Decomposition of the problem-solving strategy of the domain expert

As mentioned before the problem-solving strategy of the domain expert is that he starts with the pacemaker function analysis followed by the pacemaker function diagnosis. Figure 5.2, 5.3, and 5.4 shows diagrams of respectively the first, second and third prototype expert system.

![Figure 5.2](image1)

**Figure 5.2** Problem solving strategy of the first prototype expert system.

![Figure 5.3](image2)

**Figure 5.3** Problem solving strategy of the second prototype expert system.

![Figure 5.4](image3)

**Figure 5.4** Problem solving strategy of the third prototype expert system.
The first prototype, as mentioned before, can simulate the expert’s reasoning strategy very well, but the system could not validate its solution on a pacemaker function failure. The third prototype solves that problem with a return loop from the diagnosis to the analysis function.

The first and second prototypes are not completed for single chamber pacemakers. The pacemaker analysis function and diagnosis function of the first prototype is not completed. The second prototype has only the pacemaker analysis function and the system always asks the user when a pacemaker follow-up ends. This feature will not enhance the user friendliness of the system and thus the third prototype makes no use of this attribute.

Although the problem solving strategy of the prototypes looks alike, the knowledge structure of the different prototypes is all different. The first and third prototype use heuristic knowledge. The difference between the first and third prototypes is that the third prototype makes use of a sort of push and pop function, which makes the third prototype more flexible and better maintainable than the first prototype. The ‘push-pop’ function will be explained in paragraph 5.4. The second prototype uses model-based knowledge. This is a totally different approach from the other prototypes. Figure 5.5 shows a detailed strategy of the third prototype expert system and the knowledge implemented in the system. See appendix E for the strategy and implemented knowledge of the first and second prototype.

![Figure 5.5 The problem solving strategy; bidirectional arrows are new.](image-url)

For detailed information on the knowledge structure of the second prototype expert system see [Wouters, 1997].
The third prototype contains the tasks described in paragraph 5.1. The pacemaker evaluation starts with the output check. The output check can identify a noncapture, (intermittent) no output, and oversensing failure. If there is a correct pacemaker output, the system continues with the capture check. When a correct capture is established, the pacemaker sensing function is checked. If no malfunction is found or if there is a malfunction, which have been solved by the system, the evaluation ends.

For example, when pacemaker oversensing is detected during the sensing check, the system starts with diagnosing pacemaker oversensing. If the cause of oversensing is found and solved, the system will ask the user whether he wants to check if the solution, given by the expert system, is correct. If the user wants to re-evaluate the sensing function, the system returns to the sensing check and evaluates the sensing function of the pacemaker again. If not, then the evaluation ends.

**Output check**
During the output check the system checks for a correct pulse generator stimulus. The stimulus can be recognized by a peak in the ECG. The place of the stimulus relative to the basic ECG features, the intermittent absence of stimuli, and the time interval between two stimuli gives an indication of the function or malfunction of the pacemaker output function. The expert systems output check can detect oversensing, continuous no output, intermittent no output, and continuous noncapture problems.

**Capture check**
During the capture check the system checks how the heart responds to pacemaker stimuli. If the heart contracts as result of the stimulus, correct capture is established. A correct capture can be recognized as a paced atrial beat, a paced QRS complex, and as a fusion QRS complex (see Figure 3.6 paragraph 3.1.6). If there is a different pacemaker characteristic visible in the recorded ECG then this may indicate continuous or intermittent noncapture or undersensing.

**Sensing check**
During the sensing check the system evaluates the sensitivity setting of the pacemaker. If the pacemaker sensitivity is too high, the pacemaker will detect unwanted electrical heart activities. This can be recognized as a prolongation of the time interval between two beats. The opposite problem can occur, a reduction of the time interval between two beats, when the sensitivity is too low. Another problem that can be identified during the sensing check is intermittent no output.

**Diagnosis continuous no output**
Continuous no output can be caused by a problem with the lead (galvanic discontinuity), air socket around the pacemaker casing, a short circuit in the leads, a pacemaker circuit failure, or by a battery failure.
**Diagnosis intermittent no output**
The problems with intermittent no output are the same as the one describe in diagnosis no output. Except for battery failure: this problem can only cause a 'no output problem'. The difference is that now the causes are intermittent in nature.

**Diagnosis continuous noncapture**
Continuous noncapture can be caused by a problem with the lead integrity, galvanic discontinuity in the lead system, short circuit in the lead system, an incorrect position of the stimulation electrode, problem with the stimulation threshold, or by a pulse generator failure.

**Diagnosis intermittent noncapture**
Intermittent discontinuity in the lead system and intermittent short circuit of the lead system can cause intermittent noncapture. Other problems that can lead to intermittent noncapture are a Wenckebach\(^{12}\) periodicity between electrode and myocardium, an unstable electrode, or an output stimulus close to the stimulation threshold.

**Diagnosis oversensing**
Oversensing is sensing of signals you do not want to sense. Such signals could originate from the heart itself (T,R,P-wave sensing), myopotentials, lead short circuits, lead discontinuities, or by exogenous\(^{13}\) signals.

**Diagnosis undersensing**
Undersensing can be caused by an inadequate intracardiac electrogram\(^{14}\), by a short circuit in the lead, or by a pulse generator failure.

During a pacemaker consultation only the tasks which have to be checked for that specific pacemaker problem will be evaluated. All the other tasks will be skipped. The system will never ask unnecessary questions. For example if the patient has a VVI pacemaker the prototype expert system will never ask questions about atrial pacing systems. The system can even find alternative questions. For example if the system does not have a marker channel\(^{15}\) the expert system will use an alternative question for detecting P-waves.

---

\(^{12}\) Wenckebach phenomenon is a progressive prolongation of the PR- interval in cycles preceding the dropped beat.

\(^{13}\) Exogenous signals are signals that originate from outside the human body e.g. EM interference.

\(^{14}\) If a signal cannot be sensed by a pulse generator at its maximum sensitivity, without oversensing of other cardiac related signals, it should be considered to be an inadequate signal. Possible causes are electrode displacement, fibrosis formation, myocardial infarct, or changes in intraventricular conduction.

\(^{15}\) With a marker channel\(^{\text{TM}}\) P-wave sensing can easily be detected. For information on marker channels see [Gelder,1995].
5.3 The reasoning strategy of the prototype expert system

For capturing the expert’s reasoning strategy the prototype expert system uses a backward­ and a forward-chaining mechanism described in paragraph 2.3 and the heuristic matching method discussed in the previous chapter. The strategy used during the analysis of the pacemaker function is a little different from the strategy applied for diagnosing implanted pacemaker failures.

The strategy of the analysis will be explained first because the strategy of the diagnosis is an expansion of the analysis strategy. Figure 5.6 shows a graph which represents the reasoning strategy used during the pacemaker analysis.

Consider the above-mentioned graph as an AND/OR tree\(^ {16} \). For simplification reasons we will use the graph illustrated in Figure 5.6, the ‘analysis tree’. The analysis tree represents one analysis task, e.g., the output check. The tree has 3 levels, the meta-level, the hypothesis-level, and the symptom-level. The symptom-level contains the questions of the expert system. The questions are also called the primitives. Primitives are rules that cannot be proven by other rules. The primitives are connected with the hypotheses. The hypotheses are the pacemaker function failures, e.g., oversensing, no output, noncapture etc.

Because not every hypothesis is relevant for a particular pacing system configuration, meta­knowledge has been used for the reduction of the evaluation of the number of hypotheses. Meta-knowledge is knowledge about knowledge. Meta-knowledge provides direction to the expert system during the session. It is used to enhance the system’s efficiency by directing its

\(^{16}\) The expert system building tool Simplexy uses AND/OR trees for connecting the production rules.
reasoning ‘attention’ to the most promising group of hypotheses. Meta-knowledge is also the knowledge in which order the questions should be asked and/or which questions can be skipped.

Backward-chaining is used for selecting the correct hypothesis. As mentioned in paragraph 2.3 backward-chaining remains focused on a given goal. This produces a series of questions on related hypotheses, a situation that is comfortable for the user. It is how a human solves a problem by forming a hypothesis and then seeing if it can be proven.

When a hypothesis is selected, a heuristic match is used to determine the next (diagnostic) task. Heuristic matching is done with a forward-chaining method, the push-pop function. Figure 5.7 gives an illustration how a heuristic match is used during the pacemaker analysis.

The questions of the analysis tree are abstracted in the hypotheses. If a hypothesis is true then evaluate the corresponding heuristic match. For example when an oversensing problem has been detected during the pacemaker analysis then evaluate the task diagnosis oversensing. Solution refinement is finding the cause of a particular pacemaker malfunction. In our case by asking questions with the backward-chaining method.
The strategy used during the diagnosis is an expansion of the analysis strategy. It is a combination of the methods illustrated in the two previous figures (Figure 5.6 and 5.7). But instead of heuristic matching, a goal orientated reasoning is used. A diagnosis task consists out of more than one AND/OR trees. These trees are linked with each other by a goal. The forward-chaining methods used for diagnosing pacemaker failures is the Simplexys goal. Which means ‘IF this_rule_is_true THEN do ...’. The THEN denotes the Simplexys goal. For example the following rule occurs in diagnosis task oversensing (with a different syntax):

- IF the intracardiac electrogram shows spurious signals THEN check the lead system for intermittent contacts.

The first part of this rule is a hypothesis and the second part is the root of another tree.

The backward-chaining, forward-chaining, heuristic matching, meta-knowledge, decomposing the domain knowledge, and decomposing the problem solving strategy gives the third prototype the ability to mimic an instructional assistant in a consistent and reliable manner.

5.4 The Push-Pop function

As mentioned in paragraph 5.1 the knowledge of the prototype expert system is divided into nine tasks. As described in the previous paragraph every task is a large AND/OR tree. The push-pop function applied for heuristic matching works with the tasks as if they were single elements. If the expert system evaluates an analysis task and switches to a diagnostic task, the identification name of the analysis task is then stored in a record. When the system has executed the diagnostic task, it switches automatically back to the last evaluated analysis task by looking at the identification name stored in the record. The record is then emptied by the system, thus creating a so-called push-pop function.

The reason for the implementation of the above explained method is that two different analysis tasks can point, with heuristic matching, to the same diagnostic task. Another problem is that the knowledge base will expand during the development of newer prototypes. Knowledge is added to the different tasks, new tasks and new heuristic matches will be added to the knowledge base. The push-pop function is an elegant way to solve the task switching problem. See appendix D for detailed information on how the push-pop function is implemented in the rule base of the prototype expert system.
6 Development of the third prototype expert system

6.1 Development tools

For the development of the prototype expert system Delphi, a visual design environment for Windows application, and the Simplexys expert system toolbox were used. Delphi was used for the development of the user interface and Simplexys for simulating the reasoning strategy of the domain expert.

6.1.1 Simplexys

Simplexys is an expert system toolbox, designed for developing real time expert systems. It is developed in the third generation programming language Pascal. By using Pascal it is possible to adapt Simplexys to the special needs of an expert system\(^{17}\).

In a Simplexys expert system goals and protocols can be implemented. Goals determine what the next action should be and the protocols describe when an action should take place. The Simplexys programming language is rule based: the knowledge in the knowledge base is encoded into production rules. Simplexys rules are not realized as implications, but as assignments. For example, the production rule:

- IF A AND B THEN C is most approximately translated into the Simplexys format C:= A AND B.

Simplexys has a rule compiler that translates the rules into a semantic network. The network represents the values and relations between the rules. By using a network, which represents the knowledge base, the expert system can efficiently search through the rules. The Simplexys inference engine employs both forward reasoning and backward reasoning.

The Simplexys toolbox contains also a semantic checker and a protocol checker. The semantic checker checks the knowledge base for several semantic errors. The protocol checker is used for the detection of syntax, topology, and dynamic errors.

For information on the Simplexys toolbox see [Blom, 1990].

---

\(^{17}\) See appendix A, B, and C for the adjustments in Simplexys.
6.1.2 Delphi

For the design of the Windows interface Borland Delphi for Windows 3.1 was used. Delphi is a visual design environment for the creating of sophisticated Windows applications. Delphi includes an object-oriented Pascal compiler and debugger, a visual design environment and database tools. Borland designed Delphi for developing prototypes and converting them into commercial products.

Because Delphi is an object oriented and an event triggered programming-environment it differs from the traditional 'linear' programming style. A Delphi program consists of objects. Every object has a clearly specified purpose and the objects communicate with one other.

The Simplexys tools are programmed in the so-called traditional programming style and developed for the MSDOS operating system. Thus some adjustments were necessary before the Simplexys tools could be used in combination with Delphi. For information on the adjustments see appendix B.

6.2 The prototype expert system

There are two versions of the third prototype, a DOS version and a Windows 3.1 version. In this paragraph only the interface of the Windows version will be explained. The Windows version is user-friendlier than the DOS version. It has more explanation facilities and the interaction with the users is easier. The only thing that the user has to do is to specify the pacemaker type and parameters and activated the consultation program. The program will ask a question. The user has to answer that question with a 'yes' or a 'no'. If there is a problem with the interface, an online user manual is available. For detailed information on the prototype expert system see appendix B and C.

The knowledge base of the DOS and Windows version of the prototype expert system are almost identical. There is a problem with the DOS version: it is full. If more rules are added to the knowledge base, a Pascal compiler error appears. The code segment of the compiled DOS program will be larger than 64 Kb. Programs in a 16-bits software environment cannot have segments larger than 64 Kb. Although Windows 3.1 is also a 16-bits operating system, the segments of Windows version of the prototype expert system are all smaller than 64 Kb. This is caused by the object orientated programming of the Windows interface. By dividing the programming code in more segments none of the segments exceeds the 64 Kb limit.
6.2.1 The user interface

The interaction between an expert system and the user is conducted in a natural language style. The interaction is also highly interactive and closely follows the conversation between humans. To conduct this process in a manner that is acceptable to the user places special demands on designing the user interface. Particular attention is paid to the question’s design to obtain reliable information from the user. A basic design requirement of the interface is to ask questions. The interface consists of 3 windows: the dialogue window, the data window, and the help window. There are 3 different interface sizes: 640 by 480, 800 by 600, and 1042 by 768 pixels. The system will automatically select the correct user interface. Although the interfaces are a little different, they all have the same functionality.

The dialogue window

![Figure 6.1 The Windows interface of the third prototype expert system.](image)

The dialogue window has four parts. The question window, the question box, the result window, and the information windows. The window at the center of the user interface is the question window. This window displays the questions asked by the program. The user has to respond through the question box with a yes or no. The system evaluates the answer and replies to the user by copying the last question and the system’s conclusion in the result window. That is the uppermost window of the dialogue window. The two windows at the bottom display the current pacemaker settings and are called the information windows.
The data window

![Data Window](image)

Figure 6.2 The data acquisition window.

Before a user starts with the consultation the system needs to know some pacemaker specifications: the pacemaker mode, the lead type, and some special pacemaker programming feature. The pacemaker mode can be selected by the `pacemaker type` radio box at the upper left corner of the data window. The `polarity` radio box at the center of the window displays the kind of leads the expert system can recognize. At the bottom of the window the user can select the special pacemaker programming features.

The help functions

The expert system has different types of help functions. Those functions can be activated with help buttons or with the help pull-down menu. There is an online user’s manual available which explains the basics of the user interface. There is also an online definition list. This list contains more than one hundred and twenty explanations on terms used in the field of pacemaker technology and pacemaker diagnostics.

The expert system also has a `why` facility. The system can explain with this function why it is asking a given question. For example when an individual consults with a human expert, the conversation is highly interactive. On occasion the individual may ask the expert why he is pursuing a certain line of reasoning. The explanation given can make the user feel more comfortable with the line of questioning. The `why` function of the expert system shows which hypothesis is currently evaluated. The `why` facility has to be activated before the consultation starts. An extra window will then appear on the user interface, the progress window.
During the consultation there is an interactive question and answer relation between the expert system and the user. It can occur that the user does not understand, partly or completely, the questions asked by the system. For that purpose a help function is designed. If a question is not clear to the user, he can then activate the help function with the help button on the question box. This function will activate the Winhelp program. In Figure 6.3 an example is given how the help on a question may look like. At the top of the window the question is repeated followed by an explanation. The definition list is also accessible with this help function. By activating the search\textsuperscript{18} button on the Winhelp window the definition list will appear.

When the consultation ends, the reasoning steps used during the consultation by the expert system can be printed on paper. This is very useful for students who want to remember the protocol that is used to identify a particular problem.

The expert system has also a debug mode. If this mode is activated the rule number and symbolic\textsuperscript{19} rule name of the rules fired during a consultation are displayed on the result window. This mode is only useful for the knowledge engineer. Modifying the knowledge base on faults is then easier.

\textsuperscript{18} For the Dutch version of Windows the ‘zoeken’ button.

\textsuperscript{19} ‘Symbolic rule name’ is the identification name of a rule. A rule can also be identified by a number, the so called rule number.
Example:
The following example shows the conversation between the user and the prototype expert system. The interaction between the user and the prototype is visible on the result window of the dialogue window. After answering a question of the expert system by the user, the prototype can make some remarks or some orders that should be executed by the user. This goes on until the cause of pacemaker malfunction is found. The pacemaker configuration in this example is a VVI-pacemaker with a unipolar lead with no hysteresis and no real-time telemetry. This example shows a pacemaker oversensing problem caused by an inaccurate pacemaker sensitivity setting. The pacemaker senses the preceding T-wave. A reduction of the sensitivity setting solves the oversensing problem.

---

QUESTION: Three channel ECG, leads I/II/III, shows: Pacing spikes? : [Yes]
Pacing spikes present. Proceeding with capture check ......

QUESTION: Three channel ECG, leads I/II/III, shows: Ventricular pacing spike continuously followed by a paced or paced fusion QRS complex? : [Yes]
Ventricular capture has been established. Ventricular pacing parameters should be determined:
- Battery voltage (for check on EOL)
- Stimulation threshold (for check on encapsulation)
- Lead impedance (for check on lead integrity)
Proceeding with sensing check ......

QUESTION: Is there a paced-intrinsic interval visible on the ECG? : [No]

QUESTION: Is there a intrinsic-paced interval visible on the ECG? : [No]

QUESTION: Is there a paced-paced interval visible on the ECG? : [Yes]

QUESTION: The time interval between the two paced beats is longer the escape interval? : [Yes]

QUESTION: The prolonged interval is an exact multiple of the escape interval in the ECG? : [No]
Oversensing has been established Proceeding with diagnosis for probable causes...

QUESTION: Lead II/III, shows: All prolongations of the escape interval are more or less the same size? : [Yes]

QUESTION: Lead II/III, shows: The point of oversensing corresponds with the preceeding T-wave? : [Yes]

QUESTION: Lead II/III, shows: During Longer observation (>3 min) a stable escape prolongation? : [Yes]
T-wave oversensing detected Reduce sensitivity of the pacemaker

QUESTION: after reduction of the sensitivity lead II/III shows: normal escape interval? : [Yes]
T-wave oversensing problem solved

[** End Of Simplexsys Pacemaker Function Evaluation **]
6.3 System design of the prototype expert system

Figure 6.4 The system design of the prototype expert system

Figure 6.4 shows the system design of the expert system. The illustration is almost the same as Figure 2.1 in paragraph 2.2. The only difference is that the explanation box in Figure 2.1 is less detailed than the one shown here. The inference engine, knowledge base, and the database are already explained in paragraph 2.2.

The user interacts with the system through the user interface. The system wants to know the pacemaker specification data, the answers on the questions asked by the system, and when to activate a help function. When the evaluation starts, the inference engine gets information on the rules from the knowledge base. The inference engine also selects the text strings for the rules and displays those strings on the user interface. After every answer of the user the inference engine evaluates all the current information and stores the result in the knowledge base.

If help on the questions is required then Winhelp will be activated. Winhelp uses the symbolic rule name to select the correct help file. The inference engine is the only part of the system that knows which rule is currently in use. Because the inference engine uses rule numbers to identify the rules and Winhelp the symbolic rule names a transformation is necessary.
If the debug mode is activated the symbolic rule names and rule numbers are displayed on the screen. This function is also linked with the inference engine as the help-functions.

6.4 The current status of the project

The pacemaker expert system is not finished. It will never be finished. It is a classification system and thus can only diagnose the problems implemented in the expert system. Every day new types of problems are found and pacemaker technology gets more complex every year. The third prototype contains the domain knowledge of the single chamber pacemaker. It contains most known causes for pacemaker function failures and because those failures seldom occur, an expert system is always useful.

New knowledge on single chambers should always be appended to the knowledge base. The system is specially designed for single chamber inhibited pacing systems. The expert system detects pacemakers in asynchronous mode. Subsequently the system requests that asynchronous pacemakers, should be programmed in the inhibited mode otherwise no evaluation can be made.

The system is not ready for triggered pacing systems. According to the expert, triggered systems are seldom used. This does not mean that there is no knowledge on triggered systems in the knowledge base. The expert uses the triggered mode is as a diagnostic tool. This type of knowledge is implemented in the system.

The help-file for the help on the questions, asked by the expert system, is not finished. The help-file explains only a small number of questions. Completing the help-file is a job for the expert. A knowledge engineer is not qualified to write the help-file. Only the domain expert is qualified to write the explanations for the questions.

Although the Simplexys rule compiler can generate a RTF help-file, which is necessary for the Windows help program Winhelp, it is just a frame. The help-file generated by the rule compiler contains only the questions without the explanations. For more information on the help files see appendix A.

The third prototype contains 722 rules and 208 of them are ask rules. The Simplexys rule base is 243 Kb ASCII text large. Although this is equivalent to 95 A4-pages text, the system contains only part of the domain knowledge described in Van Gelder’s thesis. An expert system that contains the complete knowledge of Van Gelder’s thesis doubles or triples the size of the current knowledge base.
Test and evaluation of the prototype expert system

Every expert system has to be tested and validated. The prototype expert system should be validated on the implemented domain knowledge and tested on the formulation of the questions and on the user-friendliness of the user-interface.

The third proto expert system was not elaborately tested. The only objective test on the expert system can be done by testing the system on real pacemaker patients and not by documented cases. During an evaluation of the paced ECG’s the expert system will instruct the clinician to do some operations on the patient. The system will then ask the clinician if the paced ECGs changes and what the changes are. This is not possible if the system is evaluated with documented cases. According to Van Gelder only one of every 25 pacemaker patients has a pacemaker function problem. Of those 25 patients a large part possesses a dual chamber pacemaker. Thus, testing the prototype expert system on single chamber pacemakers is a time-consuming process.

The prototype expert system has been elaborately tested, by the domain expert, on identifying correct pacemaker ECGs. There are only three different types of single chamber paced ECGs that suggest a correct pacemaker function:

- An ECG-recording with only paced beats.
- An ECG-recording with only intrinsic beats.
- An ECG-recording with paced and intrinsic beats.

The third prototype expert system can identify the above-mentioned ECGs as a correct pacemaker function.

As mentioned paragraph 5.1 the evaluation of paced ECGs is done in two phases. The expert system starts with the analysis of the ECGs and proceeds with the diagnosis of the ECGs. Analyzing the ECGs for detecting pacemaker malfunction is relatively easy. During the analysis of ECGs the expert system needs to know the basic features of the recorded ECGs. Thus, testing the analysis-section of the prototype expert system can be done with cases.
Although the domain knowledge of the analysis is differently implemented in the knowledge base of the third prototype, the essence of the reasoning strategy is the same compared with the previous prototypes. The analysis-section of the expert system has been tested by Wouters. He tested 25 cases from which 24 were correctly identified\textsuperscript{20}.

Only five cases were used for testing the domain knowledge on the analysis and diagnosis of paced ECGs. Those five cases were:

- T-wave oversensing.
- Wenckebach periodicity between ventricular electrode and myocardium.
- Interruption in the lead connector.
- Battery failure.
- R-wave oversensing.

All the five cases were correctly diagnosed by the prototype expert system. The last three cases were also used for testing the user-interface. Three persons have tested the user-interface of the third prototype: a nurse, an assistant cardiologist, and a cardiologist. They confirm that the layout of the Windows user-interface is comprehensible. The help-functions, if they are complete, are useful and their attention on the dialogue window was immediately focused on the question window. However they were missing the possibility to rectify an answer during the pacemaker evaluation and to exit the program during an evaluation.

The cardiologist observed that during the sensing check, the ECG was split into three types of intervals. He had some problem with this approach. Normally he uses his intuition to select the right type of interval. The prototype expert system asks explicit question about those intervals. The cardiologist is not used to do that.

The tests on the prototype also show that the expert system was too difficult for the nurse and too straightforward for the cardiologist. The cardiologist also remarked that if a user depends solely on the expert system he would not think any more how to solve a problem and that is dangerous. The assistant cardiologist however was very enthusiastic over the system. He would definitely use the expert system for educational purposes.

The tests on the prototype indicate that the reasoning strategy used by the domain expert can be simulated by the expert system. From those tests it cannot be concluded how well the prototype expert system can identify the cause of a pacemaker malfunction. The only indication of the quality of the prototype expert system comes from Van Gelder. He feels that this prototype is a good basis for the final version of the expert system.

\textsuperscript{20} For more information on the tests carry out by Wouters see [Wouters, 1997]
Conclusions and Recommendations

The building of an expert system for the interpretation of paced ECGs is a project that will take many years of development. Two prototypes have already been designed. This report discusses the continuation of the ‘pacemaker expert system project’: The development of the third prototype expert system. The third prototype contains the domain knowledge of both the previous prototypes and more.

Conclusions

Acquiring the domain knowledge with a combination of the KADS and rapid prototyping methodology was very successful. With the KADS method the global structure of the domain knowledge was obtained. This simplifies the knowledge acquisition process. The details of the domain knowledge have been acquired with the rapid prototyping method. This method gives the knowledge engineer more feedback and it also gets the domain expert closer to the development of the system and that is important for the continuation of the project.

The implementation of the domain knowledge of all the tasks necessary for the interpretation of paced ECGs for single chamber pacemaker functions and the design of a Windows interface made it possible to demonstrate and test the system on potential users. The positive response from the potential users suggests that there is a demand for such an expert system.

A small test performed by three medical personnel at the Catharina hospital shows that the expert system is not suited for novice users. The people who use the expert system need some moderate knowledge on ECG interpretation.

The Windows interface and the help functions were considered as an improvement compared with the users-interfaces of the first and the second prototypes.

The three test persons were content about the reasoning strategy used by the expert system. The questions and remarks from the expert system were according the test persons most of the time very clear. Only the questions used during the sensing check were sometimes confusing. This problem can be solved by reformulating the questions.
Recommendations

The prototype expert system is not finished and therefore the system cannot be elaborately tested. The tests preformed on the prototype indicate a very good performance. Nevertheless, as stated before this is just an indication. A valid performance rate can only be found if the expert system is tested on many pacemaker patients. It is recommended to add the knowledge of dual chamber pacemakers to the expert system before testing the system. Testing the system is than easier. As mentioned in chapter 7 testing and evaluating the expert system is a time-consuming process. If all pacemaker types are implemented in the expert system, every pacemaker patient can be used to test the system. There are only single or dual chamber pacemakers; other pacemaker types do not exist.

Do not forget to evaluate the reasoning strategy of the expert system and how the questions are formulated. Comprehensible questions and a clear-cut reasoning strategy are very important for the acceptance of the expert system by the potential users.

The Windows-interface of the prototype expert system has two drawbacks. In the first place it misses the possibility to rectify an answer. Secondly the expert system cannot be interrupted during an evaluation. Both problems can be solved by modifying the Simplexsys inference engine. A possible solution for the first problem is to keep track of all the answers during an evaluation. If the user wants to rectify an answer, the system will than automatically rerun the pacemaker evaluation until the second last answer. This process goes very fast so that the user will not notice it. With this method all the answers can be corrected by the user. The second problem can be solved by adapting the Simplexsys inference engine for an object oriented and event triggered environment. This will allow to stop or activate the inference engine on demand.

The knowledge base of the third prototype is almost full. The 64-Kb border is virtually reached. Only a few new rules in the knowledge base are needed to surpass this 64-kb border. This problem only occurs in a 16 bits operating system environment thus, by modifying the inference engine for Win95, which is a 32 bits operating system environment, will solve this problem.

The automatically generated help-file for the ‘questions’ has to be improved. The only rules which need help are the ask rules (the primitives). The ask rules are the questions of the expert system. A possibility is to add a new command to the Simplexsys language. This command will contain the help text for the questions. In appendix A is an example given how this new command should be used.
Glossary

Bradycardia
When the cardiac output does not meet physiologic demands, caused by a very low heart frequency, then that means a ‘Bradyarrhythmia’. Bradyarrhythmia means that cardiac function/work proceeds more slowly than usual (less than 60 beats/min). Bradycardia is a term which refers to the heart as a whole.

Diagnostician
A person who performs a pacemaker follow-up. That can be a cardiologist, an assistant cardiologist, a pacemaker technician, a physician, etc. Although every medical personnel can operate the expert system, chances are that not everyone can understand the questions asked by the system and how to perform some of the operations on the patients requested by the system. To operate the expert system successfully some moderated knowledge on paced ECG interpretation is necessary.

Domain expert
An individual who is widely recognized as having the knowledge and knowhow necessary to solve a particular type or class of problem. This person has learned to focus quickly on the important facets of the problem. This individual contributes expertise, in collaboration with the knowledge engineer, for the creation of a system that can function as an expert in a given field, solving problems efficiently and effectively [Martin, 1988].

Follow-up
Implanted pacemaker systems must be tested periodically, this is called the pacemaker follow-up. This normally takes place 1,3,6, and 12 months after implantation the first year and then once or twice a year. After a pacemaker replacement in which the old electrodes are retained, follow-up can be on an annual basis after the initial post-op appointment. These checkups test whether pacemaker sensing and stimulating is done correctly, that the remaining battery capacity is satisfactory and that no infection or other local problems have developed around the implanted system. The pacemaker operating mode and/or parameters may be programmed at these visits in order to optimize treatment or adapt the pacemaker to new conditions developing when the degree of block or heart disease changes. How the pacemaker mode is changed depends on the patient and on the type of pacemaker [Lindgren, 1992].

Knowledge
Cognizance; the fact or condition of knowing something with familiarity gained through experience or association; the fact or condition of being aware of something; the range of one’s information or understanding; the fact or condition of having information; the sum of what, an accumulation of the body of truth, information, and principles acquired by an individual or by humanity. Encoding of facts affording the ability of using these encoded facts in practical interactions. The ability to form a mental model that accurately describes the object and represents the actions that can be performed by and on that object. Facts, beliefs, and heuristic rules. The integration of a collection of facts and relations [Martin, 1988].
Knowledge acquisition
The main objective of the knowledge acquisition task is to produce and verify the knowledge required by the expert system. There are several techniques available to acquire the knowledge from the expert. Those acquisition techniques are necessary for structuring, updating, and maintaining the knowledge-base of the expert system.

Knowledge-base
Computer programs using knowledge and inference procedures for solving problems that are difficult enough normally to require a significant amount of human expertise to arrive at their solution. They structure data and reasoning rules that link the evidence about a problem to derived conclusions. Such systems, which contain the knowledge of a particular expert on a specific subject, may be used as a substitute for an expert human consultant who is unavailable at the time needed and may incorporate knowledge acquired from human experts and apply it in novel ways. Knowledge-based system components include a knowledge base (consisting of facts and rules of thumb about the domain), a database of current dynamically changing data, and control mechanisms for finding and using the knowledge [Martin, 1988].

Pacemaker programmer
Programmability often allows noninvasive correction of pacemaker malfunction. The information sent and received between an external programmer and pacemaker is called the pacemaker telemetry data. Before a pacing problem is corrected with an external pacemaker programmer, the problem must be appropriately diagnosed and the clinical status of the patient (especially the degree of the patient's pacemaker dependency and the safety margin left after programming) must be considered [Weston, 1991].

Real-time expert system
A knowledge-based system that contains the knowledge of a narrow focused field. The knowledge is acquired from a specialist, the domain expert. The system contains the facts, rules, and reasoning strategy of the human expert. Knowledge based system operating in a real time situation will typically need to respond to a changing task environment involving an asynchronous flow of events and dynamically chaining requirements with limitations on time, hardware and other resources. In contrast to 'normal' expert systems, real time expert systems must be able to recognize and respond to an external event within a certain period of time. Real time expert systems operate in environment in which the data is not static.

(pacemaker) Telemetry data
Pacemaker telemetry allows information to be transmitted from the pulse generator (pacemaker) via a radio frequency signal to a receiver in the pacemaker programmer head. There is a variety of information that is available by pacemaker telemetry. The standard information includes the programmed settings of the pacing system and the pacemaker identification [Van Gelder, 1995].
References

KENNISYSTEMEN: TOEPASSINGEN VAN KUNSTMATIGE INTELLIGENTIE.

NEW PERSPECTIVES IN CARDIAC PACING 2

NOTATION SYSTEM AND OVERLAY DIAGRAM FOR ANALYSIS OF PACED ELECTROCARDIOGRAMS.

[Blom, 1990] Blom, J.A.
THE SIMPLEXYS EXPERIMENT. REAL TIME EXPERT SYSTEM IN PATIENT MONITORING.

[Boutkan, 1969] Boutkan J.
ABC VAN HET ECG. INLEIDING TOT DE ELECTROCARDIOGRAFIE.

[Clancey, 1985] Clancey, W.J.
HEURISTIC CLASSIFICATION.

[Durking, 1994] Durkin J.
EXPERT SYSTEMS. DESIGN AND DEVELOPMENT

[Greenhut, 1992] Greenhut, S.E.
COMPUTERIZED INTERPRETATION OF THE PACED ECG.

COMPUTER MODELLING AND INTERPRETATION OF PACED ELECTROGRAMS FOR ANALYSIS OF PACEMAKER FUNCTION.

[Jackson, 1990] Jackson P.
INTRODUCTION TO EXPERT SYSTEMS. SECOND EDITION.
KNOWLEDGE ACQUISITION FOR EXPERT SYSTEMS: A PRACTICAL HANDBOOK

HEART PHYSIOLOGY AND STIMULATION, AN INTRODUCTION

BUILDING EXPERT SYSTEMS: A TUTORIAL.

[Normann, 1988] Normann, R.A.
PRINCIPLES OF BIOINSTRUMENTATION

THE CARDIAC RHYTHMS: A SYSTEMATIC APPROACH TO INTERPRETATION.

[Smith, 1987] Smith, E.D.
EXPERT SYSTEM FOR IMPLANTED PACEMAKER ANALYSIS

THE ECG IN THE DIAGNOSIS OF PACEMAKER FUNCTION AND MALFUNCTION.

A PRACTICAL GUIDE TO CARDIAC PACING.

KENNISSYSTEEMEN EN MEDISCHE BESLUITVORMING.

[WPE, 1989] onder red. van B. van Mastrigt, B.J. en C.J. van de velde
SLAGEN EN FALEN VAN KENNISSYSTEMEN: RESULTATEN VAN EEN ONDERZOEK ONDER 128 NEDERLANDSE ORGANISATIES.

PRINCIPES VAN KLINISCH DENKEN EN HANDELEN.
Utrecht: Bohnm Scheltema & Holkema, 1980
Appendices

Appendix A: The help function for the questions.


Appendix D: The Push-Pop function.

Appendix E: The implemented knowledge and strategy used in the first and second prototype.
Appendix A: The Help function for the questions.

When help for the questions is requested, the expert system will activate the Microsoft Windows help program "Winhelp". This program can read an RTF-file that stores the help text for the questions. The following files and programs are necessary to create the help file:

- `hc31.exe`: The help compiler.
- `askhelp.rtf`: The help file generated by ruc41ex1.pas.
- `askhelp.hpj`: The help project file.

The following tools and files are needed to create a help file:

- The Winhelp application: WINHELP.EXE. When you create a Help system, you are actually creating a resource file that is used by the WinHelp application. Therefore, you (and your users) also need the Winhelp application to run the help file.

- A word processor for creating topics. The word processor must be able to:
  - Save files in Rich Text Format (.RTF).
  - Create footnotes that are identified by custom marks.
  - For simple Help files, you will create footnotes identified by the custom marks $, #, K, and +.

- A word processor or text editor that can save files as ASCII text, for creating the Help project (.HPJ) file.

- The Microsoft Windows Help compiler (HCP.EXE or HC31.EXE). Both HCP.EXE and HC31.EXE compile Help files to be used with Windows 3.1. They cannot create Help files to be used with Windows 3.0. HCP.EXE is the "protected mode" compiler, which makes better use of system memory. Use HCP.EXE to compile a Help file from a DOS window under Windows.

- The error message resource file for the Help compiler you are using (HCP.ERR or HC31.ERR). This file contains the warning and error messages that WinHelp produces if there are any problems during the compile.

Overview of the process
These are the basic steps for creating a Help file:

1. Write the topics that make up the Help file. Save them as Rich Text Format (.RTF) files.
2. Write a Contents topic. Save it as a Rich Text Format (.RTF) file.
3. Write a Help project (.HPJ) file. Save it as a text file.
4. Compile the topics into a Help resource (.HLP) file.

For more detailed information on how to create a Windows help file see the help file "Creating Windows help". This file is one of the help files that is part of the Delphi 1.0 programmer's toolbox.
Generation of the askhelp.rtf file

The frame of the RTF-file is generated with the program ruc41ex1.exe. This file contains only the questions and not the explanations on the questions. Those explanations have to be appended to the file by the domain expert or by the knowledge engineer.

To generate the RTF-file, change the boolean variable “make_RTF_ASKHELP” to true. This variable can be found in the procedure “output_info” in the ruc41ex1.pas file. Compile ruc41ex1.pas with a pascal compiler and run the executable. Ruc41ex1.exe will then generate the RTF-file “temp.hlp”. The next step is to complete the RTF-file with a word processor and change the name to askhelp.rtf.

Recommendation

Completing the help file with a word processor is laborious. Simplexys, as it is now, does not support the generations of a complete help file. A possibility to create those help files automatically is by adjusting Simplexys. Two new commands should be added to the Simplexys language; ASKHELP and END_ASKHELP. The following example will explain how those new commands should be used.

- Naam_ask_rule: ’the question of the ask rule’
  ASK
  ASKHELP
    '{'help text'}
    '{'help text'}
    '{'help text'}
  END_ASKHELP

The above-mentioned rule shows how an ask-rule looks in a rule base. The text (strings) between ASKHELP and END_ASKHELP is the help text, the explanation on the question. When the rule compiler compiles the Simplexys rule base, the help text should be stored in a file in the same way as the questions of the ask-rules are stored. That file should be used for the generation of the RTF help file.

Read appendix C, the programmer reference of the DOS version, before studying the programmer reference of the Windows version. Appendix C is more comprehensible then Appendix B. In this section are only the differences between the DOS version and the Windows version explained.

- wproto10.rul: The Simplexys rule base for pacemaker evaluation.
- ruc41ex1.pas: The rule compiler & RTF-help file generator.
- unit2.pas: The expert system builder & Windows interface.
- about.pas: The about window.
- data.pas: The data acquisition window.
- mes.pas: The question box.

The rule bases of the DOS and Windows version are almost compatible. The RULES section and the PROGRESS section of both versions are for 99.9% identical. The DOS version asks the user if he wants to evaluate another paced ECG. This is not necessary in the Windows version. The Window version will be active until the user deactivates the program.

Porting the Simplexys expert system builder to Windows means that all DOS screen operations have to be redirected to the Windows environment. Adapting Simplexys to Windows means that most of the procedures and functions describe in the rule base of the DOS version (proto10.rul) are adjusted or removed.

The following procedures were not necessary in the Windows environment:

```pascal
procedure getvideo(var videoboard: videoname);
procedure init;
procedure kader(s: string; x1, y1, x2, y2: byte);
procedure analysis_screen;
procedure restart_same_screen;
procedure pause_hide_cursor;
```

The following functions and procedures were adjusted for Windows:

```pascal
function intstr(l: longint; n: integer): string;
procedure write_dialogue(s: string);
procedure write_window(window_nr: integer; s: string);
procedure restart_same
procedure get_info;
function keyhandler: string;
```

function keyhandler;

The functionality of "Keyhandler" is expanded. This function reports to the user which rule is evaluated at the moment, as in the DOS version. But the keyhandler is now also the link between the inference engine and the Windows help program Winhelp. The Keyhandler is activated when the ASKyn or ASKval Simplexys functions are active.

- proto010.rul: The Simplexys rule base for pacemaker evaluation.
- ruc41 ex.pas: The rule compiler.
- sim41 ex.pas: The expert system builder.

See readme.txt for the adjustments.

The outline of the Simplexys rule base (program)

A Simplexys rule base consists of seven sections. The first five program sections are optional; they are necessary only if the rules need to interface with Pascal code (e.g., to perform acquisition of data, print output etc.). Program sections 6 and 7 are mandatory. The program sections need to appear in the following order:

1. DECLS declarations
2. INITG global initializations
3. INITR run initializations
4. EXITR run exit code
5. EXITG global exit code
6. RULES rules
7. PROCESS protocol

In the following paragraph the program sections will be examined.

1. Declarations

In this section, all variables must be declared and their type specified. The declaration must be represented in valid Pascal. Similarly for all procedures and functions used in the rule base. In the text below, the procedures and functions are described individually, so that the knowledge engineer can use these procedures/functions during the implementation of the rules.

function intstr(I:longint; n:integer):string;
    This function converts a numeric value of the type longint into a string with n characters. It uses the standard pascal procedure Str.

procedure keyhandler; interrupt; forward;
    While procedural precedence enforces a desirable order in the program, there are times when this is not desirable. For these cases, Turbo Pascal provides the forward declaration, which informs the compiler that a procedure exists before it specifies what the procedure does. To declare a procedure as an interrupt handler, you need only append the interrupt directive to the end of the procedure declaration. The interrupt is activated by the Simplexys functions ASKval and ASKyn.

procedure getvideo(var videoboard:videoname);
    For optimal use of computer display, the video adapter must be determined. The video adapter is a circuit board that connects the CPU to a monitor. This procedure returns the video adapter (VGA, EGA, HERC, MDA, CGA or UNKNOWN) to set a video display mode.

procedure init;
    Personal computers have two fundamental video modes: text and graphics. After determining the video adapter, the text mode is initialized only for a VGA, EGA or HERC monitor; otherwise, the program is terminated. Personal computers support up to ten different text display modes. These modes control both the size of the characters and the colors in the
display. Each character has its own foreground and background color. The foreground color is the color of the character itself (white); the background color is the color of the space around the character (blue). The active screen size is defined with the procedure \texttt{Window(X1,Y1,X2,Y2:byte)}. This procedure restricts the active screen to the rectangle defined by coordinates \((X1,Y2)\) and \((X2,Y2)\). Each time the computer is activated, DOS fills the interrupt vector table with addresses to standard interrupt routines. It is however possible to write a specific interrupt handling procedure, which will be executed by replacing an existing interrupt. An interrupt address is changed in the program as following: \texttt{Getintvec($99,intsave)}; \texttt{Setintvec($99,Addr(keyhandler))}; The function \texttt{Addr} returns the address of the procedure \texttt{keyhandler}. The contents of the interrupt procedure will be explained below.

\begin{verbatim}
procedure write_dialogue(s:string);
    To communicate with the user, a window structure is chosen for the display. To emphasize the lay out, the characters are made light-green. Each string is also written to a dumpfile (the file simplex.dmp), so that at the end of the evaluation the dialogue can be read back.

procedure kader(s:string;x1 ,y1 ,x2,y2:byte);
    This procedure creates a frame to support the efficiency of the expert system. Visual aspects can stimulate the understanding of the reasoning process.

procedure analysis_screen;
    The screen layout consists of three blue window displays. The upper window displays pacemaker specifications regarding the type of pacemaker, electrode configuration, available options and timing intervals. The middle window is called the "progress of analysis" window. This window shows which task is evaluated currently, so that the user understands the background of the question in the dialogue window. In the dialogue window the user answers the questions raised by the expert system simply by "y", "n" or "?" (yes, no, unknown).

procedure restart_same_screen;
    After an evaluation of pacemaker function, the question to repeat the evaluation with the same pacemaker specifications is asked. If the user agrees, the redundant data on the screen will be removed.

procedure write_window(window_nr:integer;s:string);
    Restricting output to just to one of the three windows is desirable. For example, the command \texttt{Window(2,2,40,6)} restricts the program's display to a 39 by 5 rectangle starting at column 2 and row 2. Turbo Pascal treats the window as if it were the entire screen; when text runs off the bottom of the window, the screen scrolls up one line. After the data is written, the command \texttt{Window(x1_0,y1_0,x2_0,y2_0)} activates the normal operation of the monitor. The cursor returns to the position of the full screen, before the procedure was started. A case statement declares 4 different windows instead of two; consider that the procedure \texttt{write_dialogue} establishes the user-computer communication regarding the evaluation of the pacemaker function. The upper window is divided into two sections, so that the pacemaker specifications fit in the upper window. The third window informs the user of the progress of the analysis. The last window is a single line to inform the user of the next action (Press any key to continue). The position of the fourth window is at the bottom of the screen.

procedure pause_hide_cursor;
    The function of this procedure is to attract the attention of the user to a certain feature. This feature can be a conclusion or a remark based on the evaluation of the diagnosis. To inform the user how to continue, the following command line is used: \texttt{write_window(4,'Press any key to continue').} The cursor is hidden during the execution of the procedure.
\end{verbatim}
procedure restart_same
Each run ends with the question whether to end or to restart the program. The user can then
test the pacemaker function of a different patient. This procedure is started whenever the
same pacemaker data is used.

procedure get_info;
The pacemaker data of the patient is acquired and is stored in the record paceinfo. This
procedure makes use of some Simplexys utilities:
\texttt{ASKyn(t:string):bool}
\texttt{ASKint(t:string;imin,imax:integer):integer}
The first function prints string t and waits for a 'y' or 'n' followed by a return. The second
function waits for an integer input, which must be between imin and imax, followed by a
return. An error message appears on the screen whenever an inconsistent input is given by
the user.

procedure keyhandler;
This procedure reports to the user which rule is evaluated on the moment. During the process
of diagnosis the user is informed about the reasoning process.

2 Global initializations
The statements in INITG section will be executed when the expert system starts up. The
expert system starts accordingly with the execution of the procedures \texttt{init} and \texttt{get_info}.

3 Run initializations
The statements in INITR section will be executed at the start of each run, including the first
one.

4 Run exit code
The statements in EXITR section will be executed at the end of each run, including the
last one.

5 Global exit code
The statements in EXITG section will be executed only once, at the end of the last run.
During this section the initialized screen lay out is first undone and secondly the interrupt
vector table is restored.

6 Rules
Section 6 starts with the keyword RULES. All the rules must be inserted here; a rule consists
of two to four parts:
rule header (name and text string) mandatory
rule type of rule's expression mandatory
initial conclusion optional
THELSEs optional

The execution of the program depends on the state-rules. The STATE rule
PACEMAKER_CHECK is initially true, so that the OUTPUT_CHECK becomes the (first) goal
rule, the evaluation starts then.

7 Protocol
Program section 7 starts with the keyword PROCESS. This section describes the time
strategy used during the pacemaker follow-up.
Appendix D: The Push-Pop function.

The Push-Pop function is implemented with two variables and three pascal procedures. Three sections of the Simplexys rule base are used to apply this function; the declaration section (DECLS), the rule section (RULES), and the protocol section (PROCESS) (see appendix C for more information on the Simplexys program sections).

In the declaration section the variables and procedures are declared and their type is specified. In the rule section the variables and procedures are combined with Simplexys rules. In the protocol section some “combined” rules (the trigger rules) are used to determine what the next pacemaker evaluation task should be.

Declarations

In the declaration section the following variables and procedures are declared:

**current_process**
- This variable stores the identification name of the pacemaker evaluation task that is currently being evaluated, e.g., output check, capture check, sensing check, diagnosis oversensing etcetera.

**Next_process**
- The identification name of a pacemaker evaluation task that subsequently should be evaluated, is stored in the variable Next_process.

The contents of those variables are changed by three procedures.

**procedure Goto_next_process(CurrentProcess, NextProcess);**
- This procedure decides what the next task will be. The variables “Current_process” and “Next_process” are changed in this procedure.
  - Next_process := NextProcess;
  - Current_process := CurrentProcess;

**procedure Move_current_process_val2Next_process;**
- When the expert system evaluates the next task (next_process) then it should remember what the previous task was. To remember the last evaluated task, the identification name of the previous task (current_process) is stored in the variable “next_process”. The previous task becomes the next task that should be evaluated.
  - pre: VAR next_process, VAR current_process
  - post: next_process :=current_process & current_process := false

**procedure Proceed_with_last_check_or_End_prog;**
- This procedure asks, after an evaluation of a diagnosis task, if the user want to stop or continue with the evaluation. If the user wants to re-evaluate the previous (analysis) task then the program will continue with the task stored in the “next_process” variable. If the user wants to stop, the “next_process” variable will get the identification name “end_program”. This is a sign to the program to end the pacemaker evaluation.
Rules section

In the rule section the procedures are combined with the rules by means of "THELSEs". This is a Simplexys command to link Pascal procedures with Simplexys rules.

As mentioned in this report the rule base can be divided into an analysis and a diagnosis section. The analysis tasks use the procedure Goto_next_process and the diagnosis tasks use the procedure Proceed_with_last_check_or_End_prog. Procedure Move_current_process_val2Next_process is used in the "trigger rules" which are required for the process section of the rule base.

All the hypotheses of the analysis trees are linked with a Goto_next_process procedure. This linking is part of the heuristic matching process. A hypothesis decides what the heuristic match should be and thus the next task that should be evaluated.

The procedure Proceed_with_last_check_or_End_prog is placed in the root of the diagnosis trees. Without this procedure the expert system would automatically re-evaluate the previous (analysis) task.

Process section

This section stores the time strategy of the pacemaker evaluation process. After every run the expert system looks in this section for the next task. The expert systems walk through a list of trigger rules. Every trigger rule represents a different pacemaker evaluation task. The trigger rules are linked with the variable "next_process" and thus the trigger rules are indirectly linked with the procedures Move_current_process_val2Next_process, Goto_next_process procedure, and Proceed_with_last_check_or_End_prog. Those procedures are linked with the rules of the rule base. By using this construction the rule base of the third prototype is easier to maintain then the first and second prototypes. Another advantage of this method is that the third prototype is more flexible to switch from one task to another task, compared with its predecessors.
Appendix E

Implemented domain knowledge and the applied strategy knowledge:
The first prototype

The analysis and diagnosis function of this prototype were not complete. The sensing check, diagnosis oversensing, diagnosis undersensing, and diagnosis intermittent output are not implemented in the first prototype. The flow diagram shows that the system can follow just one direction. If a task is evaluated, it cannot be reevaluated again. This is caused by the method used for switching from one task to another.

The system keeps a record of all the (goal) tasks it should evaluate. By removing a task from the record the system knows that a specific task has been evaluated or does not have to be evaluated. For example, when the pacemaker evaluation starts with the output check, the record consists of the following list:

Record : evaluate the following (goals) tasks:
- Capture check
- Diagnosis no output
- Diagnosis continues noncapture
- Diagnosis intermittent noncapture

When, while the output check continues, noncapture is established, the record looks as follows:

Record : evaluate the following (goal) tasks:
- Diagnosis continuous noncapture

All the other tasks are removed from the list and no new tasks are appended to the list. This is the reason a task cannot be re-evaluated.
Appendix E

Implemented domain knowledge and the applied strategy knowledge:

The second prototype

The second prototype evaluates only the pacemaker analysis process. The pacemaker output function is checked at two different places in the system. The output check at the beginning consists out of one single question: 'Do you see pacing spikes?'. The output check at the end checks the time interval of two consecutively intrinsic beats.

The amount of domain knowledge in the knowledge base is far less than in the first prototype. The first prototype can evaluate more tasks and contains more knowledge about the pacemaker output function and capture function than the second prototype.

The second prototype makes more use of protocols then the first prototype. In that respect the second prototype is more flexible than the first prototype. However, the system does not use this flexibility at a high conceptual knowledge level.