Object Oriented Formal Specifications: Application in the Development of an Automatic Exercise ECG Processing System

LC Carvalho¹, GHMB Motta², SL Meira²

(1) Federal University of Paraiba, J. Pessoa, Brazil
(2) Federal University of Pernambuco, Recife, Brazil

Abstract

Present methods used to develop medical software are empirical or based on personal knowledge, lacking the rigorous approach that applications in this field require. This paper describes the use of object oriented formal specifications in the development of an automatic exercise ECG processing system. Many benefits were achieved using this method: unambiguous and precise documentation; greater reliability, code generality and reusability; better code structure; reduction of program errors and a greater degree of abstraction in the project, that leads to a closer relationship between the application domain and the implemented software. The system acquires and processes three ECG leads reporting various ECG measurements. Good results were obtained using the system in pilot experiments with twenty normal individuals.

1. Introduction

Software developed for medical applications is, each day, more important and is, actually, substituting many tasks previously performed by hardware. Each day it becomes more complex, requiring a great deal of effort to develop and maintain. It has to be reliable, efficient and easily reused. Most of the time, however, methods used to conceive them are inadequate, strongly based on pragmatic considerations, without taking formal aspects into account.

This paper deals with the application of modern Software Engineering techniques to develop a system that acquires and automatically processes exercise ECG's. An object oriented formal specification language, based on Z [1], called MooZ [2], was used as a notation to make software description, documentation and design.

It was considered essential to have well-defined steps in the software construction. These steps were: 1) system requirement analysis 2) formal specification 3) project implementation 5) validation.

1.1. Formal Specifications

Current approaches to software system development generally involve natural language specification of system requirements, functions and properties. Although this approach may lead to an adequate system description, many important details to accomplish steps 3 and 4 are not precisely described. The final result is that such descriptions exhibit many deficiencies as ambiguities and inconsistencies, leading to interpretation errors that propagate throughout all development phases.

Formal specifications allow for the correction of these deficiencies, because they describe system requirements consistently, using mathematical models, in such a fashion that it is possible to verify system properties, even before one writes a single line of code. Correction and completeness of the software can be checked against the specification, using formal proofs. The final product is more reliable, better structured and development cost is probably less, when compared to conventional techniques.

1.2. Object Orientation

The object orientation paradigm includes fundamental properties such as encapsulation, abstraction and modularity, resulting in powerful means to define and manage large and complex systems. Among these properties there are concepts such as object, classes and inheritance. This paradigm has a great power in modeling and classification, allowing easy program evolution and reusability. Systems are developed in an incremental and modular manner, comprising objects that communicate among themselves through messages.

2. System Development

The development was divided in abstraction and satisfaction tasks. In the abstraction task the system specification resulted in a concrete specification, regarding
computing data structures and program language implementation. The specification uses a set of abstract definitions of classes, interrelated by a hierarchy. Table 1 shows the main classes hierarchy. For instance, the Signal class is an ancestor of several kinds of signals in this application domain and is modelled as a partial function of time to volts. The class SampledSignal is defined as a subclass of Signal being, thus, a partial function and having, in addition, a state variable to store the sampling frequency. The whole MooZ specification has 32 classes with 188 operations attached to them.

Table 1 Classes system hierarchy

<table>
<thead>
<tr>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnalogSignal</td>
</tr>
<tr>
<td>SampledSignal</td>
</tr>
<tr>
<td>ECG</td>
</tr>
<tr>
<td>Digital Filter</td>
</tr>
<tr>
<td>EventDetector</td>
</tr>
<tr>
<td>HamTomDetector</td>
</tr>
<tr>
<td>LagunaFiducialMarkDetector</td>
</tr>
<tr>
<td>Converter</td>
</tr>
<tr>
<td>Channel</td>
</tr>
<tr>
<td>VisualChannel</td>
</tr>
<tr>
<td>ECGChannel</td>
</tr>
<tr>
<td>Person</td>
</tr>
<tr>
<td>Patient</td>
</tr>
<tr>
<td>Doctor</td>
</tr>
<tr>
<td>Protocol</td>
</tr>
<tr>
<td>Tables</td>
</tr>
<tr>
<td>MedicalSystem</td>
</tr>
<tr>
<td>StressTestSystem</td>
</tr>
</tbody>
</table>

In order to illustrate the formal specification, an operation related to QRS detection, defined in the class SampledSignal and used by the HamTomQRSDetector class, is shown below.

\[
\text{timeAveraging: fun:} \mathbb{P}((\text{TIME} \times \text{VOLTS}) \times \text{xs}) : \mathbb{P}((\text{PERIOD} \times \text{TIME} \times \text{VOLTS}))
\]

\[
\forall \; tw: \text{TIME} \quad \exists \; n: \mathbb{N} \quad n \times \text{ps} = tw \quad \ast
\]

\[
\text{timeAveraging}(\text{fun}, \text{ps}, tw) = \lambda \; t: \text{dom}(\text{fun}) \mid t \leq \text{max}(\text{dom}(\text{fun})) \cdot tw + ps \ast
\]

\[
(\text{ps}/\text{tw}) \times \sum i: t \cdot i + tw - ps \ast \text{fun}(i)
\]

During the implementation phase, all classes were rigorously coded using the object oriented TurboPascal 6.0 language, based on the concrete specification obtained during the project phase.

3. Signal Acquisition and Processing

The software runs on IBM PC compatible microcomputers and acquires three simultaneous leads from a high performance three channel ECG amplifier, having a 0.1 to 100 Hz bandwidth. Good quality electrodes, proper skin cleaning and careful electrode and cable positioning, provided ECG signals with a high S/N ratio and small amount of baseline shifts. The sampling frequency is 200 samples/s/channel and the A/D converter resolution is twelve bits. All the classical exercise ECG protocols for treadmills are pre-programmed and the physician can also introduce customized protocols. During the various exercise stages the program displays in real time, using a high resolution monitor, the three ECG leads, the heart rate and other information about that particular stage, as shown in figure 1a. At the end of each stage, ten seconds of the ECG leads are recorded in a hard disk data file, after entering in the keyboard, diastolic and systolic blood pressure values, manually measured.

Signal processing is performed off line. For each lead and exercise stage a reliable QRS detection algorithm [3] detects R (or S) wave positions. A median algorithm is employed to obtain an ECG median waveform [4]. Median algorithms tend to eliminate much of the noise contaminating the signal, including small baseline deviations, but additional smoothing is used, and a special linear baseline correction scheme has been devised, fitting a straight line through adjacent TP segments, to correct residual baseline shifts. The next step is waveform segmentation, using algorithms to detect fiducial points. Waveform segmentation is made using Laguna et al algorithms. These algorithms are based on the original ECG waveform, its derivative, a low pass version of this derivative, plus several thresholds [5]. They detect QRS onset, T wave end and peak, Q and R waves, in single leads. Since there was no provision for QRS offset and S wave detection in Laguna algorithms, we developed, using similar principles, algorithms to detect this fiducial mark and wave [6,7]. ST segment end is set to a point located at 3/8ths of the interval from QRS offset to T wave end. This approach to find ST segment end has been used earlier, with good results [8,9]. Thus, the segmentation procedure is able to detect not only the fiducial points, but it also provides waveform positions. Figure 1b illustrates several steps of the signal processing described above.

After segmentation and waveforms position detection, the program measures waveform amplitudes, intervals, and those parameters related to the ST segment, such as ST60, ST80, ST integral, ST index, reporting all measurements in various kinds of written reports, including high quality graphical reports, with ECG traces and measurements, as shown in figure 1c.
Figure 1 Screen during exercise test (a). In (b) steps of signal processing. In (c) a graphical report with measurements.
4. Results

The system described above was initially evaluated in a pilot study with twenty normal males and females, ranging from 18 to 51 years of age, in the University Hospital of the Federal University of Paraiba. The Ellestad protocol was used. Leads CM5, V2 and a modified D2 lead were monitored and recorded during the tests. Graphical reports containing all measured parameters and ECG traces for all exercise stages and for the twenty individuals, such as shown in figure 1c, were plotted and individually examined by a specialist. Additional trend plots of blood pressures, heart rate and double product were also plotted. In all cases, no abnormal parameters were observed. Failure to detect fiducial points correctly was observed in one case. Refinements of these algorithms are being conducted and tested in a large ECG database.

5. Conclusions

The application of object oriented formal specifications in this project allowed a good conception of the system and avoided errors during the implementation phase. Although the system is quite complex, it was implemented in only 5 months and the specification methodology certainly helped to speed up this process. The use of an abstract and structured mathematical notation helped to have a good understanding of the studied problem during the initial phases of the project, even considering its complexity. An unambiguous and precise documentation, which is independent of the implementation and useful for maintenance purposes, resulted from the formal specification. The classes created are general and powerful, and can easily be used in similar applications. These benefits are already being observed in a 12 ECG leads processing system, under development in our laboratory. Good results were obtained in the pilot study, where high quality processed ECG's allowed the measurement of many important exercise ECG parameters. The system is now being further evaluated and enhanced to complete clinical validation.

Acknowledgements

We wish to thank the National Research Council (CNPq) support in the beginning of this project.

References


Address for correspondence:

Luís Carlos Carvalho
NETEB/MEB/DFP - Centro de Ciências da Saúde
Campus I da UFPB - Cidade Universitária
João Pessoa - Paraíba CEP 58000
Brazil
E-mail LCARCCSN@BRUFPB.BITNET