libRASCH: A Programming Framework for Signal Handling

R Schneider, A Bauer, P Barthel, G Schmidt

Munich University of Technology, Munich, Germany

Abstract

For the analysis of physiological signals, such as ECGs, continuous blood pressure recordings etc., access to the raw signal data as well as to processed data is mandatory. Up to now, there is no computer program which allows access to raw and processed data independently from the file formats used. Thus, programs have to be adapted to each new file format. The aim of the programming library 'libRASCH', is to provide an interface which allows the access to physiological signals in a consistent way. libRASCH is written in C and runs under Linux and Windows. The source code of libRASCH is published under the GNU LGPL. A plugin mechanism for extension of the library was implemented. Support for some widely used data formats (e.g. EDF(+), MIT/BIH) is already available. API bindings for several programming languages are available. On the libRASCH web-site www.librasch.org, the source code of libRASCH and further informations are available.

1. Introduction

When analyzing biological signals, access to the raw data as well as to processed data (e.g. position of QRS complexes in an ECG) is mandatory. Because of different demands, both, the industry and the research community created a high number of different data formats for signal storage and signal distribution [1, 2, 3, 4, 5, 6]. To analyze data stored in a new data format, either a program which converts the new format to an already supported one has to be written or the access functionality for the new data format has to be added to the analyzing program(s).

One way to solve this problem is the use of a standard file format, which is powerful enough to handle all needs for storing and distributing signals. For physiological signals, the 'File Exchange Format for Vital Signs' (FEF) [7] tries to accomplish this task. If this format (or a similar one) is accepted by the research community and if the industry provides the possibility to export the signal data in the FEF format, the data access will be facilitated.

Our approach, moreover, is different. We assume that there will be always different data formats (some "standard" formats and a lot of proprietary formats). Therefore we decided to develop the programming library libRASCH, which hides the differences of the data formats behind a common application programming interface (API).

Programs using libRASCH no longer need to be adapted to each new data format. The implementation of a new data format needs only to be done once for libRASCH and all libRASCH based programs can handle the new format without further changes.

Additionally, libRASCH provides the infrastructure to perform processing algorithms in a standardized way and provide support to display the signal data on a computer screen.

2. Design of libRASCH

libRASCH is made up of two parts, the libRASCH core and the libRASCH plugins (see Figure 1). The core is responsible for the management of the plugins and for the communication with the outside world (handling API calls). All other tasks (e.g. access to the different data formats) are done in the plugins.

The libRASCH-API provides functions for the following tasks:

- get informations about libRASCH and the loaded plugins (e.g. name and version of a plugin)
- get informations about a measurement (e.g. patient name, recording date, number of channels)
- access the measurement raw and processed data (e.g. the positions of heartbeats in an ECG)
- process a measurement (e.g. calculate Heart Rate Variability parameters)
- display the recorded data on the computer screen

3. libRASCH Plugins

In libRASCH three types of plugins are available: 1) data access plugins, 2) data process plugins and 3) data display plugins.

Access plugins are responsible for the access of raw and processed data stored on disk. libRASCH automatically selects the access plugin, which can handle the actual data format. The user do not need to specify the data

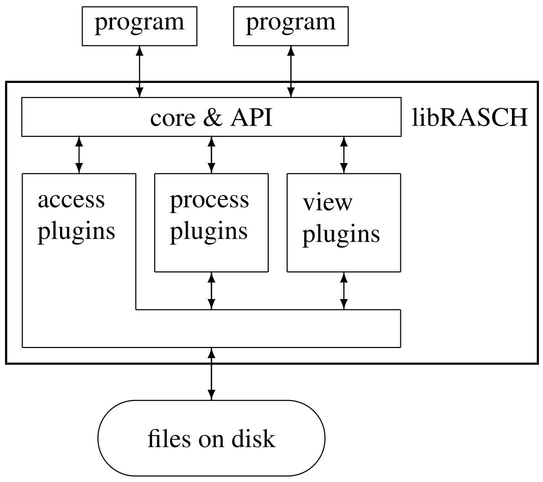

Figure 1. General structure of libRASCH. The core is responsible for the management of the plugins and handles the API calls. The plugins provide access to signals, process signals and display signals.

format used for the current measurement. Table 1 lists the supported data formats.

Process plugins perform actions like the detection of heartbeats in an ECG or the calculation of Heart Rate Variability parameters from a 24h-ECG recording. When implementing processing algorithms as process plugins, the data processing is done in a standardized way. This is because all programs based on libRASCH are using the same implementation of the processing algorithm. Table 2 lists the current available process plugins.

View plugins provide graphical user interface (GUI) elements. These GUI elements are dialog boxes (e.g. measurement selection dialog box) and display windows. Display windows either show the signal raw data or some processed data (e.g. tachograms from 24h-ECGs) on the computer screen. Table 3 lists the current available view plugins.

4. Implementation

The libRASCH core and libRASCH plugins are written in C, the view plugins are written in C++. Different application programming interfaces (API) are available for the following programming languages/environments:

- C
- C++ (Object Oriented interface)
- Perl
- Python
- Octave (Linux version)
- Scilab (Linux version)
- Matlab

Table 1. Data formats supported by libRASCH

Name	Description
CFS	CED Filing System
CTG-rasch	libRASCH specific format for Cardiotocograms recorded with the telemetry system from GMT
DasyLab	Format used in DasyLab Software
EDF/EDF+	European Data Format (plus) [2, 5]
ISHNE-Holter	The ISHNE Holter Standard Output File Format
MIT/BIH	Formats used for the physiological signal databases provided by Physionet [1, 3]
Oxford	Holter ECGs evaluated with Oxford Excel2 systems
Pathfinder	Holter ECGs evaluated with Reynolds Pathfinder 700 systems
Portapres	Arterial blood pressure signals recorded with Portapres
Poly5/TMS32	Format used by Porti ADC system
RRI	libRASCH specific format for heartbeat-intervals

For an updated list, visit www.librasch.org/librasch/formats.html

When designing libRASCH, one prerequisite was that libRASCH can be ported easily to new platforms. Therefore the amount of platform specific function calls was kept as minimal as possible (when possible, functions from the C standard library where used). Parts which are platform specific are marked clearly or were implemented in their own source code files. At the moment Linux and Windows are supported, porting libRASCH to MAC OS X is planned. The view plugins for Linux are using Qt from Trolltech whereas for Windows the Microsoft Foundation Classes (MFC).

Internationalization (i18n) and localization (l10n) is supported by libRASCH. For i18n, all strings inside libRASCH are stored as Unicode (UTF-8) characters. The strings are converted using the C-locale settings when they enter/leave libRASCH. For l10n, the GNU gettext library [9] is used. At the moment translations (additionally to English) for German (complete) and Spanish (partial) are available.

4.1. License

The source code of libRASCH is published under the GNU Lesser General Public License (LGPL) [10]. This license provides free access to the source code of libRASCH and it allows the user to modify/enhance the library when needed.

Table 2. List of available process plugins in libRASCH

Name	Description
ap-morphology	gets systolic and diastolic values/positions of arterial pressure waves
dawes-redman	calculates Dawes/Redman criteria for CTGs
detect-simple	a simple heart beat detection algorithm
detect-ctg	detects uterine contractions in CTGs
ecg	performs ECG-specific calculations (e.g. calculations of RR intervals)
fiducial-point	finds fiducial point of a peak
freq-analysis	transforms a signal from the time-domain to the frequency domain
HRV	calculates heart rate variability parameters [8]
template	combines specific patterns in templates

For an updated list, visit www.librasch.org/librasch/plugins.html

5. Application

Providing support for different programming languages/ systems, libRASCH can be used for the following tasks:

• large scale applications (complex programs for signal analysis with a graphical user interface; mostly implemented using C/C++)

• command line programs, scripts (short programs performing a specific task; mostly implemented using a scripting language like Perl or Python)

• ad-hoc data processing ("playing" with the data; mostly done in a programming environment like Matlab or Octave)

An example for the first type of applications is RASCHlab (coming with the libRASCH distribution), shown in Figure 2. We use the program to evaluate measurements, performed in our Institution. It accomplishes the following tasks:

- detects heartbeats in the ECG
- gages the blood pressure signal
- displays the recorded signal
- provides functions for inspection and correction of the automatic evaluation

All these tasks are implemented as libRASCH plugins. RASCHlab has only to initialize libRASCH, creating a main application window and transfering user input to the responsible libRASCH plugin[1].

[1] After the evaluation of nearly 1000 recordings, libRASCH/RASCHlab has shown its usefullnes and practicability.

Table 3. List of available view plugins in libRASCH

Name	Description
ch-select-dlg	dialog for selecting channels which will be used for further processing
cont-ap-view	view for continuous arterial pressure recordings
ctg-view	view for Cardiotocograms
ecg-view	view for ECGs
eval-dlg	dialog showing informations about the evaluations of a measurement
plot-view	plot data (e.g. tachogram of a Holter recording)
plugin-info-dlg	dialog showing informations about the loaded libRASCH plugins
rasch-view	general view for libRASCH (container for other view plugins)
sig-sel-dlg	"file selector" dialog for measurements handled by libRASCH
template-view	view for templates
ts-view	view for time-series data

For an updated list, visit www.librasch.org/librasch/plugins.html

In our working group, the development of new risk stratification methods is mostly done in Matlab. To access the recorded data in Matlab, we had to export the data in an ASCII file. With the Matlab interface for libRASCH, this time consuming and error-prone task (misinterpretation of the exported values, not up-to-date exported data files) can be skipped and we can directly access the original measurement data in Matlab.

6. Conclusion

When using libRASCH, programmers need only to know and use one API to access a wide variety of different physiological signal formats. It is no longer necessary to add/maintain the access functionality for different data formats in each analyzing program. This reduces the possibility of programming errors and decreases the time needed to implement data access.

The availability of libRASCH for different programming languages allows the programmer to choose the language which fits best to the current problem. One is not tied to one specific programming language because only in this language the access functionality is already available.

Additionally, with the support for processing plugins, the results of the data processing tasks are the same for all libRASCH based programs. Every program use the same implementation of the task (the libRASCH plugin) regardless which programming language is used.

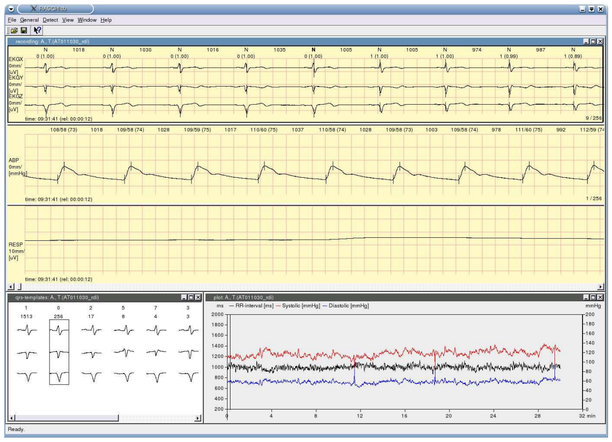
Figure 2. Screen-shot of RASCHlab. The upper half shows the recording (three ECG channels, one continuous blood pressure and one respiration channel). The lower left part shows templates of QRS complexes found in the ECG. The lower right part shows tachograms of the heart rate and the systolic and diastolic blood pressure. All views are implemented as libRASCH plugins.

When new functionality (access to a new data format or calculation of new parameters) is needed, only the libRASCH plugin, which implements the new functionality, has to be written. After copying this plugin in the libRASCH plugin directory, all libRASCH based programs have instantly access to this new functionality.

Up to now, the main focus of libRASCH is to handle physiological signals (e.g. ECG, EEG, blood pressure) but the design of libRASCH allows to handle any kind of signals. There are no problems to access, process and display signals from other research fields (e.g. seismographic recordings).

On the libRASCH web-site [11], the source code of libRASCH, a pre-compiled Windows distribution and further informations are available.

References

[1] Goldberger AL, Amaral LA, Glass J, Hausdorff JM, Ivanov PC, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. PhysioBank, PhysioToolkit and PhysioNet: Components of a new research resource for complex physiological signals. Circulation 2000;101:E215–220.

[2] Kemp B, Värii A, Rosa AC, Nielsen KD, Gade J. A simple format for exchange for the study of digitized polygraphic recordings. Clin Neurophysiol 1992;82:391–393.

[3] Moody GB, Mark RG, Goldberger AL. PhysioNet: a webbased resource for the study of physiologic signals. IEEE Eng Med Biol Mag 2001;20:70–75.

[4] Penzel T, Kemp B, Klosch G, Schlögl A, Hasan J, Varri A, Korhonen I. Acquisition of biomedical signals databases. IEEE Eng Med Biol Mag 2001;20:25–32.

[5] Kemp B, Olivan J. European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data. Clin Neurophysiol 2003;114:1755–1761.

[6] Varri A, Kemp B, Penzel T, Schlögl A. Standards for biomedical signal databases. IEEE Eng Med Biol Mag 2001;20:33–37.

[7] Project team CEN/TC251/PT-40. File exchange format for vital signs, interim report, revision 1.3. Technical report, TC251 Secretariat, Stockholm, Sweden, 2002.

[8] Task Force of the European Society of Cardiology and the American Society of Pacing and Electrophysiology. Heart rate variability: Standards of measurement, physiological interpretation, and clinical use. Circulation 1996;93:1043–1065.

[9] Website of GNU gettext. URL `http://www.gnu.org/software/gettext/`.

[10] Free Software Foundation. GNU lesser general public license version 2.1, 1999. URL `http://www.gnu.org/licenses/lgpl.html`.

[11] Website of the libRASCH project. URL `http://www.librasch.org`.

Address for correspondence:

Raphael Schneider
Klinikum r.d. Isar, 1. Med. Klinik / Ismaninger Str. 22 / D-81675 München / Germany
tel./fax: +49-89-4140-4100/4862
rasch@med1.med.tum.de