Diabetes decision-support module (Perl)[1]

RWD Nickalls,
Department of Anaesthesia,
Nottingham University Hospitals,
City Hospital Campus,
Nottingham, UK.

dick@nickalls.org
www.nickalls.org

An open source anaesthesia workstation (Linux)
revision: 2009α

[1] `http://www.nickalls.org/dick/papers/xenon/hand-diabetes.pdf`

Chapter 14

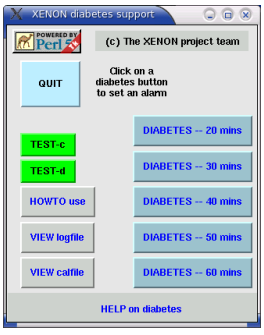
Diabetes decision-support module (Perl)

These notes, which are not comprehensive, guided my own interfacing programs for use in conjunction with an experimental Anaesthesia Workstation.

14.1 Introduction

This Perl diabetes decision-support module consists of a diabetes widget which offers information and support as well as an alerting system to remind the anaesthetist to repeat blood sugars etc. This alert system uses the excellent Linux KDE **Kalarm** utility (see below). The **Kalarm** version currently being used with the Xenon workstation (v.0.8.3).

Kalarm is a sophisticated system, and the latest version (1.4.0) is capable of sending emails, displaying text files, triggering an audible voice message, as well as displaying a coloured alert banner following a specified alarm interval, or at a specified date/time. The **Kalarm** system allows input either via a 'form' or via the command-line. The 'form' input method (mouse & keyboard) is, however, rather too complicated and time consuming for use in the theatre environment—input errors would be likely, making the system sufficiently unreliable for anaesthesia use. It was therefore decided to write a Perl-Tk program to generate a widget and info system, which would allow a diabetes alert to be set easily and reliably, simply by clicking on an appropriate widget button.



14.2 Screenshots

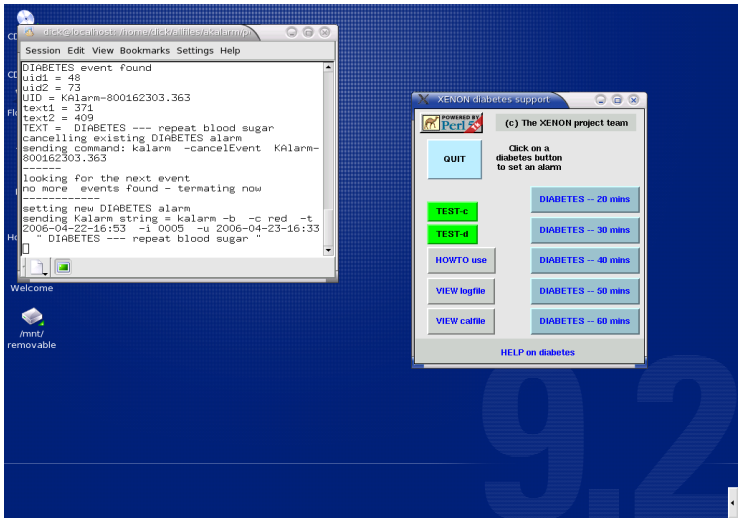


Figure 14.1:
Screen showing the diabetes alarm widget (right) and the Linux command-line window (left). The widget displays 5 blue time-option buttons (20–60 minutes) which initiate the red interval alarm as shown in the following figure.

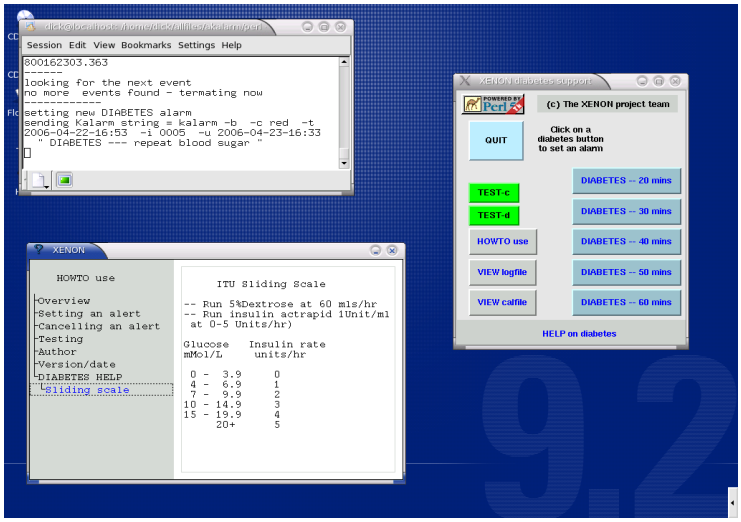


Figure 14.2:
Screen showing the alarm help-window (bottom left) which opens by clicking on the 'HOWto use' button. The help-window doubles as a diabetes management information as well as a help feature for using the alarm widget itself.

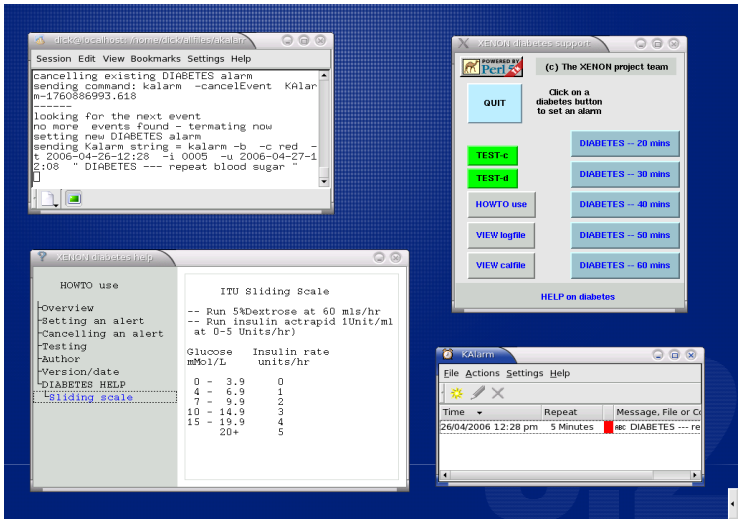


Figure 14.3:
Screen showing in additon the Linux alarm window (bottom right) which opens by clicking on the ‘alarm’ icon on bottom bar.

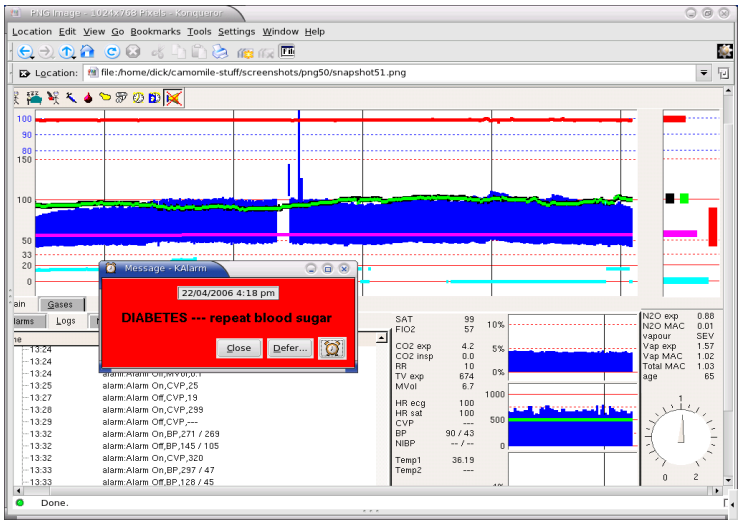


Figure 14.4:
Screen showing the pop-up diabetes alarm. Clicking the ‘close’ button causes the alarm to close and re-appear in 5 minutes. Once a blood sugar has been done, then a new interval alarm is set by clicking on one of the time-option buttons (20–60 minutes) on the diabetes widget.

14.2.1 Kalarm and the iCalendar standard

Kalarm data is written to a text file encoded using the iCalendar Syntax Reference Standard 2445 (RFC 2445), which uses a number of nested so-called V-items, e.g. Valarm, Vevent etc. The following extract is from the Wikipedia entry for *iCalendar* (http://en.wikipedia.org/wiki/ICalendar).

> iCalendar is a standard (RFC 2445 or RFC2445 Syntax Reference) for calendar data exchange. The standard is also known as "iCal", which is the name of the Apple Computer calendar program that was the first software implementation of the standard.
>
> iCalendar allows users to send meeting requests and tasks to other users through emails. Recipients of the iCalendar email (with supported software) can respond to the sender easily or counter propose another meeting date/time. It is implemented/supported by a large number of products, including 30Boxes, Google Calendar, Apple iCal application and iPod, Chandler, Lotus Notes, ScheduleWorld, KOrganizer, Lovento, Mozilla Calendar (including Mozilla Sunbird), Mulberry, Novell Evolution, Kronolith, Simple Groupware, Windows Calendar, Nuvvo, Upcoming.org and to some extent, Microsoft Outlook ... iCalendar data is typically exchanged using traditional email, but the standard is designed to be independent of the transport protocol. For example, it can also be shared and edited by using a WebDav server. Simple web servers (using just the HTTP protocol) are often used to distribute iCalendar data about an event and to publish busy times of an individual. Event sites on the web are embedding iCalendar data in web pages using hCalendar, a 1:1 representation of iCalendar in semantic XHTML.

14.2.2 VALARM specification from the RFC-2445 manual (v:2, Nov 1998)

```
Internet Calendaring and Scheduling Core Object Specification (iCalendar)
Copyright (C) The Internet Society (1998).  All Rights Reserved.


4.6.6 Alarm Component
   Component Name: VALARM
   Purpose: Provide a grouping of component properties that define an
   alarm.

   Formal Definition: A "VALARM" calendar component is defined by the
   following notation:

          alarmc     = "BEGIN" ":" "VALARM" CRLF
                       (audioprop / dispprop / emailprop / procprop)
                       "END" ":" "VALARM" CRLF

     audioprop  = 2*(

                ; 'action' and 'trigger' are both REQUIRED,
                ; but MUST NOT occur more than once

                action / trigger /

                ; 'duration' and 'repeat' are both optional,
                ; and MUST NOT occur more than once each,
                ; but if one occurs, so MUST the other
```

```
            duration / repeat /

            ; the following is optional,
            ; but MUST NOT occur more than once

            attach /

            ; the following is optional,
            ; and MAY occur more than once

            x-prop

            )


dispprop    = 3*(

            ; the following are all REQUIRED,
            ; but MUST NOT occur more than once

            action / description / trigger /

            ; 'duration' and 'repeat' are both optional,
            ; and MUST NOT occur more than once each,
            ; but if one occurs, so MUST the other

            duration / repeat /

            ; the following is optional,
            ; and MAY occur more than once

            *x-prop

            )


emailprop   = 5*(

            ; the following are all REQUIRED,
            ; but MUST NOT occur more than once

            action / description / trigger / summary

            ; the following is REQUIRED,
            ; and MAY occur more than once

            attendee /

            ; 'duration' and 'repeat' are both optional,
            ; and MUST NOT occur more than once each,
            ; but if one occurs, so MUST the other
```

```
            duration / repeat /

            ; the following are optional,
            ; and MAY occur more than once

            attach / x-prop

            )


procprop    = 3*(

            ; the following are all REQUIRED,
            ; but MUST NOT occur more than once

            action / attach / trigger /

            ; 'duration' and 'repeat' are both optional,
            ; and MUST NOT occur more than once each,
            ; but if one occurs, so MUST the other

            duration / repeat /

            ; 'description' is optional,
            ; and MUST NOT occur more than once

            description /

            ; the following is optional,
            ; and MAY occur more than once

            x-prop

            )

Description: A "VALARM" calendar component is a grouping of component
properties that is a reminder or alarm for an event or a to-do. For
example, it may be used to define a reminder for a pending event or
an overdue to-do.

The "VALARM" calendar component MUST include the "ACTION" and
"TRIGGER" properties. The "ACTION" property further constrains the
"VALARM" calendar component in the following ways:

When the action is "AUDIO", the alarm can also include one and only
one "ATTACH" property, which MUST point to a sound resource, which is
rendered when the alarm is triggered.

When the action is "DISPLAY", the alarm MUST also include a
"DESCRIPTION" property, which contains the text to be displayed when
the alarm is triggered.
```

```
When the action is "EMAIL", the alarm MUST include a "DESCRIPTION"
property, which contains the text to be used as the message body, a
"SUMMARY" property, which contains the text to be used as the message
subject, and one or more "ATTENDEE" properties, which contain the
email address of attendees to receive the message. It can also
include one or more "ATTACH" properties, which are intended to be
sent as message attachments. When the alarm is triggered, the email
message is sent.

When the action is "PROCEDURE", the alarm MUST include one and only
one "ATTACH" property, which MUST point to a procedure resource,
which is invoked when the alarm is triggered.

The "VALARM" calendar component MUST only appear within either a
"VEVENT" or "VTODO" calendar component. "VALARM" calendar components
cannot be nested. Multiple mutually independent "VALARM" calendar
components can be specified for a single "VEVENT" or "VTODO" calendar
component.

The "TRIGGER" property specifies when the alarm will be triggered.
The "TRIGGER" property specifies a duration prior to the start of an
event or a to-do. The "TRIGGER" edge may be explicitly set to be
relative to the "START" or "END" of the event or to-do with the
"RELATED" parameter of the "TRIGGER" property. The "TRIGGER" property
value type can alternatively be set to an absolute calendar date and
time of day value.

In an alarm set to trigger on the "START" of an event or to-do, the
"DTSTART" property MUST be present in the associated event or to-do.
In an alarm in a "VEVENT" calendar component set to trigger on the
"END" of the event, either the "DTEND" property MUST be present, or
the "DTSTART" and "DURATION" properties MUST both be present. In an
alarm in a "VTODO" calendar component set to trigger on the "END" of
the to-do, either the "DUE" property MUST be present, or the
"DTSTART" and "DURATION" properties MUST both be present.

The alarm can be defined such that it triggers repeatedly. A
definition of an alarm with a repeating trigger MUST include both the
"DURATION" and "REPEAT" properties. The "DURATION" property specifies
the delay period, after which the alarm will repeat. The "REPEAT"
property specifies the number of additional repetitions that the
alarm will triggered. This repitition count is in addition to the
initial triggering of the alarm. Both of these properties MUST be
present in order to specify a repeating alarm. If one of these two
properties is absent, then the alarm will not repeat beyond the
initial trigger.

The "ACTION" property is used within the "VALARM" calendar component
to specify the type of action invoked when the alarm is triggered.
The "VALARM" properties provide enough information for a specific
action to be invoked. It is typically the responsibility of a
"Calendar User Agent" (CUA) to deliver the alarm in the specified
fashion. An "ACTION" property value of AUDIO specifies an alarm that
causes a sound to be played to alert the user; DISPLAY specifies an
```

```
alarm that causes a text message to be displayed to the user; EMAIL
specifies an alarm that causes an electronic email message to be
delivered to one or more email addresses; and PROCEDURE specifies an
alarm that causes a procedure to be executed. The "ACTION" property
MUST specify one and only one of these values.

In an AUDIO alarm, if the optional "ATTACH" property is included, it
MUST specify an audio sound resource. The intention is that the sound
will be played as the alarm effect. If an "ATTACH" property is
specified that does not refer to a sound resource, or if the
specified sound resource cannot be rendered (because its format is
unsupported, or because it cannot be retrieved), then the CUA or
other entity responsible for playing the sound may choose a fallback
action, such as playing a built-in default sound, or playing no sound
at all.

In a DISPLAY alarm, the intended alarm effect is for the text value
of the "DESCRIPTION" property to be displayed to the user.

In an EMAIL alarm, the intended alarm effect is for an email message
to be composed and delivered to all the addresses specified by the
"ATTENDEE" properties in the "VALARM" calendar component. The
"DESCRIPTION" property of the "VALARM" calendar component MUST be
used as the body text of the message, and the "SUMMARY" property MUST
be used as the subject text. Any "ATTACH" properties in the "VALARM"
calendar component SHOULD be sent as attachments to the message.

In a PROCEDURE alarm, the "ATTACH" property in the "VALARM" calendar
component MUST specify a procedure or program that is intended to be
invoked as the alarm effect. If the procedure or program is in a
format that cannot be rendered, then no procedure alarm will be
invoked. If the "DESCRIPTION" property is present, its value
specifies the argument string to be passed to the procedure or
program. "Calendar User Agents" that receive an iCalendar object with
this category of alarm, can disable or allow the "Calendar User" to
disable, or otherwise ignore this type of alarm. While a very useful
alarm capability, the PROCEDURE type of alarm SHOULD be treated by
the "Calendar User Agent" as a potential security risk.

Example: The following example is for a "VALARM" calendar component
that specifies an audio alarm that will sound at a precise time and
repeat 4 more times at 15 minute intervals:

  BEGIN:VALARM
  TRIGGER;VALUE=DATE-TIME:19970317T133000Z
  REPEAT:4
  DURATION:PT15M
  ACTION:AUDIO
  ATTACH;FMTTYPE=audio/basic:ftp://host.com/pub/sounds/bell-01.aud
  END:VALARM

The following example is for a "VALARM" calendar component that
specifies a display alarm that will trigger 30 minutes before the
scheduled start of the event or the due date/time of the to-do it is
```

```
associated with and will repeat 2 more times at 15 minute intervals:

  BEGIN:VALARM
  TRIGGER:-PT30M
  REPEAT:2
  DURATION:PT15M
  ACTION:DISPLAY
  DESCRIPTION:Breakfast meeting with executive\n
   team at 8:30 AM EST.
  END:VALARM

The following example is for a "VALARM" calendar component that
specifies an email alarm that will trigger 2 days before the
scheduled due date/time of a to-do it is associated with. It does not
repeat. The email has a subject, body and attachment link.

  BEGIN:VALARM
  TRIGGER:-P2D
  ACTION:EMAIL
  ATTENDEE:MAILTO:john_doe@host.com
  SUMMARY:*** REMINDER: SEND AGENDA FOR WEEKLY STAFF MEETING ***
  DESCRIPTION:A draft agenda needs to be sent out to the attendees
    to the weekly managers meeting (MGR-LIST). Attached is a
    pointer the document template for the agenda file.
  ATTACH;FMTTYPE=application/binary:http://host.com/templates/agen
   da.doc
  END:VALARM

The following example is for a "VALARM" calendar component that
specifies a procedural alarm that will trigger at a precise date/time
and will repeat 23 more times at one hour intervals. The alarm will
invoke a procedure file.

  BEGIN:VALARM
  TRIGGER;VALUE=DATE-TIME:19980101T050000Z
  REPEAT:23
  DURATION:PT1H
  ACTION:PROCEDURE
  ATTACH;FMTTYPE=application/binary:ftp://host.com/novo-
   procs/felizano.exe
  END:VALARM
```

Before describing the 'diabetes alert' widget and the associated Perl programs initiated by clicking on the various buttons, we first give a brief overview of the **Kalarm** system and its command structure, with illustrations linked to the diabetes alarm.

14.3 Kalarm

The Linux **Kalarm** utility is an established and versatile alarm tool which can be developed for use with the anaesthesia workstation. **Kalarm** is maintained by David Jarvie (`software@astrojar.org.uk`; `http://www.astrojar.ork.uk/linux/kalarm.html`). The latest version is 1.4.0 (April 2006). **Kalarm** can be accessed either using a 'form' via the mouse from the taskbar icon, or via the command-line, and has good documentation via a standard `kalarm --help` command.

Alarms can be both initiated and cancelled using commands issued via the commandline.

14.3.1 To show Kalarm icon

To generate the **Kalarm** icon just type

```
$ kalarm
```

at the command-line, and it will appear on the bottom-bar. The diabetes alarm depends on the **Kalarm** scheduling daemon running; this can be started using the [`--reset`] option, as follows (see also documentation section below).

```
$ kalarm  --reset
```

14.3.2 Documentation

Online help is available via the command `kalarm --help-all` as follows. Detailed information is also available from the Kalarm Handbook, which can be accessed via the alarm tray widget (click on 'help'), and also from `/usr/share/doc/HTML/en/kalarm/index.docbook`

```
version 0.8.3
Usage: kalarm [Qt-options] [KDE-options] [options] [message]

       kalarm
       kalarm [-bcilLrstu] -f URL
       kalarm [-bcilLrstu] message
       kalarm [-ilLrtu] -e commandline
       kalarm --tray | --reset | --stop
       kalarm --cancelEvent eventID [--calendarURL url]
       kalarm --triggerEvent eventID [--calendarURL url]
       kalarm --handleEvent eventID [--calendarURL url]
       kalarm [generic_options]

KDE personal alarm message and command scheduler

Generic options:
  --help                    Show help about options
  --help-qt                 Show Qt specific options
  --help-kde                Show KDE specific options
  --help-all                Show all options
  --author                  Show author information
  -v, --version             Show version information
  --license                 Show license information
  --                        End of options

Qt options:
  --display <displayname>   Use the X-server display 'displayname'.
  --session <sessionId>     Restore the application for the given 'sessionId'.
  --cmap                    Causes the application to install a private colour
                            map on an 8-bit display.
  --ncols <count>           Limits the number of colours allocated in the colour
                            cube on an 8-bit display, if the application is
                            using the QApplication::ManyColor colour
                            specification.
  --nograb                  tells Qt to never grab the mouse or the keyboard.
  --dograb                  running under a debugger can cause an implicit
                            -nograb, use -dograb to override.
  --sync                    switches to synchronous mode for debugging.
```

```
  --fn, --font <fontname>   defines the application font.
  --bg, --background <color> sets the default background colour and an
                            application palette (light and dark shades are
                            calculated).
  --fg, --foreground <color> sets the default foreground colour.
  --btn, --button <color>   sets the default button colour.
  --name <name>             sets the application name.
  --title <title>           sets the application title (caption).
  --visual TrueColor        forces the application to use a TrueColour visual on
                            an 8-bit display.
  --inputstyle <inputstyle> sets XIM (X Input Method) input style. Possible
                            values are onthespot, overthespot, offthespot and
                            root.
  --im <XIM server>         set XIM server.
  --noxim                   disable XIM.
  --reverse                 mirrors the whole layout of widgets.

KDE options:
  --caption <caption>       Use 'caption' as name in the titlebar.
  --icon <icon>             Use 'icon' as the application icon.
  --miniicon <icon>         Use 'icon' as the icon in the titlebar.
  --config <filename>       Use alternative configuration file.
  --dcopserver <server>     Use the DCOP Server specified by 'server'.
  --nocrashhandler          Disable crash handler, to get core dumps.
  --waitforwm               Waits for a WM_NET compatible windowmanager.
  --style <style>           sets the application GUI style.
  --geometry <geometry>     sets the client geometry of the main widget.
  --nofork                  Don't run in the background.

Options:
  -a, --ack-confirm         Prompt for confirmation when alarm is acknowledged
  -b, --beep                Beep when message is displayed
  -c, --color <color>       Message background colour (name or hex 0xRRGGBB)
  --calendarURL <url>       URL of calendar file
  --cancelEvent <eventID>   Cancel alarm with the specified event ID
  -e, --exec <commandline>  Execute a shell command line
  -f, --file <url>          File to display
  --handleEvent <eventID>   Trigger or cancel alarm with the specified event ID
  -i, --interval <period>   Interval between alarm recurrences
  -l, --late-cancel         Cancel alarm if it cannot be triggered on time
  -L, --login               Repeat alarm at every login
  -r, --repeat <count>      Number of times to repeat alarm (after the initial occasion)
  --reset                   Reset the alarm scheduling daemon
  -s, --sound <url>         Audio file to play
  --stop                    Stop the alarm scheduling daemon
  -t, --time <time>         Trigger alarm at time [[[yyyy-]mm-]dd-]hh:mm, or date yyyy-mm-dd
  --tray                    Display system tray icon
  -u, --until <time>        Repeat until time [[[yyyy-]mm-]dd-]hh:mm, or date yyyy-mm-dd
  --displayEvent <eventID>  Obsolete: use --triggerEvent instead
  --triggerEvent <eventID>  Trigger alarm with the specified event ID

Arguments:
  message                   Message text to display
```

14.3.3 Initiating a diabetes alarm

An example of the command-line (case sensitive) code for initiating a red alarm to prompt the user to repeat a blood-sugar measurement for a diabetic patient, with a pop-up window + beep repeating at 30 mins intervals is as follows (b=beep, c=colour, u=until-hh:mm, i=interval-mmmm).

In Mandrake-Linux the details of the alarm are written to the file /home/dick/.kde/share/apps/kalarm/calendar.ics. The default 'empty' file (ie with no alarms pending) is as follows.

```
BEGIN:VCALENDAR
PRODID:-//K Desktop Environment//NONSGML KAlarm 1.2.10//EN
VERSION:2.0
END:VCALENDAR
```

An example of the command-line (case sensitive) code for initiating a red alarm to prompt the user to repeat a blood-sugar measurement for a diabetic patient, with a pop-up window + beep repeating at 30 mins intervals is as follows (b=beep, c=colour, -t=trigger time yyyy-mm-dd-hh:mm, u=until-hh:mm, i=interval-mmmm).

```
kalarm  -b   -c red  -t 2008-04-10-11:51  -i 0005  -u 2008-04-11-11:31
"DIABETES --- repeat blood sugar"
```

This command generates a new calendar.ics file, which encodes the alarm data. Note that a given alarm instance (VEVENT) may be associated with several alarms (VALARM) in different formats (eg text, displayed file, voice etc) There is one VALARM for the display of message, and another VALARM for the sound of the beep. Note the empty lines following the END: commands.

```
1  BEGIN:VCALENDAR
2  PRODID:-//K Desktop Environment//NONSGML KAlarm 1.2.10//EN
3  VERSION:2.0
4     BEGIN:VEVENT
5     DTSTAMP:20080410T113102
6     ORGANIZER:MAILTO:
7     CREATED:20080410T113102
8     UID:KAlarm-1412322138.966
9     SEQUENCE:-1232236916
10    LAST-MODIFIED:20080410T113102
11    CLASS:PUBLIC
12    PRIORITY:5
13    RRULE:FREQ=MINUTELY;UNTIL=20080411T113100;INTERVAL=5
14    DTSTART:20080410T115100
15    TRANSP:TRANSPARENT
16        BEGIN:VALARM
17        DESCRIPTION: DIABETES --- repeat blood sugar
18        ACTION:DISPLAY
19        TRIGGER;VALUE=DURATION:PT0S
20        X-KDE-KALARM-FONTCOLOR:#ff0000\;#000000\;
21        END:VALARM
22
23        BEGIN:VALARM
24        ACTION:AUDIO
25        TRIGGER;VALUE=DURATION:PT0S
26        END:VALARM
27
28    END:VEVENT
29
30 END:VCALENDAR
```

14.3.4 Displaying a file

Note that the "alarm" can be the display of a file. For example, the following code will *immediately* (since no -t option) display a HTML file in a window. Note that there must be NO display "message" argument with this command since in this case the file has taken the place of the message.

```
kalarm -b  -c red  -f "/home/dick/...../file.html"
```

14.3.5 Current alarm status

The list and status of all current outstanding alarms are displayed in the **Kalarm** tray - tabular listing seen by clicking on the **Kalarm** icon on the bottom-bar. The code to place the icon in the bottom-bar tray is as follows.

```
$ kalarm  --tray
```

14.3.6 Cancelling an alarm

The **Kalarm** command (case sensitive) for cancelling an existing alarm having the UID `197659548.1073` as shown above is as follows, which has the effect of deleting the associated `VEVENT` environment from the `calendar.ics` file.

```
Kalarm  -cancelEvent  KAlarm-197659548.1073
```

Thus in order to delete an existing alarm 'event' it is necessary to parse the `calendar.ics` file and determine the `UID` associated with the particular alarm. Consequently, in order to facilitate identifying the correct `UID` for an alarm we simply arrange that (a) only a single alarm exists at any one time, and (b) we include a key word, say DIABETES, in the text message.

14.4 Alarm widget program (`dn-tkalarm.pl`)

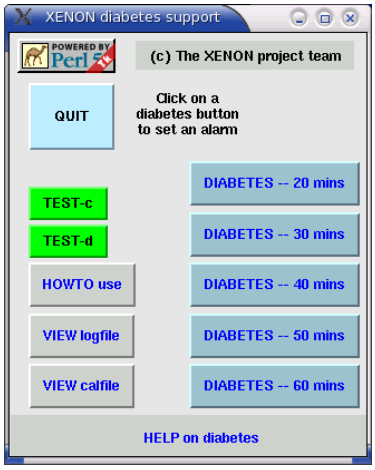


Figure 14.5:
View of the pop-up diabetes support widget. Clicking one of the blue 'diabetes' buttons (20–60 mins) sets an alert for the associated time interval. The three grey buttons are for displaying help information; the two green buttons are for generating test displays.

```
#!/usr/bin/perl
## dn-tkalarm.pl (modified from tklaunch2.pl)
## last modified April 24, 2006

my $thisprog = "[dn-tkalarm.pl]";  #to define this
    program-name in error messaages

## RWD Nickalls
## last change = Jan 22, 2006
## alarms for Xenon
## Useful books: page 301 Perl core languages (Little
    Black Book)
##-----------------------
## BOOK = Mastering Perl Tk (by: Lidie S and Walsh N
    (O'Reilly, 2002)
## to get FullScreen mode at startup (p 307)
##  -geometry widthXheight+Xoffset+Yoffset (NO
    spaces**page 409)
## $ perl tklaunch2.pl -geometry 1028x768 -0-0 ## page
    409
##  system ("perl ./tklaunch2.pl -geometry
    300x400-50-300") }
##-----------------------

use warnings;
use strict;
use Carp;
use Fatal;
use Tk;
use Tk::Help;
use Cwd;  # get this path

#--------macros----------
my $beep = "\a"; ##BEEP
my $OS_ERROR =""; ## used in viewcal SUB
my
    $kalarm_calendar_path="/home/dick/.kde/share/apps/kalarm/calendar.ics";
#-----------------

my $topwindow = MainWindow -> new();
#-------------------------------
$topwindow -> title("XENON diabetes support");
$topwindow -> Label(-text => "Click on a diabetes
    button to set an alarm",
                    -wraplength =>100,
                    -padx => 0.5,  #250
                    -height => 10 )
            -> pack();

## camel logo
if (-e  "./anim.gif"){
my $camelimage = $topwindow -> Photo(-file =>
    './anim.gif');
```

```
$topwindow -> Button(-relief => 'flat', -image =>
    $camelimage)
          -> place(-relx=>0.005, -rely=>0);
                    }
#--------------------
# QUIT button
$topwindow  -> Button (-text    => "QUIT",
                    -padx => 20, -pady => 20,
                    -relief => 'raised',
                    -background => 'LightBlue1',
                    -activebackground =>'LightBlue2',
                    -command => \&quit )
          -> place(-relx=>0.05, -rely=>0.115);
          #-> pack(-side =>'left', -expand => 1);

#-------------------------
# (c) XENON project team
$topwindow  -> Button (-text    => "(c) The XENON
    project team",
                   # -padx => 10, -pady => 10,
                    -relief => 'flat',
                    -background => 'LightGrey',
          -activebackground =>'LightGrey',
                    -foreground => 'Black',
         -activeforeground =>'Black',
                    )
                -> place(-relx=>0.35, -rely=>0.016);

#-----------------
# DIABETES 20mins button
$topwindow  -> Button (-text    => "DIABETES -- 20
    mins",
                    -padx => 10, -pady => 10,
                    -relief => 'raised',
                    -background => 'LightBlue3',
                        -activebackground
                            =>'LightBlue2',
       -foreground => 'Blue',
            -activeforeground =>'Red',

                    -command => \&diabetes20 )
                #  -> pack(-side =>'right', -expand =>
                    1);
        -> place(-relx=>0.5, -rely=>0.3);
#-------------
# DIABETES 30mins button
$topwindow  -> Button (-text    => "DIABETES -- 30
    mins",
                    -padx => 10, -pady => 10,
                    -relief => 'raised',
                    -background => 'LightBlue3',
                        -activebackground
                            =>'LightBlue2',
       -foreground => 'Blue',
```

```
            -activeforeground =>'Red',

                    -command => \&diabetes30 )
                # -> pack(-side =>'bottom', -expand
                    => 1);
          -> place(-relx=>0.5, -rely=>0.42);
#-------------
# DIABETES 40mins button
$topwindow  -> Button (-text     => "DIABETES -- 40
    mins",
                    -padx => 10, -pady => 10,
                    -relief => 'raised',
                    -background => 'LightBlue3',
                         -activebackground
                             =>'LightBlue2',
       -foreground => 'Blue',
             -activeforeground =>'Red',

                    -command => \&diabetes40 )
                # -> pack(-side =>'bottom', -expand
                    => 1);
          -> place(-relx=>0.5, -rely=>0.54);
#-------------
# DIABETES 50mins button
$topwindow  -> Button (-text     => "DIABETES -- 50
    mins",
                    -padx => 10, -pady => 10,
                    -relief => 'raised',
                    -background => 'LightBlue3',
                    -activebackground =>'LightBlue2',
        -foreground => 'Blue',
             -activeforeground =>'Red',

                    -command => \&diabetes50 )
                # -> pack(-side =>'bottom', -expand
                    => 1);
          -> place(-relx=>0.5, -rely=>0.66);
#-------------
# DIABETES 60mins button
$topwindow  -> Button (-text     => "DIABETES -- 60
    mins",
                    -padx => 10, -pady => 10,
                    -relief => 'raised',
                    -background => 'LightBlue3',
                         -activebackground
                             =>'LightBlue2',
             -foreground => 'Blue',
                      -activeforeground =>'Red',

                    -command => \&diabetes60 )
                # -> pack(-side =>'bottom', -expand
                    => 1);
          -> place(-relx=>0.5, -rely=>0.78);
#-------------
```

```
#-------------
# TEST-COFFEE  demo button
$topwindow  -> Button (-text     => "TEST-c",
                       -padx => 10, -pady => 5,
                       -relief => 'raised',
                       -background => 'Green',
                       -activebackground =>'Yellow',
                        -foreground => 'Black',
             -activeforeground =>'Red',
                       -command => \&testcoffee5 )
                    # -> pack(-side =>'bottom', -expand
                        => 1);
             -> place(-relx=>0.05, -rely=>0.36);

#-------------------------

#-------------
# TEST-diabetes  demo button
$topwindow  -> Button (-text     => "TEST-d",
                       -padx => 10, -pady => 5,
                       -relief => 'raised',
                       -background => 'Green',
                           -activebackground =>'Red',
       -foreground => 'Black',
             -activeforeground =>'Blue',
                       -command => \&testdiabetes )
                    # -> pack(-side =>'bottom', -expand
                        => 1);
             -> place(-relx=>0.05, -rely=>0.45);

#-------------------------
# HOWTO use  button
 $topwindow  -> Button (-text     => "HOWTO use",
                       -padx => 9, -pady => 10,
                       -relief => 'raised',
                       -background => 'LightGrey',
                            -activebackground =>'Grey',
                        -foreground => 'Blue',
             -activeforeground =>'Red',
                    #   -command => \&errorbox )
   #-command => \&showhelp )
-command => sub{showhelp()})
                    # -> pack(-side =>'bottom', -expand
                        => 1);
             -> place(-relx=>0.05, -rely=>0.54);


#-------------
# VIEW logfile button
$topwindow  -> Button (-text     => "VIEW logfile",
                       -padx => 10, -pady => 10,
                       -relief => 'raised',
                        -background => 'LightGrey',
```

```
                        -activebackground =>'Grey',
                        -foreground => 'Blue',
                         -activeforeground =>'Red',
                       -command => \&viewlog )
                   # -> pack(-side =>'bottom', -expand
                       => 1);
            -> place(-relx=>0.05, -rely=>0.66);

#-------------
# VIEW calendar file button
$topwindow  -> Button (-text    => "VIEW calfile",
                       -padx => 10, -pady => 10,
                       -relief => 'raised',
                        -background => 'LightGrey',
                        -activebackground =>'Grey',
                        -foreground => 'Blue',
                         -activeforeground =>'Red',
                       -command => \&viewcal )
                   # -> pack(-side =>'bottom', -expand
                       => 1);
            -> place(-relx=>0.05, -rely=>0.78);


#-----------------

## HELP button
$topwindow -> Button(-text => "HELP on diabetes",
                        -padx =>115, -pady =>10, -relief
                            => 'flat',
                                    -background =>
                                        'LightGrey',
                          -activebackground =>'Grey',
                                    -foreground =>
                                        'Blue',
                          -activeforeground =>'Red',
                                    -command => \&help )
              -> place(-relx=>0, -rely=>0.9);
#-------------------------


 my $diabetes_error_message = "...ERROR running
     dn-alarm-diabetes2 ".$thisprog;

MainLoop;

##===========SUBS========
#-------------
 sub quit {## clear the command-line terminal window
     and then exit
            system ("clear");
            exit();
           }
#----------------------
sub diabetes20 {
```

```
        ##      $topwindow -> bell;
        ##      $result = $dialog1 -> Show;
        ##     if ($result eq "OK") {};
         #    $topwindow ->destroy if
              Tk::Exists($topwindow);
             system ("perl ./dn-alarm-diabetes3.pl -t
                 20")
             and carp ($diabetes_error_message);
            # system ("perl ./dn-tkalarm.pl -geometry
                320x380-50-300");
             }
#-------------------
sub diabetes30 {
        ##      $topwindow -> bell;
        ##      $result = $dialog1 -> Show;
        ##     if ($result eq "OK") {};
        #     $topwindow ->destroy if
              Tk::Exists($topwindow);
             system ("perl ./dn-alarm-diabetes3.pl -t
                 30")
              and carp ($diabetes_error_message);
            # system ("perl ./dn-tkalarm.pl -geometry
                320x380-50-300");
             }
#-------------------
sub diabetes40 {
        ##      $topwindow -> bell;
        ##      $result = $dialog1 -> Show;
        ##     if ($result eq "OK") {};
      #        $topwindow ->destroy if
           Tk::Exists($topwindow);
             system ("perl ./dn-alarm-diabetes3.pl -t
                 40")
              and carp ($diabetes_error_message);
            # system ("perl ./dn-tkalarm.pl -geometry
                320x380-50-300");
             }
#-------------------
sub diabetes50 {
        ##      $topwindow -> bell;
        ##      $result = $dialog1 -> Show;
        ##     if ($result eq "OK") {};
      #        $topwindow ->destroy if
           Tk::Exists($topwindow);
             system ("perl ./dn-alarm-diabetes3.pl -t
                 50")
               and carp ($diabetes_error_message);
            # system ("perl ./dn-tkalarm.pl -geometry
                320x380-50-300");
             }
#-------------------
sub diabetes60 {
        ##      $topwindow -> bell;
        ##      $result = $dialog1 -> Show;
```

```
        ##     if ($result eq "OK") {};
      #       $topwindow->destroy if
          Tk::Exists($topwindow);
       system ("perl ./dn-alarm-diabetes3.pl -t 60")
            and carp ($diabetes_error_message);
          # system ("perl ./dn-tkalarm.pl -geometry
            320x380-50-300");
          } ## end of sub

#-------------------------
sub testcoffee5 {
       ## test use only 1 min test (-u 1 -i 1)
      ## as this will totally clear after 1 min
            system ("perl ./dn-alarm-coffee3.pl -u
              1");
          #  system ("perl ./dn-tkalarm.pl -geometry
            320x380-50-300");
           }


#-------------------------
sub testdiabetes {
       ## if use parameters (-u 1) only, then instant
          and no repeat!
       ## test use only 1 min test (-u 1 -i 1)
      ## as this will totally clear after 1 min

#       system ("perl ./dn-alarm-coffeeRED.pl -u 1");
  system ("perl ./dn-alarm-demoRED.pl");

# system ("kwrite ./anes-files/induction.txt -geometry
    350x380-600-300");
          ##  system ("perl ./dn-tkalarm.pl -geometry
            320x380-50-300");
           }

#-------------------------
sub errorbox {
         ## testing area
         ##      $topwindow -> bell;
         ##      $result = $dialog1 -> Show;
         ##     if ($result eq "OK") {};
             $topwindow->destroy if
                 Tk::Exists($topwindow);
                   print $beep;
           system (qq(perl ./dn-errorbox.pl --in
              "testing the message box")) ;
      ## now reinstate the Tk diabetes alarm widget
            system ("perl ./dn-tkalarm.pl -geometry
              320x380-50-300");
           }


#-----------------
```

```
 sub viewlog {
                $topwindow->destroy if
                    Tk::Exists($topwindow);
          if (-e "./dnalarm.log")
             {## use my dn-tkviewer.pl utility to view
                 the file
             system ("perl ./dn-tkviewer.pl --in
                 ./dnalarm.log") ;
             system ("perl ./dn-tkalarm.pl -geometry
                 320x380-50-300")  }
         else{ carp "....ERROR ....can't find file
             dnalarm.log [dn-tkalarm.pl]";
              system ("perl ./dn-tkalarm.pl -geometry
                  320x380-50-300");
                  };

    }  ## end of the sub

#-----------------
 sub viewcal {
                $topwindow->destroy if
                    Tk::Exists($topwindow);
    ##---------
    ## copy latest instance of the file
    ## this is a significant error if the copy fails
    my $thisdir=cwd;
    my $copy_string = "cp  ".$kalarm_calendar_path."
        ".$thisdir."/dn-calendar.ics";
    system $copy_string
           and carp "could not run $copy_string
               ($OS_ERROR)" ;
    #Perl-best-practice p 280
    #=====
    ## now view the copied file
    if (-e "./dn-calendar.ics")
        {## use my dn-tkviewer.pl utility to view the
            file
         system ("perl dn-tkviewer.pl --in
             ./dn-calendar.ics")
                and carp ("could not run Perl
                    dn-tkviewer.pl ".$thisprog."
                    ($OS_ERROR)") ;
         system ("perl ./dn-tkalarm.pl -geometry
             320x380-50-300")  }
      else{print "....ERROR:\n";
           print "....can't find file
               dn-calendar.ics>\n\n";
           system ("perl ./dn-tkalarm.pl -geometry
               320x380-50-300");
           };
    }  ## end of the sub


#-----------------------------------
```

```
 sub help {
#### this displays the main diabetes help file
             #$topwindow -> bell;
             ##  $result = $dialog2 -> Show;
    $topwindow ->destroy if Tk::Exists($topwindow);
      # if (-e "camteama5dvi.dvi")
    if (-e "../diabetes/diabetes_intro.html")
       {## first remove the Tk screen
       ## $topwindow ->destroy if
           Tk::Exists($topwindow);
       ## $topwindow-> bell; # beeps if click window (p
            296)
        # system("xdvi camteama5dvi.dvi -paper a5
           -geometry +20+20");
       #system("konqueror diabetes_intro.html");

  ## if use Simon's Konquered utility, then it needs
     the FULL path
         system("konquered -geometry 500x550+20+100
             /home/dick/allfiles/akalarm/diabetes
         /diabetes_intro.html");
         system ("perl ./dn-tkalarm.pl -geometry
             320x380-50-300");
           }
       else{print "....ERROR:\n";
            print "....can't find program
                <camteama5dvi.dvi>\n\n";
            system ("perl ./dn-tkalarm.pl -geometry
                320x380-50-300");
               };
      } # end of sub
#-------------------------------


sub showhelp {
   ## opens the small help window
        # create the array of help contents to pass to
            the help module
        my @helparray = (
    [{-title  => "\n    HOWTO use \n",
                                              -header =>
                                                  "",
                      # -text    => "This is a
                          description of my application
                          for the help."}],
             -text   => "Click on the headings."}],
#-------------------------
                                            [{-title  =>
                                                "Overview",
                                              -header =>
                                                  "\nThis
                                                  widget is
                                                   an aid
                                                  for use
```

```
                                                    when
                                                    anaesthetising
                                                     a
                                                    diabetic
                                                    patient.
          \n\nIt uses the well established Linux KDE
              Kalarm Open Source alarm utility
              (www.astrojar.org.uk/linux/\nkalarm.html/).
         \n\nOnce a diabetes alert is set, a red alert
             window (reminding you to take a blood
             sugar) will open after the set elapsed time.
        \n\nTest the diabetes alert by first clicking on
             the green TEST-d button, which will
           generate a demo red alert (simulating the red
              DIABETES alert). To trigger the TRUE
            diabetes alert system just click on one of
                the blue DIABETES buttons.
          \n\nIf you are too busy to do a blood-sugar
              when the red alert window appears, just
           close the window, and the alert will continue
              to recur at 5-min intervals until
            you set a new alert.",
      -text    => ""}],
#-------------------------
     [{-title  => "Setting an alert",
       -header => "\nSimply click on one of the blue
            'DIABETES' buttons. This will automatically
            set a new alert and delete any previous
            alert.\n\nThe new alert will appear after the
             specified time, and then recur every 5-mins
            until a new blue DIABETES alert is set---or
            until the existing alert is cancelled",
      -text    => ""}],
#-------------------------
      [{-title  => "Cancelling an alert",
        -header => "\nClick on the clock icon on the
             icon bar at the bottom of the sceeen
             (typically on the RHS). This will display
             all the current alarms (alerts).\n\nNow
             select the alarm to be cancelled (by right
             clicking on it), and then click on the
             'delete' button, and close the window.",
        -text   => ""}],

#-----------------------------
                                                [{-title  =>
                                                    "Testing",
                                                  -header =>
                                                      "\nClick
                                                      on the
                                                      green
                                                      TEST
                                                      butons:\n\nTEST-c\nThis
                                                       is
```

```
                                        generates
                                         a demo
                                        COFFEE-break
                                         reminder

                                        (yellow).\n\nTEST-d\nThis

                                        generates
                                         a RED
                                        coffee-break
                                         alert (+
                                         beep) to
                                         simulate
                                         the red
                                        DIABETES
                                         alert.",
                                    -text    =>
                                        ""}],
#-------------------------

                                   [{-title   =>
                                       "Author",
                                    -header =>
                                        "\nRWD
                                         Nickalls\nXenon
                                          project
                                         team\nDepartment
                                          of
                                         Anaesthesia,\nCity

                                         Hospital,\nNottingham,\nUK.\n\nemail:

                                         dicknickalls\@compuserve.com",
                                    -text    =>
                                        ""}],
#-------------------

                                   [{-title   =>
                                       "Version/date",
                                    -header =>
                                        "\nVersion
                                         1.1 --
                                         April 24,

                                         2006\nFixed
                                          red
                                         diabetes
                                         demo
                                         alert\n\nVersion

                                         1.0\nDecember
                                          18, 2005
                                          ",
                                    -text    =>
                                        ""}],
#-------------------
```

```
                                                [{-title  =>
                                                    "DIABETES
                                                    HELP",
                                                 -header =>
                                                    "",
                                                 -text   =>
                                                    "This is
                                                    only a
                                                    mini-help.\n--for
                                                     a
                                                    detailed
                                                    help page
                                                     click on
                                                     the
                                                    'HELP on
                                                    diabetes'
                                                     button
                                                    at the
                                                    bottom of
                                                     the
                                                    parent
                                                    widget."},
#-------------------------
                                                {-title  =>
                                                    "Sliding
                                                    scale",
                                                 -header =>
"\n     ITU Sliding Scale\n\n-- Run 5\%Dextrose at 60
    mls/hr\n-- Run insulin actrapid 1Unit/ml  at 0-5
    Units/hr)\n\nGlucose    Insulin rate\nmMol/L
    units/hr\n\n 0 -  3.9      0\n 4 -  6.9      1\n 7 -
    9.9      2\n10 - 14.9      3\n15 - 19.9      4\n
    20+       5",
                                                 -text   =>
                                                    ""}]);


##======
        # create the help object ? needs to go last ?
        my $help = $topwindow->Help(-title        =>
            "XENON diabetes help",
#change parameter values here (see file /Help.pm
    -listfontsize => '14',
    -detailsheaderfontsize =>'14',
    -detailsfontsize => '14',
    -height => '20',                # screen height =50
    -listwidth =>'20',
    -detailswidth => '30',
   # -font =>'Times', # does not work
   # -weight => 'normal', # does not work
#-------------
                                                        -variable

                                                            =>
```

```
                                                                \@helparray);
}
##-----------end--------------------
```

14.5 Test demo programs (`dn-alarm-demoRED.pl`)

There are two test buttons which trigger demo programs; these show a yellow (`dn-alarm-demoYELLOW.pl`) and a red (`dn-alarm-demoRED.pl`) demo alert. The following is the 'red' demo program.

```
#!/usr/bin/perl
## dn-alarm-demoRED.pl

# RWDN Thurs 24April2006
## to look like a diabetes alarm
## main difference is that the trigger option is NOT
    used here

use warnings;
use strict;
use Carp;  # allows croak ""
use Fatal qw(open close);  # for errors
##use Perl6::Builtins qw( system );
#use version;
#use Cwd;  ##to get this path
##-------------------

my
    $kalarm_calendar_path="/home/dick/.kde/share/apps/kalarm/calendar.ics";
my $OS_ERROR="";
##========================
# create a printer-log file
  open  my $logg, ">", "dnalarm.log"||die "ERROR: can't
       open dnalarm.log file\n";
  print {$logg} "--- TEST button pressed ---\n";
  print "--- TEST button pressed ---\n";
#=========================================================

## grab current time
my $time_now_unix=time(); ## seconds
my $time_now_string=localtime($time_now_unix);
  print {$logg} "dn-alarm.log,  ",$time_now_string,":
      Unix=",$time_now_unix,"\n";
  print {$logg} "log of Perl dn-alarm-demoRED.pl
      program \n";

##-----------------------------
## for NO recurrence
## we need NO trigger time AND ( the -u (until) delay
    must be LESS than the -i delay)
## so we make -u = (NOWtime + 2mins), and set the -i
    time to 5mins
```

```
my $until_unix= $time_now_unix+ 120; ## = 2mins in secs
my $until_string=localtime($until_unix);
my $until_ymdhm=ymdhm($until_string);
     print "until time = ",$until_ymdhm, " (= +2
         mins)\n";
     print "interval time =  5 mins\n";
     print {$logg} "until time = ",$until_ymdhm, " (= +2
          mins)\n";
     print {$logg} "interval time =  5 mins\n";
#  format is  $until="-u  2005-12-13-15:36  " (include
    terminal spaces)
my $until = "-u ".$until_ymdhm."  "; ## the period
    during which it repeats
##-------------------------------
## set a new alarm
## need -i to be > time to -u
my $message=qq(" time for a COFFEE-break.. Ahh....");
my $out= "kalarm -b  -c red -i 0005 $until $message";
     print "setting new RED COFFEE alarm\n";
     print  "sending Kalarm string = ", $out, "\n";
     print {$logg}  "setting new COFFEE alarm\n";
     print {$logg}  "sending Kalarm string = ", $out,
        "\n";
system(qq($out))
      and croak "could not run $out ($OS_ERROR)" ;
      #Perl-best-practice p 280

####===============SUBS=============================================
## ymdhm($time-string]);
## need to determine the until time in the correct
    format for kalarm

  sub ymdhm {
       ## format =  yyyy-mm-dd-hh:hh
       # passing only one time_string into array
       my ($time_string) = @_;
       ##print "---processing parameter [$time_string]
           \n";

      ## now get the until-time as yyy-mm-dd-hh:mm from
           the time_string
      ## routine modified from fields2PDATA.pl
            #-------------------------------------------
    ## note the main items are <space> separated except
        hh:mm:ss
    ## format is:    Sun Jan 25 13:24:35 2004
    ## format is:    Sun Jan  5 13:24:35 2004
    ## note **** get /two/ spaces after the Month if
        days <10
    ## see SUB tedname() in launchcam12.pl
    ##-----------------------------------------------
             # if /two/ spaces in posn 8 and 9 then
                 remove /one space/
             if (substr($time_string,7,2) eq "  ")
```

```
                {substr($time_string,7,2," ")};
              ## replace spaces with commas
              $time_string =~ tr/ /,/;
              ## make an array
             my @stgmt=split (/[,]/, $time_string);
              ##  $day=$stgmt[0];  ## not used here
             my $month=$stgmt[1];
             my $date=$stgmt[2];
             my $st=$stgmt[3];
             my $year=$stgmt[4];
              ## $noitems=$#stgmt+1;   ## not used here
                    ## now split the time hh:mm:ss -->
                        hh:mm only
            my @sthhmmss=split (/[:]/, $st);
            my $hh=$sthhmmss[0];
            my $mm=$sthhmmss[1];
             ## $ss=$sthhmmss[2];  ## not used here
             # print "the gmt part is:
                 $day,$month,$date,$st,$year\n";
             # print {$logg} "the gmt part is:
                 $day,$month,$date,$st,$year\n";
             ##---------
             ## but Kalarm requires that both month and
                 date are in numerals
             if ($month eq "Jan"){$month="01"}
             if ($month eq "Feb"){$month="02"}
             if ($month eq "Mar"){$month="03"}
             if ($month eq "Apr"){$month="04"}
             if ($month eq "May"){$month="05"}
             if ($month eq "Jun"){$month="06"}
             if ($month eq "Jul"){$month="07"}
             if ($month eq "Aug"){$month="08"}
             if ($month eq "Sep"){$month="09"}
             if ($month eq "Oct"){$month="10"}
             if ($month eq "Nov"){$month="11"}
             if ($month eq "Dec"){$month="12"}
          my
              $ymdhm=$year."-".$month."-".$date."-".$hh.":".$mm;
           return $ymdhm;
       }#end of sub
##-------------------------------
 close
 __END__
##---------------end of prog-------$
```

14.6 Diabetes alarm program (dn-alarm-diabetes3.pl)

```
#!/usr/bin/perl
```

```
# RWDN Thurs 16Dec2005
# d—demo-alarm-diabetes2.pl

use warnings;
use strict;
use Carp;  # allows croak ""
use Fatal qw(open close);  # for errors
##use Perl6::Builtins qw( system );
use Getopt::Long;  ## for commandline stuff
#use version;
use Cwd;  # grab this dir

## DN-alarm-diabetes2.pl (modified from
    dn-alarm-DIABETES1.pl)
## runs Kalarm
##==========initialising===============
my
    $kalarm_calendar_path="/home/dick/.kde/share/apps/kalarm/calendar.ics";
my $OS_ERROR="";
##====================

# create a printer-log file
  open  my $logg, ">", "dnalarm.log"||die "ERROR: can't
       open dnalarm.log file\n";
## grab current time
my $time_now_unix=time(); ## seconds
my $time_now_string=localtime($time_now_unix);
  print {$logg} "dnalarm.log,  ",$time_now_string,":
      Unix=",$time_now_unix,"\n";
  print {$logg} "log of my Perl dnalarm3.pl  program
      \n";
#=======================================================

## copy the Kalarm calendar file to this dir with new
    name
if (-e  $kalarm_calendar_path) {
    print {$logg} "copying the calendar.ics file -->
        dn-calendar.ics \n";
 ## grab the current directtory pathname
  my $thisdir=cwd;
  my $copy_string = "cp  ".$kalarm_calendar_path."
      ".$thisdir."/dn-calendar.ics";
    system $copy_string
           and croak "could not run $copy_string
               ($OS_ERROR)" ;
            #Perl-best-practice p 280
          }
   else{ print "ERROR: cannot copy the cal file\n"};
##========read the calendar file====================

## set the eventFLAG
my $eventnumber=0;  # counter to count the number of
    DIABETES events
my $eventFLAG="OFF";
```

```
open my $calfile, "<", "dn-calendar.ics" || die "ERROR:
    can't open file  dn-calendar.ics \n";

## now read each line in the file, and place parameters
     into an array
      print "...reading the CAL file line-by-line\n";
      print {$logg} "...reading the CAL file
          line-by-line\n";

 # reset these  variables to zero  BEFORE starting the
     WHILE loop
my $uid1 = 0;
my $uid2 = 0;
my $uid = "";
my $text1 = 0;
my $text2 = 0;
my $text = "";
my $dataline="";
my $event="";

           #-----------------
        LINE: while (<$calfile>){
          next LINE if /^#/;  #skip # comments
          next LINE if /^%/;  #skip % comments
         next LINE if /^$/;  #skip blank lines
          # grab the whole line as a string
          $dataline = $_;
          chomp($dataline); # removes the line-ending
#---------------
# reset variables to zero
$uid1 = 0;
$uid2 = 0;
$uid = "";
$text1 = 0;
$text2 = 0;
$text = "";
#---------------
        #### @value=split (/[,]/, $dataline);
        # print $dataline;
        ## replace CR/LF/space/  with visible chars =
            newbuffer
             #    $dataline=~ s/\r/<CR>/;
             #    $dataline=~ s/\n/<LF>/;
             #    $dataline=~ s/ /<SPACE>/;
             #    print $dataline, "\n";;
   if ($dataline=~m/BEGIN:VEVENT/) {$eventFLAG="ON",
       print "FLAG=ON\n";
                                          $event="";
                                          $event=$event.$dataline;
                                        # next LINE;
                                          };
   if ($eventFLAG eq "ON") {$event=$event.$dataline;
                              ## print
                                  "event=",$event,"\n";
```

```
                        }
if ($dataline=~m/END:VEVENT/) {
            $eventFLAG="OFF", print "FLAG=OFF\n";
            ## now analyse the event string to find
                UID and TEXT
        print "NEW event found--checking for word
            DIABETES\n";
            if ($event=~m/DIABETES/i){
                ## increment event counter
                $eventnumber=$eventnumber + 1;
    ##**   $DIABETES_event=$DIABETES_event.$event;

                # get UID
                print "DIABETES event found\n";
                #print "event = ", $event, "\n";
                ## process the event string to get
                    UID and TEXT
                ## get the index positions for UID
                    and SEQUENCE
                $uid1 = index $event, 'UID
                    :KAlarm-';
                $uid2 = index $event, 'SEQUENCE';
                print "uid1 = ",$uid1, "\n";
                print "uid2 = ",$uid2, "\n";
                $uid = substr($event, ($uid1+5),
                    ($uid2-($uid1+5)));
                print "UID = ", $uid, "\n";
                #---
                ## get the index positions for
                    TEXT and ACTION
                $text1 = index $event, 'TEXT';
                $text2 = index $event, 'ACTION';
                print "text1 = ",$text1, "\n";
                print "text2 = ",$text2, "\n";
                $text = substr($event, ($text1+5),
                    ($text2-($text1+5)));
                print "TEXT = ", $text, "\n";
                ##----
                ## cancel the event
                my $cancel= "kalarm -cancelEvent
                    ".$uid;
                print "cancelling existing
                    DIABETES alarm\n";
                print "sending command:
                    ",$cancel,"\n";
                print {$logg} "cancelling existing
                    DIABETES alarm\n";
                print {$logg} "sending command:
                    ",$cancel,"\n";
                ## if more than one DIABETES event
                    to cancel, then need to
                ## pause slightly as it takes time
                    for each cancel to take effect
                if ($eventnumber>1) {sleep 2};
```

```
                    system(qq($cancel))
                        and croak "could not run
                            $cancel ($OS_ERROR)" ;
                        #Perl-best-practice p 280


                    ##--------now look at next
                        event---------------

                    print "------\n";
                    $event="";   ## clear the event
                        string
                    print "looking for the next
                        event\n";
                    next LINE;
                     }  # end of if contains word
                         DIABETES conditional
                    else{##print "NEW event
                        found--checking for word
                        DIABETES\n";
                          print "NO DIABETES word in
                              this event, so looking for
                               next event\n";
                      #    print "event = ", $event,
                           "\n";
                          next LINE};
                ##-------------
                ## finally dump the event string and
                    start again
                };

# print "***", $dataline,"\n";
$dataline="";
} ## end of the input loop reading the {$calfile}


##-------------------------------------

 print "no more  events found - termating now\n";
print "-------------\n";
# print "event string = ", $event,"\n";

##==========================

## get the commandline options ( using Getopt::Long)
##   Perl-best-practice p 309
my $trigger_time_mins      = 30; # mins
my $repeat_interval_mins   = 5; # mins
my $until_time_mins        = 1440; # mins = 24hrs
#my $message    = qq("DIABETES:");

my $options_okay = GetOptions(
'trigger=i'  => \$trigger_time_mins,     #--trigger
    expects an integer mins
```

```
'interval=i' => \$repeat_interval_mins, # --interval
    mins
'until=i'    => \$until_time_mins,      # --until
    mins = 1440 =24hrs
#'message=s'  => \$message,              # --message
);

#-----------------
## use 2 trailing spaces (to separate items)
my $kalarm="kalarm ";
my $bell="-b  "; ## -b
my $color="-c red  ";
##$trigger_time_mins=;  ## starttime
#-----------------
##$repeat_interval_mins=5; # mins
my $intervala="0000".$repeat_interval_mins;
my $intervalb=substr($intervala,-4);
    print {$logg}  "interval= ", $intervalb,"\n";
my $repeat_interval="-i ".$intervalb."  ";
##-----------------
my $message=qq(" DIABETES --- repeat blood sugar ");

   print {$logg} "bell = ", $bell, "\n";
   print {$logg} "color = ", $color, "\n";
   print {$logg} "trigger mins = ", $trigger_time_mins,
        "\n";
   print {$logg} "interval mins = ",
       $repeat_interval_mins, "\n";
   print {$logg} "until mins = ", $until_time_mins,
       "\n";

##-------------------------------
## determine the new `trigger' time
## determine final time (= trigger-time)
my
    $trigger_unix=$time_now_unix+($trigger_time_mins*60);
     ## secs
## get local time string
my $trigger_string=localtime($trigger_unix);
## get ymdhm of  trigger_time
my $trigger_ymdhm= ymdhm($trigger_string);  ## use the
    subroutine
#   print "trigger time  hh:mm = ", $trigger_hhmm,
    "\n";
   print {$logg} "trigger time = ", $trigger_ymdhm,
       "\n";
## write the correct trigger string for the Kalarm
    commandline
my $trigger="-t ".$trigger_ymdhm."  "; ## two trailing
    spaces
##-------------------------------
## determine the correct until_time  (add 24hrs)
my $until_unix= $time_now_unix+($until_time_mins *60);
    ## secs
```

```
my $until_string=localtime($until_unix);
my $until_ymdhm=ymdhm($until_string);
    #print "until time = ",$until_ymdhm, "\n";
    print {$logg} "until time = ",$until_ymdhm, "\n";
my $until="-u ".$until_ymdhm." "; ## the period during
    which it repeats
#  format is  $until="-u  2005-12-13-15:36  ";
##-------------------------

## testing with file - use the KDE geometry option to
    get width correct
##$file = " -f
    /home/dick/allfiles/akalarm/perl/help.txt  ";
#$out= $kalarm.$color.$until.$repeat_interval.$file;
#-----------------

## set a new DIABETES alarm
#$out=
    $kalarm.$color.$trigger.$repeat_interval.$until.$message;
my $out=
    $kalarm.$bell.$color.$trigger.$repeat_interval.$until.$message;
    print "setting new DIABETES alarm\n";
    print  "sending Kalarm string = ", $out, "\n";
    print {$logg}  "setting new DIABETES alarm\n";
    print {$logg}  "sending Kalarm string = ", $out,
        "\n";
system(qq($out))
      and croak "could not run $out ($OS_ERROR)" ;
      #Perl-best-practice p 280


####==============SUBS=============================================
      ## ymdhm($time-string]);

  sub ymdhm {
       ## format =  yyyy-mm-dd-hh:hh
       # passing only one time_string into array
       my ($time_string) = @_;
       ##print "---processing parameter [$time_string]
           \n";

      ## now get the until-time as yyy-mm-dd-hh:mm from
          the time_string
      ## routine modified from fields2PDATA.pl
           #-------------------------------------------
    ## note the main items are <space> separated except
        hh:mm:ss
    ## format is:    Sun Jan 25 13:24:35 2004
    ## format is:    Sun Jan  5 13:24:35 2004
    ## note **** get /two/ spaces after the Month if
        days <10
    ## see SUB tedname() in launchcam12.pl
    ##-----------------------------------------------
```

```
            # if /two/ spaces in posn 8 and 9 then
                remove /one space/
            if (substr($time_string,7,2) eq "  ")
                {substr($time_string,7,2," ")};
            ## replace spaces with commas
            $time_string =~ tr/ /,/;
            ## make an array
            my @stgmt=split (/[,]/, $time_string);
            ##  $day=$stgmt[0];     ## not used
            my $month=$stgmt[1];
            my $date=$stgmt[2];
            my $st=$stgmt[3];
            my $year=$stgmt[4];
            ## $noitems=$#stgmt+1;    ## not used
                    ## now split the time hh:mm:ss -->
                        hh:mm only
          my @sthhmmss=split (/[:]/, $st);
          my $hh=$sthhmmss[0];
          my $mm=$sthhmmss[1];
          ##    $ss=$sthhmmss[2];    ## not used
          # print "the gmt part is:
              $day,$month,$date,$st,$year\n";
          # print {$logg} "the gmt part is:
              $day,$month,$date,$st,$year\n";
          ##---------
          ## but Kalarm requires that the month and
              date is in numerals
          if ($month eq "Jan"){$month="01"}
          if ($month eq "Feb"){$month="02"}
          if ($month eq "Mar"){$month="03"}
          if ($month eq "Apr"){$month="04"}
          if ($month eq "May"){$month="05"}
          if ($month eq "Jun"){$month="06"}
          if ($month eq "Jul"){$month="07"}
          if ($month eq "Aug"){$month="08"}
          if ($month eq "Sep"){$month="09"}
          if ($month eq "Oct"){$month="10"}
          if ($month eq "Nov"){$month="11"}
          if ($month eq "Dec"){$month="12"}
          my
              $ymdhm=$year."-".$month."-".$date."-".$hh.":".$mm;
          return $ymdhm;
      }#end of sub

##==================
 close
 __END__
```

14.7 File viewer program (`dn-tkviewer.pl`)

```
#!/usr/bin/perl
```

```
## RN-tkviewer.pl (modified from RNtkview.pl)
my $thisprog="[dn-tkviewer.pl]" ;  ## used in error
    messages
##
## RWD Nickalls
## Dec 16, 2005
## a simple TK fileviewer (takes filename as argument)
##
#--------now make the widget---------------
     ##
     ## BOOK = Mastering Perl Tk (by: Lidie S and Walsh
          N (O'Reilly, 2002)
     ## to get FullScreen mode at startup (p 307)
     ##  -geometry widthXheight+Xoffset+Yoffset (NO
         spaces**page 409)
     ## $ perl tklaunch2.pl -geometry 1028x768 -0-0 ##
         page 409
     ##  system ("perl ./tklaunch2.pl -geometry
         300x400-50-300") }
     ## see p 233 PerlTK book
     ## see TEXT widget p 162
     ##
##-----------------------
use warnings;
use strict;
use Tk;
use Carp;
use Fatal;  ## to give good failure error messages
use Getopt::Long;   ## for command-line (see  my prog
    ... diabetes2.pl)
#-----------------------
## get the commandline options ( using Getopt::Long)
##   Perl-best-practice p 309
## to allow an Input filename to view
my $input_filename      = '-';
my $options_okay = GetOptions(
    'in=s'        => \$input_filename, #  --in option
        expects a string
  );
## usage =  $ perl dn-tkviewer.pl --in filename
##
if ($input_filename eq '-'){croak "...ERROR -- filename
    not specified ".$thisprog};
##
## define an error message for use later
my $errormessage="...ERROR -- can't find filename
    <".$input_filename."> ".$thisprog;
## note that this errror messahe must be outside the
    if(){} statement
##
if (-e    $input_filename){
     #--------now make the widget---------------
     my $topwindow= MainWindow -> new();
```

```
      $topwindow -> title("XENON  file:
          ".$input_filename);
      my $text = $topwindow->Scrolled("Text",
                 # -background => 'LightGrey',
                 # default background colour is a very
                     pale grey
                  -font => ['courier', '14'],
                           )
                    ->pack();
      open my $VIEWFILE, "<", $input_filename|| croak
          $errormessage, " [code A]" ;
      while (<$VIEWFILE>){$text->insert('end',$_)};
      MainLoop;
      close($VIEWFILE);
      }
      else{croak  $errormessage, " [code B]"};
##-------end-----------
```

14.8 Error message widget program (dn-errorbox.pl)

```
#!/usr/bin/perl -w
## RN-errorbox.pl (modified from rntkalarm.pl)
my $thisprog = "[rn-errorbox.pl]";  #to define this
    program-name in error messaages

##---------------------
## RWD Nickalls
## April 26, 2006.
## message boxes for Xenon
## Useful books: page 301 Perl core languages (Little
    Black Book)
##---------------------
## usage:   $ perl dn-errorbox.pl -in "error message
    is...."
## requires use of the explicit --in tag
##---------------------
## BOOK = Mastering Perl Tk (by: Lidie S and Walsh N
    (O'Reilly, 2002)
## to get FullScreen mode at startup (p 307)
##  -geometry widthXheight+Xoffset+Yoffset (NO
    spaces**page 409)
## $ perl tklaunch2.pl -geometry 1028x768 -0-0 ## page
    409
##  system ("perl ./tklaunch2.pl -geometry
    300x400-50-300") }
##---------------------

use Tk;
use Carp;
```

```
use Fatal;
use Getopt::Long;   ## gets options from command-line
    (see my prog ...diabetes2.pl)

#-----------------------
## get the commandline options ( using Getopt::Long)
##   Perl-best-practice p 309
## to allow an Input filename to view
my $message     = '-';
my $options_okay = GetOptions(
'in=s'       => \$message, #  --in option expects a
    string
);
## usage =  $ perl rn-tkviewer.pl --in filename
##
if ($message eq '-'){croak "...ERROR -- message not
    specified ".$thisprog," $!"};
##
#-----------------------------------------------
## write the word ERROR underlined
my $error="ERROR
    MESSAGE\n-----------------------------------\n\n\n";
my $boxmessage = $error.$message;
##
#-----------------------
$topwindow = MainWindow -> new();
$topwindow -> title("XENON");
$topwindow -> Label(-text => $boxmessage,
                    -wraplength =>200,
                    -padx => 10,
     -background => 'Yellow',
                    -foreground => 'Black',
                    -height => 10,
                    -width => 35 )
          -> place(-anchor => 'n')
                          -> pack();
                   #   ->pack(-side => 'top'); #,-expand
                         =>1);
MainLoop;
##----------end---------$
```