НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ – София, ГОДИШНИК "Телекомуникации" 2017, том 4

New Bulgarian University – Sofia, Bulgaria, Yearbook Telecommunications 2017, vol.4

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

ПРИЛОЖЕНИЕ ЗА EMAIL СТАТИСТИКА В СИСТЕМИТЕ ЗА СБОР ДАННИ, С ГЕНЕРИРАНЕ НА ТАБЛИЦИ И ГРАФИКИ

Велизар Симеонов

Резюме: Системата представлява съвкупност от програмни редове, които изпращат генериран предварително файл с резултати извлечени чрез уред за измерване, оформят информацията в таблица и я изпращат на предварително зададена електронна поща. Допълнително към това е добавен и програмен код за улеснена инсталация на системата.
Ключови думи: генериране, графики, информация, таблици, система.

1. ВЪВЕДЕНИЕ

Целта е да бъде автоматизиран процеса по генериране на удобни за разчитане таблици и графики, които да бъдат изпращани на предварително установена електронна поща. Като цяло това ще подобри производителността на техническия екип, като спести време на инженерите и ще им предостави необходимите данни в удобен за разчитане вид.

Допълнително е добавен и удобен начин за въвеждане на системата, като се използва допълнителен модул. Това от своя страна ще оптимизира процеса по въвеждане на системата в хардуерно устройство, което преди не е разполагало с такава.

2. НЕОБХОДИМИ ПРИЛОЖЕНИЯ

За реализацията е нужно да се инсталират допълнителни модули, които да направят възможно функционирането на прокрамните скриптове.

Инсталиране на Perl модулите:

"Spreadsheet::WriteExcel"
"Excel::Template"
"Text::CSV_XS",
YAML:

```
> cpan YAML
> perl -MCPAN -e 'install "Spreadsheet::WriteExcel"'
> perl -MCPAN -e 'install "Excel::Template"'
> perl -MCPAN -e 'install "Text::CSV_XS"'
```

Ако по време на инсталацията има въпрос за "host proxy", въвежда се:
http://www.cpan.org

- Инсталиране на mutt и sendmail - за изпращане на генерираните таблици

```
> apt-get install mutt
>apt-get install sendmail
```

- Конфигуриране на sendmail

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

>nano /etc/mail/sendmail.mc

В последните редове въвеждаме (домейнът може да е друг):

define(`MAIL_HUB', `nbu.bg.')dnl
define(`LOCAL_RELAY', `nbu.bg.')dnl

>sendmailconfig
> service sendmail restart

- Трябва да се сложи име на машината, от която ще се изпращат таблиците. По подразбиране е localhost, което обикновено прави проблем.

> hostname test.nbu.bg

3. ПРОГРАМЕН КОД ЗА ГЕНЕРИРАНЕ НА ТАБЛИЦА И ГРАФИКА

Приемаме по подразбиране, че файлът с извлечените стойности се намира в директорията /var/log, името му е report.log, както и за разделител се използва [,].

<code>

```
#!/usr/bin/perl
(Заб.: какви модули ще се използват по време на изпълнението на скрипта)
use strict;
use warnings;
use Spreadsheet::WriteExcel;
use Text::CSV_XS;
(Заб.: тук указваме името на генерирания файл, който съдържа графиката и таблицата )
my $name = 'report.xls';
(Заб.: как се казва файла с логовете)
# Log files we are reading
my $file_report = "/var/log/report.log";
(Заб.: много е важно да поддържаме динамично броя резултати в лог. файла, т.е. да не се налага да вкарваме броя редове допълнително. Тук проверяваме колко реда е лог. файла)
#check how many lines we have in log file
my $lines = `wc -l < $file_report`;
(Заб.: генерираме таблицата)
# Excel file we generate. $name is set above
my $workbook = Spreadsheet::WriteExcel->new("/var/log/$name");
(Заб.: как ще се казва страницата в Excel)
# worksheet
my $report = $workbook->addworksheet("Reports");
(Заб.: как ще се казва всяка колона)
my @array_report = ( 'Hour','Value 1','Value 2' );
(Заб.: в случай, че бъдат необходими по-натам, указваме шрифтове и големина на буквите, както и цвят)
my %cf = (
  font => 'Arial',
  border => 1,
  border_color => 'black'
  );

my %bold = (
  bold => 1
  );
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
my %ch = (
  align => 'center',
  valign => 'center',
  font => 'Arial',
  size => 11,
  border => 1,
  border_color => 'black',
  bg_color => 'yellow'
 );

my $ch = $workbook->addformat(%ch, %bold); # header
my $cn = $workbook->addformat(%cf, size => 10); # normal text
my $cs = $workbook->addformat(%cf, size => 8); # small text
```

(Заб.: в случай на нужда, добавяме филтър за всяка колона)

```
$report->autofilter('A1:C1');
```

(Заб.: тук посочваме стилът и дължината на всяка колона, т.е първата е 15 символа и тн)

```
set_columns($report, "A",
  [15, 16, 16],
  [$cn, $cn, $cn]);
```

(Заб.: Прочитаме от лог файла и пишем в таблицата)

```
# this is where we read the data from log files and write to the xls
$report->write('A1', [ @array_report ], $ch);
write_from_log($report, $file_report);

$report->activate(); # I'd like the first worksheet to be active
```

(Заб.: Генерираме графиката и я добавяме към настоящата страница)

(Заб.: Разполагаме с няколко вида графики, които можем да указваме чрез „type”, а именно:

- area: Creates an Area (filled line) style chart. See Spreadsheet::WriteExcel::Chart::Area.
- bar: Creates a Bar style (transposed histogram) chart. See Spreadsheet::WriteExcel::Chart::Bar.
- column: Creates a column style (histogram) chart. See Spreadsheet::WriteExcel::Chart::Column.
- line: Creates a Line style chart. See Spreadsheet::WriteExcel::Chart::Line.
- pie: Creates an Pie style chart. See Spreadsheet::WriteExcel::Chart::Pie.
- scatter: Creates an Scatter style chart. See Spreadsheet::WriteExcel::Chart::Scatter.
- stock: Creates an Stock style chart. See Spreadsheet::WriteExcel::Chart::Stock.

Заб.: т.е ако искаме да използваме таблица от вид „line”, ще я въведем така:

```
my $chart = $workbook->add_chart( type => 'line', embedded => 1 );

# Chart generate
my $chart = $workbook->add_chart( type => 'column', embedded => 1 );
```

(Заб.: в графиката ще използваме 2 серии, защото имаме две стойности)

```
# Configure the first series. (Sample 1)
$chart->add_series(
      name       => 'Values 1',
      categories => "=Reports!\$A\$2:\$A\$$lines",
      values     => "=Reports!\$B\$2:\$B\$$lines"
   );
# Configure the second series. (Sample 2)
$chart->add_series(
      name       => 'Values 2',
      categories => "=Reports!\$A\$2:\$A\$$lines",
      values     => "=Reports!\$C\$2:\$C\$$lines"
   );
```

(Заб.: всяка серия има свое име, както и свой уникален източник на информация,т.е Серия 1 има за източник на информация values – ‘=Страница Reports! Колона B$ позиция 2 до (:) Колона B$ позиция $lines, която стойност се извлича по-горе в програмния ред).

(Заб.: пишем имена в графиката)

```
# Add a chart title and some axis labels.
$chart->set_title( name => 'Daily results report' );
$chart->set_x_axis( name => 'Time' );
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
        $chart->set_y_axis( name => 'Values' );
(Заб.: къде ще се намира графиката, от коя колона се разполага и на какво разтояние)
        # Insert the chart into the worksheet (with an offset).
        $report->insert_chart( 'E2', $chart, 50, 10 );
(Заб.: затваряме таблицата)
        $workbook->close();  # play nice
        exit(0);

        # Setup the print options
        # usage:

        # print_setup($worksheet);

(Заб.: ако ще отпечатваме)
        print_setup($report);

        # $worksheet is a reference to a worksheet object
        #
        # These are the print options

        sub print_setup {
          my $report = shift;
          $report->set_landscape();
          $report->center_horizontally();
          $report->set_margins_LR(0.25);
          $report->set_margin_top(0.5);
          $report->set_margin_bottom(0.36);
          $report->set_footer('&L&D&R&P of &N', 0.17);
          $report->hide_gridlines();
        }

        sub set_columns {
          my ($ws, $col, $width, $format, $hidden) = @_;
          for (my $i=0; $i < @$width; $i++) {
            my $column = "$col:$col";
            $ws->set_column($column, $$width[$i], $$format[$i], $$hidden[$i]);
            $col++;
          }
        }
(Заб.: тук се указват стойностите за разчитане на лог файла, разделители и др)
        sub write_from_log {
          my ($worksheet, $file) = @_;

          # Create a new CSV parsing ojbect with the CSV options that I needed
          my $csv = Text::CSV_XS->new({
             'quote_char'  => '', # what?  no quote character?  you got it!
             'allow_whitespace' => '1', #No white space allowed - can cause issues with numbers format
             'escape_char' => '\\', # a backslash
             'sep_char'    => ',', # set separate char
             'binary'      => 0
           });

          # Row and column are zero indexed!
          my $row = 1;
          open (CSVFILE, "<", $file) || die "Unable to open $file for reading: $!, stopped";
          while (<CSVFILE>) {
            if ($csv->parse($_)) {
               my @Fld = $csv->fields;
               my $col = 0;
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
        # keep that line (row)?
        my $keep = 1;
         if ($keep) {
           foreach my $token (@Fld) {
            # I added the ability to store a .bmp with a .csv and import it
            # not the most elegant solution but it works
            # <img src="test.bmp" loc="D18">
            # I have not tested if it has to be in the same place in the .csv file
            # a later version will remove the need for the loc="Xnn" part
            # read: quick hack
            if ($token =~ /^<img/) {
              $token =~ /<img src="(.*)" loc="(.*)">/i;
              my ($img, $loc) = ($1, $2);
              $worksheet->insert_bitmap($loc, $img);
             } else {
              $worksheet->write($row, $col, $token);
             }
             $col++;
           }
          }
          $row++ if ($keep);
       } else {
         my $err = $csv->error_input;
         print "Text::CSV_XS parse() failed on argument: ", $err, "\n";
       }
     }
   }
```

IV. ПРОГРАМЕН КОД ЗА ИЗПРАЩАНЕ НА ГЕНЕРИРАНИ ТАБЛИЦИ

<code>

```
#!/bin/bash
```

(Заб.: този ред оказва изпращането на мейла – заглавие, текст, файл, който ще се изпраща. Както ще забележите, изпраща се ОТ (NBU alert system alerts@Nbu.bg) към nbu@nbu.bg), а заглавието съдържа Daily report и дата).

```
# Send mail | Echo is mail body  | EMAIL =  Set From (overrided default settings)
echo "Please see attached report" | EMAIL="NBU alert system <alerts@Nbu.bg>" mutt -s "Daily report - $( date '+%d/%m/%Y' )" "NBU <nbu@nbu.bg>" -a "/var/log/report.xls"
```

(Заб.: трябва да направим проверка дали има създадена директория за архивите)

```
# check if there is old_reports directory - logs archives. If not - mkdir
mkdir -p /var/log/old_reports
cp /var/log/report.xls /var/log/old_reports/report-`date +"%Y%m%d"`-`date +"%H%M%S"`.xls
```

(Заб.: изчакваме и след това нулираме лог файла)

```
# put the script into sleep mode
sleep 2

# null old report log
echo -n > /var/log/report.log

# output with info
echo "Alert sent. XLS file moved into Old Reports directory. Renamed with date included."
```

5. АВТОМАТИЗИРАНИ ЗАДАЧИ

Нужно е автоматично да се стартират горните програмни кодове. За тази цел имаме автоматизирани задачи, т. нар. Cron Jobs. Това представлява автоматизиран процес за стартиране на определени задачи и процеси. Чрез него можете да задавате време за

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

изпълнение на даден процес, както и да настройвате период за повторение на изпълнението на процеса.

За да видим настоящо какво се съдържа в списъка с автоматизирани задачи:

>crontab -l

За да променим или въведем редовете:

>crontab -e

<code>

```
# m h  dom mon dow   command
50 6 * * * /usr/local/bin/excel_convert.pl > /dev/null 2>&1
0 7 * * * /usr/local/bin/send_mail.sh > /dev/null 2>&1
```

Горният пример стартира програмните кодове всеки ден, 6:00ч и 7:00ч сутринта. В 6:50 стартира скрипта за генериране на таблицата и графиката, а в 07:00ч го изпраща.

6. ПРОГРАМЕН КОД ЗА АВТОМАТИЗИРАНЕ НА ПРОЦЕСА НА ВЪВЕЖДАНЕ

<code>

```
#!/bin/bash
#
# checks if you are root or not
if test "`id -u`" -ne 0
  then
  echo "You need to run this script as root!"
  exit
fi

program=dialog;
program2=locate;
program3=cpan;
perl_module_1=Spreadsheet::WriteExcel;
perl_module_2=Text::CSV_XS;
BACKTITLE="New Bulgarian University - Alert system";

#####################################################
#              Checking availability of dialog utility              #
#####################################################

# dialog is a utility installed by default on all major Linux distributions.
# But it is good to check availability of dialog utility on your Linux box.

condition=$(which $program 2>/dev/null | grep -v "not found" | wc -l)
if [ $condition -eq 0 ] ; then
   echo "$program is not installed. Installing now..."
   apt-get install $program -y > /dev/null
fi

condition=$(which $program2 2>/dev/null | grep -v "not found" | wc -l)
if [ $condition -eq 0 ] ; then
   echo "$program2 is not installed. Installing now..."
   apt-get install $program2 -y > /dev/null
   updatedb 2> /dev/null
fi

###################### Check CPAN initial configuration ####################
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
condition=$(locate MyConfig.pm 2>/dev/null | grep 'MyConfig' | wc -l)
if [ $condition -eq 0 ] ; then
   echo "$program3 is not configured. Configuring now..."

mkdir /root/.cpan > /dev/null
mkdir /root/.cpan/CPAN > /dev/null
cp /usr/lib/i386-linux-gnu/perl/5.20.2/Config.pm ~/.cpan/CPAN/MyConfig.pm
cat > ~/.cpan/CPAN/MyConfig.pm <<'EOL'
$CPAN::Config = {
  'applypatch' => q[],
  'auto_commit' => q[0],
  'build_cache' => q[100],
  'build_dir' => q[/root/.cpan/build],
  'build_dir_reuse' => q[0],
  'build_requires_install_policy' => q[yes],
  'bzip2' => q[/bin/bzip2],
  'cache_metadata' => q[1],
  'check_sigs' => q[0],
  'colorize_output' => q[0],
  'commandnumber_in_prompt' => q[1],
  'connect_to_internet_ok' => q[1],
  'cpan_home' => q[/root/.cpan],
  'ftp_passive' => q[1],
  'ftp_proxy' => q[],
  'getcwd' => q[cwd],
  'gpg' => q[/usr/bin/gpg],
  'gzip' => q[/bin/gzip],
  'halt_on_failure' => q[0],
  'histfile' => q[/root/.cpan/histfile],
  'histsize' => q[100],
  'http_proxy' => q[],
  'inactivity_timeout' => q[0],
  'index_expire' => q[1],
  'inhibit_startup_message' => q[0],
  'keep_source_where' => q[/root/.cpan/sources],
  'load_module_verbosity' => q[none],
  'make' => q[/usr/bin/make],
  'make_arg' => q[],
  'make_install_arg' => q[],
  'make_install_make_command' => q[/usr/bin/make],
  'makepl_arg' => q[INSTALLDIRS=site],
  'mbuild_arg' => q[],
  'mbuild_install_arg' => q[],
  'mbuild_install_build_command' => q[./Build],
  'mbuildpl_arg' => q[--installdirs site],
  'no_proxy' => q[],
  'pager' => q[/usr/bin/less],
  'patch' => q[/usr/bin/patch],
  'perl5lib_verbosity' => q[none],
  'prefer_external_tar' => q[1],
  'prefer_installer' => q[MB],
  'prefs_dir' => q[/root/.cpan/prefs],
  'prerequisites_policy' => q[follow],
  'recommends_policy' => q[1],
  'scan_cache' => q[atstart],
  'shell' => q[/bin/bash],
  'show_unparsable_versions' => q[0],
  'show_upload_date' => q[0],
  'show_zero_versions' => q[0],
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
  'suggests_policy' => q[0],
  'tar' => q[/bin/tar],
  'tar_verbosity' => q[none],
  'term_is_latin' => q[1],
  'term_ornaments' => q[1],
  'test_report' => q[0],
  'trust_test_report_history' => q[0],
  'unzip' => q[/usr/bin/unzip],
  'urllist' => [q[http://www.cpan.org/]],
  'use_prompt_default' => q[0],
  'use_sqlite' => q[0],
  'version_timeout' => q[15],
  'wget' => q[/usr/bin/wget],
  'yaml_load_code' => q[0],
  'yaml_module' => q[YAML],
};
1;
__END__
EOL
   updatedb 2> /dev/null
fi

######################## deletetempfiles function ################################

# This function is called by trap command
# For conformation of deletion use rm -fi *.$$

deletetempfiles()
{
   rm -f *.$$
}

############ Check and install MAIL application ##################

install_mail()
{
  dialog --infobox "Please wait.. checking.." 5 45
# check for mail app (sendmail)
condition=$(which sendmail | wc -l)
if [ $condition -eq 0 ] ; then
   echo "SENDMAIL is missing. Installing now...." > install_mail.$$
      apt-get -y install sendmail > /dev/null

      COUNT=10
      (
      while test $COUNT != 110
      do
            echo $COUNT
            echo "XXX"
            echo "Progress ($COUNT percent)"
            echo "XXX"
      COUNT=`expr $COUNT + 10`
      sleep 1
      done
      ) |
      dialog --title "Installing SENDMAIL" --gauge "Installing... Please wait..." 20 70 0
      updatedb 2> /dev/null

      else
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
        echo "Sendmail found. No need to install." > install_mail.$$
fi

# check if mail app is installed (mutt)
condition=$(which mutt | wc -l)
if [ $condition -eq 0 ] ; then
    echo "MUTT mail app has been installed." >> install_mail.$$
        apt-get -y install mutt >  /dev/null

  COUNT=10
  (
  while test $COUNT != 110
  do
                echo $COUNT
                echo "XXX"
                echo "Progress ($COUNT percent)"
                echo "XXX"
  COUNT=`expr $COUNT + 10`
  sleep 1
  done
  ) |
  dialog --title "Installing MUTT" --gauge "Installing... Please wait..." 20 70 0
  updatedb 2> /dev/null

        else
        echo "MUTT mail app found. No need to install" >> install_mail.$$
fi

  dialog --backtitle "$BACKTITLE" \
  --title "CHECK AND INSTALL MAIL APPLICATIONS" \
  --textbox install_mail.$$ 10 90


}

######################### PERL function ###################################

# Check and install PERL modules

install_perl()
{
  dialog --infobox "Please wait.. checking.." 5 45
# check for Perl module 1
condition=$(cpan -a | grep "$perl_module_1" | wc -l)
if [ $condition -eq 0 ] ; then
    echo "Perl module $perl_module_1 installed." > install_perl.$$
        perl -MCPAN -e "install '$perl_module_1'" > /dev/null

        COUNT=10
        (
        while test $COUNT != 110
        do
              echo $COUNT
              echo "XXX"
              echo "Progress ($COUNT percent)"
              echo "XXX"
        COUNT=`expr $COUNT + 10`
        sleep 1
        done
```

```
        ) |
        dialog --title "Installing '$perl_module_1'"--gauge "Installing... Please wait..." 20 70 0
        updatedb 2> /dev/null

        else
        echo "Perl module $perl_module_1 found" > install_perl.$$
    fi

    # check if module 2 is installed
    condition=$(cpan -a | grep "$perl_module_2" | wc -l)
    if [ $condition -eq 0 ] ; then
       echo "Perl module $perl_module_2 installed." >> install_perl.$$
        perl -MCPAN -e "install '$perl_module_2'" >  /dev/null

        COUNT=10
        (
        while test $COUNT != 110
        do
              echo $COUNT
              echo "XXX"
              echo "Progress ($COUNT percent)"
              echo "XXX"
        COUNT=`expr $COUNT + 10`
        sleep 1
        done
        ) |
        dialog --title "Installing '$perl_module_2'"--gauge "Installing... Please wait..." 20 70 0
        updatedb 2> /dev/null

        else
        echo "Perl module $perl_module_2 found" >> install_perl. $$
    fi

      dialog --backtitle "$BACKTITLE" \
      --title "CHECK AND INSTALL PERL MODULES" \
      --textbox install_perl.$$ 10 90

    }

    ######## generate EXCEL script #################

    excel_generate()

    {
    condition=$(if locate excel_convert.pl | grep excel > /dev/null ; then echo "1"; else echo "0" ; fi)
    if [ $condition -eq 0 ] ; then

    ################ EXCEL SCRIPT ################################
    #touch /usr/local/bin/excel_convert.pl
    cat > /usr/local/bin/excel_convert.pl <<'EOL'
    #!/usr/bin/perl
    ################## EXCEL SCRIPT START ############################
    use strict;
    use warnings;
    use Spreadsheet::WriteExcel;
    # This is a module used for templating Excel files. Its genesis came from the need to use the same
datastructure as HTML::Template,
    # but provide Excel files instead.
    #
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
#use Excel::Template;
use Text::CSV_XS;

#get date
my $date=`date +%Y-%m-%d`;
chomp($date);

#filename that we will generate
my $name = "report-$date.xls";

# Log files we are reading
my $file_report = "/var/log/report.log";

#check how many lines we have in log file
my $lines = `wc -l < $file_report`;

# Excel file we generate. $name is set above
my $workbook = Spreadsheet::WriteExcel->new("/var/log/$name");

# worksheet
my $report = $workbook->addworksheet("Reports");

my @array_report   =   ( 'Hour','Value 1','Value 2' );

my %cf = (
  font => 'Arial',
  border => 1,
  border_color => 'black'
  );

my %bold = (
  bold => 1
  );

my %ch = (
  align => 'center',
  valign => 'center',
  font => 'Arial',
  size => 11,
  border => 1,
  border_color => 'black',
  bg_color => 'yellow'
 );

my $ch = $workbook->addformat(%ch, %bold); # header
my $cn = $workbook->addformat(%cf, size => 10); # normal text
my $cs = $workbook->addformat(%cf, size => 8); # small text

$report->autofilter('A1:C1');

set_columns($report, "A",
  [15, 16, 16],
  [$cn, $cn, $cn]);

# this is where we read the data from log files and write to the xls

$report->write('A1', [ @array_report ], $ch);

write_from_log($report, $file_report);
```

```
$report->activate(); # I'd like the first worksheet to be active

# Chart generate
# available types are area, bar, column, line, pie, scatter, stock
my $chart = $workbook->add_chart( type => 'column', embedded => 1 );

# Configure the first series. (Sample 1)
$chart->add_series(
      name       => 'Values 1',
      categories => "=Reports!\$A\$2:\$A\$$lines",
      values     => "=Reports!\$B\$2:\$B\$$lines"
   );

# Configure the second series. (Sample 2)
$chart->add_series(
      name       => 'Values 2',
      categories => "=Reports!\$A\$2:\$A\$$lines",
      values     => "=Reports!\$C\$2:\$C\$$lines"

   );

# Add a chart title and some axis labels.
$chart->set_title( name => 'Daily results report' );
$chart->set_x_axis( name => 'Time' );
$chart->set_y_axis( name => 'Values' );

# Insert the chart into the worksheet (with an offset).
# insert_chart variables - column and line (Column E, line 2), chart variable ($chart), position X, position Y,
size X (4 is 400%), size Y (1.2 is 120%)
$report->insert_chart( 'E2', $chart, 50, 10, 4, 1.2 );

$workbook->close();  # play nice
exit(0);

# Setup the print options
# usage:

# print_setup($worksheet);

print_setup($report);

# $worksheet is a reference to a worksheet object
#
# These are the print options

sub print_setup {
  my $report = shift;
  $report->set_landscape();
  $report->center_horizontally();
  $report->set_margins_LR(0.25);
  $report->set_margin_top(0.5);
  $report->set_margin_bottom(0.36);
  $report->set_footer('&L&D&R&P of &N', 0.17);
  $report->hide_gridlines();
}

sub set_columns {
  my ($ws, $col, $width, $format, $hidden) = @_;
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
  for (my $i=0; $i < @$width; $i++) {
    my $column = "$col:$col";
    $ws->set_column($column, $$width[$i], $$format[$i], $$hidden[$i]);
    $col++;
  }
}

sub write_from_log {
  my ($worksheet, $file) = @_;

  # Create a new CSV parsing ojbect with the CSV options that I needed
  my $csv = Text::CSV_XS->new({
     'quote_char'  => '', # what?  no quote character?  you got it!
     'allow_whitespace' => '1', #No white space allowed - can cause issues with numbers format
     'escape_char' => '\\', # a backslash
     'sep_char'    => ',', # set separate char
     'binary'      => 0
   });

  # Row and column are zero indexed!
  my $row = 1;
  open (CSVFILE, "<", $file) || die "Unable to open $file for reading: $!, stopped";
  while (<CSVFILE>) {
    if ($csv->parse($_)) {
        my @Fld = $csv->fields;
        my $col = 0;
        # keep that line (row)?
        my $keep = 1;
         if ($keep) {
           foreach my $token (@Fld) {
            # I added the ability to store a .bmp with a .csv and import it
            # not the most elegant solution but it works
            # <img src="test.bmp" loc="D18">
            # I have not tested if it has to be in the same place in the .csv file
            # a later version will remove the need for the loc="Xnn" part
            # read: quick hack
            if ($token =~ /^<img/) {
              $token =~ /<img src="(.*)" loc="(.*)">/i;
              my ($img, $loc) = ($1, $2);
              $worksheet->insert_bitmap($loc, $img);
            } else {
              $worksheet->write($row, $col, $token);
            }
            $col++;
          }
        }
        $row++ if ($keep);
    } else {
      my $err = $csv->error_input;
      print "Text::CSV_XS parse() failed on argument: ", $err, "\n";
    }
  }
}

########## EXCEL SCRIPT END ###################
EOL
chmod +x /usr/local/bin/excel_convert.pl
updatedb 2> /dev/null
    echo "EXCEL generator installed." > excel_generate.$$
```

```
        else
        echo "EXCEL generator found." > excel_generate.$$

    fi

      dialog --backtitle "$BACKTITLE" \
      --title "CHECK AND GENERATE EXCEL CONVERT SCRIPT" \
      --textbox excel_generate.$$ 10 90

    }

    ###################### SEND MAIL SCRIPT CHECK / GENERATE ##########

    sendmail_script_check()

    {
    condition=$(if locate /usr/local/bin/send_mail.sh | grep mail > /dev/null ; then echo "1"; else echo "0" ; fi)
    if [ $condition -eq 0 ] ; then

    cat > /usr/local/bin/send_mail.sh <<'EOL'
    ############# SEND MAIL SCRIPT START ##################
    #!/bin/bash

    # check if internet connection is available
    while ! ping -c1 8.8.8.8 &>/dev/null
        do echo "Ping Fail - `date`"
    done

    echo Internet connection detected. Continue...

    # Send mail | Echo is mail body  | EMAIL =  Set From (overrided default settings)
    echo "Please see attached report" | EMAIL="NBU alert system <alerts@Nbu.bg>" mutt -s "Daily report - $(
date '+%d/%m/%Y' )" "NBU <info@vsimeonov.com>" -a /var/log/report*.xls

    # check if there is old_reports directory - logs archives. If not - mkdir
    mkdir -p /var/log/old_reports
    mv /var/log/report*.xls /var/log/old_reports/report-`date +"%Y-%m-%d"`-`date +"%H-%M"`.xls
    mv /var/log/report*.log /var/log/old_reports/report-`date +"%Y-%m-%d"`-`date +"%H-%M"`.log

    # create new report log
    touch /var/log/report.log
    updatedb 2> /dev/null

    # output with info
    echo "Alert sent. XLS file moved into Old Reports directory. Renamed with date included."

    ############# SEND MAIL SCRIPT END ##########################
    EOL

    chmod +x /usr/local/bin/send_mail.sh
    updatedb 2> /dev/null
       echo "SEND MAIL script installed." > send_mail.$$
        else
        echo "SEND MAIL script found." > send_mail.$$

    fi

      dialog --backtitle "$BACKTITLE" \
      --title "CHECK AND GENERATE SEND MAIL SCRIPT" \
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
  --textbox send_mail.$$ 10 90
}

######################## crontab function - check ############################

crontab()

{
condition=$(if cat  /var/spool/cron/crontabs/root | grep -q excel ; then echo "1"; else echo "0" ; fi)
if [ $condition -eq 0 ] ; then
  echo "Scheduled tasks" > crontab.$$
  echo "" >> crontab.$$
  echo "Minute (0 - 59) | Hour (0 - 23) | Day of month (1 - 31) | Month (1 - 12) | Day of week (0 - 6) (Sunday=0) | Command " >> crontab.$$
      echo "" >> crontab.$$
      echo "--------------------------------------------------------------------------------" >> crontab.$$
  echo "No Cron task installed!" >> crontab.$$
  else
  echo "Scheduled tasks" > crontab.$$
  echo "" >> crontab.$$
  echo "Minute (0 - 59) | Hour (0 - 23) | Day of month (1 - 31) | Month (1 - 12) | Day of week (0 - 6) (Sunday=0) | Command " >> crontab.$$
  echo "" >> crontab.$$
  echo "--------------------------------------------------------------------------------" >> crontab.$$
  cat /var/spool/cron/crontabs/root | grep -v '#' >> crontab.$$

fi

  dialog --backtitle "$BACKTITLE" \
  --title "CHECK CRONTAB" \
  --textbox crontab.$$ 30 140
}

################### crontab_add function ##########################

add_crontab()

{
condition=$(if cat  /var/spool/cron/crontabs/root | grep -q excel ; then echo "1"; else echo "0" ; fi)
if [ $condition -eq 0 ] ; then

echo "5 * * * * /usr/local/bin/excel_convert.pl > /dev/null 2>&1" >> /var/spool/cron/crontabs/root
echo "0 * * * * /usr/local/bin/send_mail.sh > /dev/null 2>&1" >> /var/spool/cron/crontabs/root
echo "Scheduled tasks added in crontab '/var/spool/cron/crontabs/root'" > add_crontab.$$
      else
      echo "Scheduled tasks found." > add_crontab.$$

fi

  dialog --backtitle "$BACKTITLE" \
  --title "ADD CRONTAB" \
  --textbox add_crontab.$$ 10 90
}

######## change_hostname function ##########################

change_hostname()

{
```

```
tempfile=`tempfile 2>/dev/null` || tempfile=/tmp/change_hostname.$$
trap "rm -f $tempfile" 0 1 2 5 15

dialog --title "Change hostname input box" --clear \
       --inputbox "\n
  \n
  Try entering your name below:" 16 51 2> $tempfile

retval=$?

case $retval in
 0)
   echo "Input string is `cat $tempfile`";;
 1)
   echo "Cancel pressed.";;
 255)
   if test -s $tempfile ; then
     cat $tempfile
   else
     echo "ESC pressed."
   fi
   ;;
esac

  dialog --backtitle "$BACKTITLE" \
  --title "CHANGE HOSTNAME" \
  --textbox change_hostname.$$ 10 90
}

####### hostname function ##########################

hostname()

{

  dialog --backtitle "$BACKTITLE" \
  --title "CHECK HOSTNAME" \
  --textbox hostname.$$ 10 90
}

####### diskstats function ############################

diskstats()
{
  df -h | grep "^/" > statsfile.$$
  dialog --backtitle "$BACKTITLE" \
  --title "DISK STATISTICS" \
  --textbox statsfile.$$ 20 90
}

####### check for installed modules ###################

check()
{

  dialog --infobox "Please wait.. checking.." 5 45

condition=$(cpan -a | grep "$perl_module_1" | wc -l)
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
if [ $condition -eq 0 ] ; then
   perl1="$perl_module_1 not installed."
      else
   perl1="$perl_module_1 found"
      fi

condition=$(cpan -a | grep "$perl_module_2" | wc -l)
if [ $condition -eq 0 ] ; then
   perl2="$perl_module_2 not installed."
      else
   perl2="$perl_module_2 installed"
      fi

condition=$(which sendmail | wc -l)
if [ $condition -eq 0 ] ; then
   sendmail="SENDMAIL not found."
  else
   sendmail="SENDMAIL installed"
      fi

condition=$(which mutt | wc -l)
if [ $condition -eq 0 ] ; then
   mutt="MUTT not found."
      else
   mutt="MUTT installed"
      fi

condition=$(if cat  /var/spool/cron/crontabs/root | grep -q excel ; then echo "1"; else echo "0" ; fi)
if [ $condition -eq 0 ] ; then
   cron="Alert system CRON not found"
      else
   cron="Alert system CRON found"
      fi

condition=$(if cat  /usr/local/bin/send_mail.sh | grep -q bash ; then echo "1"; else echo "0" ; fi)
if [ $condition -eq 0 ] ; then
   mail_script="Mail script not found"
      else
   mail_script="Mail script found"
      fi
echo "Mail: $sendmail
Mail: $mutt
Perl modules: $perl1
Perl modules: $perl2
Email script: $mail_script
CRON: $cron" > status.$$

dialog --backtitle "$BACKTITLE" \
  --title "Alert System status" \
  --textbox status.$$ 10 90

}

############## GENERATE TEST REPORT LOG #############

report_log()
{
condition=$(if locate /var/log/report.log | grep report > /dev/null ; then echo "1"; else echo "0" ; fi)
if [ $condition -eq 0 ] ; then
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
dialog --infobox "REPORT LOG file not found. Installing test report log file..." 5 45

touch /var/log/report.log
cat > /var/log/report.log <<'EOL'
############## Report LOG START ###############
00:01,12,14
00:12,40,20
00:15,40,60
00:40,20,100
01:20,10,20
02:30,20,21
06:20,30,40
08:20,10,10
11:30,100,20
EOL

updatedb 2> /dev/null
   echo "Report log installed." > report_log.$$
      else
      echo "Report log found." > report_log.$$

fi

  dialog --backtitle "$BACKTITLE" \
  --title "CHECK FOR REPORT LOG FILE" \
  --textbox report_log.$$ 10 90
}

#####################################################
#                    MAIN STRATS HERE                #
#####################################################

trap 'deletetempfiles'  EXIT     # calls deletetempfiles function on exit

while :
do

# Dialog utility to display options list

   dialog --clear --backtitle "$BACKTITLE" --title "MAIN MENU" \
   --menu "Use [UP/DOWN] key to move" 30 90 15\
   "CHECK"     "Check for installed modules"\
   "MAIL"  "Check and install MAIL app" \
   "PERL"  "Check and install PERL modules" \
   "EXCEL" "Check and generate EXCEL script" \
   "MAIL_SCRIPT" "Check and generate MAIL script"\
   "CRONTAB"   "Check scheduled tasks"\
   "ADD_CRONTAB" "Add alert system scheduled tasks"\
   "CHANGE_HOSTNAME"   "Change hostname"\
   "HOSTNAME"   "Check current hostname"\
   "REPORT_LOG" "Check for Report Log file"\
   "DISK"      "Display disk statistics" \
   "EXIT"      "EXIT" 2> menuchoices.$$

   retopt=$?
   choice=`cat menuchoices.$$`

   case $retopt in
```

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

```
        0) case $choice in

             MAIL)   install_mail ;;
            PERL)   install_perl ;;
             EXCEL)  excel_generate ;;
             MAIL_SCRIPT) sendmail_script_check;;
             CRONTAB)    crontab;;
             ADD_CRONTAB) add_crontab;;
             CHANGE_HOSTNAME) change_hostname;;
             HOSTNAME)  hostname;;
            DISK)       diskstats ;;
             REPORT_LOG)             report_log ;;
             CHECK)            check;;
            EXIT)       clear; exit 0;;

          esac ;;

       *)clear ; exit ;;
   esac

done
```

7. ОТСТРАНЯВАНЕ НА ГРЕШКИ

- при грешка ^M: bad interpreter - инсталираме dos2unix

>apt-get install dos2unix

стартира се dos2unix на проблемния файл (пример: dos2unix excel_convert.pl)

Горната грешка обикновено се получава, ако се прави трансфер на готови скриптове от Windows към Linux през WinSCP клиент.

- При проблем с изпращането на email

Изпълняваме двата скрипта. След това проверяваме последните 15 реда от мейл логовете:

>tail -n15 /var/log/mail.log

8. ПОЛЕЗНИ ИНСТРУМЕНТИ

- инсталиране на locate (за намиране на файлове)

>apt-get install locate

- обновяване на базата данни за locate (това е нужно тогава, когато искаме да използваме командата веднага. В противен случай периодично се обновява базата с данни и не е нужно да се изпълнява updatedb)

>updatedb

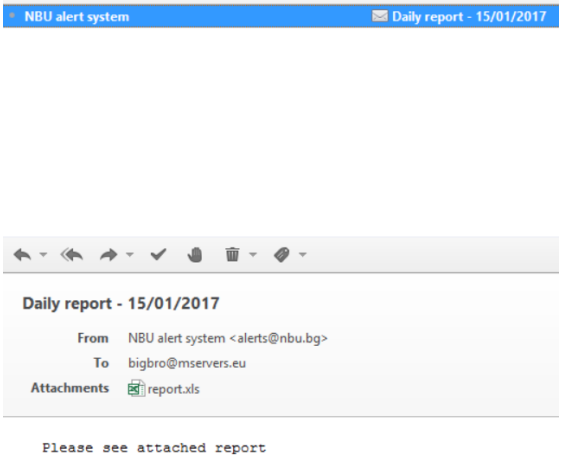
9. РЕЗУЛТАТИ

- как изглежда директорията с архивите

```
6144 Jan 14 23:49 report-20170114-234910.xls
6656 Jan 15 00:58 report-20170115-005810.xls
6656 Jan 15 13:00 report-20170115-130004.xls
```

Фиг. 1. Структуриране на генерираните файлове

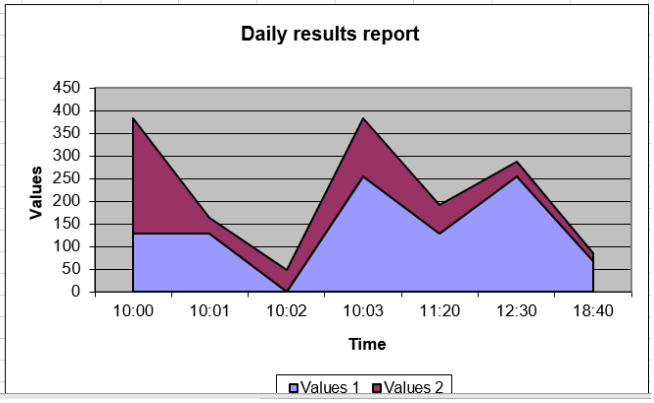
- как изглежда изпратеното писмо с прикачен генериран файл



Фиг. 2 Изпратено писмо на електронна поща с прикачен файл

- как изглеждат генерираните таблица и графика.

Hour	Value 1	Value 2
10:00	128	255
10:01	128	36
10:02	0	48
10:03	255	128
11:20	128	64
12:30	255	32
18:40	68	16



Фиг. 3. Графика и резултати изведени в табличен вид.

10. ИЗВОДИ

ISSN 2534-854X (online) http://www.telecommunications.nbu.bg/bg/godishnik

След въвеждането на системата се постига оптимизиране на процеса по следене на показателите, спестява се време на инженерите, като им предоставя информацията в удобен за разчитане вид.

Важно е да се знае, че по време на изпълнението на всеки такъв проект е необходимо да се пресъздаде работната среда, където ще се изпълнява програмния код. Също така е наложително програмния код да се въвежда в нова машина с фабрични найстройки, за да се опише процеса в техническата документация, заедно с резултатите от тестовите постановки.

ИЗПОЛЗВАНИ ИНСТРУМЕНТИ:

[1] Oracle VirtualBox
[2] Огледално копие на : PIXEL + Debian ISO (Raspbery PI clone).

За контакти:
Велизар Симеонов Симеонов, F70406, студент в учебна програма "Телекомуникации" на Департамент ”Телекомуникации”, НБУ, ул. Монтевидео № 21, e-mail:

Дата на постъпване на ръкописа: 06.03.2017
Дата на получена рецензия: 25.03.2017
Дата на приемане за публикуване: 25.03.2017

ANNEX FOR EMAIL STATISTICS IN DATA COLLECTION SYSTEMS, TABLES AND GRAPHICS GENERATION

Velizar Simeonov

Abstract: The system is a set of program rows that send a pre-generated output file with a measurement device, format the information in a table, and send it to a predefined e-mail. Additionally, a program code has been added to facilitate installation of the system.
Keywords: generation, charts, information, tables, system.