The Newsletter for SIR UK Users

SIR REPORTER

Contents

Editorial

Hello playmates! Yes, it's been a year since the last issue of SIR Reporter and there is lots of news: we had a brilliant conference in Stratford upon Avon last summer – if you were not lucky enough to be there, Adrian Hodgson has kindly provided a full report in this issue. Since then John Lemon has carried out a survey of conference attenders on our behalf, to gauge their opinions on conferences and venues, with the result that we are having another London conference with training this June, and are working towards an international conference in 2008 in a venue we know you will love!

There has just been a major new release of SIR/XS which can be downloaded from

http://www.sir.com.au/download/index.html

and David Baxter from SIR Oz has provided some serious training in the use of the SYSPROC file in this issue of Reporter.

Please make sure you register to keep receiving news of Reporter at http://www.soton.ac.uk/~sug/edit.html because this is the last printed version.

Finally, thanks to all the contributors to this issue – we hope you enjoy it.

Kathy Brooks
Kathy@forvus.co.uk

SIR is at http://www.sir.com.au
UK SIR Users Group is at http://www.soton.ac.uk/~sug

Chairman's Chat

Welcome to the last printed copy of Reporter, from now on it will be distributed electronically only. This and back-issues for several years are available at

http://www.soton.ac.uk/~sug/reporter/index.html

if you would like an email when a new issue is available please fill in the postcard included with this issue or go to

http://www.soton.ac.uk/~sug/edit.html

and sign up.

This issue also has details of a one day conference at the Institute of Education in London on the 15th of June for £40, preceded by a training day available at an extra £50. The booking web page is at

http://www.soton.ac.uk/~sug/confbook.html

I would also like to take this opportunity to thank Patrick Brown for his services to the committee he will be missed. He has resigned from the committee as he no longer works with SIR.

If you feel you have something to contribute to the user group please contact me and we can make sure you are considered at the AGM to be held at the conference in June.

Dave Doulton

SIR Training and Conference 14-15 June 2007, London

Thursday June 14th 2007	**SIR Intermediate Training**
Venue	Institute of Education, Gordon Square, London WC1H 0NT Rooms 101/102
Price	£50.00
Friday June 15th 2007	**SIR Conference and AGM**
Venue	Institute of Education, Elvin Hall, London WC1H 0AL
Price	£40.00

Please book your place at:

http://www.soton.ac.uk/~sug/confbook.html

SIR UK USERS GROUP AGM AGENDA 15th June 2007

1. Apologies for Absence
2. Minutes of the last Annual General Meeting
3. Matters Arising
4. Chair's report (Dave Doulton)
5. Treasurer's report (Jon Johnston)
6. Elections
7. A.O.B.

The committee currently consists of:

- Kathy Brooks ,Dave Doulton (Chair), Adrian Hodgson, Jon Johnson(Treasurer), John Lemon and Fran Williams who were elected at the 2006 AGM and continue to serve (Bylaws article6(b))

- Mo Reardon who is the ex-officio member appointed by the SIR distibutor (Bylaws article 5(b)).

There are thus 2 vacant positions to fill by elections (Bylaws article 5(a))

Nominations for committee membership will be received by the current chair (Dave Doulton) up to 11am on 15th June 2007 (Bylaws article 6(e))

SIR Users Conference and Training 28-30 June 2006

Conference write-up by Adrian Hodgson

Solving and creating Suduko puzzles using SIR – Dave Doulton (University of Southampton)

Dave's interest in solving all the worlds Sudoku puzzles started when he observed a colleague spending hours puzzling over a tricky variety in coffee breaks and lunch times.

'Hmm' … said Dave. 'I can write a SIR program to do that!'

The standard format of the number puzzle involves a square of 81 cells of nine by nine, partitioned into nine sub-squares of 3 x 3 cells, with some of the correct numbers already completed at the start.
The challenge of Sudoko is to complete the empty cells in the grid using the numbers 1 to 9 in each row, column, and sub-square. The numbers 1 – 9 must appear once and once only in each row, column and sub-square. The standard puzzle varies in difficulty depending on the number of cells that are already completed in advance.

Dave of course started with some desk research and came up with some interesting statistics about Sudoku: -

The number of newspapers worldwide that feature sudoku: about 140.
Sudoku books available to buy on Amazon: 328
Results for "sudoku" generated by Google search: 90,800,000.
Real name: suuji wa dokushin ni kagiru (the digits must remain single).
The number of different valid sudoku grids, as calculated by mathematicians Bertram Felgenhauer and Frazer Jarvis: 6,670,903,752,021,072,936,960.
The date at which we might, therefore, run out of puzzles: some time early in the 3,607,744th century.
The size of a sudoku puzzle carved into a hillside overlooking the M4 in Gloucestershire to promote a Sudoku gameshow: 6,400sq metres.

From Indypedia ©Independent News & Media 2006

Dave's first task was to develop some rules that could be used by the SIR program to solve a standard
9 x 9 sudoko puzzle. For those blank cells where the solution digit was not already completed fill in the cell with the candidate digits 1-9.

He ended up with twelve rules as follows: -

1. Check each cell with candidate digits in a column, and remove any digits as candidates that appear in the column as a single digit elsewhere
2. Check each cell with candidate digits in a row, and remove any digits as candidates that appear in the row as a single digit elsewhere
3. Check each cell with candidate digits in a sub-square, and remove any digits as candidates that appear in the sub-square as a single digit elsewhere
4. Check if a number is only a candidate for selection in one cell in the column
5. Check if a number is only a candidate for selection in one cell in the row
6. Check if a number is only a candidate for selection in one cell in the square
7. Check if a number is only in one of the columns of a sub-square. If it is then it must be in that sub-square for that column, and can be removed as a candidate in the rest of the column.
8. Check if a number is only in one of the rows of a sub-square. If it is then it must be in that sub-square for that row, and can be removed as a candidate in the rest of the row
9. Check for n numbers appearing in n cells in a column. If they do then these numbers can be removed as candidates from the rest of the column. e.g. 1,5 appearing as candidates in two different cells.
10. Check for n numbers appearing in n cells in a row. If they do then these numbers can be removed as candidates from the rest of the row.
11. Check for n numbers appearing in n cells in a square. If they do then these numbers can be removed as candidates from the rest of the square.
12. if all else fails - GUESS !

Rules 7 -11 are thanks to an experienced sudoku player and colleague of Dave's called Mark Johnston

Sometimes there may be more than one solution. Dave's SIR solver will only find one of the possible solutions and relies on the user to spot any variations.

For graphical examples refer to Dave's slides on the SIR User Group website.

There are several variations of the standard game available including: -

- Smaller grids of six by six using the numbers 1 - 6
- 9 x 9 grids using letters in place of numbers (called 'Godoku')
- 16 x 16 grids using the numbers 1 – 9 and letters A – F (called 'Super Sudoko')
- Variations where extra work is required .e.g. the sum of adjacent cell contents are given as a starting clue.

Creating a sudoku puzzle

In order to create a puzzle Dave proceeds as follows: -

Find a solution grid

- Randomly fill the diagonal squares with the digits 1 – 9 as these do not interact with one another
- Try adding extra numbers that are possible values i.e. do not already occur in the row column or square
- When a completed grid has been found store in a tabfile if not already present.

Find a partially completed subset of this grid that is solvable using the SIR solver

- Choose n cells at random (n dependent on how difficult you want the answer to be, the higher the number the easier the solution)
- loop
- Try to solve the generated problem.
- If it solves then exit loop
- else
- Choose another cell
- End loop
- loop
- If number of cells=n then exit loop
- Clear a random cell with a value
- Try to solve
- If it solves then next loop
- Else
- Reinstate value to cleared cell
- End loop

Note exit creation if time gets greater than specified time.
If solution not already in tabfile then store solution in tabfile

Have you got previous FORM (s), SIR? – Jackie Palmer (University of Edinburgh)

Data Collection, past present and future in the Edinburgh Study of Youth Transitions and Crime.
Jackie introduced us to the murky world of Youth crime. The Edinburgh Study of Youth Transitions and Crime is a study following a cohort of 4300 young people through their teens into adulthood, focusing on delinquent behaviour. The Study began in 1998 as pupils started secondary school across the city, and unlike other research, includes mainstream, independent and special school pupils of both genders.
The main aims of the project were to: -

- Identify the factors which impact on young people's involvement in offending behaviour
- To examine the differences between males and females in the extent and patterns of criminal offending
- To develop new theories that will help to contribute to practical policies targeting young people.

Data was collected from numerous sources.

Annual collections were used for the Cohort data, school attendance records, social work records, and Scottish Children's Reporter Administration (SCRA) information.

Neighbourhood data was gathered from Lothian and Borders Police crime statistics and census data, and one-off collections were used for information from parents, class form teachers, school attainment records and police JLO warnings (that's a Juvenile Liaison Officer and nothing to do with Jenifer Lopez ! that appears if you "Google JLO" - Adrian)

In future the project is hoping to have access to the adult records for the cohort from the Scottish Prison Service, the Procurator Fiscal/Crown Office, the Police and Adult Social Work records. One success has already been achieved in being able to process records from 350 individuals from the Scottish Criminal Records Office.

Jackie showed us the variety of buildings she had visited during the course of data collection ranging from fake victorian gothic splendour (possibly used for Hogwarts film shoots) to a huge 60's tower block school which has to be closed in high winds.

Anecdotes of particular fascination included: -

- "A Blair" carved in graffiti on the oak-panelled walls of one school
- Collecting information from a 13 year old who disappeared to take drugs in the middle of the interview and came back to cause havoc
- Interviewing in a residential unit whilst social workers tried to calm down a boy brandishing a broken CD as a weapon
- Visiting the remand unit of the local prison and standing in line with visiting families to have thumbprints recorded on the way in and the way out
- Conducting interviews with teenagers in varying states of undress due to dragging them out of bed too early
- Visits to teenage boys bedrooms!

As for the technical side, data is entered via SIR FORMS into one of up to twenty databases and a PQL program is used to define missing values for filtered variables. SIR is used to undertake a basic check and cleaning of the data and then an SPSS file is written. Further checks are carried out in SPSS, before the researchers carry out their analysis in SPSS or transfer subsets of data to Stata or SAS. A potential development in future would be to create a web interface in SIR, and use on-line questionnaires as the cohort has dispersed around the globe.

Finally Jackie presented some key findings from the research: -

- Victims become offenders and vice – versa
- Boys are more likely to offend than girls, more often and more seriously
- The peak age group for offending is 14 – 15
- The impact of parenting is weaker during mid adolescence as teenagers are more influenced by peer groups.

Randy Banks (University of Essex) - PQL Programming using Object-Oriented Methods

Randy introduced us to the world of OOD (Object Oriented Design). OOD is a conceptual modelling technique which can be used for a wide range of software engineering problems including general database design. It is a useful method to help in the creation of code that is reliable, reusable, maintainable and robust. OOD is used to subdivide a topic of interest into classes, objects and the associations between them.

One of the tools used to support OOD is the Unified Modelling Language (UML) which is maintained by the Object Management Group (OMG), and is one well known example of an Object-Oriented Programming Language (OOPL).

The Sir Procedural Query language (PQL) is not an OOPL, but the concepts of Object-Oriented design can be mapped on to PQL.

Randy has been working on a real life example to create a generic software-independent means of storing/transferring survey metadata, including SIR/DBMS schema information.

The objective is to deliver the required metadata in a Survey Delivery System Exchange Format or Survey Delivery System XML Format known as SDSX. Randy has been developing SIR/SDSX to achieve this as an application to write database schema information into SDSX format.

He outlined his application in terms of OOD packages, which are used to organise and group related classes together. In the SIR/SDSX application there are User Interface, Database, Record and variable packages amongst others.
Randy concluded that even though PQL is not an OOPL, the techniques of OOD can be applied to PQL applications.

The extra effort at the design stage will have dividends in terms of less hassle when writing and maintaining code later.
Further detail is provided in the PowerPoint presentation from the conference on the SIR Users Web site.

Viva Las Vegas
John Lemon

John is a regular contributor to the SPSS and Snap support forums and was contacted by a desperate Snap user about some help with setting up a scanning exercise.

After John's initial reply with a list of do's and don'ts John was surprised to get another call from Bill asking if he could come and run the operation, and by the way it's in Las Vegas

The requirement turned out to be for the National Automotive Parts Authority (NAPA) conference in Las Vegas with an anticipated 13,000 delegates on 4 sites. The requirement was to scan session attendance forms

Conference sessions consisted of 12 themes and 3 sub-themes resulting in 36 different form types.

The last session closed at 18.30 and results were required by 12.00 noon the next day.

Just to make matters worse, the analysis team only had 1 Snap license available, the temporary staff couldn't use Snap, the original scanner machines was far too slow (7 pages per minute), and the paper size used was different to the UK A4 size that John was expecting.

If that wasn't enough to contend with, NAPA were changing the goalposts daily, one batch of printed forms were burnt in a printer fire, 1 temp only spoke Spanish, and a couple of memory sticks were burnt out by the high levels of static electricity in Vegas.

Save Our Student records
John Lemon

The summer school at the University is a pre-cursor for direct entry.
They contacted John with a corrupt Access database. They had tried using the repair tools in Access but to no avail. John imported the data into SIR using ODBC; Record 17 of the imported data looked corrupt – with what looked like Chinese characters. John managed to work round this and ended up with 108 records in the SIR database version for 106 students. The extra records were two null records at the start and the end of the file.

Building a SIR application for a randomized clinical trial: Pushing PqlForms to its limits - Leif Spange Mortensen (UNI-C Aarhus)

MANTRA-PAF (Medical antiarrhythmic treatment or radiofrequency ablation in paroxysmal atrial fibrillation) is an ongoing randomized clinical trial comparing standard antiarrhythmic medical treatment with a new invasive technique for treatment of atrial fibrillation. Centres in Finland, Germany, Sweden and Denmark participate and the study secretariat is located in Aarhus, Denmark.
SIR2002 was chosen as the data management tool for the study. The data base structure, with approximately 25 record types, is rather complex, and on top of this PqlForms and PQL have been used to construct a comprehensive and user friendly data base application letting the user be happily unaware of SIR details. During development of the application numerous e-mails about PqlForms issues have been exchanged between Sydney and Aarhus.

Main features of the database application are:

- A login and user administration system
- An extensive data validation suite – including validation of dates to avoid negative elapsed time periods
- Widespread use of default values

Three main types of user are defined which are Database Administrator, Super User and Basic.
A database connect subroutine is stored as a compiled version in the SYSPROC family. The source code is stored in a member inside the database itself, and is compiled from there.
The SYSPROC.SYSTEM family contains code to deal with user passwords. Passwords input by users that result in unsuccessful logins can be recorded for audit purposes. Passwords are pseudo encrypted by padding to length 10 with blanks and by adding a special ASCII code to each password. General users are defined in their user profile and are directed to a basic system functionality with a statement CALL SYSPROC.SYSTEM.BASIC2.This directs the basic users to the specific application and prevents them from having access to the general features in SIR.
This also ensures that some of the main screen buttons are greyed out or hidden for basic users. Examples of this are the disabling of a system admin button and of the normal exit button:
DISABLE ITEM abutton
DISABLE ITEM xbutton - disabling the exit button so that Basic users are forced to log out properly for audit trail purposes.
Some of the button disabling work required a non-elegant solution and involved the examination of low level visual PQL code generated by the PqlForms compiler to detect button ID's.
Leif has also written code for dealing with database backups. The current date is built into the backup file name and the filename ends in one of the letters a-z. The program cycles through backups already on the backup device checking for the last letter of the filename and unloads the backup data using the first suffix letter for the given date that has not already been used. Backup is always generated in C:\MANTRABU, but may also be generated on devices E:\, … . H:\ (USB hard-drive, USB-stick etc.) if a directory with the above specific name exists on the device.
Improvements introduced in SIR/XS, partially as result of the numerous e-mails exchanged between Sydney and Aarhus, include:

- The ID n clause on FBUTTON
- The REMOVE clause on FBUTTON is now documented.
- Entering local program variables in SCREEN MENU's now works

Leif concluded that PqlForms is now a maturing SIR product and that more G is desirable in the GUI part of PqlForms.

Managing Research Projects with SIR: Interfacing with Labs

Lori Hoepner, Greg Neils and David Merle
Columbia University, New York

Traditionally, the use of SIR databases in health sciences research has focused on the organization, documentation and analysis of the research data itself, with the database directly accessible only by data management staff.
SIR functionality, together with the connectivity provided by the internet, now make it possible to build interfaces that allow data entry staff, labs, project coordinators, investigators and statisticians to interact dynamically and directly with the project database from remote locations.
Ideally, each customized interface in a project is designed to provide functionality specific to each role on the project, and information is made available securely, on a 'need to know' basis.
We demonstrate several interfaces that demonstrate the ways in which SIR can enhance scientific project management as well as data management.

Greg Neils

The team has 49 current databases, the largest one having 5,103 cases and 288,035 records. The average database has a lifespan of five years and in the ideal scenario the team like to get involved with the clients before they 'muck it up'.
The majority of end users want to do "nothing" and want data/ reports in Excel or SPSS format. They have developed generic code, with help from David Baxter, to work on databases to facilitate generic clickable user friendly ways of selecting data and print outs.
This includes an activity history window which stores a list of the last ten activities requested, so the end user can select an option to repeat if required.
Greg ended his session with a touch of irony.
"For those of you who want to see lines of coding here is a screen shot of about 12 pages of code which is very small. I will buy you a beer if you find a mistake but you will have to come and get it (from the States)! "

David Merle

David talked about two databases to support a study into the prevalence of Alzheimer's.
These are a national database for the National Alzheimer's Coordinating Center (NACC) and a local database for the Alzheimer's disease Research Center (ADRC). The ADRC database uses old style SIR forms, whereas NACC uses new style PQLFORMS.

The users sometimes have a need for combined analysis across the two databases and David has been looking at options for making this more efficient, as some users currently get confused about which of the tw0 databases they are using. An example of this is coordinating visit numbers across the two databases.

Option1

Get rid of the ADRC database and merge into an ADRC /NACC database.
Issue: - There are 4000 ADRC subjects who are ineligible for NACC

Option 2 – The Frankenstein solution

Leave ADRC as it is and take parts of ADRC to interact with NACC.
Issue: -This would create problems with the synchronization of the ADRC data that would be common to both databases

Option 3

Create one interface to connect to both the NACC and ADRC databases using SIR 2002, with a main menu and Forms to interact with the data.

This is the chosen option. A default database attribute is created NACC and two attributes are set to switch to the require current database (OLDADRC and NEWNACC)

Lori Hoepner

Lori introduced us to a tracking participant database (PARTLIST) for the Columbia Center for Children's Environmental Health. She acknowledged the help of Tom Shriver, Greg, David and Ming Feng.

The database is used to track 723 Dominican and African American mother – infant pairs from the 3rd Trimester through to the child being eight years old. It is used to help identify health risks for young children such as exposure to environmental toxins in New York City.

It has been developed to replace the original application which was held in an Excel spreadsheet. This was a live application being accessed and added to every day by a minimum of 11 researchers. The Excel system had developed numerous problems including: -

- Exceeding the maximum number of columns for Excel
- Constantly freezing and crashing
- Took a long time to load on older PC's

- Users sometimes used SAVE rather than SAVE AS and overwrote the original spreadsheet with a filtered
- dataset, effectively losing some data
- Security risks as it was not password protected

Lori reviewed the options for replacing the spreadsheet. The first idea was to rip the application apart and build a brand new database from scratch. This was rejected due to lack of time and budget.

The alternative approach selected was to keep the original spreadsheet structure in a single record type using Tom Shriver's DV Painter with SIR. Users can now only edit data on their subjects and only one user can enter dust allergens data. It is hoped in time to connect PARTLIST to the research database to enable joint reports.

True Audit trails using SIRxs
Tom Shriver

The introduction of the enhanced journaling capabilities in SIRxs is an exciting development which means that it is now possible to process overnight all the daily transactions in order to produce a summary of the days activities.

In older versions of the software, batch data input could be audit trailed using processing on the log and summary files, but this and any other audit trails needed masses of PQL code, so this tended to put most users off making the effort.

Tom is actively considering preparing code for generic audit trails using the Journal record entries and would like to hear from all those users that are interested in this so that he can start an e-mail forum to capture the varied requirements of the users. The longer term aim could be a joint multi-national presentation at the next conference of SIRxs generic audit trails

Twenty into 1 does go – Tony Reardon SIR PTY

Tony guided us through an entertaining story about a support project for Hanover medical school, referring to copies of the e-mail dialogue along the way.
The original data was held in an Access database.

The issues that needed to be dealt with were: -

Unsuitable variable names for SIR 2002 (Too long and contained some special characters)
Different formats and naming for the same variable on different record types
Embedded Carriage returns and Line feeds within the data
A solution was needed urgently, in order to present some results at an EU commission meeting.

One of the parameters for loading the tables was the number of record keys required in SIR.
The records had between 0 to 3 keys. For simplicity and speed of processing Tony decided that this element was best catered for by having four different versions of the program, rather than trying to have one generic program to deal with varying numbers of keys.

e.g. RECORD IS <1> (key1, key2)
RECORD IS <2> (key1, key2, key3)

Time constraints meant that some of the processing was best done by manual editing of the spreadsheets (e.g. ensuring that the key fields were at the start of each spreadsheet in the correct order)

The end result was a fairly straightforward PQL program of 168 lines to undertake the data loading from Excel to SIR via ODBC using the VARPUT function.

Lessons learnt

- Generic ODBC didn't work very well because it always produced 255 character strings which were not very helpful for numeric variables
- Exploratory programs were needed to understand the interaction between ODBC and the data files
- Some hand editing of spreadsheets was needed to save time
- The SIR team are available for detailed support if required

Adrian Hodgson

Subroutines of the SYSPROC file and how to use them

The TOOLS and other families in the SYSPROC (sirproc.srp) contain various programs and subroutines that may be useful in your

programming. This article lists some the members available in the SIR/XS sysproc file (though many of them are available in earlier versions) and shows how to use them. The member SYSPROC.TOOLS.ABOUT in the SIR/XS documents all the members in that family.

In most cases the members are subroutine sources (:T) and executable (:O) pairs with the same family and base member name.

Many of these routines are used by the dialogs in the default main menu.

SUBROUTINE TOOLS.BUFDLG
(buffer_name) RETURNING (rc)
Displays the text from a buffer as labels in an infobox style dialog.

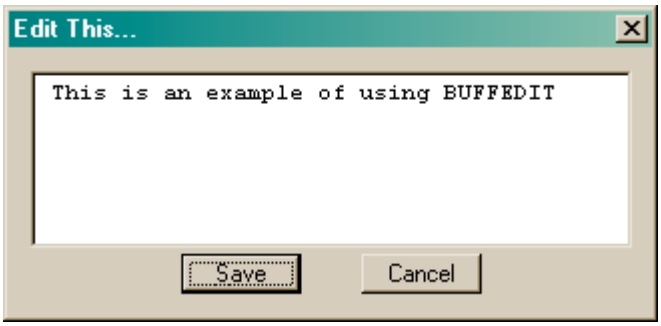
```
PROGRAM
INTEGER RC
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 " This is an example of using"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 2
   FROM
 "    SYSPROC.TOOLS.BUFDLG"
EXECUTE SYSPROC.TOOLS.BUFDLG("MESSAGE")
 RETURNING (RC)
END PROGRAM
```

SUBROUTINE TOOLS.BUFFCOPY
(source, target)
Copies a buffer from source to target, overwriting target if it exists or creating it if it does not.

```
PROGRAM
INTEGER RC
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "This is an example of using BUFFCOPY"
EXECUTE SYSPROC.TOOLS.BUFFCOPY
 ("MESSAGE","NEW MESSAGE")
END PROGRAM
```

SUBROUTINE TOOLS.BUFFEDIT
(title, buffer_name, rows, cols)
Displays a simple buffer editor in a dialog with title and a text box containing the buffer text.

```
PROGRAM
INTEGER RC
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "This is an example of using BUFFEDIT"
EXECUTE SYSPROC.TOOLS.BUFFEDIT
 ("Edit This..." "MESSAGE",4,50)
END PROGRAM
```



SUBROUTINE TOOLS.BUFFFIND
(buffer_name, string)
RETURNING (line)
Finds a given line in a sorted buffer using a binary search - returning line number or zero if not found.

```
PROGRAM
INTEGER RC
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "apples"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 2
   FROM
 "bananas"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 3
   FROM
 "needles"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 4
   FROM
 "pawpaw"
EXECUTE SYSPROC.TOOLS.BUFFFIND
 ("MESSAGE", "needles") RETURNING (RC)
IFTHEN (RC GT 0)
WRITE "Found at " RC
ELSE
WRITE "Not found"
ENDIF
END PROGRAM
```

SUBROUTINE TOOLS.BUFFMAKE
(buffer_name) RETURNING (rc)
Creates the buffer if it does not exist but preserve the buffer and contents if it does. If the buffer is created then the return code is 1.

(If you want to create a buffer and want to clear it if it does exist then use
```
 DELETE BUFFER name
 CREATE BUFFER name
```
As CREATE BUFFER will give an error if the buffer exists but DELETE BUFFER will not give an error if the buffer does not exist.)

```
PROGRAM
INTEGER RC
EXECUTE SYSPROC.TOOLS.BUFFMAKE("MESSAGE")
 RETURNING (RC)
```

```
END PROGRAM
```

SUBROUTINE TOOLS.BUFFSIZE
(buffer_name)
RETURNING (lines)
Returns the number of lines in a buffer using a binary search

```
PROGRAM
INTEGER LINES
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "apples"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 2
   FROM
 "bananas"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 3
   FROM
 "needles"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 4
   FROM
 "pawpaw"
EXECUTE SYSPROC.TOOLS.BUFFSIZE("MESSAGE")
 RETURNING (LINES)
WRITE "MESSAGE is " LINES " lines long"
END PROGRAM
```

SUBROUTINE TOOLS.BUFFSORT
(buffer_name, target, ord)
Sorts a buffer into target with ord being "A" or "D".

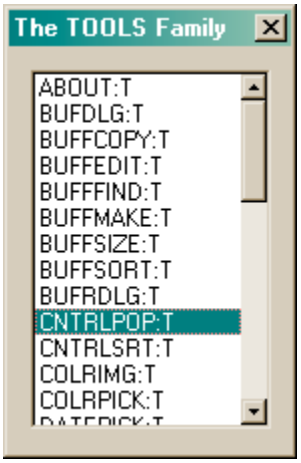
```
PROGRAM
INTEGER LINES
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "pawpaw"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 2
   FROM
 "bananas"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 3
   FROM
 "needles"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 4
   FROM
 "apples"
EXECUTE SYSPROC.TOOLS.BUFFSORT
 ("MESSAGE","NEW MESSAGE","A")
EXECUTE SYSPROC.TOOLS.BUFFEDIT
 ("Sorted","NEW MESSAGE",6,30)
END PROGRAM
```

SUBROUTINE TOOLS.BUFRDLG
(buffer_name) RETURNING (rc)
Displays the text from a buffer in a dialog. This is called the same way as BUFDLG but the display is different and you can use some HTML tags in the buffer source.

```
PROGRAM
INTEGER RC
DELETE BUFFER "MESSAGE"
CREATE BUFFER "MESSAGE"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 1
   FROM
 "<center>An example of using<br>"
PUT LINE TO BUFFER "MESSAGE" NUMBERED 2
   FROM
 "<b>SYSPROC.TOOLS.BUFRDLG<b></center>"
EXECUTE SYSPROC.TOOLS.BUFRDLG("MESSAGE")
 RETURNING (RC)
END PROGRAM
```

SUBROUTINE TOOLS.CNTRLPOP
(id, type, opt1)
RETURNING (num)
CNTRLPOP populates a dialog list or choice control with the named type. OPT1 can contain more information. The TYPE parameter can be one of the following strings:
"DATABASE" - lists connected databases
"RECORD" - Lists records in the default database (if OPT1 is "CIR" then CIR is included)
"VARS" - Variables in record named by OPT1
"CVARS" - Variables including common in record named by OPT1
"INDEXES" - Secondary indexes in record named by OPT1
"TABFILE" - Connected tabfiles
"TABLE" - Tables in the tabfile named by OPT1
"COLUMN" - Columns in the tabfile.table named by OPT1
"TINDEXES" - Indexes on the tabfile.table named by OPT1
"FAMILY" - Families in the default procfile
"MEMBER" - Members in the default family (OPT1 can be any/all the letters TOEV to list specific types of members)
"ATTRIB" - Lists defined file attributes
"ATTRDSN" - Lists filenames from the attributes
"GLOBALN" - Lists defined global names
"GLOBALV" - Lists defined global values
"BUFFERS" - Lists buffers
"BUFFLINE" - Lists the text lines from the buffer named by OPT1

```
SET PROCFILE SYSPROC
SET FAMILY TOOLS
PROGRAM
INTEGER*2 M_ID, M_ARG1, M_ARG2
INTEGER NUM
DIALOG "The TOOLS Family"
LIST 1  ,   0, 8,   0,  80, 0
INITIAL
EXECUTE SYSPROC.TOOLS.CNTRLPOP
 (1,"MEMBER","TE")
 RETURNING (NUM)
END INITIAL
MESSAGE ALL M_ID, M_ARG1, M_ARG2
IF (M_ID EQ 0) EXIT MESSAGE
END MESSAGE
END DIALOG
END PROGRAM
```

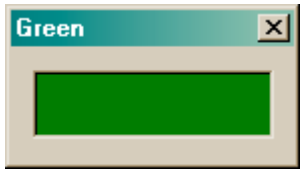


SUBROUTINE TOOLS.CNTRLSRT (id, order)
Sorts the items in a dialog list with control id ID. ORDER is 0 (invert), 1 (ascending) -1 (descending). The current selection is maintained.

```
PROGRAM
INTEGER*2 M_ID, M_ARG1, M_ARG2
INTEGER*1 ID_00001; PRESET ID_00001 (1 )
DIALOG "Sort A Control"
POSTYPE 1
LIST     ID_00001,   0,  86,   0,  80, 0
INITIAL
APPEND ITEM ID_00001,"TESTING"
APPEND ITEM ID_00001,"SORT"
APPEND ITEM ID_00001,"CONTROL"
APPEND ITEM ID_00001,"USING"
APPEND ITEM ID_00001,"A"
APPEND ITEM ID_00001,"TOOLS"
APPEND ITEM ID_00001,"SUBROUTINE"
EXECUTE SYSPROC.TOOLS.CNTRLSRT(ID_00001,1)
END INITIAL
MESSAGE ALL M_ID, M_ARG1, M_ARG2
IF (M_ID EQ 0) EXIT MESSAGE
END MESSAGE
END DIALOG
END PROGRAM
```

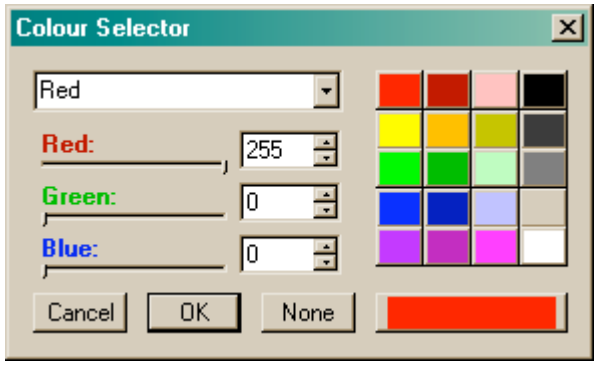
SUBROUTINE TOOLS.COLRIMG
(id, height, width, hexin)
Sets an image (or button in SIR/XS) control to the given hex colour. Height and Width determine the height and width (in pixels) of the bitmap created to display in the image or button.

```
PROGRAM
INTEGER*2 M_ID, M_ARG1, M_ARG2
INTEGER*1 ID_00001; PRESET ID_00001 (1 )
DIALOG "Green"
POSTYPE 1
IMAGE    ID_00001  ,   0,  21,   0,  80, 1
INITIAL
EXECUTE SYSPROC.TOOLS.COLRIMG
 (ID_00001,40,120,"#008000")
END INITIAL
MESSAGE ALL M_ID, M_ARG1, M_ARG2
IF (M_ID EQ 0) EXIT MESSAGE
END MESSAGE
END DIALOG
END PROGRAM
```



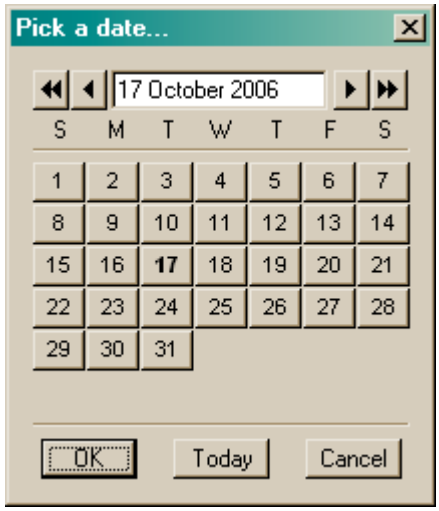
SUBROUTINE TOOLS.COLRPICK
(hexin) RETURNING (hexout)
Displays a small dialog with various methods of selecting a colour. Hexin and Hexout are in the format "#FFFFFF" - a return value of "" indicates "No Colour" was selected. This only works in SIR/XS.

```
PROGRAM
STRING COLOUR
EXECUTE SYSPROC.TOOLS.COLRPICK("#FF0000")
RETURNING (COLOUR)
END PROGRAM
```



SUBROUTINE TOOLS.DATEPICK
(title, default)
RETURNING (selected, rc)
Displays a date selector - pass a title and default date and it returns selected date integer and a return code -1 = cancel; 0 = ok

```
PROGRAM
DATE SELECTED ("DD MMM YYYY")
INTEGER RC
EXECUTE SYEPROC.TOOLS.DATEPICK
 ("Pick A Date...",TODAY(0))
 RETURNING (SELECTED,RC)
END PROGRAM
```

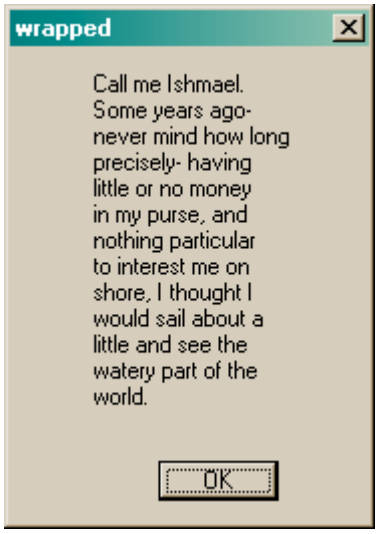


SUBROUTINE TOOLS.FILENAME
(filename)
RETURNING (path, name, ext)
This routine takes a filename and returns parts. If filename is a path it will return the parent directory for the path.

```
PROGRAM
STRING*256 FILENAME PATH NAME
STRING*8   EXT
INTEGER RC
DISPLAY OPENBOX 'Filename',
 'All Files(*.*)|*.*|','',1
 RESPONSE RC,FILENAME
IFTHEN (RC GT 0)
EXECUTE SYSPROC.TOOLS.FILENAME (FILENAME)
 RETURNING (PATH,NAME,EXT)
WRITE PATH / NAME / EXT
ENDIF
END PROGRAM
```

SUBROUTINE TOOLS.LONGWRAP
(long_string, width, buffer)
RETURNING (lines)
Takes a long string and wraps it into a buffer so that no line is longer than the given width. The number of lines in the buffer is returned.

```
PROGRAM
INTEGER RC
STRING*4000 LONGSTR
COMPUTE LONGSTR =
 "Call me Ishmael. Some years ago- never "
 + "mind how long precisely- "
 + "having little or no money in my purse,
   "
 + "and nothing particular "
 + "to interest me on shore, I thought I "
 + "would sail about a little "
 + "and see the watery part of the world."
EXECUTE SYSPROC.TOOLS.BUFFMAKE ("WRAPPED")
 RETURNING (RC)
EXECUTE SYSPROC.TOOLS.LONGWRAP
 (LONGSTR,20,"WRAPPED") RETURNING (RC)
EXECUTE SYSPROC.TOOLS.BUFDLG ("WRAPPED")
 RETURNING (RC)
END PROGRAM
```

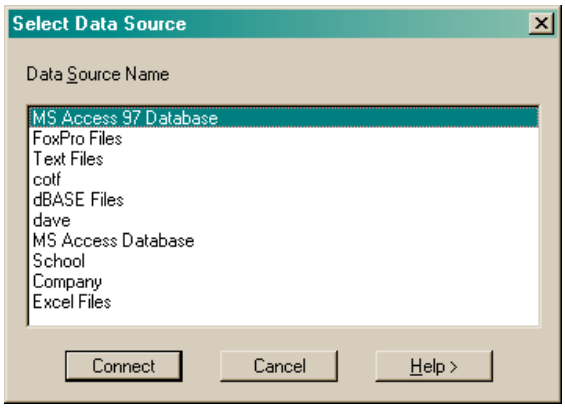


SUBROUTINE TOOLS.NAMEFILE
(name, ext)
RETURNING (filename)
This routine takes a SIR name and extension and returns a valid filename.

```
PROGRAM
INTEGER RC
STRING*256 FILENAME
EXECUTE SYSPROC.TOOLS.NAMEFILE
 ("COMPANY", "exp")
 RETURNING (FILENAME)
WRITE FILENAME
END PROGRAM
```

SUBROUTINE TOOLS.ODBCLIST
RETURNING (database)
Displays a dialog listing the ODBC data sources available and returns the data source name selected.

```
PROGRAM
STRING*80 SOURCE
EXECUTE SYSPROC.TOOLS.ODBCLIST
 RETURNING (SOURCE)
WRITE SOURCE
END PROGRAM
```



SUBROUTINE TOOLS.RECSTAT
(recnum) RETURNING (status)
Returns the record status for RECNUM: status is: -1 = the record does not exist; 0 = the record exists but has no data; 1 = record exists with data; 9 = the record is locked pending restructure.

```
PROGRAM
INTEGER RC
EXECUTE SYSPROC.TOOLS.RECSTAT(2)
 RETURNING (RC)
WRITE "Status = " RC
END PROGRAM
```

SUBROUTINE TOOLS.SCRFILE
(mask)
RETURNING (filename,rc)

This routine returns a non-existent scratch file name based on MASK (or MASK.EXT). The filename will be MASKnnnnn.EXT and will include the temp directory path name if it is

defined. EXT defaults to “tmp” and MASK defaults to “SIR”. You can then open the file for write without overwriting an existing file. RC contains –1 if it cannot find a non-existent filename.

```
PROGRAM
INTEGER RC
STRING*80 SCR
EXECUTE SYSPROC.TOOLS.SCRFILE (“TEST.tmp”)
 RETURNING (SCR,RC)
WRITE SCR
END PROGRAM
```

SUBROUTINE TOOLS.SIRNAME
(name, len)
RETURNING (newname, rc)
Takes a string name and checks if it is a valid sir name. In SIR2002 this would check if the name started with an alphabetic character and contained only alphanumeric and special ($#_@) characters and was no longer than LEN characters. In any case would return a valid name with RC = 0 if it was unchanged.
In SIR/XS the naming rules are more liberal. Names can have any characters but need to be enclosed in curly brackets if they are non-standard names.
- if it is wrapped in quotes or {}s then it is taken literally and put in {}s
- if it contains special characters it is wrapped in {}
- if it contains a mix of case it is wrapped in {}
- if it does not start with an alpha it is wrapped in {}

```
PROGRAM
STRING NEWNAME
EXECUTE SYSPROC.TOOLS.SIRNAME("!^235 Z",8)
 RETURNING (NEWNAME,RC)
WRITE NEWNAME
END PROGRAM
```

SUBROUTINE
SYSPROC.TOOLS.SOUNDEX
(longname)
RETURNING (sound)
Takes a string and returns a short string code. Based on “An Algorithm For Variable Length Proper-Name Compression” James L Dolby Journal Of Library Automation Volume 3/4 December 1970

This is not the "SOUNDEX" algorithm

This might be used with a secondary index on the SOUND of words or names. When a user enters a new name then a list of similar sounding words/names could be displayed.

```
PROGRAM
STRING NAME SOUND
COMPUTE NAME = "CHICKEN SOUP"
EXECUTE SYSPROC.TOOLS.SOUNDEX(NAME)
 RETURNING (SOUND)
WRITE SOUND
COMPUTE NAME = "COKEN SOAP"
EXECUTE SYSPROC.TOOLS.SOUNDEX(NAME)
 RETURNING (SOUND)
WRITE SOUND
END PROGRAM
```

SUBROUTINE TOOLS.VERIFY (patch)
RETURNING (n,c,w)
Verifies the database and return numbers of non-correctable errors, correctable errors and warnings. If patch is 1 the verify first attempts the patch and then does another verify to confirm that it worked.

This is a handy one to run at the start of a batch check and backup process.

```
PROGRAM
INTEGER N C W
EXECUTE SYSPROC.TOOLS.VERIFY (0)
 RETURNING (N,C,W)
IFTHEN (N EQ 0 AND C GT 0)
EXECUTE SYSPROC.TOOLS.VERIFY (1)
 RETURNING (N,C,W)
ENDIF
IFTHEN (SUM(N,C,W) EQ 0)
WRITE "All OK..."
ENDIF
END PROGRAM
```

SUBROUTINE
SYSPROC.MENU.EDITOR
(pqlfile,type)
Starts the “internal” SIR editor on the named source. If type is 1 then the PQLFILE is a file; if type is 2 then it is a member; if type is 3 then PQLFILE is a buffer.

```
PROGRAM
EXECUTE SYSPROC.MENU.EDITOR
 ("SYSPROC.MENU.EDITOR",2)
END PROGRAM
```

SUBROUTINE SYSPROC.TOOLS.SAVENAME
(TYPE,FILENAME)
Stores the filename in a list of recently opened files.

Type is a string naming the type of file. The sir.ini file is checked for a key called "user."+TYPE+"n" who's value is the number of recently opened files to keep.

If this key is not there then it is created with a value of 20. The keys "user."+TYPE+"1" through nn are checked to see if the file is already there and if so it is promoted to number 1, otherwise it is pushed into the number one position and older files shuffle up.

```
PROGRAM
EXECUTE
   SYSPROC.TOOLS.SAVENAME("database",DSN("S
   IR"+FORMAT(SYSTEM(39))+"1"))
END PROGRAM
```

So, as you see there are many potentially useful routines here. Please feel free to use them and

any others not mentioned here. Copy the source code and modify it for your own needs if you wish. The PQL source code is included deliberately and is not subject to copyright.

Note: if you modify the procedures in the sysproc file then you may lose them when a revision is released – or the revised procfile may not overwrite your procfile (if yours is newer) so you will not receive the updated procfile.

David Baxter
SIR Pty Ltd

SIR UK User Group Committee Members 2006/2007

Kathy Brooks (Reporter)
Forvus
53 Clapham Common
South Side
London SW4 9BX
Tel: 020 7819 1012
Fax: 020 7819 1010
email: kathy@forvus.co.uk

Dave Doulton (Chair)
University of Southampton
Computing Services
Highfield
Southampton SO17 1BJ
Tel: 023 8059 3541
Fax: 023 8059 3131
email: D.C.Doulton@soton.ac.uk

Adrian Hodgson
ORC International
5th Floor City Point
701 Chester Road
Stretford
Manchester M32 0RW
Tel: 0161 888 8006
Fax: 0161 872 3997
Email: adrian.hodgson@orc.co.uk

Jon Johnson (Treasurer)
Room 641
Centre for Longitudinal Studies
Institute of Education
20 Bedford Way
London WC1H 0AL
Tel: 020 7612 6571
Fax: 020 7612 6686
Email: jj@cls.ioe.ac.uk

John S. Lemon
Aberdeen University Computing Centre
Edward Wright Building
Dunbar Street
Aberdeen AB24 3QY
Tel: 01224 273350
Fax: 01224 273372
Email: j.s.lemon@abdn.ac.uk

Mo Reardon/Tony Reardon
SIR PTY LTD
312 Mona Vale Road
Terrey Hills
NSW 2084, Australia
Tel: 00612 9450 2354
Fax: 00612 9475 1430
email: mo@sir.com.au
email: tony@sir.com.au

Frances Williams (Secretary)
ISER
University of Essex
Wivenhoe Park
Colchester CO4 3SQ
Tel: 01206 873568
Fax: 01206 873151
email: fwill@essex.ac.uk

You can email the whole committee by addressing your message to **sug.comm@isermail.essex.ac.uk**