CABIOS

Vol 10. no 2. 1994
Pages 105 – 109

Simplified user poll and experience report language (SUPER): implementation and application*

L.Rosenthaler and R.Doelz

Abstract

Biological computing is generally organized as standalone implementation on a PC-type computer or on a central facility (e.g. a university computer center). Services provided by central facilities need to be tailored to the user community. Unless very work-intensive individual contacts are used, the feedback must be collected with generalized tools, such as questionnaires distributed in the form of a newslettter. We have developed a method to have such polls automated and tailored, as well as having multiple-choice questions combined with branching after fundamental questions. As the evaluation of the results needs to know the questions asked, we have also included a method to process the answers and give detailed tables on the answers. SUPER was applied in a poll to query the academic usership in Switzerland on the usage of molecular biology databases.

Introduction

Biological computer applications are frequently called 'unfriendly' or 'difficult'. However, service providers at central facilities are limited in resources at their disposal to provide optimal services. Individual needs of researchers might be met only partially. Feedback obtained from users on a voluntary basis is rarely suited to planning for new investment, improvements or modifications. Any attempt to poll a user community for their profile by sending questionnaires suffers from a return quote of usually $< 10\%$.

In order to achieve better yields in user surveys, methods have been used to query users upon starting the setup of the specific environment. After successfully passing the questions asked, a flag is set to make the routine not show up upon subsequent log-ins. The procedure and programs used for this purpose need to be retaylored for each poll. Error-handling and consistency checking of results (e.g. entering non-numerical or non-existing answers in a mutiple-choice question) is rarely implemented consistently. Therefore, the evaluation of the results of such a poll needs to be done manually in most cases.

Our work aimed at the development of a generally applicable toolset which uses a descriptive language to define the questions, resulting in a Simplified User Poll and Experience Report language (SUPER). After having presented these to the user in consistent fashion, the service provider can use SUPER to perform either a formatting into a spreadsheet format or evaluations can be computed directly.

Systems and methods

The SUPER toolset is written in standard ANSI C language on computers running the Unix or VMS operating system. SUPER can generate C code which also will need a standard C compiler to generate subsequent executables for result

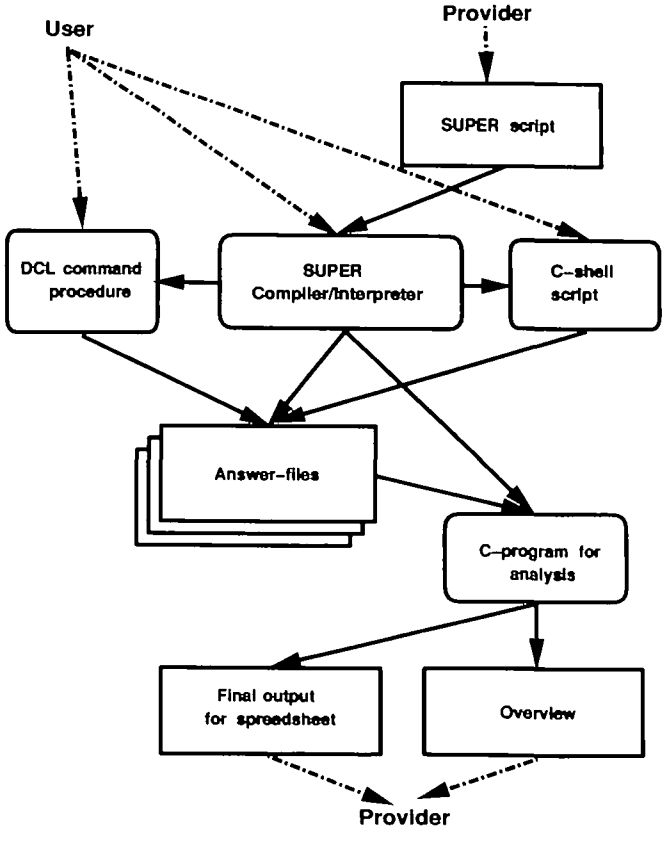

Fig. 1. Operation of the SUPER toolset. The SUPER script (defined by the provider) serves as input for the SUPER executable, which can either be run by the user in order to view the questions directly, or is run by the provider with a special flag to provide either DCL or csh shells. These need to be executed by the user to obtain the questionnaire if a direct run of SUPER is not possible. The output of the polls is collected, and processed by another program which is generated by the provider using SUPER with a special command line switch. The provider will need to evaluate the final output only (as opposed to all individual replies).

Biocomputing, Biozentrum der Universitaet Basel, Switzerland

*This paper was first published in CABIOS 10(1) pp. 35–39 It is republished in full here with a corrected list of authors. The publishers would like to apologise for any inconvenience caused by this printing error.

© Oxford University Press

analysis. Acting as a compiler, SUPER can also generate routines in script languages for Unix (csh) and VMS (DCL). Very little memory and disk is used for the program and the script itself, but, depending on the questions asked, each answer can take up to several kbytes of disk space.

Mode of operation

SUPER is an executable which performs several functions. First, it can be used as an executable on the platform where is was compiled on. Secondly, it can be used as interpreter which generates scripts to be run on other platforms; the options available are DCL command language and C Shell scripts. Finally, SUPER can be used as program generator which writes a C program for result evaluation. The overview of the functionality is presented in Figure 1.

In order to achieve a very general applicability, SUPER uses a very simple template language for generating the poll dialog. The supported data types (see Table I) can be used to supply either a numerical choice or a free-text input. Questions answered by numeric input are processed at the time of the poll dialog for permitted values. Variables can be assigned to individual replies, so that conditional branching can be achieved. Assigned variables can be re-evaluated later in order to implement complex trees of questions.

Due to the distinction of SUPER with respect to variables and actual text presented to the user, it is possible to create a homogeneous output with different scripts written in several languages.

SUPER will create the source code of a C program, which, after compilation, will read the collected output files and create text files suited for spreadsheet program input. Alternatively, human-readable surveys which can be printed.

Table I. SUPER data types

NUMBER	number with digits, but without exponential coefficient
MENU	up to 10 options are supported in a menu
TEXT	a single line of free text; terminated by <RETURN>
FREETEXT	multiple lines of free text; terminated by an empty line finished with <RETURN>
RETURN	receives no value but just a <RETURN>; can be used as pager function

Implementation

Syntax

SUPER statements are to be written in upper-case characters and are listed in Table II. One of the main features of the

Table II. Syntax of the SUPER script language

SUPER prog_id	start of SUPER program, 'prog_id' will be the base nameof the output file which has the form 'prog_id.out'
LABEL label_id [text]	label statement. The 'label_id' is used for GOTO and ON (x) GOTO
[text]	statements. The optional text on the same line is printed as a title
[...]	in underline mode to the screen. The optional text on the following lines will be printed up to the next statement to the screen. Text lines containing only a dot '.' are printed as empty lines.
MENU	the MENU statement builds a menu with item-text numbered
[item-text]	text menu options given on the lines following
[...]	the MENU statement
GET	the GET statements reads input from the user. The user may enter a '?', which gives a short help message, or a '!' to repeat everything from the most recent LABEL statement on, or a '/', 'QUIT' or 'quit' to exit the SUPER session without writing a output file
GET NUMBER	reads a number from the user which has to be between 'min' and 'max'.
(min,max)	Otherwise an error message is written to the screen and the user is asked again
GET MENU	this statement has to follow a MENU statement. It reads the selection from the user and checks if it's valid.
GET TEXT	allows the user to enter a single line of text
GET FREETEXT	allows the user to enter one ore more lines of text. Entering an empty line quits accepting text
GET RETURN	waits for the user to press the return key. Does not write to the output file
ON (*n*) GOTO label_id	jumps to the LABEL given by 'label_id',if the user has selected the menu option *n*.ON (*n*) GOTO statements have to follow immediately a GET MENU statement
GOTO label_id	unconditional jump to the LABEL given by 'label_id'
WRITE [text]	write 'text' (or empty line) into output file
SET varname	sets the variable 'varname' to the value given by 'value'.
{value|$}	If 'value' is the $ character, then 'varname' will be set to the last entry of GET MENU
EVAL varname	this simulates a user input. Following ON (*n*) GOTO statements will act as if the user had entered the value of 'varname'
EXIT	exit the SUPER script, save output file.

SUPER language is that a series of lines will be converted into a menu which is presented to the user in order to pick an option. The validity of the answer entered is checked on-line and requestioned if needed. Subsequent evaluation of the answer permits conditional or non-conditional jumps. It is important to restrict the elements of the language to simple handling in order to permit reliable flow control. The current version of SUPER requests the input of precisely one answer or option (i.e. multiple options cannot be entered).

Format of the output file

SUPER will write human-readable output files, which show the keyword of the question (the 'label' as described in Table II) and the corresponding answer. Free text will be displayed, and menu options will be shown as numerical replies.

Example

The provider who wants to set up a poll will have to design a flow schema in order to present the questions to the user. After having composed a SUPER script using the expressions as defined in the language specification (tables I and II), the SUPER executable is run to generate either C Shell or DCL scripts. An example of such a SUPER script is shown in the Appendix. Typically, a question is formulated as follows:

```
LABEL experience Experiences about Computer
What are your personal experiences with computers?
MENU
   I had only the best experiences, no troubles at all
   I like computers, but sometimes they drive me crazy
   I hate computers, but sometimes they help a little
   I work only with computers because my boss want it
GET MENU
ON (1) GOTO best_exp // we will ask some different
questions
ON (2) GOTO good_exp // option 3 and 4 are treated
equally
```

If invoked without command line switches, SUPER will run as an interpreter:

```
super demo.super
```

If this command is placed in the general setup of the user environment, the dialog will be presented as defined in the SUPER script by the provider. For example, the menu asking the first question in the example of the Appendix will look like:

```
Experiences about Computers
==========================
What are your personal experiences with computers ?
1) I had only the best experiences, no troubles at all
2) I like computers, but sometimes they drive me crazy
3) I hate computers, but sometimes they help a little
4) I work only with computers because my boss wants it
?>
```

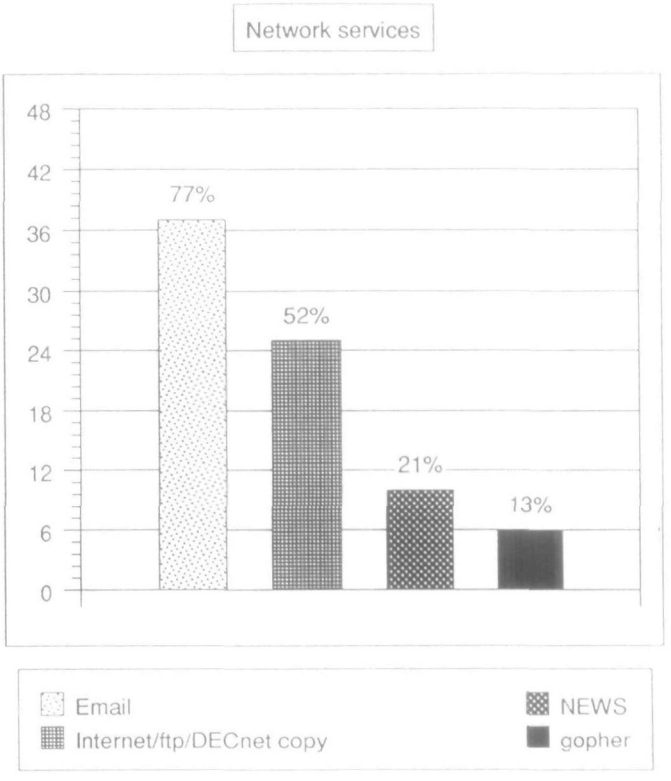

Fig. 2. Usage of network services by molecular biologists in Switzerland. In total, 48 sites using sequence databases were interviewed.

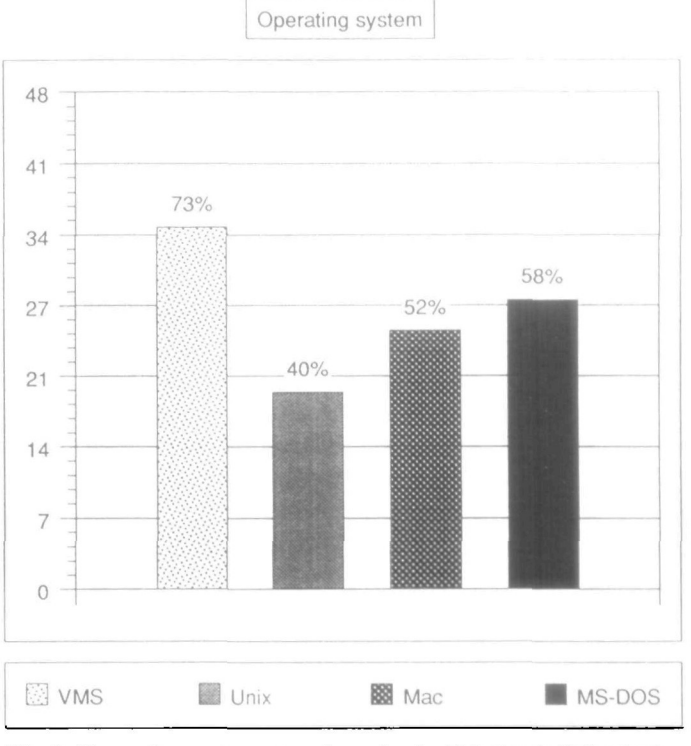

Fig. 3. Usage of computer systems by molecular biologists in Switzerland. Nearly all of the sites had access to either workstation or computing center hardware networked to the international networks. In addition, many sites employed personal computers for various work.

In order to evaluate the results of the poll, SUPER needs to be invoked again. This time, a switch is employed to generate a C source code which resembles the program needed to evaluate the results from the individual SUPER output files:

```
super -rep demo.super
```

The output file will be demo.c, which, after standard compilation, will produce an output suited for processing in spreadsheet data processing programs. Additionally, all menu options will be keyed and evaluated for manual inspection. The first question of the example might look as follows:

```
1: experience → options:
0="invalid" (0),
1="I had only the best experiences, no troubles at all" (2)
2="I like computers, but sometimes they drive me crazy" (1)
3="I hate computers, but sometimes they help a little" (2)
4="I work only with computers because my boss wants it" (3)
```

Application

SUPER has been used already in a variety of applications. For example, it has been utilized as network implementation for users (R.Harper, CSC Helsinki, personal communication). In the following, the application of SUPER for polling the community of Swiss biologists will be shown. SUPER was used on-line as a networked query system within a DCL command procedure, as well as a networked, Unix-based executable. Additionally, individual polls were performed on a person-to-person interview guided by a SUPER poll running on the local system. Due to the flexibility of SUPER, questions were prepared in English, German and French, but resulted in output files which were automatically evaluated due to their identical contents.

In addition to the results obtained from the automatic procedures, addresses for interview partners were obtained by screening electronic phone dictionaries. Most of the persons responsible for biocomputing at the local sites were identified only after several follow-up calls. From >200 telephone calls nearly 50 sites were identified which rely on computational methods and depend on the availability of the databases.

Results from the poll

Nearly all sites allowed for the extensive use of electronic mail (Figure 2). More sophisticated tools of network usage like USENET NEWS and GOPHER are known only to a minority. The computers used for network access, and computational biology, are either mainframes, workstations or personal computers attached to networks using terminal emulation programs (Figure 3). The figures were created with a spreadsheet software. The publication of the detailed results of this poll with respect to the use of sequence databases by biologists is in preparation.

Discussion

SUPER is a program package which can be effectively used in polling user communities for detailed information. It has been successfully used to query the Swiss academic user community on the basis of either automated usage, or individual questions. Besides the benefit of language-specific polls, results obtained by SUPER were found to be more homogeneous than those collected previously using non-standardized procedures. In order to write complex poll scripts which are suited to generate statistical output, the built-in evaluation of SUPER can help in refining and classifying the user community into classes or groups. After an initial poll, SUPER can target specialized user groups more effectively.

We have also used SUPER as a guide in conducting phone interviews. By running the dialog on the screen, an interview can be much more structured and thus more consistent data can be obtained in large surveys.

Availability

SUPER is available by anonymous ftp from the bioftp.unibas.ch file server [131.152.8.1] or the EMBL file server, as well as the mirrors thereof and is in the public domain

Acknowledgements

This work was supported by Basel university and a grant from the Swiss Science Foundation to R.D.

Appendix

```
SUPER sample script
// This is a demo program written in the SUPER script language
LABEL welcome Welcome to SUPER
Hello User, welcome to SUPER. In the following, you will be asked some
questions.
//
// first question
//
LABEL experience Experiences about Computer
  What are your personal experiences with computers ?
MENU
    I had only the best experiences, no troubles at all
    I like computers, but sometimes they drive me crazy
    I hate computers, but sometimes they help a little
    I work only with computers because my boss wants it
GET MENU
ON (1) GOTO best_exp // we will ask some different questions
ON (2) GOTO good_exp // option 3 and 4 are treated equally
// here's the question for the very unlucky user
LABEL very_bad_Exp
  Was in your opinion the training sufficient you received?
MENU
  yes
  no
  don't know
GET MENU
GOTO all_users // No matter what the reply was jump here
// heres the question for the 'normal' user
LABEL good_exp
```

```
  What was your most frustrating experience? You may use several
  lines for description, enter an empty line to proceed.
GET FREETEXT
GOTO all_users
// heres the question for the lucky user
LABEL best_exp
  So far you had only very good experiences (We hope that this will
  not change...). What was your best experience so far?
  You may use several lines for description, an empty line will get
  you to the next question.
GET FREETEXT
LABEL all_users Personal info
  Do you want to enter your name and address or do you prefer to
  be anonymous ?
MENU
  I'm giving name and address
  I want to stay anonymous
GET MENU
ON (2) GOTO stop
// Ask personal question
WRITE
WRITE Personal Information:
WRITE
LABEL name
  Please enter name:
GET TEXT
LABEL street
  Please enter the street you live:
GET TEXT
LABEL city
  Please enter the city:
GET TEXT
LABEL tele
  Please enter your telephone number:
GET TEXT
LABEL stop
  Good Bye

EXIT
```