SOAP—A program which documents and edits ALGOL 60 programs

R. S. Scowen, D. Allin*, A. L. Hillman, and M. Shimell†

Central Computer Unit, National Physical Laboratory, Teddington, Middlesex

This paper describes a program called SOAP (Simplify Obscure Algol Programs) which reads an ALGOL 60 program as data, cleans it up and outputs it in a form which clarifies its structure thus making it easier to understand. The paper states some of the advantages of using SOAP and specifies the layout of an edited program. SOAP is an ALGOL 60 procedure.

(Received March 1970)

1. Introduction

There is no standard which defines the best layout for an ALGOL 60 program. Various journals generally conform to their own standard but no two are the same. SOAP is an ALGOL 60 procedure which reads an ALGOL 60 program as data, cleans it up, and outputs it with a new layout. SOAP thus has the same purpose as the procedure 'Algol edit' (McKeeman, 1965); however, it is much more ambitious and correspondingly larger.

The main advantage of using SOAP to edit an ALGOL program is that it makes it easier for a programmer to examine and follow the program. In fact SOAP edits a program so that many important properties of an ALGOL program can be found by making a scan of the left hand edge of the edited program. Examples of such properties are: assignments to a variable, calls of a procedure, the declaration of an identifier, the position of a label. SOAP achieves this by putting every label and statement on to a separate line. Other properties are indicated by indenting some lines more than others; for example the various parts of a conditional or for statement, the extent of a block or procedure declaration.

Another advantage of using SOAP is that the edited program is known to have a completely consistent style of layout.

2. Method

SOAP makes a single pass through the ALGOL program using input/output routines which read and print an ALGOL basic symbol. Indentation is achieved by outputting either tabulation characters or the corresponding number of spaces.

SOAP decides how to edit by partially analysing the syntax of the ALGOL program. Inside the editor there are separate procedures to edit different syntactic parts of an ALGOL program, e.g. block, statement, specifications or declarations, comments, bracketed structures. These procedures call one another in a mutually recursive way, e.g. while editing a program it is possible that at one point 'block' calls 'statement' calls 'block' calls 'specifications or declarations' calls 'proc declaration' calls 'statement'. The recursive procedures have no parameters by name, and usually have a compound statement (not a block) as body; they communicate their results using a small number of variables global to SOAP. These features simplify a manual rewriting of SOAP into an assembly language, because the recursive procedures can be written as subroutines which stack the return address on entry and unstack it on exit.

The modular structure ensures that SOAP is easy to extend or change.

The methods and techniques used in SOAP were originally developed for the Babel compiler. (Babel is a high-level computer language (Scowen, 1969) and compiler being developed at the National Physical Laboratory.)

3. The properties of an edited program

All editing characters outside strings and comments are removed from the original program and instead SOAP inserts a standard number of spaces before and after each basic symbol as in 'Algol edit' (McKeeman, 1965).

Each statement occupies at least one line and successive statements are indented by the same amount. When any statement cannot be fitted on to one line then the second and subsequent lines are indented so that it is easy to see the extent of the statement. It is often possible to see which procedures are called and which variables are assigned values by scanning the identifiers at the left hand edge of each line and ignoring indented lines.

Example*

```
statement one;
statement two;
a very long statement which has to be indented
    on the second line;
statement three
```

SOAP decides when a line is too long by counting the number of basic symbols. It would be preferable to count the number of characters but there are several reasons why this is not done:

1. The basic output routine used in SOAP outputs one basic symbol.
2. The number of characters equivalent to any basic symbol varies from one output device to another.
3. Basic symbols are machine independent, characters are not.
4. The number of characters on an edited line which are needed for 'n' basic symbols is nearly always less than '$n+20$'. This has proved to be a satisfactory working rule.

A tabulation symbol counts as six basic symbols and a space symbol counts as one basic symbol.

When a line is too long, SOAP breaks it at the last space outside a string whenever possible.

*D. Allin is now at the University of Birmingham
†M. Shimell is now at University College, Oxford

*This example, like all further examples, is not actual SOAP output. Its purpose is merely to clarify the previous text.

SOAP edits programs so that it is easy to find the extent of a block by putting each **begin** and **end** symbol on a line by itself and indenting every declaration and statement of the block.

Example:

```
begin
    declarations;
    statements;
    begin
        declarations;
        more statements
    end;
    yet more statements
end
```

For statements are edited by indenting the controlled statement on a separate line; when there is more than one for list element, the second and subsequent ones are on separate lines. This method of editing implies that a for statement extends up to the first line which is not indented more than the **for** symbol.

Example:

```
for i := for list element do
    for j := for list element,
        element,
        element do
        statement one;
statement two
```

The method of editing conditional statements is most simply described by an example:

```
if condition one then
    statement one
else if condition two then
    statement two
else if condition three then
    statement three
else
    statement four;
statement five;
```

Note that:

1. each statement is indented;
2. each **else** symbol is put under the first **if** symbol;
3. the complete conditional statement extends up to the first line which does not start with **else** and is not indented more than the first **if**. This method of editing a conditional statement makes clear the circumstances in which each statement would be executed, the extent of each statement and the extent of the whole conditional statement.

Conditional expressions which are not in brackets are edited in a similar way.

Labels must be edited so that they are easy to find; SOAP edits a label by putting it on a separate line and indenting less than the following statement, for example:

```
begin
        statement one;
    label one:
        statement two
end
```

SOAP edits declarations so that each variable, array or switch element is on a line by itself. This may seem wasteful of space but has the advantage of making it as easy as possible for the programmer to find the declaration of any entity. It is even simpler if all declarations are put in alphanumeric order. Although SOAP does not put them in order, to have one identifier declared on each line often simplifies the process of changing a program while still preserving the list of identifiers in alphanumeric order. For instance, on KDF9 the editing system works on complete lines; when any line is changed the whole line must be re-typed.

Example:

```
real
    a,
    b;
array
    square one,
    square two [1 : n, 1 : n],
    rectangle [1 : m, 1 : p];
```

Procedure declarations are edited so that the value part, specification part and body are each on separate lines and indented. SOAP also inserts a blank line after each procedure body in order to make the extent of the declaration even clearer.

Example:

```
procedure p(formal);
    value
        formal;
    integer
        formal;
    statement body;

    next declaration
```

Strings and comments are left as the original programmer wrote them, except that the second and subsequent lines are indented. SOAP also emphasises comments by putting a blank line before and after each one.

Example:

```
statement;

    comment this example shows how a comment is edited;

    another statement;
```

There are two exceptions to the above rules:

1. if the indented statement of a conditional or for statement is a block or compound statement, then the declarations and statements are not indented further; and
2. a dummy statement before **end** is not put on a separate line.

Example:

```
if condition then
    begin
    statement;
    ;
    statement;
    end
```

4. Extensions

The version of SOAP described above deals with pure ALGOL 60 and it is written in ALGOL 60. For any practical compiler and system it is necessary to put the procedure SOAP into a program, and then to modify and extend it in order to cope with the vagaries of the actual system.

At the National Physical Laboratory the main computer is an ICL KDF9. The Whetstone and Kidsgrove ALGOL compilers on this computer deal with almost complete subsets of ALGOL 60 which are practically identical; in addition there are facilities for code bodies, insertion of library text, segmentation of programs, and also for storing the text of a program on a disc file.

The first extension of SOAP is called SOAP-A and copes with complete KDF9 ALGOL. The ALGOL program to be edited can be input either from 8-hole KDF9 paper tape or from the disc (if the program is in the 'Prompt' operating system). The output is either on to the line printer or on paper tape; in the latter case the edited program can be read back into the 'Prompt' operating system.

A small change is made in the actual editing; a semicolon is inserted after a label whenever possible. This is to overcome

a peculiarity of 'Prompt' and ensure that the layout is preserved if the ALGOL program is re-established.

SOAP-A has itself been extended into SOAP-C which is designed to document an ALGOL program for publication purposes. It splits the program into pages with a specified number of lines on each page. SOAP-C also starts a new page whenever a procedure body has ended near the bottom of a page. Each page is given a heading containing a page number and the identifier of the program; the heading is also designed to make it easy to count the number of tabs starting each line. In addition SOAP-C puts a line number at the start of every line. The final difference from SOAP-A is to add suitable **end**-comments at the end of procedure bodies and the controlled statements of for loops; no **end**-comment is inserted if one is already present.

At the end of the program text SOAP-C prints a list of all the uses and declarations of every identifier in the program. This list is sorted (Boothroyd, 1967) into alphanumeric order of the identifiers.

SOAP-D is another extension of SOAP-A; this version enables any identifier in the program to be systematically changed everywhere. Before editing the ALGOL program SOAP-D reads a list of the identifiers to be changed and the new identifiers which will replace them.

A limitation which has not proved onerous is that if an identifier is declared more than once in a program, SOAP-D cannot be used to change one usage but not the others.

5. Practical uses of SOAP

These various versions have been used in a number of ways. The most obvious and common use is clarifying an ALGOL program by giving it a consistent layout. This is especially useful for large untidy programs which have been written and amended over a long period by many different people, and for ALGOL programs which have been generated by another program.

While SOAP-C can be used to document a program for publication, another use is to aid the analysis of a program. The use of SOAP-C makes it easy to find every call of any procedure, every use and assignment of any variable and all the goto statements to any label. The information obtained may be used to debug, shorten or speed up the program.

Probably the most straightforward use of SOAP-D is to expand a programmer's incomprehensible abbreviation, although it has other uses as well. When a program is to be transferred to a version of ALGOL which only has identifiers in small letters, SOAP-D can be used to convert all identifiers containing capital letters. SOAP-D has also been used to aid the conversion of an ALGOL program into KDF9 Usercode (the assembly language): each ALGOL identifier was replaced by the equivalent Usercode instruction. SOAP-D could also be used to convert the identifiers of a French ALGOL program into the equivalent English identifiers.

6. Technical details

The declaration of SOAP is:

procedure *soap* (*linelimit*, *id*, *od*, *failure*);
value *linelimit*, *id*, *od*;
integer *linelimit*, *id*, *od*; **label** *failure*;
comment *The meaning of the parameters is:*
linelimit – the maximum number of basic symbols to be output on any one line
id – the Input Device from which the Algol 60 *program is read*
od – the Output Device to which the edited program is to be sent
failure – a label to which SOAP will jump if anything goes wrong;

The body of SOAP is about 900 lines of ALGOL and is too large to be printed in full; a listing of SOAP-A is given in Scowen *et al.*

The main reason for SOAP jumping to the label *failure* is because the value of *linelimit* is too small; other possible causes are:

1. SOAP detecting an error in the ALGOL program, or
2. a machine error causing an internal check inside SOAP to fail.

When SOAP is run using the KDF9 Kidsgrove ALGOL compiler, it edits and outputs programs at about 600 lines per minute. The size of the program is about 1,600 words plus another 540 words of Input/Output routines.

The work described above has been carried out at the National Physical Laboratory.

References

BOOTHROYD, J. (1967). Algorithm 26, Order the subscripts of an array section according to the magnitudes of the elements, *The Computer Journal*, Vol. 10, p. 308.

MCKEEMAN, W. M. (1965). Algorithm 268, ALGOL 60 reference language editor, *Comm. ACM*, Vol. 8, p. 667.

SCOWEN, R. S., HILLMAN, A. L., and SHIMELL, M. (1969). SOAP—Simplify Obscure Algol Programs, *NPL CCU Report* No. 6.

SCOWEN, R. S. (1969). Babel, a new general programming language, *NPL CCU Report* No. 7.