PARKER84

J.R. Parker A Practical Examination of Prefixed Procedures INFOR, August 1984

The idea of prefixed procedures was originally introduced in the article 'Prefixed Procedures: A Structuring Concept for Operations' by J.G. Vaucher (INFOR, October 1975). That article presented what was essentially an implementation independent view, while in this paper, Parker addresses some of the issues involved in the implementation of prefixed procedures.

It is not necessary to have had previous exposure to the concept of prefixing, since Parker begins with a brief tutorial, illustrating the basic prefixing mechanism by example. Consider an arbitrary procedure P as follows:

```
procedure P (argsP); locsP; /* local variables */ begin
    code1P; /* initial code */ inner; code2P; /* terminating code */ end;
```

Define a prefixed procedure Q that uses procedure P as a prefix:

```
prefix P procedure Q (argsQ); locsQ; begin codeQ;
end;
```

The effect is as if the procedure Q had been defined as:

```
procedure Q (argsP, argsQ); locsP; locsQ; begin
    code1P; codeQ; code2P; end;
```

The procedure P has been treated like a template. The arguments and the local of Q have been concatenated with those of P. The Q is to be placed.

While it is useful to think of procedure P as a template, it is important to realize that it is just a procedure and can be called directly. In this case the 'inner' statement behaves like a NOP.

The goals of prefixing can be stated as follows:

(i) Provide a bracketing facility to enhance the structure and reliability of operations that are implemented as procedures.

(II) Eliminate duplication of code when many procedures have similar specifications.

It should be noted that the goals of prefixing can, to a large extent, be realized by using an ADA-like generic package facility. Parker himself mentions this near the end of the article.

Before discussing the implementation of prefixed procedures, Parker develops some rules for prefixing. He also considers the following basic design issues:

(i) Nesting - Prefixed procedures can be nested. The prefix chain for a procedure is defined as the list of prefix identifiers (i.e. procedures) applied to the procedure.

(ii) Recursion - A prefixed procedure is recursively defined if it appears in its own prefix chain. This is not allowed.

(iii) Scope - A procedure may be used as a prefix if it is defined in an enclosing block of the prefixed procedure.

Two approaches to the implementation of prefixed procedures are discussed.

The first approach is a straightforward pre-processor implementation. Parker discusses a pre-processor developed for Pascal. The primary drawbacks to this approach were that it added significant overhead to the compilation process and that it placed restrictions on some of the nesting and scope rules.

The second approach was to include prefixed procedures as part of a new language. Because the prefixing of procedures is a static activity, it is possible to create a table of entry points at compile time. When a prefixed procedure is entered, this table (called the inner table) is included in the activation record and used to determine the execution path of the procedure. Parker gives a clear explanation as to how the inner table is used to pass control to the various procedures in the prefix chain and how control is returned. He does not however, discuss how the procedures reference their share of the concatenated parameter list.

The idea of prefixed procedures is an interesting one and Parker's article provides a good example of the process of taking a new language facility from theory to implementation.