**Astradyne (UK) Ltd**
4 Knoll Road, Fleet, Hampshire. GU51 4PR

Tel: 01252 400955 Fax:01252 400956
Email: info@astradyne-uk.com

# Technical Document

**Document No:** 07-02003
**Document Title:** Named Indicators in RPG IV / ILE
**Category:** Hints, Tips & FAQ
**Functional Area:** Programming
**OS/400 Release:**

## Document Description:

There is a school of thought out there in the AS400, Iseries and i5 world that is of the opinion that RPG programmers should move away from the use of indicators. Why, for example, when there are built-in functions in the modern RPG language – such as %Found and %Error – that can be used in place of indicators are indicators still used.

One of the reasons, and a fairly major reason, is the use of display files. Although RPG has advanced over the years, the Data Definition Specifications (DDS) used for display files has remained fairly static and although you can code around the use of response indicators for Function Keys on a display file, you are still forced to use indicators to condition the highlighting of fields in error, or for subfile keywords in particular.

So how can we make the use of indicators in our RPG program more user friendly? The answer is to use named indicators to make the program more readily understandable. To do this you need to do two things in your RPG Definition specification:

1. Define a pointer to contain the memory address of the indicator (*INxx) storage area, and
2. Define a 99 character data structure that is based on the pointer previously defined.

The following code snippet shows an example of this:

```
     ** ----------------------------------------------------------
     ** D A T A   S T R U C T U R E S
     ** ----------------------------------------------------------

      * Named Indicators
     D IndPtr          S               *   Inz(%Addr(*In))
     D IndAra          DS             99    Based(IndPtr)
     D  F3                             3     3
     D  F5                             5     5
     D  F12                           12    12
     D  SflDsp                        72    72
     D  SflEnd                        73    73
```

Individual indicators are named according to their positions within the data structure so, for example, indicator *IN03 would be found in position 3, *IN12 in position 12 and so on. By making use of named constants and work fields, you can then make your RPG program easier to read:

```
     ** ------------------------------------------------------------
     ** G E N E R A L   W O R K   F I E L D S
     ** ------------------------------------------------------------

     D Yes             c                   Const('1')
     D No              c                   Const('0')
     D True            s                1  Inz(Yes)
     D False           s                1  Inz(No)

     D Pressed         s                1  Inz(No)
```

…

```
     C                   Select
     C                   When      F3 = Pressed
     C                   ...
     C                   When      F5 = Pressed
     C                   ...
     C                   When      F12 = Pressed
     C                   ...
     C                   Other
     C                   ...
     C                   EndSl
...
...
     C                   Eval      SflDsp = Yes
```

It is far more beneficial to the programmer to be able to understand code like this, than the same thing written using indicators:

```
     C                   Select
     C                   When      *IN03 = *ON
     C                   ...
     C                   When      *IN05 = *ON
     C                   ...
     C                   When      *IN12 = *ON
     C                   ...
     C                   Other
     C                   ...
     C                   EndSl
...
...
     C                   Eval      *IN72 = *ON
```

** End of Document **