CERES Software Bulletin 95-02

CERES Routine Prologue, May 15, 1995

1.0 Purpose:
This bulletin communicates the approved standard prologue which is required to appear at the beginning of each CERES software routine. This standard is based on EOSDIS document 423-16-01, “Data Production Software and Science Computing Facility (SCF) Standards and Guidelines”, January 14, 1994 and “CERES Software Coding Guidelines”, Appendix B, November, 1994, modified May 3, 1995.

2.0 Originator/DMO Approval:
Maria Mitchum (M.V.Mitchum@larc.nasa.gov).

3.0 Description:

CERES Routine Prologue

Modification to CERES Software Coding Guidelines, Appendix B
as approved by CERES DMT, May 3, 1995

Routine Name (with or without parameters)

```
!**************************************************************
! !F90
!  Name:
!
! !Description:
!  Routine ID: (start with subsystem ID)
!
!  Purpose:
!
! !Input Parameters:
!
!  Input/Output Parameters:
!
! !Output Parameters:
!
! !Revision History:
!
! !Team-Unique Header:      (optional)
!
```

```
!  External Files:
!
!  Discussion of Complex Algorithms:
!
!  Implementation/Unique Features:
!
!  Optional Headings: (Called Modules, Global Variables, Local Variables, Constants,
!                               Error Handling, Comments)
!
! !end
!*****************************************************************
                                    <code follows here>
```

Note:
All headings in bold large script (**!Command**) are mandatory headings as issued by ESDIS requirements. This is a FORTRAN 90 example where the first ! is the symbol for a comment; not to be confused with the mandantory !Command that is to appear in all languages.
All other headings are CERES project-specific mandatory headings, where applicable.
If any of the headings under !Team-Unique Header are applicable, then the words '!Team-Unique Header:' must appear in the prologue.
As a general procedure, start with routine (include file, subroutine, module, function, etc.) name and parameters, if they exist, followed by the prologue.

!F90
Start of prolog. Initial marker can take the following values:
- !Fxy - contains FORTRANxy (xy = 77 or 90) executable statements
- !C   - contains C executable statements
- !ADA - contains Ada executable statements
- !Fxy-INC - FORTRANxy (xy = 77 or 90) include file
- !C-INC   - C include file
- !ADA-INC - Ada include file

Name:
Name of routine or include file. Include full argument list, if applicable.

!Description:
Any references for methods and/or algorithms should be included. Use as many lines as necessary.

Routine ID:
Start with the subsystem ID, followed by a dash (-), followed by any logical numbering scheme designed by the subsystem; for example:
Subsystem source code: Subsystem 7.2 should use: 7.2-x.y.etc
General shared routines (within a subsystem): G.7.2-x.y.etc
General cereslib shared routines: GC- x.y.etc, where x.y.etc., are assigned by the cereslib committee

Purpose:

A concise but complete summary of the overall function of the module.

!Input Parameters:
Input Parameters, in the order they are presented to the module, with a short 1-2 line description of the parameter; include: units, optional feature, value ranges where appropriate.

Input/Output Parameters:
Same format as for Input Parameters.

!Output Parameters:
Same format as for Input Parameters.

!Revision History:
If you are using an automated tool for revision control, you should insert any statements required immediately after the Modification History Log Header.
Each revision should contain, as a minimum, the revision number, date, time, person, and e-mail address, with a short description of ALL changes made. The first revision should include the original author of the code. Revisions should be ordered with the latest first. [Note: this revision information can be supplemented with more detailed comments in the code referencing the revision number].

!Team-Unique Header: **(mandatory if any of the following headings exist)**

External Files:
List all External Files such as input files, output files, their purpose and origin.

Discussion of Complex Algorithms:
In addition to mentioning them in the description, Complex Algorithms should be discussed in some detail here.

Implementation/Unique Features:
Where an algorithm has been implemented in a controversial (as opposed to non-portable) manner and Unique Features should appear here.

Optional Headings: (Called Modules, Global Variables, Local Variables, Constants,
Error Handling, Comments)
All of these headings (and more) are left up to the discretion of the developer and are optional. It is expected that if, for instance, significant error traps are in the routine, they should be documented in the prologue as ‘Error Handling’.

!end
End of source code prolog.

<u>Example Prologue in C:</u>

Routine Name (with or without parameters)

```
/*
!C
Name:
!Description:
Routine ID:
Purpose:
!Input Parameters:
Input/Output Parameters:
!Output Parameters:
!Revision History:
!Team-Unique Header:
External files:
Discussion of Complex Algorithms:
Implementation/Unique Features:
Optional Headings: (Called Modules, Global Variables, Local Variables, Constants,
                    Error Handling)
!end
*/
                         <code follows here>
```

<u>Example Prologue in Ada:</u>

```
------------------------------------------------------------------------
--!ADA
--Name
--!Description:
--Routine ID:
--Purpose:
--!Input Parameters:
--Input/Output Parameters:
--!Output Parameters:
--!Revision History:
--!Team-Unique Header:
--External files:
--Discussion of Complex Algorithms:
--Implementation/Unique Features:
--Optional Headings: (Called Modules, Global Variables, Local Variables, Constants,
--                    Error Handling)
--!end
-----------------------------------------------------------------------
                         <code follows here>
```