NAME

issue - Issue File

DESCRIPTION

The issue file, issue, resides in an appropriate /type?? directory for each switching machine (SPCS) that the SCCS supports. This file contains information for each SPCS generic and issue that is supported.

The issue file contains one or more generic-issue messages. Each generic-issue message begins with a message delimiting character whose present value is defined in the header file, issfil.h. Basically, a generic-issue message lists all of the SPCS issues that are officially supported for a specific SPCS generic. Thus, each generic-issue message consists of one generic record followed by one or more issue records that are supported for this generic.

All generic records are fixed size and have the structure **IF_GENREC**, as defined in the header file, issfil.h. All information in a generic record is in ASCII; hence all elements are defined as character strings. Each generic record must be initialized to contain blanks in all elements or unused portions of elements that do not contain data. In general, data in a generic record is left-justified in each element and leading zeroes appear where necessary, such as the generic ID, to insure the required precision.

All issue records are fixed size and have the structure **IF_ISSREC**, as defined in the header file, issfil.h. All information in an issue record is in ASCII; hence all elements are defined as character strings. Each issue record must be initialized to contain blanks in all elements or unused portions of elements that do not contain data. In general, data in an issue record is left-justified in each element.

The following is a listing of the issfil.h header file.

```
/*
   Header file to define the layout of the issue file that resides
   in the appropriate /type?? directory.
*/


/* Define name of issue file. */

#define ISS_FIL "issue"


/*
    Define valid return codes for the
```

```
    library subroutine GEN_LIST().
*/

#define GLR_NME 0       /* No more entries exist */
#define GLR_ERR -1      /* Error detected */


/*
    Define valid function codes and return codes for the
    library subroutine GEN_NAME().
*/

#define GNF_GNAM 0      /*  Extract generic name */
#define GNF_SLANG 1     /*  Extract generic slang name */

#define GNR_EF 1        /* Entry found */
#define GNR_ENF 0       /* Requested entry does not exist */
#define GNR_ERR -1      /* Error detected */


/*
    Define valid function codes and return codes for the
    library subroutine GET_GEN().
*/

#define GGF_GNAM 0      /*
                            Use generic name as the generic
                            search key.
                        */
#define GGF_SLANG 1     /*
                            Use generic slang name as the
                            generic search key.
                        */
#define GGF_GID 2       /*
                            Use generic ID as the generic
                            search key.
                        */

#define GGR_ENF 0       /* Requested entry not found */
#define GGR_ERR -1      /* Error detected */


/*
    Define valid return code for the
    library subroutine GET_ISS().
*/

#define GIR_ENF 0       /* Requested entry not found */
```

```
/*
    Define valid return code for the
    library subroutine ISS_LIST().
*/

#define ILR_NME 0       /* No more entries exist */


/*
    Declare types of values returned by library subroutines.
*/

char *gen_list();
char *get_gen();
char *get_iss();
char *iss_list();


/* Define array sizes for structures IF_GENREC and IF_ISSREC */

#define IF_ENDSZ 2
#define IF_GFILLSZ 2
#define IF_GNAMSZ 12
#define IF_SLGSZ 8
#define IF_GIDSZ 6
#define IF_IFILLSZ 6
#define IF_INAMSZ 8
#define IF_IOPSYSZ 8


/* Define ending sequence for each entry in the ISS_FIL file */

#define IF_ENDSEQ "*0


/*
   Define generic-issue message delimiter; this is
   used by the GTMSG subroutine to extract generic-
   issue messages from an "issue" file.  A generic-
   issue message consists of one generic record fol-
   lowed by one or more issue records that are
   associated with the generic record.
*/

#define IF_GIMSG_DLM 03       /* Generic-issue message delimiter */


/* Define a union for an entry in the ISS_FIL file */

union IF_REC
```

```
{
    char *if_recptr;                /* Pointer to start of record */
    struct IF_GENREC *if_genrec;    /* Pointer to generic record */
    struct IF_ISSREC *if_issrec;    /* Pointer to issue record */
};

/* Define structure of a generic entry in the ISS_FIL file */

struct IF_GENREC
{
    char if_gfill[IF_GFILLSZ];
                        /*
                           Record type and white space.
                           See Note 1.
                        */

    char if_gnam[IF_GNAMSZ];
                        /*
                           Generic name, see Note 2.
                        */

    char if_gslang[IF_SLGSZ];
                        /*
                           Generic slang name, see Note 3.
                        */

    char if_gid[IF_GIDSZ];
                        /*
                           Generic ID, see Note 4.
                        */

    char if_gend[IF_ENDSZ];
                        /*
                           End sequence.
                        */
};


/* Define structure of an issue entry in the ISS_FIL file */

struct IF_ISSREC
{
    char if_ifill[IF_IFILLSZ];
                        /*
                           Record type and white space.
                           See Note 1.
                        */

    char if_inam[IF_INAMSZ];
                        /*
```

```
                            Issue and point-issue name, see
                            Note 5.
                         */

    char if_iopsys[IF_IOPSYSZ];
                         /*
                            Operating system generic issue and
                            point-issue, see Note 6.
                         */

    char if_iend[IF_ENDSZ];
                         /*
                            End sequence.
                         */
};


/* Define array sizes for structure GEN_ID (generic ID) */

#define GID_BASESZ 2
#define GID_INFOSZ 4


/* Define structure of generic ID field */

struct GEN_ID
{
    char gid_base[GID_BASESZ];          /* Generic base - see Note 4. */
    char gid_info[GID_INFOSZ];
                         /*
                            Generic information - see
                            Note 4.
                         */
};


/*
NOTES:

    1.  The fill field contains the generic-issue message delimiter
        and/or white space.  This white space must contain only
        blank characters (040);  tabs are not permitted.  This
        requirement exists so that all entries of the same record
        type will be fixed length records.

        The fill field for a generic record contains the generic-
        issue message delimiter, which is a single-character code
        that must be left-justified in the fill field.  The value
        of this message delimiter is defined elsewhere in this
        header file.  The remainder of the fill field must be padded
        on the right with blanks.
```

The fill field for an issue record contains the specified number of blanks.

2. The generic name is the official generic name that has been assigned by the appropriate SPCS development group; this name is not necessarily the PG number. For example, in ESS1A the official name for a generic might be 1AE(C2B4), while in ESS101 the official name might be PG-1H002. The generic name must be left-justified in this field and padded on the right with blanks.

3. The generic slang name is an abbreviated name that is sometimes used in place of the official generic name. Examples are 1E3, 1AE4, etc. The generic slang name, if needed, must be left-justified in this field and padded on the right with blanks. If the slang name is not needed, then this field must be filled with IF_SLGSZ blanks.

4. The generic ID is a two-to-five digit decimal number that uniquely identifies a particular SPCS generic. The format of a generic ID is

bb[xyz]

where bb is a two-digit decimal number that identifies a generic base. Examples are:

11 for ESS1, generic 1e3
12 for ESS1, generic 1e4

xyz is an optional one-to-three digit decimal number that provides additional information that is needed for some SPCS types to uniquely identify a specific generic. Examples are:

ESS1:
xyz is a single-digit decimal number that identifies the central processor configuration. A value of 0 identifies those systems that have only a CC; whereas, a value of 1 identifies those systems that have both a CC and a SP. Thus, the generic ID for a 1e3 system having only a CC is 110 and the generic ID for a 1e4 system having both a CC and a SP is 121.

EPSCS, E911, TN, VSS:
xyz is a three-digit number that identifies the generic issue and point issue of the operating system that is used in the auxiliary processor.

5. The issue name contains the official issue and point issue that have been assigned by the appropriate development group. Examples are 3.1, 6a.3, and 10c.14, where all

characters to the left of the "." identify the issue and those characters to the right of the "." identify the point issue. The issue name must be left-justified in this field and padded on the right with blanks.

6. The operating system generic issue and point-issue are primarily used for the Auxillary Processor systems that the SCCS supports. It identifies which issue and point-issue of the auxillary processor operating system is being used for the application, such as TN, E911, and VSS.

*/

FILES

/type??/issue	Data File
/usr/include/issfil.h	Header File