Consultative Committee for Space Data Systems

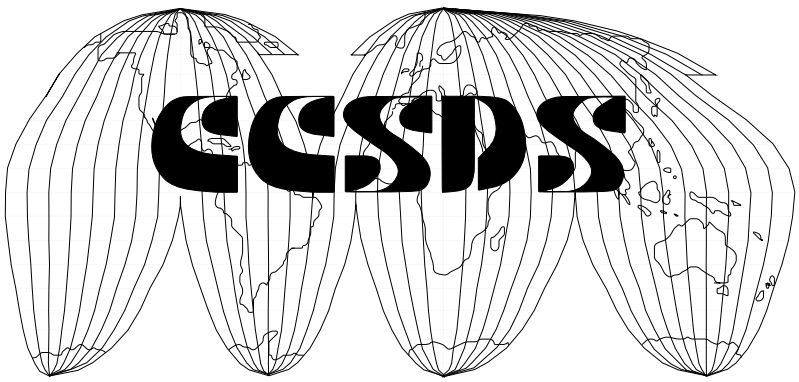
RECOMMENDATION FOR SPACE DATA SYSTEM STANDARDS

STANDARD FORMATTED DATA UNITS — REFERENCING ENVIRONMENT

CCSDS 622.0-B-1

BLUE BOOK

May 1997



AUTHORITY

Issue:	Blue Book, Issue 1
Date:	May 1997
Location:	São José dos Campos São Paulo, Brazil

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS Recommendations is detailed in reference [B1], and the record of agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the address below.

This document is published and maintained by:

CCSDS Secretariat
Program Integration Division (Code MG)
National Aeronautics and Space Administration
Washington, DC 20546, USA

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of member space Agencies. The Committee meets periodically to address data systems problems that are common to all participants and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommendations** and are not considered binding on any Agency.

This **Recommendation** is issued by, and represents the consensus of, the CCSDS Plenary body. Agency endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- Whenever an Agency establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommendation**. Establishing such a **standard** does not preclude other provisions which an Agency may develop.

- Whenever an Agency establishes a CCSDS-related **standard**, the Agency will provide other CCSDS Member Agencies with the following information:

  — The **standard** itself.

  — The anticipated date of initial operational capability.

  — The anticipated duration of operational service.

- Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommendation** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommendation** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or, (3) be retired or canceled.

In those instances when a new version of a **Recommendation** is issued, existing CCSDS-related Agency **standards** and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each Agency to determine when such **standards** or implementations are to be modified. Each Agency is, however, strongly encouraged to direct planning for its new implementations towards the later version of the **Recommendation**.

FOREWORD

This Recommendation extends the standardization of the Standard Formatted Data Unit (SFDU) concept in support of the digital transfer of space-related information. This Recommendation defines valid CCSDS Referencing Environments and provides syntax specifications for expressing file names within those Referencing Environments.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommendation is therefore subject to CCSDS document management and change control procedures which are defined in Reference [B1]. Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/ccsds/

Questions relating to the contents or status of this Recommendation should be addressed to the CCSDS Secretariat at the address on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- British National Space Centre (BNSC)/United Kingdom.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- Deutsche Forschungsanstalt für Luft- und Raumfahrt e.V. (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- National Aeronautics and Space Administration (NASA)/USA.
- National Space Development Agency of Japan (NASDA)/Japan.
- Russian Space Agency (RSA)/Russian Federation.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- Centro Tecnico Aeroespacial (CTA)/Brazil.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- Communications Research Laboratory (CRL)/Japan.
- Danish Space Research Institute (DSRI)/Denmark.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Federal Service of Scientific, Technical & Cultural Affairs (FSST&CA)/Belgium.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Industry Canada/Communications Research Centre (CRC)/Canada.
- Institute of Space and Astronautical Science (ISAS)/Japan.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- MIKOMTEK: CSIR (CSIR)/Republic of South Africa.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- National Oceanic & Atmospheric Administration (NOAA)/USA.
- National Space Program Office (NSPO)/Taipei.
- Swedish Space Corporation (SSC)/Sweden.
- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	Title	Date	Status/ Remarks
CCSDS 622.0-B-1	Recommendation for Space Data Systems Standards: Standard Formatted Data Units - Referencing Environment, Issue 1	May 1997	Original Issue

CONTENTS

CONTENTS (continued)

Figure Page

Table

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

The purpose of this Recommendation is to define a number of valid CCSDS Referencing Environments to facilitate open system data interchange. Each Referencing Environment is defined in terms of a syntax specification for referencing external objects within each of these environments.

1.2 APPLICABILITY

This Recommendation applies to the use of Standard Formatted Data Unit (SFDU) Replacement Service Objects, i.e. Label Value Objects (LVOs) with Class ID = R. The use of CCSDS Referencing Environments within these objects is required for open system data interchange.

1.3 RATIONALE

The Consultative Committee for Space Data Systems (CCSDS) defined the SFDU concept for the implementation of standard data structures to be used for the interchange of data within and among space agencies. The format of these objects are defined in the SFDU - Structure and Construction Rules Recommendation (Reference [1]).

When SFDU products are generated that reference data objects that are not located in the immediate sequence of octets, they make use of Replacement Service Objects, i.e., Label Value Objects with Class ID = R. The CCSDS realizes it is essential to have at least a small number of well known referencing environments defined to support open system data interchange.

1.4 DOCUMENT STRUCTURE

This document is structured as follows:

- Section 1 contains introductory material;
- Section 2 is an overview of this Recommendation;
- Section 3 contains the specifications of the CCSDS Referencing Environment;
- Section 4 defines the conformance level of this specification;
- Annex A presents the acronyms and abbreviations used in this document;
- Annex B provides an informative reference list;
- Annex C provides a tutorial on the use of the CCSDS Referencing Environments;
- Annex D provides information on the prototype $CCSDS3 referencing environment;

A proper understanding of this document requires familiarity with the SFDU concept, the Parameter Value Language (PVL) used to construct statements, and the specific terminology used in this document. If the reader does not have this familiarity, it is recommended that the documents identified in Annex B - Informative References List be read prior to reading this Recommendation. Users may also find it useful to peruse the documents listed in 1.6. This reference list contains documents whose provisions are required for implementing this Recommendation. Annex C - Referencing Environment Tutorial - will be useful to users and implementers of this specification.

1.5 DEFINITIONS

1.5.1 TERMS

For the purposes of this document, the following definitions apply.

Authority and Description Identifier (ADID): The concatenation of the Control Authority Identifier (CAID) and the Data Description Identifier (DDID).

Control Authority (CA): An organization under the auspices of CCSDS which supports the transfer and usage of SFDUs by providing operational services of registration, archiving and dissemination of data descriptions. It comprises:

a) The CCSDS Secretariat supported by the Control Authority Agent (CA Agent); and

b) Member Agency Control Authority Offices (MACAOs).

Control Authority Identifier (CAID): A four character restricted ASCII string that identifies an individual Member Agency Control Authority Office (MACAO) or the CCSDS Secretariat.

Data Description Identifier (DDID): A four character restricted ASCII string, assigned by a Member Agency Control Authority Office (MACAO) or the CCSDS, to distinguish among descriptions with the same Control Authority Identifier (CAID).

Label Value Object (LVO): The basic SFDU building block comprised of a LABEL field and a VALUE field. This structure is the fundamental structural element used to build SFDUs.

Member Agency Control Authority Office (MACAO): An individual CCSDS Participating Agency organization that has accepted the operational responsibilities and constraints specified within CCSDS Recommendations on Control Authority (CA) operations.

Open System Data Interchange: The process of transferring data from one open system to another. An open system is one which uses publicly available formats and protocols, so that anyone can communicate with the open system by following the open system standards. It should be noted that open system does not imply an uncontrolled or unrestricted access to the data.

Standard Formatted Data Unit (SFDU): Data that conform to CCSDS SFDU Recommendations for structure, construction rules, and field specification definition.

1.5.2 NOMENCLATURE

The following conventions apply throughout this Recommendation:

a) the words 'shall' and 'must' imply a binding and verifiable specification;

b) the word 'should' implies an optional, but desirable, specification;

c) the word 'may' implies an optional specification;

d) the words 'is', 'are', and 'will' imply statements of fact.

1.5.3 CONVENTIONS

This document uses syntax diagrams to illustrate the syntax of the Reference Environment Name Specifications. The following conventions are used.

a) The item on the left of the := symbol is being defined.

b) The diagram on the right of the := symbol is the definition.

c) A vertical branch represents a choice.

d) A repetition is indicated by a loop backwards covering the object(s) that may be repeated.

1.6 NORMATIVE REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Recommendation are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS Recommendations.

[1] *Standard Formatted Data Units—Structure and Construction Rules.* Recommendation for Space Data Systems Standards, CCSDS 620.0-B-2. Blue Book. Issue 2. CCSDS, May 1992 or later issue.

[2] *Parameter Value Language Specification (CCSD0006).* Recommendation for Space Data Systems Standards, CCSDS 641.0-B-1. Blue Book. Issue 1. CCSDS, May 1992 or later issue.

2 OVERVIEW

The SFDU concept was developed to support information interchange between and within space agencies. Under the SFDU concept, methods have been developed for packaging data and metadata in various data objects to preserve information content. Any type of data can be integrated into the SFDU domain; what is standardized is the technique of packaging together the various data objects into a logical SFDU data product.

Data product producers may wish to include data objects in a product even if those units are not stored contiguously with the rest of the product. Within the SFDU domain, this technique is called the "Replacement Service." This technique makes use of an LVO with Class ID = R as described in the SFDU-Structure and Construction Rules Recommendation (Reference [1]).

As one method of implementing Replacement Services, Reference [1] includes the specification for ADID = CCSD0003. The CCSD0003 specification describes a simple LVO Value field for specifying parameters for use by the CCSDS replacement services provided by an LVO with Class ID = R. One of the PVL keywords defined for CCSD0003 is REFERENCETYPE, which defines a referencing environment to be used. Another PVL keyword defined for CCSD0003 is REFERENCE, which provides for specifying a number of data objects within the current referencing environment that are external to the current object. The third and final keyword is LABEL, which specifies the label which will be associated with the external data objects.

This Recommendation specifies a number of referencing environments that can be used in the context of Replacement Services LVOs. This Recommendation also defines the syntax to be used for specifying data objects within each of these referencing environments. Additionally, this recommendation supplies values for the REFERENCETYPE and REFERENCE statements. The LABEL statement values are not addressed by this Recommendation.

3 REFERENCING ENVIRONMENTS SPECIFICATIONS

3.1 BASIC REFERENCING ENVIRONMENT - CCSDS1

This specification maps directly to all current popular filenaming specifications. External data object names within the **CCSDS1** referencing environment are composed of a filename, preceded by an optional directory specification and followed by an optional extension specification, as shown in Figure 3-1.

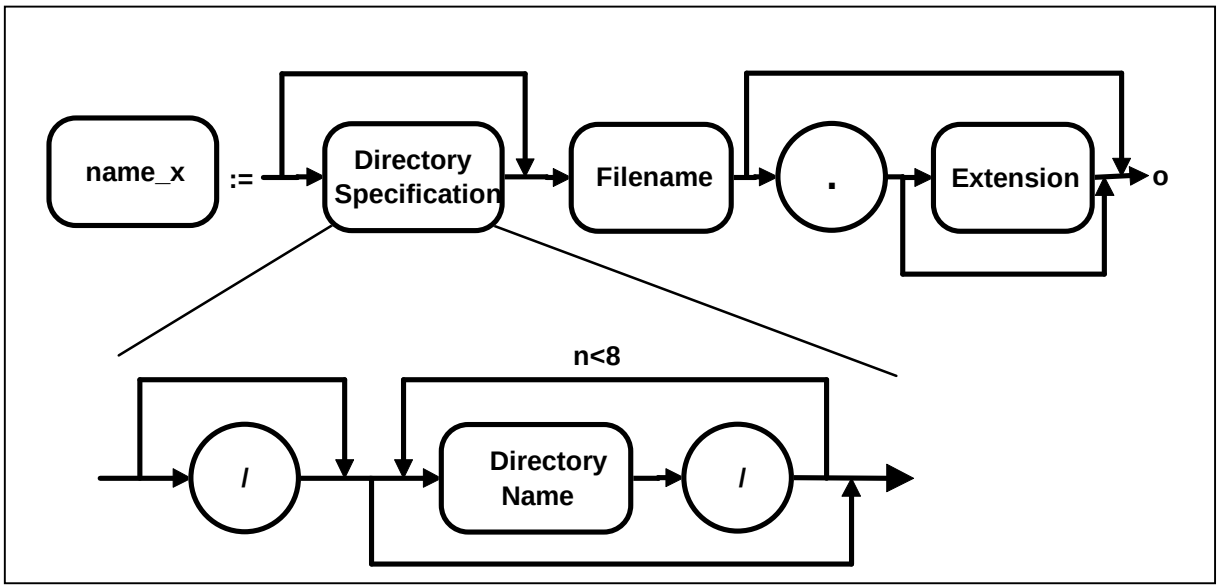


Figure 3-1: Structure Diagram of CCSDS1 Name Specification

The following rules apply:

a) The characters permitted in each field of the specification are "A" to "Z", "0" to "9" and "_" (underscore). Additional rules also permit the wildcards "*" (asterisk) and "?" (question mark) in some locations to represent one of these permitted **CCSDS1** characters.

b) Each **Directory Name** and the **Filename** is a string of 1 to 8 permitted **CCSDS1** characters, while **Extension** is a string of 1 to 3 permitted **CCSDS1** characters.

c) A maximum of eight directory names are allowed.

d) If there is no initial "/" (forward slash), the directory specification is relative to the directory that contains the Referencing LVO. If an initial "/" is present, the directory specification is interpreted as if it is preceded by the directory path needed to reach the highest level directory of the mounted volume or file system which contains the Referencing LVO.

NOTE 1 - A file that is more than eight directory levels deep may be referenced when specified by a relative reference, i.e., if the first character of the directory name is not a "/" (forward slash).

e) The **CCSDS1** specification provides two wildcards: "*" (asterisk) and "?" (question mark). A wildcard is permitted in the **Filename** and/or **Extension** fields. The presence of a wildcard indicates that the field is replaced by the sequence of names found in the referenced environment which match the specified pattern. The order of the external data objects matched is not specified. If the order is significant, then wildcards should not be used.

f) The "?" (question mark) wildcard will match any single **CCSDS1** character in the same position in a **Filename** and/or **Extension**.

g) The "*" (asterisk) wildcard will match a string of zero or more **CCSDS1** characters in a **Filename** and/or **Extension** starting from the position of the "*". Note that the "*" character will match the separator between the **Filename** and **Extension**. The asterisk character must be the last character in the **Filename** and/or **Extension**.

h) If the period separating the **Filename** and **Extension** is not included, the specification is equivalent to specifying the **Filename** followed by a period and a blank **Extension**.

NOTE 2 - Any filename defined using the prototype $CCSDS1 referencing environment can be interpreted in the same manner as one defined using the CCSDS1 referencing environment.

3.2 EXTENDED REFERENCING ENVIRONMENT - CCSDS2

This specification provides extensions to the **CCSDS1** specification for users who find the **CCSDS1** specification too restrictive. It maps directly to a number of major file-naming specifications, but is less portable than **CCSDS1**. External data object names within the **CCSDS2** referencing environment are comprised of a filename, preceded by an optional directory specification, as shown in Figure 3-2. A complete directory specification and filename taken as a unit is termed a **Pathname**.

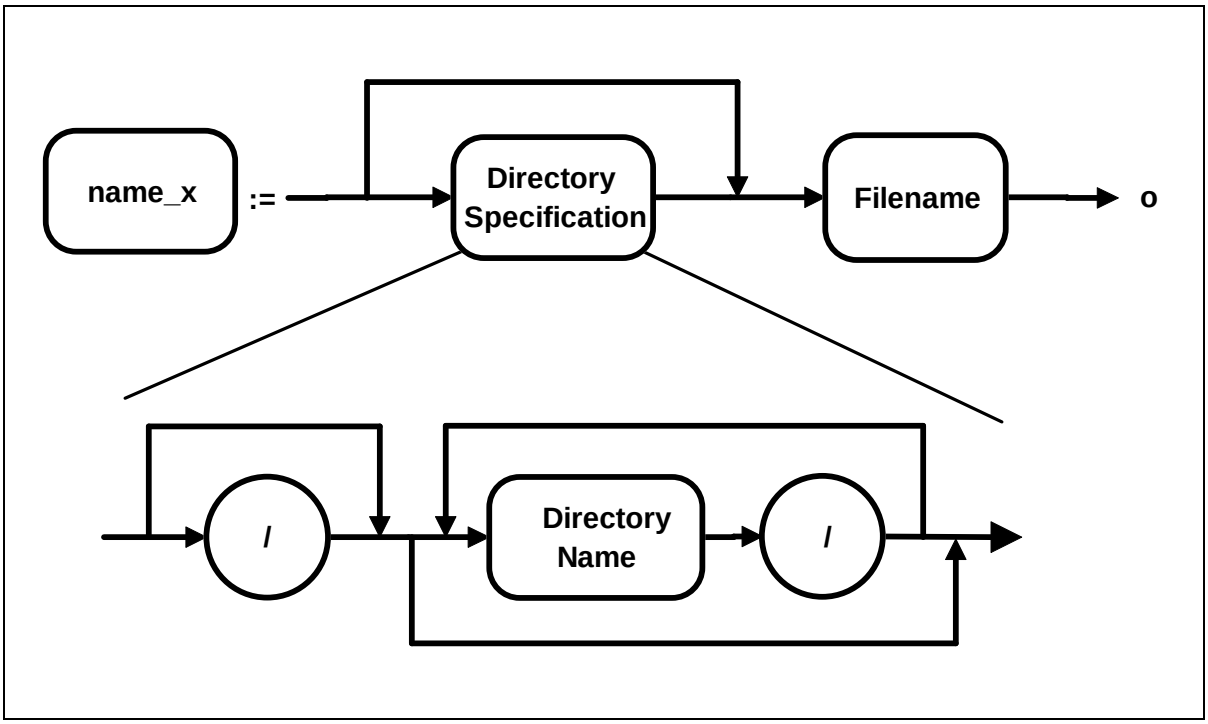


Figure 3-2: Structure Diagram of CCSDS2 Name Specification

The following rules apply:

a) The characters permitted in each **Directory Name** and **Filename** field of the specification are "A" to "Z", "a" to "z", "0" to "9", "." (period or full-stop), "-" (hyphen or dash), and "_" (underscore). Additional rules also permit the wildcards "*" (asterisk) and "?" (question mark) in some locations to represent one of these permitted **CCSDS2** characters.

b) A **Directory Name** or **Filename** must not begin with a "-" (hyphen or dash).

c) The total length of **Pathname** cannot exceed 255 characters.

d) Each **Directory Name** and **Filename** is a string of any number of characters, including any number of periods or full stops, as long as the total **Pathname** length is not exceeded.

e) Any number of **Directory Names** are allowed as long as the overall **Pathname** length is not exceeded.

f) If there is no initial "/" (forward slash), the directory specification is relative to the directory that contains the Referencing LVO. If an initial "/" is present, the directory specification is interpreted as if it is preceded by the directory path needed to reach the highest level directory of the mounted volume or file system which contains the Referencing LVO.

g) The **CCSDS2** specification provides two wildcards: "*" (asterisk) and "?" (question mark). A wildcard is permitted only in the **Filename** field. The presence of a wildcard indicates that the field is replaced by the sequence of names found in the reference environment which match the specified pattern. The order of the external data objects matched is not specified. If the order is significant then wildcards should not be used.

h) The "?" (question mark) wildcard will match any single **CCSDS2** character in the same position in a **Filename** field.

i) The "*" (asterisk) wildcard will match a string of zero or more **CCSDS2** characters in a **Filename** starting from the position of the "*". Note that the "*" character will match the "." (period or full stop). The asterisk character must be the last character in the **Filename**.

NOTE 1 - The character set in CCSDS2 corresponds to a portable POSIX filename character set as specified in the POSIX standard (reference [B2]).

NOTE 2 - Any filename defined using the prototype $CCSDS2 referencing environment can be interpreted in the same manner as one defined using the CCSDS2 referencing environment.

3.3 SEQUENTIAL MEDIA REFERENCING ENVIRONMENT - CCSDS3

This specification provides a method for specifying files on sequential media when the underlying protocol does not associate a name with the file. External data object names within the **CCSDS3** referencing environment consist of an optionally signed, ASCII-formatted decimal integer. An absolute reference to an external object is represented by an unsigned decimal integer consisting of up to 10 digits. The absolute reference corresponds to the position of the file on the volume. The first file on the volume is expressed as "1", the second as "2", and so forth. Alternatively, a file may be referenced relative to the current file by including a sign as part of the **File_Number**. The relative reference is represented by a signed decimal integer of up to 10 digits. The values of -0, 0, and +0 are invalid in the **CCSDS3** Referencing Environment. The **CCSDS3** specification is shown in Figure 3-3.

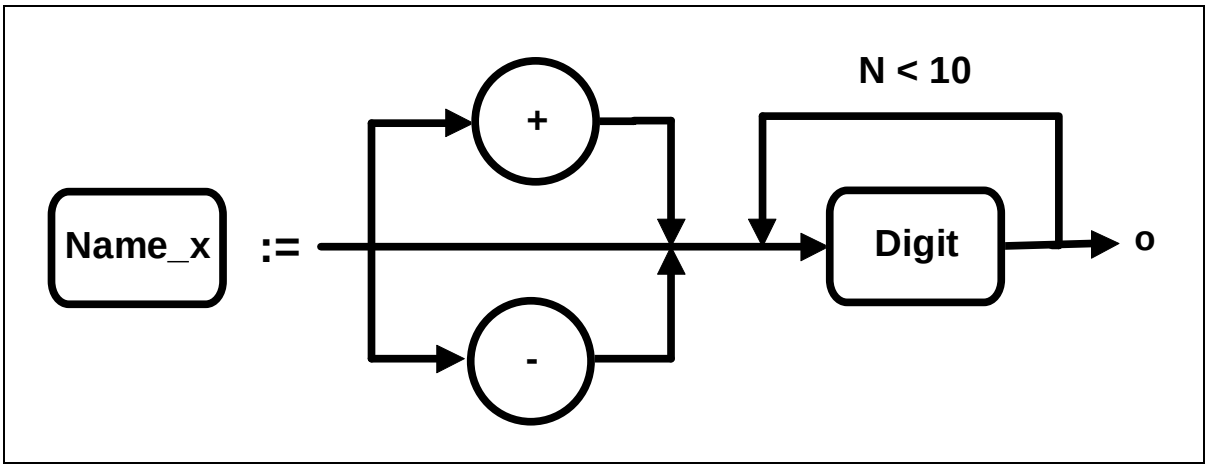


Figure 3-3: Structure Diagram of CCSDS3 Name Specification

The file being specified is understood to consist of a sequence of octets followed by a media defined end-of-file mark.

NOTE - For example, sequential media to which ISO standard labels have been added will store a single logical file as three files of which only the middle file is available for user data. Each of these three files would be included in the file count using this referencing environment.

3.4 COMBINED REFERENCING ENVIRONMENT - CCSDS0

This specification combines the filenaming specifications for any number of referencing environments into one construct. This allows the user to define references which take advantage of each referencing environment while avoiding the need to rewrite the referencing object when it is moved from one environment to another.

External data object names within the **CCSDS0** referencing environment are specified as a sequence of one or more Tagged Names, the sequence being represented as a PVL sequence of strings as defined in the Parameter Value Language Recommendation (Reference [2]). A PVL sequence is represented by including the sequence in parenthesis and separating sequence items with commas. Each Tagged Name is a PVL quoted string composed of a Reference Tag, followed by an equal sign (=), followed by a Reference Name that is legal in the referencing environment indicated by the Reference Tag, e.g., `"CCSDS1=filename.ext"`. A Reference Tag is the name of the corresponding CCSDS referencing environment (e.g., a **CCSDS1** tag indicates that the name conforms to the rules for a **CCSDS1** name (see 3.1), while a **CCSDS2** tag indicates that the name conforms to the rules for a **CCSDS2** name (see 3.2), etc.). A Reference Tag for any particular referencing environment may appear at most once. Wildcards are not permitted in the Reference Names.

A structure diagram for the sequence is given in Figure 3-4. Each of the elements in this diagram may be separated by White Space as defined in the Parameter Value Language Recommendation (Reference [2]); that is one or more of any of the ASCII characters - space, horizontal tab, vertical tab, line feed, form feed, and carriage return.

The structure diagrams and formation rules for the Reference Name are the same as in the specification of the reference environment specified by the Reference Tag, except for the fact that in all cases wildcarding is not permitted.

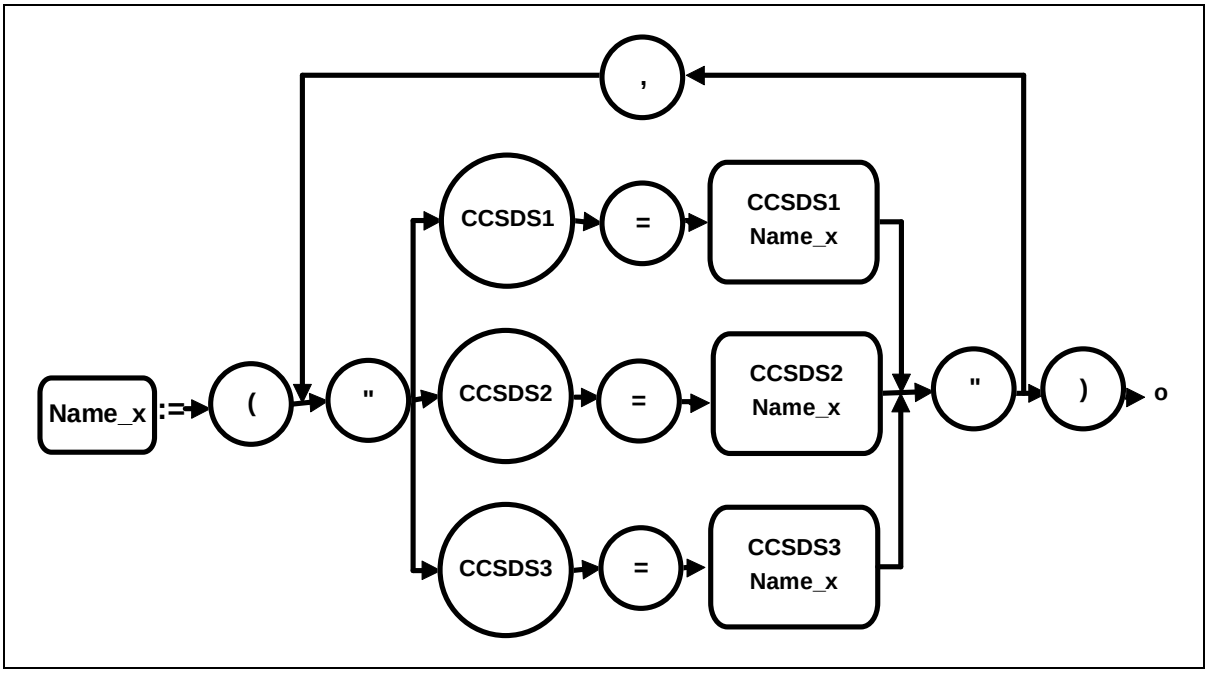


Figure 3-4: Structure Diagram of CCSDS0 Name Specification

As a number of possible names are specified for this environment, a convention for usage of the names is required. This convention is a follows; the referencing object shall first try the first referencing environment listed in the sequence; if this fails (e.g., the **CCSDS2** Tagged Name will fail if the local environment is unable to support the **CCSDS2** names, or if the **CCSDS2** name does not exist even though the local environment may support the **CCSDS2** name format.), then the next Tagged Name shall be used, and so forth.

NOTE - Although the prototype $CCSDS3 referencing environment and the CCSDS0 referencing environment serve similar functions and appear similar, their respective formats are sufficiently different such that a $CCSDS3 specification cannot be directly interpreted as a CCSDS0 specification. See Annex D for details.

4 CONFORMANCE

Data conforming to a Recommendation may be said to be in conformance at some identified level. Identifying conformance levels provides a standard way to classify the required capabilities of generating and receiving systems.

The Reference Environment Recommendation recognizes only one conformance level, and that is the entire specification. Therefore recipient systems which are said to be in conformance to this Recommendation shall recognize the entire specification.

ANNEX A

ACRONYMS AND ABBREVIATIONS

(This annex **is not** part of the Recommendation.)

Purpose:

This annex defines the acronyms and abbreviations which are used throughout this Recommendation to describe the concepts and elements of the Standard Formatted Data Units - Referencing Environment.

Term	Meaning
ADID	Authority and Description Identifier
ASCII	American Standard Code for Information Interchange
CA	Control Authority
CAID	Control Authority Identifier
CCSDS	Consultative Committee for Space Data Systems
DDID	Data Description Identifier
DDU	Description Data Unit
EDU	Exchange Data Unit
ISO	International Organization for Standardization
LVO	Label Value Object
MACAO	Member Agency Control Authority Office
PVL	Parameter Value Language
SFDU	Standard Formatted Data Unit

ANNEX B

INFORMATIVE REFERENCES

(This annex **is not** part of the Recommendation.)

Purpose:

This annex provides a list of references which may be valuable to the user of this Recommendation as background material or to provide implementation guidelines for using this Recommendation.

[B1] *Procedures Manual for the Consultative Committee for Space Data Systems.* CCSDS A00.0-Y-6. Yellow Book. Issue 6. CCSDS, May 1994 or later issue.

[B2] *Portable Operating System Interchange for Computer Environments.* ISO/IEC 9945-1:1990. Institute of Electrical and Electronic Engineers, 1990.

[B3] *Standard Formatted Data Units—A Tutorial*. Report Concerning Space Data Systems Standards, CCSDS 621.0-G-1. Green Book. Issue 1. Washington, D.C.: CCSDS, May 1992 or later issue.

[B4] *Parameter Value Language—A Tutorial.* Report Concerning Space Data Systems Standards, CCSDS 641.0-G-1. Green Book. Issue 1. Washington, D.C.: CCSDS, May 1992 or later issue.

ANNEX C

REFERENCING ENVIRONMENT TUTORIAL

(This annex **is not** part of the Recommendation.)

Purpose:

This annex provides tutorial material to aid implementers and users to correctly apply this Recommendation.

C.1 REFERENCING TECHNIQUES - THE REPLACEMENT SERVICE

The Standard Formatted Data Unit (SFDU) concept was developed to support information interchange between and within space agencies. Under the SFDU concept, methods have been developed for packaging data and metadata in various data objects to preserve information content. Any type of data can be integrated into the SFDU domain; what is standardized is the technique of packaging together the various data objects into a logical SFDU data product.

The basic SFDU building block is comprised of a LABEL field and a VALUE field and is referred to as a Label Value Object (LVO). This structure is the fundamental structural element used to build SFDUs. The VALUE field may contain any form of data that can be described by a user. The method used to delimit the VALUE field and the description of the VALUE field are identified through the associated LABEL field.

LVOs may be either a simple LVO, which consists of a single LABEL field with an associated VALUE field, or a compound LVO, which consists of a LABEL field indicating that the VALUE field is composed of one or more other LVOs. To facilitate recognition of SFDU data products, legal SFDUs always start out with a compound LVO logically containing the entire SFDU data product. This outermost compound LVO is known as an Exchange Data Unit (EDU). Thus to be a legal SFDU data product, the first LABEL field must have a Class ID = Z, which indicates that the LVO is an EDU, and an Authority and Description Identifier (ADID) = CCSD0001, which identifies the format of the VALUE field as being composed of one or more LVOs.

Occasionally, data product producers may wish to include data objects in a product even if those units are not stored contiguously with the rest of the product. Within the SFDU domain, this technique is called the "Replacement Service", because it allows logical inclusion of physically separate data objects (e.g., files) into SFDU products in place of the object providing the Replacement Service. The Replacement Service technique makes use of a Label Value Object (LVO) with Class ID = R. The general concept is shown in Figure C-1, where the LVO with Class ID = R is envelope packaged in an EDU. The VALUE field of the LVO with Class ID = R then points to the referenced file.

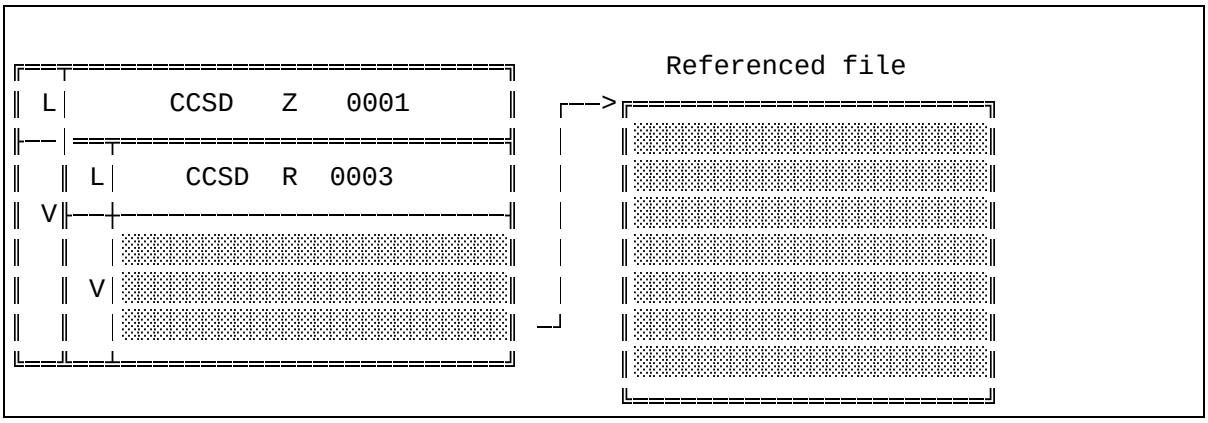


Figure C-1: The Replacement Service - Referencing a File

C.2 REFERENCING TECHNIQUES - THE CCSD0003 LVO

As one method of providing the Replacement Services, CCSDS has included the specification for ADID = CCSD0003 in the SFDU-Structure and Construction Rules Recommendation (Reference [1]). Other methods may also be defined by either the CCSDS or by users.

The CCSD0003 specification describes a simple LVO Value field for specifying parameters for use by the CCSDS replacement services provided by an LVO with Class ID = R. The Parameter Value Language (PVL) (see Reference [2]) is used to specify the parameters in a CCSD0003 VALUE field. CCSD0003 defines three parameters - REFERENCETYPE, LABEL, and REFERENCE.

The permitted ordering of these three statements in a VALUE field of an LVO with ADID= CCSD0003 is as depicted in Figure C-2.

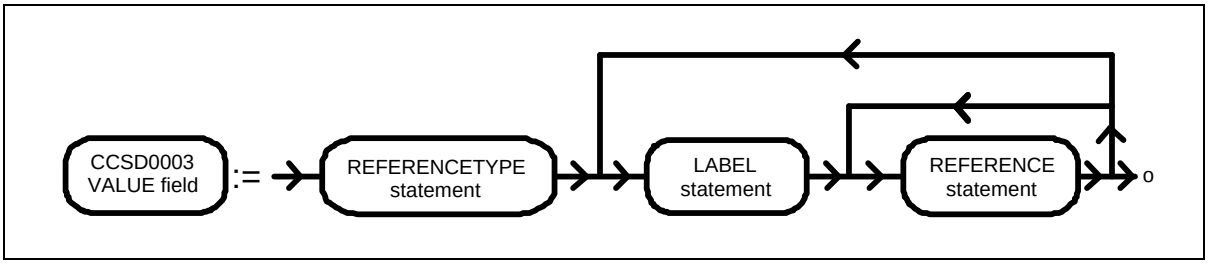


Figure C-2: Structure Diagram of the PVL Statements within an LVO with ADID = CCSD0003

The following should be noted concerning Figure C-2:

a) There shall be one, and only one, **REFERENCETYPE** statement. This shall be the first statement.

b) There shall be at least one **LABEL** statement.

c) There shall be at least one **REFERENCE** statement.

d) There shall be one, and only one, **LABEL** statement before the first **REFERENCE** statement.

e) There shall be no more than one **LABEL** statement between any two consecutive **REFERENCE** statements.

f) A **LABEL** statement can apply to many **REFERENCE** statements. The most recent **LABEL** statement will be used.

g) The last statement in a VALUE field shall be a **REFERENCE** statement.

Three parameters (**REFERENCETYPE**, **LABEL** and **REFERENCE**) are specified by the CCSD0003 ADID. They perform the following three tasks:

a) Define the name (**REFERENCETYPE**) of a referencing environment that is to be used to access external data object(s) from the Referencing LVO, for example, the filenaming convention and the type of data storage medium.

b) Provide a means to optionally add an LVO LABEL (**LABEL**) to external data object(s).

c) Provide a pointer (**REFERENCE**) to the start of external data object(s) within the specified referencing environment.

These statements are defined below where any parameter names or values shown in upper case are keywords, while lower case is used to indicate user specified values. All keywords in a CCSD0003 VALUE field must be expressed in upper case.

a) **REFERENCETYPE=refenv;**

The **refenv** value names the referencing environment to be used. If **refenv** begins with the **$** character then it names a provisional CCSDS defined referencing environment otherwise it names an approved CCSDS defined referencing environment.

The convention of using a **$** as the first character of this name is intended to allow: The use of **$CCSDS1** - **$CCSDSn** to indicate that the referencing environment to be used is a proposed CCSDS developed referencing environment. Three such proposed referencing environments - $CCSDS1, $CCSDS2, and $CCSDS3 - were defined. The intention was that when such a proposed CCSDS defined referencing environment has been fully tested and evaluated, it will be incorporated into a CCSDS Recommendation then its name will not have a **$** at the beginning.

This Recommendation describes the approved CCSDS defined referencing environments. CCSDS has currently defined three base referencing environments - CCSDS1, CCSDS2, and CCSDS3 -and one additional referencing environment - CCSDS0 - to allow data product designers to customize their product for several different environments by specification of references in more than one referencing environment.

b) **LABEL=string;**

The **string** value allows an LVO LABEL field to be logically added to the beginning of the external data object(s). If **string** is the keyword **ATTACHED** then it means the external data object(s) shall have an LVO LABEL at the beginning and do not require a further LABEL. Otherwise, **string** shall conform to an LVO LABEL specification that is composed of printable ASCII characters (decimal codes 32 to 126). Although this does not strictly forbid delimitation by binary length, it is considered extremely bad form and use of delimitation by binary length is strongly discouraged.

c) **REFERENCE=(name_1, name_2, . . . name_n);**

Each **name_x** specifies the beginning of one or more external data objects within the defined referencing environment. The parentheses are optional if there is only one **name_x.** This Recommendation defines the syntax to be used for specifying data object names (i.e. **name_x**) within each of these referencing environments.

A fully compliant CCSDS parser for a VALUE field with ADID = CCSD0003 must be able to understand a **REFERENCETYPE** statement with a value of any of the approved CCSDS referencing environments. The fully compliant parser must then correctly interpret the **name_x** values given in the **REFERENCE** statement and access the necessary external data objects. It is possible, but not required, that parsers will also recognize a number of the CCSDS proposed or locally proposed referencing environments.

The general concept is displayed in Figure C-3. The following observations can be made about this example:

- The LVO with Class ID=R is packaged inside an EDU.
- The REFERENCETYPE statement indicates the use of the CCSDS1 Referencing Environment.
- The LABEL statement indicates that the referenced file includes the label.
- The REFERENCE statement references the file called **CCSDIMG** containing an LVO (both LABEL and VALUE fields).

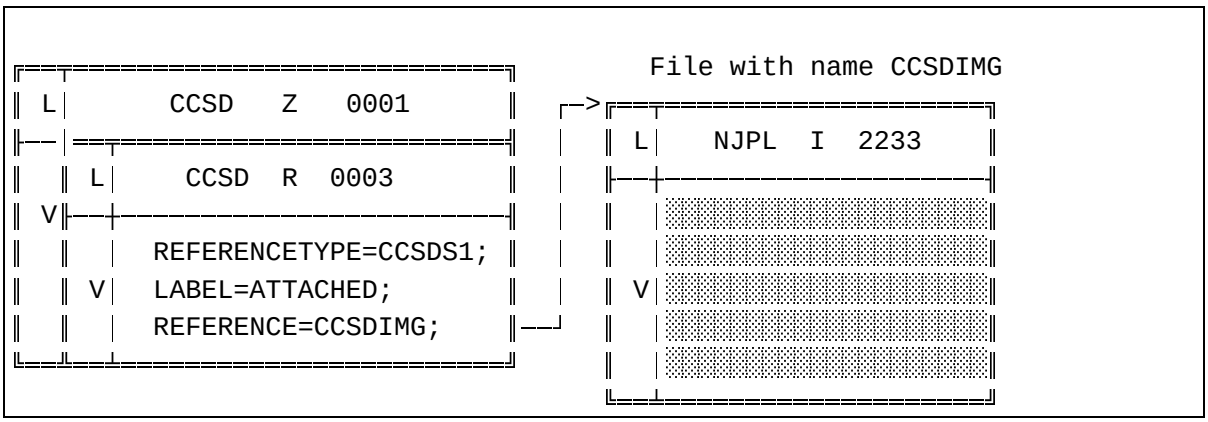


Figure C-3: The Replacement Service - Referencing a File Containing an LVO

A more complicated case is shown in Figure C-4 in which:

- The LVO with Class ID=R is packaged inside an EDU.
- Other LVOs are packaged together with the LVO with Class ID = R to form a more complex product.
- The REFERENCETYPE statement indicates the use of the CCSDS1 Referencing Environment.
- The data in the referenced file is unlabeled and thus a LABEL has to be provided by the referencing data object (the class R LVO).
- The LABEL supplied by the **LABEL** statement specifies that the ADID = ESOC1122, Class ID = I and Delimitation ID = F (Shared End of File).

- The REFERENCE statement references the file called **IMAGE.ERS** containing data without a label.

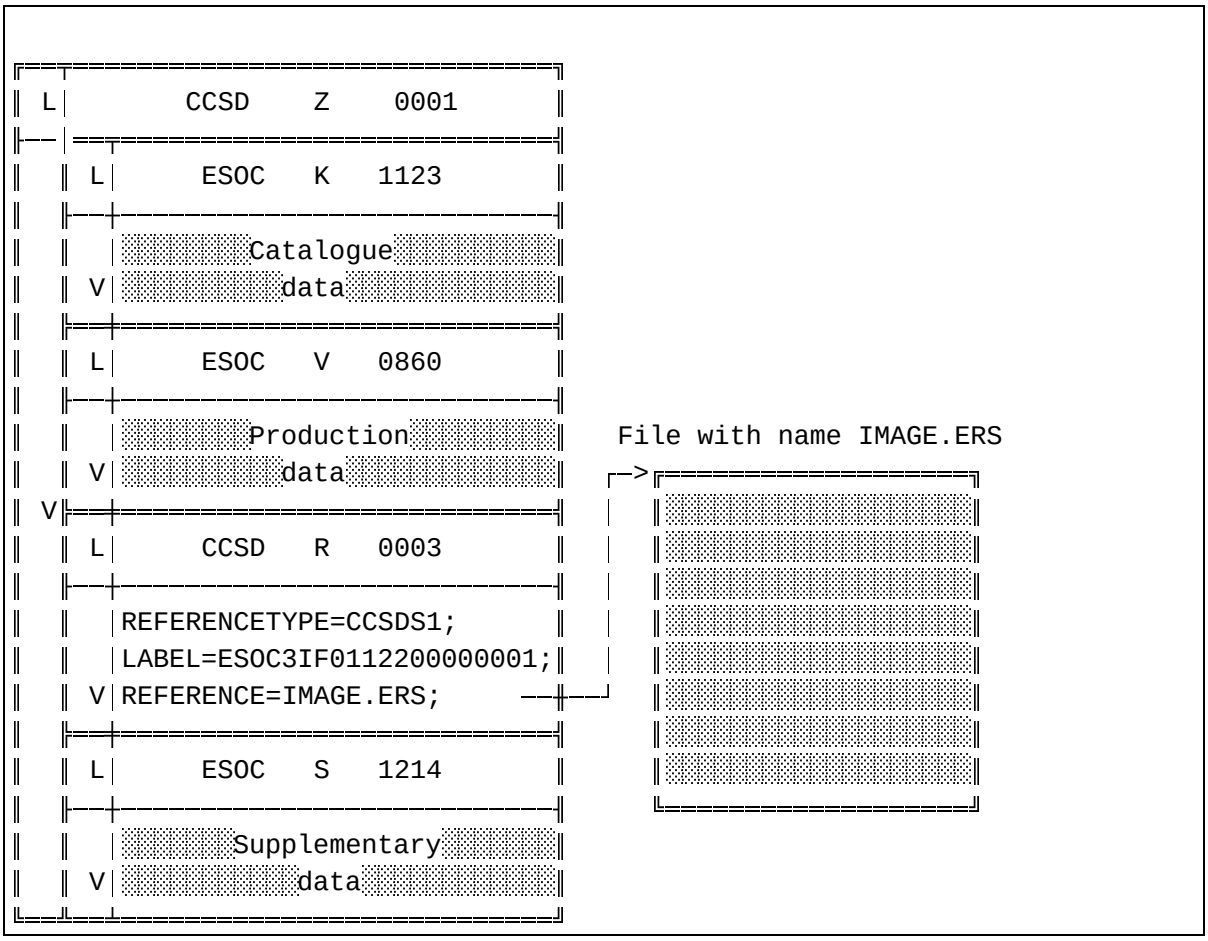


Figure C-4: The Replacement Service - Referencing an Unlabeled Data Object in a File

The following rules apply to interpreting the CCSD0003 statements:

a) External data objects are referenced in the order of the **REFERENCE** statements. If a **REFERENCE** statement specifies more than one external data object, then these external data objects are referenced in the order that they appear in the **REFERENCE** statement.

b) If a **REFERENCE** statement refers to more than one external data object, then the most recent LABEL applies to each of the individual external data objects.

c) Each external data object terminates when the VALUE field delimited by the first LABEL (whether attached or supplied by a **LABEL** statement) has been completed. Additional information in the file is ignored.

d) Each external data object shall comply with the structure rules applying at the point where the external data object was referenced. For example a Data Description Unit (DDU) with ADID = CCSD0005 is required to be packaged within an EDU and to be an LVO with Class ID = F containing an LVO with Class ID = C followed by data description data. Figure C-5 shows a DDU which contains the mandatory LVO with Class ID = C and then a LVO with Class ID = R. The LVO with Class ID = R references two files, **FILE1.DAT** that contains a labeled LVO with Class ID = D (a syntax description) and **FILE2.DAT** that contains a labeled LVO with Class ID = E (a semantic description). The two LVOs are logically included in the DDU thus forming a legal EDU structure.

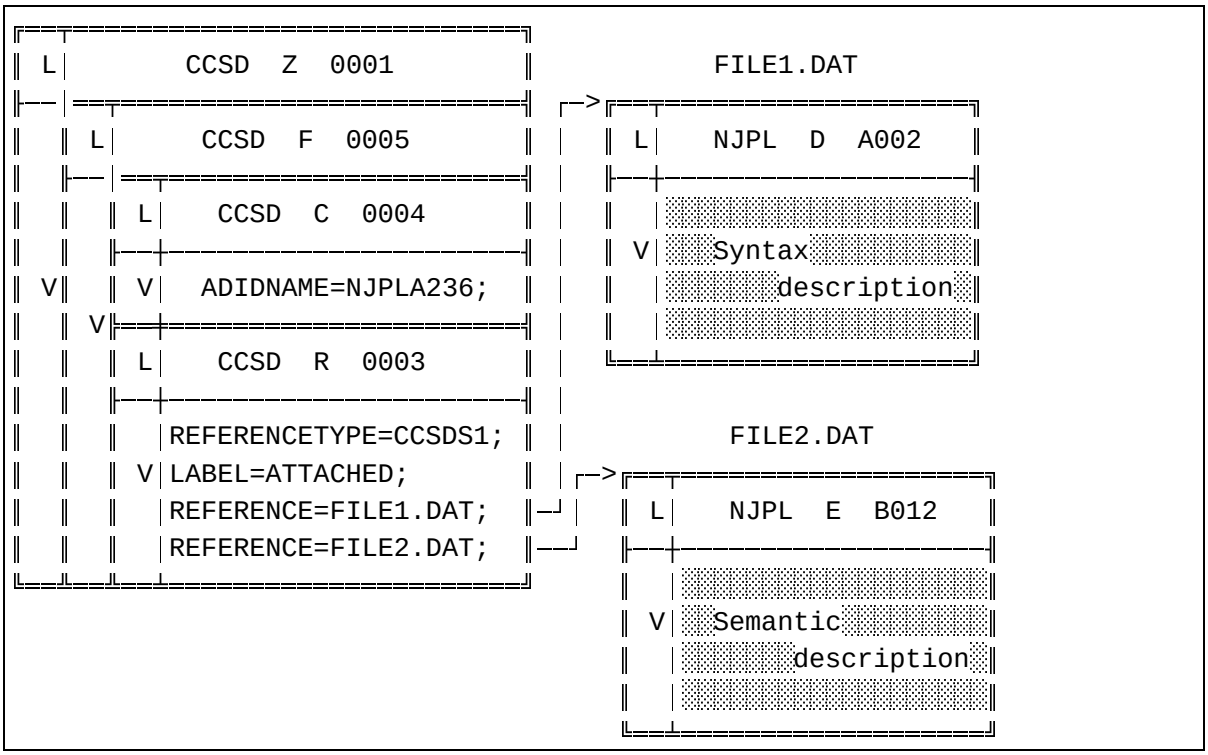


Figure C-5: Following the Structure Rules when Referencing LVOs

Meanwhile, Figure C-6 shows another DDU with a similar structure. The mandatory LVO with Class ID = C is present and then also the LVO with Class ID = R. The LVO with Class ID = R references two files, **FILE1.DAT** that contains a labeled LVO with Class ID = D (a syntax description) and **FILE2.DAT** that contain a labeled LVO with Class ID = I (application data). The LVO with Class ID = D can be legally included, but it is illegal to have LVOs with Class ID = I (application data) within a DDU. Therefore, the EDU structure shown is illegal.

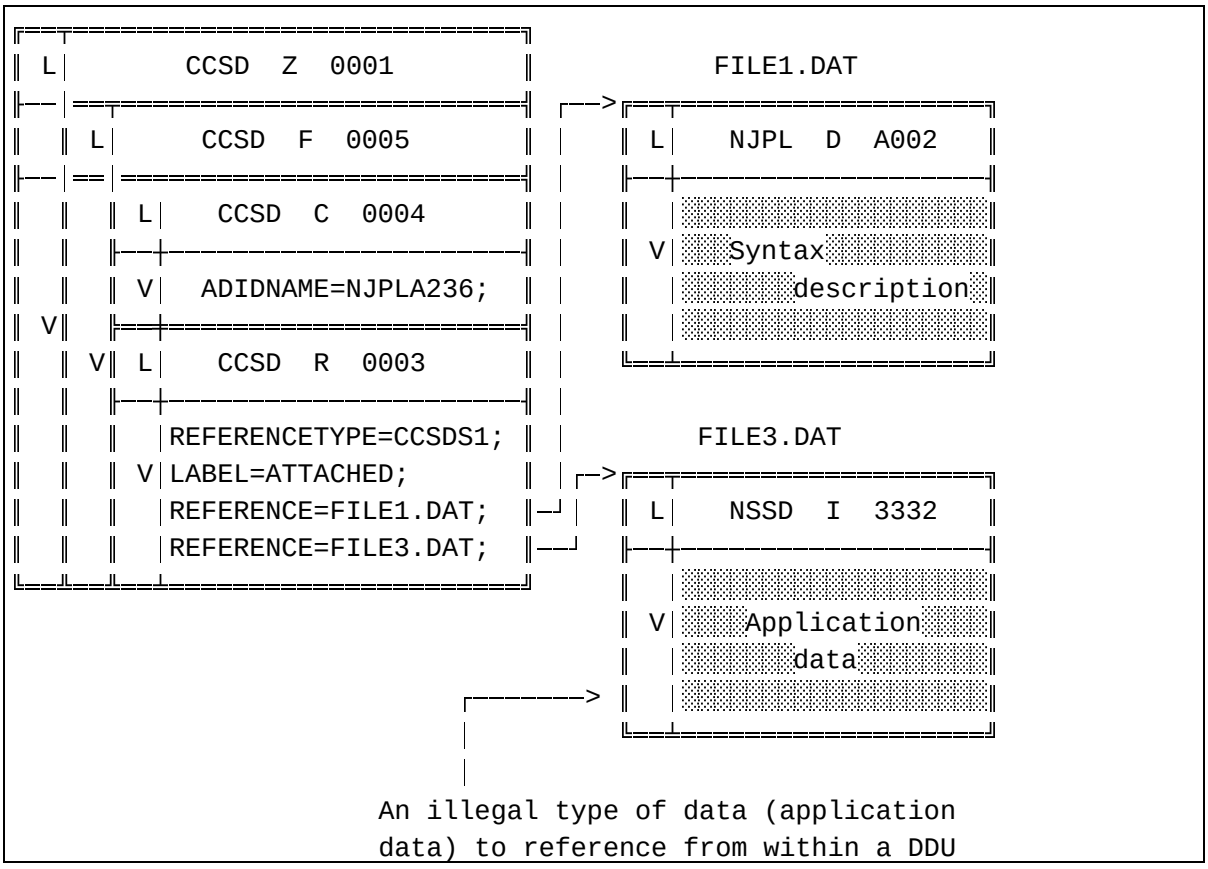


Figure C-6: Example of an ERROR due to NOT Following the Structure Rules when Referencing an LVO

e) The delimitation of the Referencing LVO, and the LVOs within which it is contained, is not affected by the external data objects (e.g., the octet count of a length-delimited Referencing LVO does not take into account the octets of any external data object(s)).

C.3 APPROVED REFERENCING ENVIRONMENT SPECIFICATIONS

The following four CCSDS-defined referencing environments, CCSDS1, CCSDS2, CCSDS3, and CCSDS0, have been approved by the CCSDS at the time of issue of this Recommendation. The existence of only four approved referencing environments at present should not be considered a restriction for the future. New reference environments can be introduced as user needs and technology evolve.

C.3.1 CCSDS1 REFERENCING ENVIRONMENT

Table C-1 shows a number of examples of how a filename referenced within a **CCSDS1** context would translate to various popular filenaming conventions.

Table C-1: Interpretation of CCSDS1 File Specifications

CCSDS1 specification	UNIX filename	DOS filename	VAX VMS® filename
FOOBAR.DAT	FOOBAR.DAT found in the same directory as the Referencing LVO	FOOBAR.DAT found in the same directory as the Referencing LVO	FOOBAR.DAT; found in the same directory as the Referencing LVO
FOOBAR	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR.; found in the same directory as the Referencing LVO
FOOBAR. *NOTE: FOOBAR and FOOBAR. are equivalent specifications in CCSDS1*	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR.; found in the same directory as the Referencing LVO
FOO.BAR.DAT	An ILLEGAL **CCSDS1** filename (only one **.** (dot) permitted in a filename)		
LONG_FOOBAR_NAME.DATA	An ILLEGAL **CCSDS1** filename (only 8 characters permitted in the **filename** and 3 characters in the **extension**)		
MYDIR/FOOBAR.DAT	FOOBAR.DAT found in the MYDIR/ child directory of the directory where the Referencing LVO resides	FOOBAR.DAT found in the MYDIR\ child directory of the directory where the Referencing LVO resides	FOOBAR.DAT; found in the [.MYDIR] child directory of the directory where the Referencing LVO resides
mydir/foobar.dat	An ILLEGAL **CCSDS1** filename (only uppercase characters permitted)		
/YOURDIR/MYDIR/FOOBAR.DAT	FOOBAR.DAT found in the /YOURDIR/MYDIR/ directory off the root directory of the volume	FOOBAR.DAT found in the \YOURDIR\MYDIR\ directory off the root directory of the volume	FOOBAR.DAT; found in the [YOURDIR.MYDIR] directory off the root directory of the volume
/FOOBAR.DAT	FOOBAR.DAT found in the root directory of the volume	FOOBAR.DAT found in the root directory of the volume	FOOBAR.DAT; found in the root directory of the volume
\ADIR\FOOBAR.DAT	An ILLEGAL **CCSDS1** filename (only forward slashes permitted)		

Table C-2 shows some examples of how the wildcarding features are applied. First a sample directory is shown. The table then provides a list of the filenames returned when that directory is referenced with the filename including the wildcard. Note that the order is not significant since the order of returned filenames may differ and wildcards should only be used when order is not significant. Since the period separating the filename and extension fields is always implicitly included in the **CCSDS1** Referencing Environment, the period is included in all the file specifications in the directory list and in each wildcarded specification in the table. The table illustrates that implementers should not rely on the native searching mechanisms to return the correct files. For example, some systems allow the asterisk character to be specified in the middle of a file specification or allow it to match the period.

Table C-2: Interpretation of CCSDS1 Wildcard

Files in sample directory:	AAAA.DAT ABCDE.DAT FFGG.	ABXDE.SYS ABC. ABCDE.SYS	FOO. ABZDE.DAT FOO.DAT	ABCDE. ABYYY.SAM

Wildcarded specification	Returned list of filenames			
AB?.	ABC.			
???.	ABC.	FOO.		
*.DAT	AAAA.DAT	ABCDE.DAT	ABZDE.DAT	FOO.DAT
*.	ABC.	ABCDE.	FFGG.	FOO.
FOO.*	FOO.	FOO.DAT		
foo.*	Illegal as lower case is not allowed in CCSDS1 references.			
AB*.SYS	ABCDE.SYS	ABXDE.SYS		
AB*DE.DAT	Illegal as the ***** wildcard must be the last character in a **filename** or **extension** field			
AB*.S*	ABCDE.SYS	ABXDE.SYS	ABYYY.SAM	
AB?DE.*	ABCDE. ABZDE.DAT	ABCDE.DAT	ABCDE.SYS	ABXDE.SYS
AB?*.	ABC.	ABCDE.		
AB*?.DAT	Illegal as the ***** wildcard must be the last character in a **filename** or **extension** field			
AB*/FOO.DAT	Illegal as the ***** wildcard is not permitted in a **directory name** segment			

C.3.2 CCSDS2 REFERENCING ENVIRONMENT

Table C-3 shows a number of examples of how a filename referenced within a **CCSDS2** context would translate to various popular filenaming conventions. Please note that some common file systems are unable to fully support the **CCSDS2** referencing environment. Normally you should choose to use another referencing environment, such as the **CCSDS1** referencing environment. However, the following table shows a possible set of translations that might be usable in specific circumstances.

Table C-3: Interpretation of CCSDS2 File Specifications

CCSDS2 specification	UNIX filename	DOS filename	VAX VMS® filename
FOOBAR.DAT	FOOBAR.DAT found in the same directory as the Referencing LVO	FOOBAR.DAT found in the same directory as the Referencing LVO	FOOBAR.DAT; found in the same directory as the Referencing LVO
FOOBAR	FOOBAR found in the same directory as the Referencing LVO	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR.; found in the same directory as the Referencing LVO
FOOBAR. *Note: FOOBAR and FOOBAR. are NOT equivalent specifications in CCSDS2*	FOOBAR. found in the same directory as the Referencing LVO	FOOBAR. found in the same directory as the Referencing LVO *NOTE: FOOBAR and FOOBAR. are equivalent names in DOS. DOS is unable to fully support the CCSDS2 Referencing Environment.*	FOOBAR.; found in the same directory as the Referencing LVO *NOTE: FOOBAR and FOOBAR. are equivalent names in VAX VMS. VAX VMS is unable to fully support the CCSDS2 Referencing Environment.*
FOO.BAR.DAT	FOO.BAR.DAT found in the same directory as the Referencing LVO	- - - *NOTE: DOS is unable to fully support the CCSDS2 Referencing Environment and is unable to handle this reference.*	- - - *NOTE: VAX VMS is unable to fully support the CCSDS2 Referencing Environment and is unable to handle this reference.*
LONG_FOOBAR_NAME.DATA	LONG_FOOBAR_NAME.DATA found in the same directory as the Referencing LVO	- - - *NOTE: DOS is unable to fully support the CCSDS2 Referencing Environment and is unable to handle this reference.*	LONG_FOOBAR_NAME.DATA; found in the same directory as the Referencing LVO. *Note: If the filename was longer than 31 characters, VAX VMS would be unable to handle the reference.*
MYDIR/FOOBAR.DAT	FOOBAR.DAT found in the MYDIR/ child directory of the directory where the Referencing LVO resides	FOOBAR.DAT found in the MYDIR\ child directory of the directory where the Referencing LVO resides	FOOBAR.DAT; found in the [.MYDIR] child directory of the directory where the Referencing LVO resides
mydir/foobar.dat *NOTE: MYDIR/FOOBAR.DAT and mydir/foobar.dat are NOT equivalent specifications in CCSDS2.*	foobar.dat found in the mydir/ child directory of the directory where the Referencing LVO resides	FOOBAR.DAT found in the MYDIR\ child directory of the directory where the Referencing LVO resides *NOTE: MYDIR/FOOBAR.DAT and mydir/foobar.dat are equivalent names in DOS. DOS is unable to fully support the CCSDS2 Referencing Environment.*	FOOBAR.DAT found in the [.MYDIR] child directory of the directory where the Referencing LVO resides *NOTE: MYDIR/FOOBAR.DAT and mydir/foobar.dat are equivalent names in VAX VMS. VAX VMS is unable to fully support the CCSDS2 Referencing Environment.*
/YOURDIR/MYDIR/FOOBAR.DAT	FOOBAR.DAT found in the /YOURDIR/MYDIR/ directory off the root directory of the volume	FOOBAR.DAT found in the \YOURDIR\MYDIR\ directory off the root directory of the volume	FOOBAR.DAT; found in the [YOURDIR.MYDIR] directory off the root directory of the volume
/FOOBAR.DAT	FOOBAR.DAT found in the root directory of the volume	FOOBAR.DAT found in the root directory of the volume	FOOBAR.DAT; found in the root directory of the volume
\ADIR\FOOBAR.DAT	An ILLEGAL **CCSDS2** filename (only forward slashes permitted)		

Table C-4 shows some examples of how the wildcarding features are applied. First a sample directory is shown. The table then provides a list of the filenames returned when that directory is referenced with the filename including the wildcard. Note that the order is not significant since the order of returned filenames may differ and wildcards should only be used when order is not significant. Note also that filenames and wildcarded specifications are included that do not contain a period. Unlike the **CCSDS1** referencing environment, such filenames are valid in the **CCSDS2** referencing environment.

Table C-4: Interpretation of CCSDS2 Wildcard Specification

Files in sample directory:	AAAA.DAT ABCDE.DAT FFGG	ABXDE.SYS ABC ABCDE.SYS	FOO ABZDE.DAT FOO.DAT	ABCDE ABYYY.SAM foo.dat

Wildcarded specification	Returned list of filenames			
AB?	ABC			
???	ABC	FOO		
*.DAT	Illegal as the ***** wildcard must be the last character in a **filename** field			
*	AAAA.DAT ABCDE.SYS FFGG	ABC ABXDE.SYS FOO	ABCDE ABYYY.SAM FOO.DAT	ABCDE.DAT ABZDE.DAT
FOO.*	FOO.DAT			
foo.*	foo.dat			
AB*.SYS	ABCDE.SYS	ABXDE.SYS		
AB*DE.DAT	Illegal as the ***** wildcard must be the last character in a **filename** field			
AB*.S*	Illegal as the ***** wildcard must be the last character in a **filename** field			
AB?DE.*	ABCDE.DAT	ABCDE.SYS	ABXDE.SYS	ABZDE.DAT
AB?*	ABC ABXDE.SYS	ABCDE ABYYY.SAM	ABCDE.DAT ABZDE.DAT	ABCDE.SYS
AB*?.DAT	Illegal as the ***** wildcard must be the last character in a **filename** field			
AB*/FOO.DAT	Illegal as the ***** wildcard is not permitted in a **directory name** segment			

C.3.3 CCSDS3 REFERENCING ENVIRONMENT

This specification provides a method for specifying files on sequential media when the underlying protocol does not associate a name with the file. External data object names within the CCSDS3 referencing environment consist of an optionally signed, ASCII-formatted decimal integer. Implementation is straight forward if the following items are noted:

- both absolute numbering and relative numbering are allowed;
- the first file on the volume is expressed as "1", i.e. there is no file "0";
- a relative reference of +0 or -0 would correspond to the current file and is not allowed; and
- Every file on the media is counted. For example, sequential media to which ISO standard labels have been added will store a single logical file as three files of which only the middle file is available for user data. Each of these three files would be included in the file count using this referencing environment.

C.3.4 CCSDS0 REFERENCING

This specification combines the filenaming specifications for any number of referencing environments into one construct. This allows the user to define references which take advantage of each referencing environment while avoiding the need to rewrite the referencing object when it is moved from one environment to another.

The following items should be noted:

- A Reference Tag for any particular referencing environment may appear at most once.
- Wildcards are not permitted in the Reference Names. This allows a one to one match of files regardless of the environment.
- If the medium being used is a sequential medium that does not support the use of filenames, then only the CCSDS3 Tagged Names have any meaning.
- When accessing a data product, the referencing environments are tried in the order they are listed. For example, if a CCSDS2 Tagged Name is listed followed by a CCSDS1 Tagged Name and if your local environment supports both referencing environments, then you would attempt to use the CCSDS2 name first.

- If you are accessing a data product and if a Tagged Name from the first listed referencing environment does not exist in your local environment even though the local environment may support that referencing environment, then the Tagged Name from the next referencing environment shall be used.

- If you are creating a data product, you may wish to supply recommended names for a number of different possible environments via the referencing object. However on a single volume you are creating, you should name all files in accordance with a single referencing environment.

- If you are migrating a data product from one media to another, you should name all files in accordance with a single referencing environment. If the new media is unable to support the referencing environments currently provided, you may want to perform the conversion and then add an additional Tagged Name for the referencing environment you use on the new media.

ANNEX D

THE $CCSDS3 REFERENCING ENVIRONMENT

(This annex **is not** part of the Recommendation.)

Purpose:

This annex provides tutorial material to aid implementers and users of the prototype $CCSDS3 referencing environment in converting to the use of the CCSDS0 referencing environment.

The material appearing in this Annex is deprecated. The CCSDS0 referencing environment should be used instead of the $CCSDS3 referencing environment described in this Annex.

D.1 COMBINED REFERENCING ENVIRONMENT - $CCSDS3

This specification combines the filenaming specifications for $CCSDS1 and $CCSDS2 into one construct. This allows the user to define directory/filenames which take advantage of each referencing environment while avoiding the need to rewrite the referencing object as when it is moved from one referencing environment to another.

External data object names within the $CCSDS3 referencing environment are specified as a pair of tagged names, the pair being enclosed in double quote marks. A **$1** tag indicates that the name conforms to the rules for a **$CCSDS1** name, while a **$2** tag indicates that the name conforms to the rules for a **$CCSDS2** name. Wildcarding is not permitted in either case.

As two possible names are specified for this environment, a convention for usage of the names is required. This convention is a follows; the referencing object shall first try the **$2** tagged name, if this fails, then the **$1** tagged name shall be used. (The **$2** tagged name will fail if the local environment uses **$1** tagged names, or if the **$2** tagged name does not exist even though the local environment may support the **$2** tagged name format.*)*

A structure diagram of the quoted string to be used for these name pairings is given in Figure D-1. Each of the elements in this diagram may be separated by White Space as defined in the Parameter Value Language specification (Reference [2]); that is one or more of any of ASCII characters space, horizontal tab, vertical tab, line feed, form feed, and carriage return.

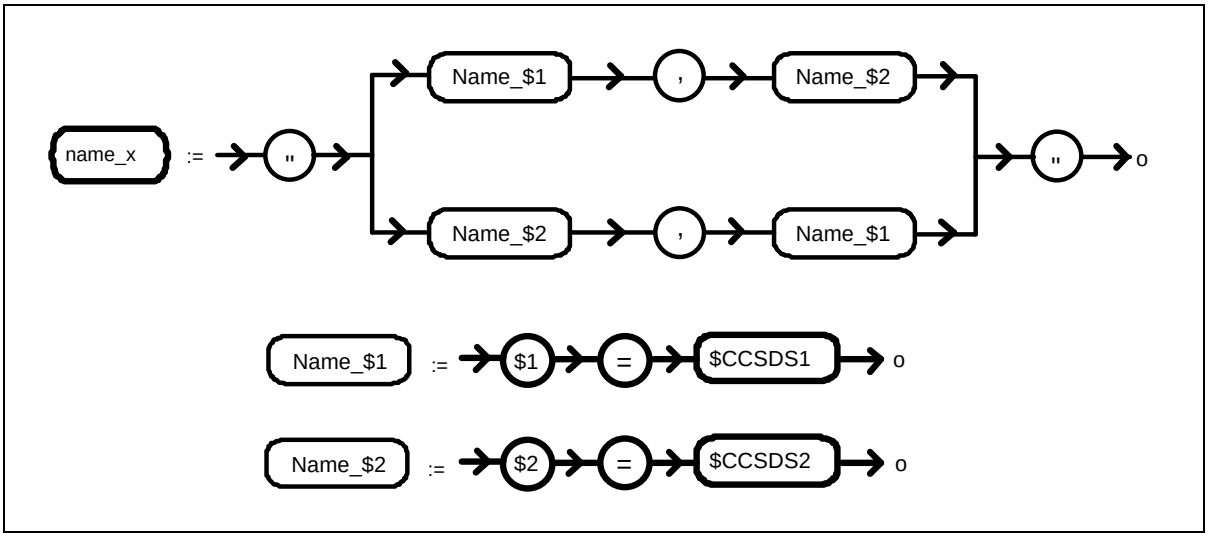


Figure D-1: Structure Diagram of $CCSDS3 Name Specification

NOTE Although the prototype $CCSDS3 referencing environment and the CCSDS0 referencing environment serve similar functions and appear similar, their respective formats are sufficiently different such that a $CCSDS3 specification cannot be directly interpreted as a CCSDS0 specification. A specification in the proposed $CCSDS3 referencing environment can be converted into a specification in the CCSDS0 referencing environment by breaking the single $CCSDS3 quoted string into separate quoted strings for each reference environment, changing $1 tags to CCSDS1 and $2 tags to CCSDS2, etc., and then including the CCSDS1 specification string before the CCSDS2 specification string.

INDEX