INTERNATIONAL STANDARD

ISO/IEC 10728

First edition
1993-04-15

AMENDMENT 1
1995-12-15

Information technology — Information Resource Dictionary System (IRDS) Services Interface

AMENDMENT 1: C language binding

Technologies de l'information — Interface de services du gestionnaire de ressources du système d'informations (IRDS)

AMENDEMENT 1: Liant de langage C

Reference number
ISO/IEC 10728:1993/Amd.1:1995(E)

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Amendment 1 to International Standard ISO/IEC 10728:1993 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 21, *Open systems interconnection, data management and open distributed processing*.

ISO/IEC Copyright Office • Case Postale 56 • CH-1211 Genève 20 • Switzerland
Printed in Switzerland

Information technology — Information Resource Dictionary System (IRDS) Service Interface

AMENDMENT 1: C language binding

Page v

Contents

Add a new entry to the Table of Contents as follows:

"Annex C - C language binding"

Page 1

Clause 1

Add a new sentence in clause 1 paragraph 2, before the last sentence.

"A language binding for the C language is provided in annex C."

Page 5

Subclause 4.4

Add a new sentence in subclause 4.4.

"Data structures for use with the C language are defined in annex C."

Page 5

Subclause 4.5

Add a new sentence in subclause 4.5.

"C language bindings for the services are provided in annex C."

Page 71

Subclause 8.1

Amend the first sentence of the NOTE in subclause 8.1 to read:

"For the Pascal language binding specified in this clause and the C language binding specified in annex C, enumerated types are ...."

Page 75

Clause 9

Amend the first sentence of the second paragraph of Clause 9 to read:

"The service formats are specified in this clause using ISO standard Pascal. Alternative service formats for use with the C language are specified in annex C."

Page 105

Add a new annex C as follows:

Annex C
(normative)
C language binding

The IRDS Services Interface language bindings for the C language are presented in the form of a C .h file as set out in clause C.3 below.

C.1 Strategy for the Language Binding

In this binding the data names and data structures defined in clause 8 have been adhered to except where the C language does not provide an appropriate construct.

In this binding the procedure names and their parameters defined in clause 8 have been adhered to except where the C language does not provide an appropriate construct.

Macros have been used for various limits instead of const-qualified variables. Standard C requires that the length of an array be a compile-time constant integer expression and a const-qualified variable does not have this property.

Use of this language binding requires the use of a C implementation that conforms to ISO/IEC 9899:1990.

C.2 General Rules

1. Those data names in C.2 below that also appear in Clause 8 shall have the same meaning as is defined in Clause 8. The same rules for the use of separators as defined in Clause 8 shall apply.

2. The function and parameter names in C.2 below shall have the same meaning as is defined in Clause 9.

3. The Service Return Codes returned shall be those defined by Clause 9 and Annex A and they shall have the same meaning.

4. The following mappings from SQL data types to C data types have been used:

SQL DATA TYPE	C DATA TYPE	
CHARACTER	pIrdsText	/* See 8.2.1 */
CHARACTER VARYING	pIrdsText	/* See 8.2.1 */
NATIONAL CHARACTER	pIrdsText	/* See 8.2.1 */
NATIONAL CHARACTER VARYING	pIrdsText	/* See 8.2.1 */
REAL	float	
DOUBLE PRECISION	double	
FLOAT	float	
INTEGER	long	
SMALLINT	short	
NUMERIC	long	
DECIMAL	long	
DATE	IrdsDate	/* See 8.2.1 */
TIME	IrdsTime	/* See 8.2.1 */

TIMESTAMP	IrdsTimeStamp	/* See 8.2.1 */
INTERVAL	IrdsInterval	/* See 8.2.1 */

5. Every function returns an int which is to be set to the value of the NumStates field of the RetCode returned by the function.

C.3 Notes for Implementors

1. To ensure uniqueness all externally visible names start with the characters Irds.

2 For readability and because of the above requirement, many names are not unique within the first six characters of the name. Those implementors whose implementations do not support long names are to add suitable #define statements to ensure that all names are distinguished by their implementation. For example,

    #define IrdsOpen I_O

    #define IrdsOpenCursor I_OC

C.4 C language .h file

```
/*    This file is called irds.h
*
*     This version of this header conforms to:
*           ISO/IEC 10728:1993(E)
*           IRDS Services Interface
*
*     © Copyright ISO 1995
*/


/* Protect against multiple inclusion of the header file */
#ifndef IRDS_H
#define IRDS_H

/*    Allow use by C++ Compilers  */

#ifdef __cplusplus
extern "C" {
#endif

#include <stddef.h>
      /* to ensure that NULL is defined */

/*    Implementor defined length limits  */

/* Clause 8.1.1  */

#define IrdsNameLim     t1    /*    t1 is to be replaced by an
                                    implementor defined
                                    value  */
```

```
                  /* IrdsNameLim is used in 8.2.2 below      */

#define IrdsVarLim      t2     /*     t2 is to be replaced by an
                                      implementor defined
                                      value  */
                  /* IrdsVarLim is used in 8.2.2 below       */

/* Clause 8.1.2  */

#define IrdsTextLim     t3     /*     t3 is to be replaced by an
                                      implementor defined
                                      value  */

/*     Note that although no use is made of IrdsTextLim in this
language
       binding, it is included so that a program may refer to the
value
       to test the length of a string. */


/* Clause 8.1.3  - these definitions are used in 8.2.3 below */

/*     The numeric values n1, n2, n3, n4 in this clause are to be
       replaced by implementor defined values.  */

#define IrdsSessIdLim          n1
                  /* IrdsSessIdLim used in 8.2.3 below       */
#define IrdsCurIdLim           n2
                  /* IrdsCurIdLim used in 8.2.3 below */
#define IrdsImpDicNameLen      n3
                  /* IrdsImpDicNameLen used in 8.2.3 below */

#define IrdsKeyLen             n4
                  /* IrdsKeyLen used in 8.2.1 below    */
/* Clause 8.1.4 Data Types    */

typedef enum
       {
              IrdsDataTypeChar,              /* SQL CHARACTER   */
              IrdsDataTypeCharVar,
                                   /* SQL CHARACTER VARYING       */
              IrdsDataTypeNatChar,
                                   /* SQL NATIONAL CHARACTER      */
              IrdsDataTypeNatCharVar,
                            /* SQL NATIONAL CHARACTER VARYING   */
              IrdsDataTypeReal,              /* SQL REAL        */
              IrdsDataTypeDouble,
                                          /* SQL DOUBLE PRECISION */
              IrdsDataTypeFloat,          /* SQL FLOAT            */
              IrdsDataTypeInteger,        /* SQL INTEGER          */
              IrdsDataTypeSmallint,
                                          /* SQL SMALLINT         */
              IrdsDataTypeNumeric,        /* SQL NUMERIC          */
              IrdsDataTypeDecimal,        /* SQL DECIMAL          */
              IrdsDataTypeDate,           /* SQL DATE             */
              IrdsDataTypeTime,           /* SQL TIME             */
              IrdsDataTypeTimestamp,
                                          /* SQL TIMESTAMP        */
              IrdsDataTypeInterval,     /* SQL INTERVAL           */
              IrdsDataTypeIrdsKey         /* SQL IRDS KEY         */
       } IrdsDataType;


/*  Clause 8.1.5 IRD Content Status Classes     */

typedef enum
       {

              IrdsDcsClsUcntl,            /* Uncontrolled         */
              IrdsDcsClsCntl,             /* Controlled           */
              IrdsDcsClsArch              /* Archived             */
       } IrdsDcsCls;
```

```
/*    Clause 8.1.6 Close Type parameter */

typedef enum
      {

            RequestIrdsCommit,              /* COMMIT          */
            RequestIrdsRollback             /* ROLLBACK        */
      } IrdsCloseType;

/*    Clause 8.2.1 Column data types */

typedef char * pIrdsText;
/*    Note that in this language binding a pointer to a text
      string is used instead of an array of char as specified
      in the Pascal binding in Clause 8.  This removes the
      length limit on text.
*/

typedef struct
      {
            char Year[4];
            char Sep1;
            char Month[2];
            char Sep2;
            char Day[2];
      }  IrdsDate;

typedef struct
      {
            char Hour[2];
            char Sep1;
            char Minute[2];
            char Sep2;
            char Second[2];
            char Sep3;
            char Fraction[3];
      }     IrdsTime;

typedef struct
      {
            IrdsDate     Date;
            char         SepT;
            IrdsTime     Time;
      }     IrdsTimestamp;
```

```
typedef struct
      {
            char        Days[7];
            char        SepI;
            IrdsTime    Time;
      }     IrdsInterval;

typedef char IrdsKey[IrdsKeyLen];
                  /* IrdsKeyLen is defined in 8.1.3 above */

/* Clause 8.2.2  Object Names */

typedef char IrdsSQLName[128];
                  /*    128 is set by ISO/IEC 9075:1992 database
                        Language SQL */

typedef char IrdsName[IrdsNameLim];
                  /* IrdsNameLim is defined in 8.1.1 above  */

typedef char IrdsVarName[IrdsVarLim];
                  /* IrdsVarLim is defined in 8.1.1 above   */

typedef char UserId[IrdsNameLim];
                  /* IrdsNameLim is defined in 8.1.1 above  */


/* Clause 8.2.3  Control Identifiers */

typedef char IrdsSessId[IrdsSessIdLim];
            /* IrdsSessIdLim is defined in 8.1.3 above      */

typedef char IrdsCurId[IrdsCurIdLim];
            /* IrdsCurIdLim is defined in 8.1.3 above       */

typedef char IrdsImpDicName[IrdsImpDicNameLen];
            /* IrdsImpDicNameLen is defined in 8.1.3 above  */


/* Clause 8.2.4  Diagnostics Area */

typedef struct
      {
            char  StateClass[2];
            char  StateSubClass[3];
      }     IrdsState;
                        /* IrdsState is used in 8.2.5 below */
```

```
typedef struct
      {
            int          IrdStateSeq;
            IrdsState    IrdReturnedState;
            IrdsSQLName  IrdConstraintSchema;
            IrdsSQLName  IrdConstraintName;
            IrdsSQLName  IrdSchemaName;
            IrdsSQLName  IrdTableName;
            int          IrdColumnNumber;
            IrdsSQLName  IrdColumnName;
      }     IrdsStateRec;
                         /*    IrdsStateRec is used in the
                               Get Diagnostcs Service in
                               9.1.8 below                    */


/*    Clause 8.2.5  Service Return Code */

typedef struct
      {
            int          NumStates;
            IrdsState    State;
      }     IrdsRetCode;


/*  Clause 8.2.6 Column List parameters */

/*    In the C binding the column list actually is a list.
      In the Pascal binding the list was actually an array.
      In IrdsColList the FirstColumn points to the first
      IrdsSingleColSpec.
      In each IrdsSingleColSpec, NextColumn  points to the following
      IrdsSingleColSpec and so on.

      The definition of a boolean data type is not included
      in the C language definition, but many environments
      defines one automatically.  To be secure, an Irds_Boolean
      is defined.

*/

typedef enum {Irds_False = 0, Irds_True = 1} Irds_Boolean;
```