Unit 3, The Paddock
Hambridge Road
Newbury
Berkshire RG14 5TQ

Project: ONE

Definition for Extensions to the DBV OSI II API for Access Control, Resource Control, Trigger Resource Control, Explain and AccessControlPrompt-1

Document.-No.: D-008P1.doc

Version: P1 Draft A

Author:

R. Bull, Crossnet Systems Limited

Printed : 27.08.1996

Table of ContentsError! Bookmark not defined.

1. Introduction

This document describes extensions to the DBV OSI II API that have been developed as a software patch to the formally released API package, but as yet are not formally included in the main package.

These extensions will be included in the main package when formally conformance tested.

All extensions are in the same style as the main DBV OSI II API package.

This document should be read in conjunction with the main API manual:

DBV-OSI II API (Application Programmers Interface), D-008V5.DOC

In chapter 2 of this document the new extensions are summarised.

In Chapter 3, the new API functions are described.

In Annex A, the structure definitions of the API extensions are provided.

1.1 Background of API Releases

The main release of the software is version 1.8. This software is released on public ftp servers together with appropriate documentation:

- API manual, (D-008V5)
- SES (System External Specification)
- SIS (System Internal Specification)

The following services and Externals are supported in Version 1.8:

Services:

Initialise
Search
Present
Segment
Sort
Resource Report
Scan
DeleteResultSet
Extended Service
Close

Externals:

Unstructured Text Record, SUTRS

OPAC Record,

Summary Record,

Resource Report-1

Resource Report-2

Extended Service Task package

Extended Services:

Update

A new conformance tested release of the API package (Version 1.9.x) will contain the following extensions:

Item Order Extended Service

ILL Request,

ILL Status

SUBITO ILL extension.

Additional API functions and services are also available as a patch to 1.9.x which have been funded external to the DBV OSI II project. When these functions and services are conformance tested, they will be formally included in a Version 2.0 of the API package. These functions and services are described in the following Chapters.

1.2 Installing the Patch

Refer to the README file instructions supplied with the patch kit.

2. Summary of API Extensions

In this chapter a summary of the API extensions is provided.

The following API services are provided:

Access Control
Resource Control
Trigger Resource Control

Explain Record
AccessControlPrompt-1 External
Support for Exp.1 Attribute Set

All additions are in the same style as the previous releases. Use of this version of the API and the combined patch are entirely backward compatible with existing applications except for a change in the structure of ResourceReport-2 which was considered a bug with respect to the standard.

Please note that the file sitypes2.h contains additional definitions for future extensions of the API. Only those that are referred to above are actually supported at the moment.

2.1 Style of the API Functions

The API provides:

- Functions that provide session Association - connection and disconnection
- Functions that provide data transfer of SR-/Z39.50 services and
- utility functions that are not communications orientated.

All Functions provide a "Return Value" which is either **SI_OK** or **SI_NOTOK**.

If the Functions returns SI_OK, then it is to be assumed that the function has worked.

If the function returns SI_NOTOK, then it is to be assumed that the function has not worked, and that the reason for failure is to be found in a datastructure as below:

```
typedef struct DbvError {
  DbvErrorClass                eSIErrorClass;
  union {
```

```
    DbvErrorCode          lIfOsCoError;
    DbvAbort              *psSIAbort;
  }                  uErrorCode
  DbvAssociationState eAssociationState
} SIError;
```

The Parameter *eSIErrorClass* provides the error class, which will be one of:

SI_Communication_Error	failure to send a PDU etc.
SI_Interface_Error	Use of the function was incorrect due to a parameter error etc.
SI_OperatingSystem_Error	The function cannot be realised due to a error that is not under control of the API itself.
SI_Abort	A fatal system error has occured.

A union exists across SI_Interface_Error, SI_Communication_Error and SI_OperatingSystem_Error, in which the reason is fould in the structure component *lIfOsCoError*.

If a SI_Abort error is received, the error is described in the DbvAbort data structure.

Each parameter of the API functions is defined as one of "IN", "OUT_C" or "OUT_F". These are described as:

IN: This is an input parameter and must be correctly entered.

OUT_C: A pointer to a value that has been initialised before the function is called.

OUT_F: A pointer to a value which will be initialised within the Function. The value passed in should be NULL.

2.2 Naming Conventions

The following naming conventions are used for definitions of Constants, DataStructures, Elements of Enumerated Types and variables:

Constants:	Upper case letters in the range {A..Z, _} (e.g, . SI_NOTOK).
Names of structure datatypes:	All start with an Upper case letter with subsequent lower case letters. Subsequent words in the name are likely to use Upper case letters as well. (e.g, SearchReq).

Elements in "enumeration types":	Upper case letters or numbers in the range {A..Z, a..z, _, 1..9}, typically starting with Upper case letters indicating the name of the type, an underscore, and then the name of the enumeration type. (e.g, AT_OpacRecord).
Variables:	Upper case and lower case letters in the range {A..Z, a..z}, the first letters imply the type, (e.g, psSearchReq).

For the variable types, the following naming is used for the variable prefix in lower case leters:

long	lAssocId;	/* l: *long* Variable */
long	*plAssocId;	/* pl: Pointer to a *long* Variable */
DbvErrorClass	eErrorClass;	/* e: "enumeration type" */
DbvErrorClass	peErrorClass;	/* pe: "Pointer to a enumeration type" */
Structure	sMyStruct;	/* s: Structure */
SearchReq	*psSearchReq;	/* ps: Pointer to a Structure */
String	pcAttributeSetId;	/* pc: Pointer to a String */
DbvError	**ppsSIError;	/* pps: Pointer to a Pointer to a DataStructure */
DbvBool	fStatus;	/* f: Boolean type */
DbvBool	*pfStatus;	/* pf: Pointer to a Boolean type */
UnsignedLong	uLong;	/* u: Unsigned Long */
UnsignedLong	*puLong;	/* pu: Pointer to a Unsigned Long */

3. Additional API-Functions

This chapter describes the extentions to the DBV-OSI II API.

From the API users point of view, the behaviour of the functions provided for the transfer of SR-/Z39.50 data is exactly the same in both environments.

There are no changes to the functions provided for association/connection handling.

Chapter 3.1 describes the extensions to API functions available to the "Origin System".

Chapter 3.2 describes the extensions to API functions available to the "Target System".

3.1 Additional Functions Used by the Origin System

Additional Functions available to the API-user are for both the OSI-, and TCP-environments.

The following additional functions are available to the API-user:

3.1.1 DbvAccessControlResponse()

Purpose:

Send Access Control Response APDU to Target.

Function Name and Parameters:

```
DbvAccessControlResponse(
DbvAssocId                  lAssocId,                /* IN */
DbvAccessControlRsp         *psAccessControlRsp,     /* IN */
DbvError                    **ppsSIError             /* OUT_F */
)
```

Return Value:

SI_OK:	if no error occured.
SI_NOTOK:	on error.

Parameter Description:

lAssocId	The unique identifier of the association, as received by a call of function DbvAssociateRequest().
psAccessControlRsp	Pointer to the data to be transferred. The definition of the data type *DbvAccessControlRsp* corresponds to the definition of the ACCESS CONTROL Response service primitive.
ppsSIError	The address of a pointer of type **DbvError*. When calling the function, the pointer should refer to (DbvError*)NULL. The pointer is updated only if the function returns SI_NOTOK. In this case the pointer refers to a filled-in *DbvError* structure providing information about the failure occured.

Function Description:

Note, the Access Control Service is initiated as a Target Challenge to the Origin.

The caller of the function provides in parameter *psAccessControlRsp* the data corresponding to an ACCESS CONTROL Response service primitive. The function checks the data for completeness, forms and submits an AccessControlResponse-APDU.

If the function returns SI_OK, the data had been submitted successfully.

If the function returns SI_NOTOK, the data had not been transferred. In this case the *DbvError* structure indicates the reason for the transfer failure.

Usage:

```
#include "si_defs.h"
#include "si_types.h"
#include "dbvproto.h"
#include "si_types2.h"
#include "iord_ext.h"
#include "itemordr.h"

DbvAssocId                lAssocId = NO_ASSOCIATION_ID_ASSIGNED;
DbvAccessControlRsp      *psAccessControlRsp = NULL;
DbvError                 *psError = NULL;


/*
    Suppose the call to function DbvAssociateRequest()
    had been successful and lAssocId is a valid identifier.
    During an operation, the Target has issued an AccessControl
    Challenge
    Allocate buffer for psAccessControlRsp and fill in structure.
*/


if (DbvAccessControlResponse(lAssocId, psAccessControlRsp, &psError) == SI_OK)
{
   FreeAccessControlRsp(&psAccessControlRsp);
                 :
                 :
}
else
{
   PrivateErrorHandling(psError);
   FreeAccessControlRsp(&psAccessControlRsp);
   FreeErrorCode(&psSIError); /* release buffer, assign NULL pointer */
                 :
                 :
}
                 :
                 :
```

3.1.2 DbvResourceControlResponse()

Purpose:

Send Resource Control Response APDU to the Target.

Function Name and Parameters:

```
DbvResourceControlResponse(
DbvAssocId                      lAssocId,                /* IN */
DbvResourceControlRsp         *psResourceControlRsp,     /* IN */
DbvError                     **ppsSIError               /* OUT_F */
)
```

Return Value:

SI_OK:	if no error occured.
SI_NOTOK:	on error.

Parameter Description:

lAssocId	The unique identifier of the association, as received by a call of function DbvAssociateRequest().
psResourceControlRsp	Pointer to the data to be transferred. The definition of the data type *DbvResourceControlRsp* corresponds to the definition of the RESOURCE CONTROL Response service primitive.
ppsSIError	The address of a pointer of type **DbvError*. When calling the function, the pointer should refer to (DbvError*)NULL. The pointer is updated only if the function returns SI_NOTOK. In this case the pointer refers to a filled-in *DbvError* structure providing information about the failure occured.

Function Description:

Note, the Resource Control Service is initiated as a Target Challenge to the Origin. The Target may or may not require a response. This function is used if a response is required.

The caller of the function provides in parameter *psResourceControlRsp* the data corresponding to an RESOURCE CONTROL Response service primitive. The function checks the data for completeness, forms and submits an ResourceControlResponse-APDU.

If the function returns SI_OK, the data had been submitted successfully.

If the function returns SI_NOTOK, the data had not been transferred. In this case the *DbvError* structure indicates the reason for the transfer failure.

Usage:

```
#include "si_defs.h"
#include "si_types.h"
#include "dbvproto.h"
#include "si_types2.h"
#include "iord_ext.h"
#include "itemordr.h"

DbvAssocId               lAssocId = NO_ASSOCIATION_ID_ASSIGNED;
DbvResourceControlRsp   *psResourceControlRsp = NULL;
DbvError                *psError = NULL;


/*
    Suppose the call to function DbvAssociateRequest()
    had been successful and lAssocId is a valid identifier.
    During an operation, the Target has issued a ResourceControl
    Challenge and requires that a response is sent to the Target.
    Allocate buffer for psResourceControlRsp and fill in structure.
*/


if (DbvResourceControlResponse(lAssocId, psResourceControlRsp, &psError) ==
SI_OK)
{
   FreeResourceControlRsp(&psResourceControlRsp);
                 :
                 :
}
else
{
   PrivateErrorHandling(psError);
   FreeResourceControlRsp(&psResourceControlRsp);
   FreeErrorCode(&psSIError); /* release buffer, assign NULL pointer */
                 :
                 :
}
                 :
                 :
```

3.1.3 DbvTriggerResourceControlRequest()

Purpose:

Send Trigger Resource Control Request APDU to the Target.

Function Name and Parameters:

```
DbvTriggerResourceControlRequest(
DbvAssocId                        lAssocId,                         /* IN */
DbvTriggerResourceControlReq      *psTriggerResourceControlReq,     /* IN */
DbvError                          **ppsSIError                      /* OUT_F */
)
```

Return Value:

SI_OK:	if no error occured.
SI_NOTOK:	on error.

Parameter Description:

lAssocId	The unique identifier of the association, as received by a call of function DbvAssociateRequest().
psTriggerResourceControlReq	Pointer to the data to be transferred. The definition of the data type *DbvTriggerResourceControlReq* corresponds to the definition of the TRIGGER RESOURCE CONTROL Request service primitive.
ppsSIError	The address of a pointer of type **DbvError*. When calling the function, the pointer should refer to (DbvError*)NULL. The pointer is updated only if the function returns SI_NOTOK. In this case the pointer refers to a filled-in *DbvError* structure providing information about the failure occured.

Function Description:

The caller of the function provides in parameter *psTriggerResourceControlReq* the data corresponding to a TRIGGER RESOURCE CONTROL Request service primitive. The function checks the data for completeness, forms and submits a TriggerResourceControlRequest-APDU.

If the function returns SI_OK, the data had been submitted successfully.

If the function returns SI_NOTOK, the data had not been transferred. In this case the *DbvError* structure indicates the reason for the transfer failure.

Usage:

```
#include "si_defs.h"
#include "si_types.h"
#include "dbvproto.h"
#include "si_types2.h"
#include "iord_ext.h"
#include "itemordr.h"

DbvAssocId                     lAssocId = NO_ASSOCIATION_ID_ASSIGNED;
DbvTriggerResourceControlReq  *psTriggerResourceControlReq = NULL;
DbvError                      *psError = NULL;


/*
    Suppose the call to function DbvAssociateRequest()
    had been successful and lAssocId is a valid identifier.
    Allocate buffer for psTriggerResourceControlReq and fill in structure.
*/


if (DbvTriggerResourceControlRequest(lAssocId, psTriggerResourceControlReq,
&psError) == SI_OK)
{
   FreeTriggerResourceControlReq(&psTriggerResourceControlReq);
                 :
                 :
}
else
{
   PrivateErrorHandling(psError);
   FreeTriggerResourceControlReq(&psTriggerResourceControlReq);
   FreeErrorCode(&psSIError); /* release buffer, assign NULL pointer */
                 :
                 :
}
                 :
                 :
```

3.2 Additional Functions Used by the Target System

Additional Functions available to the API-user are for both the OSI-, and TCP-environments.

The following additional functions are available to the API-user:

3.2.1 DbvAccessControlRequest()

Purpose:

Send Access Control Request APDU to Origin.

Function Name and Parameters:

```
DbvAccessControlRequest(
DbvAssocId                         lAssocId,                  /* IN */
DbvAccessControlReq         *psAccessControlReq,        /* IN */
DbvError                              **ppsSIError               /* OUT_F */
)
```

Return Value:

SI_OK:	if no error occured.
SI_NOTOK:	on error.

Parameter Description:

lAssocId	The unique identifier of the association, as received by a call of function DbvAssociateRequest().
psAccessControlReq	Pointer to the data to be transferred. The definition of the data type *DbvAccessControlReq* corresponds to the definition of the ACCESS CONTROL Request service primitive.
ppsSIError	The address of a pointer of type **DbvError*. When calling the function, the pointer should refer to (DbvError*)NULL. The pointer is updated only if the function returns SI_NOTOK. In this case the pointer refers to a filled-in *DbvError* structure providing information about the failure occured.

Function Description:

The caller of the function provides in parameter *psAccessControlReq* the data corresponding to an ACCESS CONTROL Request service primitive. The function checks the data for completeness, forms and submits an AccessControlRequest-APDU.

If the function returns SI_OK, the data had been submitted successfully.

If the function returns SI_NOTOK, the data had not been transferred. In this case the *DbvError* structure indicates the reason for the transfer failure.

Usage:

```
#include "si_defs.h"
#include "si_types.h"
#include "dbvproto.h"
#include "si_types2.h"
#include "iord_ext.h"
#include "itemordr.h"
DbvAssocId                     lAssocId = NO_ASSOCIATION_ID_ASSIGNED;
DbvAccessControlReq           *psAccessControlReq = NULL;
DbvError                      *psError = NULL;


/*
    Suppose the call to function DbvAssociateResponse()
    had been successful.
    Allocate buffer for psAccessControlReq and fill in structure.
*/


if (DbvAccessControlRequest(lAssocId, psAccessControlReq, &psError) == SI_OK)
{
   FreeAccessControlReq(&psAccessControlReq);
                 :
                 :
}
else
{
   PrivateErrorHandling(psError);
   FreeAccessControlReq(&psAccessControlReq);
   FreeErrorCode(&psSIError); /* release buffer, assign NULL pointer */
                 :
                 :
}
                 :
                 :
```

3.2.2 DbvResourceControlRequest()

Purpose:

Send Resource Control Request APDU to the Origin.

Function Name and Parameters:

```
DbvResourceControlRequest(
DbvAssocId                      lAssocId,                /* IN */
DbvResourceControlReq          *psResourceControlReq,    /* IN */
DbvError                      **ppsSIError               /* OUT_F */
)
```

Return Value:

SI_OK:	if no error occured.
SI_NOTOK:	on error.

Parameter Description:

lAssocId	The unique identifier of the association, as received by a call of function DbvAssociateRequest().
psResourceControlReq	Pointer to the data to be transferred. The definition of the data type *DbvResourceControlReq* corresponds to the definition of the RESOURCE CONTROL Request service primitive.
ppsSIError	The address of a pointer of type **DbvError*. When calling the function, the pointer should refer to (DbvError*)NULL. The pointer is updated only if the function returns SI_NOTOK. In this case the pointer refers to a filled-in *DbvError* structure providing information about the failure occured.

Function Description:

The caller of the function provides in parameter *psResourceControlReq* the data corresponding to an RESOURCE CONTROL Request service primitive. The function checks the data for completeness, forms and submits an ResourceControlRequest-APDU.

If the function returns SI_OK, the data had been submitted successfully.

If the function returns SI_NOTOK, the data had not been transferred. In this case the *DbvError* structure indicates the reason for the transfer failure.

Usage:

```
#include "si_defs.h"
#include "si_types.h"
#include "dbvproto.h"
#include "si_types2.h"
#include "iord_ext.h"
#include "itemordr.h"


DbvAssocId                     lAssocId = NO_ASSOCIATION_ID_ASSIGNED;
DbvResourceControlReq         *psResourceControlReq = NULL;
DbvError                      *psError = NULL;


/*
    Suppose the call to function DbvAssociateResponse()
    had been successful.
    Allocate buffer for psResourceControlReq and fill in structure.
*/


if (DbvResourceControlRequest(lAssocId, psResourceControlReq, &psError) ==
SI_OK)
{
   FreeResourceControlReq(&psResourceControlReq);
                 :
                 :
}
else
{
   PrivateErrorHandling(psError);
   FreeResourceControlReq(&psResourceControlReq);
   FreeErrorCode(&psSIError); /* release buffer, assign NULL pointer */
                 :
                 :
}
                 :
                 :
```

Annex A: Extensions to DBV-OSI II API Datatypes

These extensions are relating to the Access Control service, Resource Control service, Trigger Resource Control service, Explain Record and AccessControlPrompt-1 Record.

```
/* -----------  sitypes2.h --------------------------------------------*/



/* When using this file with other DBV-OSI II includes, include as follows


#include "sidefs.h"
#include "sitypes.h"
#include "dbvproto.h"
#include "sitypes2.h"
#include "iord_ext.h"
#include "itemordr.h"

*/



/*
   New Structure Definitions For ONE Extensions to DBV-OSI II
   -----------------------------------------------------------


   Definitions contained in this file:



   New PDU Definitions
   ===================


   DbvAccessControlReq
   DbvAccessControlRsp
   DbvResourceControlReq
   DbvResourceControlRsp
   DbvTriggerResourceControlReq
```

```
Notes:
1) All optional fields are pointers.
2) All choice enumerations start from 1.



New External Definitions
========================


AT_ExplainRecord
AT_GenericRecord

AT_AccessControlPrompt1
AT_AccessControlDes1
AT_AccessControlKrb1

AT_PerResSetExtendedService
AT_PerQueryExtendedService
AT_PerQuerySchExtendedService
AT_ItemOrderExtendedService (in itemordr.h)
AT_UpdateExtendedService (in sitypes.h)
AT_ExpSpecExtendedService
AT_ExpInvoExtendedService

AT_UserInfoSearchResult

AT_ElementSpec1

Notes:
1) All optional fields not in unions are pointers
(fields which are pointers to chars however
(pc) can be compulsory, depending on the field).
2) All choice enumerations start from 1.
3) Int enumerations are as defined in Z39.50 (some start from 0).
4) Pointers to DbvNull structures are declared as DbvNull.
5) DbvNull structures are not declared.
```

```
   6) All union members except enumerations, unsigned ints and bools are
   decalared as pointers (in keeping with sitypes.h style).
   7) Sequences are always represented as **, irrespective or being compulsory
   or optional.


   Structure ASN1External in sitypes.h has been updated to take account of
   the new types.
*/




/*  Module Name: @(#)sitypes2.h     4.7
    Version: 8/20/96
    Author: John Hough

    Crossnet Systems Ltd.
    Unit 3, The Paddock,
    Hambridge Road,
    Newbury,
    Berkshire, RG14 5TQ

*/

/*
    Type definitions for ONE extensions to DBV-OSI II
*/

/*
    Note:
    Optional fields are pointers, compulsory fields are non-pointers.

    but also..

    Lists (optional or compulsory) are always pointers (**).
```

```
    Union members which are structures are specified as pointers.

*/

#if !defined(SI_TYPES2_H)
#define SI_TYPES2_H


/* ------------------------------------------------------------*/
/*               Types and Sub-Structures                     */
/* ------------------------------------------------------------*/


/* -------------   DbvLanguageCode   ----------------------------- */

typedef DbvInternationalString DbvLanguageCode;


/* -------------   DbvNull   ----------------------------- */

typedef void* DbvNull;


/* -------------   DbvSecurityChallengeReq  ----------------------------- */

typedef enum DbvSecurityChallengeType {
                   SECT_SimpleForm        = 1,
                   SECT_ExternallyDefined = 2,
                   SECT_Invalid                  = -1
} DbvSecurityChallengeType;



typedef struct DbvSecurityChallengeReq
{
DbvSecurityChallengeType   eSecurityChallengeReqType;
```

```
union {
    DbvOctetString              *psSimpleForm;
    DbvASN1External             *psExternallyDefinedInfo;
  }                  uSecurityChallengeReq;
} DbvSecurityChallengeReq;
```

/* ------------- DbvSecurityChallengeRsp ----------------------------- */

```
typedef struct DbvSecurityChallengeRsp
{
DbvSecurityChallengeType   eSecurityChallengeRspType;
union {
    DbvOctetString              *psSimpleForm;
    DbvASN1External             *psExternallyDefinedInfo;
  }                  uSecurityChallengeRsp;
} DbvSecurityChallengeRsp;
```

```
/* -------------------------------------------------------------------*/
/*                  PDU-Datastructures      (Top-Level structures)    */
/* -------------------------------------------------------------------*/


/* ------------------    ACCESSCONTROL   --------------------------------- */

typedef struct DbvAccessControlReq
{
DbvOctetString               *psReferenceId;
DbvSecurityChallengeReq       sSecurityChallengeReq;
DbvOtherInformation         **ppsOtherInformation;
} DbvAccessControlReq;


typedef struct DbvAccessControlRsp
{
DbvOctetString               *psReferenceId;
DbvSecurityChallengeRsp      *psSecurityChallengeRsp;
DbvDiagRec                   *psDiagnostic;
DbvOtherInformation         **ppsOtherInformation;
} DbvAccessControlRsp;


/* ------------------    RESOURCECONTROL   --------------------------------- */

typedef enum DbvPartialResultsAvailable {
              PRA_Subset   = 1,
              PRA_Interim2 = 2,
              PRA_None2    = 3,
              PRA_Invalid  = -1
} DbvPartialResultsAvailable;



typedef struct DbvResourceControlReq
```

```
{
DbvOctetString               *psReferenceId;
DbvBool                      *pfSuspendedFlag;
DbvASN1External              *psResourceReport;
DbvPartialResultsAvailable   *pePartialResultsAvailable;
DbvBool                       fResponseRequired;
DbvBool                      *pfTriggeredRequestFlag;
DbvOtherInformation         **ppsOtherInformation;
} DbvResourceControlReq;



typedef struct DbvResourceControlRsp
{
DbvOctetString               *psReferenceId;
DbvBool                       fContinueFlag;
DbvBool                      *pfResultSetWanted;
DbvOtherInformation         **ppsOtherInformation;
} DbvResourceControlRsp;



/* -----------    TRIGGERRESOURCECONTROL   -------------------------- */


typedef enum DbvRequestedAction {
              RA_ResourceReport1   = 1,
              RA_ResourceControl   = 2,
              RA_Cancel            = 3,
              RA_Invalid           = -1
} DbvRequestedAction;



typedef struct DbvTriggerResourceControlReq
{
```

```
DbvOctetString               *psReferenceId;
DbvRequestedAction            eRequestedAction;
DbvObjectIdentifier           pcPreferredResourceReportFormat;
DbvBool                      *pfResultSetWanted;
DbvOtherInformation         **ppsOtherInformation;
} DbvTriggerResourceControlReq;



/* TriggerResourceControlRequest is sent from origin to target.
   There is no TriggerResourceControlResponse in Z39.50 V3 */
```

```
/* -----------------------------------------------------------*/
/*    Appendix REC Record Syntaxes                            */
/* -----------------------------------------------------------*/

/* -------------- Record Syntax "DbvExplainRecord" ----------------*/



typedef enum DbvExplainRecordType {
                   ERT_TargetInfo                = 1,
                   ERT_DatabaseInfo        = 2,
                   ERT_SchemaInfo          = 3,
                   ERT_TagSetInfo          = 4,
                   ERT_RecordSyntaxInfo          = 5,
                   ERT_AttributeSetInfo          = 6,
                   ERT_TermListInfo        = 7,
                   ERT_ExtendedServicesInfo    = 8,
                   ERT_AttributeDetails          = 9,
                   ERT_TermListDetails           = 10,
                   ERT_ElementSetDetails       = 11,
                   ERT_RetrievalRecordDetails  = 12,
                   ERT_SortDetails             = 13,
                   ERT_Processing              = 14,
                   ERT_Variants                = 15,
                   ERT_Units                   = 16,
                   ERT_CategoryList        = 17,
                   ERT_Invalid                 = -1
} DbvExplainRecordType;



/* -------------- DbvTargetInfo ----------------*/


typedef struct DbvCommonInfo
{
DbvGeneralizedTime            pcDateAdded;
DbvGeneralizedTime            pcDateChanged;
DbvGeneralizedTime            pcExpiry;
```

```
DbvLanguageCode              *psHumanStringLanguage;
DbvOtherInformation         **ppsOtherInformation;
} DbvCommonInfo;




typedef struct DbvHumanString
{
DbvLanguageCode          *psLanguage;
DbvInternationalString   sText;
} DbvHumanString;




typedef enum DbvBodyTypeType {
                   BTT_IanaType           = 1,
                   BTT_Z3950Type    = 2,
                   BTT_OtherType    = 3,
                   BTT_Invalid          = -1
} DbvBodyTypeType;




typedef struct DbvBodyType
{
DbvBodyTypeType   eBodyTypeType;
union {
    DbvInternationalString      *psIanaType;
    DbvInternationalString      *psZ3950Type;
    DbvInternationalString      *psOtherType;
  }                  uBodyType;
} DbvBodyType;




typedef struct DbvIconObject
```

```
{
DbvBodyType       sBodyType;
DbvOctetString    sContent;
} DbvIconObject;



typedef struct DbvContactInfo
{
DbvInternationalString    *psName;
DbvHumanString           **ppsDescription;
DbvHumanString           **ppsAddress;
DbvInternationalString    *psEmail;
DbvInternationalString    *psPhone;
} DbvContactInfo;



typedef struct DbvDatabaseList
{
DbvInternationalString **ppsInnerDatabases; /* mand. */
} DbvDatabaseList;




typedef enum DbvNetworkAddressType {
                   NAT_InternetAddress          = 1,
                   NAT_OsiPresentationAddress   = 2,
                   NAT_OtherAddress             = 3,
                   NAT_Invalid                  = -1
} DbvNetworkAddressType;




typedef struct DbvInterAddress
{
DbvInternationalString    sHostAddress;
Ulong                     ulPort;
```

```
} DbvInterAddress;




typedef struct DbvOsiPresentationAddress
{
DbvInternationalString    sPsel;
DbvInternationalString   *psSSel;
DbvInternationalString   *psTSel;
DbvInternationalString    sNSap;
} DbvOsiPresentationAddress;




typedef struct DbvOtherAddress
{
DbvInternationalString    sType;
DbvInternationalString    sAddress;
} DbvOtherAddress;




typedef struct DbvNetworkAddress
{
DbvNetworkAddressType   eNetworkAddressType;
union {
    DbvInterAddress                *psInternetAddress;
    DbvOsiPresentationAddress      *psOsiPresentationAddress;
    DbvOtherAddress                *psOtherAddress;
  }                  uNetworkAddress;
} DbvNetworkAddress;




typedef enum DbvQueryTypeDetailsType {
```

```
         QTDT_Private     = 1,
         QTDT_Rpn         = 2,
         QTDT_Iso8777     = 3,
         QTDT_Z39_58      = 4,
         QTDT_Erpn        = 5,
         QTDT_RankedList  = 6,
         QTDT_Invalid     = -1
} DbvQueryTypeDetailsType;




typedef struct DbvOperators
{
DbvInternationalString    sOperator;
DbvHumanString           **ppsDescription;
} DbvOperators;




typedef struct DbvSearchKey
{
DbvInternationalString    sSearchKey;
DbvHumanString           **ppsDescription;
} DbvSearchKey;




typedef struct DbvHumanStringSeq
{
DbvHumanString     **ppsHumanString;
} DbvHumanStringSeq;




typedef struct DbvPrivateCapabilities
{
```

```
DbvOperators       **ppsOperators;
DbvSearchKey       **ppsSearchKeys;
DbvHumanStringSeq  **ppsDescription;
} DbvPrivateCapabilities;


typedef enum DbvUnitsSupportedType {
       UST_Known   = 1,
       UST_Private = 2,
       UST_Invalid = -1
} DbvUnitsSupportedType;


typedef struct DbvPrivate
{
Ulong           ulUnit;
DbvHumanString **ppsDescription;
} DbvPrivate;



typedef struct DbvUnitsSupported
{
DbvUnitsSupportedType eUnitsSupportedType;
union {
   Ulong        ulKnown;
   DbvPrivate *psPrivate;
  } uUnitsSupported;
} DbvUnitsSupported;



typedef struct DbvProximitySupport
{
DbvBool               fAnySupport;
DbvUnitsSupported  **ppsUnitsSupported;
```

```
} DbvProximitySupport;




typedef struct DbvRpnCapabilities
{
Ulong              **ppulOperators;
DbvBool              fResultSetAsOperandSupported;
DbvBool              fRestrictionOperandSupported;
DbvProximitySupport *psProximity;
} DbvRpnCapabilities;




typedef struct DbvIso8777Capabilities
{
DbvSearchKey     **ppsSearchKeys;
DbvHumanString   **ppsRestrictions;
} DbvIso8777Capabilities;




typedef struct DbvQueryTypeDetails
{
DbvQueryTypeDetailsType     eQueryTypeDetailsType;
union {
     DbvPrivateCapabilities  *psPrivate;
     DbvRpnCapabilities      *psRpn;
     DbvIso8777Capabilities  *psIso8777;
     DbvHumanString         **ppsZ39_58;
     DbvRpnCapabilities      *psErpn;
     DbvHumanString         **ppsRankedList;
  } uQueryTypeDetails;
} DbvQueryTypeDetails;
```

```
typedef enum DbvAccessType {
    AT_Any              = 0,
    AT_Search           = 1,
    AT_Present          = 2,
    AT_SpecificElements = 3,
    AT_ExtendedServices = 4,
    AT_ByDatabase       = 5,
    AT_Invalid          = -1
} DbvAccessType;
```

```
typedef struct DbvAccessRestrictions
{
DbvAccessType            eAccessType;
DbvHumanString         **ppsAccessText;
DbvObjectIdentifier     *ppcAccessChallenges;
} DbvAccessRestrictions;
```

```
typedef struct DbvCharge
{
DbvIntUnit       sCost;
DbvUnit         *psPerWhat;
DbvHumanString **ppsText;
} DbvCharge;
```

```
typedef struct DbvOtherCharge
{
DbvHumanString **ppsForWhat;
DbvCharge        sCharge;
} DbvOtherCharge;
```

```
typedef struct DbvCosts
{
DbvCharge       *psConnectCharge;
DbvCharge       *psConnectTime;
DbvCharge       *psDisplayCharge;
DbvCharge       *psSearchCharge;
DbvCharge       *psSubscriptCharge;
DbvOtherCharge **ppsOtherCharge;
} DbvCosts;


/* A ** is not needed for ppcDiagnosticsSets etc. because
   DbvObjectIdentifier is a typedefed char * (see sitypes.h) */

typedef struct DbvAccessInfo
{
DbvQueryTypeDetails           **ppsQueryTypesSupported;
DbvObjectIdentifier             *ppcDiagnosticsSets;
DbvObjectIdentifier             *ppcAttributeSetIds;
DbvObjectIdentifier             *ppcSchemas;
DbvObjectIdentifier             *ppcRecordSyntaxes;
DbvObjectIdentifier             *ppcResourceChallenges;
DbvAccessRestrictions         **ppsRestrictedAccess;
DbvCosts                        *psCostInfo;
DbvObjectIdentifier             *ppcVariantSets;
DbvInternationalString        **ppsElementSetNames;
DbvInternationalString        **ppsUnitSystems;
} DbvAccessInfo;



typedef struct DbvTargetInfo
{
```

```
DbvCommonInfo              *psCommonInfo;
DbvInternationalString      sName;
DbvHumanString            **ppsRecentNews;
DbvIconObject             **ppsIcon;
DbvBool                     fNamedResultSets;
DbvBool                     fMultipleDBSearch;
Ulong                      *pulMaxResultSets;
Ulong                      *pulMaxResultSize;
Ulong                      *pulMaxTerms;
DbvIntUnit                 *psTimeoutInterval;
DbvHumanString            **ppsWelcomeMessage;
DbvContactInfo             *psContactInfo;
DbvHumanString            **ppsDescription;
DbvInternationalString    **ppsNicknames;
DbvHumanString            **ppsUsageRest;
DbvHumanString            **ppsPaymentAddr;
DbvHumanString            **ppsHours;
DbvDatabaseList           **ppsDbCombinations;
DbvNetworkAddress         **ppsAddresses;
DbvAccessInfo              *psCommonAccessInfo;
} DbvTargetInfo;


/* -------------- DbvDatabaseInfo ----------------*/

typedef enum DbvRecordCountType {
     RCT_ActualNumber = 1,
     RCT_ApproxNumber = 2,
     RCT_Invalid      = -1
} DbvRecordCountType;


typedef struct DbvRecordCount
{
DbvRecordCountType eRecordCountType;
union {
```

```
     Ulong ulActualNumber;
     Ulong ulApproxNumber;
  } uRecordCount;
} DbvRecordCount;



typedef struct DbvDataBaseInfoSeq
{
DbvHumanString           **ppsDataBaseInfo;
} DbvDataBaseInfoSeq;



typedef struct DbvDatabaseInfo
{
DbvCommonInfo             *psCommonInfo;
DbvInternationalString     sName;
DbvNull                    pvExplainDatabase; /* Set to NULL or
                                     1 alloced byte */
DbvInternationalString   **ppsNicknames;
DbvIconObject            **ppsIcon;
DbvBool                    fUserFee;
DbvBool                    fAvailable;
DbvHumanString           **ppsTitleString;
DbvDataBaseInfoSeq       **ppsKeywords;
DbvHumanString           **ppsDescription;
DbvInternationalString   **ppsAssociatedDbs;
DbvInternationalString   **ppsSubDbs;
DbvHumanString           **ppsDisclaimers;
DbvHumanString           **ppsNews;
DbvRecordCount            *psRecordCount;
DbvHumanString           **ppsDefaultOrder;
Ulong                     *pulAvRecordSize;
Ulong                     *pulMaxRecordSize;
DbvHumanString           **ppsHours;
DbvHumanString           **ppsBestTime;
DbvGeneralizedTime         pcLastUpdate;
```

```
DbvIntUnit                *psUpdateInterval;
DbvHumanString           **ppsCoverage;
DbvBool                   *pfProprietary;
DbvHumanString           **ppsCopyrightText;
DbvHumanString           **ppsCopyrightNotice;
DbvContactInfo            *psProducerContactInfo;
DbvContactInfo            *psSupplierContactInfo;
DbvContactInfo            *psSubmissionContactInfo;
DbvAccessInfo             *psAccessInfo;
} DbvDatabaseInfo;


/* -------------- DbvSchemaInfo ----------------*/

typedef struct DbvTagTypeMapping {
Ulong                   ulTagType;
DbvObjectIdentifier     pcTagSet;
DbvNull                 pvDefaultTagType; /* Set to either NULL or
                                   1 alloced byte  */
} DbvTagTypeMapping;



typedef struct DbvPath {
Ulong              ulTagType;
DbvStringOrNumeric sTagValue;
} DbvPath;



typedef struct DbvElementInfo {
DbvInternationalString      sElementName;
DbvPath                   **ppsElementTagPath;
struct DbvElementDataType   *psDataType;
DbvBool                     fRequired;
DbvBool                     fRepeatable;
```

```
DbvHumanString             **ppsDescription;
} DbvElementInfo;




typedef enum DbvElementDataTypeType {
   EDTT_Primitive  = 1,
   EDTT_Structured = 2,
   EDTT_Invalid    = -1
} DbvElementDataTypeType;




typedef enum DbvPrimitiveDataType {
   PDT_OctetString     = 0,
   PDT_Numeric         = 1,
   PDT_Date            = 2,
   PDT_External        = 3,
   PDT_String          = 4,
   PDT_TrueOrFalse     = 5,
   PDT_Oid             = 6,
   PDT_IntUnit         = 7,
   PDT_Empty           = 8,
   PDT_NoneOfTheAbove  = 100,
   PDT_Invalid         = -1
} DbvPrimitiveDataType;




typedef struct DbvElementDataType {
DbvElementDataTypeType eElementDataTypeType;
union {
     DbvPrimitiveDataType       ePrimitive;
     DbvElementInfo           **ppsStructured;
  } uElementDataType;
} DbvElementDataType;
```

```
typedef struct DbvSchemaInfo {
DbvCommonInfo           *psCommonInfo;
DbvObjectIdentifier      pcSchema;
DbvInternationalString   sName;
DbvHumanString         **ppsDescription;
DbvTagTypeMapping      **ppsTagTypeMapping;
DbvElementInfo         **ppsRecordStructure;
} DbvSchemaInfo;


/* -------------- DbvTagSetInfo ----------------*/

typedef struct DbvElements
{
DbvInternationalString   sElementName;
DbvInternationalString **ppsNicknames;
DbvStringOrNumeric       sElementTag;
DbvHumanString         **ppsDescription;
DbvPrimitiveDataType    *peDataType;
DbvOtherInformation    **ppsOtherInformation;
} DbvElements;



typedef struct DbvTagSetInfo
{
DbvCommonInfo           *psCommonInfo;
DbvObjectIdentifier      pcTagSet;
DbvInternationalString   sName;
DbvHumanString         **ppsDescription;
DbvElements            **ppsElements;
} DbvTagSetInfo;
```

```
/* -------------- DbvRecordSyntaxInfo ----------------*/

typedef struct DbvRecordSyntaxInfo
{
DbvCommonInfo           *psCommonInfo;
DbvObjectIdentifier      pcRecordSyntax;
DbvInternationalString   sName;
DbvObjectIdentifier     *ppcTransferSyntaxes;
DbvHumanString         **ppsDescription;
DbvInternationalString  *psAsn1Module;
DbvElementInfo         **ppsAbstractStructure;
} DbvRecordSyntaxInfo;


/* -------------- DbvAttributeSetInfo ----------------*/

typedef struct DbvAttributeDescription
{
DbvInternationalString  *psName;
DbvHumanString         **ppsDescription;
DbvStringOrNumeric       sAttributeValue;
DbvStringOrNumeric     **ppsEquivalentAttributes;
} DbvAttributeDescription;



typedef struct DbvAttributeType
{
DbvInternationalString    *psName;
DbvHumanString           **ppsDescription;
Ulong                      ulAttributeType;
DbvAttributeDescription **ppsAttributeValues;
} DbvAttributeType;
```

```
typedef struct DbvAttributeSetInfo
{
DbvCommonInfo               *psCommonInfo;
DbvObjectIdentifier          pcAttributeSetId;
DbvInternationalString       sName;
DbvAttributeType           **ppsAttributes;
DbvHumanString             **ppsDescription;
} DbvAttributeSetInfo;



/* -------------- DbvTermListInfo ----------------*/

typedef enum DbvSearchCost {
   SC_Optimized = 0,
   SC_Normal    = 1,
   SC_Expensive = 2,
   SC_Filter    = 3,
   SC_Invalid   = -1
} DbvSearchCost;



typedef struct DbvTermList
{
DbvInternationalString    sName;
DbvHumanString          **ppsTitle;
DbvSearchCost            *peSearchCost;
DbvBool                   fScanable;
DbvInternationalString  **ppsBroader;
DbvInternationalString  **ppsNarrower;
} DbvTermList;



typedef struct DbvTermListInfo
```

```
{
DbvCommonInfo          *psCommonInfo;
DbvInternationalString  sDatabaseName;
DbvTermList           **ppsTermLists;
} DbvTermListInfo;



/* -------------- DbvExtendedServicesInfo ----------------*/

typedef enum DbvExtendedServicesInfoInt
{
   ESII_WaitSupported    = 1,
   ESII_WaitAlways       = 2,
   ESII_WaitNotSupported = 3,
   ESII_Depends          = 4,
   ESII_NotSaying        = 5,
   ESII_Invalid          = -1
} DbvExtendedServicesInfoInt;




typedef struct DbvExtendedServicesInfo
{
DbvCommonInfo               *psCommonInfo;
DbvObjectIdentifier          pcType;
DbvInternationalString      *psName;
DbvBool                      fPrivateType;
DbvBool                      fRestrictionsApply;
DbvBool                      fFeeApply;
DbvBool                      fAvailable;
DbvBool                      fRetentionSupported;
DbvExtendedServicesInfoInt   eWaitAction;
DbvHumanString             **ppsDescription;
DbvASN1External             *psSpecificExplain;
DbvInternationalString      *psEsASN;
} DbvExtendedServicesInfo;
```

```
/* -------------- DbvAttributeDetails ----------------*/

typedef struct DbvOmittedAttributeInterpretation
{
DbvStringOrNumeric  *psDefaultValue;
DbvHumanString     **ppsDefaultDescription;
} DbvOmittedAttributeInterpretation;



typedef struct DbvAttributeValues
{
DbvStringOrNumeric    sValue;
DbvHumanString      **ppsDescription;
DbvStringOrNumeric  **ppsSubAttributes;
DbvStringOrNumeric  **ppsSuperAttributes;
DbvNull               pvPartialSupport;  /* Set to either NULL or
                                  1 alloced byte */
} DbvAttributeValues;



typedef struct DbvAttributesByType
{
Ulong                              ulAttributeType;
DbvOmittedAttributeInterpretation *psOptionalType;
DbvAttributeValues               **ppsAttributeValues;
} DbvAttributesByType;


typedef struct DbvAttributesBySet
{
DbvObjectIdentifier       pcAttributeSet;
DbvAttributesByType     **ppsAttributesByType;
```

```
} DbvAttributesBySet;



typedef enum DbvAttributeOccurrenceChoiceType {
    AOT_AnyOrNone             = 1,
    AOT_Specific              = 2,
    AOT_Invalid               = -1
} DbvAttributeOccurrenceChoiceType;




typedef struct DbvAttributeOccurrenceChoice
{
DbvAttributeOccurrenceChoiceType   eAttributeOccurrenceChoiceType;
union {
     DbvNull               pvAnyOrNone;  /* Set to either NULL or
                                  1 alloced byte */
     DbvStringOrNumeric  **ppsSpecific;
  } uAttributeOccurrenceChoiceType;
} DbvAttributeOccurrenceChoice;




typedef struct DbvAttributeOccurrence
{
DbvObjectIdentifier                 pcAttributeSet;
Ulong                               ulAttributeType;
DbvNull                             pvMustBeSupplied;
DbvAttributeOccurrenceChoice        sAttributeValues;
} DbvAttributeOccurrence;




/* Note that the following typedef is DbvAttributeCombination, typedef
   DbvAttributeCombinations (which is different [extra 's' at the end]
   follows later */
```

```
typedef struct DbvAttributeCombination
{
DbvAttributeOccurrence           **ppsAttributeOccurrence;
} DbvAttributeCombination;


typedef struct DbvAttributeCombinations
{
DbvObjectIdentifier                pcDefaultAttributeSet;
DbvAttributeCombination          **ppsLegalCombinations;
} DbvAttributeCombinations;



typedef struct DbvAttributeDetails
{
DbvCommonInfo            *psCommonInfo;
DbvInternationalString    sDatabaseName;
DbvAttributesBySet      **ppsAttributesBySet;
DbvAttributeCombinations *psAttributeCombinations;
} DbvAttributeDetails;


/* -------------- DbvTermListDetails ----------------*/

typedef struct DbvScanInfo
{
Ulong           *pulMaxStepSize;
DbvHumanString **ppsCollatingSequence;
DbvBool         *pfIncreasing;
} DbvScanInfo;



typedef struct DbvTermListDetails
```

```
{
DbvCommonInfo             *psCommonInfo;
DbvInternationalString     sTermListName;
DbvHumanString           **ppsDescription;
DbvAttributeCombinations  *psAttributes;
DbvScanInfo               *psScanInfo;
Ulong                     *pulEstNumberTerms;
DbvTerm                  **ppsSampleTerms;
} DbvTermListDetails;


/* -------------- DbvElementSetDetails ----------------*/

typedef struct DbvRecordTag
{
DbvStringOrNumeric *psQualifier;
DbvStringOrNumeric  sTagValue;
} DbvRecordTag;



typedef struct DbvPathA {
DbvPath                  **ppsSchemaTags;
} DbvPathA;

typedef struct DbvDetailsPerElement
{
DbvInternationalString    *psName;
DbvRecordTag              *psRecordTag;
DbvPathA                 **ppsSchemaTags;
Ulong                     *pulMaxSize;
Ulong                     *pulMinSize;
Ulong                     *pulAvgSize;
Ulong                     *pulFixedSize;
DbvBool                    fRepeatable;
DbvBool                    fRequired;
DbvHumanString           **ppsDescription;
```

```
DbvHumanString           **ppsContents;
DbvHumanString           **ppsBillingInfo;
DbvHumanString           **ppsRestrictions;
DbvInternationalString   **ppsAlternateNames;
DbvInternationalString   **ppsGenericNames;
DbvAttributeCombinations  *psSearchAccess;
} DbvDetailsPerElement;




typedef struct DbvElementSetDetails
{
DbvCommonInfo             *psCommonInfo;
DbvInternationalString     sDatabaseName;
DbvInternationalString     sElementSetName;
DbvObjectIdentifier        pcRecordSyntax;
DbvObjectIdentifier        pcSchema;
DbvHumanString           **ppsDescription;
DbvDetailsPerElement     **ppsDetailsPerElement;
} DbvElementSetDetails;


/* -------------- DbvRetrievalRecordDetails ----------------*/


typedef struct DbvRetrievalRecordDetails
{
DbvCommonInfo            *psCommonInfo;
DbvInternationalString    sDatabaseName;
DbvObjectIdentifier       pcSchema;
DbvObjectIdentifier       pcRecordSyntax;
DbvHumanString          **ppsDescription;
DbvDetailsPerElement    **ppsDetailsPerElement;
} DbvRetrievalRecordDetails;
```

```
/* -------------- DbvSortDetails ----------------*/

typedef enum DbvSortTypeType {
     STT_Character  = 1,
     STT_Numeric    = 2,
     STT_Structured = 3,
     STT_Invalid    = -1
} DbvSortTypeType;




typedef struct DbvSortType
{
DbvSortTypeType       eSortTypeType;
union {
    DbvHumanString **ppsStructured; /* Set if eSortTypeType == STT_Structured */
     } uSortType;
} DbvSortType;




typedef enum DbvCaseSensitivity
{
    CSE_Always     = 0,
    CSE_Never      = 1,
    CSE_DefaultYes = 2,
    CSE_DefaultNo  = 3,
    CSE_Invalid    = -1
} DbvCaseSensitivity;




typedef struct DbvSortKeys
{
DbvHumanString           **ppsDescription;
DbvSpecification         **ppsElementSpecifications;
```

```
DbvAttributeCombinations  *psAttributeSpecifications;
DbvSortType               *psSortType;
DbvCaseSensitivity        *peCaseSensitivity;
} DbvSortKeys;



typedef struct DbvSortDetails
{
DbvCommonInfo            *psCommonInfo;
DbvInternationalString    sDatabaseName;
DbvSortKeys             **ppsSortKeys;
} DbvSortDetails;



/* -------------- DbvProcessingInformation ----------------*/

typedef enum DbvProcessingContext
{
    PC_Access            = 0,
    PC_Search            = 1,
    PC_Retrieval         = 2,
    PC_RecordPresentation = 3,
    PC_RecordHandling    = 4,
    PC_Invalid           = -1
} DbvProcessingContext;



typedef struct DbvProcessingInformation
{
DbvCommonInfo             *psCommonInfo;
DbvInternationalString     sDatabaseName;
DbvProcessingContext       eProcessingContext;
DbvInternationalString     sName;
DbvObjectIdentifier        pcOid;
DbvHumanString           **ppsDescription;
DbvASN1External           *psInstructions;
```

```
} DbvProcessingInformation;



/* -------------- DbvVariantSetInfo ----------------*/

typedef enum DbvValueSetType {
     VT_Range      = 1,
     VT_Enumerated = 2,
     VT_Invalid    = -1
} DbvValueSetType;



typedef enum DbvValueDescriptionType {
    VDT_Integer       = 1,
    VDT_String        = 2,
    VDT_Octets        = 3,
    VDT_Oid           = 4,
    VDT_Unit          = 5,
    VDT_ValueAndUnit  = 6,
    VDT_Invalid       = -1
} DbvValueDescriptionType;




typedef struct DbvValueDescription
{
DbvValueDescriptionType eValueDescriptionType;
union {
     Ulong                   ulInteger;
     DbvInternationalString *psString;
     DbvOctetString         *psOctets;
     DbvObjectIdentifier     pcOid;
     DbvUnit                *psUnit;
     DbvIntUnit             *psValueAndUnit;
 } uValueDescription;
} DbvValueDescription;
```

```
typedef struct DbvValueRange
{
DbvValueDescription *psLower;
DbvValueDescription *psUpper;
} DbvValueRange;
```

```
typedef struct DbvValueSet
{
DbvValueSetType   eValueSetType;
union {
     DbvValueRange         *psRange;
     DbvValueDescription  **ppsEnumerated;
     } uValueSet;
} DbvValueSet;
```

```
typedef struct DbvVariantValue
{
DbvPrimitiveDataType  eDataType;
DbvValueSet          *psValues;
} DbvVariantValue;
```

```
typedef struct DbvVariantTypes
{
DbvInternationalString   *psName;
DbvHumanString          **ppsDescription;
Ulong                     ulVariantType;
DbvVariantValue          *psVariantValue;
```

```
} DbvVariantTypes;




typedef struct DbvVariantClass
{
DbvInternationalString  *psName;
DbvHumanString         **ppsDescription;
Ulong                    ulVariantClass;
DbvVariantTypes        **ppsVariantTypes;
} DbvVariantClass;




typedef struct DbvVariantSetInfo
{
DbvCommonInfo           *psCommonInfo;
DbvObjectIdentifier      pcVariantSet;
DbvInternationalString   sName;
DbvVariantClass        **ppsVariants;
} DbvVariantSetInfo;


/* -------------- DbvUnitInfo ----------------*/


typedef struct DbvUnits
{
DbvInternationalString  *psName;
DbvHumanString         **ppsDescription;
DbvStringOrNumeric       sUnit;
} DbvUnits;


typedef struct DbvUnitType
{
```

```
DbvInternationalString  *psName;
DbvHumanString         **ppsDescription;
DbvStringOrNumeric       sUnitType;
DbvUnits               **ppsUnits;
} DbvUnitType;



typedef struct DbvUnitInfo
{
DbvCommonInfo           *psCommonInfo;
DbvInternationalString   sUnitSystem;
DbvHumanString         **ppsDescription;
DbvUnitType            **ppsUnitTypes;
} DbvUnitInfo;



/* -------------- DbvCategoryList ----------------*/

typedef struct DbvCategoryInfo
{
DbvInternationalString   sCategory;
DbvInternationalString  *psOriginalCategory;
DbvHumanString         **ppsDescription;
DbvInternationalString  *psAsn1Module;
} DbvCategoryInfo;



typedef struct DbvCategoryList
{
DbvCommonInfo     *psCommonInfo;
DbvCategoryInfo  **ppsCategories;
} DbvCategoryList;




/* ------------------------------------------------------------*/
```

```
typedef struct DbvExplainRecord
{
DbvExplainRecordType   eExplainRecordType;
union {
    DbvTargetInfo               *psTargetInfo;
    DbvDatabaseInfo             *psDatabaseInfo;
    DbvSchemaInfo               *psSchemaInfo;
    DbvTagSetInfo               *psTagSetInfo;
    DbvRecordSyntaxInfo         *psRecordSyntaxInfo;
    DbvAttributeSetInfo         *psAttributeSetInfo;
    DbvTermListInfo             *psTermListInfo;
    DbvExtendedServicesInfo     *psExtendedServicesInfo;
    DbvAttributeDetails         *psAttributeDetails;
    DbvTermListDetails          *psTermListDetails;
    DbvElementSetDetails        *psElementSetDetails;
    DbvRetrievalRecordDetails   *psRetrievalRecordDetails;
    DbvSortDetails              *psSortDetails;
    DbvProcessingInformation    *psProcessing;
    DbvVariantSetInfo           *psVariants;
    DbvUnitInfo                 *psUnits;
    DbvCategoryList             *psCategoryList;
  }                    uExplainRecord;
} DbvExplainRecord;



/* -------------- Record Syntax "GenericRecord" ----------------*/

typedef enum DbvElementDataGType {
    EDT_Octets               = 1,
    EDT_Numeric              = 2,
    EDT_Date                 = 3,
    EDT_Ext                  = 4,
    EDT_String               = 5,
    EDT_TrueOrFalse          = 6,
```

```
    EDT_Oid                 = 7,
    EDT_IntUnit             = 8,
    EDT_ElementNotThere     = 9,
    EDT_ElementEmpty        = 10,
    EDT_NoDataRequested     = 11,
    EDT_Diagnostic          = 12,
    EDT_Subtree             = 13,
    EDT_Invalid             = -1
} DbvElementDataGType;


typedef struct DbvOrder
{
DbvBool fAscending;
Ulong   ulOrder;
} DbvOrder;


typedef enum DbvUsageIntType {
   UIT_Redistributable = 1,
   UIT_Restricted      = 2,
   UIT_LicensePointer  = 3,
   UIT_Invalid         = -1
} DbvUsageIntType;



typedef struct DbvUsage
{
DbvUsageIntType         eType;
DbvInternationalString *psRestriction;
} DbvUsage;


typedef struct DbvHits
{
```

```
DbvTerm        *psSatisfier;
DbvIntUnit     *psOffsetIntoElement;
DbvIntUnit     *psLength;
Ulong          *pulHitRank;
DbvOctetString *psTargetToken;
} DbvHits;



typedef enum DbvVariantValueType {
    VVT_Int                 = 1,
    VVT_InternationalString = 2,
    VVT_Octs                = 3,
    VVT_Oid                 = 4,
    VVT_Bool                = 5,
    VVT_Null                = 6,
    VVT_Unit                = 7,
    VVT_ValueAndUnit        = 8,
    VVT_Invalid             = -1
} DbvVariantValueType;



typedef struct DbvValue
{
DbvVariantValueType eVariantValueType;
union {
     Ulong                   ulInt1;
     DbvInternationalString *psInternationalString;
     DbvOctetString         *psOcts;
     DbvObjectIdentifier     pcOid;
     DbvBool                 fBool;
     DbvUnit                *psUnit;
     DbvIntUnit             *psValueAndUnit;
   } uVariantValueType;
} DbvValue;
```

```
typedef struct DbvTriples
{
DbvObjectIdentifier  pcVariantSetId;
Ulong                ulClass;
Ulong                ulType;
DbvValue           **ppsValue;
} DbvTriples;
```

```
typedef struct DbvVariant
{
DbvObjectIdentifier  pcGlobalVariantSetId;
DbvTriples         **ppsTriples;
} DbvVariant;
```

```
typedef struct DbvTagPath
{
Ulong               *pulTagType;
DbvStringOrNumeric   sTagValue;
Ulong               *pulTagOccurrence;
} DbvTagPath;
```

```
typedef struct DbvElementMetaData
{
DbvOrder                *psSeriesOrder;
DbvUsage                *psUsageRight;
DbvHits                **ppsHits;
DbvInternationalString  *psDisplayName;
DbvVariant             **ppsSupportedVariants;
DbvInternationalString  *psMessage;
```

```
DbvOctetString          *psElementDescriptor;
DbvTagPath             **ppsSurrogateFor;
DbvTagPath             **ppsSurrogateElement;
DbvASN1External         *psOther;
} DbvElementMetaData;
```

```
typedef struct DbvElementDataG
{
DbvElementDataGType   eElementDataGType;
union {
     DbvOctetString            *psOctets;
     Ulong                      ulNumeric;
     DbvGeneralizedTime         pcDate;
     DbvASN1External           *psExt;
     DbvInternationalString    *psString;
     DbvBool                    fTrueOrFalse;
     DbvObjectIdentifier        pcOid;
     DbvIntUnit                *psIntUnit;
     DbvASN1External           *psDiagnostic;
     struct DbvTaggedElement  **ppsSubTree;
   } uElementDataG;
} DbvElementDataG;
```

```
typedef struct DbvTaggedElement
{
Ulong                   *pulTagType;
DbvStringOrNumeric       sTagValue;
Ulong                   *pulTagOccurrence;
DbvElementDataG          sContent;
DbvElementMetaData      *psMetaData;
```

```
DbvVariant              *psAppliedVariant;
} DbvTaggedElement;


typedef struct DbvGenericRecord
{
DbvTaggedElement        **ppsTaggedElement;
} DbvGenericRecord;
```

```
/* -----------------------------------------------------------*/
/*    Appendix ACC Access Control Formats                     */
/* -----------------------------------------------------------*/



/* -------------- Access Control Format "Prompt 1" ----------------*/



typedef enum DbvPromptObjectType {
    POT_Challenge = 1,
    POT_Response  = 2,
    POT_Invalid   = -1
} DbvPromptObjectType;




typedef enum DbvPromptIdType {
    PIT_EnumeratedPrompt     = 1,
    PIT_NoneEnumeratedPrompt = 2,
    PIT_Invalid              = -1
} DbvPromptIdType;




typedef enum DbvEPromptIdType {
    EPIT_GroupId     = 0,
    EPIT_UserId      = 1,
    EPIT_Password    = 2,
    EPIT_NewPassword = 3,
    EPIT_Copyright   = 4,
    EPIT_SessionId   = 5,
    EPIT_Invalid     = -1
} DbvEPromptIdType;


typedef struct DbvEnumeratedPrompt
```

```
{
DbvEPromptIdType          eType;
DbvInternationalString  *psSuggestedString;
} DbvEnumeratedPrompt;
```

```
typedef struct DbvPromptId
{
DbvPromptIdType ePromptIdType;
union {
     DbvEnumeratedPrompt    *psEnumeratedPrompt;
     DbvInternationalString *psNonEnumeratedPrompt;
  } uPromptId;
} DbvPromptId;
```

```
typedef enum DbvPromptInfoType
{
    PIIT_Character = 1,
    PIIT_Encrypted = 2,
    PIIT_Invalid   = -1
} DbvPromptInfoType;
```

```
typedef struct DbvEncryption
{
DbvOctetString *psCryptType;
DbvOctetString *psCredential;
DbvOctetString  sData;
} DbvEncryption;
```

```
typedef struct DbvPromptInfo
{
DbvPromptInfoType ePromptInfoType;
union {
     DbvInternationalString *psCharacter;
     DbvEncryption          *psEncrypted;
  } uPromptInfo;
} DbvPromptInfo;
```

```
typedef enum DbvDataType {
   DTTT_Integer      = 1,
   DTTT_Date1        = 2,
   DTTT_Float        = 3,
   DTTT_AlphaNumeric = 4,
   DTTT_UrlUrn       = 5,
   DTTT_Boolean      = 6,
   DTTT_Invalid      = -1
} DbvDataType;
```

```
typedef struct DbvChallenge
{
DbvPromptId               sPromptId;
DbvInternationalString  *psDefaultResponse;
DbvPromptInfo           *psPromptInfo;
DbvInternationalString  *psRegExpr;
DbvNull                   pvResponseRequired;
DbvInternationalString **ppsAllowedValues;
DbvNull                   pvShouldSave;
DbvDataType             *peDataType;
DbvASN1External         *psDiagnostic;
} DbvChallenge;
```

```
typedef enum DbvResponseType {
   RT_String      = 1,
   RT_Accept      = 2,
   RT_Acknowledge = 3,
   RT_Diagnostic  = 4,
   RT_Encrypted   = 5,
   RT_Invalid    = -1
} DbvResponseType;




/* NOTE: No union member for RT_Acknowledge */
typedef struct DbvPromptResponse
{
DbvResponseType eResponseType;
union {
     DbvInternationalString *psString;      /* RT_String */
     DbvBool                 fAccept;       /* RT_Accept */
     DbvDiagRec             *psDiagnostic;  /* RT_Diagnostic */
     DbvEncryption          *psEncryption;  /* RT_Encrypted */
  } uResponse;
} DbvPromptResponse;




typedef struct DbvResponse
{
DbvPromptId       sPromptId;
DbvPromptResponse sPromptResponse;
} DbvResponse;




typedef struct DbvPromptObject
{
DbvPromptObjectType ePromptObjectType;
```

```
union {
     DbvChallenge **ppsChallenge;
     DbvResponse  **ppsResponse;
   } uPromptObject;
} DbvPromptObject;


/* -------------- Access Control Format "Des 1" ----------------*/


typedef enum DbvDES_RN_ObjectType {
    DROT_Challenge = 1,
    DROT_Response  = 2,
    DROT_Invalid  = -1
} DbvDES_RN_ObjectType;


typedef struct DbvDRNType
{
DbvOctetString  *psUserId;
DbvOctetString  *psSalt;
DbvOctetString  sRandomNumber;
} DbvDRNType;


typedef struct DbvDES_RN_Object
{
DbvDES_RN_ObjectType eDES_RN_ObjectType;
union {
     DbvDRNType *psChallenge;
     DbvDRNType *psResponse;
  } uDES_RN_Object;
} DbvDES_RN_Object;


/* -------------- Access Control Format "Krb 1" ----------------*/
```

```
typedef enum DbvKrbObjectType {
   KOT_Challenge = 1,
   KOT_Response  = 2,
   KOT_Invalid   = -1
} DbvKrbObjectType;
```

```
typedef struct DbvKRBRequest
{
DbvInternationalString  sService;
DbvInternationalString *psInstance;
DbvInternationalString *psRealm;
} DbvKRBRequest;
```

```
typedef struct DbvKRBResponse
{
DbvInternationalString *psUserId;
DbvOctetString          sTicket;
} DbvKRBResponse;
```

```
typedef struct DbvKrbObject {
DbvKrbObjectType eKrbObjectType;
union {
     DbvKRBRequest  *psChallenge;
     DbvKRBResponse *psResponse;
   } uKrbObject;
} DbvKrbObject;
```

```
/* -----------------------------------------------------------*/
/*    Appendix EXT Extended Services                          */
/* -----------------------------------------------------------*/

/* -------------  "Persistent Result Set" Extended Service ---------------- */



typedef enum DbvPerResSetReqOrRsp {
                    PRSROR_Request  = 1,
                    PRSROR_Response = 2,
                    PRSROR_Invalid  = -1
} DbvPerResSetReqOrRsp;




typedef enum DbvReplaceOrAppendType
{
   ROAT_Replace = 1,
   ROAT_Append  = 2,
   ROAT_Invalid = -1
} DbvReplaceOrAppendType;




typedef struct DbvPerResSetOriginPartNotToKeep
{
DbvInternationalString    *psOriginSuppliedResultSet;
DbvReplaceOrAppendType    *peReplaceOrAppend;
} DbvPerResSetOriginPartNotToKeep;



typedef struct DbvPerResSetRequestPackage
{
DbvPerResSetOriginPartNotToKeep *psNotToKeep;
} DbvPerResSetRequestPackage;
```

```
typedef struct DbvPerResSetTargetPart
{
DbvInternationalString *psTargetSuppliedResultSet;
Ulong                  *pulNumberOfRecords;
} DbvPerResSetTargetPart;




typedef struct DbvPerResSetTaskPackage
{
DbvPerResSetTargetPart *psTargetPart;
} DbvPerResSetTaskPackage;




typedef struct DbvPerResSetExtendedService
{
DbvPerResSetReqOrRsp ePerResSetReqOrRsp;
union {
    DbvPerResSetRequestPackage    *psPerResSetReqRequest;
    DbvPerResSetTaskPackage       *psPerResSetReqResponse;
  } uPerResSetReqOrRsp;
} DbvPerResSetExtendedSevice;
```

```
/* -------------  "Persistent Query" Extended Service ---------------- */
```

```
typedef enum DbvPerQueryReqOrRsp {
                    PQROR_Request  = 1,
                    PQROR_Response = 2,
                    PQROR_Invalid = -1
} DbvPerQueryReqOrRsp;
```

```
typedef struct DbvPerQueryOriginPartToKeep
{
DbvInternationalString **ppsDbNames;
DbvOtherInformation    **ppsAdditionalSearchInfo;
} DbvPerQueryOriginPartToKeep;




typedef enum DbvPerQueryOriginPartNotToKeepType {
    PQOPNTKT_Package = 1,
    PQOPNTKT_Query   = 2,
    PQOPNTKT_Invalid = -1
} DbvPerQueryOriginPartNotToKeepType;



typedef struct DbvPerQueryOriginPartNotToKeep
{
DbvPerQueryOriginPartNotToKeepType ePerQueryOriginPartNotToKeepType;
union {
     DbvInternationalString *psPackage;
     DbvQuery               *psQuery;
   } uPerQueryOriginPartNotToKeep;
} DbvPerQueryOriginPartNotToKeep;




typedef struct DbvPerQueryRequestPackage
{
DbvPerQueryOriginPartToKeep      *psToKeep;
DbvPerQueryOriginPartNotToKeep    sNotToKeep;
} DbvPerQueryRequestPackage;



typedef struct DbvPerQueryTaskPackage
```

```
{
DbvPerQueryOriginPartToKeep *psOriginPart;
DbvQuery                    *psTargetPart;
} DbvPerQueryTaskPackage;


typedef struct DbvPerQueryExtendedService
{
DbvPerQueryReqOrRsp ePerQueryReqRsp;
union {
    DbvPerQueryRequestPackage    *psPerQueryRequest;
    DbvPerQueryTaskPackage       *psPerQueryResponse;
  } uPerQueryReqOrRsp;
} DbvPerQueryExtendedService;




/* -------------  "Periodic Query Schedule" Extended Service ---------------- */


typedef enum DbvPerQuerySchReqOrRsp {
                    PQSROR_Request  = 1,
                    PQSROR_Response = 2,
                    PQSROR_Invalid = -1
} DbvPerQuerySchReqOrRsp;


typedef enum DbvResultSetDispositionType {
   RSDT_Replace1   = 1,
   RSDT_Append1    = 2,
   RSDT_CreateNew  = -1
} DbvResultSetDispositionType;
```

```
typedef enum DbvDestinationType {
    DT_PhoneNumber    = 1,
    DT_FaxNumber      = 2,
    DT_X400Address    = 3,
    DT_EmailAddress   = 4,
    DT_PagerNumber    = 5,
    DT_FtpAddress     = 6,
    DT_FtAmAddress    = 7,
    DT_PrinterAddress = 8,
    DT_Other          = 9,
    DT_Invalid        = -1
} DbvDestinationType;
```

```
typedef struct DbvOther
{
DbvInternationalString  *psVehicle;
DbvInternationalString  *psDestination;
} DbvOther;
```

```
typedef struct DbvDestination
{
DbvDestinationType eDestinationType;
union {
     DbvInternationalString *psPhoneNumber;
     DbvInternationalString *psFaxNumber;
     DbvInternationalString *psX400Address;
     DbvInternationalString *psEmailAddresss;
     DbvInternationalString *psPagerNumber;
     DbvInternationalString *psFtpAddresss;
     DbvInternationalString *psFtpAmAddress;
     DbvInternationalString *psPrinterAddress;
     DbvOther               *psOther;
```

```
  } uDestinationType;
} DbvDestination;




typedef enum DbvPerQuerySchOriginKeepChoiceType {
    OPTKCT_PackageName   = 1,
    OPTKCT_ExportPackage = 2,
    OPTKCT_Invalid       = -1
} DbvPerQuerySchOriginKeepChoiceType;




typedef enum DbvExpSpecType {
    EST_EsRequest   = 1,
    EST_TaskPackage = 2,
    EST_Invalid     = -1
} DbvExpSpecType;




typedef struct DbvPerQuerySchOriginKeep
{
DbvCompSpec    sComposition;
DbvDestination sExportDestination;
} DbvPerQuerySchOriginKeep;




typedef struct DbvPerQuerySchOriginPartKeepChoice
{
DbvPerQuerySchOriginKeepChoiceType ePerQuerySchOriginKeepChoiceType;
union {
     DbvInternationalString           *psPackageName;
     struct DbvExpSpecExtendedService *psExportPackage;
 } uPerQueryScheduleOriginPartToKeepChoice;
```

```
} DbvPerQuerySchOriginPartKeepChoice;




typedef struct DbvPerQuerySchOriginPartKeep
{
DbvBool                                fActiveFlag;
DbvInternationalString               **ppsDatabaseNames;
DbvResultSetDispositionType            eResultSetDisposition;
DbvDestination                        *psAlertDestination;
DbvPerQuerySchOriginPartKeepChoice    *psExportParameters;
} DbvPerQuerySchOriginPartKeep;




typedef enum DbvPerQuerySchOriginNotKeepChoiceType {
   PQONK_ActualQuery = 1,
   PQONK_PackageName = 2,
   PQONK_Invalid     = -1
} DbvPerQuerySchOriginPartNotKeepChoiceType;



typedef struct DbvPerQuerySchOriginPartNotToKeepChoice
{
DbvPerQuerySchOriginPartNotKeepChoiceType
ePerQuerySchOriginPartNotKeepChoiceType;
union {
     DbvQuery               *psActualQuery;
     DbvInternationalString *psPackageName;
  } uPerQueryScheduleOriginPartNotToKeepChoice;
} DbvPerQuerySchOriginPartNotKeepChoice;




typedef enum DbvPeriodType {
    PT_Unit          = 1,
```

```
    PT_BusinessDaily = 2,
    PT_Continuous    = 3,
    PT_Other         = 4,
    PT_Invalid       = -1
} DbvPeriodType;



typedef struct DbvPeriod
{
DbvPeriodType eDbvPeriodType;
union {
     DbvIntUnit             *psUnit;
     DbvInternationalString *psOther;
  } uDbvPeriod;
} DbvPeriod;



typedef struct DbvPerQuerySchOriginPartNotKeep
{
DbvPerQuerySchOriginPartNotKeepChoice *psQuerySpec;
DbvPeriod                             *psOriginSuggestedPeriod;
DbvGeneralizedTime                     pcExpiration;
DbvInternationalString                *psResultSetPackage;
} DbvPerQuerySchOriginPartNotKeep;




typedef struct DbvPerQuerySchRequestPackage
{
DbvPerQuerySchOriginPartKeep      sToKeep;
DbvPerQuerySchOriginPartNotKeep   sNotToKeep;
} DbvPerQuerySchRequestPackage;




typedef struct DbvPerQuerySchTargetPart
```

```
{
DbvQuery                sActualQuery;
DbvPeriod               sTargetStatedPeriod;
DbvGeneralizedTime      pcExpiration;
DbvInternationalString *psResultSetPackage;
DbvGeneralizedTime      pcLastQueryTime;
Ulong                   ulLastResultNumber;
Ulong                  *pulNumberSinceModify;
} DbvPerQuerySchTargetPart;



typedef struct DbvPerQuerySchTaskPackage
{
DbvPerQuerySchOriginPartKeep   *psOriginPart;
DbvPerQuerySchTargetPart       *psTargetPart;
} DbvPerQuerySchTaskPackage;




typedef struct DbvPerQuerySchExtendedService
{
DbvPerQuerySchReqOrRsp ePerQuerySchReqOrRsp;
union {
    DbvPerQuerySchRequestPackage   *psPerQuerySchRequest;
    DbvPerQuerySchTaskPackage      *psPerQuerySchResponse;
  } uPerQuerySchReqOrRsp;
} DbvPerQuerySchExtendedService;
```

```
/* Item Order definition is in itemordr.h */


/* Update definition is in sitypes.h */
```

```
/* -------------  "Export Specification" Extended Service ---------------- */

typedef enum DbvExpSpecReqOrRsp {
                    ESROR_Request  = 1,
                    ESROR_Response = 2,
                    ESROR_Invalid = -1
} DbvExpSpecReqOrRsp;




typedef struct DbvExpSpecOriginPartToKeep
{
DbvCompSpec     sComposition;
DbvDestination  sExportDestination;
} DbvExpSpecOriginPartToKeep;



typedef struct DbvExpSpecRequestPackage
{
DbvExpSpecOriginPartToKeep sToKeep;
} DbvExpSpecRequestPackage;



typedef struct DbvExpSpecTaskPackage
{
DbvExpSpecOriginPartToKeep sOriginPart;
} DbvExpSpecTaskPackage;




typedef struct DbvExpSpecExtendedService
{
DbvExpSpecReqOrRsp eExpSpecReqOrRsp;
union {
    DbvExpSpecRequestPackage   *psExpSpecRequest;
    DbvExpSpecTaskPackage      *psExpSpecResponse;
```

```
  } uExpSpecReqOrRsp;
} DbvExpSpecExtendedService;



/* -------------  "Export Invocation" Extended Service ---------------- */



typedef enum DbvExpInvoReqOrRsp {
                    EIROR_Request  = 1,
                    EIROR_Response = 2,
                    EIROR_Invalid = -1
} DbvExpInvoReqOrRsp;



typedef enum DbvExpInvoOriginPartToKeepChoiceType {
   EI_PackageName = 1,
   EI_PackageSpec = 2,
   EI_Invalid     = -1
} DbvExpInvoOriginPartToKeepChoiceType;




typedef struct DbvExpInvoOriginPartToKeepChoice
{
DbvExpInvoOriginPartToKeepChoiceType eExpInvoOriginPartToKeepChoiceType;
union {
     DbvInternationalString        *psPackageName;
     DbvExpSpecExtendedService     *psPackageSpec;
   } uExpInvoOriginPartToKeepChoice;
} DbvExpInvoOriginPartToKeepChoice;




typedef struct DbvExpInvoOriginPartToKeep
{
DbvExpInvoOriginPartToKeepChoice   sExportSpec;
```

```
Ulong                             ulNumberOfCopies;
} DbvExpInvoOriginPartToKeep;



typedef enum DbvExpInvoRecordsType {
  RTT_All      = 0,
  RTT_Ranges   = 1,
  RTT_Invalid  = -1
} DbvExpInvoRecordsType;




typedef struct DbvRanges
{
Ulong   ulstart;
Ulong  *pulCount;
} DbvRanges;



typedef struct DbvExpInvoRecords
{
DbvExpInvoRecordsType eDbvExpInvoRecordsType;
union {
     DbvRanges **ppsRanges;
  } uDbvExpInvoRecords;
} DbvExpInvoRecords;



typedef struct DbvExpInvoOriginPartNotToKeep
{
DbvInternationalString       sResultSetId;
DbvExpInvoRecords            sRecords;
} DbvExpInvoOriginPartNotToKeep;



typedef struct DbvExpInvoRequestPackage
```

```
{
DbvExpInvoOriginPartToKeep    sToKeep;
DbvExpInvoOriginPartNotToKeep sNotToKeep;
} DbvExpInvoRequestPackage;


typedef struct DbvExpInvoTargetPart
{
DbvIntUnit  *psEstimatedQuantity;
DbvIntUnit  *psQuantitySoFar;
DbvIntUnit  *psEstimatedCost;
DbvIntUnit  *psCostSoFar;
} DbvExpInvoTargetPart;


typedef struct DbvExpInvoTaskPackage
{
DbvExpInvoOriginPartToKeep sOriginPart;
DbvExpInvoTargetPart       sTargetPart;
} DbvExpInvoTaskPackage;


typedef struct DbvExpInvoExtendedService
{
DbvExpInvoReqOrRsp eExpInvoReqOrRsp;
union {
    DbvExpInvoRequestPackage   *psExpInvoRequest;
    DbvExpInvoTaskPackage      *psExpInvoResponse;
  } uExpInvoReqOrRsp;
} DbvExpInvoExtendedService;
```

```
/* ------------------------------------------------------------*/
/*    Appendix USR User Information Formats                    */
/* ------------------------------------------------------------*/

/* -------------- User Info Format "Search Result 1" ----------------*/



typedef enum DbvResultsByDBSeqChoiceType
{
    RBDSCT_All  = 1,
    RBDSCT_List = 2,
    RBDSCT_Invalid = -1
} DbvResultsByDBSeqChoiceType;




typedef enum DbvQueryExpressionType {
   QET_Term     = 1,
   QET_Query    = 2,
   QET_Invalid  = -1
} DbvQueryExpressionType;



typedef struct DbvQueryExpressionSeq
{
DbvTerm                 sQueryTerm;
DbvInternationalString *psTermComment;
} DbvQueryExpressionSeq;



typedef struct DbvQueryExpression
{
DbvQueryExpressionType eQueryExpressionType;
union {
     DbvQueryExpressionSeq *psTerm;
     DbvQuery              *psQuery;
```

```
   } uQueryExpression;
} DbvQueryExpression;




typedef struct DbvResultsByDBSeqChoice
{
DbvResultsByDBSeqChoiceType eResultsByDBSeqChoiceType;
union {
   DbvInternationalString **ppsList;
  } uResultsByDBSeqChoice;
} DbvResultsByDBSeqChoice;



typedef struct DbvResultsByDB
{
DbvResultsByDBSeqChoice sDatabases;
Ulong                   *pulCount;
DbvInternationalString  *psResultSetName;
} DbvResultsByDB;



typedef struct DbvSearchInfoReportSeq {
DbvInternationalString *psSubqueryId;
DbvBool                 fFullQuery;
DbvQueryExpression     *psSubqueryExpression;
DbvQueryExpression     *psSubqueryInterpretation;
DbvQueryExpression     *psSubqueryRecommendation;
Ulong                  *pulSubqueryCount;
DbvIntUnit             *psSubqueryWeight;
DbvResultsByDB        **ppsresultsByDB;
} DbvSearchInfoReportSeq;



typedef DbvSearchInfoReportSeq **DbvUserInfoSearchResult;
```

```
/* -----------------------------------------------------------*/
/*    Appendix ESP Element Specification Formats              */
/* -----------------------------------------------------------*/

/* -------------- Element Specification Format "eSpec1" ----------------*/


typedef enum DbvElementRequestType
{
    ERTT_SimpleElement    = 1,
    ERTT_CompositeElement = 2,
    ERTT_Invalid          = -1
} DbvElementRequestType;


typedef enum DbvTagPath1Type {
   TPT_SpecificTag = 1,
   TPT_WildThing   = 2,
   TPT_wildPath    = -1
} DbvTagPath1Type;



typedef enum DbvOccurrencesType
{
   OT_All     = 1,
   OT_Last    = 2,
   OT_Values  = 3,
   OT_Invalid = -1
} DbvOccurrencesType;



typedef struct DbvXValues
{
Ulong       ulStart;
```

```
Ulong      *pulHowMany;
} DbvXValues;



typedef struct DbvOccurrences
{
DbvOccurrencesType eOccurrencesType;
union {
     DbvXValues *psValues;
   } uDbvOccurrences;
} DbvOccurrences;



typedef struct DbvTagPathChoiceSeq
{
Ulong               *pulTagType;
DbvStringOrNumeric  sTagValue;
DbvOccurrences     *psOccurrence;
} DbvTagPathChoiceSeq;



typedef struct DbvTagPath1
{
DbvTagPath1Type eTagPath1Type;
union {
    DbvTagPathChoiceSeq *psSpecificTag;
    DbvOccurrences      *psWildThing;
   } uDbvTagPath1;
} DbvTagPath1;



typedef struct DbvSimpleElement
{
DbvTagPath1 **ppsPath;
DbvVariant   *psVariantRequest;
} DbvSimpleElement;
```

```
typedef enum DbvElementListType {
   ELT_Primitives = 1,
   ELT_Specs      = 2,
   ELT_Invalid    = -1
} DbvElementListType;
```

```
typedef struct DbvElementList
{
DbvElementListType eDbvElementListType;
union {
     DbvInternationalString  **ppsPrimitives;
     DbvSimpleElement        **ppsSpecs;
   } uElementList;
} DbvElementList;
```

```
typedef struct DbvElementRequestSeq
{
DbvElementList      sElementList;
DbvTagPath1       **ppsDeliveryTag;
DbvVariant         *psVariantRequest;
} DbvElementRequestSeq;
```

```
typedef struct DbvElementRequest
{
DbvElementRequestType eElementRequestType;
union {
     DbvSimpleElement      *psSimpleElement;
     DbvElementRequestSeq  *psCompositeElement;
  } uElementRequest;
```

```
} DbvElementRequest;



typedef struct DbvElementSpec1
{
DbvInternationalString **ppsElementSetNames;
DbvObjectIdentifier      pcDefaultVariantSetId;
DbvVariant              *psDefaultVariantRequest;
Ulong                   *pulDefaultTagType;
DbvElementRequest      **ppsElements;
} DbvElementSpec1;



#endif
```