INTERNATIONAL STANDARD

ISO/IEC 13719-4

First edition
1998-10-01

Information technology — Portable Common Tool Environment (PCTE) —

Part 4:
IDL binding (Interface Definition Language)

Technologies de l'information — Environnement d'outil courant portable (PCTE) —

Partie 4: Liaison IDL (langage de définition d'interface)

Reference number
ISO/IEC 13719-4:1998(E)

Contents

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.

International Standard ISO/IEC 13719-4 was prepared by ECMA (as Standard ECMA-230) and was adopted, under a special "fast-track procedure", by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

ISO/IEC 13719 consists of the following parts, under the general title *Information technology - Portable Common Tool Environment (PCTE):*

– *Part 1: Abstract specification*

– *Part 2: C programming language binding*

– *Part 3: Ada programming language binding*

– *Part 4: IDL binding (Interface Definition Language)*

Annex C forms an integral part of this part of ISO/IEC 13719. Annexes A and B are for information only.

Information technology - Portable Common Tool Environment (PCTE) -

Part 4:
IDL binding (Interface Definition Language)

1 Scope

This part of ISO/IEC 13719 defines the standard binding of the Portable Common Tool Environment (PCTE), as specified in ISO/IEC 13719-1, to the CORBA Interface Definition Language (IDL) defined in ISO/IEC CD 14750.

A number of features are not completely defined in ISO/IEC 13719-1, some freedom being allowed to the implementer. Some of these features are specified as implementation limits. Some constraints are placed on these implementation limits by this IDL Binding Standard. These constraints are specified in clause 24, Implementation Limits.

PCTE is an interface to a set of facilities that forms the basis for constructing environments supporting systems engineering projects. These facilities are designed particularly to provide an infrastructure for programs which may be part of such environments. Such programs, which are used as aids to systems development, are often referred to as tools.

2 Conformance

An implementation of PCTE conforms to this part of ISO/IEC 13719 if it conforms to 2.2 of ISO/IEC 13719-1, where the binding referred is taken to be the IDL Binding defined in clauses 1 to 5 and 8 to 25 of this part of ISO/IEC 13719. All other clauses in this part of ISO/IEC 13719 are provided as assistance to the reader and are not normative.

The IDL Binding defined in this part of ISO/IEC 13719 conforms to 2.1 of ISO/IEC 13719-1.

3 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 13719. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 13719 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 13719-1:1998, *Information technology - Portable Common Tool Environment (PCTE) - Part 1: Abstract specification.*

ISO/IEC 13719-2:1998, *Information technology - Portable Common Tool Environment (PCTE) - Part 2: C programming language binding.*

ISO/IEC 14750:—[1)], *Information technology - Open Distributed Processing - Interface Definition Language.*

1) To be published.

4 Definitions

All technical terms used in this part of ISO/IEC 13719, other than a few in widespread use, are defined in the body of this part of ISO/IEC 13719 or in the referenced documents.

5 Formal notations

For the IDL binding for each operation, the function syntax is used as defined in ISO/IEC CD 14750.

6 Outline of the Standard

Clause 7 describes the strategy used to develop this binding specification

Clause 8 contains the mapping from the datatypes that are used in the Abstract Specification to the IDL datatypes.

Clause 9 to 22 define the binding of datatypes and operations in the corresponding clauses of ISO/IEC 13719-1. The extensions for fine-grain objects are added at the end of clause 11.

Clause 23 defines the binding of object, attribute, link, and type references, as specified in 23.1.2 and 23.2 of ISO/IEC 13719-1.

Clause 24 defines the binding of the implementation limit functions described in clause 24 of ISO/IEC 13719-1.

Clause 25 defines the binding of the error conditions described in annex C of ISO/IEC 13719-1, and defines binding-defined error conditions for the IDL Binding.

There are 2 informative annexes. Annex A compares the structures of this IDL binding and of the C binding of ISO/IEC 13719-2, explaining the differences. Annex B describes the source file structure of the IDL binding.

Annex C, which is normative, contains the extensions for object orientation, corresponding to annex G of ISO/IEC 13719-1.

7 Binding strategy

7.1 IDL standard

This part of ISO/IEC 13719 conforms to the definition of IDL in ISO/IEC 14750.

7.2 General principles

The following general principles were applied when generating the binding in this part of ISO/IEC 13719.

The C interface generated from the IDL binding should be as close as possible to the PCTE C language binding of ISO/IEC 13719-2, so as to minimize changes to existing C applications.

The binding should leave open the possibility for an implementation of the binding to allow a non-PCTE process to access the PCTE object base without being statically linked to the PCTE

interface. This implies that the implementation of the static bindings generated from the IDL must not make use of any PCTE operations. The IDL binding has been structured, through the use of Pseudo-IDL (PIDL), to leave this implementation option open.

The majority of the operations accept a Pcte_object_reference as controlling object. Therefore, ideally, there should exist a Pcte_object_reference interface which inherits from almost all other interfaces. This approach would allow for a static type checking but it is awkward. It has been decided instead to allow casting and let the PCTE implementation raise an exception if the passed controlling object is not of the right type.

Many operations said to be applied to a process object are only applicable to the current process object. This is specified whenever it is necessary. This is also meant sometimes by the comment: /* Operation is applied to self */.

Sequences should be implemented as pseudo-objects to be mapped internally into CORBA sequences. This implies that each operation accepting or returning a sequence must map it in the correct format for the PCTE implementation server. In the case of a sequence of object or link references, each reference must be mapped to a CORBA interface and returned to the client as such. The major reason for this is that in general an object reference may not be easily mapped by an implementation into a format meaningful for network transport. It is easier to assume that the object references are kept on the implementation side, and that at the client side CORBA brings an object handle. This mapping allows the use of dynamic bindings as well as static bindings.

The possibility should be left open of a special implementation choice to implement the PCTE CORBA static bindings stubs to make direct use of the current PCTE C interface: this could be more efficient, but does not allow a distributed implementation of the IDL interface and might preclude the use of dynamic bindings.

7.3 Sets and sequences

All sequence operations are grouped under the Pcte_sequence interface. A difficulty is that the operation *create* is not part of the interface of an object. To keep the resulting generated C code in line with ISO/IEC 13719-2, it is still part of the Pcte_sequence interface, but the controlling object is a constant.

The input and/or result of a sequence create, insert, or get has been mapped to the IDL type **any**.

7.4 References and names

A departure from ISO/IEC 13719-1 is the introduction of an extra interface called PCTE_RF (Reference Factory), which contains those operations that return a reference but do not use a reference as a controlling object.

The rest of the mapping is straightforward, with three interfaces Pcte_object_reference, Pcte_link_reference, and Pcte_type_reference.

7.5 Implementation aspects

7.5.1 Source files

The source file structure is described in annex B. To simplify the IDL compilation process a few new IDL source files are introduced; this is because the ISO/IEC 13719-2 header structure includes both types and operations, where in many cases the latter are not needed. With IDL this leads to many forward references, eliminated by the introduction of oms_types.idl, discretionary_types.idl and mandatory_types.idl.

7.5.2 Naming changes in the IDL

All parameters with name containing 'attribute' have been renamed with 'attribute' replaced by 'attribute_ref'.

All parameters with name containing 'object' have been removed (i.e. as controlling object) or renamed with 'object' replaced by 'object_ref'.

The enumeration values PCTE_KEY, PCTE_NON_KEY to PCTE_KEY_ATTR, PCTE_NON_KEY_ATTR have been renamed to avoid clashes of the first item with Pcte_key, as IDL does not allow two identifiers which differ only by case to be used in the same scope.

The sequence enumeration items have been renamed to avoid clashes with the typedef of the sequences.

7.5.3 Difference in generated C code

All unions have extra '_d' and '_u' fields and are introduced by means of typedef. A result is that the resulting C code must be changed to use these extra fields.

The enumeration items cannot have a user-defined value. The generated header files must be changed manually.

8 Datatype mapping

8.1 Basic datatypes

The datatype mapping for basic types follows ISO/IEC 13719-2 closely.

- string is mapped to the IDL type **string**;
- natural and integer are mapped to the IDL type **long**;
- boolean is mapped to the IDL type **boolean**;
- float is mapped to the IDL type **float**;
- Pcte_pathname, Pcte_object_reference, etc. as identifier and interface name have been changed to be interfaces or pseudo-objects;

As IDL does not allow two identifiers which differ only by case to be used in the same scope, an "_EI" suffix has been added to the enumeration items of the Pcte_sequence_type (which otherwise would have been conflicting with sequence names).

8.2 Sequences

```
/* The source file "sequences.idl" */
#ifndef PCTE_SEQUENCES_INCLUDED
#define PCTE_SEQUENCES_INCLUDED 1
#include "types.idl"
enum Pcte_sequence_type {
    PCTE_ACCOUNTING_FILE_EI, PCTE_ACL_EI, PCTE_AUDIT_FILE_EI,
    PCTE_ATTRIBUTE_ASSIGNMENTS_EI, PCTE_H_ATTRIBUTE_ASSIGNMENTS_EI,
    PCTE_ATTRIBUTE_NAMES_EI, PCTE_ATTRIBUTE_REFERENCES_EI,
    PCTE_BUFFER_EI, PCTE_CONFIDENTIALITY_CRITERIA_EI,
    PCTE_ENUMERATION_VALUE_TYPE_EI,
    PCTE_H_ENUMERATION_VALUE_TYPE_EI,
    PCTE_ENUMERATION_VALUE_TYPE_IN_SDS_EI, PCTE_GENERAL_CRITERIA_EI,
    PCTE_INTEGRITY_CRITERIA_EI, PCTE_KEY_TYPES_EI, PCTE_H_KEY_TYPES_EI,
    PCTE_KEY_TYPES_IN_SDS_EI, PCTE_LINK_NAMES_EI,
    PCTE_LINK_SET_DESCRIPTORS_EI, PCTE_H_LINK_SET_DESCRIPTORS_EI,
    PCTE_LINK_REFERENCES_EI, PCTE_MESSAGE_TYPES_EI,
    PCTE_NAME_SEQUENCE_EI, PCTE_OBJECT_CRITERIA_EI,
    PCTE_OBJECT_REFERENCES_EI, PCTE_TYPE_NAMES_EI,
    PCTE_TYPE_NAMES_IN_SDS_EI, PCTE_TYPE_REFERENCES_EI,
    PCTE_USER_CRITERIA_EI, PCTE_VOLUME_INFOS_EI,

    /* New Object-Oriented extension sequences */

    PCTE_PARAMETER_ITEMS_EI,
    PCTE_METHOD_REQUESTS_EI,
    PCTE_CONTEXT_ADOPTIONS_EI,
    PCTE_METHOD_REQUEST_IDS_EI
    };
typedef Object Pcte_sequence_element;
typedef Object Pcte_array_of_sequence_elements;
interface Pcte_sequence;
#define Pcte_null_sequence (Pcte_sequence) NULL
typedef Pcte_sequence Pcte_accounting_file;
typedef Pcte_sequence Pcte_audit_file;
typedef Pcte_sequence Pcte_attribute_names;
typedef Pcte_sequence Pcte_attribute_references;
typedef Pcte_sequence Pcte_buffer;
typedef Pcte_sequence Pcte_confidentiality_criteria;
typedef Pcte_sequence Pcte_enumeration_value_type;
typedef Pcte_sequence Pcte_h_enumeration_value_type;
typedef Pcte_sequence Pcte_enumeration_value_type_in_sds;
```

```
typedef Pcte_sequence Pcte_general_criteria;
typedef Pcte_sequence Pcte_integrity_criteria;
typedef Pcte_sequence Pcte_key_types;
typedef Pcte_sequence Pcte_h_key_types;
typedef Pcte_sequence Pcte_key_types_in_sds;
typedef Pcte_sequence Pcte_link_set_descriptors;
typedef Pcte_sequence Pcte_h_link_set_descriptors;
typedef Pcte_sequence Pcte_link_names;
typedef Pcte_sequence Pcte_link_references;
typedef Pcte_sequence Pcte_message_types;
typedef Pcte_sequence Pcte_name_sequence;
typedef Pcte_sequence Pcte_object_criteria;
typedef Pcte_sequence Pcte_object_references;
typedef Pcte_sequence Pcte_type_names;
typedef Pcte_sequence Pcte_type_names_in_sds;
typedef Pcte_sequence Pcte_type_references;
typedef Pcte_sequence Pcte_user_criteria;
typedef Pcte_sequence Pcte_volume_infos;
typedef Pcte_sequence Pcte_parameters_items;
typedef Pcte_sequence Pcte_method_requests;
typedef Pcte_sequence Pcte_method_requests;
typedef Pcte_sequence Pcte_method_request_ids;
interface Pcte_sequence {                                          //PIDL
/* Mapped to a CORBA sequence. */
/* This interface is conventionally applied to the PCTE object type "process". */
Pcte_error_type create (
    in Pcte_sequence_type                type,
    in Pcte_array_of_sequence_elements   data,
    in Pcte_natural                      count,
    out Pcte_sequence                    out_sequence
);
Pcte_error_type discard (
);
```

```
Pcte_error_type copy (
    out Pcte_sequence    destination_list,
    in Pcte_natural      index,
    in Pcte_natural      source_index,
    in Pcte_natural      count
);

Pcte_error_type insert_elements (
    in Pcte_natural                          index,
    in Pcte_array_of_sequence_elements       data,
    in Pcte_natural                          count
);

Pcte_error_type delete (
    in Pcte_natural    index,
    in Pcte_natural    count
);

Pcte_error_type are_equal (
    in Pcte_sequence second_sequence,
    out Pcte_boolean     equality
);

Pcte_error_type get_index (
    in Pcte_sequence_element   element,
    out Pcte_integer           index
);

Pcte_error_type get_length (
    out Pcte_natural  length
);

Pcte_error_type get_elements (
    in Pcte_natural                          index,
    out Pcte_array_of_sequence_elements data,
    in Pcte_natural                          count
);

Pcte_error_type get (
    in Pcte_natural                index,
    out Pcte_sequence_element      element
);

Pcte_error_type insert (
    in Pcte_natural            index,
    in Pcte_sequence_element   element
);

Pcte_error_type replace (
    in Pcte_natural            index,
    in Pcte_sequence_element   element
);
```

```
Pcte_error_type append (
    in Pcte_sequence_element   element
);

Pcte_error_type normalize (
);

};

#endif
```

8.3 The global pcte source file

```
/* The source file "pcte.idl" */

#ifndef PCTE_INCLUDED
#define PCTE_INCLUDED 1

#include   "types.idl"            // 8.4
#include   "sequences.idl"        // 8.2
#include   "references.idl"       // clause 23
#include   "limits.idl"           // clause 24
#include   "errors.idl"           // clause 25

#include   "oms.idl"              // clause 9
#include   "sms.idl"              // clause 10
#include   "devices.idl"          // clause 11
#include   "contents.idl"         // clause 12
#include   "execution.idl"        // clause 13
#include   "messages.idl"         // clause 14
#include   "notification.idl"     // clause 15
#include   "activities.idl"       // clause 16
#include   "replication.idl"      // clause 17
#include   "network.idl"          // clause 18
#include   "discretionary.idl"    // clause 19
#include   "mandatory.idl"        // clause 20
#include   "auditing.idl"         // clause 21
#include   "accounting.idl"       // clause 22

/* #include directive used for cluster management */

#include   "clusters.idl"

/* #include directives used by Pcte object-oriented extensions */

#include   "interfaces.idl"
#include   "methods.idl"

 #endif      // ! PCTE_INCLUDED
```

8.4 The PCTE basic type source file

```
/* The source file "types.idl" */

#ifndef PCTE_TYPES_INCLUDED
#define PCTE_TYPES_INCLUDED 1

typedef unsigned long time_t;
#include "errors.idl"

#define PCTE_OK 0
#define PCTE_ERROR 1

typedef unsigned short Pcte_boolean;

#define PCTE_TRUE (Pcte_boolean) 1
#define PCTE_FALSE (Pcte_boolean) 0

typedef long            Pcte_integer;

typedef unsigned long   Pcte_natural;

typedef float           Pcte_float;

typedef time_t          Pcte_time;

#define Pcte_time_accuracy_factor (Pcte_natural) <implementation-defined>

#define Pcte_reference_time (Pcte_time) <implementation-defined>

#define Pcte_null_time (Pcte_time) <implementation-defined>

typedef octet Pcte_octet;

struct Pcte_string {
Pcte_natural   size;
Pcte_octet array;
};

#endif      // !PCTE_TYPES_INCLUDED
```

9 Object management

9.1 Object management datatypes

```
/* The source file "oms_types.idl" */

#define PCTE_OMS_TYPES_INCLUDED 1

enum Pcte_category {
    PCTE_COMPOSITION,
    PCTE_EXISTENCE,
    PCTE_REFERENCE,
    PCTE_DESIGNATION,
    PCTE_IMPLICIT
};

typedef Pcte_natural Pcte_categories ;
```

```
#define PCTE_ALL_CATEGORIES (Pcte_natural) PCTE_COMPOSITION |\
    PCTE_EXISTENCE |\
    PCTE_REFERENCE |\
    PCTE_DESIGNATION |\
    PCTE_IMPLICIT

enum Pcte_value_type {
    PCTE_BOOLEAN_ATTRIBUTE,
    PCTE_INTEGER_ATTRIBUTE,
    PCTE_NATURAL_ATTRIBUTE,
    PCTE_FLOAT_ATTRIBUTE,
    PCTE_STRING_ATTRIBUTE,
    PCTE_TIME_ATTRIBUTE,
    PCTE_ENUMERATION_ATTRIBUTE
};

union Pcte_value_value switch (long) {
    case 1 : Pcte_boolean    v_boolean;
    case 2 : Pcte_integer    v_integer;
    case 3 : Pcte_natural    v_natural;
    case 4 : Pcte_float      v_float;
    case 5 : Pcte_string     v_string;
    case 6 : Pcte_time       v_time;
    case 7 : Pcte_natural    v_enumeral_type_position;
};

struct Pcte_attribute_value {
    Pcte_value_type      type;
    Pcte_value_value     value;
};

struct Pcte_attribute_assignment {
    Pcte_attribute_name      name;
    Pcte_attribute_value     value;
};

struct Pcte_h_attribute_assignment {
    Pcte_attribute_reference     reference;
    Pcte_attribute_value         value;
};

enum Pcte_link_scope {
    PCTE_INTERNAL_LINKS, PCTE_EXTERNAL_LINKS, PCTE_ALL_LINKS
};

enum Pcte_type_ancestry {
    PCTE_EQUAL_TYPE, PCTE_ANCESTOR_TYPE,
    PCTE_DESCENDANT_TYPE, PCTE_UNRELATED_TYPE
};
```

```
enum Pcte_version_relation {
    PCTE_ANCESTOR_VSN, PCTE_DESCENDANT_VSN, PCTE_SAME_VSN,
    PCTE_RELATED_VSN, PCTE_UNRELATED_VSN
};

enum Pcte_object_scope {
    PCTE_ATOMIC, PCTE_COMPOSITE
};

#define PCTE_MAX_EXACT_IDENTIFIER_SIZE PCTE_MAX_KEY_SIZE

typedef Pcte_octet
    Pcte_exact_identifier [PCTE_MAX_EXACT_IDENTIFIER_SIZE + 1];

#endif      // !PCTE_TYPES_INCLUDED

/* The source file "oms.idl" */

#ifndef PCTE_OMS_INCLUDED
#define PCTE_OMS_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "oms_types.idl"
#include "sequences.idl"
#include "contents_types.idl"

typedef Object Pcte_contents;

typedef Pcte_sequence Pcte_attribute_assignments;

typedef Pcte_sequence Pcte_h_attribute_assignments;

enum Pcte_volume_accessibility {
    PCTE_ACCESSIBLE, PCTE_INACCESSIBLE, PCTE_UNKNOWN
};

#include "devices.idl"

struct Pcte_volume_info {
    Pcte_volume_identifier      volume;
    Pcte_volume_accessibility   mounted;
};
struct Pcte_link_set_descriptor {
    Pcte_object_reference    origin;
    Pcte_link_names          links;
};

struct Pcte_h_link_set_descriptor {
    Pcte_object_reference    origin;
    Pcte_link_references     links;
};

#include "discretionary.idl"
```

9.2 Link operators

```
interface Pcte_link {

/* This interface is applied to the PCTE object type "object" */

/* 9.2.1 LINK_CREATE */

Pcte_error_type create (
    in Pcte_link_name           new_link,
    in Pcte_object_reference    dest,
    in Pcte_key                 reverse_key
);

/* 9.2.2 LINK_DELETE */

Pcte_error_type delete (
    in Pcte_link_name    link
);

/* 9.2.3 LINK_DELETE_ATTRIBUTE */

Pcte_error_type delete_attribute (
    in Pcte_link_name           link,
    in Pcte_attribute_reference attribute_ref
);

/* 9.2.4 LINK_GET_ATTRIBUTE */

Pcte_error_type get_attribute (
    in Pcte_link_name           link,
    in Pcte_attribute_name      name,
    out Pcte_attribute_value    value
);

/* 9.2.5 LINK_GET_DESTINATION_VOLUME */

Pcte_error_type get_destination_volume (
    in Pcte_link_name       link,
    out Pcte_volume_info    volume_info
);

/* 9.2.6 LINK_GET_KEY */

Pcte_error_type get_key (
    in Pcte_link_name    link,
    out Pcte_key         key
);
```

```
/* 9.2.7 LINK_GET_REVERSE */
Pcte_error_type get_reverse (
    in Pcte_link_name           link,
    out Pcte_link_name          reverse_link,
    out Pcte_object_reference   dest
);
/* 9.2.8 LINK_GET_SEVERAL_ATTRIBUTES */
Pcte_error_type get_attributes_in_working_schema (
    in Pcte_link_name                   link,
    out Pcte_attribute_assignments      values
);
Pcte_error_type get_attributes_of_types (
    in Pcte_link_name                   link,
    in Pcte_attribute_names             attributes,
    out Pcte_attribute_assignments      values
);
/* 9.2.9 LINK_REPLACE */
Pcte_error_type replace (
    in Pcte_link_name           link,
    in Pcte_object_reference    new_origin,
    in Pcte_link_name           new_link,
    in Pcte_key                 new_reverse_key
);
/* 9.2.10 LINK_RESET_ATTRIBUTE */
Pcte_error_type reset_attribute (
    in Pcte_link_name           link,
    in Pcte_attribute_reference attribute_ref
);
/* 9.2.11 LINK_SET_ATTRIBUTE */
Pcte_error_type set_attribute (
    in Pcte_link_name       link,
    in Pcte_attribute_name  attribute_ref,
    in Pcte_attribute_value value
);
/* 9.2.12 LINK_SET_SEVERAL_ATTRIBUTES */
Pcte_error_type set_several_attributes (
    in Pcte_link_name               link,
    in Pcte_attribute_assignments   attributes
);
```

```
/* 11.2.7 LINK_GET_DESTINATION_ARCHIVE */
Pcte_error_type get_destination_archive (
    in Pcte_link_name            link,
    out Pcte_archive_identifier  archive_identifier
);
};
interface Pcte_h_link {
/* This interface is applied to the PCTE object type "object" \f B*/
/* 9.2.1 LINK_CREATE */
Pcte_error_type create (
    in Pcte_link_reference       new_link,
    in Pcte_object_reference     dest,
    in Pcte_key                  reverse_key
);
/* 9.2.2 LINK_DELETE */
Pcte_error_type delete (
    in Pcte_link_reference   link
);
/* 9.2.3 LINK_DELETE_ATTRIBUTE */
Pcte_error_type delete_attribute (
    in Pcte_link_reference       link,
    in Pcte_attribute_reference  attribute_ref
);
/* 9.2.4 LINK_GET_ATTRIBUTE */
Pcte_error_type get_attribute (
    in Pcte_link_reference       link,
    in Pcte_attribute_reference  attribute_ref,
    out Pcte_attribute_value     value
);
/* 9.2.5 LINK_GET_DESTINATION_VOLUME */
Pcte_error_type get_destination_volume (
    in Pcte_link_reference   link,
    out Pcte_volume_inf      volume_info
);
/* 9.2.6 LINK_GET_KEY */
Pcte_error_type get_key (
    in Pcte_link_reference   link,
    out Pcte_key             key
);
```

```
/* 9.2.7 LINK_GET_REVERSE */
Pcte_error_type get_reverse (
    in Pcte_link_reference      link,
    out Pcte_link_reference     reverse_link,
    out Pcte_object_reference   dest
);
/* 9.2.8 LINK_GET_SEVERAL_ATTRIBUTES */
Pcte_error_type get_attributes_in_working_schema (
    in Pcte_link_reference              link,
    out Pcte_h_attribute_assignments    values
);
Pcte_error_type get_attributes_of_types (
    in Pcte_link_reference              link,
    in Pcte_attribute_references        attributes,
    out Pcte_h_attribute_assignments    values
);
/* 9.2.9 LINK_REPLACE */
Pcte_error_type replace (
    in Pcte_link_reference      link,
    in Pcte_object_reference    new_origin,
    in Pcte_link_reference      new_link,
    in Pcte_key                 new_reverse_key
);
/* 9.2.10 LINK_RESET_ATTRIBUTE */
Pcte_error_type reset_attribute (
    in Pcte_link_reference      link,
    in Pcte_attribute_reference attribute_ref
);
/* 9.2.11 LINK_SET_ATTRIBUTE */
Pcte_error_type set_attribute (
    in Pcte_link_reference      link,
    in Pcte_attribute_reference attribute_ref,
    out Pcte_attribute_value    value
);
/* 9.2.12 LINK_SET_SEVERAL_ATTRIBUTES */
Pcte_error_type set_several_attributes (
    in Pcte_link_reference              link,
    in Pcte_h_attribute_assignments     attributes
);
```

```
/* 11.2.7 LINK_GET_DESTINATION_ARCHIVE */
Pcte_error_type get_destination_archive (
    in Pcte_link_reference          link,
    out Pcte_archive_identifier     archive_identifier
);
};
```

9.3 Object operations

```
interface Pcte_object {
/* This interface is applied to the PCTE object type "object" */
/* 9.3.1 OBJECT_CHECK_TYPE */
Pcte_error_type check_type (
    in Pcte_type_name       type2,
    in Pcte_type_ancestry   relation
);
/* 9.3.2 OBJECT_CONVERT */
Pcte_error_type convert (
    in Pcte_type_name   type
);
/* 9.3.3 OBJECT_COPY */
Pcte_error_type copy (
    in Pcte_link_name               new_link,
    in Pcte_key                     reverse_key,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_object
);
/* 9.3.4 OBJECT_CREATE */
Pcte_error_type create (
    in Pcte_type_name               type,
    in Pcte_link_name               new_link,
    in Pcte_key                     reverse_key,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_object
);
/* 9.3.5 OBJECT_DELETE */
Pcte_error_type delete (
    in Pcte_link_name   link
);
```

```
/* 9.3.6 OBJECT_DELETE_ATTRIBUTE */
Pcte_error_type delete_attribute (
    in Pcte_attribute_name   attribute_ref
);
/* 9.3.7 OBJECT_GET_ATTRIBUTE */
Pcte_error_type get_attribute (
    in Pcte_attribute_name       attribute_ref,
    out Pcte_attribute_value     value
);
/* 9.3.8 OBJECT_GET_PREFERENCE */
Pcte_error_type get_preference (
    out Pcte_key             key,
    out Pcte_type_name       type
);
/* 9.3.9 OBJECT_GET_SEVERAL_ATTRIBUTES */
Pcte_error_type get_attributes_in_working_schema (
    out Pcte_attribute_assignments      values
);
Pcte_error_type get_attributes_of_types (
    in Pcte_attribute_names             attributes,
    out Pcte_attribute_assignments      values
);
/* 9.3.10 OBJECT_GET_TYPE */
Pcte_error_type get_type (
    out Pcte_type_name       type
);
/* 9.3.11 OBJECT_IS_COMPONENT */
Pcte_error_type is_component (
    in Pcte_object_reference component,
    out Pcte_boolean         value
);
/* 9.3.12 OBJECT_LIST_LINKS */
Pcte_error_type list_all_links (
    in Pcte_link_scope              extent,
    in Pcte_object_scope            scope,
    in Pcte_categories              categories,
    out Pcte_link_set_descriptors   links
);
```

```
Pcte_error_type list_links_in_working_schema (
    in Pcte_link_scope              extent,
    in Pcte_object_scope            scope,
    in Pcte_categories              categories,
    out Pcte_link_set_descriptors   links
);
Pcte_error_type list_links_of_types (
    in Pcte_link_scope              extent,
    in Pcte_object_scope            scope,
    in Pcte_type_names              types,
    out Pcte_link_set_descriptors   links
);
/* 9.3.13 OBJECT_LIST_VOLUMES */
Pcte_error_type list_volumes (
    out Pcte_volume_infos   volumes
);
/* 9.3.14 OBJECT_MOVE */
Pcte_error_type move (
    in Pcte_object_reference    on_same_volume_as,
    in Pcte_object_scope        scope
);
/* 9.3.15 OBJECT_RESET_ATTRIBUTE */
Pcte_error_type reset_attribute (
    in Pcte_attribute_name   attribute_ref
);
/* 9.3.16 OBJECT_SET_ATTRIBUTE */
Pcte_error_type set_attribute (
    in Pcte_attribute_name   attribute_ref,
    in Pcte_attribute_value  value
);
/* 9.3.17 OBJECT_SET_PREFERENCE */
Pcte_error_type set_preference (
    in Pcte_type_name   type,
    in Pcte_key         key
);
/* 9.3.18 OBJECT_SET_SEVERAL_ATTRIBUTES */
Pcte_error_type set_several_attributes (
    in Pcte_attribute_assignments   attributes
);
```

```
/* 9.3.19 OBJECT_SET_TIME_ATTRIBUTES */
Pcte_error_type set_time_attributes (
    in Pcte_time            last_access,
    in Pcte_time            last_modification,
    in Pcte_object_scope    scope
);
/* 9.3.20 VOLUME_LIST_OBJECTS */
/* See 11.2. */
/* 20.2.5 OBJECT_SET_CONFIDENTIALITY_LABEL */
Pcte_error_type set_confidentiality_label (
    in Pcte_security_label  label
);
/* 20.2.6 OBJECT_SET_INTEGRITY_LABEL */
Pcte_error_type set_integrity_label (
    in Pcte_security_label  label
);
/* 19.2.2 OBJECT_CHECK_PERMISSION */
Pcte_error_type check_permission (
    in Pcte_discretionary_access_modes  modes,
    in Pcte_object_scope                scope,
    out Pcte_boolean                    accessible
);
/* 19.2.3 OBJECT_GET_ACL */
Pcte_error_type get_acl (
    in Pcte_object_scope    scope,
    out Pcte_acl            acl
);
/* 19.2.4 OBJECT_SET_ACL_ENTRY */
Pcte_error_type set_acl_entry (
    in Pcte_group_identifier        group,
    in Pcte_requested_access_rights modes,
    in Pcte_object_scope            scope
);
/* 11.2.1 ARCHIVE_CREATE */
Pcte_error_type archive_create (
    in Pcte_natural                 archive_identifier,
    in Pcte_object_reference        on_same_volume_as,
    out Pcte_atomic_access_rights   access_mask,
    out Pcte_object_reference       new_archive
);
```

```
/* 12.2.6 CONTENTS_OPEN */
Pcte_error_type contents_open (
    in Pcte_contents_access_mode    opening_mode,
    in Pcte_boolean                 non_blocking_io,
    in Pcte_boolean                 inheritable,
    out Pcte_contents               contents
);
};
```

9.4 Version operations

```
interface Pcte_version {
/* This interface is applied to the PCTE object type "object". */
/* 9.4.1 VERSION_ADD_PREDECESSOR */
Pcte_error_type add_predecessor (
    in Pcte_object_reference    new_predecessor
);
/* 9.4.2 VERSION_IS_CHANGED */
Pcte_error_type is_changed (
    in Pcte_key             predecessor,
    out Pcte_boolean        changed
);
/* 9.4.3 VERSION_REMOVE */
Pcte_error_type remove (
);
/* 9.4.4 VERSION_REMOVE_PREDECESSOR */
Pcte_error_type remove_predecessor (
    in Pcte_object_reference    predecessor
);
/* 9.4.5 VERSION_REVISE */
Pcte_error_type revise (
    in Pcte_object_reference        new_origin,
    in Pcte_link_name               new_link,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_version
);
```

```
/* 9.4.6 VERSION_SNAPSHOT */

Pcte_error_type snapshot (
    in Pcte_object_reference        new_origin,
    in Pcte_link_name               new_link,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_version
);

/* 9.4.7 VERSION_TEST_ANCESTRY */

Pcte_error_type test_ancestry (
    in Pcte_object_reference    version2,
    out Pcte_version_relation   ancestry
);

/* 9.4.8 VERSION_TEST_DESCENT */

Pcte_error_type test_descent (
    in Pcte_object_reference    version2,
    out Pcte_version_relation   descent
);

};
```

9.5 Object and version operations – reference interfaces

```
interface Pcte_h_object {

/* This interface is applied to the PCTE object type "object". */

/* 9.3.1 OBJECT_CHECK_TYPE */

Pcte_error_type check_type (
    in Pcte_type_reference  type2,
    in Pcte_type_ancestry   relation
);

/* 9.3.2 OBJECT_CONVERT */

Pcte_error_type convert (
    in Pcte_type_reference  type
);
```

```
/* 9.3.3 OBJECT_COPY */

Pcte_error_type copy (
    in Pcte_object_reference        new_origin,
    in Pcte_link_reference          new_link,
    in Pcte_key                     reverse_key,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_object
);

/* 9.3.4 OBJECT_CREATE */

Pcte_error_type create (
    in Pcte_type_reference          type,
    in Pcte_link_reference          new_link,
    in Pcte_key                     reverse_key,
    in Pcte_object_reference        on_same_volume_as,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_object
);

/* 9.3.5 OBJECT_DELETE */

Pcte_error_type delete (
    in Pcte_link_reference    link
);

/* 9.3.6 OBJECT_DELETE_ATTRIBUTE */

Pcte_error_type delete_attribute (
    in Pcte_attribute_reference attribute_ref
);

/* 9.3.7 OBJECT_GET_ATTRIBUTE */

Pcte_error_type get_attribute (
    in Pcte_attribute_reference     attribute_ref,
    out Pcte_attribute_value        value
);

/* 9.3.8 OBJECT_GET_PREFERENCE */

Pcte_error_type get_preference (
    out Pcte_key                key,
    out Pcte_link_reference     type
);

/* 9.3.9 OBJECT_GET_SEVERAL_ATTRIBUTES */

Pcte_error_type get_attributes_in_working_schema (
    out Pcte_h_attribute_assignments   values
);
```

```
Pcte_error_type get_attributes_of_types (
    in Pcte_attribute_references          attributes,
    out Pcte_h_attribute_assignments      values
);

/* 9.3.10 OBJECT_GET_TYPE */

Pcte_error_type get_type (
    out Pcte_type_reference      type
);

/* 9.3.12 OBJECT_LIST_LINKS */

Pcte_error_type list_all_links (
    in Pcte_link_scope                    extent,
    in Pcte_object_scope                  scope,
    in Pcte_categories                    categories,
    out Pcte_h_link_set_descriptors       links
);

Pcte_error_type list_links_in_working_schema (
    in Pcte_link_scope                    extent,
    in Pcte_object_scope                  scope,
    in Pcte_categories                    categories,
    out Pcte_h_link_set_descriptors       links
);

Pcte_error_type list_links_of_types (
    in Pcte_link_scope                    extent,
    in Pcte_object_scope                  scope,
    in Pcte_type_references               types,
    out Pcte_h_link_set_descriptors       links
);

/* 9.3.15 OBJECT_RESET_ATTRIBUTE */

Pcte_error_type reset_attribute (
    in Pcte_attribute_reference  attribute_ref
);

/* 9.3.16 OBJECT_SET_ATTRIBUTE */

Pcte_error_type set_attribute (
    in Pcte_attribute_reference  attribute_ref,
    in Pcte_attribute_value      value
);

/* 9.3.17 OBJECT_SET_PREFERENCE */

Pcte_error_type set_preference (
    in Pcte_type_reference   type,
    in Pcte_key              key
);
```

```
/* 9.3.18 OBJECT_SET_SEVERAL_ATTRIBUTES */

Pcte_error_type set_several_attributes (
    in Pcte_h_attribute_assignments    attributes
);

/* 11.2.1 ARCHIVE_CREATE */

Pcte_error_type archive_create (
    in Pcte_natural                    archive_identifier,
    in Pcte_object_reference           on_same_volume_as,
    out Pcte_atomic_access_rights      access_mask,
    out Pcte_object_reference          new_archive
);

/* 12.2.6 CONTENTS_OPEN */

Pcte_error_type contents_open (
    in Pcte_contents_access_mode       opening_mode,
    in Pcte_boolean                    non_blocking_io,
    in Pcte_boolean                    inheritable,
    out Pcte_contents                  contents
);

};

interface Pcte_h_version {

/* This interface is applied to the PCTE object type "object" \f B. */

/* 9.4.5 VERSION_REVISE */

Pcte_error_type revise (
    in Pcte_object_reference           new_origin,
    in Pcte_link_reference             new_link,
    in Pcte_object_reference           on_same_volume_as,
    in Pcte_atomic_access_rights       access_mask,
    out Pcte_object_reference          new_version
);

/* 9.4.6 VERSION_SNAPSHOT */

Pcte_error_type snapshot (
    in Pcte_object_reference           new_origin,
    in Pcte_link_reference             new_link,
    in Pcte_object_reference           on_same_volume_as,
    in Pcte_atomic_access_rights       access_mask,
    out Pcte_object_reference          new_version
);

};

#endif
```

10 Schema management

```
/* The source file "sms.idl" */

#ifndef PCTE_SMS_INCLUDED
#define PCTE_SMS_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"#include "oms_types.idl"
```

10.1 Schema management datatypes

```
enum Pcte_definition_mode_value {
    PCTE_CREATE_MODE,
    PCTE_DELETE_MODE,
    PCTE_READ_MODE,
    PCTE_WRITE_MODE,
    PCTE_NAVIGATE_MODE
};

typedef Pcte_natural Pcte_definition_mode_values;

enum Pcte_duplication {
    PCTE_DUPLICATED, PCTE_NOT_DUPLICATED
};

enum Pcte_exclusiveness {
    PCTE_SHARABLE, PCTE_EXCLUSIVE
};

enum Pcte_stability {
    PCTE_ATOMIC_STABLE, PCTE_COMPOSITE_STABLE, PCTE_NOT_STABLE
};

enum Pcte_contents_type {
    PCTE_NO_CONTENTS, PCTE_FILE_TYPE,
    PCTE_PIPE_TYPE, PCTE_DEVICE_TYPE,
    PCTE_AUDIT_FILE_TYPE, PCTE_ACCOUNTING_LOG_TYPE
};
/*  Pcte_contents_type corresponds to the PCTE datatype Contents_type.  The value   */
/*  PCTE_NO_CONTENTS corresponds to the absence of a Contents_type result from      */
/*  SDS_GET_OBJECT_TYPE_PROPERTIES and                                              */
/*  WS_GET_OBJECT_TYPE_PROPERTIES.                                                  */

struct Pcte_link_flags {
    Pcte_category       category;
    Pcte_stability      stability;
    Pcte_exclusiveness  exclusiveness;
    Pcte_duplication    duplication;
};
```

```
struct Pcte_link_type_properties {
    Pcte_link_flags   link_type_flag;
    Pcte_natural      lower_bound, upper_bound;
};

/*  Pcte_link_type_properties corresponds to a number of parameter types in      */
/*  SDS_CREATE_RELATIONSHIP_TYPE, and to a number of result types of             */
/*  SDS_GET_LINK_TYPE_PROPERTIES and                                             */
/*  WS_GET_LINK_TYPE_PROPERTIES.                                                 */

enum Pcte_attribute_scan_kind {
    PCTE_OBJECT, PCTE_OBJECT_ALL, PCTE_LINK_KEY, PCTE_LINK_NON_KEY
};

enum Pcte_link_scan_kind {
    PCTE_ORIGIN, PCTE_ORIGIN_ALL, PCTE_DESTINATION,
    PCTE_DESTINATION_ALL, PCTE_KEY_ATTR, PCTE_NON_KEY_ATTR
};

enum Pcte_object_scan_kind {
    PCTE_CHILD, PCTE_DESCENDANT, PCTE_PARENT, PCTE_ANCESTOR,
    PCTE_ATTRIBUTE, PCTE_ATTRIBUTE_ALL, PCTE_LINK_ORIGIN,
    PCTE_LINK_ORIGIN_ALL, PCTE_LINK_DESTINATION,
    PCTE_LINK_DESTINATION_ALL
};

enum Pcte_type_kind {
    PCTE_OBJECT_TYPE, PCTE_LINK_TYPE, PCTE_ATTRIBUTE_TYPE,
    PCTE_ENUMERAL_TYPE
};

#define PCTE_MAX_ENUMERAL_TYPE_IMAGE_SIZE PCTE_MAX_NAME_SIZE

typedef Pcte_octet Pcte_enumeral_type_image
    [PCTE_MAX_ENUMERAL_TYPE_IMAGE_SIZE + 1];
```

10.2 Update operations

```
interface Pcte_sds {

/*  This interface is applied to the PCTE object type "sds".      */

/* 10.2.1 SDS_ADD_DESTINATION */

Pcte_error_type add_destination (
    in Pcte_type_name_in_sds   link_type,
    in Pcte_type_name_in_sds   object_type
);
```

```
/* 10.2.2 SDS_APPLY_ATTRIBUTE_TYPE */
Pcte_error_type apply_attribute_type (
    in Pcte_type_name_in_sds   attribute_type,
    in Pcte_type_name_in_sds   type
);
/* 10.2.3 SDS_APPLY_LINK_TYPE */
Pcte_error_type apply_link_type (
    in Pcte_type_name_in_sds   link_type,
    in Pcte_type_name_in_sds   object_type
);
/* 10.2.4 SDS_CREATE_BOOLEAN_ATTRIBUTE_TYPE */
Pcte_error_type create_boolean_attribute_type (
    in Pcte_name                    local_name,
    in Pcte_boolean                 initial_value,
    in Pcte_duplication             duplication,
    out Pcte_type_name_in_sds       new_type
);
/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL. The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value as */
/*  PCTE_FALSE.                                                                                 */
/* 10.2.5 SDS_CREATE_DESIGNATION_LINK_TYPE */
Pcte_error_type create_designation_link_type (
    in Pcte_name                    local_name,
    in Pcte_natural                 lower_bound,
    in Pcte_natural                 upper_bound,
    in Pcte_duplication             duplication,
    in Pcte_key_types_in_sds        key_types,
    out Pcte_type_name_in_sds       new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.  The effect of not providing the optional        */
/*  parameter upper_bound to the abstract operation is achieved by specifying upper_bound       */
/*  as 0.                                                                                       */
/* 10.2.6 SDS_CREATE_ENUMERAL_TYPE */
Pcte_error_type create_enumeral_type (
    in Pcte_name                    local_name,
    out Pcte_type_name_in_sds       new_type
);
/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */
```

```
/* 10.2.7 SDS_CREATE_ENUMERATION_ATTRIBUTE_TYPE */

Pcte_error_type create_enumeration_attribute_type (
    in Pcte_name                local_name,
    in Pcte_type_names_in_sds   values,
    in Pcte_duplication         duplication,
    in Pcte_natural             initial_value,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.  The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value   */
/*  as 0.                                                                                       */

/* 10.2.8 SDS_CREATE_FLOAT_ATTRIBUTE_TYPE */

Pcte_error_type create_float_attribute_type (
    in Pcte_name                local_name,
    in Pcte_float               initial_value,
    in Pcte_duplication         duplication,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.  The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value   */
/*  as 0.0.                                                                                     */

/* 10.2.9 SDS_CREATE_INTEGER_ATTRIBUTE_TYPE */

Pcte_error_type create_integer_attribute_type (
    in Pcte_name                local_name,
    in Pcte_integer             initial_value,
    in Pcte_duplication         duplication,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL. The effect of not providing the optional         */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value   */
/*  as 0.                                                                                       */

/* 10.2.10 SDS_CREATE_NATURAL_ATTRIBUTE_TYPE */

Pcte_error_type create_natural_attribute_type (
    in Pcte_name                local_name,
    in Pcte_natural             initial_value,
    in Pcte_duplication         duplication,
    out Pcte_type_name_in_sds   new_type
);
```

```
/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL. The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value  */
/*  as 0.                                                                                       */

/* 10.2.11 SDS_CREATE_OBJECT_TYPE */

Pcte_error_type create_object_type (
    in Pcte_name                local_name,
    in Pcte_type_names_in_sds   parents,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */

/* 10.2.12 SDS_CREATE_RELATIONSHIP_TYPE */

Pcte_error_type create_relationship_type (
    in Pcte_name                forward_local_name,
    in Pcte_link_type_properties forward_properties,
    in Pcte_key_types_in_sds    forward_key_types,
    in Pcte_name                reverse_local_name,
    in Pcte_link_type_properties reverse_properties,
    in Pcte_key_types_in_sds    reverse_key_types,
    out Pcte_type_name_in_sds   forward_type,
    out Pcte_type_name_in_sds   reverse_type
);

/*  The effect of not providing the optional parameter forward_local_name to the abstract       */
/*  operation is achieved by specifying forward_local_name as NULL. The effect of not          */
/*  providing the optional parameter reverse_local_name to the abstract operation is achieved  */
/*  by specifying reverse_local_name as NULL. The effect of not providing the optional         */
/*  parameter forward_upper_bound to the abstract operation is achieved by specifying          */
/*  forward_properties.upper_bound as 0. The effect of not providing the optional              */
/*  parameter reverse_upper_bound to the abstract operation is achieved by specifying          */
/*  reverse_properties.upper_bound as 0.                                                        */

/* 10.2.13 SDS_CREATE_STRING_ATTRIBUTE_TYPE */

Pcte_error_type create_string_attribute_type (
    in Pcte_name                local_name,
    in Pcte_string              initial_value,
    in Pcte_duplication         duplication,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL. The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value  */
/*  as NULL.                                                                                    */
```

```
/* 10.2.14 SDS_CREATE_TIME_ATTRIBUTE_TYPE */

Pcte_error_type create_time_attribute_type (
    in Pcte_name                local_name,
    in Pcte_time                initial_value,
    in Pcte_duplication         duplication,
    out Pcte_type_name_in_sds   new_type
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.  The effect of not providing the optional        */
/*  parameter initial_value to the abstract operation is achieved by specifying initial_value  */
/*  as Pcte_reference_time.                                                                     */

/* 10.2.15 SDS_GET_NAME */

Pcte_error_type get_name (
    out Pcte_name    name
);

/* 10.2.16 SDS_IMPORT_ATTRIBUTE_TYPE */

Pcte_error_type import_attribute_type (
    in Pcte_object_reference    from_sds,
    in Pcte_type_name_in_sds    type,
    in Pcte_name                local_name
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */

/* 10.2.17 SDS_IMPORT_ENUMERAL_TYPE */

Pcte_error_type import_enumeral_type (
    in Pcte_object_reference    from_sds,
    in Pcte_type_name_in_sds    type,
    in Pcte_name                local_name
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */

/* 10.2.18 SDS_IMPORT_LINK_TYPE */

Pcte_error_type import_link_type (
    in Pcte_object_reference    from_sds,
    in Pcte_type_name_in_sds    type,
    in Pcte_name                local_name
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */
```

```
/* 10.2.19 SDS_IMPORT_OBJECT_TYPE */

Pcte_error_type import_object_type (
    in Pcte_object_reference     from_sds,
    in Pcte_type_name_in_sds     type,
    in Pcte_name                 local_name
);

/*  The effect of not providing the optional parameter local_name to the abstract operation is  */
/*  achieved by specifying local_name as NULL.                                                  */

/* 10.2.20 SDS_INITIALIZE */

Pcte_error_type initialize (
    in Pcte_name      name
);

/* 10.2.21 SDS_REMOVE */

Pcte_error_type remove (
);

/* 10.2.22 SDS_REMOVE_DESTINATION */

Pcte_error_type remove_destination (
    in Pcte_type_name_in_sds   link_type,
    in Pcte_type_name_in_sds   object_type
);

/* 10.2.23 SDS_REMOVE_TYPE */

Pcte_error_type remove_type (
    in Pcte_type_name_in_sds   type
);

/* 10.2.24 SDS_SET_ENUMERAL_TYPE_IMAGE */

Pcte_error_type set_enumeral_type_image (
    in Pcte_type_name_in_sds       type,
    in Pcte_enumeral_type_image    image
);

/*  The effect of not providing the optional parameter image to the abstract operation is       */
/*  achieved by specifying image as NULL.                                                       */

/* 10.2.25 SDS_SET_TYPE_MODES */

Pcte_error_type set_usage_mode (
    in Pcte_type_name_in_sds          type,
    in Pcte_definition_mode_values    usage_mode
);
```

```
Pcte_error_type set_export_mode (
    in Pcte_type_name_in_sds        type,
    in Pcte_definition_mode_values  export_mode
);
/*  The effect of not providing the optional parameter export_mode to the abstract operation  */
/*  is obtained by calling Pcte_sds_set_usage_mode.  The effect of not providing the optional  */
/*  parameter usage_mode is obtained by calling Pcte_sds_set_export_mode.  The effect of       */
/*  providing both optional parameters usage_mode and export_mode is obtained by calling       */
/*  Pcte_sds_set_usage_mode and Pcte_sds_set_export_mode in sequence. As an operation          */
/*  call with neither optional parameter has no effect, no means for making such a call is     */
/*  provided.                                                                                  */

/* 10.2.26 SDS_SET_TYPE_NAME */

Pcte_error_type set_type_name (
    in Pcte_type_name_in_sds  type,
    in Pcte_name              local_name
);
/*  The effect of not providing the optional parameter image to the abstract operation is      */
/*  achieved by specifying image as NULL.                                                      */

/* 10.2.27 SDS_UNAPPLY_ATTRIBUTE_TYPE */

Pcte_error_type unapply_attribute_type (
    in Pcte_type_name_in_sds  attribute_type,
    in Pcte_type_name_in_sds  type
);

/* 10.2.28 SDS_UNAPPLY_LINK_TYPE */

Pcte_error_type unapply_link_type (
    in Pcte_type_name_in_sds  link_type,
    in Pcte_type_name_in_sds  object_type
);
```

10.3 Usage operations

```
/* 10.3.1 SDS_GET_ATTRIBUTE_TYPE_PROPERTIES */

Pcte_error_type get_attribute_type_properties (
    in Pcte_type_name_in_sds                type,
    out Pcte_duplication                    duplication,
    out Pcte_value_type                     value_type,
    out Pcte_enumeration_value_type_in_sds  enumeration_value_type,
    out Pcte_attribute_value                initial_value
);
/*  If the abstract operation returns an enumeration value type in value_type then value_type  */
/*  is set to PCTE_ENUMERATION_VALUE_TYPE and enumeration_value_type                           */
/*  contains the sequence of enumeration value type nominators.                                */
```

```
/* 10.3.2 SDS_GET_ENUMERAL_TYPE_IMAGE */

Pcte_error_type get_enumeral_type_image (
    in Pcte_type_name_in_sds        enumeral_type,
    out Pcte_enumeral_type_image    image
);

/* 10.3.3 SDS_GET_ENUMERAL_TYPE_POSITION */

Pcte_error_type get_enumeral_type_position (
    in Pcte_type_name_in_sds  enumeral_type,
    in Pcte_type_name_in_sds  attribute_type,
    out Pcte_natural          position
);

/* 10.3.4 SDS_GET_LINK_TYPE_PROPERTIES */

Pcte_error_type get_link_type_properties (
    in Pcte_type_name_in_sds       type,
    out Pcte_link_type_properties  properties,
    out Pcte_key_types_in_sds      key_types,
    out Pcte_type_name_in_sds      reverse
);

/*  The category, lower bound, upper bound, exclusiveness, stability, duplication, key types,  */
/*  and reverse values are returned in the members with the corresponding names of the         */
/*  Pcte_link_type_properties object pointed to by properties.  If the abstract operation      */
/*  returns no value in reverse , reverse is set to NULL.                                      */

/* 10.3.5 SDS_GET_OBJECT_TYPE_PROPERTIES */

Pcte_error_type get_object_type_properties (
    in Pcte_type_name_in_sds       type,
    out Pcte_contents_type         contents_type,
    out Pcte_type_names_in_sds     parents,
    out Pcte_type_names_in_sds     children
);

/*  If the abstract operation returns no value in contents_type then contents_type is set to   */
/*  PCTE_NO_CONTENTS.                                                                          */

/* 10.3.6 SDS_GET_TYPE_KIND */

Pcte_error_type get_type_kind (
    in Pcte_type_name_in_sds  type,
    out Pcte_type_kind        type_kind
);
```

```
/* 10.3.7 SDS_GET_TYPE_MODES */

Pcte_error_type get_type_modes (
    in Pcte_type_name_in_sds          type,
    out Pcte_definition_mode_values   usage_mode,
    out Pcte_definition_mode_values   export_mode,
    out Pcte_definition_mode_values   max_usage_mode
);

/* 10.3.8 SDS_GET_TYPE_NAME */

Pcte_error_type get_type_name (
    in Pcte_type_name_in_sds  type,
    out Pcte_type_name        name
);

/* 10.3.9 SDS_SCAN_ATTRIBUTE_TYPE */

Pcte_error_type scan_attribute_type (
    in Pcte_type_name_in_sds      type,
    in Pcte_attribute_scan_kind   scanning_kind,
    out Pcte_type_names_in_sds    types
);

/* 10.3.10 SDS_SCAN_ENUMERAL_TYPE */

Pcte_error_type scan_enumeral_type (
    in Pcte_type_name_in_sds      type,
    out Pcte_type_names_in_sds    types
);

/* 10.3.11 SDS_SCAN_LINK_TYPE */

Pcte_error_type scan_link_type (
    in Pcte_type_name_in_sds      type,
    in Pcte_link_scan_kind        scanning_kind,
    out Pcte_type_names_in_sds    types
);

/* 10.3.12 SDS_SCAN_OBJECT_TYPE */

Pcte_error_type scan_object_type (
    in Pcte_type_name_in_sds      type,
    in Pcte_object_scan_kind      scanning_kind,
    out Pcte_type_names_in_sds    types
);

/* 10.3.13 SDS_SCAN_TYPES */

Pcte_error_type scan_types (
    in Pcte_type_kind             kind,
    out Pcte_type_names_in_sds    types
);
```

```
    Pcte_error_type scan_all_types (
    out Pcte_type_names_in_sds    types
);

/*  The effect of not providing the optional parameter kind to the abstract operation is   */
/*  achieved by the operation Pcte_sds_scan_all_types.                                     */

};
```

10.4 Working schema operations

```
interface Pcte_ws {

/*  This interface is conventionally applied to the PCTE object type "process".  The      */
/*  controlling object must be the current process Pcte_current_process.                   */

/* 10.4.1 WS_GET_ATTRIBUTE_TYPE_PROPERTIES */

Pcte_error_type get_attribute_type_properties (
    in Pcte_type_name                 type,
    out Pcte_duplication              duplication,
    out Pcte_value_type               value_type,
    out Pcte_enumeration_value_type   enumeration_value_type,
    out Pcte_attribute_value          initial_value
);

/* 10.4.2 WS_GET_ENUMERAL_TYPE_IMAGE */

Pcte_error_type get_enumeral_type_image (
    in Pcte_type_name                 enumeral_type,
    out Pcte_enumeral_type_image      image
);

/* 10.4.3 WS_GET_ENUMERAL_TYPE_POSITION */

Pcte_error_type get_enumeral_type_position (
    in Pcte_type_name   enumeral_type,
    in Pcte_type_name   attribute_type,
    out Pcte_natural    position
);

/* 10.4.4 WS_GET_LINK_TYPE_PROPERTIES */

Pcte_error_type get_link_type_properties (
    in Pcte_type_name               type,
    out Pcte_link_type_properties   properties,
    out Pcte_key_types              key_types,
    out Pcte_type_name              reverse
);
```

```
/* 10.4.5 WS_GET_OBJECT_TYPE_PROPERTIES */

Pcte_error_type get_object_type_properties (
    in Pcte_type_name       type,
    out Pcte_contents_type  contents_type,
    out Pcte_type_names     parents,
    out Pcte_type_names     children
);

/*  If the abstract operation returns no value in contents_type then contents_type is set to   */
/*  PCTE_NO_CONTENTS.                                                                           */

/* 10.4.6 WS_GET_TYPE_KIND */

Pcte_error_type get_type_kind (
    in Pcte_type_name   type,
    out Pcte_type_kind  type_kind
);

/* 10.4.7 WS_GET_TYPE_MODES */

Pcte_error_type get_type_modes (
    in Pcte_type_name                   type,
    out Pcte_definition_mode_values     usage_modes
);

/* 10.4.8 WS_GET_TYPE_NAME */

Pcte_error_type get_type_name (
    in Pcte_type_name       type,
    out Pcte_type_name      name
);

/* 10.4.9 WS_SCAN_ATTRIBUTE_TYPE */

Pcte_error_type scan_attribute_type (
    in Pcte_type_name               type,
    in Pcte_attribute_scan_kind     scanning_kind,
    out Pcte_type_names             types
);

/* 10.4.10 WS_SCAN_ENUMERAL_TYPE */

Pcte_error_type scan_enumeral_type (
    in Pcte_type_name       type,
    out Pcte_type_names     types
);
```

```
/* 10.4.11 WS_SCAN_LINK_TYPE */

Pcte_error_type scan_link_type (
    in Pcte_type_name          type,
    in Pcte_link_scan_kind     scanning_kind,
    out Pcte_type_names        types
);

/* 10.4.12 WS_SCAN_OBJECT_TYPE */

Pcte_error_type scan_object_type (
    in Pcte_type_name          type,
    in Pcte_object_scan_kind   scanning_kind,
    out Pcte_type_names        types
);

/* 10.4.13 WS_SCAN_TYPES */

Pcte_error_type scan_types (
    in Pcte_type_kind          kind,
    out Pcte_type_names        types
);

};

interface Pcte_h_ws {

/* This interface is conventionally applied to the PCTE object type "process". */

/* 10.4.1 WS_GET_ATTRIBUTE_TYPE_PROPERTIES */

Pcte_error_type get_attribute_type_properties (
    in Pcte_type_reference              type,
    out Pcte_duplication                duplication,
    out Pcte_value_type                 value_type,
    out Pcte_h_enumeration_value_type   enumeration_value_type,
    out Pcte_attribute_value            initial_value
);

/* 10.4.2 WS_GET_ENUMERAL_TYPE_IMAGE */

Pcte_error_type get_enumeral_type_image (
    in Pcte_type_reference     enumeral_type,
    out Pcte_string            image
);

/* 10.4.3 WS_GET_ENUMERAL_TYPE_POSITION */

Pcte_error_type get_enumeral_type_position (
    in Pcte_type_reference     enumeral_type,
    in Pcte_type_reference     attribute_type,
    out Pcte_natural           position
);
```

```
/* 10.4.4 WS_GET_LINK_TYPE_PROPERTIES */

Pcte_error_type get_link_type_properties (
    in Pcte_type_reference          type,
    out Pcte_link_type_properties   properties,
    out Pcte_h_key_types            key_types,
    out Pcte_type_reference         reverse
);

/* 10.4.5 WS_GET_OBJECT_TYPE_PROPERTIES */

Pcte_error_type get_object_type_properties (
    in Pcte_type_reference      type,
    out Pcte_contents_type      contents_type,
    out Pcte_type_references    parents,
    out Pcte_type_references    children
);

/*  If the abstract operation returns no value in contents_type then contents_type is set to    */
/*  PCTE_NO_CONTENTS.                                                                           */

/* 10.4.6 WS_GET_TYPE_KIND */

Pcte_error_type get_type_kind (
    in Pcte_type_reference  type,
    out Pcte_type_kind      type_kind
);

/* 10.4.7 WS_GET_TYPE_MODES */

Pcte_error_type get_type_modes (
    in Pcte_type_reference              type,
    out Pcte_definition_mode_values     usage_modes
);

/* 10.4.8 WS_GET_TYPE_NAME */

Pcte_error_type get_type_name (
    in Pcte_type_reference  type,
    out Pcte_type_name      name
);

/* 10.4.9 WS_SCAN_ATTRIBUTE_TYPE */

Pcte_error_type scan_attribute_type (
    in Pcte_type_reference          type,
    in Pcte_attribute_scan_kind     scanning_kind,
    out Pcte_type_references        types
);
```

```
/* 10.4.10 WS_SCAN_ENUMERAL_TYPE */

Pcte_error_type scan_enumeral_type (
    in Pcte_type_reference      type,
    out Pcte_type_references    types
);

/* 10.4.11 WS_SCAN_LINK_TYPE */

Pcte_error_type scan_link_type (
    in Pcte_type_reference      type,
    in Pcte_link_scan_kind      scanning_kind,
    out Pcte_type_references    types
);

/* 10.4.12 WS_SCAN_OBJECT_TYPE */

Pcte_error_type scan_object_type (
    in Pcte_type_reference      type,
    in Pcte_object_scan_kind    scanning_kind,
    out Pcte_type_references    types
);

/* 10.4.13 WS_SCAN_TYPES */

Pcte_error_type scan_types (
    in Pcte_type_kind           kind,
    out Pcte_type_references    types
);

};

#endif
```


11 Volumes, devices, archives, and clusters

```
/* The source file "devices.idl" */

#ifndef PCTE_DEVICES_INCLUDED
#define PCTE_DEVICES_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "discretionary_types.idl"
#include "mandatory_types.idl"
```

11.1 Volume, device, archive, and cluster datatypes

```
typedef Pcte_natural Pcte_volume_identifier;
```

```
struct Pcte_volume_status {
    Pcte_natural            total_blocks;
    Pcte_natural            free_blocks;
    Pcte_natural            block_size;
    Pcte_natural            num_objects;
    Pcte_volume_identifier  volume_identifier;
};

typedef Pcte_natural Pcte_device_identifier;

enum Pcte_archive_status {
    PCTE_PARTIAL, PCTE_COMPLETE
};

typedef Pcte_natural Pcte_archive_identifier;
```

11.2 Volume, device, and archive operations

```
interface Pcte_archive {

/* This interface is applied to the PCTE object type "archive". */

/* 11.2.1 ARCHIVE_CREATE */

/* See 9.3 and 9.5.*/

/* 11.2.2 ARCHIVE_REMOVE */

Pcte_error_type remove (
);

/* 11.2.3 ARCHIVE_RESTORE */

Pcte_error_type restore (
    in Pcte_object_reference    device,
    in Pcte_object_reference    archive,
    in Pcte_object_references   objects,
    in Pcte_object_reference    on_same_volume_as,
    out Pcte_archive_status     restoring_status
);

Pcte_error_type restore_all (
    in Pcte_object_reference    device,
    in Pcte_object_reference    archive,
    in Pcte_object_reference    on_same_volume_as,
    out Pcte_archive_status     restoring_status
);
```

```
/* 11.2.4 ARCHIVE_SAVE */

Pcte_error_type save (
    in Pcte_object_reference      device,
    in Pcte_object_reference      archive,
    in Pcte_object_references     objects,
    out Pcte_archive_status       archiving_status
);

};

interface Pcte_device {

/* This interface is applied to the PCTE object type "device". */

/* 11.2.5 DEVICE_CREATE */

/* See 18.5. */

/* 11.2.6 DEVICE_REMOVE */

Pcte_error_type remove (
    in Pcte_object_reference      device
);

/* 11.2.7 LINK_GET_DESTINATION_ARCHIVE */

/* See 9.2. */

/* 20.2.1 DEVICE_SET_CONFIDENTIALITY_RANGE */

Pcte_error_type set_confidentiality_range (
    in Pcte_security_label    high_label,
    in Pcte_security_label    low_label
);

/* 20.2.2 DEVICE_SET_INTEGRITY_RANGE */

Pcte_error_type set_integrity_range (
    in Pcte_security_label    high_label,
    in Pcte_security_label    low_label
);

};

interface Pcte_volume {

/* This interface is applied to the PCTE object type "volume" */
```

```
/* 11.2.8 VOLUME_CREATE */

Pcte_error_type create (
    in Pcte_object_reference        device,
    in Pcte_natural                 volume_id,
    in Pcte_atomic_access_rights    access_mask,
    in Pcte_string                  volume_characteristics,
    out Pcte_object_reference       new_volume
);

/* 11.2.9 VOLUME_DELETE */

Pcte_error_type delete (
    in Pcte_object_reference    volume
);

/* 11.2.10 VOLUME_GET_STATUS */

Pcte_error_type get_status (
    in Pcte_object_reference    volume,
    out Pcte_volume_status      volume_status
);

/* 11.2.11 VOLUME_MOUNT */

Pcte_error_type mount (
    in Pcte_object_reference    device,
    in Pcte_volume_identifier   volume_identifier,
    in Pcte_boolean             read_only
);

/* 11.2.12 VOLUME_UNMOUNT */

Pcte_error_type unmount (
    in Pcte_object_reference    volume
);

/* 20.2.8 VOLUME_SET_CONFIDENTIALITY_RANGE */

Pcte_error_type set_confidentiality_range (
    in Pcte_security_label  high_label,
    in Pcte_security_label  low_label
);

/* 20.2.8 VOLUME_SET_INTEGRITY_RANGE */

Pcte_error_type set_integrity_range (
    in Pcte_security_label  high_label,
    in Pcte_security_label  low_label
);
```

```
/* 9.3.20 VOLUME_LIST_OBJECTS */

Pcte_error_type list_objects (
in Pcte_type_names          types,
out Pcte_object_references  objects
);

};

interface Pcte_h_volume {

/* This interface is applied to the PCTE object type "volume". */

/* 9.3.20 VOLUME_LIST_OBJECTS */

Pcte_error_type list_objects (
in Pcte_type_references     types,
out Pcte_object_references  objects
);

};

#endif
```

11.3 Cluster operations

```
/* The source file "clusters.idl" */

#ifndef    PCTE_CLUSTERS_INCLUDED
#define    PCTE_CLUSTERS_INCLUDED  1

#include       "types.idl"
#include       "references.idl"
#include       "sequences.idl"
#include       "security.idl"

interface Pcte_cluster {

/* This interface is applied to the PCTE object type "cluster". */

/* 11.3.1 CLUSTER_CREATE */

Pcte_error_type create (
    Pcte_object_reference        volume,
    Pcte_natural                 cluster_id,
    Pcte_atomic_access_rights    access_mask,
    Pcte_string                  cluster_characteristics,
    Pcte_object_reference        new_cluster
);

/* 11.3.2 CLUSTER_DELETE */

Pcte_error_type delete (
    Pcte_object_reference    cluster
);
```

```
/* 11.3.3 CLUSTER_LIST_OBJECTS */

Pcte_error_type list_objects (
    Pcte_object_reference    cluster,
    Pcte_type_references     types,
    Pcte_object_references   objects
);

};

#endif
```

12 Files, pipes, and devices

```
/* The source file "contents.idl" */

#ifndef PCTE_CONTENTS_INCLUDED
#define PCTE_CONTENTS_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "contents_types.idl"
```

12.1 File, pipe, and device datatypes

```
/* The source file "contents_types.idl" */

#ifndef PCTE_CONTENTS_TYPES_INCLUDED
#define PCTE_CONTENTS_TYPES_INCLUDED 1

enum Pcte_contents_access_mode {
    PCTE_READ_WRITE, PCTE_READ_ONLY,
    PCTE_WRITE_ONLY, PCTE_APPEND_ONLY
};

enum Pcte_seek_position {
    PCTE_FROM_BEGINNING, PCTE_FROM_CURRENT, PCTE_FROM_END
};

enum Pcte_set_position {
    PCTE_AT_BEGINNING, PCTE_AT_POSITION, PCTE_AT_END
};

enum Pcte_positioning_style {
    PCTE_SEQUENTIAL, PCTE_DIRECT, PCTE_SEEK
};

#endif
```

12.2 File, pipe, and device operations

```
interface Pcte_position_handle;

interface Pcte_contents {
```

```
/* This interface is applied to the PCTE object type "file". */

/* 12.2.1 CONTENTS_CLOSE */

Pcte_error_type close (
);

/* 12.2.2 CONTENTS_GET_HANDLE_FROM_KEY */

/* See 13.2 and 13.8. */

/* 12.2.3 CONTENTS_GET_KEY_FROM_HANDLE */

Pcte_error_type get_key_from_handle (
    out Pcte_natural  open_object_key
);

/* 12.2.4 CONTENTS_GET_POSITION */

Pcte_error_type get_position (
    out Pcte_position_handle    position
);

/* 12.2.5 CONTENTS_HANDLE_DUPLICATE */

Pcte_error_type handle_duplicate (
    in Pcte_boolean       inheritable,
    out Pcte_contents     new_contents
);

Pcte_error_type handle_duplicate_to_key (
    in Pcte_natural       new_key,
    in Pcte_boolean       inheritable,
    out Pcte_contents     new_contents
);

/* 12.2.6 CONTENTS_OPEN */

/* See 9.3 and 9.5. */

/* 12.2.7 CONTENTS_READ */

Pcte_error_type read (
    in Pcte_natural    size,
    out Pcte_octet     data,
    out Pcte_natural   data_size
);

/* 12.2.8 CONTENTS_SEEK */

Pcte_error_type seek (
    in Pcte_integer            offset,
    in Pcte_seek_position      whence,
    out Pcte_natural           new_position
);
```

```
/* 12.2.9 CONTENTS_SET_POSITION */

Pcte_error_type set_position (
    in Pcte_position_handle position_handle,
    in Pcte_set_position    set_mode
);

/* 12.2.10 CONTENTS_SET_PROPERTIES */

Pcte_error_type set_properties (
    in Pcte_positioning_style    positioning
);

/* 12.2.11 CONTENTS_TRUNCATE */

Pcte_error_type truncate (
);

/* 12.2.12 CONTENTS_WRITE */

Pcte_error_type write (
    in Pcte_octet      data,
    in Pcte_natural    data_size,
    out Pcte_natural   actual_size
);

/* 12.2.13 DEVICE_GET_CONTROL */

Pcte_error_type get_control (
    in Pcte_natural    operation,
    out Pcte_string    control_data
);

/* 12.2.14 DEVICE_SET_CONTROL */

Pcte_error_type set_control (
    in Pcte_natural    operation,
    in Pcte_string     control_data
);

/* 18.3.1 CONTENTS_COPY_FROM_FOREIGN_SYSTEM */

Pcte_error_type copy_from_foreign_system (
    in Pcte_object_designator    foreign_system,
    in Pcte_string               foreign_name,
    in Pcte_string               foreign_parameters
);

/*  The effect of not providing the optional parameter foreign_parameters to the    */
/*  abstract operation is achieved by specifying foreign_parameters as NULL.        */
```

```
/* 18.3.2 CONTENTS_COPY_TO_FOREIGN_SYSTEM */

Pcte_error_type copy_to_foreign_system (
    in Pcte_object_designator   foreign_system,
    in Pcte_string              foreign_name,
    in Pcte_string              foreign_parameters
);

/*  The effect of not providing the optional parameter foreign_parameters to the abstract   */
/*  operation is achieved by specifying foreign_parameters as NULL.                         */

};

interface Pcte_position_handle {

/* This interface not is applied to any specific PCTE object type */

Pcte_error_type discard (                                                    //PIDL
);

};

#endif      // !PCTE_CONTENTS_INCLUDED
```

13 Process execution

```
/* The source file "execution.idl" */

#ifndef PCTE_EXECUTION_INCLUDED
#define PCTE_EXECUTION_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "discretionary_types.idl"
#include "accounting.idl"
#include "auditing.idl"
```

13.1 Process execution datatypes

```
typedef <implementation-defined> Pcte_address;

/*  Pcte_address corresponds to the PCTE datatype Address which must be                     */
/*  defined for each implementation.                                                        */

enum Pcte_initial_status {
    PCTE_SUSPENDED, PCTE_RUNNING, PCTE_STOPPED
};

#define PCTE_EXIT_SUCCESS           0
#define PCTE_EXIT_ERROR             1
#define PCTE_FORCED_TERMINATION     2
#define PCTE_SYSTEM_FAILURE         3
#define PCTE_ACTIVITY_ABORTED       4
#define PCTE_UNAVAILABLE            5
```

```
/*  An implementation may provide further values for the termination status of a process by  */
/*  extending this list of values.                                                           */

typedef long Pcte_profile_handle;

#include "mandatory.idl"

typedef Object Pcte_contents;

interface Pcte_process;

Pcte_current_process Pcte_process;

/* The PCTE current process is the process to be used as controlling object for operations  */
/* which can be invoked only for the current process.                                       */
```

13.2 Process execution operations

```
interface Pcte_h_process {

/* This interface is applied to the PCTE object type "process". */

/* 13.2.1 PROCESS_CREATE */

Pcte_error_type create (
    in Pcte_object_reference        static_context,
    in Pcte_type_reference          process_type,
    in Pcte_h_process               parent,
    in Pcte_object_reference        site,
    in Pcte_boolean                 implicit_deletion,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_h_process              new_process
);

/* 12.2.2 CONTENTS_GET_HANDLE_FROM_KEY */

Pcte_error_type get_handle_from_key (
    in Pcte_natural     open_object_key,
    out Pcte_contents   contents
);

};

interface Pcte_process {

/* This interface is applied to the PCTE object type "process". */

/* 13.2.1 PROCESS_CREATE */

/* Operation is applied to self. */
```

```
Pcte_error_type create (
    in Pcte_object_reference        static_context,
    in Pcte_type_name               process_type,
    in Pcte_process                 parent,
    in Pcte_object_reference        site,
    in Pcte_boolean                 implicit_deletion,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_object_reference       new_process
);

/*  The effect of not providing the optional parameter parent to the abstract operation is  */
/*  achieved by specifying parent as Pcte_null_object_reference.  The effect of not providing */
/*  the optional parameter site to the abstract operation is achieved by specifying site as  */
/*  Pcte_null_object_reference.                                                              */

/* 13.2.2 PROCESS_CREATE_AND_START */

/* Operation is applied to self. */

Pcte_error_type create_and_start (
    in Pcte_object_reference        static_context,
    in Pcte_string                  arguments,
    in Pcte_string                  environment,
    in Pcte_object_reference        site,
    in Pcte_boolean                 implicit_deletion,
    in Pcte_atomic_access_rights    access_mask,
    out Pcte_process                new_process
);

/*  The effect of not providing the optional parameter site to the abstract operation is     */
/*  achieved by specifying site as Pcte_null_object_reference.                               */

/* 13.2.3 PROCESS_GET_WORKING_SCHEMA */

Pcte_error_type get_working_schema (
    out Pcte_name_sequence    sds_sequence
);

/* 13.2.4 PROCESS_INTERRUPT_OPERATION */

Pcte_error_type interrupt_operation (
);

/* 13.2.5 PROCESS_RESUME */

Pcte_error_type resume (
);

/* 13.2.6 PROCESS_SET_ALARM */

Pcte_error_type set_alarm (
    in Pcte_natural    duration
);
```

```
/* 13.2.7 PROCESS_SET_FILE_SIZE_LIMIT */

Pcte_error_type set_file_size_limit (
    in Pcte_natural    fslimit
);

/* 13.2.8 PROCESS_SET_OPERATION_TIME_OUT */

Pcte_error_type set_operation_time_out (
    in Pcte_natural    duration
);

/* 13.2.9 PROCESS_SET_PRIORITY */

Pcte_error_type set_priority (
    in Pcte_natural    priority
);

/* 13.2.10 PROCESS_SET_REFERENCED_OBJECT */

Pcte_error_type set_referenced_object (
    in Pcte_key                 reference_name,
    out Pcte_object_reference   referenced_object
);

/* 13.2.11 PROCESS_SET_TERMINATION_STATUS */

Pcte_error_type set_termination_status (
    in Pcte_integer    termination_status
);

/* 13.2.12 PROCESS_SET_WORKING_SCHEMA */

Pcte_error_type set_working_schema (
    in Pcte_name_sequence    sds_sequence
);

/* 13.2.13 PROCESS_START */

Pcte_error_type start (
    in Pcte_string              arguments,
    in Pcte_string              environment,
    in Pcte_object_reference    site,
    in Pcte_initial_status      initial_status
);

/*  The effect of not providing the optional parameter site to the abstract operation is achieved */
/*  by specifying site as Pcte_null_object_reference.                                              */

/* 13.2.14 PROCESS_SUSPEND */

Pcte_error_type suspend (
    in Pcte_natural    alarm
);
```

```
Pcte_error_type suspend_unlimited (
);
/*  The effect of not providing the optional parameter alarm to the abstract operation is  */
/*  achieved by the operation suspend_unlimited.                                           */

/* 13.2.15 PROCESS_TERMINATE */

Pcte_error_type terminate (
    in Pcte_integer    termination_status
);
/*  The effect of not providing the optional parameter termination_status to the abstract  */
/*  operation is achieved by specifying termination_status as                              */
/*  PCTE_FORCED_TERMINATION.                                                               */

/* 13.2.16 PROCESS_UNSET_REFERENCED_OBJECT */

Pcte_error_type unset_referenced_object (
    in Pcte_key    reference_name
);

/* 13.2.17 PROCESS_WAIT_FOR_ANY_CHILD */

Pcte_error_type wait_for_any_child (
    out Pcte_integer  termination_status,
    out Pcte_natural  child
);

/* 13.2.18 PROCESS_WAIT_FOR_CHILD */

Pcte_error_type wait_for_child (
    in Pcte_object_reference    child,
    out Pcte_integer            termination_status
);
```

13.3 Security operations

```
/* 13.3.1 PROCESS_ADOPT_USER_GROUP */

Pcte_error_type adopt_user_group (
    in Pcte_object_reference    user_group
);

/* 13.3.2 PROCESS_GET_DEFAULT_ACL */

Pcte_error_type get_default_acl (
    out Pcte_acl   acl
);
```

```
/* 13.3.3 PROCESS_GET_DEFAULT_OWNER */

Pcte_error_type get_default_owner (
    out Pcte_group_identifier    group
);

/* 13.3.4 PROCESS_SET_ADOPTABLE_FOR_CHILD */

Pcte_error_type set_adoptable_for_child (
    in Pcte_object_reference     user_group,
    in Pcte_boolean              adoptability
);

/* 13.3.5 PROCESS_SET_DEFAULT_ACL_ENTRY */

Pcte_error_type set_default_acl_entry (
    in Pcte_group_identifier             group,
    in Pcte_requested_access_rights      modes
);

/* 13.3.6 PROCESS_SET_DEFAULT_OWNER */

Pcte_error_type set_default_owner (
    in Pcte_group_identifier     group
);

/* 13.3.7 PROCESS_SET_USER */

Pcte_error_type set_user (
    in Pcte_object_reference     user,
    in Pcte_object_reference     user_group
);
```

13.4 Profiling operations

```
/* 13.4.1 PROCESS_PROFILING_OFF */

Pcte_error_type profiling_off (
    in Pcte_profile_handle   handle,
    in Pcte_buffer           buffer
);

/* 13.4.2 PROCESS_PROFILING_ON */

Pcte_error_type profiling_on (
    in Pcte_address              start,
    in Pcte_address              end,
    in Pcte_natural              count,
    out Pcte_profile_handle      handle
);
```

13.5 Monitoring operations

```
/* 13.5.1 PROCESS_ADD_BREAKPOINT */

Pcte_error_type add_breakpoint (
    in Pcte_address   breakpoint
);

/* 13.5.2 PROCESS_CONTINUE */

Pcte_error_type continue (
);

/* 13.5.3 PROCESS_PEEK */

Pcte_error_type peek (
    in Pcte_address       address,
    out Pcte_octet        process_data,
    out Pcte_natural      process_data_size
);

/*  process_data_size is the number of octets to be read.  The octets read are returned in      */
/*  process_data and the number of octets read is returned in process_data_size.  If there is  */
/*  not enough space in process_data, the error PCTE_STRING_TOO_SHORT is raised.                */

/* 13.5.4 PROCESS_POKE */

Pcte_error_type poke (
    in Pcte_address       address,
    out Pcte_octet        process_data,
    out Pcte_natural      process_data_size
);

/*  process_data is the octets to be written, and process_data_size is the number of octets to  */
/*  be written.  If process_data_size is bigger than the number of octets allocated in          */
/*  process_data, the error PCTE_ACCESS_AT_INVALID_ADDRESS is raised.                           */

/* 13.5.5 PROCESS_REMOVE_BREAKPOINT */

Pcte_error_type remove_breakpoint (
    in Pcte_address   breakpoint
);

/* 13.5.6 PROCESS_WAIT_FOR_BREAKPOINT */

Pcte_error_type wait_for_breakpoint (
    out Pcte_address      breakpoint
);
```

13.6 Mandatory security operations

```
/* 20.4.1 PROCESS_SET_CONFIDENTIALITY_LABEL */

Pcte_error_type set_confidentiality_label (
    in Pcte_security_label    confidentiality_label
);

/* 20.4.2 PROCESS_SET_FLOATING_CONFIDENTIALITY_LEVEL */

Pcte_error_type set_floating_confidentiality_level (
    in Pcte_floating_level    floating_mode
);

/* 20.4.3 PROCESS_SET_FLOATING_INTEGRITY_LEVEL */

Pcte_error_type set_floating_integrity_level (
    in Pcte_floating_level    floating_mode
);

/* 20.4.4 PROCESS_SET_INTEGRITY_LABEL */

Pcte_error_type set_integrity_label (
    in Pcte_security_label    integrity_label
);
```

13.7 Consumer identity operations

```
/* 22.3.1 PROCESS_SET_CONSUMER_IDENTITY */

Pcte_error_type set_consumer_identity (
    in Pcte_consumer_group    group
);

/* 22.3.2 PROCESS_UNSET_CONSUMER_IDENTITY */

Pcte_error_type unset_consumer_identity (
);
```

13.8 Contents handle operation

```
/* 12.2.2 CONTENTS_GET_HANDLE_FROM_KEY */

Pcte_error_type get_handle_from_key (
    in Pcte_natural        open_object_key,
    out Pcte_contents      contents
);

};

#endif
```

14 Message queues

```
/* The source file "messages.idl" */
#ifndef PCTE_MESSAGES_INCLUDED
#define PCTE_MESSAGES_INCLUDED 1
#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "notification.idl"
#include "messages_types.idl"
```

14.1 Message queue datatypes

```
/* The source file "messages_types.idl" */
#ifndef PCTE_MESSAGES_TYPES_INCLUDED
#define PCTE_MESSAGES_TYPES_INCLUDED 1
enum Pcte_standard_message_type {
    PCTE_INTERRUPT_MSG, PCTE_QUIT_MSG, PCTE_FINISH_MSG,
    PCTE_SUSPEND_MSG, PCTE_END_MSG, PCTE_ABORT_MSG,
    PCTE_DEADLOCK_MSG, PCTE_WAKE_MSG
};
union Pcte_message_type_type switch (long) {
    case 1: Pcte_standard_message_type    standard;
    case 2: Pcte_notification_message_type notification;
    case 3: Pcte_natural                  implementation_message;
    case 4: Pcte_natural                  undefined;
};
enum Pcte_message_kind {
    PCTE_STANDARD_MESSAGE, PCTE_NOTIFICATION_MESSAGE,
    PCTE_IMPLEMENTATION_MESSAGE, PCTE_UNDEFINED_MESSAGE
};
struct Pcte_message_type {
    Pcte_message_kind      kind;
    Pcte_message_type_type type;
};
#define Pcte_all_message_types (Pcte_message_types) NULL
struct Pcte_message {
    Pcte_string       data;
    Pcte_message_type message_type;
};
struct Pcte_received_message {
    Pcte_message message;
    Pcte_natural position;
};
```

```
#endif
typedef Object Pcte_handler;                    // Pseudo-object, cached locally
```

14.2 Message queue operations

```
interface Pcte_queue {
/* This interface is applied to the PCTE object type "message_queue". */
/* 14.2.1 MESSAGE_DELETE */
Pcte_error_type delete (
    in Pcte_natural    position
);
/* 14.2.2 MESSAGE_PEEK */
Pcte_error_type peek (
    in Pcte_message_types       type,
    in Pcte_natural             position,
    out Pcte_received_message   message
);
/*  The effect of specifying types as ALL_MESSAGE_TYPES to the abstract operation is    */
/*  achieved by specifying types as Pcte_all_message_types.  The effect of not providing the  */
/*  optional parameter position to the abstract operation is achieved by specifying position  */
/*  as 0.  If the abstract operation returns no value in message then message is set to NULL.  */
/* 14.2.3 MESSAGE_RECEIVE_NO_WAIT */
Pcte_error_type receive_no_wait (
    in Pcte_message_types       types,
    in Pcte_natural             position,
    out Pcte_received_message   message
);
/*  The effect of specifying types as ALL_MESSAGE_TYPES to the abstract operation is    */
/*  achieved by specifying types as Pcte_all_message_types.  The effect of not providing the  */
/*  optional parameter position to the abstract operation is achieved by specifying position  */
/*  as 0.  If the abstract operation returns no value in message then message is set to NULL.  */
/* 14.2.4 MESSAGE_RECEIVE_WAIT */
Pcte_error_type receive_wait (
    in Pcte_message_types       types,
    in Pcte_natural             position,
    out Pcte_received_message   message
);
/*  The effect of not providing the optional parameter position to the abstract operation is  */
/*  achieved by specifying position as 0.                                                     */
```

```
/* 14.2.5 MESSAGE_SEND_NO_WAIT */

Pcte_error_type send_no_wait (
    in Pcte_message      message
);

/* 14.2.6 MESSAGE_SEND_WAIT */

Pcte_error_type send_wait (
    in Pcte_message      message
);

/* 14.2.7 QUEUE_EMPTY */

Pcte_error_type empty (
);

/* 14.2.8 QUEUE_HANDLER_DISABLE */

Pcte_error_type handler_disable (
);

/* 14.2.9 QUEUE_HANDLER_ENABLE */

Pcte_error_type handler_enable (
    in Pcte_message_types  types,
    in Pcte_handler        handler
);

/*  The effect of specifying types as ALL_MESSAGE_TYPES to the abstract operation is  */
/*  achieved by specifying types as Pcte_all_message_types.                           */

/* 14.2.10 QUEUE_RESERVE */

Pcte_error_type reserve (
);

/* 14.2.11 QUEUE_RESTORE */

Pcte_error_type restore (
    in Pcte_object_reference     file
);

/* 14.2.12 QUEUE_SAVE */

Pcte_error_type save (
    in Pcte_object_reference     file
);

/* 14.2.13 QUEUE_SET_TOTAL_SPACE */

Pcte_error_type set_total_space (
    in Pcte_natural      total_space
);
```

```
/* 14.2.14 QUEUE_UNRESERVE */

Pcte_error_type unreserve (
);

};

#endif      // !PCTE_MESSAGES_INCLUDED
```

15 Notification

```
/* The source file "notification.idl" */

#ifndef PCTE_NOTIFICATION_INCLUDED
#define PCTE_NOTIFICATION_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "notification_types.idl"
#include "messages_types.idl"
```

15.1 Notification datatypes

```
/* The source file "notification_types.idl" */

#ifndef PCTE_NOTIFICATION_TYPES_INCLUDED
#define PCTE_NOTIFICATION_TYPES_INCLUDED 1

enum Pcte_access_event {
    PCTE_MODIFICATION_EVENT,
    PCTE_CHANGE_EVENT,
    PCTE_DELETE_EVENT,
    PCTE_MOVE_EVENT
};

typedef Pcte_natural Pcte_access_events;

enum Pcte_notification_message_type {
    PCTE_MODIFICATION_MSG, PCTE_CHANGE_MSG,
    PCTE_DELETE_MSG, PCTE_MOVE_MSG,
    PCTE_NOT_ACCESSIBLE_MSG, PCTE_LOST_MSG
};

#endif      // !PCTE_NOTIFICATION_TYPES_INCLUDED
```

15.2 Notification operations

```
interface Pcte_notify {

/* This interface is applied to the PCTE object type "message_queue". */
```

```
/* 15.2.1 NOTIFICATION_MESSAGE_GET_KEY */

Pcte_error_type message_get_key (
    in Pcte_received_message   message,
    out Pcte_natural           notifier_key
);

/* 15.2.2 NOTIFY_CREATE */

Pcte_error_type create (
    in Pcte_natural            notifier_key,
    in Pcte_object_reference   monitored_object
);

/* 15.2.3 NOTIFY_DELETE */

Pcte_error_type delete (
    in Pcte_natural        notifier_key
);

/* 15.2.4 NOTIFY_SWITCH_EVENTS */

Pcte_error_type switch_events (
    in Pcte_natural            notifier_key,
    in Pcte_access_events      access_events
);

};

#endif
```

16 Concurrency and integrity control

```
/* The source file "activities.idl" */

#ifndef PCTE_ACTIVITIES_INCLUDED
#define PCTE_ACTIVITIES_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "discretionary_types.idl"
#include "oms_types.idl"
```

16.1 Concurrency and integrity control datatypes

```
enum Pcte_activity_class {
    PCTE_UNPROTECTED, PCTE_PROTECTED, PCTE_TRANSACTION
};
```

```
enum Pcte_lock_set_mode {
    PCTE_READ_UNPROTECTED, PCTE_READ_SEMIPROTECTED,
    PCTE_WRITE_UNPROTECTED, PCTE_WRITE_SEMIPROTECTED,
    PCTE_DELETE_UNPROTECTED, PCTE_DELETE_SEMIPROTECTED,
    PCTE_READ_PROTECTED, PCTE_WRITE_PROTECTED,
    PCTE_DELETE_PROTECTED, PCTE_WRITE_TRANSACTIONED,
    PCTE_DELETE_TRANSACTIONED, PCTE_READ_DEFAULT,
    PCTE_WRITE_DEFAULT, PCTE_DELETE_DEFAULT
};

typedef Pcte_lock_set_mode Pcte_lock_internal_mode;
```

16.2 Concurrency and integrity control operations

```
interface Pcte_activity {

/* This interface is applied to the PCTE object type "activity". */

/* 16.2.1 ACTIVITY_ABORT */

Pcte_error_type abort (
);

/* 16.2.2  ACTIVITY_END */

Pcte_error_type end (
);

/* 16.2.3  ACTIVITY_START */

Pcte_error_type start (
    in Pcte_activity_class    activity_class
);

};

interface Pcte_lock {

/* This interface is applied to the PCTE object type "object". */

/* 16.2.4 LOCK_RESET_INTERNAL_MODE */

Pcte_error_type reset_internal_mode (
);

/* 16.2.5 LOCK_SET_INTERNAL_MODE */

Pcte_error_type set_internal_mode (
    in Pcte_lock_internal_mode    lock_mode,
    in Pcte_boolean               wait_flag
);
```

```
/*  If the value PCTE_READ_DEFAULT, PCTE_WRITE_DEFAULT,                    */
/*  PCTE_DELETE_DEFAULT, PCTE_DELETE_PROTECTED,                            */
/*  PCTE_WRITE_TRANSACTIONED, or PCTE_DELETE_TRANSACTIONED is passed       */
/*  to lock_mode , the error PCTE_VALUE_IS_OUT_OF_RANGE is raised.         */

/* 16.2.6 LOCK_SET_OBJECT */

Pcte_error_type set_object (
    in Pcte_lock_set_mode  lock_mode,
    in Pcte_boolean        wait_flag,
    in Pcte_object_scope   scope
);

/* 16.2.7 LOCK_UNSET_OBJECT */

Pcte_error_type unset_object (
    in Pcte_object_scope   scope
);

};

#endif      // !PCTE_ACTIVITIES_INCLUDED
```

17 Replication

```
/* The source file "replication.idl" */

#ifndef PCTE_REPLICATION_INCLUDED
#define PCTE_REPLICATION_INCLUDED 1

#include "types.idl"
#include "references.idl"
```

17.1 Replication datatypes

```
/* None. */
```

17.2 Replication operations

```
interface Pcte_replica_set {

/* This interface is applied to the PCTE object type "replica_set". */

/* 17.2.1 REPLICA_SET_ADD_COPY_VOLUME */

Pcte_error_type add_copy_volume (
    in Pcte_object_reference   copy_volume
);
```

```
/* 17.2.2 REPLICA_SET_CREATE */

Pcte_error_type create (                                          //PIDL
    in Pcte_object_reference    master_volume,
    in Pcte_natural             identifier,
    out Pcte_object_reference   replica_set
);

/* 17.2.3 REPLICA_SET_REMOVE */

Pcte_error_type remove (
);

/* 17.2.4 REPLICA_SET_REMOVE_COPY_VOLUME */

Pcte_error_type remove_copy_volume (
    in Pcte_object_reference    copy_volume
);

};

interface Pcte_replicated_object {

/* This interface is applied to the PCTE object type "object". */

/* 17.2.5 REPLICATED_OBJECT_CREATE */

Pcte_error_type create (
    in Pcte_object_reference    replica_set
);

/* 17.2.6 REPLICATED_OBJECT_DELETE_REPLICA */

Pcte_error_type object_delete_replica (
    in Pcte_object_reference    copy_volume
);

/* 17.2.7 REPLICATED_OBJECT_DUPLICATE */

Pcte_error_type object_duplicate (
    in Pcte_object_reference    volume,
    in Pcte_object_reference    copy_volume
);

/* 17.2.8 REPLICATED_OBJECT_REMOVE */

Pcte_error_type object_remove (
);

/* 17.2.9 WORKSTATION_SELECT_REPLICA_SET_VOLUME */

/* See 18.5. */

/* 17.2.10 WORKSTATION_SELECT_REPLICA_SET_VOLUME */

/* See 18.5. */

};
```

```
#endif      // !PCTE_REPLICATION_INCLUDED
```

18 Network connection

```
/* The source file "network.idl" */

#ifndef PCTE_NETWORK_INCLUDED
#define PCTE_NETWORK_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "devices.idl"
```

18.1 Network connection datatypes

```
enum Pcte_work_status_item {
    PCTE_ACTIVITY_REMOTE_LOCKS, PCTE_ACTIVITY_LOCAL_LOCKS,
    PCTE_TRANSACTION_REMOTE_LOCKS, PCTE_TRANSACTION_LOCAL_LOCKS,
    PCTE_QUEUE_REMOTE, PCTE_QUEUE_LOCAL, PCTE_RECEIVE_REMOTE,
    PCTE_RECEIVE_LOCAL, PCTE_CHILD_REMOTE, PCTE_CHILD_LOCAL
};

typedef Pcte_natural Pcte_work_status;

enum Pcte_connection_status {
    PCTE_LOCAL, PCTE_CLIENT, PCTE_CONNECTED, PCTE_AVAILABLE
};

typedef Pcte_connection_status Pcte_requested_connection_status;

struct Pcte_new_administration_volume {
    Pcte_string             foreign_device;
    Pcte_volume_identifier  administration_volume;
    Pcte_string             volume_characteristics;
    Pcte_device_identifier  device;
    Pcte_string             device_characteristics;
};

struct Pcte_workstation_status {
    Pcte_connection_status  connection;
    Pcte_work_status        work;
};

#define PCTE_MAX_MACHINE_NAME_SIZE PCTE_MAX_NAME_SIZE

typedef Pcte_octet Pcte_machine_name [PCTE_MAX_MACHINE_NAME_SIZE + 1];

#define PCTE_MAX_NODE_NAME_SIZE PCTE_MAX_NAME_SIZE

typedef Pcte_octet Pcte_node_name [PCTE_MAX_NODE_NAME_SIZE + 1];
```

18.2 Network connection operations

```
interface Pcte_workstation {

/* This interface is applied to the PCTE object type "workstation". */

/*  When the parameter is absent in the abstract specification, the local workstation is assumed.*/

/* 18.2.1 WORKSTATION_CONNECT */

/* Applied to the local workstation. */

    Pcte_error_type connect (
    in Pcte_requested_connection_status   status
);

/*  If the value PCTE_AVAILABLE is passed to the parameter status the error          */
/*  PCTE_VALUE_OUT_OF_RANGE is raised.                                               */

/* 18.2.2 WORKSTATION_CREATE */

/* Applied to the local workstation. */

Pcte_error_type create (
    in Pcte_natural                    execution_site_identifier,
    in Pcte_new_administration_volume  administration_volume,
    in Pcte_atomic_access_rights       access_mask,
    in Pcte_node_name                  node_name,
    in Pcte_machine_name               machine_name
);

Pcte_error_type create_with_existing_admin_volume (
    in Pcte_natural                    execution_site_identifier,
    in Pcte_object_reference           existing_administration_volume,
    in Pcte_atomic_access_rights       access_mask,
    in Pcte_string                     node_name,
    in Pcte_string                     machine_name
);

/*  The effect of specifying administration_volume as a new administration volume to the  */
/*  abstract operation is achieved by the operation                                       */
/*  Pcte_workstation_create_with_existing_admin_volume.  The effect of specifying         */
/*  administration_volume as a volume designator to the abstract operation is achieved by */
/*  the operation Pcte_workstation_create.                                                */

/* 18.2.3 WORKSTATION_DELETE */

Pcte_error_type delete (
);

/* 18.2.4 WORKSTATION_DISCONNECT */

Pcte_error_type disconnect (
);
```

```
/* 18.2.5 WORKSTATION_GET_STATUS */

Pcte_error_type get_status (
    out Pcte_workstation_status     status
);

/*  The effect of not providing the optional parameter station to the abstract operation is   */
/*  achieved by specifying station as Pcte_null_object_reference.                             */

/* 18.2.6 WORKSTATION_REDUCE_CONNECTION */

Pcte_error_type reduce_connection (
    in Pcte_requested_connection_status   status,
    in Pcte_boolean                       force
);

/*  The effect of not providing the optional parameter station to the abstract operation is   */
/*  achieved by specifying station as Pcte_null_object_reference. If the value                */
/*  PCTE_AVAILABLE is passed to the parameter status the error                                */
/*  PCTE_VALUE_OUT_OF_RANGE is raised.                                                        */
```

18.3 Foreign system operations

```
/* 18.3.1 CONTENTS_COPY_FROM_FOREIGN_SYSTEM */

/* See 12.2. */

/* 18.3.2 CONTENTS_COPY_TO_FOREIGN_SYSTEM */

/* See 12.2.
```

18.4 Time operations

```
/* 18.4.1 TIME_GET */

Pcte_error_type time_get (
    out Pcte_time     time
);

/* 18.4.2 TIME_SET */

Pcte_error_type time_set (
    in Pcte_time   time
);
```

18.5 Other workstation operations

```
/* 17.2.9 WORKSTATION_SELECT_REPLICA_SET_VOLUME */

Pcte_error_type select_replica_set_volume (
    in Pcte_object_reference    replica_set,
    in Pcte_object_reference    volume
);

/* 17.2.10 WORKSTATION_UNSELECT_REPLICA_SET_VOLUME */

Pcte_error_type unselect_replica_set_volume (
    in Pcte_object_reference    replica_set
);

/* 11.2.5 DEVICE CREATE */

Pcte_error_type device_create (
    in Pcte_type_name               device_type,
    in Pcte_atomic_access_rights    access_mask,
    in Pcte_natural                 device_identifier,
    in Pcte_string                  device_characteristics,
    out Pcte_object_reference       new_device
);

Pcte_error_type h_device_create (
    in Pcte_type_reference          device_type,
    in Pcte_atomic_access_rights    access_mask,
    in Pcte_natural                 device_identifier,
    in Pcte_string                  device_characteristics,
    out Pcte_object_reference       new_device
);

};

#endif      // !PCTE_NETWORK_INCLUDED
```

19 Discretionary security

```
/* The source file "discretionary.idl" */

#ifndef PCTE_DISCRETIONARY_INCLUDED
#define PCTE_DISCRETIONARY_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "oms_types.idl"
#include "discretionary_types.idl"
```

19.1 Discretionary security datatypes

```
/* The source file "discretionary_types.idl" */

#ifndef PCTE_DISCRETIONARY_TYPES_INCLUDED
#define PCTE_DISCRETIONARY_TYPES_INCLUDED 1

#define PCTE_ALL_USERS (Pcte_natural)          1
#define PCTE_SECURITY (Pcte_natural)           2
#define PCTE_AUDIT (Pcte_natural)              3
#define PCTE_EXECUTION (Pcte_natural)          4
#define PCTE_REPLICATION (Pcte_natural)        5
#define PCTE_CONFIGURATION (Pcte_natural)      6
#define PCTE_HISTORY (Pcte_natural)            7
#define PCTE_SCHEMA_UPDATE (Pcte_natural)      8

enum Pcte_discretionary_access_mode {
    PCTE_NAVIGATE, PCTE_READ_ATTRIBUTES, PCTE_READ_LINKS,
    PCTE_READ_CONTENTS, PCTE_APPEND_LINKS, PCTE_APPEND_IMPLICIT,
    PCTE_APPEND_CONTENTS, PCTE_WRITE_IMPLICIT, PCTE_WRITE_ATTRIBUTES,
    PCTE_WRITE_LINKS, PCTE_WRITE_CONTENTS, PCTE_DELETE, PCTE_EXECUTE,
    PCTE_EXPLOIT_DEVICE, PCTE_EXPLOIT_SCHEMA,
    PCTE_EXPLOIT_CONSUMER_IDENTITY, PCTE_CONTROL_DISCRETIONARY,
    PCTE_CONTROL_MANDATORY, PCTE_CONTROL_OBJECT, PCTE_OWNER,
    PCTE_STABILIZE
};

typedef Pcte_natural Pcte_discretionary_access_modes;

struct Pcte_access_rights {
    Pcte_discretionary_access_modes  denied_rights;
    Pcte_discretionary_access_modes  granted_rights;
};

typedef Pcte_access_rights Pcte_atomic_access_rights;

typedef Pcte_access_rights Pcte_requested_access_rights;

typedef Pcte_natural Pcte_group_identifier;

struct Pcte_acl_entry {
    Pcte_group_identifier    group;
    Pcte_access_rights       access_rights;
};

typedef Pcte_sequence Pcte_acl;

#endif
```

19.2 Discretionary access control operations

```
interface Pcte_group {

/* This interface is applied to the PCTE object type "security_group". */
```

```
/* 19.2.1 GROUP_GET_IDENTIFIER */

Pcte_error_type get_identifier (
    in Pcte_object_reference     group,
    out Pcte_group_identifier    identifier
);

/* 19.2.2 OBJECT_CHECK_PERMISSION */

/* See 9.3. */

/* 19.2.3 OBJECT_GET_ACL */

/* See 9.3. */

/* 19.2.4 OBJECT_SET_ACL_ENTRY */

/* See 9.3. */
```

19.3 Discretionary security administration operations

```
/* 19.3.1 GROUP_INITIALIZE */

Pcte_error_type initialize (
    in Pcte_object_reference     group,
    in Pcte_group_identifier     identifier
);

/* 19.3.2 GROUP_REMOVE */

Pcte_error_type remove (
    in Pcte_object_reference     group
);

/* 19.3.3 GROUP_RESTORE */

Pcte_error_type restore (
    in Pcte_object_reference     group,
    in Pcte_group_identifier     identifier
);

/* 20.3.2 GROUP_DISABLE_FOR_CONFIDENTIALITY_DOWNGRADE */

Pcte_error_type disable_for_confidentiality_downgrade (
    in Pcte_object_reference     confidentiality_class
);

/* 20.3.3 GROUP_DISABLE_FOR_INTEGRITY_UPGRADE */

Pcte_error_type disable_for_integrity_upgrade (
    in Pcte_object_reference     integrity_class
);
```

```
/* 20.3.4 GROUP_ENABLE_FOR_CONFIDENTIALITY_DOWNGRADE */

Pcte_error_type enable_for_confidentiality_downgrade (
    in Pcte_object_reference    confidentiality_class
);

/* 20.3.5 GROUP_ENABLE_FOR_INTEGRITY_UPGRADE */

Pcte_error_type enable_for_integrity_upgrade (
    in Pcte_object_reference    integrity_class
);

};

interface Pcte_program_group {

/* This interface is applied to the PCTE object type "program_group". */

/* 19.3.4 PROGRAM_GROUP_ADD_MEMBER */

Pcte_error_type add_member (
    in Pcte_object_reference    group,
    in Pcte_object_reference    program
);

/* 19.3.5 PROGRAM_GROUP_ADD_SUBGROUP */

Pcte_error_type add_subgroup (
    in Pcte_object_reference    group,
    in Pcte_object_reference    subgroup
);

/* 19.3.6 PROGRAM_GROUP_REMOVE_MEMBER */

Pcte_error_type remove_member (
    in Pcte_object_reference    group,
    in Pcte_object_reference    program
);

/* 19.3.7 PROGRAM_GROUP_REMOVE_SUBGROUP */

Pcte_error_type remove_subgroup (
    in Pcte_object_reference    group,
    in Pcte_object_reference    subgroup
);

};

interface Pcte_user_group {

/* This interface is applied to the PCTE object type "user_group". */
```

```
/* 19.3.8 USER_GROUP_ADD_MEMBER */

Pcte_error_type add_member (
    in Pcte_object_reference    group,
    in Pcte_object_reference    user
);

/* 19.3.9 USER_GROUP_ADD_SUBGROUP */

Pcte_error_type add_subgroup (
    in Pcte_object_reference    group,
    in Pcte_object_reference    subgroup
);

/* 19.3.10 USER_GROUP_REMOVE_MEMBER */

Pcte_error_type remove_member (
    in Pcte_object_reference    group,
    in Pcte_object_reference    user
);

/* 19.3.11 USER_GROUP_REMOVE_SUBGROUP */

Pcte_error_type remove_subgroup (
    in Pcte_object_reference    group,
    in Pcte_object_reference    subgroup
);

};

#endif      // !PCTE_DISCRETIONARY_INCLUDED
```

20 Mandatory security

```
/* The source file "mandatory.idl" */

#ifndef PCTE_MANDATORY_INCLUDED
#define PCTE_MANDATORY_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "mandatory_types.idl"
```

20.1 Mandatory_security datatypes

```
/* The source file "pcte_mandatory_types" */

#ifndef PCTE_MANDATORY_TYPES_INCLUDED
#define PCTE_MANDATORY_TYPES_INCLUDED 1

typedef Pcte_string Pcte_security_label;

/*  The PCTE datatype Pcte_security_label_string is mapped to the IDL datatype  */
/*  Pcte_security_label.                                                         */
```

```
enum Pcte_floating_level {
    PCTE_NO_FLOAT, PCTE_FLOAT_IN,
    PCTE_FLOAT_OUT, PCTE_FLOAT_IN_OUT
};

#endif
```

20.2 Operations for mandatory security operation

```
/* 20.2.1 DEVICE_SET_CONFIDENTIALITY_RANGE */

/* See 11.2. */

/* 20.2.2 DEVICE_SET_INTEGRITY_RANGE */

/* See 11.2. */

interface Pcte_execution_site {

/* This interface is applied to the PCTE object type "execution_site" \f B. */

/* 20.2.3 EXECUTION_SITE_SET_CONFIDENTIALITY_RANGE */

Pcte_error_type set_confidentiality_range (
    in Pcte_security_label    high_label,
    in Pcte_security_label    low_label
);

/* 20.2.4 EXECUTION_SITE_SET_INTEGRITY_RANGE */

Pcte_error_type set_integrity_range (
    in Pcte_security_label    high_label,
    in Pcte_security_label    low_label
);

};

/* 20.2.5 OBJECT_SET_CONFIDENTIALITY_LABEL */

/* See 9.3. */

/* 20.2.6 OBJECT_SET_INTEGRITY_LABEL */

/* See 9.3. */

/* 20.2.7 VOLUME_SET_CONFIDENTIALITY_RANGE */

/* See 11.2. */

/* 20.2.8 VOLUME_SET_INTEGRITY_RANGE */

/* See 11.2. */
```

20.3 Mandatory security administration operations

```
interface Pcte_confidentiality_class {
```

```
/* This interface is applied to the PCTE object type "confidentiality_class". */

/* 20.3.1 CONFIDENTIALITY_CLASS_INITIALIZE */

Pcte_error_type initialize (
    in Pcte_name                class_name,
    in Pcte_object_reference    to_be_dominated
);

};

/* 20.3.2 GROUP_DISABLE_FOR_CONFIDENTIALITY_DOWNGRADE */

/* See 19.3. */

/* 20.3.3 GROUP_DISABLE_FOR_INTEGRITY_UPGRADE */

/* See 19.3. */

/* 20.3.4 GROUP_ENABLE_FOR_CONFIDENTIALITY_DOWNGRADE */

/* See 19.3. */

/* 20.3.5 GROUP_ENABLE_FOR_INTEGRITY_UPGRADE */

    /* See 19.3. */

interface Pcte_integrity_class {

/* This interface is applied to the PCTE object type "integrity_class". */

/* 20.3.6 INTEGRITY_CLASS_INITIALIZE */

Pcte_error_type initialize (
    in Pcte_name                class_name,
    in Pcte_object_reference    to_be_dominated
);

};

interface Pcte_user {

/* This interface is applied to the PCTE object type "user". */

/* 20.3.7 USER_EXTEND_CONFIDENTIALITY_CLEARANCE */

Pcte_error_type extend_confidentiality_clearance (
    in Pcte_object_reference    confidentiality_class
);

/* 20.3.8 USER_EXTEND_INTEGRITY_CLEARANCE */

Pcte_error_type extend_integrity_clearance (
    in Pcte_object_reference    integrity_class
);
```

```
/* 20.3.9 USER_REDUCE_CONFIDENTIALITY_CLEARANCE */

Pcte_error_type reduce_confidentiality_clearance (
    in Pcte_object_reference      confidentiality_class
);

/* 20.3.9 USER_REDUCE_CONFIDENTIALITY_CLEARANCE */

Pcte_error_type reduce_integrity_clearance (
    in Pcte_object_reference      integrity_class
);

};

#endif
```

21 Auditing

```
/* The source file "auditing.idl" */

#ifndef PCTE_AUDITING_INCLUDED
#define PCTE_AUDITING_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "oms_types.idl"
#include "discretionary_types.idl"
#include "mandatory_types.idl"
```

21.1 Auditing datatypes

```
enum Pcte_selectable_event_type {
    PCTE_WRITE, PCTE_READ, PCTE_COPY, PCTE_ACCESS_CONTENTS,
    PCTE_EXPLOIT, PCTE_CHANGE_ACCESS_CONTROL_LIST,
    PCTE_CHANGE_LABEL, PCTE_USE_PREDEFINED_GROUP,
    PCTE_SET_USER_IDENTITY,
    PCTE_WRITE_CONFIDENTIALITY_VIOLATION,
    PCTE_READ_CONFIDENTIALITY_VIOLATION,
    PCTE_WRITE_INTEGRITY_VIOLATION,
    PCTE_READ_INTEGRITY_VIOLATION,
    PCTE_COVERT_CHANNEL, PCTE_INFORMATION_EVENT
};

enum Pcte_mandatory_event_type {
    PCTE_CHANGE_IDENTIFICATION, PCTE_SELECT_AUDIT_EVENT,
    PCTE_SECURITY_ADMINISTRATION
};
```

```
union Pcte_event_type_event_type switch (long) {
    case 1:     Pcte_selectable_event_type      selectable_event_type;
    case 2:     Pcte_mandatory_event_type       mandatory_event_type;
};

enum Pcte_event_kind {
    PCTE_SELECTABLE, PCTE_MANDATORY
};

struct Pcte_event_type {
    Pcte_event_kind                 kind;
    Pcte_event_type_event_type      type;
};

/*  Pcte_event_type corresponds to the PCTE datatypes Selectable_event_type and     */
/*  Mandatory_event_type.                                                           */

enum Pcte_selected_return_code {
    PCTE_FAILURE, PCTE_SUCCESS, PCTE_ANY_CODE
};

typedef Pcte_selected_return_code Pcte_return_code;

struct Pcte_object_auditing_record {
    Pcte_group_identifier   user;
    Pcte_time               time;
    Pcte_exact_identifier   workstation;
    Pcte_event_type         type;
    Pcte_return_code        return_code;
    Pcte_exact_identifier   process;
    Pcte_exact_identifier   objectaud;
};

struct Pcte_exploit_auditing_record     {
    Pcte_group_identifier   user;
    Pcte_time               time;
    Pcte_exact_identifier   workstation;
    Pcte_event_type         type;
    Pcte_return_code        return_code;
    Pcte_exact_identifier   process;
    Pcte_exact_identifier   new_process;
    Pcte_exact_identifier   exploited_object;
};
```

```
struct Pcte_information_auditing_record {
    Pcte_group_identifier    user;
    Pcte_time                time;
    Pcte_exact_identifier    workstation;
    Pcte_event_type          type;
    Pcte_return_code         return_code;
    Pcte_exact_identifier    process;
    Pcte_string              text;
};

struct Pcte_copy_auditing_record {
    Pcte_group_identifier    user;
    Pcte_time                time;
    Pcte_exact_identifier    workstation;
    Pcte_event_type          type;
    Pcte_return_code         return_code;
    Pcte_exact_identifier    process;
    Pcte_exact_identifier    source;
    Pcte_exact_identifier    destination;
};

struct Pcte_security_auditing_record {
    Pcte_group_identifier    user;
    Pcte_time                time;
    Pcte_exact_identifier    workstation;
    Pcte_event_type          type;
    Pcte_return_code         return_code;
    Pcte_exact_identifier    process;
    Pcte_exact_identifier    group;
};

union Pcte_auditing_record_record switch (long) {
    case 1: Pcte_object_auditing_record          objectaud;
    case 2: Pcte_exploit_auditing_record         exploit;
    case 3: Pcte_information_auditing_record     user_defined;
    case 4: Pcte_copy_auditing_record            copy;
    case 5: Pcte_security_auditing_record        security;
};

enum Pcte_auditing_record_type {
    PCTE_OBJECT_RECORD, PCTE_EXPLOIT_RECORD,
    PCTE_INFORMATION_RECORD, PCTE_COPY_RECORD,
    PCTE_SECURITY_RECORD
};

struct Pcte_auditing_record {
    Pcte_auditing_record_type      type;
    Pcte_auditing_record_record    record;
};
```

```
enum Pcte_audit_status {
    PCTE_ENABLED, PCTE_DISABLED
};
struct Pcte_general_criterion {
    Pcte_selectable_event_type      selectable_event_type;
    Pcte_selected_return_code       return_code;
};
struct Pcte_user_criterion {
    Pcte_selectable_event_type      selectable_event_type;
    Pcte_group_identifier           user;
};
struct Pcte_confidentiality_criterion {
    Pcte_selectable_event_type      selectable_event_type;
    Pcte_security_label             security_label;
} ;
typedef Pcte_confidentiality_criterion Pcte_integrity_criterion;
struct Pcte_object_criterion {
    Pcte_selectable_event_type      selectable_event_type;
    Pcte_object_reference           objectaud;
};
enum Pcte_criterion_type {
    PCTE_GENERAL, PCTE_USER_DEPENDENT,
    PCTE_CONFIDENTIALITY_DEPENDENT,
    PCTE_INTEGRITY_DEPENDENT, PCTE_OBJECT_DEPENDENT
};
union Pcte_selection_criterion_criterion switch (long) {
    case 1: Pcte_general_criterion          general;
    case 2: Pcte_user_criterion             user;
    case 3: Pcte_confidentiality_criterion  confidentiality;
    case 4: Pcte_integrity_criterion        integrity;
    case 5: Pcte_object_criterion           objectaud;
};
struct Pcte_selection_criterion {
    Pcte_criterion_type                 type;
    Pcte_selection_criterion_criterion  criterion;
};
typedef Pcte_selection_criterion Pcte_specific_criterion;
union Pcte_criteria_criteria switch (long) {
    case 1: Pcte_general_criteria           general;
    case 2: Pcte_user_criteria              user;
    case 3: Pcte_confidentiality_criteria   confidentiality;
    case 4: Pcte_integrity_criteria         integrity;
    case 5: Pcte_object_criteria            objectaud;
};
```

```
struct Pcte_criteria {
    Pcte_criterion_type      type;
    Pcte_criteria_criteria   criteria;
};
```

21.2 Auditing operations

```
interface Pcte_audit {

/* This interface is applied to the PCTE object type "workstation". */

/* All the operations are applied to the local station. */

/* 21.2.1 AUDIT_ADD_CRITERION */

Pcte_error_type add_criterion (
    in Pcte_selection_criterion   criterion
);

/* 21.2.2 AUDIT_FILE_COPY_AND_RESET */

Pcte_error_type file_copy_and_reset (
    in Pcte_object_reference     source,
    in Pcte_object_reference     destination
);

/* 21.2.3 AUDIT_FILE_READ */

Pcte_error_type file_read (
    in Pcte_object_reference     audit_file,
    out Pcte_audit_file          records
);

/* 21.2.4 AUDIT_GET_CRITERIA */

Pcte_error_type get_criteria (
    in Pcte_criterion_type    criterion_type,
    out Pcte_criteria         criteria
);

/* 21.2.5 AUDIT_RECORD_WRITE */

Pcte_error_type record_write (
    in Pcte_string      text
);

/* 21.2.6 AUDIT_REMOVE_CRITERION */

Pcte_error_type remove_criterion (
    in Pcte_specific_criterion    criterion
);

/*  If a value of type Pcte_general_criterion is passed to criterion then the error   */
/*  PCTE_VALUE_OUT_OF_RANGE is raised.                                                 */
```

```
Pcte_error_type remove_criterion_of_event_type (
    in Pcte_selectable_event_type   criterion
);

/*  The effect specifying criterion as a specific criterion to the abstract operation is achieved  */
/*  by the operation Pcte_audit_remove_criterion.  The effect specifying criterion as a            */
/*  selectable event type to the abstract operation is achieved by the operation                   */
/*  Pcte_audit_remove_criterion_of_event_type.                                                     */

/* 21.2.7 AUDIT_SELECTION_CLEAR */

Pcte_error_type selection_clear (
);

/* 21.2.8 AUDIT_SWITCH_OFF_SELECTION */

Pcte_error_type switch_off_selection (
);

/* 21.2.9 AUDIT_SWITCH_ON_SELECTION */

Pcte_error_type switch_on_selection (
);

/* 21.2.10 AUDITING_GET_STATUS */

Pcte_error_type get_status (
    out Pcte_audit_status     status
);

};

#endif
```

22 Accounting

```
/* The source file "accounting.idl" */

#ifndef PCTE_ACCOUNTING_INCLUDED
#define PCTE_ACCOUNTING_INCLUDED 1

#include "types.idl"
#include "references.idl"
#include "sequences.idl"
#include "discretionary_types.idl"
#include "oms_types.idl"
```

22.1 Accounting datatypes

```
typedef Pcte_natural Pcte_consumer_identifier;

typedef Pcte_natural Pcte_resource_identifier;
```

```
enum Pcte_resource_kind {
    PCTE_WORKSTATION, PCTE_FILE, PCTE_PIPE, PCTE_DEVICE,
    PCTE_STATIC_CONTEXT, PCTE_SDS, PCTE_MESSAGE_QUEUE,
    PCTE_INFORMATION
};

struct Pcte_workstation_accounting_record {
    Pcte_group_identifier    security_user;
    Pcte_group_identifier    adopted_user_group;
    Pcte_exact_identifier    consumer_group;
    Pcte_exact_identifier    resource_group;
    Pcte_resource_kind       resource_kind;
    Pcte_time                start_time;
    Pcte_float               duration;
    Pcte_float               cpu_time;
    Pcte_float               sys_time;
};

typedef Pcte_workstation_accounting_record Pcte_static_context_accounting_record;

struct Pcte_sds_accounting_record {
    Pcte_group_identifier    security_user;
    Pcte_group_identifier    adopted_user_group;
    Pcte_exact_identifier    consumer_group;
    Pcte_exact_identifier    resource_group;
    Pcte_resource_kind       resource_kind;
    Pcte_time                start_time;
};

struct Pcte_device_accounting_record {
    Pcte_group_identifier    security_user;
    Pcte_group_identifier    adopted_user_group;
    Pcte_exact_identifier    consumer_group;
    Pcte_exact_identifier    resource_group;
    Pcte_resource_kind       resource_kind;
    Pcte_time                start_time;
    Pcte_float               duration;
    Pcte_natural             read_count;
    Pcte_natural             write_count;
    Pcte_natural             read_size;
    Pcte_natural             write_size;
};

typedef Pcte_device_accounting_record Pcte_file_accounting_record;

typedef Pcte_device_accounting_record Pcte_pipe_accounting_record;

enum Pcte_operation_kind {
    PCTE_SEND, PCTE_RECEIVE, PCTE_RESERVE
};
```

```
struct Pcte_message_queue_accounting_record {
    Pcte_group_identifier     security_user;
    Pcte_group_identifier     adopted_user_group;
    Pcte_exact_identifier     consumer_group;
    Pcte_exact_identifier     resource_group;
    Pcte_resource_kind        resource_kind;
    Pcte_time                 start_time;
    Pcte_operation_kind       operation;
    Pcte_natural              message_size;
};

struct Pcte_information_accounting_record {
    Pcte_group_identifier     security_user;
    Pcte_group_identifier     adopted_user_group;
    Pcte_exact_identifier     consumer_group;
    Pcte_exact_identifier     resource_group;
    Pcte_resource_kind        resource_kind;
    Pcte_time                 start_time;
    Pcte_string               information;
};

union Pcte_resource switch (long) {
    case 1:   Pcte_workstation_accounting_record         workstation;
    case 2:   Pcte_static_context_accounting_record      static_context;
    case 3:   Pcte_sds_accounting_record                 sds;
    case 4:   Pcte_device_accounting_record              device;
    case 5:   Pcte_file_accounting_record                file;
    case 6:   Pcte_pipe_accounting_record                pipe;
    case 7:   Pcte_message_queue_accounting_record       message_queue;
    case 8:   Pcte_information_accounting_record         information;
};

struct Pcte_accounting_record {
    Pcte_resource_kind  resource_kind;
    Pcte_resource       resource;
};
```

22.2 Accounting administration operations

```
interface Pcte_accounting {

/* This interface is applied to the PCTE object type "accounting_log". */

/* 22.2.1 ACCOUNTING_LOG_COPY_AND_RESET */

Pcte_error_type log_copy_and_reset (
    in Pcte_object_reference     destination_log
);
```

```
/* 22.2.2 ACCOUNTING_LOG_READ */

Pcte_error_type log_read (
    out Pcte_accounting_file     records
);

/* 22.2.3 ACCOUNTING_OFF */

Pcte_error_type off (                                                    //PIDL
    in Pcte_object_reference     station
);

/* 22.2.4 ACCOUNTING_ON */

Pcte_error_type on (
    in Pcte_object_reference     station
);

/* 22.2.5 ACCOUNTING_RECORD_WRITE */

Pcte_error_type record_write (
    in Pcte_string     information
);

};

interface Pcte_consumer_group {

/* This interface is applied to the PCTE object type "consumer_group". */

/* 22.2.6 CONSUMER_GROUP_INITIALIZE */

Pcte_error_type initialize (
    out Pcte_consumer_identifier    identifier
);

/* 22.2.7 CONSUMER_GROUP_REMOVE */

Pcte_error_type remove (
);

};

interface Pcte_resource_group {

/* This interface is applied to the PCTE object type "resource_group". */

/* 22.2.8 RESOURCE_GROUP_ADD_OBJECT */

Pcte_error_type add_object (
    in Pcte_object_reference     added_object
);

/* 22.2.9 RESOURCE_GROUP_INITIALIZE */

Pcte_error_type initialize (
    out Pcte_resource_identifier     identifier
);
```

```
/* 22.2.10 RESOURCE_GROUP_REMOVE */

Pcte_error_type remove (
);

/* 22.2.11 RESOURCE_GROUP_REMOVE_OBJECT */

Pcte_error_type remove_object (
    in Pcte_object_reference     removed_object
);

};

#endif      // !PCTE_ACCOUNTING_INCLUDED

/* 22.3.1 PROCESS_SET_CONSUMER_IDENTITY */

/* See 13.7. */

/* 22.3.2 PROCESS_UNSET_CONSUMER_IDENTITY */

/* See 13.7. */
```

23 References

```
/* The source file "references.idl" */

#ifndef PCTE_REFERENCES_INCLUDED
#define PCTE_REFERENCES_INCLUDED 1

#include "types.idl"
```

23.1 Reference datatypes

```
#define Pcte_null_object_reference (Object) 0

enum Pcte_evaluation_point {
    PCTE_NOW, PCTE_FIRST_USE, PCTE_EVERY_USE
};

enum Pcte_evaluation_status {
    PCTE_INTERNAL, PCTE_EXTERNAL
};

enum Pcte_reference_equality {
    PCTE_EQUAL_REF, PCTE_UNEQUAL_REF, PCTE_EXTERNAL_REF
};

define PCTE_MAX_NAME_SIZE <implementation-defined>;

typedef string <PCTE_MAX_NAME_SIZE + 1> Pcte_name;

define PCTE_MAX_TYPE_NAME_SIZE <implementation-defined>;

typedef string <PCTE_MAX_TYPE_NAME_SIZE + 1> Pcte_type_name;

typedef Pcte_type_name Pcte_attribute_name;
```

```
typedef Pcte_type_name Pcte_type_name_in_sds;
#define PCTE_MAX_KEY_SIZE <implementation-defined>;
typedef string <PCTE_MAX_KEY_SIZE + 1> Pcte_key;
#define PCTE_MAX_LINK_NAME_SIZE <implementation-defined>;
typedef string <PCTE_MAX_LINK_NAME_SIZE + 1> Pcte_link_name;
typedef string Pcte_pathname;
typedef string Pcte_relative_pathname;
struct Pcte_key_value {
    enum enum_type {
    PCTE_NATURAL_KEY, PCTE_STRING_KEY
    } type;
    Pcte_natural  v_natural;
    Pcte_key      v_string;
};
interface Pcte_object_reference;
interface Pcte_link_reference;
interface Pcte_type_reference;
typedef Pcte_type_reference Pcte_attribute_reference;
```

23.2 Reference creation and discarding

```
interface Pcte_RF {
/* This interface is applied to the PCTE object type "process". */
Pcte_error_type discard (
    in Pcte_pathname    pathname
);
/* 23.2.5 OBJECT_REFERENCE_SET_ABSOLUTE */
Pcte_error_type set_absolute (
    in Pcte_pathname            pathname,
    in Pcte_evaluation_point    point,
    out Pcte_object_reference   new_reference
);
/* 23.3.8 LINK_REFERENCE_SET */
Pcte_error_type set_from_name (
    in Pcte_link_name           link_name,
    in Pcte_evaluation_point    point,
    out Pcte_link_reference     new_link_reference
);
```

```
/* 23.4.6 TYPE_REFERENCE_SET */

Pcte_error_type set (
    in Pcte_type_name           type_name,
    in Pcte_evaluation_point    point,
    out Pcte_type_reference     new_type_reference
);

};
```

23.3 Object reference operations

```
interface Pcte_object_reference {

/* This interface is applied to the PCTE object type "object". */

/* 23.2.1 OBJECT_REFERENCE_COPY */

Pcte_error_type copy (
    in  Pcte_evaluation_point       point,
    out Pcte_object_reference       new_reference
);

/* 23.2.2 OBJECT_REFERENCE_GET_EVALUATION_POINT */

Pcte_error_type get_evaluation_point (
    out Pcte_evaluation_point   point
);

/* 23.2.3 OBJECT_REFERENCE_GET_PATH */

Pcte_error_type get_path (
    out Pcte_pathname       pathname
);

/* 23.2.4 OBJECT_REFERENCE_GET_STATUS */

Pcte_error_type get_status (
    out Pcte_evaluation_status  status
);

/* 23.2.6 OBJECT_REFERENCE_SET_RELATIVE */

Pcte_error_type set_relative (
    in Pcte_relative_pathname   pathname,
    in Pcte_evaluation_point    point,
    out Pcte_object_reference   new_reference
);

/* 23.2.7 OBJECT_REFERENCE_UNSET */

Pcte_error_type unset (
);
```

```
/* 23.2.8 OBJECT_REFERENCES_ARE_EQUAL */

Pcte_error_type are_equal (
    in Pcte_object_reference        compare_reference,
    out Pcte_reference_equality     equal
);

};
```

23.4 Link reference operations

```
interface Pcte_link_reference {

/* This interface is applied to the PCTE object type "object". */

/* 23.3.1 LINK_REFERENCE_COPY */

Pcte_error_type copy (
    in Pcte_evaluation_point    point,
    out Pcte_link_reference     new_link_reference
);

/* 23.3.2 LINK_REFERENCE_GET_EVALUATION_POINT */

Pcte_error_type get_evaluation_point (
    out Pcte_evaluation_point   point
);

/* 23.3.3 LINK_REFERENCE_GET_KEY */

Pcte_error_type get_key (
    out Pcte_key      key
);

/* 23.3.4 LINK_REFERENCE_GET_KEY_VALUE */

Pcte_error_type get_key_value (
    in Pcte_natural           index,
    out Pcte_key_value        key_value
);

/* 23.3.5 LINK_REFERENCE_GET_NAME */

Pcte_error_type get_name (
    out Pcte_link_name        link_name
);

/* 23.3.6 LINK_REFERENCE_GET_STATUS */

Pcte_error_type get_status (
    out Pcte_evaluation_status   status
);
```

```
/* 23.3.7 LINK_REFERENCE_GET_TYPE */

Pcte_error_type get_type (
    out Pcte_type_reference      type_reference
);

/* 23.3.8 LINK_REFERENCE_SET */

/* See 23.2. */

/* 23.3.9 LINK_REFERENCE_UNSET */

Pcte_error_type unset (
);

/* 23.3.10 LINK_REFERENCES_ARE_EQUAL */

Pcte_error_type are_equal (
    in Pcte_link_reference          second_link_reference,
    out Pcte_reference_equality     equal
);

};
```

23.5 Type reference operations

```
interface Pcte_type_reference {

/* This interface is applied to the PCTE object type "type". */

Pcte_error_type set_link (
    in Pcte_evaluation_point    point,
    out Pcte_link_reference     new_link_reference
);

Pcte_error_type set_link_from_key (
    in Pcte_key                 key,
    in Pcte_evaluation_point    point,
    out Pcte_link_reference     new_link_reference
);

/* 23.4.1 TYPE_REFERENCE_COPY */

Pcte_error_type copy (
    in Pcte_evaluation_point    point,
    out Pcte_type_reference     new_type_reference
);

/* 23.4.2 TYPE_REFERENCE_GET_EVALUATION_POINT */

Pcte_error_type get_evaluation_point (
    out Pcte_evaluation_point   point
);
```

```
/* 23.4.3 TYPE_REFERENCE_GET_IDENTIFIER */

Pcte_error_type get_identifier (
    out Pcte_type_name      type_identifier
);

/* 23.4.4 TYPE_REFERENCE_GET_NAME */

Pcte_error_type get_name (
    out Pcte_type_name      type_name
);

/* 23.4.5 TYPE_REFERENCE_GET_STATUS */

Pcte_error_type get_status (
    out Pcte_evaluation_status   status
);

/* 23.4.6 TYPE_REFERENCE_SET */

/* See 23.2. */

/* 23.4.7 TYPE_REFERENCE_UNSET */

Pcte_error_type unset (
);

/* 23.4.8 TYPE_REFERENCES_ARE_EQUAL */

Pcte_error_type are_equal (
    in Pcte_type_reference          second_type_reference,
    out Pcte_reference_equality     equal
);

};

    #endif
```

24 Implementation limits

```
/* The source file "limits.idl" */

#ifndef PCTE_LIMITS_INCLUDED
#define PCTE_LIMITS_INCLUDED 1

#include "types.idl"
```

24.1 Implementation limit datatypes

```
/*  The implementation limits MAX_NAME_SIZE, MAX_KEY_SIZE, and                */
/*  MAX_LINK_REFERENCE_SIZE (represented by PCTE_MAX_LINK_NAME_SIZE),         */
/*  which define the maximum size of the corresponding texts Pcte_name, Pcte_key, and */
/*  Pcte_link_name, are defined in 23.1.  All other implementation limits are defined in this */
/*  clause.                                                                   */
```

```
enum Pcte_limit_category {
    PCTE_STANDARD, PCTE_IMPLEMENTATION, PCTE_REMAINING
};

/*  An implementation of this binding must return three sets of those implementation limits   */
/*  which are defined in this clause:                                                         */
/*    STANDARD: The value specified in ISO/IEC 13719-1                                        */
/*    IMPLEMENTATION: The value supported by the implementation                               */
/*    REMAINING: Where appropriate, the value remaining at the current time (after the        */
/*    usage of some resources).                                                               */

enum Pcte_limit_name {
    PCTE_DELTA_ACCOUNT_DURATION,
    PCTE_MAX_ACCESS_CONTROL_LIST_LENGTH,
    PCTE_MAX_ACCOUNT_DURATION,
    PCTE_MAX_ACCOUNT_INFORMATION_LENGTH,
    PCTE_MAX_ACTIVITIES,
    PCTE_MAX_ACTIVITIES_PER_PROCESS,
    PCTE_MAX_AUDIT_INFORMATION_LENGTH,
    PCTE_MAX_DIGIT_FLOAT_ATTRIBUTE,
    PCTE_MAX_FILE_SIZE,
    PCTE_MAX_FLOAT_ATTRIBUTE, PCTE_MIN_FLOAT_ATTRIBUTE,
    PCTE_MAX_INTEGER_ATTRIBUTE, PCTE_MIN_INTEGER_ATTRIBUTE,
    PCTE_MAX_KEY_VALUE,
    PCTE_MAX_MESSAGE_QUEUE_SPACE,
    PCTE_MAX_MESSAGE_SIZE,
    PCTE_MAX_MOUNTED_VOLUMES,
    PCTE_MAX_OPEN_OBJECTS,
    PCTE_MAX_OPEN_OBJECTS_PER_PROCESS,
    PCTE_MAX_PIPE_SIZE,
    PCTE_MAX_PRIORITY_VALUE,
    PCTE_MAX_PROCESSES,
    PCTE_MAX_PROCESSES_PER_USER,
    PCTE_MAX_SDS_IN_WORKING_SCHEMA,
    PCTE_MAX_SECURITY_GROUPS,
    PCTE_MAX_STRING_ATTRIBUTE_SIZE,
    PCTE_MAX_TIME_ATTRIBUTE, PCTE_MIN_TIME_ATTRIBUTE,
    PCTE_SMALLEST_FLOAT_ATTRIBUTE
};

union Pcte_limit_value_value switch (long) {
    case 1: Pcte_float    v_float;
    case 2: Pcte_integer  v_integer;
    case 3: Pcte_natural  v_natural;
    case 4: Pcte_time     v_time;
};
```

```
enum Pcte_limit_value_type {
    PCTE_FLOAT_LIMIT, PCTE_INTEGER_LIMIT,
    PCTE_NATURAL_LIMIT, PCTE_TIME_LIMIT
};

struct Pcte_limit_value {
    Pcte_limit_value_type   type;
    Pcte_limit_value_value  value;
};
```

24.2 Implementation limit operations

```
interface Pcte_limit {

/* This interface is by convention applied to the PCTE object type "process". */

/* 24.2.1 LIMIT_GET_VALUE */

Pcte_error_type get_value (
    in Pcte_limit_category   category,
    in Pcte_limit_name       name,
    out Pcte_limit_value     value,
    out Pcte_boolean         unlimited
);

/*  If there is no limit value, PCTE_TRUE is returned in unlimited.  Otherwise unlimited is    */
/*  set to PCTE_FALSE and the limit value is returned into the value pointed to by value.     */

};

#endif      // !PCTE_LIMITS_INCLUDED
```

25 Error conditions

```
/* The source file "pcte_errors.idl" */

#ifndef PCTE_ERRORS_INCLUDED
#define PCTE_ERRORS_INCLUDED 1
```

25.1 Error condition datatypes

```
enum Pcte_error_type {

    PCTE_NO_ERROR,

    /* Errors defined in ISO/IEC 13719-1, annex C */

    PCTE_ACCESS_CONTROL_WOULD_NOT_BE_GRANTED,
    PCTE_ACCESS_MODE_IS_INCOMPATIBLE,
    PCTE_ACCESS_MODE_IS_NOT_ALLOWED,
    PCTE_ACCOUNTING_LOG_IS_NOT_ACTIVE,
    PCTE_ACTIVITY_IS_OPERATING_ON_A_RESOURCE,
```

PCTE_ACTIVITY_STATUS_IS_INVALID,
PCTE_ACTIVITY_WAS_NOT_STARTED_BY_CALLING_PROCESS,
PCTE_ARCHIVE_EXISTS,
PCTE_ARCHIVE_HAS_ARCHIVED_OBJECTS,
PCTE_ARCHIVE_IS_INVALID_ON_DEVICE,
PCTE_ARCHIVE_IS_UNKNOWN,
PCTE_ATOMIC_ACL_IS_INCOMPATIBLE_WITH_OWNER_CHANGE,
PCTE_ATTRIBUTE_TYPE_IS_NOT_VISIBLE,
PCTE_ATTRIBUTE_TYPE_OF_LINK_TYPE_IS_NOT_APPLIED,
PCTE_ATTRIBUTE_TYPE_OF_OBJECT_TYPE_IS_NOT_APPLIED,
PCTE_AUDIT_FILE_IS_NOT_ACTIVE,
PCTE_BREAKPOINT_IS_NOT_DEFINED,
PCTE_CARDINALITY_IS_INVALID,
PCTE_CATEGORY_IS_BAD,
PCTE_CLASS_NAME_IS_INVALID,
PCTE_CONFIDENTIALITY_CONFINEMENT_WOULD_BE_VIOLATED,
PCTE_CONFIDENTIALITY_CRITERION_IS_NOT_SELECTED,
PCTE_CONFIDENTIALITY_LABEL_IS_INVALID,
PCTE_CONFIDENTIALITY_WOULD_BE_VIOLATED,
PCTE_CONNECTION_IS_DENIED,
PCTE_CONSUMER_GROUP_IS_IN_USE,
PCTE_CONSUMER_GROUP_IS_KNOWN,
PCTE_CONSUMER_GROUP_IS_UNKNOWN,
PCTE_CONTENTS_IS_NOT_EMPTY,
PCTE_CONTENTS_IS_NOT_FILE_CONTENTS,
PCTE_CONTENTS_IS_NOT_OPEN,
PCTE_CONTENTS_OPERATION_IS_INVALID,
PCTE_CONTROL_WOULD_NOT_BE_GRANTED,
PCTE_DATA_ARE_NOT_AVAILABLE,
PCTE_DEFAULT_ACL_WOULD_BE_INCONSISTENT_WITH_DEFAULT_OBJECT_OWNER,
PCTE_DEFAULT_ACL_WOULD_BE_INVALID,
PCTE_DEFINITION_MODE_VALUE_WOULD_BE_INVALID,
PCTE_DESTINATION_OBJECT_TYPE_IS_INVALID,
PCTE_DEVICE_CHARACTERISTICS_ARE_INVALID,
PCTE_DEVICE_CONTROL_OPERATION_IS_INVALID,
PCTE_DEVICE_EXISTS,
PCTE_DEVICE_IS_BUSY,
PCTE_DEVICE_IS_IN_USE,
PCTE_DEVICE_IS_UNKNOWN,
PCTE_DEVICE_LIMIT_WOULD_BE_EXCEEDED,
PCTE_DEVICE_SPACE_IS_FULL,
PCTE_DISCRETIONARY_ACCESS_IS_NOT_GRANTED,
PCTE_ENUMERAL_TYPE_IS_INVALID,
PCTE_ENUMERAL_TYPE_IS_NOT_IN_ATTRIBUTE_VALUE_TYPE,
PCTE_ENUMERAL_TYPE_IS_NOT_VISIBLE,
PCTE_ENUMERAL_TYPES_ARE_MULTIPLE,
PCTE_EVALUATION_STATUS_IS_INCONSISTENT_WITH_EVALUATION_POINT,
PCTE_EVENT_TYPE_IS_NOT_SELECTED,
PCTE_EXECUTION_CLASS_HAS_NO_USABLE_EXECUTION_SITES,
PCTE_EXECUTION_SITE_IS_INACCESSIBLE,
PCTE_EXECUTION_SITE_IS_NOT_IN_EXECUTION_CLASS,
PCTE_EXECUTION_SITE_IS_UNKNOWN,
PCTE_EXTERNAL_LINK_IS_BAD,
PCTE_EXTERNAL_LINK_IS_NOT_DUPLICABLE,
PCTE_FOREIGN_DEVICE_IS_INVALID,
PCTE_FOREIGN_EXECUTION_IMAGE_HAS_NO_SITE,

PCTE_FOREIGN_EXECUTION_IMAGE_IS_BEING_EXECUTED,
PCTE_FOREIGN_OBJECT_IS_INACCESSIBLE,
PCTE_FOREIGN_SYSTEM_IS_INACCESSIBLE,
PCTE_FOREIGN_SYSTEM_IS_INVALID,
PCTE_FOREIGN_SYSTEM_IS_UNKNOWN,
PCTE_GROUP_IDENTIFIER_IS_IN_USE,
PCTE_GROUP_IDENTIFIER_IS_INVALID,
PCTE_IMAGE_IS_ALREADY_ASSOCIATED,
PCTE_IMAGE_IS_DUPLICATED,
PCTE_INTEGRITY_CONFINEMENT_WOULD_BE_VIOLATED,
PCTE_INTEGRITY_CRITERION_IS_NOT_SELECTED,
PCTE_INTEGRITY_LABEL_IS_INVALID,
PCTE_INTEGRITY_WOULD_BE_VIOLATED,
PCTE_INTERPRETER_IS_INTERPRETABLE,
PCTE_INTERPRETER_IS_NOT_AVAILABLE,
PCTE_KEY_ATTRIBUTE_TYPE_UNAPPLY_IS_FORBIDDEN,
PCTE_KEY_IS_BAD,
PCTE_KEY_IS_NOT_SYSTEM_KEY,
PCTE_KEY_SYNTAX_IS_WRONG,
PCTE_KEY_TYPE_IS_BAD,
PCTE_KEY_TYPES_ARE_MULTIPLE,
PCTE_KEY_UPDATE_IS_FORBIDDEN,
PCTE_KEY_VALUE_AND_EVALUATION_POINT_ARE_INCONSISTENT,
PCTE_KEY_VALUE_DOES_NOT_EXIST,
PCTE_LABEL_IS_OUTSIDE_RANGE,
PCTE_LABEL_RANGE_IS_BAD,
PCTE_LAN_ERROR_EXISTS,
PCTE_LIMIT_WOULD_BE_EXCEEDED,
PCTE_LINK_DESTINATION_DOES_NOT_EXIST,
PCTE_LINK_DESTINATION_IS_NOT_VISIBLE,
PCTE_LINK_DOES_NOT_EXIST,
PCTE_LINK_EXCLUSIVENESS_WOULD_BE_VIOLATED,
PCTE_LINK_EXISTS,
PCTE_LINK_NAME_IS_TOO_LONG_IN_CURRENT_WORKING_SCHEMA,
PCTE_LINK_NAME_SYNTAX_IS_WRONG,
PCTE_LINK_REFERENCE_IS_NOT_EVALUATED,
PCTE_LINK_REFERENCE_IS_UNSET,
PCTE_LINK_TYPE_CATEGORY_IS_BAD,
PCTE_LINK_TYPE_IS_NOT_APPLIED_TO_OBJECT_TYPE,
PCTE_LINK_TYPE_IS_NOT_VISIBLE,
PCTE_LINK_TYPE_IS_UNKNOWN,
PCTE_LINK_TYPE_PROPERTIES_AND_KEY_TYPES_ARE_INCONSISTENT,
PCTE_LINK_TYPE_PROPERTIES_ARE_INCONSISTENT,
PCTE_LOCK_COULD_NOT_BE_ESTABLISHED,
PCTE_LOCK_INTERNAL_MODE_CANNOT_BE_CHANGED,
PCTE_LOCK_IS_NOT_EXPLICIT,
PCTE_LOCK_MODE_IS_NOT_ALLOWED,
PCTE_LOCK_MODE_IS_TOO_STRONG,
PCTE_LOWER_BOUND_WOULD_BE_VIOLATED,
PCTE_MANDATORY_CLASS_IS_ALREADY_DOMINATED,
PCTE_MANDATORY_CLASS_IS_KNOWN,
PCTE_MANDATORY_CLASS_IS_UNKNOWN,
PCTE_MANDATORY_CLASS_NAME_IS_IN_USE,
PCTE_MAXIMUM_USAGE_MODE_WOULD_BE_EXCEEDED,
PCTE_MEMORY_ADDRESS_IS_OUT_OF_PROCESS,
PCTE_MEMORY_REGION_IS_NOT_IN_PROFILING_SPACE,

PCTE_MESSAGE_POSITION_IS_NOT_VALID,
PCTE_MESSAGE_QUEUE_HAS_BEEN_DELETED,
PCTE_MESSAGE_QUEUE_HAS_BEEN_WOKEN,
PCTE_MESSAGE_QUEUE_HAS_NO_HANDLER,
PCTE_MESSAGE_QUEUE_IS_BUSY,
PCTE_MESSAGE_QUEUE_IS_NOT_RESERVED,
PCTE_MESSAGE_QUEUE_IS_RESERVED,
PCTE_MESSAGE_QUEUE_TOTAL_SPACE_WOULD_BE_TOO_SMALL,
PCTE_MESSAGE_QUEUE_WOULD_BE_TOO_BIG,
PCTE_MESSAGE_TYPES_NOT_FOUND_IN_QUEUE,
PCTE_NON_BLOCKING_IO_IS_INVALID,
PCTE_NOTIFIER_KEY_DOES_NOT_EXIST,
PCTE_NOTIFIER_KEY_EXISTS,
PCTE_OBJECT_ARCHIVING_IS_INVALID,
PCTE_OBJECT_CANNOT_BE_STABILIZED,
PCTE_OBJECT_CRITERION_IS_NOT_SELECTED,
PCTE_OBJECT_HAS_COPIES,
PCTE_OBJECT_HAS_EXTERNAL_LINKS_PREVENTING_DELETION,
PCTE_OBJECT_HAS_GROUP_WHICH_IS_ALREADY_OWNER,
PCTE_OBJECT_HAS_INTERNAL_LINKS_PREVENTING_DELETION,
PCTE_OBJECT_HAS_LINKS_PREVENTING_DELETION,
PCTE_OBJECT_IS_A_PROCESS,
PCTE_OBJECT_IS_A_REPLICA_SET,
PCTE_OBJECT_IS_ALREADY_IN_RESOURCE_GROUP,
PCTE_OBJECT_IS_ARCHIVED,
PCTE_OBJECT_IS_IN_USE_FOR_DELETE,
PCTE_OBJECT_IS_IN_USE_FOR_MOVE,
PCTE_OBJECT_IS_INACCESSIBLE,
PCTE_OBJECT_IS_INACCESSIBLY_ARCHIVED,
PCTE_OBJECT_IS_LOCKED,
PCTE_OBJECT_IS_NOT_ACCOUNTABLE_RESOURCE,
PCTE_OBJECT_IS_NOT_ARCHIVED,
PCTE_OBJECT_IS_NOT_IN_RESOURCE_GROUP,
PCTE_OBJECT_IS_NOT_LOCKED,
PCTE_OBJECT_IS_NOT_MASTER_REPLICATED_OBJECT,
PCTE_OBJECT_IS_NOT_MOVABLE,
PCTE_OBJECT_IS_NOT_ON_ADMINISTRATION_VOLUME,
PCTE_OBJECT_IS_NOT_ON_MASTER_VOLUME_OF_REPLICA_SET,
PCTE_OBJECT_IS_NOT_REPLICABLE,
PCTE_OBJECT_IS_NOT_REPLICATED_ON_VOLUME,
PCTE_OBJECT_IS_OF_WRONG_TYPE,
PCTE_OBJECT_IS_OPERATED_ON,
PCTE_OBJECT_IS_PREDEFINED_REPLICATED,
PCTE_OBJECT_IS_REPLICATED,
PCTE_OBJECT_IS_STABLE,
PCTE_OBJECT_LABEL_CANNOT_BE_CHANGED_IN_TRANSACTION,
PCTE_OBJECT_OWNER_CONSTRAINT_WOULD_BE_VIOLATED,
PCTE_OBJECT_OWNER_VALUE_WOULD_BE_INCONSISTENT_WITH_ATOMIC_ACL,
PCTE_OBJECT_REFERENCE_IS_INTERNAL,
PCTE_OBJECT_REFERENCE_IS_INVALID,
PCTE_OBJECT_REFERENCE_IS_UNSET,
PCTE_OBJECT_TYPE_IS_ALREADY_IN_DESTINATION_SET,
PCTE_OBJECT_TYPE_IS_INVALID,
PCTE_OBJECT_TYPE_IS_NOT_IN_DESTINATION_SET,
PCTE_OBJECT_TYPE_IS_NOT_VISIBLE,
PCTE_OBJECT_TYPE_IS_UNKNOWN,

PCTE_OBJECT_TYPE_WOULD_HAVE_NO_PARENT_TYPE,
PCTE_OBJECT_TYPES_MISMATCH,
PCTE_OPEN_KEY_IS_INVALID,
PCTE_OPENING_MODE_IS_INVALID,
PCTE_OPERATION_HAS_TIMED_OUT,
PCTE_OPERATION_IS_INTERRUPTED,
PCTE_OPERATION_IS_NOT_ALLOWED_ON_TYPE,
PCTE_PARENT_BASIC_TYPES_ARE_MULTIPLE,
PCTE_PATHNAME_SYNTAX_IS_WRONG,
PCTE_POSITION_HANDLE_IS_INVALID,
PCTE_POSITION_IS_INVALID,
PCTE_POSITIONING_IS_INVALID,
PCTE_PREFERENCE_DOES_NOT_EXIST,
PCTE_PREFERRED_LINK_KEY_IS_BAD,
PCTE_PREFERRED_LINK_TYPE_IS_UNSET,
PCTE_PRIVILEGE_IS_NOT_GRANTED,
PCTE_PROCESS_CONFIDENTIALITY_IS_NOT_DOMINATED,
PCTE_PROCESS_HAS_NO_UNTERMINATED_CHILD,
PCTE_PROCESS_INTEGRITY_DOES_NOT_DOMINATE,
PCTE_PROCESS_IS_IN_TRANSACTION,
PCTE_PROCESS_IS_INACCESSIBLE,
PCTE_PROCESS_IS_INITIAL_PROCESS,
PCTE_PROCESS_IS_NOT_ANCESTOR,
PCTE_PROCESS_IS_NOT_CHILD,
PCTE_PROCESS_IS_NOT_TERMINABLE_CHILD,
PCTE_PROCESS_IS_NOT_THE_CALLER,
PCTE_PROCESS_IS_THE_CALLER,
PCTE_PROCESS_IS_UNKNOWN,
PCTE_PROCESS_LABELS_WOULD_BE_INCOMPATIBLE,
PCTE_PROCESS_LACKS_REQUIRED_STATUS,
PCTE_PROCESS_TERMINATION_IS_ALREADY_ACKNOWLEDGED,
PCTE_PROFILING_IS_NOT_SWITCHED_ON,
PCTE_PROGRAM_GROUP_IS_NOT_EMPTY,
PCTE_RANGE_IS_OUTSIDE_RANGE,
PCTE_REFERENCE_CANNOT_BE_ALLOCATED,
PCTE_REFERENCE_NAME_IS_INVALID,
PCTE_REFERENCED_OBJECT_IS_NOT_MUTABLE,
PCTE_REFERENCED_OBJECT_IS_UNSET,
PCTE_RELATIONSHIP_TYPE_PROPERTIES_ARE_INCONSISTENT,
PCTE_REPLICA_SET_COPY_IS_NOT_EMPTY,
PCTE_REPLICA_SET_HAS_COPY_VOLUMES,
PCTE_REPLICA_SET_IS_NOT_EMPTY,
PCTE_REPLICA_SET_IS_NOT_KNOWN,
PCTE_REPLICATED_COPY_IS_IN_USE,
PCTE_REPLICATED_COPY_UPDATE_IS_FORBIDDEN,
PCTE_RESOURCE_GROUP_IS_KNOWN,
PCTE_RESOURCE_GROUP_IS_UNKNOWN,
PCTE_REVERSE_KEY_IS_BAD,
PCTE_REVERSE_KEY_IS_NOT_SUPPLIED,
PCTE_REVERSE_KEY_IS_SUPPLIED,
PCTE_REVERSE_LINK_EXISTS,
PCTE_SDS_IS_IN_A_WORKING_SCHEMA,
PCTE_SDS_IS_KNOWN,
PCTE_SDS_IS_NOT_EMPTY_NOR_VERSION,
PCTE_SDS_IS_UNDER_MODIFICATION,
PCTE_SDS_IS_UNKNOWN,

PCTE_SDS_NAME_IS_DUPLICATE,
PCTE_SDS_NAME_IS_INVALID,
PCTE_SDS_WOULD_APPEAR_TWICE_IN_WORKING_SCHEMA,
PCTE_SECURITY_GROUP_ALREADY_HAS_THIS_SUBGROUP,
PCTE_SECURITY_GROUP_IS_ALREADY_ENABLED,
PCTE_SECURITY_GROUP_IS_IN_USE,
PCTE_SECURITY_GROUP_IS_KNOWN,
PCTE_SECURITY_GROUP_IS_NOT_A_SUBGROUP,
PCTE_SECURITY_GROUP_IS_NOT_ADOPTABLE,
PCTE_SECURITY_GROUP_IS_NOT_ENABLED,
PCTE_SECURITY_GROUP_IS_PREDEFINED,
PCTE_SECURITY_GROUP_IS_REQUIRED_BY_OTHER_GROUPS,
PCTE_SECURITY_GROUP_IS_UNKNOWN,
PCTE_SECURITY_GROUP_WOULD_BE_IN_INVALID_GRAPH,
PCTE_SECURITY_POLICY_WOULD_BE_VIOLATED,
PCTE_STATIC_CONTEXT_CONTENTS_CANNOT_BE_EXECUTED,
PCTE_STATIC_CONTEXT_IS_ALREADY_MEMBER,
PCTE_STATIC_CONTEXT_IS_BEING_WRITTEN,
PCTE_STATIC_CONTEXT_IS_IN_USE,
PCTE_STATIC_CONTEXT_IS_NOT_MEMBER,
PCTE_STATIC_CONTEXT_REQUIRES_TOO_MUCH_MEMORY,
PCTE_STATUS_IS_BAD,
PCTE_TIME_CANNOT_BE_CHANGED,
PCTE_TRANSACTION_CANNOT_BE_COMMITTED,
PCTE_TYPE_HAS_DEPENDENCIES,
PCTE_TYPE_HAS_NO_LOCAL_NAME,
PCTE_TYPE_IDENTIFIER_IS_INVALID,
PCTE_TYPE_IDENTIFIER_SYNTAX_IS_WRONG,
PCTE_TYPE_IDENTIFIER_USAGE_IS_INVALID,
PCTE_TYPE_IS_ALREADY_APPLIED,
PCTE_TYPE_IS_ALREADY_KNOWN_IN_SDS,
PCTE_TYPE_IS_NOT_APPLIED,
PCTE_TYPE_IS_NOT_DESCENDANT,
PCTE_TYPE_IS_NOT_VISIBLE,
PCTE_TYPE_IS_OF_WRONG_KIND,
PCTE_TYPE_IS_UNKNOWN,
PCTE_TYPE_IS_UNKNOWN_IN_SDS,
PCTE_TYPE_IS_UNKNOWN_IN_WORKING_SCHEMA,
PCTE_TYPE_NAME_IN_SDS_IS_DUPLICATE,
PCTE_TYPE_NAME_IS_INVALID,
PCTE_TYPE_OF_OBJECT_IS_INVALID,
PCTE_TYPE_REFERENCE_IS_INVALID,
PCTE_TYPE_REFERENCE_IS_UNSET,
PCTE_UNLOCKING_IN_TRANSACTION_IS_FORBIDDEN,
PCTE_UPPER_BOUND_WOULD_BE_VIOLATED,
PCTE_USAGE_MODE_ON_ATTRIBUTE_TYPE_WOULD_BE_VIOLATED,
PCTE_USAGE_MODE_ON_LINK_TYPE_WOULD_BE_VIOLATED,
PCTE_USAGE_MODE_ON_OBJECT_TYPE_WOULD_BE_VIOLATED,
PCTE_USER_CRITERION_IS_NOT_SELECTED,
PCTE_USER_GROUP_IS_IN_USE,
PCTE_USER_GROUP_LACKS_ALL_USERS_AS_SUPERGROUP,
PCTE_USER_GROUP_WOULD_NOT_HAVE_ALL_USERS_AS_SUPERGROUP,
PCTE_USER_IS_ALREADY_CLEARED_TO_CLASS,
PCTE_USER_IS_ALREADY_MEMBER,
PCTE_USER_IS_IN_USE,
PCTE_USER_IS_NOT_CLEARED,

```
    PCTE_USER_IS_NOT_CLEARED_TO_CLASS,
    PCTE_USER_IS_NOT_MEMBER,
    PCTE_USER_IS_UNKNOWN,
    PCTE_VALUE_TYPE_IS_INVALID,
    PCTE_VERSION_GRAPH_IS_INVALID,
    PCTE_VERSION_IS_REQUIRED,
    PCTE_VOLUME_CANNOT_BE_MOUNTED_ON_DEVICE,
    PCTE_VOLUME_EXISTS,
    PCTE_VOLUME_HAS_OBJECT_OUTSIDE_RANGE,
    PCTE_VOLUME_HAS_OBJECTS_IN_USE,
    PCTE_VOLUME_HAS_OTHER_LINKS,
    PCTE_VOLUME_HAS_OTHER_OBJECTS,
    PCTE_VOLUME_IDENTIFIER_IS_INVALID,
    PCTE_VOLUME_IS_ADMINISTRATION_VOLUME,
    PCTE_VOLUME_IS_ALREADY_COPY_VOLUME_OF_REPLICA_SET,
    PCTE_VOLUME_IS_ALREADY_MOUNTED,
    PCTE_VOLUME_IS_FULL,
    PCTE_VOLUME_IS_INACCESSIBLE,
    PCTE_VOLUME_IS_MASTER_VOLUME_OF_REPLICA_SET,
    PCTE_VOLUME_IS_NOT_COPY_VOLUME_OF_REPLICA_SET,
    PCTE_VOLUME_IS_NOT_MASTER_OR_COPY_VOLUME_OF_REPLICA_SET,
    PCTE_VOLUME_IS_READ_ONLY,
    PCTE_VOLUME_IS_UNKNOWN,
    PCTE_WORKSTATION_EXISTS,
    PCTE_WORKSTATION_HAS_NO_CHOICE_OF_VOLUME_FOR_REPLICA_SET,
    PCTE_WORKSTATION_IDENTIFIER_IS_INVALID,
    PCTE_WORKSTATION_IS_BUSY,
    PCTE_WORKSTATION_IS_CONNECTED,
    PCTE_WORKSTATION_IS_NOT_CONNECTED,
    PCTE_WORKSTATION_IS_UNKNOWN,
/* IDL-binding-specific errors */

    PCTE_ACCESS_MASK_IS_INVALID,
    PCTE_ACCESS_AT_INVALID_ADDRESS,
    PCTE_OUT_OF_MEMORY,
    PCTE_SEQUENCE_INVALID_TYPE,
    PCTE_SEQUENCE_BAD_HANDLE,
    PCTE_SEQUENCE_OUT_OF_DATA,
    PCTE_SEQUENCE_INVALID_INDEX,
    PCTE_STRING_TOO_SHORT,
    PCTE_VALUE_IS_OUT_OF_RANGE,
    PCTE_VALUE_TYPE_IDENTIFIER_DOES_NOT_MATCH,
/* New error conditions for fine-grain support *

    PCTE_OBJECT_CANNOT_BE_CLUSTERED,
    PCTE_OBJECT_IS_FINE_GRAIN,
    PCTE_CLUSTER_EXISTS,
    PCTE_CLUSTER_HAS_OTHER_LINKS,
    PCTE_CLUSTER_IS_UNKNOWN,
```

```
/* New error conditions for object orientation */

    PCTE_NUMBER_OF_PARAMETERS_IS_WRONG,
    PCTE_OPERATION_METHOD_CANNOT_FOUND,
    PCTE_OPERATION_METHOD_CANNOT_BE_ACTIVATED,
    PCTE_TYPE_IS_ALREADY_CONSTRAINED,
    PCTE_TYPE_OF_PARAMETER_IS_WRONG,

};
#endif
```

Annex A

(informative)

Comparison with ISO/IEC 13719-2

This annex describes the differences between the IDL source files and the corresponding C headers in ISO/IEC 13719-1.

A.1 Object Management

In this clause there are three interfaces (and their corresponding '_h' versions): Pcte_link, Pcte_object and Pcte_version. Pcte_version is at the same level as Pcte_object for consistency reasons.

A.2 Schema Management

There are no major differences, except for the operations on the working schema. As there is no "working_schema" object type, these operations are pseudo-operations.

A.3 Volumes, devices and archives

There are no major differences. There are five interfaces implementing all the operations on devices.

A.4 File, pipes and devices

The 'open' operation is a pseudo-operation as the controlling object is a constant and the operation returns a reference to an object supporting the Pcte_contents interface. There is also an extra interface (Pcte_position_handle) used to discard the position handle.

A.5 Process execution

The 'create' operation is applied to the process issuing the operation (self), so there is a slight asymmetry in the use of this interface.

A.6 Message queues

This clause presents only one major difficulty: the message queue handler. There are two issues connected with the use of this functionality, when the client and the server implementing the PCTE interface are not in the same process space:

- the handle is meaningless for the other process. In this case the solution is to have a generic handle on the CORBA server side and send a request back to the client, signalling the wake-up event.

- CORBA does not support asynchronous invokes, so there is no obvious mechanism to wake-up the client when a message is deposited in the queue. The client must be able to work as a server to accept the wake-up coming from the server accessing the PCTE object base.

A.7 Notification

There is no "notification" object that could be used as a controlling object; the message queue is used as controlling object instead. As a result a "message_queue" object supports also the Pcte_notify interface.

A.8 Concurrency and Integration Control

This clause has two interfaces Pcte_activity and Pcte_lock, and no major differences.

A.9 Replication

The interface Pcte_replica_set has a one pseudo-IDL operation: Pcte_replica_set_create is applied to a pseudo-object. This interface is applied to a "replica_set object". The Pcte_replicated_object interface is applied to any object reference.

A.10 Network connection

The interface has many pseudo-operations as some operations are implicitly applied to the local workstation and have no controlling "workstation" object.

A.11 Discretionary security

The most notable difference is that a few mandatory security operations of the have been moved into the Pcte_group interface.

A.12 Mandatory security

This clause has been split into two parts: one for the type definitions and one for the interface definitions.

A.13 Auditing

There are no major differences.

A.14 Accounting

Most of the operations have been put into the interface Pcte_accounting, rather than 'Pcte_accounting_log', so as to accommodate under the same interface also the 'on', 'off' and 'record_write' operations.

A.15 Operation reshuffling

get_control and set_control are moved to the Pcte_contents interface.

copy_from_foreign_system and copy_to_foreign_system are moved to the Pcte_contents interface.

time_set and time_get are moved to the Pcte_workstation interface.

select_replica_set_volume and unselect_replica_set_volume are moved to the Pcte_workstation interface.

Pcte_accounting_log is renamed Pcte_accounting_file to avoid clashes with the interface of the same name and to agree with the enumeration style.

check_permission, get_acl_entry, and set_acl_entry are moved to the Pcte_object interface.

disable_for_confidentiality_downgrade, enable_for_confidentiality_downgrade, disable_for_integrity_upgrade, and enable_for_integrity_upgrade are moved to the Pcte_group interface.

set_confidentiality_label, set_integrity_label, set_floating_confidentiality_level, and set_floating_integrity_level are moved to the Pcte_process interface.

Annex B

(informative)

IDL file structure

file name	file dependences	interface name	supporting object type
accounting.idl	types.idl references.idl sequences.idl oms_types.idl discretionary_types.idl mandatory_types.idl	Pcte_accounting	accounting_log
		Pcte_consumer_group	consumer_group
		Pcte_resource_group	resource_group
activities.idl	types.idl references.idl oms_types.idl discretionary_types.idl	Pcte_activity	activity
		Pcte_lock	object
auditing.idl	types.idl references.idl sequences.idl oms_types.idl discretionary_types.idl mandatory_types.idl	Pcte_audit	audit_file
contents.idl	types.idl references.idl	Pcte_contents	file
		Pcte_position_handle	NONE
devices.idl	types.idl references.idl sequences.idl discretionary_types.idl mandatory_types.idl	Pcte_archive	archive
		Pcte_device	device
		Pcte_h_device	device
		Pcte_volume	volume
		Pcte_h_volume	volume