Programmer's Guide to the IE Facility

A Facility for Examining the Information Entities in an Information Object

Pei Weng
Stephen M. Moore

Mallinckrodt Institute of Radiology
Electronic Radiology Laboratory
510 South Kingshighway Boulevard
St. Louis, Missouri 63110
314/362-6965 (Voice)
314/362-6971 (FAX)

Version 2.10.0

August 3, 1998

This document describes a subroutine library which is used to examine a DICOM Information Object by parsing information entities and modules.

---

1 Introduction

Part 3 of the DICOM V3 Standard defines several information models and specifies Information Object Classes for composite and normalized objects. Each object contains information that is classified into one or more *Information Entities*. *Modules* provide a further level of abstraction below Information Entity and contain "a set of Attributes within an Information Entity or Normalized IOD which are logically related to each other".

Figure 1 shows a model of a composite Information Object. Note that one attribute is contained in two different modules. This implies that a strict hierarchical model cannot be applied to this data.

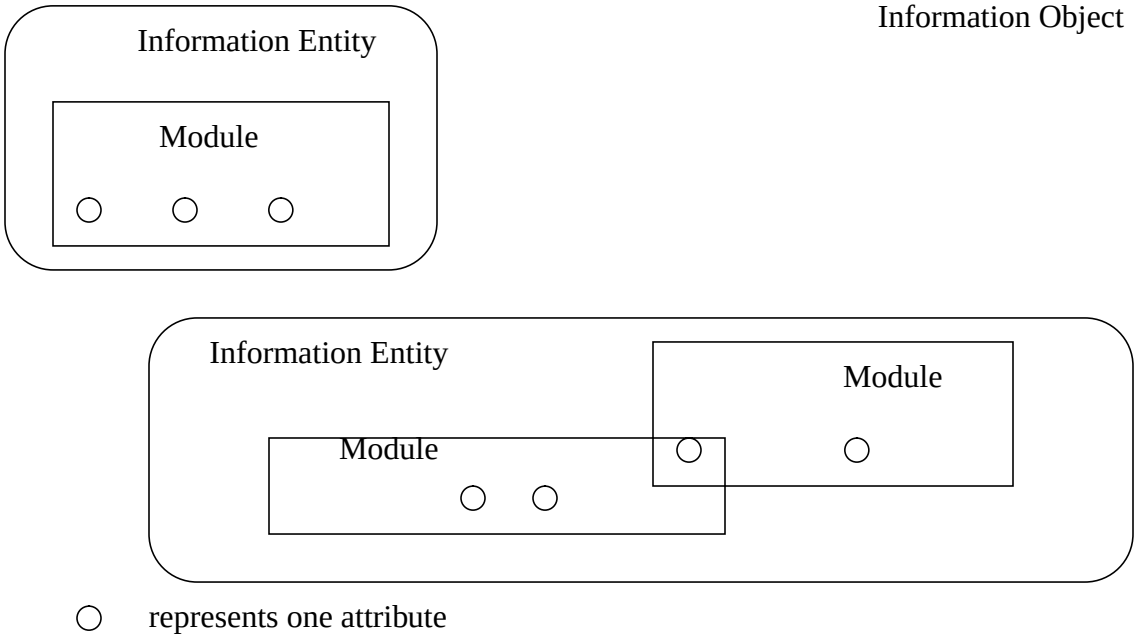

FIGURE 1. Example Composite Information Object

Part 3 of the Standard contains tables which define the Modules and Information Entities which are present in each Information Object. Modules are classified as mandatory or user optional. Other tables in Part 3 define the Attributes which are present in a Module. As with the V2 Standard, individual attributes are classified as type 1, 2 or 3.

This facility provides functions which allow the user to examine Information Objects according to the rules defined by the tables in Part 3 of the Standard. Users create or import Information Objects using the DCM facility. These Information Objects can then be examined by the IE facility to determine which Information Entities, Modules and Attributes are present and/or missing.

This facility also provides lookup functions which allow the caller to determine the Information Entities, Modules and Attributes by SOP Class without providing an instance of an Information Object.

2 Data Structures

The include file with this facility defines four data structures that are important to the user of this facility. The structures are defined hierarchically according to the IOD definition in Part 3 of the DICOM standard.

2.1 IE_OBJECT

The include file defines the IE_OBJECT as a structure with the following items:

```
void                    *reserved[2]
IE_STRUCTURETYPE        structureType
char                    classUID[DICOM_UI_LENGTH+1]
char                    objectDescription[64]
IE_STATUS               status
LST_HEAD                *ieList
```

The two reserved void pointers are necessary when the structure is managed by the LST facility.

structureType and *status* are enumerated types which are defined in the include file for the IE facility. Valid structureTypes are:

IE_K_INFORMATIONOBJECT	DICOM Information Object
IE_K_INFORMATIONENTITY	Information Entity
IE_K_MODULE	Module
IE_K_ATTRIBUTE	Attribute

Valid status types are:

IE_MISSING	NULL structure
IE_INCOMPLETE	Incomplete Structure
IE_COMPLETE	Structure Cmplete

classUID identifies the SOP Class of the DICOM information object the structure represents.

objectDescription gives an ASCII description of the object. This field is normally filled by the IE_facility.

ieList is the head of a list of Information Entities in the DICOM Information Object.

2.2 IE_INFORMATIONENTITY

The include file defines the IE_INFORMATIONENTITY as a structure with the following items:

```
void                    *reserved[2]
IE_STRUCTURETYPE        structureType
IE_IETYPE               ieType
char                    ieDescription[64]
IE_IEREQUIREMENT        requirement
IE_STATUS               status
LST_HEAD                *moduleList
```

The two reserved void pointers are necessary when the structure is managed by the LST facility.

structureType, ieType, requirement and *status* are enumerated types which are defined in the include file for the IE facility. Vaild structureTypes are:

IE_K_INFORMATIONOBJECT	DICOM Information Object
IE_K_INFORMATIONENTITY	Information Entity
IE_K_MODULE	Module
IE_K_ATTRIBUTE	Attribute

Valid ieTypes are:

IE_K_PATIENTIE	Patient Information Entity
IE_K_STUDYIE	Study Information Entity
IE_K_SERIESIE	Series Information Entity
IE_K_FRAMEOFREFERENCEIE	Frame of Reference Information Entity
IE_K_EQUIPMENTIE	Equipment Information Entity
IE_K_IMAGEIE	Image Information Entity
IE_K_OVERLAYIE	Overlay Information Entity
IE_K_CURVEIE	Curve Information Entity

Valid requirement types are:

IE_K_REQUIRED	mandatory
IE_K_OPTIONAL	optional

Valid status types are:

IE_MISSING	NULL structure
IE_INCOMPLETE	Incomplete Structure
IE_COMPLETE	Structure Complete

ieDescription gives an ASCII description to the information entity. This field is usually filled by the IE facility.

moduleList is the head pointer which points to a list of modules within an Information Entity.

2.3 IE_MODULE

The include file defines the IE_MODULE as a structure with the following items:

```
void                    *reserved[2]
IE_STRUCTURETYPE        structureType
IE_MODULETYPE           moduleType
char                    moduleDescription[64]
IE_IEREQUIREMENT        requirement
STATUS                  status
LST_HEAD                *attributeList
```

The two reserved void pointers are necessary when the structure is managed by the LST facility.

structureType, ieType, requirement and *status* are enumerated types which are defined in the include file for the IE_facility.

Vaild structureTypes are:

IE_K_INFORMATIONOBJECT	DICOM Information Object
IE_K_INFORMATIONENTITY	Information Entity
IE_K_MODULE	Module
IE_K_ATTRIBUTE	Attribute

Valid moduleTypes are:

IE_K_PATIENTMODULE	Patient Module
IE_K_GENERALSTUDYMODULE	General Study Module
IE_K_PATIENTSTUDYMODULE	Patient Study Module
IE_K_GENERALSERIESMODULE	General Series Module
IE_K_CRSERIESMODULE	CR Series Module
IE_K_NMSERIESMODULE	NM Series Module
IE_K_NMSPECTACQSERIESMODULE	NM SPECT ACQ Series Module
IE_K_NMMULTIACQSERIESMODULE	NM Multi-gated ACQ Series Module
IE_K_USSERIESMODULE	IE_K_USSERIESMODULE
IE_K_USLOOPSERIESMODULE	US Loop Series Module
IE_K_REGIONCALBRATIONMODULE	Region Calibration Module
IE_K_FRAMEOFREFERENCEMODULE	Frame of Reference Module
IE_K_GENERALEQUIPMENTMODULE	General Equipment Module
IE_K_NMEQUIPMENTMODULE	NM Equipment Module
IE_K_FPEQUIPMENTMODULE	FP Equipment Module
IE_K_SOPCOMMONMODULE	SOP Comon Module

IE_K_IMAGEPLANEMODULE	Image Plane Module
IE_K_IMAGEPIXELMODULE	Image Pixel Module
IE_K_CINEMODULE	Cine Module
IE_K_MULTIFRAMEMODULE	Multi Frame Module
IE_K_CONTRASTMODULE	Contrast Module
IE_K_CRIMAGEMODULE	CR Image Module
IE_K_MRIMAGEMODULE	MR Image Module
IE_K_NMIMAGEMODULE	NM Image Module
IE_K_USIMAGEMODULE	US Image Module
IE_K_FDIMAGEMODULE	FD Image Module
IE_K_OVERLAYPLANEMODULE	Overlay Module
IE_K_LOOKUPTABLEMODULE	Lookup Table Module
IE_K_CURVEMODULE	Curve Module
IE_K_OVERLAYIDENTIFICATIOMODULE	Overlay ID Module
IE_K_CURVEIDENTIFICATIONMODULE	Curve ID Module
IE_K_CURVEPLANEMODULE	Curve Plane Module

Valid requirement types are:

IE_K_REQUIRED	mandatory
IE_K_OPTIONAL	optional

Valid status types are:

IE_MISSING	NULL structure
IE_INCOMPLETE	Incomplete Structure
IE_COMPLETE	Structure Complete

moduleDescription gives an ASCII description to the module. This field is usually filled by the IE facility. *attributeList* is the head of a list of attributes which form the Module.

2.4 IE_ATTRIBUTE

The include file defines the IE_ATTRIBUTE as a structue with the following items:

```
void                    *reserved[2]
IE_STRUCTURETYPE        structureType
DCM_ELEMENT             element
IE_ATTRIBUTESREQUIREMENT requirement
STATUS                  status
```

The two reserved void pointers are necessary when the structure is managed by the LST facility.

structureType, requirement and *status* are enumerated types which are defined in the include file for the IE facility.

Vaild structureTypes are:

`IE_K_INFORMATIONOBJECT`	DICOM Information Object
`IE_K_INFORMATIONENTITY`	Information Entity
`IE_K_MODULE`	Module
`IE_K_ATTRIBUTE`	Attribute

Valid requirement types are:

`IE_K_TYPE1`	Mandatory
`IE_K_TYPE1C`	Mandatory, conditional
`IE_K_TYPE2`	Mandatory, but length of the data could be 0.
`IE_K_TYPE2C`	Equal to IE_K_TYPE2, but conditional
`IE_K_TYPE3`	Optional

Valid status types are:

`IE_MISSING`	NULL structure
`IE_INCOMPLETE`	Incomplete Structure
`IE_COMPLETE`	Structure Complete

element is the individual DICOM data element defined in the include file for the DCM facility.

3 Include Files

To use IE functions, applications need to include these files in the order given below:

```
#include “dicom.h”
#include “condition.h”
#include “lst.h”
#include “dicom_objects.h”
#include “dicom_ie.h”
```

4 Return Values

The following returns are possible from the IE facility:

IE_NORMAL	Normal, successful completion.
IE_ILLEGALOBJEC	Caller attempted function on an object that is not a legal DICOM information object.
IE_OBJECTINCOMPLETE	Missing required information in a DICOM information object according to its definition.
IE_IEINCOMPLETE	Missing required information in an Information Entity according to its definition.
IE_IEMISSING	Missing required information in an Information Entity.
IE_MODULEINCOMPLETE	Missing required information in a Module according to its definition.
IE_MODULEMISSING	Missing all required information in a Module.
IE_LISTFAILURE	Failure in a list operation which caused IE function to fail.
IE_MALLOCFAILURE	Failure in memory allocation which caused IE function to fail.

5 IE Routines

This section provides detailed documentation for each IE facility routine.

IE_ExamineInformationEntity

Name

IE_ExamineInformationEntity - examine an Information Entitiy in a DICOM Information Object to determine the Modules which are present and to determine if the Information Entity is complete.

Synopsis

CONDITION IE_ExamineInformationEntity(DCM_OBJECT **dcmObject, IE_TYPE type,
IE_INFORMATIONENTITY **ieEntity)

dcmObject	Address of caller's pointer to the input DICOM object.
ietype	Type of the particular Information Entity which the caller wants to examine.
ieEntity	Address of caller's pointer to an IE_INFORMATIONENTITY. This function allocates an IE_INFORMATIONENTITY structure and places the address of the structure in caller's pointer.

Description

IE_ExamineInformationEntity examines one Information Entity in a DICOM object and creates an IE_INFORMATIONENTITY structure. The structure contains a description of the Information Entity, a flag indicating if the Information Entity is required or optional, and a list of Modules which should be present in this Information Entity.

Each Module in the IE_INFORMATIONENTITY list contains a description of that Module, a flag indicating if it is optional or required, and a flag which indicates if the Module is complete, incomplete, or missing. The function does not fill in the list of Attributes in each Module structure. The IE_INFORMATIONENTITY structure is tagged "complete" if all required Modules are complete.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
IE_IEINCOMPLETE
IE_IEMISSING
IE_LISTFAILURE
IE_MALLOCFAILURE
```

IE_ExamineModule

Name

IE_ExamineModule - examine a Module in a DICOM Information Object to determine the attributes which are present and if the Module is complete.

Synopsis

CONDITION IE_ExamineModule(DCM_OBJECT **dcmObject, IE_IETYPE itType,
IE_MODULETYPE moduleType, IE_MODULE **ieModule)

dcmObject	Address of caller's pointer to the input DICOM object.
ieType	Type of the Information Entity to be examined.
moduleType	Type of Module to examine.
ieModule	Address of caller's pointer to an IE_MODULE. This function allocates an IE_MODULE structure and places the address of the structure in the caller's pointer.

Description

IE_ExamineModule examines one Module in a DICOM object and creates an IE_MODULE structure. The structure contains a description of the Module, a flag indicating if this Module is required or optional and a list of Attributes which should be present in the Module.

Each Attribute in the IE_MODULE list contains a DCM_ELEMENT structure, a flag which indicates if this Attribute is optional or required, and a flag indicating if this Attribute is missing or complete. The IE_MODULE structure is tagged "complete" if all required Attributes are present.

Because Modules may be context sensitive (they may depend on the Information Object and Information Entity), the caller specifies the Information Entity which contains the Module (through the ieType argument) as well as the type of the Module to be examined.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
IE_MODULEINCOMPLETE
IE_MODULEMISSING
IE_LISTFAILURE
IE_MALLOCFAILURE
```

IE_ExamineObject

Name

IE_ExamineObject - examine an DICOM Information Object to determine the Information Entities which are present and if the Information Object is complete.

Synopsis

CONDITION IE_ExamineObject(DCM_OBJECT **dcmObject, IE_OBJECT **ieObject)

dcmObject	Address of caller's pointer to the input DICOM object.
ieObject	Address of caller's pointer to an IE_OBJECT. This function allocates an IE_OBJECT structure and places the address of the structure in the caller's pointer.

Description

IE_ExamineObject examines a DICOM information object and creates an IE_OBJECT structure. The structure contains a description of the object and a list of Information Entities which should be present in this DICOM object (as defined in Part 3 of the DICOM standard).

Each Information Entity in the IE_OBJECT list a contains a description of that Information Entity, a flag which indicates if it is optional or required, and a flag indicating if the Information Entity is complete, incomplete or missing. This function does not fill in the list of Modules in each information Entity structure. The IE_OBJECT is tagged "complete" if all required Information Entities are present.

Notes

Return Values

```
IE_NORMAL
IE_OBJECTINCOMPLETE
IE_ILLEGALDCMOBJECT
IE_LISTFAILURE
IE_MALLOCFAILURE
```

IE_Free

Name

IE_Free - free any structure created by the IE facility.

Synopsis

CONDITION IE_Free(void **object)

object Address of caller's pointer to an IE structure.

Description

IE_Free frees any structure that has been created by a function in the IE_facility. This includes functions:

IE_ExamineObject
IE_ExamineInformationEntity
IE_ExamineModule
IE_ObjectRequirements
IE_IERequirements
IE_ModuleRequirements

This function determines the type of structure passed by the caller and frees any lists contained in the structure.After the lists are free, the function frees the structure itself.

The caller passes the address of a pointer to an IE structure. After the structure is freed, this function destroys the caller's reference to the structure by writing NULL into the caller's pointer.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
```

IE_IERequirements

Name

IE_IERequirements - determine the Modules that are required for an Information Entity.

Synopsis

CONDITION IE_IERequirements(char *classUID, IE_IETYPE ieType,
IE_INFORMATIONENTITY **ieEntity)

classUID	An ASCII string which identifies the SOP Class of the DICOM information object.
ieType	Identifies the type of the Information Entity the caller wants to examine.
ieEntity	Address of caller's pointer to an IE_INFORMATIONENTITY. This function allocates an IE_INFORMATIONENTITY structure and places the address of the structure in the caller's pointer.

Description

IE_IERequirements determines which Modules are required for an Information Entity in a DICOM object and returns an IE_INFORMATIONENTITY structure which contains those requirements. The structure contains a description of this Information Entity, a flag indicating if it is required or optional and a list of Modules required for this Information Entity.

Each Module in the IE_INFORMATIONENTITY list contains a description of the Module and a flag which indicates if it is required. This function does not fill in the list of Attributes in each IE_MODULE structure.

This function is used to determine requirements for an Information Entity in an image belonging to the SOP Class defined by the caller's class UID argument.

The ieType argument is used to specify the type of the Information Entity the caller wants to examine.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
IE_LISTFAILURE
IE_MALLOCFAILURE
```

IE_ModuleRequirements

Name

IE_ModuleRequirements - determine which Attributes are required in a Module.

Synopsis

CONDITION IE_ModuleRequirements(char *classUID, IE_IETYPE ieType,
IE_MODULETYPE moduleType, IE_MODULE **ieModule)

classUID	A string which identifies the SOP Class of the input DICOM Information Object.
ieType	Identifies the type of the Information Entity.
moduleType	Identifies the type of the Module the caller wants to examine.
ieModule	Address of caller's pointer to an IE_MODULE. This function allocates an IE_MODULE structure and places the address of the structure in the caller's pointer.

Description

IE_ModuleRequirements determines which Attributes are required for a Module in a DICOM object and returns an IE_MODULE structure which contains those requirements. The structure contains a description of this Module, a flag indicating if it is required or optional, and a list of Attributes required (type1, type2) for this Module.

Each Attribute in the IE_MODULE list contains a DCM_ELEMENT structure and a flag indicating if it is optional or required. This function fills in the DCM_TAG field in the DCM_ELEMENT structure, other fields of the structure are empty.

This function is used to determine requirements for a Module in Information Object in the SOP Class defined by the caller's classUID argument. Because the Module definition may depend on the SOP Class as well as the Information Entity which contains the Module, the caller also specifies the ieType as well as the moduleType.

This function is used as a dictionary for classes of SOP Information Objects. This function does not examine an instance of an SOP Information Object.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
IE_LISTFAILURE
IE_MALLOCFAILURE
```

IE_ObjectRequirements

Name

IE_ObjectRequirements - determine which Information Entities are required in a DICOM Information Object

Synopsis

CONDITION IE_ObjectRequirements(char *classUID, IE_OBJECT **object)

classUID	An ASCII string which identifies the SOP Class of the DICOM Information Object.
object	Address of caller's pointer to an IE_OBJECT structure. This function allocates an IE_OBJECT structure and places the address of the structure in the caller's pointer.

Description

IE_ObjectRequirements determines what Information Entities are required for a DICOM Information Object belonging to an SOP Class (such as MR, CT, Secondary Capture) and returns an IE_OBJECT structure which contains those requirements. The structure contains a description of the object and a list of Information Entities required (as defined in Part 3 of the Standard) for the object.

Each Information Entity in the IE_OBJECT list contains a description of this Information Entity, and a flag which indicates it is required. This function does not fill in the list of Modules in the IE_INFORMATIONENTITY structure.

This function is used to determine requirements for any image belonging to an SOP Class as defined by the caller's UID argument. This function does not examine a DICOM Information Object.

Notes

Return Values

```
IE_NORMAL
IE_ILLEGALDCMOBJECT
IE_LISTFAILURE
IE_MALLOCFAILURE
```