Date: May 2011

OBJECT MANAGEMENT GROUP

Identity Cross-Reference Service (IXS)

Version 1.0

OMG Document Number:	formal/2011-05-05
Standard Document URL:	http://www.omg.org/spec/IXS/1.0/
Associated File(s)*:	http://www.omg.org/spec/IXS/20101101
	http://www.omg.org/spec/IXS/20101101/EISAdminEditorInterface.wsdl
	http://www.omg.org/spec/IXS/20101101/EISMetaDataInterface.wsdl
	http://www.omg.org/spec/IXS/20101101/EISMgmtAndQueryInterface.wsdl
	http://www.omg.org/spec/IXS/20101101/swpEISAdminEditorInterface.wsdl
	http://www.omg.org/spec/IXS/20101101/swpEISMgmtAndQueryInterface.wsdl

* original file(s): dtc/2010-11-25 (machine-consumable files)

OMG’s Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page *http://www.omg.org*, under Documents, Report a Bug/Issue (http://www.omg.org/technology/agreement.htm).

Table of Contents

Preface

About the Object Management Group

OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at *http://www.omg.org/*.

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A catalog of all OMG Specifications is available from the OMG website at:

http://www.omg.org/technology/documents/spec_catalog.htm

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications

OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM)

Platform Specific Model and Interface Specifications

- CORBAservices
- CORBAfacilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult *http://www.iso.org*

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to *http://www.omg.org/technology/agreement.htm*.

1 Scope

The scope of this specification is the

- interfaces,
- operations, and
- information structure

required to uniquely identify various kinds of entities within disparate systems within a single enterprise and/or across a set of collaborating enterprises.

These service interfaces are intended to be used within a healthcare setting, in which the security of data exchanges is both important and regulated by laws such as HIPAA in the United States.

However, the following aspects of this service, while both important and required, are addressed elsewhere and are out of scope for this specification:

- Security
- Privacy
- Versioning
- Logging
- Internationalization

Separating these functions from the IXS service facilitates the application of a uniform policies across a set of services.

2 Conformance

Compliance [Conformance] to this specification happens by way of conformance to profiles, which are detailed in Chapter 8. Conformance Profiles consist of at least one Functional (or Behavioral) Profile and at least one Semantic Profile.

Functional Profiles are named subsets of the overall set of operations defined within the specification that provide a cohesive set of functionality that makes sense from the service client’s perspective. Note that this is orthogonal to the internal interface structure of the Service from the service provider's perspective. A reflection style interface is also made available by IXS instances that will identify which profiles it supports at any given point in time.

Similarly, Semantic Profiles are named information models that define the “Semantic Signifiers” that are mapped to and used in the inputs and outputs of operations. This allows the same behavioral interface to be used with different content models, e.g., HL7 V2 data structures, HL7 V3 data structures or OpenEHR Archetypes.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

- Web Services Description Language (WSDL) 1.1, W3C Note 15 March 2001, http://www.w3.org/TR/wsdl
- Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000, http://www.w3.org/TR/SOAP
- XML Schema Part 2: Datatypes, W3C Recommendation 02 May 2001, http://www.w3.org/TR/xmlschema-2
- HL7 Entity Identification Service, Release 1, Service Functional Model Specification, http://www.hl7.org/dstucomments/index.cfm
- ISO Reference Model of Open Distributed Processing (RM-ODP), http://www.iso.org
- Service Specifications Framework (SSF), Healthcare Services Specification Project (HSSP), http://hssp.wikispaces.com
- IHE PIX and PDQ including Pediatric Demographics and V3 PIX/PDQ

4 Acronyms

There are a number of acronyms used in this document, and in standards or other documents related to this specification, and here is a brief list of what the most common ones stand for:

Acronym	Full Name
ANSI	American National Standards Institute (U.S.A.)
IXS	Identity Cross-Reference Service
HIPAA	Health Insurance Portability and Accountability Act (U.S.A.)
HITSP	Health Information Technology Standards Panel (U.S.A.) of ANSI
HL7	Health Level 7
HSSP	Healthcare Services Specification Project
IHE	Integrating the Healthcare Enterprise
ISO	International Standards Organization
OMG	Object Management Group
PFM	Platform Functional Model
PIM	Platform Independent Model
PIX – PDQ	Patient Index Query – Patient Demographic Query (from IHE)
PSM	Platform Specific Model
RIM	Reference Information Model defined by HL7
RFP	Request for Proposal

RM-ODP	**Reference Model of Open Distributed Processing defined by ISO**
SDO	**Standards Development Organization**
SFM	**Service Functional Model**
UML	**Universal Modeling Language**

5 Symbols

There are no symbols defined in this specification.

6 Additional Information

6.1 Acknowledgements

The following organizations contributed to this specification:

- Intel Corporation
- Kaiser Permanente
- Ocean Informatics
- Oracle
- Software Partners
- Web Reach

6.2 Internationalization

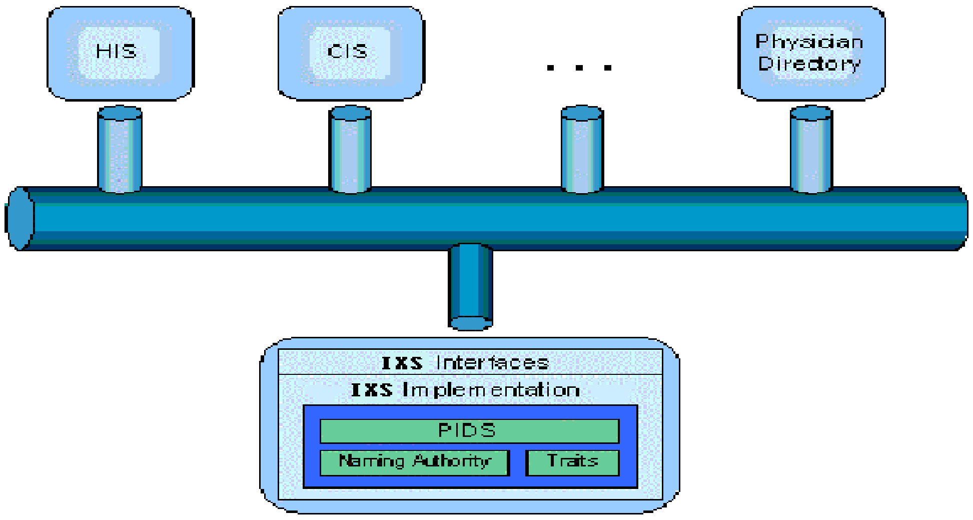
This IXS specification primarily reflects the contributions of U.S.-based submitters and supporters. However, certain measures were taken to maximize international alignment. First, some submitters and supports do business internationally and will have their initial implementations in these environments. Second, by aligning with the trends of the internationally-based SDOs mentioned above, specifically, HL7 and IHE, we hope to indirectly harmonize with international requirements. Finally, the use of semantic signifiers as traits parameters introduces a degree of flexibility that we hope will optimally facilitate international differences.

6.3 Relation to Existing OMG Specifications

The following is extracted from Section 6.3 of the OMG IXS RFP:

Reference	Description	Relationship to IXS
OMG Person Identification Service (PIDS) Specification Version 1.1, April 2001	A specification of a Person Identification Service with UML models and a complete specification in CORBA IDL. This includes most of the functions for managing a patient identity. The traits of an identity in this specification used the HL7 v2.5 and the vCard specifications.	This specification has been used as the basis for many implementations in different industries. This has been used as a primary source for many of the ideas in this specification. This specification could be seen as bringing PIDS up to date, although the additional "Entity" abstraction provides for additional flexibility of use.

IXS defines 'generic' interfaces that would allow name-value pairs to be associated with an entity, similar to its fore-runner, the Person Identification Service (PIDS). In fact, for person related functionality, the service interfaces defined in this document could be implemented over an implementation of the PIDS module as shown below.



The PIDS module

Other emerging OMG Specifications that may be related to this specification are the Organization Structure Metamodel and the UML Profile and Metamodel for Services.

7 IXS Platform Independent Model

7.1 General

The Platform Independent Model (PIM) for IXS represents a wire and payload implementation independent means to describe the IXS interfaces.

The PIM is detailed in the sub-sections below containing a UML model and associated descriptions which depicts the structure of the interfaces provided by IXS. It is important to note that the structure of interfaces is specifically defined for the Service provider and reflects a natural structure of components that may provide the IXS functionality. These are effectively orthogonal to the groupings of capabilities defined in Functional Profiles that form the basis for conformance and for consumption of the service by Clients. In some cases, the functional profiles may approximate the interface structure.

7.2 Definitions

IXS embodies a number of key concepts that should be understood up front before attempting to read and understand this specification. Key terms are described below:

Term	Description
Semantic Signifier	This represents a schema definition and an associate set of validation instructions for the IXS Traits. It defines the structure and the means by which that structure should be validated. In XML for example, it might be an XSD and associated Schematron.
Logical Record	An instance of the semantic signifier. It could either be in document or message format depending on the implementation.
Real World Entity (RWE)	Represents the actual physical thing itself (e.g., the actual Person, the actual Device, etc.).
Entity	The software representation of a RWE - a record.
Domain	A set of values in which each is unique.
Source	A system generating an Entity.
Identifier (ID)	A value within a Domain that is associated with an object – an Entity, a Source, etc. - and uniquely identifies it within the scope of the Domain.
Entity ID	An Identifier associated with an Entity.
Source ID	An Identifier associated with a Source.
Identity	A single (Entity ID, Source ID) pair, or set of linked (Entity ID, Source ID) pairs.
IXS ID	An Identifier within the Domain of an IXS that is associated by the IXS with an Identity. It is not the same as an Entity ID, since it represents the entire set of linked Identities that represent a single Real World Entity.
Trait	A data element used by an IXS matching algorithm to link a particular (Source ID, Entity ID) pair with an existing Identity, based upon a conclusion that the Entities they are identifying represent the same RWE.

7.3 Theory of Solution

IXS primarily deals with the maintenance and use of data from the Entity class of the HL7 v3 RIM (Reference Information Model). It may optionally deal with the Role class of the RIM, but it does not store information related to Acts or Participations.

IXS is based upon the creation and maintenance of an index consisting of a linked set of Source ID/Entity ID pairs representing the same Real World Entity (RWE). Although the subject of this technical specification is a service, the acronym “IXS” is also used here to refer to the index itself and associated Traits. A Source ID and Entity ID are supplied in pairs in order that they may uniquely identify an Entity with *the Domain of the IXS*. (An Entity ID alone uniquely identifies an Entity *within the Domain of the Source*).

7.3.1 Cascades

IXSs may be cascaded. This is done through:

- the assignment of an IXS ID to each Identity that the IXS maintains, plus
- the existence of a Source ID associated with the IXS itself.

In this case, the Source ID representing the IXS and the IXS ID representing the Entity can be passed as parameters to another IXS, creating a cascade. This is also known as a parent-child relationship between IXSs. Cascades are referred to in the SFM as “XIS.”

IXSs may also be associated in a reciprocal relationship by registering their (Source ID, Entity ID) pairs with each other. This is also known as a peer-to-peer relationship between IXSs.

7.3.2 Register and Find

The bulk of the work of the IXS is performed by two operations, the register (RegisterEntityWithIdentity) operation and the find (FindIdentitiesByTraits) operation. In the register operation, a Source ID/Entity ID pair is supplied to the IXS for matching and linking with existing Identities. In the query operation, Identities are retrieved based upon matches of Identities or their associated Traits to query parameters.

The semantics of both the register and find operations depends largely upon the state of the IXS at the time the operation is invoked. The state in which the IXS was found is returned in the operation parameter of type IXSState. Details of operation semantics associated with different IXS states are discussed in the descriptions of the behavior of operations, below.

7.3.3 Duplicates

Each RWE should be associated with only one Identity by an IXS. A state in which two or more Identities may represent the same RWE indicates a possible duplicate. Unlink, link, and merge operations should be used to correct this situation.

8 IXSManagementAndQueryInterface

8.1 General

The IXS Management and Query Interface is modeled below.

Operation	Returns	Parameters
RegisterEntityWithIdentity	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **out ixsId : string(idl)**
CreateIdentityFromEntity	**IXSStatus**	**in sourceId : string(idl)** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **out entityId : string(idl)** **out ixsId : string(idl)**
UpdateEntityTraitValues	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in updateMethod : IXSUpdateMethod** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier**
RemoveIdentity	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier**
GetEntityTraitValues	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **out traits: IXSSemanticSignifier**
FindIdentitiesByTraits	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **In ixsId : string(idl)** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **in searchQualifier : IXSSearchQualifier** **out matchingResultSet : IXSResultSet**
ListLinkedIdentities	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in searchQualifier : IXSSearchQualifier** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **out matchingResultSet : IXSResultSet**
ListUnlinkedIdentities	**IXSStatus**	**in entityId: string(idl)** **in sourceId : string(idl)** **in searchQualifier : IXSSearchQualifier** **in semanticSignifierName : string(idl)** **in traits: IXSSemanticSignifier** **out matchingResultSet : IXSResultSet**

8.2 RegisterEntityWithIdentity

Behavior

This operation inserts a Source ID/Entity ID pair and supplied Traits into the IXS with implicit linking to other matching Source ID/ Entity ID pairs, based on the configured internal matching algorithm.

It is the normal operation for the insertion of new Entity data into the IXS.

A full discussion of the structure and content of IXSState (the OUT parameter) and IXSStatus (the Return value) can be found in Chapter 6 of this specification.

Parameters

??entityId: String (IN)

??sourceId : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

??ixsId : STRING (OUT)

??ixsState : IXSState (OUT)

Semantics of OUT Parameters

The following truth table defines all the possible state changes resulting from operations in an IXS that create or update Identities.

	IXS starting state (ixsState)					**IXS ending state**			**Semantics**
	Input matches single Identity				**Input matches multiple Identities**	**IXS ID**	**input Source ID/ input Entity ID**		**Meaning**
	IXS ID	**Input Source ID /input Entity ID**	**Input Source ID/ other Entity ID**	**Other Source ID/ any Entity ID**				**Duplicate Detected?**	
1	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**Record does not exist in the IXS currently. Identity is created with IXS ID. Input Source ID/ input Entity ID and input Traits are recorded in IXS.**

2	0	0	0	1	0	1	1	0	Indicates a match to an "unlinked" input record, such as one waiting for manual review. Put record in a queue for manual review.
3	0	0	1	0	0	1	1	1	
4	0	0	1	1	0	1	1	1	
5	0	1	0	0	0	1	1	0	
6	0	1	0	1	0	1	1	0	
7	0	1	1	0	0	1	1	1	
8	0	1	1	1	0	1	1	1	
9	1	0	0	0	0	1	1	0	This can happen if meta-traits are associated with IXS ID and the input matched the meta-traits but not any actual trait set from which the meta-traits were derived. Input Source ID/ input Entity ID is linked with IXS Identity.
10	1	0	0	1	0	1	1	0	Single match found. Input Source ID/ input Entity ID is linked with IXS Identity.
11	1	0	1	X	0	1	1	1	Single match found. Input Source ID/ input Entity ID is linked with IXS Identity.
12	1	1	0	X	0	1	1		Single match found. Traits are updated.
13	1	1	1	X	0	1	1	1	Single match found. Traits are updated.
14	X	X	0	X	1	1	1	0	Multiple matches found. Queue for manual review.
15	X	X	1	X	1	1	1	1	Multiple matches found. Queue for manual review.

Return

???IXSStatus

Status Conditions

code	severity	Message
1002	**Info**	**Entity has been successfully registered with IXS ID ‘%1’ in the IXS.**
2023	**Error**	**The Semantic Signifier identified by ‘%1’ does not exist in the IXS.**
2001	**Error**	**The mandatory Trait ‘%1’ for the Entity Type ‘%2’ was not provided.**
2002	**Error**	**The Trait identified by ‘%1’ does not exist in the IXS.**
2003	**Error**	**The Identifying Trait list provided contains a duplicate Trait ‘%1.’**
2004	**Error**	**An invalid value was passed for the Trait ‘%1.’**
2005	**Error**	**An entity with that ID already exists.**
3001	**Warning**	**The Entity referenced by the ID information provided may already exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS.**

8.3 CreateIdentityFromEntity

Behavior

This operation generates an Entity ID on behalf of a Source that does not generate Entity IDs. It inserts the Source ID/ newly-generated Entity ID pair and supplied Traits into the IXS with implicit linking to other matching Source ID/ Entity ID pairs, based on the configured internal matching algorithm.

This operation may be used in environments where the RegisterEntityWithIdentity cannot be used because the Source does not assign Identifiers to Entities.

Parameters

???systemIdSourceId : SYSTEMID (IN)String (IN)

???semanticSignifierName : String (IN)

???traits: IXSSemanticSignifier (IN)

??entityId : String (OUT)

??ixsState : IXSState (OUT)

Return

??IXSStatus

Status Conditions

code	severity	Message
1001	Info	An Entity ID was successfully created in the IXS.
1002	Info	Entity has been successfully registered with IXS ID ‘%1’ in the IXS.
2001	Error	The mandatory Trait ‘%1’ for the Entity Type ‘%2’ was not provided.
2002	Error	The Trait identified by ‘%1’ does not exist in the IXS.
2003	Error	The Identifying Trait list provided contains a duplicate Trait ‘%1.’
2004	Error	An invalid value was passed for Trait ‘%1.’
2023	Error	The Semantic Signifier identified by ‘%1’ does not exist in the IXS.
3001	Warning	The Entity referenced by the ID information provided may already exist in the IXS.
2977	Error	The Domain identified by SourceId ‘%1’ does not exist in the IXS.

8.4 UpdateEntityTraitValues

Behavior

This operation updates the Traits stored in the IXS for the Entity identified by the supplied Source ID/Entity ID pair. Whether this entails re-analyzing matching identities is implementation specific, although it would be done in most cases. How it would be done is also implementation specific.

Parameters

???entityId: String (IN)

???sourceId : String (IN)

??ixsUpdateQualifier : IXSUpdateQualifier(IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

??ixsState : IXSState (OUT)

Return

??IXSStatus

Status Conditions

code	severity	Message
1003	Info	Entity Trait Values have been successfully updated in the IXS.
2023	Error	The Semantic Signifier identified by ‘%1’ does not exist in the IXS.
2002	Error	The Trait identified by ‘%1’ does not exist in the IXS.
2003	Error	The Identifying Trait list provided contains 1 or more duplicates.
2004	Error	An invalid value was passed for the Trait ‘%1.’
2005	Error	Attempt to delete 1 or more mandatory Traits for the Entity Type ‘%1.’

2006	**Error**	**1 or more Traits passed in the list are not valid or do not exist for Entity Type '%1.'**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

8.5 RemoveIdentity

Behavior

This function deletes a Source ID/Entity ID pair and its associated Traits from the IXS repository. This operation is designed to correct a true invalid data condition, such as the data entry of practice or testing data.

All of the operations in the IXSAdminInterface are the normal operations to deal with changes and corrections to otherwise valid data.

Parameters

??entityId: String (IN)

??sourceId : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1004	**Info**	**Entity reference has been successfully removed in the IXS.**
2007	**Error**	**Entity Referenced does not exist in the IXS.**
3002	**Error**	**Wrong existing state for deletion.**
2100	**Error**	**Identity linked to other identities.**
2023	**Error**	**The Semantic Signifier identified by '%1' does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

8.6 GetEntityTraitValues

Behavior

Retrieves the Traits associated with a Source ID/Entity ID pair.

Parameters

???entityId: String (IN)

??sourceId : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1005	**Info**	**Entity Trait values retrieved successfully.**
2023	**Error**	**The Semantic Signifier identified by '%1' does not exist in the IXS.**
3001	**Error**	**Entity Referenced does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

8.7 FindIdentitiesByTraits

Behavior

This operation provides the means to perform a broad search of all records in the IXS whose Traits match some criteria in the supplied search criteria (such as find all records who match the name "Jones, Bob").

This operation is equivalent to the IHE PDQ transaction.

Parameters

?entityId: String (IN)

?sourceId : String (IN)

?ixsId : String (IN)

?searchQualifier : IXSSearchQualifier (IN)

?semanticSignifierName : String (IN)

?traits: IXSSemanticSignifier (IN)

??matchingResultSet : IXSResultSet (OUT)

??ixsState : IXSState (OUT)

Semantics of OUT Parameters

	What is matched				**semantics**			
	Input matches single IXS ID			**Input matches additional IXS ID**	**meaning**	**ID returned**	**Other IDs returned**	**(demographic) traits returned**
	input IXS ID matches IXS ID	**input source ID / input Entity ID matches IXS Identity**	**traits match those linked with any Identity**					

1	1	X	X	0	IXS ID was submitted & match is found	depends upon request - can return this IXS ID	depends upon request - can return any or all other IDs	depends upon request - can return any or all traits, at different (parent-child) levels
2	X	1	X	0	source ID/Entity ID was submitted & match is found			
3	X	X	1	X	Traits were submitted & match is found			
4	0	0	0	0	no matches	nothing returned	nothing returned	nothing returned
5	1	X	0	1	error			
6	X	1	0	1	error			

Return

??IXSStatus

Status Conditions

code	severity	message
1006	Info	'%1' Entities successfully retrieved.
2023	Error	The Semantic Signifier identified by '%1' does not exist in the IXS.
2002	Error	The Trait identified by '%1' does not exist in the IXS.
2007	Error	Entity Referenced does not exist in the IXS.
2977	Error	The Domain identified by SourceId '%1' does not exist in the IXS.

8.8 ListLinkedIdentities

Behaviors

This operation retrieves all the Source ID/Entity ID pairs that are linked to the supplied Source ID/Entity ID pair. The operation can be filtered with the sourceConstraintSet property of the IXSSearchQualifier to only return entities within specified Source domains.

This operation is equivalent to the IHE PIX transaction.

Parameters

??entityId: String (IN)

??sourceId : String (IN)

??searchQualifier : IXSSearchQualifier (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

??matchingResultSet : IXSResultSet (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1007	**Info**	**‘%1’ Linked Entities successfully retrieved.**
2023	**Error**	**The Semantic Signifier identified by ‘%1’ does not exist in the IXS.**
2007	**Error**	**Entity Referenced does not exist in the IXS.**
2002	**Error**	**The Trait identified by ‘%1’ does not exist in the IXS.**
2011	**Error**	**Identity Qualifier ‘%1’ does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS.**

8.9 ListUnlinkedIdentities

Behavior

This operation retrieves duplicately specified, yet unlinked records in the IXS repository. It filters the record set to only return matching, but unlinked records. The operation can be filtered with sourceConstraintSet to only return entities within specified Source domains.

Parameters

??entityId: String (IN)

??sourceId : String (IN)

??searchQualifier : IXSSearchQualifier (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

??matchingResultSet : IXSResultSet (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1008	**Info**	**‘%1’ UnLinked Entities successfully retrieved.**
2023	**Error**	**The Semantic Signifier identified by ‘%1’ does not exist in the IXS.**

2007	**Error**	**Entity Referenced does not exist in the IXS.**
2002	**Error**	**The Trait identified by ‘%1’ does not exist in the IXS.**
2011	**Error**	**Identity Qualifier ‘%1’ does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS.**

9 IXSAdminEditorInterface

9.1 General

The IXS Administration Editor Interface is summarized in the model below.

Operation	Returns	Parameters
LinkEntities	**IXSStatus**	**in sourceEntityId : string(idl)** **in sourceSourceId : string(idl)** **in targetEntityId : string(idl)** **in targetSourceId : string(idl** **in reasonCode : string(idl)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**
UnlinkEntities	**IXSStatus**	**in sourceEntityId : string(idl)** **in sourceSourceId : string(idl)** **in targetEntityId : string(idl)** **in targetSourceId : string(idl)** **in reasonCode : string(idl)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**
MergeEntities	**IXSStatus**	**in sourceEntityId : string(idl)** **in targetEntityId : string(idl)** **in targetSourceId : string(idl)** **in reasonCode : string(idl)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**
UnMergeEntities	**IXSStatus**	**in sourceEntityId : string(idl)** **in targetEntityId : string(idl)** **in targetSourceId : string(idl)** **in reasonCode : string(idl)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**
ActivateEntity	**IXSStatus**	**in entityId : string(idl)** **in sourceId : string(idl)** **in reasonCode : string(id)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**
DeactivateEntity	**IXSStatus**	**in entityId : string(idl)** **in sourceId : string(idl)** **in reasonCode : string(id)** **in semanticSignifierName: string(idl)** **in traits : IXSSemanticSignifier**

9.2 LinkEntities

Behavior

This operation provides the means to create an explicit linking between the source and target record IDs in the IXS repository.

Parameters

??sourceEntityId : String (IN)

??sourceSourceId : String (IN)

??targetEntityId : String (IN)

??targetSourceId : String (IN)

???reasonCode : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

???IXSStatus

Status Conditions

code	severity	message
1010	**Info**	**Entities successfully linked**
2020	**Error**	**Source Entity Referenced does not exist in the IXS**
2021	**Error**	**Target Entity Referenced does not exist in the IXS**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS**
2023	**Error**	**The Entity Type identified by ‘%1’ does not exist in the IXS**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS**

9.3 UnlinkEntities

Behavior

This operation provides the means for explicitly breaking the link between the source and target record IDs in the IXS repository.

Parameters

??sourceEntityId : String (IN)

???sourceSourceId : String (IN)

??targetEntityId : String (IN)

??targetSourceId : String (IN)

??reasonCode : String (IN)

??semanticSignifierName : String (IN)

???traits: IXSSemanticSignifier (IN)

Return

???IXSStatus

Status Conditions

code	severity	message
1011	**Info**	**Entities successfully unlinked.**
2020	**Error**	**Source Entity Referenced does not exist in the IXS.**
2021	**Error**	**Target Entity Referenced does not exist in the IXS.**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS.**
2023	**Error**	**The SemanticSignifier identified by '%1' does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

9.4 MergeEntities

Behavior

This operation provides the means to explicitly consolidate IXS member records in the IXS repository. The source record is effectively merged into the target, leaving only the resulting target record in the IXS repository. Identifying attributes in the target that are empty are filled from the source, and existing attributes in the target remain AS-IS. The source record is inactivated in the IXS repository at the successful end of this operation.

Parameters

??sourceEntityId : String (IN)

??sourceSourceId : String (IN)

??targetEntityId : String (IN)

?targetSourceId : String (IN)

??reasonCode : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1012	**Info**	**Entities successfully merged.**
2020	**Error**	**Source Entity Referenced does not exist in the IXS.**

2021	**Error**	**Target Entity Referenced does not exist in the IXS.**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS.**
2023	**Error**	**The SemanticSignifier identified by '%1' does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

9.5 UnMergeEntities

Behavior

Parameters

??sourceEntityId : String (IN)

??sourceSourceId : String (IN)

?targetEntityId : String (IN)

?targetSourceId : String (IN)

?reasonCode : String (IN)

?semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1013	**Info**	**Entities successfully unmerged.**
2020	**Error**	**Source Entity Referenced does not exist in the IXS.**
2021	**Error**	**Target Entity Referenced does not exist in the IXS.**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS.**
2023	**Error**	**The SemanticSignifier identified by '%1' does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId '%1' does not exist in the IXS.**

9.6 ActivateEntity

Behavior

This operation marks the record as "active" in the IXS repository, effectively marking it as logically un-deleted.

Parameters

??entityId : String (IN)

??sourceId : String (IN)

??reasonCode : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1014	**Info**	**Entity successfully activated.**
2007	**Error**	**Entity Referenced does not exist in the IXS.**
2002	**Error**	**The Trait identified by ‘%1’ does not exist in the IXS.**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS.**
2023	**Error**	**The SemanticSignifier identified by ‘%1’ does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS.**

9.7 DeactivateEntity

Behavior

This operation marks the record as “inactive” in the IXS repository, effectively marking it available for Get() or List() operations at the entity level but not from the IXS repository.

Parameters

???entityId : String (IN)

??sourceId : String (IN)

??reasonCode : String (IN)

??semanticSignifierName : String (IN)

??traits: IXSSemanticSignifier (IN)

Return

??IXSStatus

Status Conditions

Code	severity	message
1015	**Info**	**Entity successfully deactivated.**
2002	**Error**	**The Trait identified by ‘%1’ does not exist in the IXS.**
2007	**Error**	**Entity Referenced does not exist in the IXS.**
2022	**Error**	**Reason Code is not in a valid format or does not exist in the IXS.**
2023	**Error**	**The SemanticSignifier identified by ‘%1’ does not exist in the IXS.**
2977	**Error**	**The Domain identified by SourceId ‘%1’ does not exist in the IXS.**

10 IXSMetaDataInterface

10.1 General

The IXS Meta Data Interface is summarized in the model below.

Operation	Returns	Parameters
CreateSemanticSignifier	**IXSStatus**	**in semanticSignifierDefinition : IXSSemanticSignifierDefinition**
FindSemanticSignifier	**IXSStatus**	**in semanticSignifierName : string(idl)** **out semanticSignifierDefinition : IXSSemanticSignifierDefinition**
UpdateSemanticSignifier	**IXSStatus**	**inout semanticSignifierDefinition : IXSSemanticSignifierDefinition**
ListSemanticSignifiers	**IXSStatus**	**In entityTypeName : string(idl)** **out listOfSemanticSignifierNames : sequence(idl)**
ListDomains	**IXSStatus**	**out listOfDomains : sequence(idl)**
ListConformanceProfiles	**IXSStatus**	**out conformanceProfiles : arrayOf[string(idl)]**

10.2 CreateSemanticSignifier

Behavior

Creates a semantic signifier definition in the IXS meta-data repository.

Parameters

??semanticSignifierDefinition: IXSSemanticSignifierDefinition (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1032	**Info**	**SemanticSignifier successfully created.**
2023	**Error**	**The Semantic Signifier identified by '%1' does not exist in the IXS.** **(This condition is optional and based on if the SemanticSignifier is passed.)**
2022	**Error**	**A SemanticSignifier identified by '%1' already exists in the IXS.**

10.3 FindSemanticSignifier

Behavior

Locates and returns a semantic signifier definition from the IXS meta-data repository.

Parameters

??semanticSignifierName : String (IN)

??semanticSignifierDefinition : IXSSemanticSignifierDefinition (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1033	Info	Semantic Signifier successfully retrieved.
2023	Error	A Semantic Signifier identified by '%1' does not exist in the IXS.

10.4 UpdateSemanticSignifier

Behavior

Updates a semantic signifier definition in the IXS meta-data repository.

Parameters

??semanticSignifierName : String (IN)

??semanticSignifierDefinition: IXSSemanticSignifierDefinition (IN)

Return

??IXSStatus

Status Conditions

code	severity	message
1034	Info	SemanticSignifier successfully updated.
2023	Error	The Semantic Signifier identified by '%1' does not exist in the IXS. (This condition is optional and based on whether the semanticSignifier is passed.)

10.5 ListSemanticSignifiers

Behavior

List the traits registered with the system.

Parameters

??entityTypeName : String (IN)

??listOfSemanticSignifiers : List<String> (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1035	Info	SemanticSignifiers successfully listed.
2023	Error	The Semantic Signifier identified by '%1' does not exist in the IXS. (This condition is optional and based on whether the semanticSignifier is passed.)

10.6 ListDomains

Behavior

List the domain hierarchy configured in the IXS meta-data repository.

Parameters

??listOfDomains : List<String> (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1024	Info	Domains successfully listed.

10.7 ListConformanceProfiles

Behavior

Returns the list of conformance profile names and versions that this implementation of IXS supports.

Parameters

??listOfConformanceProfiles: arrayOf[ConformanceProfile (OUT)

Return

??IXSStatus

Status Conditions

code	severity	message
1025	Info	Conformance Profiles successfully listed.

11 IXS Data Structures

11.1 Complex Data Structures

11.1.1 IXSSemanticSignifierDefinition

11.1.1.1 Overview

Meta-data structure for definition of a data type processed by IXS.

11.1.1.2 Structure

An Instance of IXSSemanticSignifierDefinition has the following properties:

- semanticSignifierName : String – The label for the SemanticSignifier.
- identifier : string(idl) – A Globally unique alphanumeric ID or GUID that uniquely identifies a SemanticSignifier.
- versionIdentifier : String – A Version label for the SemanticSignifier.
- entityTypeName : String – The EntityName that the SemanticSignifier represents. Also serves to categorize SemanticSignifiers by *type*.
- schemaDefinition : String – A URI to an XSD or other meta-document that describes the definition of the SemanticSignifier schema. A given instance of a SemanticSignifier may be optionally validated against the document pointed to by this URI.
- renderingDefinition : String – A URI to an XSL or other suitable resource (e.g., a Java Class) that can be utilized to extract Traits from an instance of this SemanticSignifier.
- integrityRulesetDefinition : String – A URI to a Schematron or WSDL reference that can serve to make integrity assertions against a given instance of a SemanticSignifier.

11.1.2 IXSSemanticSignifier

11.1.2.1 Overview

Instance of the Schema Definition in IXSSemanticSignifierDefinition.

11.1.2.2 Structure

An Instance of IXSSemanticSignifier has the following single element:

- ??document : String – The document or source (e.g., raw XML, HTML, structured text, etc.) of an instance of a SemanticSignifier.

11.1.3 IXSSearchQualifier

11.1.3.1 Overview

This data structure contains optional search qualifiers that can be used by the Find operation to more finely control the behavior of the search algorithm. It is an optional parameter and if not supplied, the internal configuration of the IXS engine will dictate all search behavior.

11.1.3.2 Structure

An instance of IXSSearchQualifier is a complex data structure with the following elements:

- ??matchingModuleName : String – The name of the IXS Matching algorithm name to be used. If this value is blank, the default configured for the IXS will be used.
- ??searchConfidenceLevel : String – The search confidence level to be used for matching (0..1, or % threshold). For search operations, this value defaults to blank which indicates to the IXS that the default value configured for the IXS is to be used.
- ??maxResultSetSize : number – The maximum number of results to be returned from the given API call. This value defaults to 0 which indicates that the IXS may arbitrarily set the limit to some configured value.
- ??raiseErrorIfLimitReached : Boolean – A flag that indicates that if the value set in the maxResultSetSize property is reached, an Error StatusCondition is to be returned, indicating to the caller that the threshold they set was reached. This value defaults to *false*.
- ??sourceConstraintSet: List<String> – A list of SourceId strings that serve to constrain the results of a Search or List operation to a set of existing identity domains.

11.1.4 IXSUpdateQualifier

11.1.4.1 Overview

The IXSUpdateQualifier is meant to contain elements that are used to limit, constrain, or otherwise alter the behavior of the API update operation involved.

11.1.4.2 Structure

The IXSUpdateQualifier structure consists of the following elements:

- ??updateMode : enum – An enumeration indicating a mode to be utilized during the update process. Values include:
  - OVERWRITE (0) – Replaces the entire set of data elements with the entire supplied set. Null values in the submitted data will replace existing non-null values.
  - VALUED (1) – Replaces only those data elements that are valued in the input data. Null values in the submitted data will not replace existing non-null values.
  - UNSPECIFIED (2) – Implementation specific.
- ??updateSchemaDefinition : String – A URI to an XSL or other suitable resource (e.g., a Java Class) that can be utilized to extract Traits from an instance of this SemanticSignifier.

11.1.5 IXSStatus

11.1.5.1 Overview

All operations executed via the IXS PIM API have an OUTPUT parameter of type IXSStatus. Upon return from an API operation, the returned IXSStatus structure will contain information that indicates the outcome (success or failure) of the preceding API operation.

11.1.5.2 Structure

The IXSStatus structure consists of the following attributes:

- ???statusSuccess : Boolean – A flag representing the success or failure of the API operation. True indicates that the operation was successful and false indicates a non-successful operation.
- ??statusConditions : Array<StatusCondition> – A set of the StatusCondition structure representing all the conditions that occurred during the API call.

11.1.5.3 StatusCondition component

- ??statusCode : String – A 4 digit numeric code representation of the outcome of the API operation.
- ??statusSeverity : enum – An enumeration that indicates the severity of the status-code returned. Possible values are:
  - Informational (0) – Used to relay to the client useful information about the outcome of the call. Informational messages are usually indications of successful operations, but other useful purposes do exist.
  - Warning (1) – Operation completed, but some portion did not complete as expected.
  - Error (2) – Operation was unable to complete for identifiable data or software problems.
  - Severe (3) – Operation was unable to complete for system problems, such as “resource allocation failure,” or “hardware failure,” or unidentifiable data or software problems.
- ??statusMessage: String – An alpha-numeric message describing the outcome of the API operation.
- ??statusDetail : String – A detailed message describing the outcome of the API operation. This element is optional.

11.1.5.4 StatusCode component

IXSStatus StatusCode instances have two dimensions. First, each StatusCode is unique and represents a single specific condition that may occur within the IXS. Second, StatusCodes are numeric and their range will indicate a class of conditions that has occurred. For example, status-codes in the range of 1000-1999 indicate a successful operation.

Since there will be a number of implementers of the IXS specification, there must be the capability for implementers to indication server conditions that are specific to a given implementation.

Range	Condition Class	Example
1000-1999	**Successful operations**	**1032 - A SemanticSignifier was successfully created in IXS** **(In the case where an implementer is opting to standardize on using a single message to indicate a successful operation, the status-code of 1000 is to be used and the text of the message should be standardized as per the IXS spec. “Operation Succeeded,” for instance).**
2000-2999	**Client side or parameter specific conditions.**	**2002 - The Trait identified by ‘trait_ssn’ does not exist in the IXS.**
3000-3999	**Service side conditions not related to parameter data.**	**3001 - The Entity referenced by the provided information may already exist in the IXS.**
4000-4999	**Implementer specific error conditions not covered by specification covered errors.**	**4001 – The specified query hint ‘use-fuzzy' is not a valid hint.**

11.1.6 IXSState

11.1.6.1 Overview

Certain operations executed via the IXS PIM API have an OUTPUT parameter of type IXSState. Upon return from an API operation that registers or queries an Entity, this structure will contain detailed information about certain starting (or pre-operation) and ending (or post-operation) states of the IXS with respect to the identifiers of the Entity passed in the API operation.

Each attribute of the IXSState structure, referred to individually as *sub-states*, may take on one of three values: Exists (1), Doesn’t Exist (0), May or may not exist (-1).

11.1.6.2 Structure

The IXSState structure consists of the following elements each a *flag* or *sub-state* indicator:

- ??ixsMediatingIdStartState : Boolean
- ???inputSourceIdStartState : Boolean
- ??inputSourceOtherIdStartState : Boolean
- ??otherSourceAnyIdStartState : Boolean
- ??ixsMediatingIdEndState : Boolean
- ??inputSourceIdEndState : Boolean
- ??duplicateDetected : Boolean

11.1.7 IXSResultSet

11.1.7.1 Overview

A structure that contains the results of a Search or Find operation. IXSResultSet consists of a list of the IXSResultSetElement structure. Each instance of IXSResultSetElement contains the information about an Identity, and in addition, information with respect matching.

11.1.7.2 Structure

The IXSResultSet structure consists of the following element:

- ??ixsResultSetElements : List<IXSResultSetElement>

11.1.7.3 IXSResultSetElement structure

??entityId : String – The entityId of the Identity.

??sourceId : String – The sourceId of the Identity.

??ixsId : String – The identifier used by the IXS to correlate between identities.

??semanticSignifierName : String – The name of the SemanticSignifier that describes the content of the IXSSemanticSignifier instance of the *traits* element below:

- ??traits : IXSSemanticSignifier – An instance of a SemanticSignifier containing the Traits for the identity.

11.2 Other Simple Data Types

IXS Data Type	Description
EntityTypeName	**This is a string representing an enumeration of Entity Types supported by the IXS. A simple example would be "PERSON," but other entities such as "BEDS" or "SPECIMENS" could also be supported within a single IXS.**
Identifiers	**All identifiers referenced are intended to be GUIDs, including Source ID, Entity ID, and IXS Entity ID.**
ConformanceProfiles	**As defined by relevant standards**
IXSUpdateMethod	**Enumeration Method…**

11.3 Supplemental information on IXS Error Handling

??All IXS API calls will return the specified status object that represents the outcome (success/failure, code, message, etc.) of what occurred during the API call.

??All IXSAPI calls will return an instance of an IXSStatus structure. Example method signature: *IXSStatus API_call(param1, param2, … paramN).*

?The IXSStatus structure will indicate the success or failure of the API call.

?The IXSStatus structure will return a set of StatusCondition structures in a property named *statusConditions* that summarize in greater detail the success or failure of the API call. Each instance of the StatusCondition structure will contain information pertaining to an individual condition encountered during the API call.

??For successful API calls, while there will typically be a single StatusCondition instance returned in the *statusConditions* property of the IXSStatus structure.

??For successful API calls, implementers may choose to return a single generic INFO level StatusCondition that indicates a successful operation (e.g., *OPERATION_SUCCESSFUL*) or they may utilize the provided API call specific INFO level messages.

??For unsuccessful API calls, implementers must return at least one instance of StatusCondition corresponding one of the total set of conditions encountered during the call. The implementer may also choose to return (n) StatusCondition instances that fully communicate all the individual conditions encountered during the call.

??Implementers choosing to implement a single StatusCondition for each API call should give priority to *Errors* over the *Info* and *Warning* severities. However, this choice is up to the implementer. They may choose to simply return the first error encountered by whatever error checking algorithm they implement.

??Each StatusCondition will have a *statusSeverity* enumeration property. The severity enumeration will consist of INFO, WARN, ERROR, and SEVERE. INFO is considered to have the lowest severity and SEVERE to have the highest.

??Error conditions are to be ordered by *statusSeverity* and within severity from generic to specific. Example: An IXS CreateIdentityFromEntity operation:

- Condition 0 - Severe: storage error occurred.
- Condition 1 - Error: invalid SemanticSignifier value (specific).
- Condition 2 - Error: missing one or more required SemanticSignifier fields (generic).

??There is no inherent mechanism provided within the array of StatusCondition instances returned within the IXSStatus structure to relate any given StatusCondition instance to another. Implementers may choose to imply or hint at these types of relationships via the *statusDetail* property of the StatusCondition structure.

??IXS implementers can optionally provide localized *StatusMessage* values and in the absence of a locale, return messages in the default locale (language) for that server instance.

12 Semantic Profiles

12.1 General

This is a normative reference of implementing the IXS service interface using either HL7 v3 RIM-based structures or HL7 v2.x structures:

- a) A Person IXS in HL7 v2.x
- b) A Person IXS in HL7 v2.x using the IHE Pediatric Demographic Option

NOTE: All machine readable files are at: http://www.omg.org/spec/IXS/20101101

12.2 Profile Alignment by Service Interface and Operations

What this profile does is line up the platform independent model to create a specific service interface that supports the various IXS operations but does so by creating a specific WSDL for each data type (Provider, Patient, etc.) and then types the semantic signifier payload to the specific operations.

This could also be done more generically, with an IXS-HSSP interface with only generic signifiers to represent the data payloads and then the caller could dynamically determine the required schema definition by inspecting it at run-time via a Describe() operation. This is similar to using an *any or variant* data type instead of a specific structure to represent the data payload.

The method used below is more type-safe and has some advantages when using modern web service development tools since the more type explicit approach allows the development tools to create object wrappers around the explicit data types, where the generic approach requires the developer to write code against the XML text format of the semantic signifier directly.

12.3 Implementations and Semantic Signifiers

The IXS interface in WSDL, as shown in the next section, can process any of the following (in document or message form) that can be normalized in an XSD with an optionally associated Schematron:

- Care Record
- CDA
- Patient Administration data

The key differences in supporting another semantic signifier type include:

- a) the URI XSD which is held in the `IXStypes:IXSSemanticSignifier` meta-data structure that describes the structure of the data to be handled by the interface.
- b) The `<xs:element ref="cda:ClinicalDocument" minOccurs="1">` element in the Get(), Put(), List() parameter that corresponds to the traits(s) parameter in the platform independent model.

So, for example to support the care record message or an MS-Lab content profile, an IXS service would:

- Have its metadata registry configured to point to a URI of an appropriate defined XSD to meet requirement (a) and then
- a specific WSDL would be defined incorporating that XSD as a specific type of the traits(s) parameter in each applicable SOAP request and response satisfying the requirement listed in (b) above.

This approach allows for a flexible and loosely coupled deployment of an IXS service that supports many different kinds of data types.

12.4 Implementations

The necessary WSDLs for supporting IXS are included as separate machine-readable documents for the example shown below.

12.4.1 Implementation Example – HL7 Version 2.5 Person using IHE Pediatric Demographic option

12.4.1.1 Definition of Traits using HL7 V2.5 Segments

A mapping of Entity Traits to HL7 V2.5 fields is given in Table 12.1 and Table 12.2. The SFM included an initial Semantic Profile which identified a list of traits, modeled after those required by IHE PDQ Query, and allows for profiles to define additional traits. This profile defines additional traits as shown below.

Table 12.1 - Traits for Patient Feed

FLD	ELEMENT NAME
Traits (as defined in SFM)	
PID.3	Patient Identifier List
PID.5	Patient Name
PID.7	Date/Time of Birth
PID.8	Administrative Sex
PID.11	Patient Address
Additional Traits	
PID.6	Mother’s Maiden Name
PID.13	Patient Home Telephone Number
PID.24	Patient Multiple Birth Indicator
PID.25	Patient Birth Order
PID.33	Last Update Date / Time
PID.34	Last Update Facility

Table 12.2 - Traits Used by Query

FLD	ELEMENT NAME
Traits (as defined in SFM)	
PID.3	Patient Identifier List
PID.5	Patient Name
PID.7	Date/Time of Birth
PID.8	Administrative Sex
PID.11	Patient Address
PID.18	Patient Account Number
Additional Traits	
PID.6	Mother’s Maiden Name
PID.13	Patient Home Telephone Number

See the IHE discussions in IHE IT Infrastructure Technical Framework, vol. 2 (ITI TF-2): Transactions, pp. 199-213, Transaction “ITI-21” for the Patient Demographics Query (PDQ).

Note that the Patient Identifier List PID.3 will support the identifiers of Table 12.3. Also note that these identifiers are a CX data type, and the issuing authority must be included, particularly for identifiers that don’t imply one particular issuing authority. In this Implementation Example, OIDs are used to identify the issuing authority as specified by IHE standards.

Table 12.3 - Identifiers Supported by PID 3

ID Type (from table User Defined 0203)	Description
SS	Social Security Number
MA	Medicaid Number
MR	Medical Record Number
BR	Birth Registry Number
LR	Local Registry Number
RRI	Regional Registry Number
SR	State Registry Number
WC	WIC ID
XX	Organization Identifier

The list of patients returned by queries will be ordered, in descending sequence, by whatever means is available to the query fulfiller, so that the most likely match is shown first.

12.4.1.2 IXSManagementAndQueryInterface

12.4.1.2.1 RegisterEntityWithIdentity

The RegisterEntityWithIdentity operation uses parameters structured as an HL7 ADTA28 message with ACK response.

It has the same message structure as CreateIdentityFromEntity (see below), but requires that the Entity ID be furnished.

See The IHE discussions in IHE IT Infrastructure Technical Framework, vol. 2 (ITI TF-2): Transactions, pp. 41-50 about Transaction “ITI-30” which can be found at www.ihe.net for full details.

Also See HL7 discussion of the ADTA28 message on page 3-36.

ADTA28 Segments Needed:

- ??MSH
- ??[SFT] – used to report software vendor ID & version
- ???EVN
- ???PID
- ??[PD1] – for Publicity Code, Protection Indicator and Effective Date, Immunization Registry Status and Date
- ???[{NK1}] – repeats for mother / father & related info, as required
- ???PV1 – required for Patient Class, Point of Care, Primary Provider
- ???PV2 – for Patient Status

	PIM	**PSM**
Inputs	**✍✍SourceID** **✍✍EntityID.** **✍✍Unordered set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	**✍✍This is mapped to MSH.4 in HL7 v2.5.** **✍✍This is mapped to PID.3, first element in the list.** **✍✍Use a standard HL7 v2.5 ADT message to hold the traits.**
Outputs	**✍✍Acknowledgement**	**✍✍HL7 ACK.mapped into IXSStatus**

Error Conditions	**✍Entity reference already exists (error based on ID match)** **✍Entity reference may already exist (warning) – returns ID(s) and set of traits for matched identity** **✍Mandatory traits not supplied** **✍A trait in the list does not exist in the system metadata for the Entity Type** **✍✍There are duplicates in the trait list** **✍✍Invalid data type or code value for trait**	**Error Conditions will be reported in an ERR segment in the ACK, using definitions from the PIM above. Status condition codes returned will only support Error, Warn, and Info.** **All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.2.2 CreateIdentityFromEntity

The CreateIdentityFromEntity operation uses parameters structured as an HL7 ADTA28 message with ACK response.

It has the same message structure as RegisterEntityWithIdentity (see above), but requires that the IXS generate and return an Entity ID for the record.

See the IHE discussions in IHE IT Infrastructure Technical Framework, vol. 2 (ITI TF-2): Transactions, pp. 41-50 about Transaction “ITI-30” that can be found at www.ihe.net for full details on the PIX Query.

Also See HL7 discussion of the ADTA28 message on page 3-36.

ADTA28 Segments Needed:

- ??MSH
- ??SFT] – used to report software vendor ID & version
- ??EVN
- ??PID
- ??[PD1] – for Publicity Code, Protection Indicator and Effective Date, Immunization Registry Status and Date
- ??[{NK1}] – repeats for mother / father & related info, as required
- ??PV1 – required for Patient Class, Point of Care, Primary Provider
- ??PV2 – for Patient Status

	PIM	**PSM**
Inputs	**✍Source ID (Entity ID not submitted).** **✍Unordered set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	**✍This is mapped to MSH.4** **✍use a standard HL7 v2.5 ADT message to hold the traits in segments shown above.**
Outputs	**✍Entity ID**	**✍Entity ID and ACK mapped into IXSStatus**

Error Conditions	✍**Entity reference may already exist (warning) – returns ID(s) and set of traits for matched identity.** ✍**Mandatory traits not supplied.** ✍**A trait in the list does not exist in the system metadata for the Entity Type.** ✍**There are duplicates in the trait list.** ✍**Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the ACK, using definitions from the PIM above. Status condition codes returned will only support Error, Warn, and Info.** **All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.2.3 UpdateEntityTraitValues

The UpdateEntityTraitValues operation uses parameters structured as an HL7 ADTA31 message with ACK response.

Also See HL7 discussion of the ADTA31 message on page 3-40.

ADTA31 Segments Needed:

- ??MSH
- ??[SFT] – used to report software vendor ID & version
- ???EVN
- ???PID
- ??[PD1] – for Publicity Code, Protection Indicator and Effective Date, Immunization Registry Status and Date
- ??[{NK1}] – repeats for mother / father & related info, as required
- ??PV1 – for Patient Class, Primary Point of Care, Primary Provider

	PIM	PSM
Inputs	✍**Source ID** ✍**Entity ID** ✍**Unordered set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	✍**This is mapped to MSH.4 in HL7 v2.5.** ✍**This is mapped to PID.3, first element in the list.** ✍**Use a standard HL7 v2.5 ADT message to hold the traits in segments shown above.**
Outputs	✍**Acknowledgement**	✍**HL7 ACK**

Error Conditions	✍**Entity reference does not exist (error based on ID match).** ✍**Mandatory traits deleted.** ✍**A trait in the list does not exist in the system metadata for the Entity Type.** ✍**There are duplicates in the trait list.** ✍**Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the RSP, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.** **All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.2.4 RemoveIdentity

The RemoveIdentity operation uses parameters structured as an HL7 v2.5 ADTA29 message with ACK response. This operation will set a flag to remove it from all future user queries.

This operation is permanent.

This operation is intended for administrator use only. The identifying information in the PID segment must resolve to a single person Identity record that is not already flagged as "Removed," or this operation must fail.

See HL7 discussion on page 3-38.

ADTA29 Segments Needed:

- ?*MSH*
- ??[SFT] – used to report software vendor ID & version
- ??EVN
- ??PID
- ??PV1 – required by standard, but ignored here

	PIM	PSM
Inputs	✍**Source ID Entity ID**	✍**mapped to MSH.4 in HL7 v2.5.** ✍**It is mapped to PID.3, first element in the list.**
Outputs	✍**An acknowledgement that the Entity has been removed.**	✍**HL7 ACK message.**
Error Conditions	✍**Entity reference does not exist.** ✍**Wrong existing state for deletion.** ✍**Identity linked to other identities.**	✍**Error Conditions will be reported in an ERR segment in the ACK, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.**
Additional Details		**In this PSM, the Entity Identifier cannot be reused. This implementation will remove records by flagging them as removed.**

12.4.1.2.5 GetEntityTraitValues

The GetEntityTraitValues operation uses parameters structured as an HL7 QBPQ22 message with RSPK21 response. It returns a list of linked identifiers of patient records from all other domains that are linked to the queried information about one particular patient in a specified domain.

The parameters for Source ID and Entity ID are required for this operation. This operation will only return the Traits for the specified record or it will fail.

See IHE ITI TF volume 2 for a discussion of supported PID query parameters and for response parameters in the ITI-21. Also see HL7 discussion of the QBPQ22 message and the RSPK21 response on page 3-64 with an example on page 3-67.

QBPQ22 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version
- ?QPD
- ?RCP

RSPK21 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version
- ?MSA
- ?[ERR] – repeats, as required
- ?QAK
- ?QPD
- ?[{PID [PD1] [{NK1}] [QRI] }] – repeats, as required for each set of demographic data
- ?[DSC] – continuation indicator, repeats, as required

	PIM	PSM
Inputs	✍**Source ID** ✍**Entity ID**	✍**This is specified in the QPD segment.** ✍**This is also specified in the QPD segment.**
Outputs	✍**Unordered set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).** ✍**Entity Status**	✍**In this implementation, the trait values are returned in an HL7 v2.5 RSP message.** ✍ ✍
Error Conditions	✍**Entity Identifier does not exist (error based on ID match).**	**Error Conditions will be reported in an ERR segment in the RSP, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.**
Additional Details		

12.4.1.2.6 FindIdentitiesByTraits

The FindIdentitiesByTraits operation uses parameters structured as an HL7 QBPQ22 message with RSPK21 response. It returns a list of patient records that have a "best fit" match to the submitted Search Qualifiers.

It differs from the GetEntityTraitValues operation described above in that the query can include any available Traits as criteria, as well as combinations of the Source ID, Entity ID, and IXS ID. This operation may return no Entities, exactly one Entity, or a list of Entities. This list can be limited in length, and it can be repeatedly queried to get additional rows, as required.

Given a list of Traits (Trait Identifier or Name, Trait Value pairs), this allows for a search for matching Entities. Outputs include a quality of match.

	PIM	PSM
Inputs	**✍Unordered set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).** **✍Trait Identifiers or Names to be returned (can be different than input -default).** **✍Matching algorithm (Optional).**	**✍This implementation uses a standard HL7 v2.5 QBPQ22 message to hold the traits.** **✍Not supported in this implementation.** **✍Not supported in this implementation.**
Outputs	**✍Set of: Source ID, Entity ID, Quality of match tuples together with Status and other traits OR "zero matches found."** **✍SFM also identified a possible output as a list of domains that were covered by the search ****.**	**✍In this implementation, the list of identifiers are returned in an HL7 v2.5 RSP message. In this implementation, an HL7 ACK message is returned when there are no matches.** **✍The searchQualifier can include a list of domains to restrict the search to.**
Error Conditions	**✍A Trait Identifier/Name in the input list does not exist in metadata.**	**✍Error Conditions will be reported in an ERR segment in the RSP, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.**
Additional Details		

12.4.1.2.7 ListLinkedIdentities

The ListLinkedIdentities operation uses parameters structured as an HL7 QBPQ23 message with RSPK23 response. It returns a list of linked identifiers of patient records from all other domains that are linked to the queried information about one particular patient in a specified domain.

This corresponds to the IHE PIX Query.

See HL7 discussion of the QBPQ23 message and the RSPK23 response on page 3-67 of the HL7 2.5 specification with an example on page 3-69.

QBPQ23 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version

- ??QPD
- ??RCP

RSPK23 Segments Needed:

- ??MSH
- ??[SFT] – used to report software vendor ID & version
- ??MSA
- ??[ERR] – repeats, as required
- ??QAK
- ??QPD
- ??[PID] – repeats, as required
- ??[DSC] – continuation indicator, repeats, as required

	PIM	PSM
Inputs	✍**Source ID** ✍**Entity ID** ✍**SearchQualifiers (optional)**	✍**This is specified in the QPD segment.** ✍**This is also specified in the QPD segment.** ✍**Domains constraints are submitted as specified by IHE PIX Query.**
Outputs	✍**List of Entities (Source ID, Entity ID) that are linked to the specified Entity.**	✍**HL7 RSP, with Continuation Pointer as appropriate.**
Error Conditions	✍**Entity ID does not exist.** ✍**Source ID not known to the system.**	**Error Conditions will be reported in an ERR segment in the RSP, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.**
Additional Details		

12.4.1.2.8 ListUnlinkedEntities

The parameters are structured exactly like the GetAllEntityTraits operation described above.

Given an Entity ID, list all other entities that are not linked to the Entity (optionally constrained within one or more Entity Domains) but whose traits match the traits of the Entity.

	PIM	PSM
Inputs	✍**Source ID** ✍**Entity ID** ✍**Search Qualifiers**	✍**This is specified in the QPD segment.** ✍**This is also specified in the QPD segment.** ✍**The search can be restricted to domains specified in the QPD segment.**
Outputs	✍**List of Entities (IdentityQualifier, Entity Identifier, Status, Trait List) that could be linked to the specified Entity.**	✍**In this implementation, the list of entities are returned in an HL7 v2.5 RSP message.**

Error Conditions	✍**Entity reference does not exist.** ✍**IdentityQualifier not known to the system.**	✍**Error Conditions will be reported in an ERR segment in the RSP, using definitions from Section 6 above. Status condition codes returned will only support Error, Warn, and Info.**
Additional Details		

12.4.1.2.9 DescribeEntityTraits

This operation is not supported in this profile.

12.4.1.3 IXSAdminEditorInterface

The following operations are defined for this interface.

12.4.1.3.1 LinkEntities

The LinkEntities operation uses parameters structured as an HL7 ADTA24 message. It returns an ACK indicating the success of the action.

Also See HL7 discussion of the ADTA24 message on page 3-33.

ADTA24 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version
- ?EVN – required, but not used
- ?PID – Patient #1
- ?PID – Patient #2

	PIM	**PSM**
Inputs	✍**Source – Source ID – Entity ID.** ✍**Target – Source ID – Entity ID.** ✍**Unordered Set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	✍**This is mapped to the PID segment[1].** ✍**This is mapped to the PID segment[2].** ✍**Use a standard HL7 v2.5 ADT message to hold the source and target ID values. No other entity Traits are required.**
Outputs	✍**Acknowledgement**	✍**HL7 ACK**

Error Conditions	✍**Entity reference does not exist (error based on ID match).** ✍**Mandatory traits deleted.** ✍**A trait in the list does not exist in the system metadata for the Entity Type.** ✍**There are duplicates in the trait list.** ✍**Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the ACK. Status condition codes returned will only support Error, Warn, and Info. All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.3.2 UnLinkEntities

The UnLinkEntities operation uses parameters structured as an HL7 ADTA37 message. It returns an ACK indicating the success of the action.

Also See HL7 discussion of the ADTA37 message on page 3-45.

ADTA37 Segments Needed:

- ??MSH
- ??[SFT] – used to report software vendor ID & version
- ??EVN – required, but not used
- ??PID – Patient #1
- ??PID – Patient #2

	PIM	**PSM**
Inputs	✍**Source – Source ID – Entity ID.** ✍**Target – Source ID - Entity ID.** ✍**Unordered Set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	✍**This is mapped to the PID segment[1].** ✍**This is mapped to the PID segment[2].** ✍**Use a standard HL7 v2.5 ADT message to hold the source and target ID values. No other entity Traits are required.**
Outputs	✍**Acknowledgement**	✍**HL7 ACK**
Error Conditions	✍**Entity reference does not exist (error based on ID match).** ✍**Mandatory traits deleted.** ✍**A trait in the list does not exist in the system metadata for the Entity Type.** ✍**There are duplicates in the trait list.** ✍**Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the ACK. Status condition codes returned will only support Error, Warn, and Info. All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.3.3 MergeEntities

The MergeEntities operation uses parameters structured as an HL7 ADTA40 message. It returns an ACK indicating the success of the action.

Also See HL7 discussion of the ADTA40 message on page 3-47.

ADTA40 Segments Needed:

- ?MSH
- ??[SFT] – used to report software vendor ID & version
- ??EVN – required, but not used
- ?PID
- ??MRG – required

	PIM	PSM
Inputs	**✍Source – Source ID – Entity ID.** **✍Target – Source ID - Entity ID.** **✍Unordered Set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	**✍This is mapped to the MRG segment.** **✍This is mapped to MSH.4 and PID.3, first element in the list.** **✍Use a standard HL7 v2.5 ADT message to hold the source and target ID values. No other entity Traits are required.**
Outputs	**✍Acknowledgement**	**✍HL7 ACK**
Error Conditions	**✍Entity reference does not exist (error based on ID match).** **✍Mandatory traits deleted.** **✍A trait in the list does not exist in the system metadata for the Entity Type.** **✍There are duplicates in the trait list.** **✍Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the ACK. Status condition codes returned will only support Error, Warn, and Info. All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.3.4 UnMergeEntities

This operation is not supported in this profile.

12.4.1.3.5 ActivateEntity

The ActivateEntity operation uses parameters structured as an HL7 ADTA08 message to set a status flag in the IXS to make the Entity unavailable to general queries. The record will remain available to specific queries by Source ID and Entity ID, but will not be available in record linking and queries by traits other than Source ID and Entity ID. The operation returns an ACK indicating the success of the action.

Also See HL7 discussion of the ADTA31 message on page 3-40.

ADTA31 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version
- ?EVN – required, but not used
- ??PID
- ??PV1 – required, but not used
- ??PV2 – required to set PV2-24 Patient Status

	PIM	PSM
Inputs	**✍Source ID** **✍Entity ID** **✍Unordered Set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).**	**✍This is mapped to MSH.4 in HL7 v2.5.** **✍This is mapped to PID.3, first element in the list.** **✍Use a standard HL7 v2.5 ADT message to hold the status flag. Set PV2-24 Patient Status Code to "A" to indicate Activated Patient. Set PV2-46 to Patient Status Effective Date.**
Outputs	**✍Acknowledgement**	**✍HL7 ACK.**
Error Conditions	**✍Entity reference does not exist (error based on ID match).** **✍Mandatory traits deleted.** **✍A trait in the list does not exist in the system metadata for the Entity Type.** **✍There are duplicates in the trait list.** **✍Invalid data type or code value for trait.**	**Error Conditions will be reported in an ERR segment in the ACK. Status condition codes returned will only support Error, Warn, and Info. All error and status reporting will conform to the description in Chapter 6.**
Additional Details		

12.4.1.3.6 DeactivateEntity

The DeactivateEntity operation uses parameters structured as an HL7 ADTA08 message to set a status flag in the IXS to make the Entity unavailable to general queries. The record will remain available to specific queries by Source ID and Entity ID, but will not be available in record linking and queries by traits other than Source ID and Entity ID. The operation returns an ACK indicating the success of the action.

Also see HL7 discussion of the ADTA08 message on page 3-15.

ADTA31 Segments Needed:

- ?MSH
- ?[SFT] – used to report software vendor ID & version

- ??EVN – required, but not used
- ??PID
- ??PV1 – required, but not used
- ??PV2 – required to set PV2-24 Patient Status

	PIM	PSM
Inputs	✍Source ID ✍Entity ID ✍Unordered Set of Traits (Data Structures consisting of a single or hierarchic structure of name-value pairs).	✍This is mapped to MSH.4 in HL7 v2.5. ✍This is mapped to PID.3, first element in the list. ✍Uses a standard HL7 v2.5 ADT message to hold the status flag. Set PV2-24 Patient Status Code to "D" to indicate Deactivated Patient. Set PV2-46 to Patient Status Effective Date.
Outputs	✍Acknowledgement	✍HL7 ACK.
Error Conditions	✍Entity reference does not exist (error based on ID match). ✍Mandatory traits deleted. ✍A trait in the list does not exist in the system metadata for the Entity Type. ✍There are duplicates in the trait list. ✍Invalid data type or code value for trait.	Error Conditions will be reported in an ERR segment in the ACK. Status condition codes returned will only support Error, Warn, and Info. All error and status reporting will conform to the description in Chapter 6.
Additional Details		

12.4.1.4 IXSMetaDataInterface

No interfaces for the Meta Data are supported in this Semantic Profile.

13 Compatibility to Other Standards

13.1 Compatibility to HL7

IXS provides the means to exchange HL7 Version 2.x and Version 3 messages and documents as semantic signifier and logical record payloads.

13.2 Compatibility to IHE Standards

Correspondence between IXSMgmtAndQueryInterface methods and IHE transactions is given here.

Table 13.1 - IXS Correspondence to IHE PIX and PDQ

IXSMgmtAndQueryInterface	IHE PIX/PDQ
CreateIdentityFromEntity	Patient Identity Feed
RegisterEntityWithIdentity	Patient Identity Feed
UpdateEntityTraitValues	Patient Identity Feed
ListLinkedIdentities	PIX Query
None	PIX Notify
GetEntityTraitValues	Patient Demographics Query
FindIdentitiesByTraits	Patient Demographics Query
ListUnlinkedIdentities	Patient Demographics Query
None	Patient Demographics and Visit Query