Oracle® Healthcare Data Repository
Implementation Guide

Release 8.1.3
F52476-01
July 2022

Oracle Healthcare Data Repository Implementation Guide, Release 8.1.3

F52476-01

Contents

Preface

Documentation accessibility viii
Diversity and Inclusion viii

1 Introduction

Introduction 1-1
Consistent Business Transactions 1-1
Clinical Data Reuse and Portability 1-2
Comprehensive Community View 1-2
Overview 1-2
  HDR Architecture 1-2
Business Program Interface 1-3
System Administration 1-3
Leveraging Existing Oracle Core Technologies 1-3
HDR Usage Model 1-3
Audit 1-5
Logging 1-5

2 Implement the HDR Platform

Implementation Overview 2-1
  Implementation Process Description 2-1
  Implementation Tasks 2-1
    Implementation Tables 2-2
Enterprise Terminology Services 2-3
  ETS Implementation Procedure 2-7
  Creating New Generic Coding Schemes 2-8
  Implementing Editable Coding Schemes 2-9
    Creating an Editable Coding Scheme 2-10
    Loading an Editable Terminology 2-10
    Editing an Editable Coding Scheme 2-10
    Equivalence 2-11

- Reference 2-11
- Adding Components 2-12
- Changing Component Status 2-12
- Adding and Removing Attributes 2-12

Loading and Activating Coding Scheme Versions 2-12

- Preparing Terminology Content and Control Files 2-13
- Loading a Coding Scheme Version 2-14
- Using Oracle Database Scheduler (DBMS_SCHEDULER) 2-14
- Publishing a Coding Scheme Version 2-14
- Using Oracle Database Scheduler (DBMS_SCHEDULER) 2-15
- Activating a New Terminology Version 2-15

Implementing Interterminology and Intraterminology Equivalence 2-15

- Intraterminology Equivalence 2-16
- Interterminology Equivalence 2-16
- Combining Intraterminology and Interterminology Equivalence 2-16
- Prerequisites 2-17

Implementing Intraterminology Equivalence Using Change Files 2-17

Implementing Interterminology Equivalence Using Interterminology Mapping 2-20

Implementing Interterminology Mapping Using Cross Maps 2-20

- Guidelines: Cross Maps 2-22
- Loading Cross Maps Provided by the College of American Pathologists 2-23

Creating Local Descriptions 2-23

Managing Usage Contexts 2-23

- Associating a Usage Context with an Organization 2-24

Implementing Concept Lists 2-24

- Creating and Updating a Concept List 2-25
- Adding Concepts to a Concept List 2-25
- Adding Concepts to a Concept List 2-26
- Specializing a Concept List 2-27
- Subsetting a Concept List 2-28
- Subsetting a User Concept List 2-28
- Subsetting a Concept List of any Extensibility Type 2-28

Implement Security 2-29

Implementing ETS Classifications 2-29

- Classifications can be Linked Hierarchically 2-32
- Testing Containment 2-32
- Creating and Populating Classifications 2-32
- Creating a Classification 2-34
- Building a Classification with the HDR Maintenance Job 2-35
- Updating Published Coding Scheme Versions 2-35
- Running the HDR Maintenance Job 2-35

Scheduling the Maintenance Job 2-36

Implement Audit Services 2-37

Enabling Audit Services 2-39

Initializing Existing Audit Event Types 2-39

Creating New Audit Event Types 2-39

Invoking HDR Audit Services 2-40

Attribute Values in Audit Events 2-40

Implement HDR Object Identifiers 2-40

Implement Profile Option Services 2-42

Implement Profile Options 2-43

Creating Profile Options 2-44

Updating Profile Options 2-44

Implement Profile Option Values 2-44

Updating Profile Option Values 2-44

Implement Master Catalog and Focal Class State Transitions 2-44

Master Catalog 2-44

Master Catalog Acts 2-45

Concept Equivalence Support 2-47

Using the Code Type Attribute for Acts 2-47

Code Type ID 2-47

Code Type NULL 2-47

Code Type ANY 2-48

Master Catalog Entities 2-49

Using the Code Type Attribute for Entities 2-50

Code Type ID 2-50

Code Type NULL 2-51

Code Type ANY 2-51

Master Catalog Roles 2-52

Using the Code Type Attribute for Roles 2-54

Code Type ID 2-54

Code Type NULL 2-55

Code Type ANY 2-55

Focal Class State Transitions 2-57

Business Events 2-58

Start State and End State 2-59

Creating Master Catalog Entries 2-60

Setting Up Master Catalog 2-62

Master Catalog Object Factory 2-63

Implement Inbound Messaging Services 2-64

Configuring Interactions 2-71

Configuring Sender, Sender Interaction, and Side Effect 2-71

Invoke Inbound Messaging Services 2-72
Implement Concurrent Program Service 2-73
Implement RIM Service 2-74
Generate SQL Queries 2-75
-DCTB_SUBQRY_OPT_METHOD=NONE 2-75
-DCTB_SUBQRY_OPT_METHOD=EXISTS 2-75
-DCTB_SUBQRY_OPT_METHOD=JOIN 2-75

3 Implement the Healthcare Enterprise XDS.b Web Service

Introducing Cross-Enterprise Document Sharing-b 3-1
Configure Oracle HDR as a Document Repository 3-2
Configure Registry for Accepting Register Document Set-b Web Service Calls 3-2
Configure Repository for Syslog Audit Messages 3-3
Configure Repository Unique ID 3-3
User Creation in 'myrealm' WebLogic Server realm for IHE XDS.b Web Services 3-3
Configure HDR as a Time Client Actor 3-3
Configure HDR a Secure Node Actor 3-4
Set HDR for TLS Communication with Document Source and Document Consumer 3-4
Set Up HDR for TLS Communication with Document Registry and Syslog Audit Server 3-5
Generate Audit Event - OS Level Authentication Events 3-5
Configure ITI-42 Timeouts for IHE XDS Registry Web Service Call 3-10
httpConnTimeout 3-10
httpReadTimeout 3-11
Configure Registry for Accepting Delete Document Set ITI-62 Web Service Calls 3-11
Configure Profile Option for ITI-62 Delete Document Set Transaction 3-11
Asynchronous IHE XDS.b Web Services 3-12
Configure Profile Options for Asynchronous Web Services 3-13
Configure WLS for Asynchronous Web Services 3-13
Configure Message Receipt Timeout Value for Asynchronous Web Services 3-14
Configure HDR To Accept Web Services Invocation Over http Protocol 3-14
Configure Credential Store 3-14
Configure MTOM/ XOP for the ITI-43 Transaction (Retrieve Document Set) 3-14

4 Implement CDA (Clinical Document Architecture) Persistence Service

A Appendix A: ETS Supported Terminologies and Cross Maps

ETS Supported Terminologies and Cross Maps A-1

B Appendix B: Extensible Concept Lists

Extensible Concept Lists B-1

C Appendix C: Concept List Equivalents

Concept Lists Equivalents C-1

D Appendix D: Running HDR Terminology Jobs

Running HDR Loader Job D-1
Running HDR Importer Jobs D-2
Running HDR Maintenance Job D-2
Running HDR ETS Programs Using Scripts D-3
    Middle Tier D-3
    Database Tier D-9

E Appendix E: Act Configuration Artifacts

Act Configuration Artifacts E-1
Act Configuration Artifact Tables E-1
Act Configuration Artifact Tables for HDR Version 5 E-4

F Appendix F: Abbreviations and Acronyms

Abbreviations and Acronyms F-1

Preface

This preface contains the following sections:

Documentation accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1 Introduction

- Introduction
- Consistent Business Transactions
- Clinical Data Reuse and Portability
- Comprehensive Community View
- Overview
- Business Program Interface
- System Administration
- Leveraging Existing Oracle Core Technologies
- HDR Usage Model
- Audit
- Logging

Introduction

Oracle Healthcare Data Repository (HDR) is a comprehensive data repository and service infrastructure that provides independent software vendors, system integrators, and provider organizations with a state of the art software platform that lets them build robust and scalable healthcare applications.

HDR software components let HDR based applications centralize and consolidate patient, provider, and healthcare data, including business rules that span the enterprise. HDR helps healthcare organizations to overcome challenges unique to their industry, and to achieve the following operational benefits:

- Managing business transactions consistently in the patient care process, thereby increasing the efficiency, quality and competitive edge of the organization.
- Maximizing healthcare data reuse and portability through the use of standards, enabling seamless integration and consistent implementation between variant healthcare information systems.
- Implementing protocol-based decision support, based on a complete view of patient information.

Consistent Business Transactions

HDR provides a single definition of each business object (encounter, patients, providers, medical acts...) across the healthcare organization. This lets users create and update key patient information in a consistent manner. For example, patient encounter information can be shared by various authorized personnel across a healthcare organization-enabling the creation and maintenance of a consolidated electronic health record.

Clinical Data Reuse and Portability

In many contemporary healthcare organizations, individual departments maintain their own independent information systems. These systems typically operate autonomously and do not synchronize patient data-making it difficult to develop a consistent, integrated view of the patient.HDR provides a unified data model based on the HL7 version3 Reference Information Model (RIM), combined with sophisticated terminology mediation services incorporating standard terminologies that enable caregivers to efficiently manage and synchronize patient information. This approach lets caregivers avoid time-consuming data entry in multiple data sources while integrating patient information.

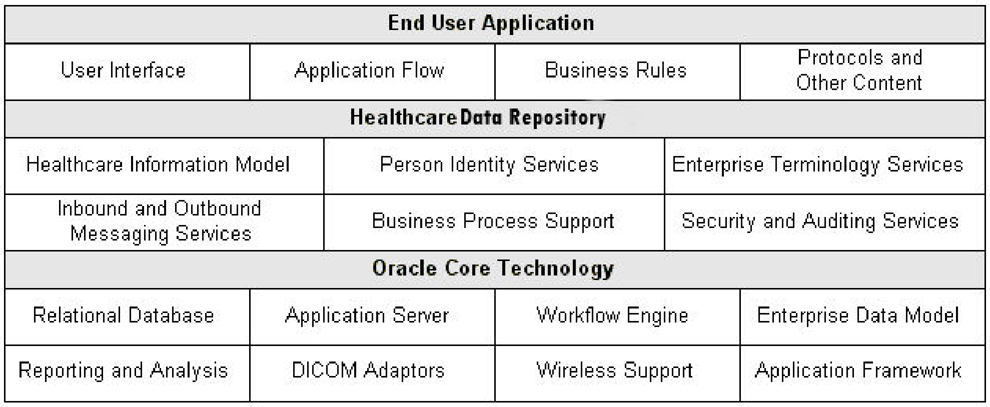
Comprehensive Community View

HDR lets multiple departments within a healthcare organization share data while maintaining a high level of autonomy. HDR provides multi-organization functionality that lets healthcare providers consistently manage and update central patient electronic healthcare records. As each department (or patient) touches a patient's specific data, the central patient record is updated accordingly. All organizations in a healthcare community can access relevant patient information (with appropriate patient consent), which translates into improved patient care, safety and reduced costs.

Overview

HDR consists of a set of services based on a foundation of selected Oracle core technologies. It includes software components that centralize and consolidate patient, provider, and clinical objects across the healthcare enterprise-providing unified access to a comprehensive healthcare infrastructure. The structure is a relational database implementation of the Reference Information Model that was developed for version 3 of the HL7 standard.

HDR Architecture



End User Application			
User Interface	Application Flow	Business Rules	Protocols and Other Content
Healthcare Data Repository			
Healthcare Information Model	Person Identity Services	Enterprise Terminology Services	
Inbound and Outbound Messaging Services	Business Process Support	Security and Auditing Services	
Oracle Core Technology			
Relational Database	Application Server	Workflow Engine	Enterprise Data Model
Reporting and Analysis	DICOM Adaptors	Wireless Support	Application Framework

The healthcare functional domains (administrative, clinical, and financial) and core services are exposed through a Java-based Applications Programming Interface (API). The underlying HDR business logic can be extended as required for specific application functionality.

The HDR platform supports industry standards, including HL7, HIPAA privacy regulations, and standard terminology sets such as SNOMED, CPT4, ICD9, and LOINC etc. You can use this platform to accelerate your migration to these industry standards while focusing on development of healthcare application functionality.

Business Program Interface

HDR Business program interfaces provide access to the content of the HDR data repository. They provide a thin layer of business logic, and depend upon the core HDR services.For example, an API for ordering a drug issues a call to a security interface to verify authorization, and issues a call to an Enterprise Terminology Service interface to validate drug codes. The business domains supported by HDR include the core functionality required by a Healthcare organization, as well as management of orders and observations.The core HDR services define a common service infrastructure for the development of functional components in healthcare applications. These services are exposed through a Java program interface. The core HDR service interfaces provide the basis for the business functionality. They also support efficient development of secure and scalable applications on a normalized and secure date repository.

System Administration

HDR platform services are managed through a combination of applications and program interface services:

- Security Manager allows System Administrators define security policy and manage security services.
- HDR Configuration APIs allow you to manage HDR configuration services.
- ETS Administration and Authoring APIs allow you to manage Enterprise Terminology Services.

See also:

- *Oracle Healthcare Data Repository Javadoc* for information about interfaces supported by HDR.

Leveraging Existing Oracle Core Technologies

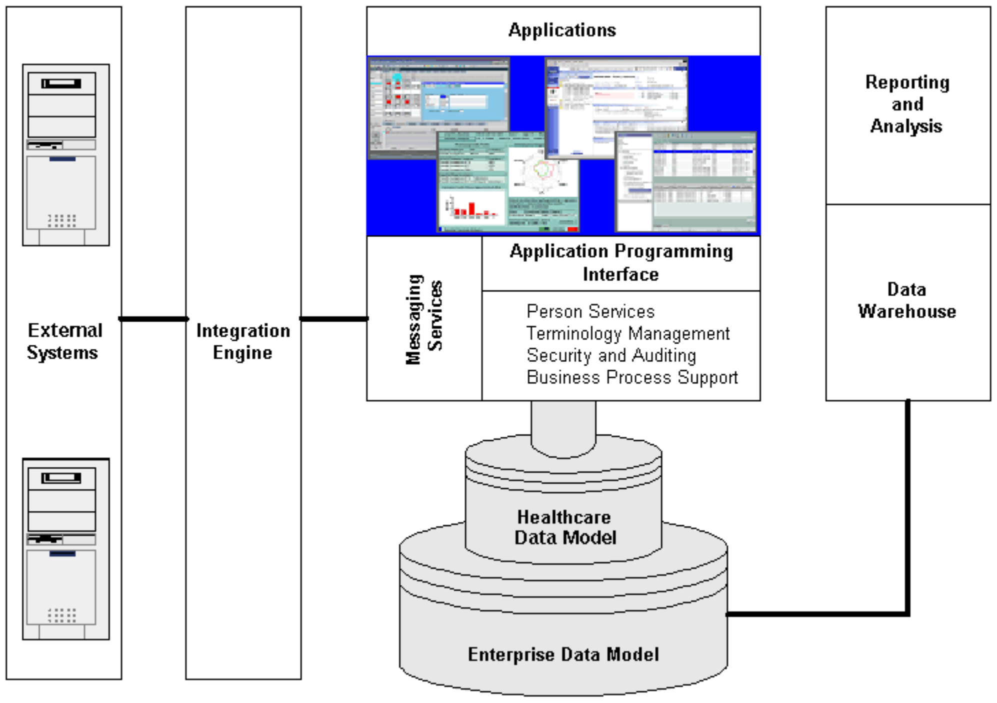
HDR uses Oracle core technologies, which provide high performance and scalability characteristics, and ensure seamless integration with other Oracle products that use the same technology foundation.

See also:

- *About Oracle Healthcare Data Repository* for additional information about Oracle core technologies.

HDR Usage Model

The following chart illustrates the usage model for the HDR platform:



The HL7 version 3 Reference Information Model (RIM) provides the basis for how HDR represents healthcare data. HDR uses this model to develop a collection of application services that are exposed through a J2EE compliant application programming interface, including the following:

- Services that help to consolidate patient data into unified records.
- Services that align the terminologies and coding system into a single unified concept-based representation.
- Rigorous security and auditing services.
- Persistence services for key healthcare business processes.

Developers use this foundation to build the next generation of applications for healthcare organizations. HDR provides a unified representation for the key business objects underlying these applications. Resulting application suites thus enjoy a high degree of interoperability.

The underlying healthcare data model is embedded in a comprehensive Enterprise Data Model. This broad foundation unifies the management of all of the data used in a healthcare enterprise.

HDR also includes a powerful set of messaging services that provide interfaces to systems developed outside of the HDR architecture. These messaging services manage the exchange of data between an HDR application and other information management systems, including *legacy* systems. The HDR Repository can thus become a comprehensive patient record for the entire enterprise.

HDR supports a transactional view of healthcare data. This view captures the dynamics of healthcare data as it evolves during business processes and supports operational applications. It also can support basic business intelligence functionality.

For example, OBIEE or other similar products can be used to build reports directly on the HDR Platform.

Alternatively, a data warehouse can be built on the HDR Repository. This data warehouse can support more complicated reports and analysis. Because of the data quality services provided by HDR, the resulting decision support functionality can be based on a comprehensive view of the enterprise.

Audit

The HDR Audit Service [AuditService] is a core HDR interface that lets you log and monitor HDR activities, to monitor security policy and regulation compliance—by recording actions taken during user sessions. Such event records can help detect actual or attempted violations of policy and operation procedures.

See also:

- *Programmer's Guide* for more information on auditing and logging.

Logging

The HDR logging system is based upon an understanding of threats, risks and server/ application health/vital signs. Even when operating under normal conditions, applications need to maintain health and efficiency in a shared ecosystem and make sure that the key metrics are adhered to.

In order to maintain a stable, efficient, and self-correcting cloud environment, and maintain wellness, the infrastructure must be able to log/capture appropriate information as deemed necessary and enable the administrator to assess the situation.

HDR logs:

- Critical activities: activities of operations and access to critical resources.
- System events: identify underlying programs and infrastructure.
- Who, what and when.
- All services and modules across all tiers (i.e. Web, APIs, Persistence, etc.).

HDR logging can be configured to use either JDK Logging or Log4J Logging Framework. The user has to specify which logging framework he would want to use.

See also:

- *Programmer's Guide* for more information on auditing and logging.

2

Implement the HDR Platform

- Implementation Overview
- Enterprise Terminology Services
- Implement Audit Services
- Implement HDR Object Identifiers
- Implement Profile Option Services
- Implement Master Catalog and Focal Class State Transitions
- Implement Inbound Messaging Services
- Implement Concurrent Program Service
- Implement RIM Service

Implementation Overview

This section describes the implementation process for the Oracle Healthcare Data Repository development platform, including the implementation task sequence.

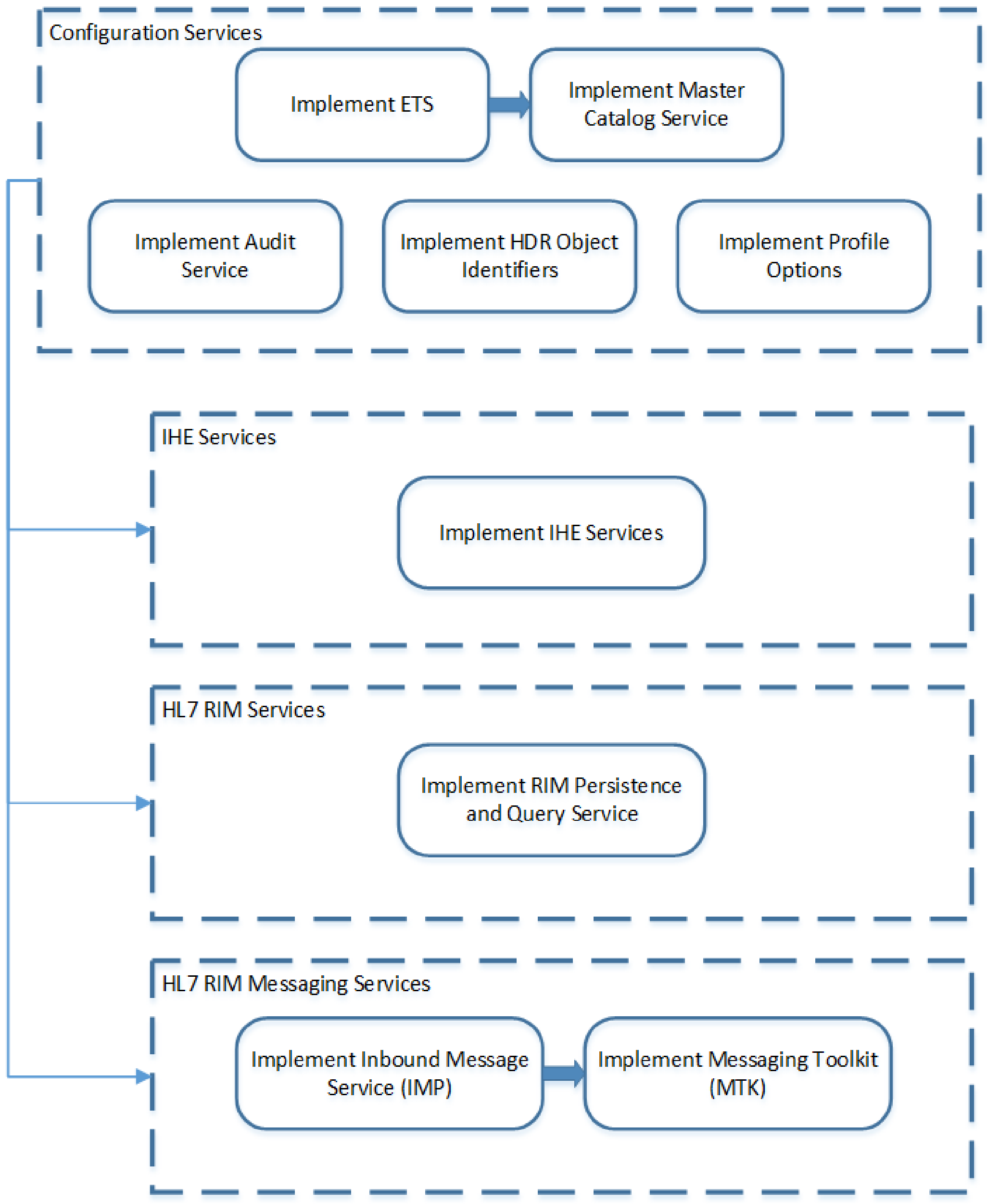
Implementation Process Description

Oracle Healthcare Data Repository provides an API call interface to implement its core services. Some implementation procedures incidentally employ a command-line interface as well.

Implementation Tasks

The following chart provides an overview of the implementation process for the HDR Platform; the table that follows it lists all implementation tasks:

Figure 2-1 Implementation Task Process: HDR Platform [Dependencies]


Configuration Services
Implement ETS
Implement Master Catalog Service
Implement Audit Service
Implement HDR Object Identifiers
Implement Profile Options
IHE Services
Implement IHE Services
HL7 RIM Services
Implement RIM Persistence and Query Service
HL7 RIM Messaging Services
Implement Inbound Message Service (IMP)
Implement Messaging Toolkit (MTK)

Implementation Tables

Table 2-1 HDR Implementation Tasks Summarized (HDR Platform)

Task	Description	Optional?
1	Implementing Enterprise Terminology Services (ETS)	No
2	Implementing Security	No
3	Implementing Audit Services	Yes

Table 2-1 (Cont.) HDR Implementation Tasks Summarized (HDR Platform)

4	Implementing HDR Object Identifiers	No
5	Implementing Profile Option Services	Yes
6	Implementing Master Catalog	No
7	Implementing Inbound Messaging Services	Yes
8	Implementing the Healthcare Enterprise Cross-Enterprise Document Sharing-b Web Service	Yes
9	Implementing Clinical Document Architecture Persistence or Clinical Document Ingest Web Service	Yes

Enterprise Terminology Services

Enterprise Terminology Services (ETS) is a core component of HDR that incorporates a range of terminology systems and provides extensive terminology services to HDR applications, including the following principal features:

- Consistent and real-time access to all terminology content - whether standards-based or user-defined.
- Support for standard-based terminologies:
  - SNOMED, LOINC, and FDB terminologies through a specialized model (referred as Core terminologies in ETS).
  - Other terminologies like ICD9, CPT4, and HCPCS through a generic model.
- Support for user-defined terminologies.
- High level of terminology integration - between different terminologies and between different versions of the same terminology.
- Support for user-defined containers of terminology content such as Concept Lists and Classifications. These containers can be used for building application interfaces, constraining and validating attribute values, and generating context-sensitive reports.
- Multi language support (MLS) on concept descriptions (except Classifications and editable terminologies).

ETS is based on the following core concepts:

- Generic and Core Terminologies
- Equivalence
- Interterminology Mapping
- Concept Lists
- ETS Classifications

- Multi Language Support

Generic and Core Terminologies

ETS uses a generic terminology model that captures the essential features of disparate terminology systems. The generic terminology model provides:

- Real-time access to terminology content.
- A generic API that provides basic terminology services for all terminologies.
- A data model for custom terminologies.

Terminologies are represented in ETS as Coding Schemes. A Coding Scheme is a generic structure that contains terminology meta-data, such as name, description, and active versions. Actual terminology content is loaded and stored in a Coding Scheme Version. Names of Coding Scheme Versions are decided by the user and are specified when the version is loaded. You can load new Coding Scheme Versions as required. For Coding Schemes that have multiple versions, exactly one version can be designated as default. Note that editable terminologies have only one version; Concepts in an editable terminology can be modified without loading a new version.

A Coding Scheme Version contains a definite set of Concepts, Descriptions, Attributes, and Relationships. A concept is the basic unit of information in a coding scheme version: It corresponds to a specific unit of meaning in the native terminology. Every concept has a Concept Code, which is the code given to it by the terminology. ETS identifies concepts and other ETS components (Descriptions and Relationships) using a system-generated identifier called ETS ID. Concepts (and other components) from different versions of a terminology have different ETS IDs, as the concept code may not correctly identify a concept in a different version.

A concept may have one or more textual descriptions. ETS supports multiple descriptions for a concept in the languages supported by that coding scheme version – whether defined by the terminology, or added later by the user. For concepts that have multiple descriptions, exactly one description must be designated as the preferred description for every language supported by the Coding Scheme Version. All other descriptions of a given language, associated with that concept are designated as synonyms. An application may use specific descriptions for designated contexts. This is done by defining Usage Contexts and associating local descriptions to those contexts.

A Relationship represents a directed relation between two concepts: from a source concept to a target concept. Relationships can be defined between concepts in the same Coding Scheme Version. These are usually provided as part of the terminology itself.

The generic terminology model serves as the base for a number of standard-based terminologies for which ETS provides special support. These terminologies are referred to as core terminologies in HDR. The following terminologies are referred to as core terminologies in ETS:

- FDB4
- HL7 v3 Code Systems (Seeded)
- LOINC
- IHTSDO

Special support for core terminologies is in the form of:

- Terminology-specific APIs (in addition to the generic APIs).
- Specific loaders (and associated integrity checking) for loading the terminologies into ETS.

As core terminologies are based on the generic terminology model, they support all features of generic terminologies, such as local descriptions, usage contexts, equivalence, attributes, and cross maps. However, core terminologies are not editable: new Coding Scheme Versions have to be loaded and activated to update the terminology content.

Equivalence

Equivalence is a symmetric, reflexive, and transitive relationship between two concepts. Two concepts—from the same or different terminologies—that have the same meaning, are considered equivalent in ETS. Equivalent concepts can be used interchangeably, without any loss of meaning. ETS provides APIs for identifying and retrieving equivalent concepts for a given concept.

> **Note:**
>
> ETS does not support authoring of equivalence information. Equivalence must be provided to ETS in one of the specified ETS file formats.

Equivalence may be Intra-terminology or Inter-terminology. Intra-terminology equivalence is defined between concepts from the same terminology. For example, the concept Cholera is represented in ICD-9-CM 2002 and in ICD-9-CM 2003 by the same concept code [001_CHOLERA]. Because both concepts have the same meaning, they can be treated as equivalents---ETS treats them as equivalent by default, as they belong to the same coding scheme and have the same concept code.

Intra-terminology equivalence information is provided to ETS as a “change file", when loading a new version of a terminology. Change files identify reuse of codes (codes that represent different meanings than the previous version), and reassignment of codes (meanings that are now represented with different codes).

Oracle provides Intra-terminology equivalence information only for core terminologies. You must author and load change files for new versions of custom terminologies not supported by HDR and generic terminologies supported by HDR.

> **Note:**
>
> ETS does not support the authoring of change files.

ETS allows concepts from two different terminologies to be defined as semantic equivalents. Equivalence between concepts from different terminologies, or Inter-terminology equivalence, is defined using cross maps. For example, the concept for the disease Cholera in the ICD-10 terminology (2016 version), and the concept representing the same disease in the IHTSDO terminology (2017 version), can be defined as equivalents using a cross map.

Interterminology Mapping

Inter-terminology Mapping provides support for any type of relationship, including equivalence, between two different terminologies. Relationships that can be defined include (but not restricted to) broader-than, narrower-than, and clinical-to-administrative relationships.

Inter-terminology Mapping is implemented using **Cross Maps**. A Cross Map defines the relationship between a source concept and a target concept. A number of Cross Maps are aggregated into a **Map Set**. ETS specifies file formats for Map Sets and Cross Maps. The HDR Loader job loads these files into ETS tables.

Concept Lists

Concept lists are arbitrary lists of ETS concepts that can be used for a variety of purposes, including validation of attribute values and populating user interface controls. Concept lists are used by all types of HDR solutions including HDR Messaging applications.

ETS Classifications

ETS Classifications are containers for grouping existing ETS concepts from different coding schemes and versions. Classifications are intended for large categorizations of concepts, while concept lists are intended for smaller sets for the purposes of validation or display. ETS Classifications provide the following features:

- Classifications can be arranged in a hierarchy.
- Tests for containment in a classification search down the levels of the hierarchy.
- Classifications can be created and populated through an API or through creating and loading text files.
- Classification contents incorporate equivalence.
- Concepts added to an ETS Classification retain their equivalence information and characteristics.
- ETS Classifications are themselves ETS Concepts--components of a special, predefined editable terminology called ETSClassifications. Accordingly the following applies:
  - A classification has a concept identifier.
  - A classification can have multiple descriptions, including local descriptions.
  - A classification's local descriptions can be associated with usage contexts.
- ETS Classifications can be defined using the API or using the HDR Loader job and HDR Importer job.

Multi Language Support

ETS provides Multi Language Support (MLS) on:

- Terminology-specified concept descriptions of non-editable terminologies.
- Locally-specified concept descriptions of terminologies (editable and non-editable) and Classifications.

MLS in ETS lets you load Coding Scheme versions with Concept descriptions in multiple languages. Each Coding Scheme version can support Concept descriptions in multiple languages, but every Concept in a Coding Scheme version must be supplied with a terminology-preferred description in the languages supported by that version.

You can create local descriptions in multiple languages. Concept descriptions (both terminology-specified and local) based on a language can be obtained by calling methods that accept a language.

Note:

- The terminology-specified Concept descriptions of editable terminologies and Classifications are created in the base language of the HDR installation.
- ETS will not translate terminology content - whether seeded or loaded.
- ETS does not perform any validation to ascertain whether a description is actually in the language that it claims to be in. You can load pseudo translated text as concept descriptions for a supported language along with the real description text. For example, you can load the data given in the following table as Concept descriptions into ETS without getting any errors:

Table 2-2 Sample Concept Descriptions

Version ID	Concept ID	Description Text	Language
v20070101	A25.66	Fracture of Femur	American English
v20070101	A25.66	Fracture of Femur	Korean
v20070101	A25.66	Fracture of Femur	Spanish
v20070101	A25.66	Fracture of Femur	Japanese
v20070101	A25.66	Fracture of Femur	German

See also:

- Implement Master Catalog and Focal Class State Transitions
- Appendix A: ETS Supported Terminologies and Cross Maps
- Appendix B: Extensible Concept Lists
- *HDR Concept Lists Index, Oracle Healthcare Data Repository Javadoc* (click HDR Concept Lists link at bottom of Javadoc pages), for seeded concept lists and values.

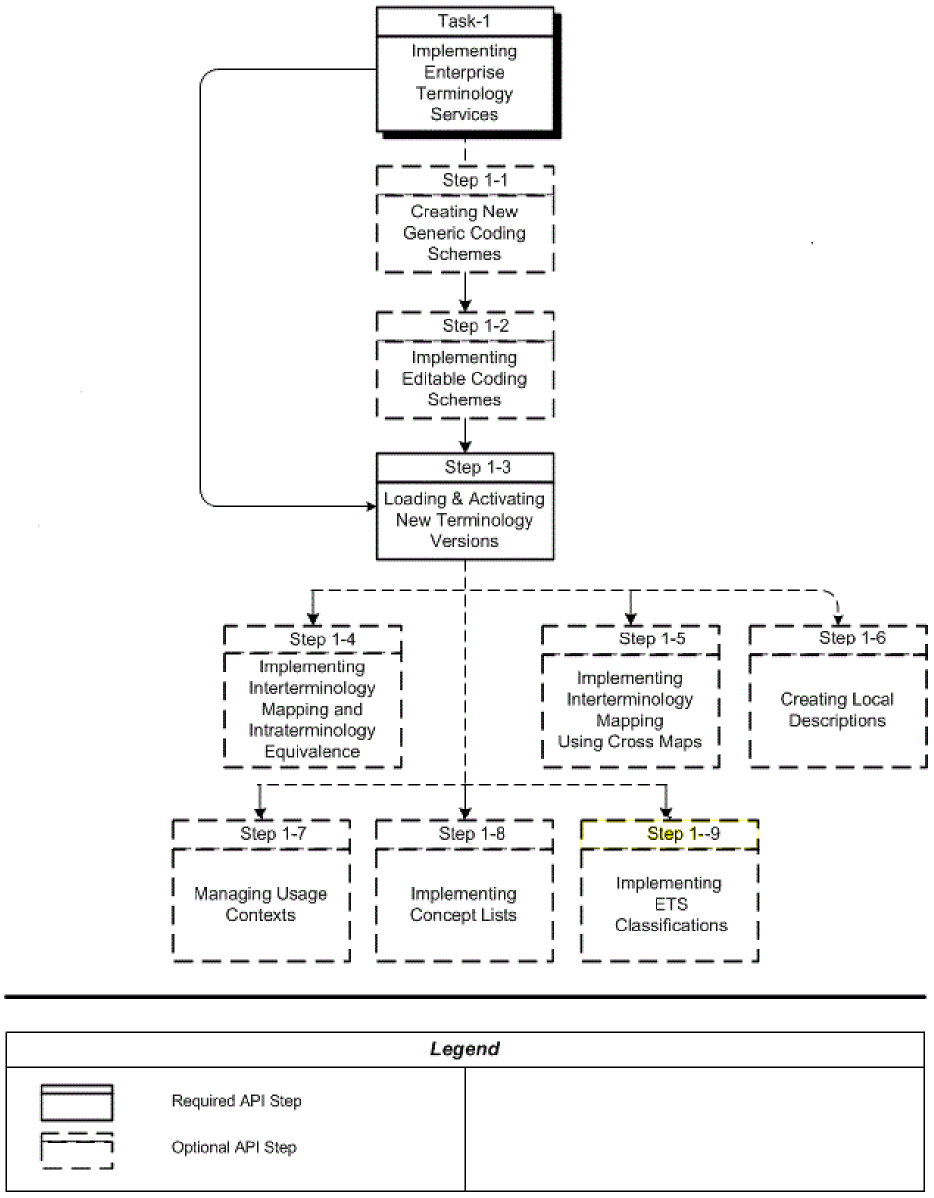
Prerequisites

- Implementing Security Services: User accounts must exist.

ETS Implementation Procedure

The following chart provides an overview of the implementation process:

Figure 2-2 Implementation Process for ETS


Task-1
Implementing Enterprise Terminology Services
Step 1-1
Creating New Generic Coding Schemes
Step 1-2
Implementing Editable Coding Schemes
Step 1-3
Loading & Activating New Terminology Versions
Step 1-4
Implementing Interterminology Mapping and Intraterminology Equivalence
Step 1-5
Implementing Interterminology Mapping Using Cross Maps
Step 1-6
Creating Local Descriptions
Step 1-7
Managing Usage Contexts
Step 1-8
Implementing Concept Lists
Step 1--9
Implementing ETS Classifications
Legend
Required API Step
Optional API Step

To implement ETS, refer to the following procedure table:

All optional steps are done via the API. Loading and activating coding scheme versions is done using the API and scripts.

Creating New Generic Coding Schemes

ETS lets users define and implement custom terminologies for specific needs. Custom terminologies must be based on the generic ETS terminology model. Coding schemes that implement the generic terminology model are known as generic coding schemes.

Caution:

Do not attempt to create coding schemes for the following core and generic terminologies supported by ETS; they are created by default when ETS is installed:

- CPT4
- FDB4
- HCPCS-2
- HL7 v3 Code Systems (Seeded)
- ICD-10
- ICD-9-CM-DRG
- ICD-9-CM-MDC
- ICD-9-CM-V1
- ICD-9-CM-V3
- LOINC
- IHTSDO

Create and implement new generic coding schemes:

1. Create the terminology files based on the specified file structure. ETS expects the terminology content to be available as a control file and a set of three terminology files. For information about the file structure, refer the /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip folder.
2. Create a coding scheme by using the HDR API.
3. Load, import, and activate the coding scheme version.

The terminology content can now be loaded using the HDR Terminology Jobs.

See also:

- Loading and Activating Coding Scheme Versions
- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip for information about the formats required for loading ETS generic terminologies.

Implementing Editable Coding Schemes

Because user-defined terminologies frequently change, ETS lets you edit generic coding schemes in place, without loading a new version. You can define generic coding schemes as editable when they are being created.

Note:

The following terminologies supported by ETS cannot be designated as editable. They are created as non-editable by default when ETS is installed:

- CPT4
- FDB4
- HCPCS-2
- HL7 v3 Code Systems (Seeded)
- ICD-10
- ICD-9-CM-DRG
- ICD-9-CM-MDC
- ICD-9-CM-V1
- ICD-9-CM-V3
- LOINC
- IHTSDO

Although editable coding schemes can be updated using ETS APIs, editable coding schemes cannot be updated by loading a new version, because they can have only one version.

Caution:

Do not attempt to load new versions for an editable coding scheme, other than the original version.

Creating an Editable Coding Scheme

Using the HDR API, you can create an editable coding scheme by creating a new generic coding scheme with the editable attribute set to true. After the coding scheme is created, you can edit the original version, but a new version is not permitted.

Loading an Editable Terminology

An initial version of the editable coding scheme must be created and loaded before it is used. Use the generic terminology loader to load an editable coding scheme version.

Editing an Editable Coding Scheme

Editing concepts, descriptions, and relationships is limited to addition and removal of attributes, changing status, and changing a description's preferred status. Other changes are made by retiring a component and adding a new component in its place. For example, a description's text cannot be changed, but the description can be retired and a new description added to replace it. The new description can optionally be designated preferred.

Equivalence

As an editable terminology has only a single version, equivalence in an editable terminology must be intraversion – in other words, equivalence may only be declared between concepts in the same version. In the initial version load, equivalence may be declared using a change file, just as for any terminology being loaded with intraversion equivalence information. After the initial load, reassignments (introduction of a new concept that has the same meaning as an existing concept but has a different concept code than the existing concept) may be declared when a new concept is added. The new concept's code may be declared, using the relevant API, to be a reassignment from an existing concept's code.

No reuse of codes (introduction of a concept whose concept code is the same as an existing code, but where the concept has a different meaning than the existing code represents) is permitted in an editable terminology.

See also:

- Implementing Interterminology and Intraterminology Equivalence
- Implementing Interterminology Mapping Using Cross Maps

Reference

Oracle Healthcare Data Repository API Documentation

The following table provided information about the ETSAuthoringService interface used to implement editable terminologies:

Oracle Healthcare Data Repository Javadoc

Table 2-3 Service and Methods: Editable Terminologies

Level	Detail
Package	*oracle.hsgbu.ets.authoring*
Interface	*ETSAuthoringService*
Methods	• *addAttributes* • *addConcepts* • *addConceptDescriptions* • *addRelationships* • *changeStatus* • *removeAttributes*

Refer to the following sections to edit the components of an Editable Coding Scheme:

- Adding Components
- Changing Component Status
- Adding and Removing Component Attributes

Note:

These changes are exclusive; any other changes to concepts, descriptions, and relationships can only be made by retiring the component and adding a new one.

Adding Components

Use the *addConcepts*, *addConceptDescriptions*, and *addRelationships* methods to add concepts, descriptions, relationships and attributes to an editable coding scheme. *Candidate* components are created first and passed to the *add* methods.

Changing Component Status

Components such as concepts, relationships and descriptions of an editable coding scheme cannot be edited or removed directly. To modify a component, it must be retired and replaced. A component can be retired or made active by changing the status flag associated with the component. Use the changeStatus method to change component status.

Adding and Removing Attributes

Use the *addAttributes* and *removeAttributes* methods to add and remove attributes, respectively.

Loading and Activating Coding Scheme Versions

Terminologies have to be loaded into ETS (as coding scheme versions), imported, and activated before they are used. This includes initial versions of core terminologies, which have to be loaded, imported and activated at implementation.

ETS provides terminology loader and importer jobs that load and import a terminology after performing required validations. ETS provides custom loader and importer jobs for core terminologies, and generic loader and importer jobs for generic and custom terminologies.

Note:

New versions of the following terminologies can be loaded if required; HDR does not seed versions of these terminologies (they are available from Apelon, Inc.):

- IETF RFC 1766
- ISO 3166-1 alpha-2
- NUBC-UB92

Caution:

Do not load versions of terminologies that are seeded in HDR. These include:

- HL7
- HDR Supplemental

If you have already loaded such versions, mark them as retired and non-default.

The procedure for implementing a new coding scheme version is the same for both generic and core terminologies. To load and activate coding scheme versions, perform the following steps:

Steps

1. Prepare the terminology files.
2. Load the terminology into ETS as a coding scheme version.
3. Publish the coding scheme version.
4. Activate the coding scheme version.

You can use HDR Terminology Jobs to load and publish coding scheme versions.

See also:

- Appendix D: Running HDR Terminology Jobs
- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip for additional information about loading.
  The notes can be categorized into DBA, General, and Terminology related notes.
  - *DBA Notes:* Gives a brief description of some common database management issues related to ETS, such as sizing issues, rollback adjustments for loads and imports, intermedia text indexes, and load/import performance, as well as a general description of database access patterns of ETS.
  - *General Notes:* Gives the basic principals common to all terminology file formats. This section must be read before moving on to the details of specific loader file formats.
  - *Terminology Notes:* Gives additional information relating to the core terminologies supported in ETS. Each of the core terminologies have a separate notes file. These files are to be referenced for information relating to the respective loader file formats.

Preparing Terminology Content and Control Files

To create the terminology files and move them into the correct folder, perform the following steps:

Steps

1. If the terminology is a generic terminology, create the terminology files in the format expected by the ETS generic loader. Otherwise, ensure that the files are in the format expected by the appropriate terminology loader. (If the terminology is supported by Apelon, this step is not required.)

   See also:

- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip, for details regarding file formats required by ETS terminology loaders.
- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme/Change_File_Formats_General.txt from hdr-1.0.0-8.0.0.zip, for details about change files.

> **Note:**
>
> - To load equivalence information for the terminology version being loaded, the change file must be specified while loading the version-ETS does not support retrospective loading of change information.
> - For equivalence processing to be performed correctly, versions must be loaded in order. Equivalence processing assumes that the codes referenced in the change file are from the version currently being loaded and its immediate predecessor.

2. Move the terminology files to a directory in the same file system as the Applications instance-a directory that is accessible by the Oracle Database Scheduler (DBMS_SCHEDULER).
3. Create a control file that reflects the locations of the terminology files and move it to a directory in the same file system as the Applications instance-a directory that is accessible by the Oracle Database Scheduler (DBMS_SCHEDULER).

Loading a Coding Scheme Version

To load a new coding scheme version, use the Oracle Database Scheduler (DBMS_SCHEDULER).

See also:

- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip, for information about control files.

Using Oracle Database Scheduler (DBMS_SCHEDULER)

Select the HDR Loader Job, and enter values for the Control File (absolute path), Coding Scheme Name, and Coding Scheme Version Name parameters. For more information, refer to Appendix D: Running HDR Terminology Jobs .

Publishing a Coding Scheme Version

A loaded terminology is staged for importation into ETS. Use HDR Importer Job directly for publishing a coding scheme version. For more information refer Appendix D: Running HDR Terminology Jobs .

The published coding scheme version is in the quarantined state by default. The coding scheme version must be activated before it can be used.

Note:

In order to support concept equivalence, the HDR Importer job process does not publish a second quarantined version of a coding scheme if one already exists. This facilitates verification of equivalence between the quarantined version and the previous version of the terminology before the quarantined version is published.

Using Oracle Database Scheduler (DBMS_SCHEDULER)

For publishing a staged coding scheme version, select the HDR Importer job, and enter values for the Load Sequence Number and Dry Run Mode parameters. You can get the Load Sequence Number from the log file of the HDR Loader job that has successfully loaded the data (coding scheme versions, classifications, and cross maps).

See also:

- *Oracle Applications System Administrator's Guide*
- Appendix D: Running HDR Terminology Jobs

Note:

In order to support concept equivalence, the HDR Importer job does not publish a second quarantined version of a terminology if one already exists. This facilitates verification of equivalence between the quarantined version and the previous version of the terminology before the quarantined version is published.

See also:

- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme, for details regarding file formats for ETS terminologies and loaders.

Activating a New Terminology Version

A quarantined coding scheme version must be activated before it can be used. A quarantined version can be activated only ETS API.

Implementing Interterminology and Intraterminology Equivalence

Over a period of time, ETS can use different concepts to record the same meaning. This happens either because of changes to the terminology or because different terminologies are used to record the same meaning. The Concept Equivalence service lets HDR solutions find data recorded using different concepts.

With Concept Equivalence, concepts from the same or different terminologies—that have the same meaning— are considered equivalent. Concept Equivalence facilitates specification and query of concepts that are equivalent.

Concept equivalence is also used by HDR solutions that implement the HDR messaging services.

Concept equivalence services include:

- Intraterminology Equivalence
- Interterminology Equivalence

Intraterminology Equivalence

Intraterminology equivalence deals with identical concepts (those with the same meaning) within a single terminology. When a new version of a terminology is released, there may be several changes to the representation and meaning of concepts when compared to the previous version. Because there is no way for ETS to automatically determine these changes, by default it treats concepts from the previous and new versions as distinct and unrelated. However, it is possible to explicitly indicate the changes that have occurred between a prior version and a new version in a change file that is loaded with the new version. Using this information, Intraterminology equivalence services can determine whether two concepts from the previous version and the new version have the same meaning.

Given a concept from a version of a terminology, ETS can retrieve equivalent concepts from all contiguous versions that have change files loaded. Given two concepts from different versions of a terminology, ETS can verify if they are equivalent, provided that the more recent version and all the intermediate versions have change files loaded.

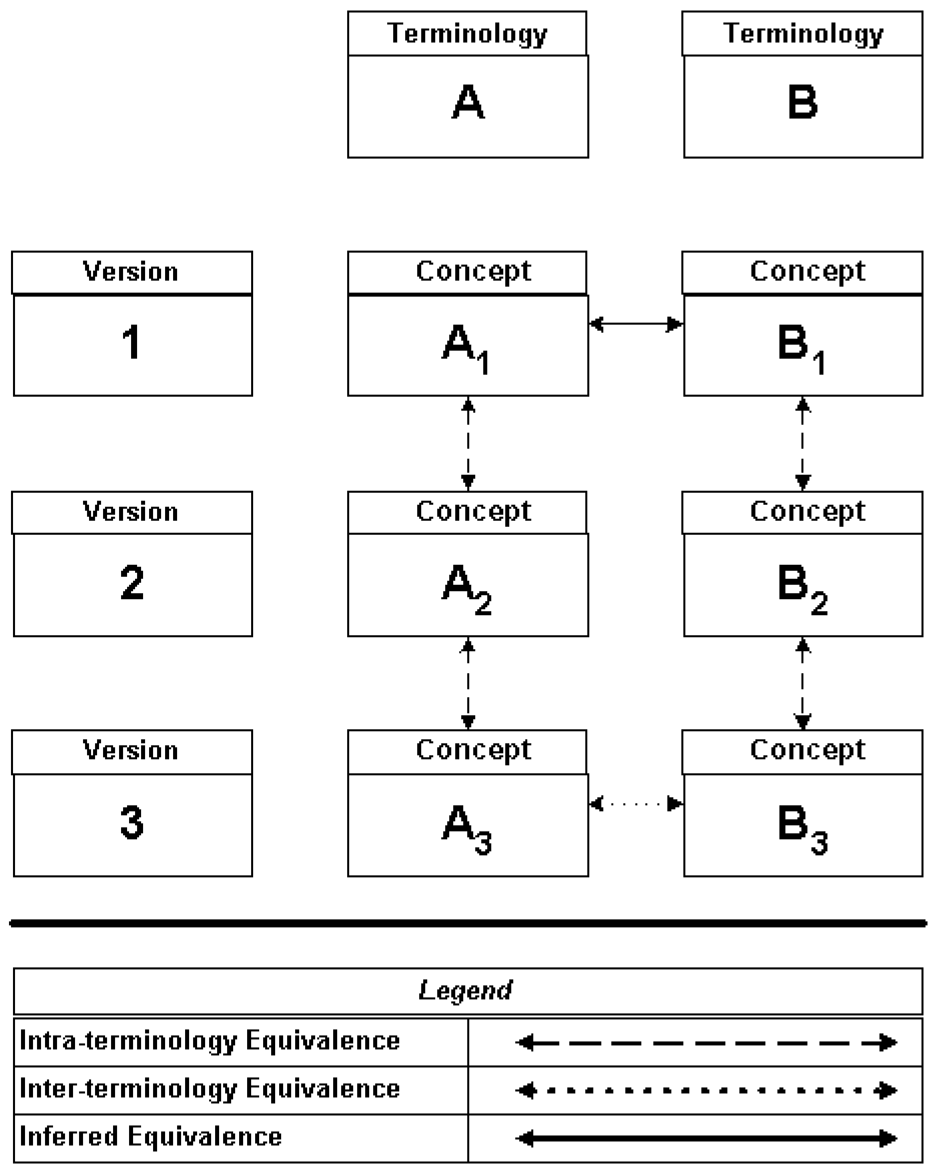
Interterminology Equivalence

Interterminology equivalence deals with identical concepts (those with the same meaning) from different terminologies. Concepts from two different terminologies can vary widely in their granularity and coverage of a domain. Because there is no way for ETS to automatically determine these differences, by default it treats concepts from the two terminologies as distinct and unrelated. However, it is possible to explicitly indicate equivalence between concepts from two versions of different terminologies in the form of an Interterminology Mapping. Using this information, Interterminology equivalence services can determine whether two concepts from different terminologies have the same meaning.

Combining Intraterminology and Interterminology Equivalence

Equivalence between concepts is a transitive relationship. In the following chart, if Concept A1 is equivalent to Concept A2, and Concept A2 is equivalent to Concept A3, it can be inferred that Concept A1 is equivalent to Concept A3. Consistent with this logic, Concept Equivalence services in ETS can determine if concepts from two terminologies are equivalent—provided that an inter terminology mapping exists between versions of the two terminologies, and, change files have been loaded for all versions.

Concept Equivalence Model



In this chart, ETS transitively combines intraterminology equivalence and interterminology equivalence information to infer that Concept A1 is equivalent to Concept B1.

Prerequisites

None

Implementing Intraterminology Equivalence Using Change Files

Change files are used to document the differences between successive versions of a terminology that are loaded into ETS, in a format that is acceptable for loading purposes. Change files are loaded at the same time as the terminology version data using the same loader and importer. Change files contain the following types of information:

- **Reassignment:** The meaning of one concept is occasionally *reassigned* to another concept represented by a different concept code. Following are some of the situations in which reassignment occurs:
  - Duplicate concepts are detected. One of them is elected to continue representing the meaning while the other is retired or deleted and reassigned to the retained concept.
  - A concept is detected to be erroneous. The erroneous concept is retired or deleted and reassigned to a correct concept.
  - The classification of a concept is changed. If the concept codes are hierarchical (as in ICD-9-CM), the change in classification translates to a change in concept code, necessitating reassignment.
  - A reassignment is indicated in a change file by a row containing an entry of type *S* (semantic reassignment), followed by a *source* concept code (the concept whose meaning is being reassigned to another code), and a *target* concept code (the concept whose code now captures the meaning).

  See also:

  - /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme/Change_File_Formats_General.txt from hdr-1.0.0-8.0.08.0.0.zip for more information about the format of the change file.
    If both the source and target concepts of a reassignment are from the new version, the reassignment is said to be *intra-version*. For example, in SNOMED-CT, if a duplicate or erroneous concept is detected, the new version carries forward the duplicate or erroneous concept in an inactive status. The reassignment in this case is from the inactive concept in the new version to an active concept in the new version.

    If the source of a reassignment is from the previous version and the target is from the new version, the reassignment is said to be *inter-version*. For example, in ICD-9-CM, if an erroneous concept is detected, it is deleted and excluded from the new version. A correct concept is provided in the new version and a reassignment is created between the concepts in the previous version and the new version. To process an inter-version reassignment contained in a change file, the ETS importer looks for the source code in the non-quarantined version (of the terminology in question) that has the latest load date. The non-quarantined version can be either *active* or *retired*.
  - Reuse: Occasionally, a concept code used in the previous version is *reused* to represent a concept with a different meaning in the new version. Note that this is considered bad terminology practice and should only be used to account for inadvertent errors. *Unless a reuse is explicitly called out in the change file, a concept in the previous version is always considered equivalent to a concept with the same concept code in the new version.*

Change files are preceded for the following terminologies (no further implementation steps are required):

- HL7
- HDR Supplemental

For each of the following terminologies, new versions and their change files are available to customers, possessing licenses, from the vendor of the terminology:

- IETF RFC 1766
- ISO 3166-1 alpha2

- NUBC-UB92
- CPT-4
- FDB
- HCPCS-2
- ICD-10
- ICD-9-CM-DRG
- ICD-9-CM-MDC
- ICD-9-CM-V1
- ICD-9-CM-V3
- LOINC
- SNOMED-CT

For all other terminologies that are loaded into ETS, change files must be created separately and loaded for each new version as described in the implementation steps.

Note that change files must be loaded concurrently with the new version of a terminology. It is not possible to load a change file for a version of a terminology *after* both versions have been loaded.

Note also that a change file can only equivalence concepts between two consecutively imported versions of a terminology. Hence the order in which versions are imported is significant if change files are being used. The following scenarios illustrate this constraint:

- A concept (concept code X) exists in version 1 of a coding scheme. The concept neither appears in version 2 nor is reassigned to an equivalent version 2 concept. A concept with code X reappears in version 3 with the same meaning as in version 1. It is not possible to indicate to ETS that concept X from version 1 is equivalent to concept X from version 3- because it spans a version.
- A concept (concept code X) exists in version 1 of a coding scheme. The concept neither appears in version 2 nor is reassigned to an equivalent version 2 concept. In version 3, another concept (concept code Y) is created that is equivalent to concept X from version 1. It is not possible to indicate to ETS that concept X from version 1 is equivalent to concept Y from version 3.

Note:

To support concept equivalence, the ETS importer does not import a second quarantined version of a terminology if one already exists. This facilitates verification of equivalence between the quarantined version and the previous version of the terminology before the quarantined version is published.

Steps

1. Determine if the reassignment information for the terminology is *inter-version* or *intra-version.* Use the following rules to make this determination:
   - Intra-version reassignment is used by terminologies that carry forward the duplicate or erroneous concept into the new version, albeit with a retired status, and reassign it to an active concept with a different concept code in the new version.

- Inter-version equivalence should be used for terminologies that do not carry over the concept to be reassigned into the new version. Such terminologies will instead reassign directly from the concept in the previous version to a concept in the new version.

2. Create the change file with the appropriate Reassignment and Reuse entries. See /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme/Change_File_Formats_General.txt from hdr-1.0.0-8.0.0.zip for details about the format of the change file. Note that the **S** ENTRY_TYPE is called *Reassignment* in this guide.
3. Set the HISTORY_TYPE property of the terminology loader control file to INTRAVERSION or INTERVERSION based on the determination made in Step 1. If the HISTORY_TYPE is set to INTRAVERSION, both the SOURCE_CONCEPT_CODE and TARGET_CONCEPT_CODE of a reassignment are assumed to be from the version being loaded. If the HISTORY_TYPE is INTERVERSION, the SOURCE_CONCEPT_CODE is assumed to be from the previous version, and the TARGET_CONCEPT_CODE is assumed to be from the new version.

Note:

If no HISTORY_TYPE is present, the value defaults to NONE and no equivalence information is processed. If the HISTORY_TYPE is INTERVERSION or INTRAVERSION, a change file must be specified using the CHANGE_FILE property. Even if no reassignment or reuse has occurred, an empty change file with the header line must be provided.

4. Specify the location of the change file using the CHANGE_FILE property of the terminology loader control file.
5. Perform the steps described in Loading and Activating New Terminology Versions to load and activate the terminology version along with the change file.

See also:

- Loading and Activating Coding Scheme Versions

Implementing Interterminology Equivalence Using Interterminology Mapping

Interterminology equivalence is implemented by creating cross maps. Cross maps that implement equivalence constitute a special case of *interterminology* mapping. Note that interterminology mapping can also be used to map source and target concepts where the source is semantically broader than or narrower than the target, or where the source and target are related in some other manner.

Cross maps can be used to implement equivalence by specifying that the source and target are equivalent in the *EQUIVALENCECONTEXT* field of the Cross Maps file.

Implementing Interterminology Mapping Using Cross Maps

Interterminology mapping provide a mechanism by which concepts from a source version in one terminology can be mapped to concepts from a target version in another terminology. Mappings are typically tailored for a specific application. For example, a data-aggregating or reporting application may require a mapping between specialized SNOMED-CT codes and coarse ICD-9-CM codes. A data retrieval application may use mappings with the opposite semantics (from less granular classifier codes to more detailed codes). These examples serve to illustrate that mappings serve multiple purposes, and not all cross maps indicate equivalence. Those cross maps that truly do indicate equivalence must be explicitly flagged by the author of the cross maps. This section describes the steps you should follow to indicate equivalence between concepts from two different terminologies using interterminology mapping files.

Perform the following steps to define interterminology mapping using cross maps:

Steps

1. Create Map Set Loader files.
2. Perform the following steps on the file referenced by the CROSS_MAP_FILE property of the Map Set Loader Control File:
   - The eighth column of the cross maps file is called EQUIVALENCECONTEXT. The Map Set Loader inspects this column in each row to determine if the cross map in that row can be used for equivalence. If this column in a particular row is left empty or set to null, the cross map in that row will not be used by ETS Concept Equivalence Services.
   - If the EQUIVALENCECONTEXT column is populated, the cross map is interpreted as indicating equivalence between the concepts represented by the MAPCONCEPTID and the MAPTARGETID.
   - Because determining similarity of meaning between concepts from different terminologies is often subjective, it may not be appropriate to use the same set of cross maps for interterminology equivalence on all occasions. For example, the requirements of a reporting application may be satisfied by a looser definition of equivalence than a clinical order entry application. The EQUIVALENCECONTEXT parameter lets each cross map be associated with the context in which its use is appropriate. At runtime, the EQUIVALENCECONTEXT can be provided as a parameter to ETS Equivalence Services to selectively use only those cross maps that are associated with that context.
   - The default EQUIVALENCECONTEXT is SYSTEM. Cross maps that are flagged with this context will be used by IMP and OMP to determine equivalents in concept lists and the master catalog. If a context is not specified in an equivalence query at runtime, this context is assumed by default.
   - If the same cross map is deemed suitable for multiple contexts, it may be repeated several times in the cross maps file, each time with a different EQUIVALENCECONTEXT.

   See also:

   - /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme/Terminology_File_Formats_MapSet.txt from hdr-1.0.0-8.0.0.zip for Map Set Loader file formats.
3. Load the Map Set files.

Loading the Map Set Files

Perform the following steps to load the map set files:

1. Create a cross map file and a control file in ETS format, and move them into a directory located in the same file system as the Applications instance-a directory that is accessible by the Oracle Database Scheduler (DBMS_SCHEDULER).
   See also:

- Guidelines: Cross Maps
- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip, for details about Cross Map file formats.

2. Use Oracle Database Scheduler (DBMS_SCHEDULER) to load the map set files. Select the HDR Loader Job. Select the coding scheme name called *Map Set Loader*, and enter values for the control file and coding scheme version name.

   See also:

   - *Oracle Applications System Administrator's Guide*

3. After completion of the load, use the HDR Importer Job to import the loaded terminology.
   Enter the load sequence number. You can get this value from the log file of the HDR Loader Job that has successfully loaded the data (coding scheme versions, classifications, and cross maps).

   Select *Off* as the value of dry run mode.

   See also:

   - *Oracle Applications System Administrator's Guide*
   - Appendix D: Running HDR Terminology Jobs

Guidelines: Cross Maps

The ETS Cross mapping model is based on the SNOMED CT cross mapping model. Cross-mapping mechanisms provide support for the following:

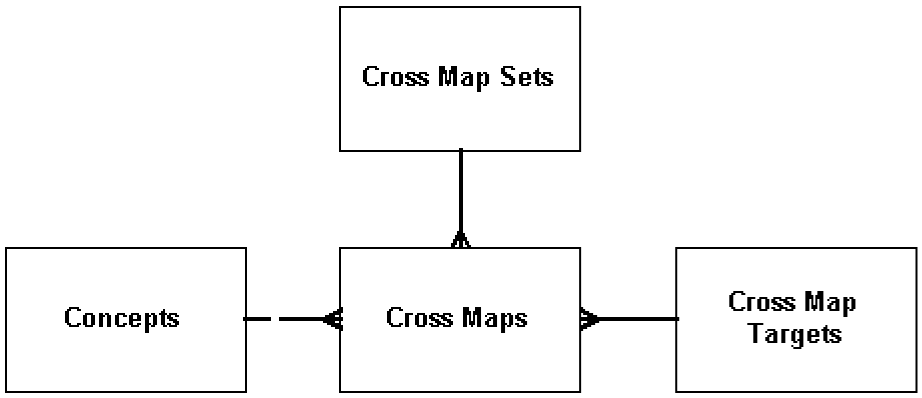
- Mapping a single concept to a target code (a one-to-one mapping).
- Mapping to a set of Target codes (a one-to-many mapping).

The current structure does not support:

- Mapping a set of Concepts to a Target.

The relationship between these tables is shown by the following chart:

Figure 2-3 ETS Cross Mapping Relationship



A map set defines a mapping between two coding scheme versions, such as Terminology A version 2003 and Terminology B version 2.01. Each map set is composed of multiple cross maps. Each cross map consists of a source concept and one or more target concepts, such as a source concept from Terminology A and one or more target concepts from Terminology B—to which it maps.

Loading Cross Maps Provided by the College of American Pathologists

The principal difference between cross map files distributed by the College of American Pathologists (with SNOMED CT) and those expected by ETS loaders is that the SNOMED CT files could contain data regarding multiple map sets in a single file. The map set file may contain multiple rows, each pertaining to a different map set. The cross map file may contain cross maps relating to multiple map sets, and the map targets file may contain targets used by multiple map sets (targets related to multiple coding schemes).

To make the SNOMED CT files suitable for ETS loading, split the files into map sets. The map set file should contain only one row, representing one map set. The cross maps file should contain only rows containing the map set ID of the chosen map set. The map targets file should contain only targets related to the target coding scheme specific to the map set.

See also:

- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme from hdr-1.0.0-8.0.0.zip for the cross-map table structures.

Creating Local Descriptions

You can specify local descriptions for any ETS concept. These descriptions can be used in place of terminology-specified descriptions for display purposes. You can assign a single usage context to each local description.

The usage context for each local description must be unique; no two local concept descriptions share the same usage context.

Note:

Local descriptions of a concept can be created for any language, not just for languages loaded with the coding scheme version.

Use ETS API to create local description. You can create local descriptions for retired or active concepts, but the typical procedure is to create a description for concepts in the active default version of the terminology.

Managing Usage Contexts

Usage Contexts let an HDR solution specify and determine which concept list or local description (of a specific ETS concept) should be used in a given application context. Usage contexts are an attribute of concept lists and local descriptions. A user may specify a usage context when a concept list or a local description is created or later.

When accessing a concept list or local description, HDR solutions may specify a usage context. Based on the specified usage context, HDR retrieves a matching concept list or local description. For example, a Utilization Review department may require diagnoses to be displayed as short names or abbreviations. For this to be implemented in HDR, first a usage context with a name such as Utilization Review must be created. Then, the required local descriptions (short names or abbreviations) with the Utilization Review usage context can be created for the appropriate concepts. Subsequently, applications developed for the department can use ETS APIs with the Utilization Review usage context to display the required local descriptions.

Each local description of a concept must have a single usage context that is unique for that concept. If a local description is assigned a usage context and a local description for that concept already exists with the same usage context, the operation succeeds—but the usage context is removed from the first local description.

A usage context can similarly be used by a concept list. associated with an organization can by an application. For example, a healthcare enterprise might have a concept list of medical services called ENT_MED_SERVICES. A hospital owned by the enterprise may require a specialization of that concept list that contains a subset of the original values. For this to be implemented in HDR, first create a usage context with a name such as Fair Oaks, and associate it with a hospital unit. Then, a specialization of the ENT_MED_SERVICES concept list can be created, with a name such as FAIR_OAKS_MED_SERVICES, and associated with the Fair Oaks usage context.

Use ETS API to manage usage context.

Associating a Usage Context with an Organization

Use the following method to associate a usage context with an organization:

associateUsageContextWithExternalOwner

See also:

- *Oracle Healthcare Data Repository Javadoc*

Implementing Concept Lists

Concept lists group ETS concepts for a variety of purposes, such as displaying certain concepts in user interface drop-down lists and other controls, or constraining values of an attribute to certain concepts.

Concept list member concepts possess activation and retirement dates; and have active, retired, or pending status within a list. Each member concept in a concept list has a code by which it is known in the list (a membership code). The membership code is unique among active or pending members of the concept list, and it can be used by HDR solutions to identify the member concept.

HDR is shipped with a set of pre-defined concept lists. These concept lists are used within HDR to validate attributes that must have coded values, and to process inbound messages. These concept lists have predefined concept names that start with either CTB_ (prior releases) or CL_ (current release).

ETS treats concept lists as of three types based on whether the concept list can be extended or not:

- SYSTEM: HDR is shipped with a set of pre-defined concept lists. These concept lists are used within HDR to validate attributes that must have coded values, and to process inbound and messages. These concept lists have predefined concept names that start with either *CTB_* (prior releases) or *CL_* (current release).

  - Do not use predefined concept lists for any other purpose (other than the defined purpose).
  - Do not use the *CTB_* and *CL_* prefix for concept lists you create.
- EXTENSIBLE: Certain concept lists have been seeded empty-the concept lists are created in the system, but no concepts are added to them. They must be filled by concepts during implementation. Predefined lists that are empty are necessarily extensible. Other predefined lists that have concepts may be extensible also: content may be added to them as required.
- USER:

You can create specializations of concept lists that are defined as EXTENSIBLE. Specializations are child concept lists that initially inherit parent concepts, but may be modified by adding or removing member concepts.

A Specialization can be associated with a usage context, which may in turn be associated with a particular HDR organization.

You can also subset seeded lists, if only a subset of the seeded values are applicable. Because HDR seeded lists are of the type SYSTEM or EXTENSIBLE, and only lists designated USER can be subsetted directly, you must employ indirect methods to subset seeded lists.

Creating and Updating a Concept List

Use ETS API to create a concept list. For more information on Concept Lists API, refer to the *Oracle Healthcare Data Repository Javadoc*.

Adding Concepts to a Concept List

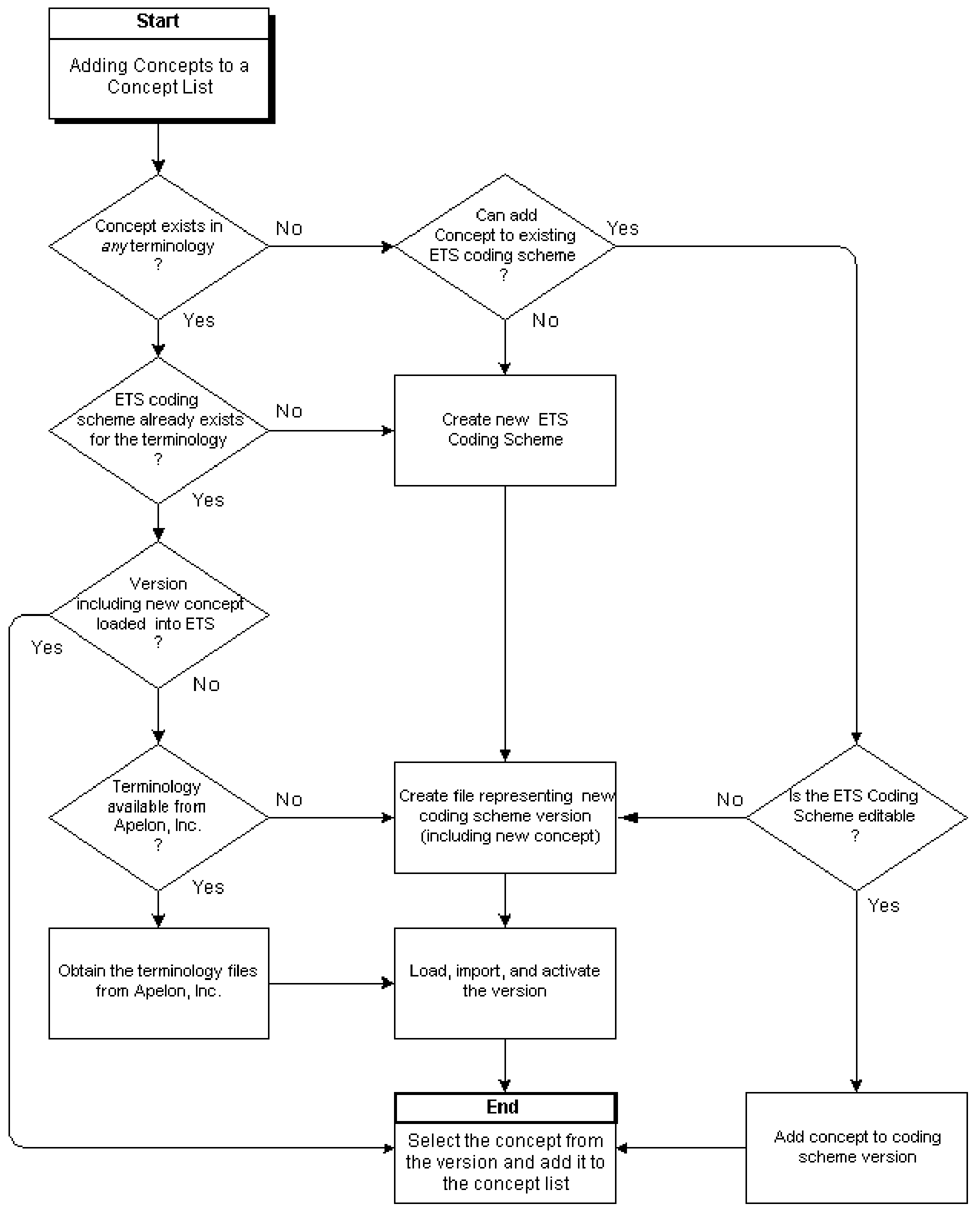
Concepts can be added to extensible concept lists. Concepts to be added to a concept list must be contained in a coding scheme version that has already been loaded into ETS. Determine if the concept exists in ETS and the coding scheme version in which it is contained.

Perform the steps described in this section to add concepts to a concept list.

Caution:

Certain extensible concept lists are empty and must be populated with concepts before using their respective functionality.

Adding Concepts to a Concept List


Start
Adding Concepts to a Concept List
Concept exists in *any* terminology ?
No
Yes
Can add Concept to existing ETS coding scheme ?
Yes
No
ETS coding scheme already exists for the terminology ?
No
Yes
Create new ETS Coding Scheme
Version including new concept loaded into ETS ?
Yes
No
Terminology available from Apelon, Inc. ?
No
Yes
Create file representing new coding scheme version (including new concept)
No
Is the ETS Coding Scheme editable ?
Yes
Obtain the terminology files from Apelon, Inc.
Load, import, and activate the version
End
Select the concept from the version and add it to the concept list
Add concept to coding scheme version

A concept cannot be added under the following conditions: the concept is already active or pending in the selected list; the selected list is the parent of an additive child specialization and the concept is already active or pending on the child; the selected list is a SYSTEM list; or the selected list is a restricted child specialization and the concept is not active or pending on the parent lists.

- If any (but not all) concepts chosen for addition are not addable, a warning diagnostic lists the concepts that will not be added. You can elect to continue or return to list selection.

- If all concepts selected for addition are not addable, the List Selection window is reloaded with an information box listing these concepts. To continue, select another list.

If any of these conditions exist, select a new membership code or activation date as appropriate, and click Next. You can alternatively click Back to return to the List Selection page, or click Cancel to exit the addition process.

Note:

If the concept is being added to a restricted specialization, and the selected activation date would cause the concept's active period to exceed that of the corresponding concept in the parent list, an exception occurs; a dialog warns you that the concept has not been added.

See also:

- *Oracle Healthcare Data Repository Javadoc*

Caution:

We strongly recommend that wherever possible, you only add concepts from the same terminology to a single concept list. Concept meanings can be sensitive to the context in which they are included in a terminology; mixing them with concepts from other terminologies may distort those meanings.

Specializing a Concept List

Concept lists can be specialized. A specialization of a concept list is a child concept list that initially inherits the active members of the parent list. It is a separate concept list, distinct from the parent list. Subsequent behavior of the specialization (a child concept list) with respect to the parent concept list depends upon the setting of its inheritance type:

- *Addition Inheritance:* Any concept added to the parent list is added to the child list.
- *Deletion Inheritance:* Any concept retired from the parent list is retired from the child list.
- *Restricted Inheritance:* A child list cannot contain any concept not contained in its parent list; before a concept is added to the child list it must first either exist in the parent list or be added to the parent list. A concept in a restricted child list also inherits certain changes to the activation and retirement dates of the corresponding concept in the parent list.

The inheritance types are not mutually exclusive. A restrictive list must also exhibit deletion inheritance. You can update a list's addition inheritance and deletion inheritance by turning them on or off, but you cannot update its restricted inheritance.

A specialization can be associated with a usage context, as can a concept local description. A usage context can in turn be associated with an organization. Accordingly, a concept list can have multiple specializations, each associated with a particular organization.

A concept list specialization is created in the same manner as any other concept list (Steps).

Values in a concept list specialization can be added or retired as for any concept list.

Subsetting a Concept List

It may be desirable to subset a concept list—using only a subset of a concept list's members, for UI display purposes, or for validating data to be stored by an application. The following sections describe how to subset a concept list:

Subsetting a User Concept List

A concept list of extensibility type User can be subsetted by retiring unwanted members from the list. A member is retired by updating its retirement date and time.

Note:

This subsetting procedure does not apply to System or System extensible concept lists.

Subsetting a Concept List of any Extensibility Type

A concept list of any extensibility type (including user) can be subsetted using either of two additional procedures. These procedures are especially useful if the list to be subsetted is a System or System Extensible list, from which members cannot be retired (two methods):

Method 1: Using the Core Member Setting of List Members

The core set of members in the list can then be retrieved using the method:

getCoreSet

Checks of individual members of the list can be performed using the method:

isCoreMember

Method 2: Using a Specialization of the Concept List and Retiring Members

Reference:

Oracle Healthcare Data Repository Javadoc

Table 2-4 Service and Methods: Specializing Concept Lists

Level	Detail
Package	oracle.hsgbu.ets.base
Class	ConceptList
Methods	getChildConceptList

To subset a concept list of any extensibility type using Procedure-2, do the following:

Create a specialization of the concept list, and specify a usage context for the child list. You can then subset the child concept list, and you can use the subsetted list as required by your application.

Access the specialization using the method:

getChildconceptList

Implement Security

HDR EJB (Enterprise Java Beans) services require all callers of the service be authenticated by the application server.

WebLogic admin user can access the HDR services. But still it is recommended to create a HDR specific WebLogic user called hdradmin.

WebLogic users can be maintained in the default security provider or any WebLogic server supported providers such as LDAP, that can be configured and used with HDR.

HDR does not require user authorization to access the services. Any authenticated user in any role can access HDR services.

Reference

- WebLogic Server Security Providers

Implementing ETS Classifications

Large amounts of healthcare data are difficult to use unless they are well organized. Creating classifications is the most common means of organizing healthcare data. As institutions generally use a combination of (standard and local) terminologies, classifications need to incorporate concepts from different coding schemes and versions.

The following table summarizes the principal interfaces referenced by this section:

Table 2-5 Service and Methods: ETS Classifications

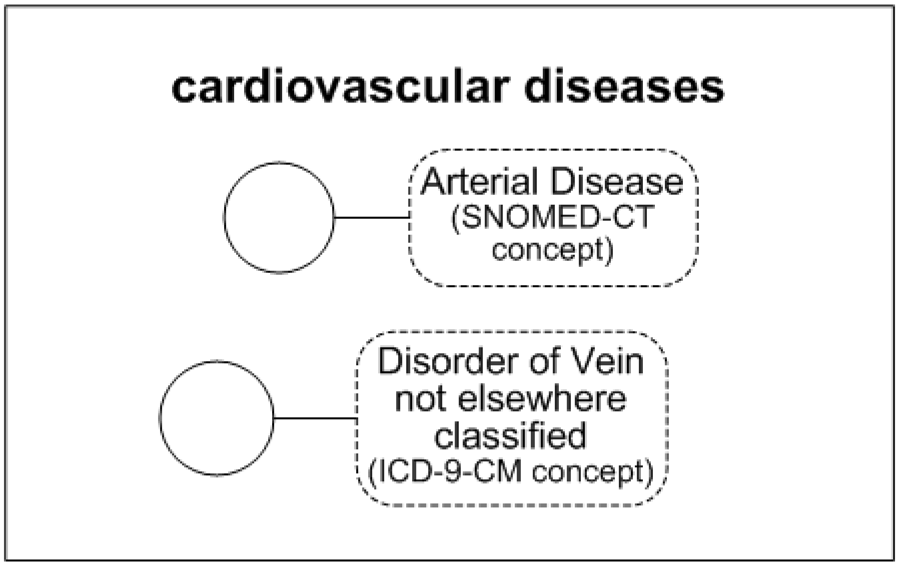
Level	Detail
Package	*oracle.hsgbu.ets.base*
Class	*ETSAdministrationService*
Methods	• *addConceptToConceptList* • *addDeclarationToClassification* • *createClassification* • *createCodingScheme* • *createConceptList* • *removeDeclarationFromClassification*
Class	*ETSService*
Methods	• *findClassificationByCode* • *findcodingScheme* • *findCodingSchemeVersion* • *findConcept* • *findConcepts* • *isEquivalent*
Class	*Classification*
Methods	• *contains* • *findChildClassifications* • *findConceptsInClassification*

ETS Classifications provide a mechanism for grouping concepts from different coding schemes and versions, and arranging the groups in hierarchies navigable by the ETS API. ETS Classifications facilitate:

- Viewing a large number of concepts
- Selection of concepts
- Class-based query of information

For example, a classification called *cardiovascular diseases* could be created and populated with concepts that represent different cardiovascular diseases from different terminologies. The following chart shows this classification:

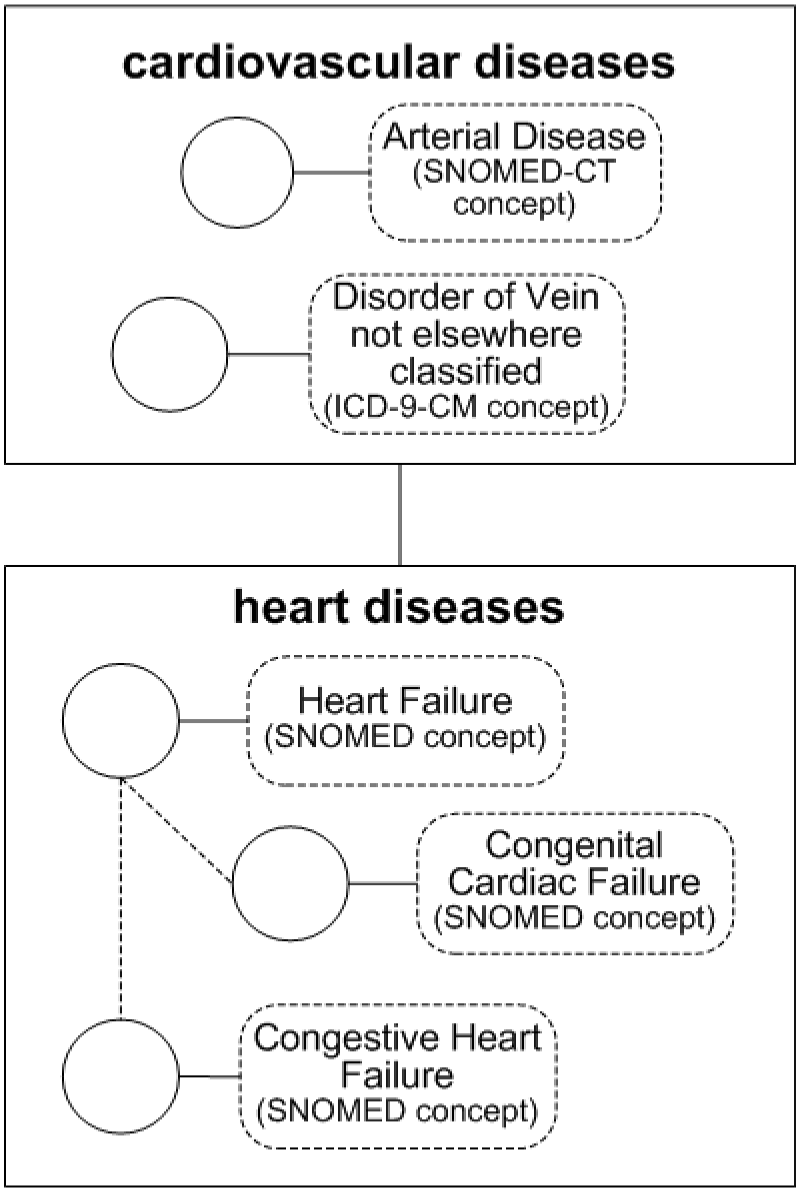
Figure 2-4 Simple ETS Classification


cardiovascular diseases
Arterial Disease (SNOMED-CT concept)
Disorder of Vein not elsewhere classified (ICD-9-CM concept)

ETS Classifications are internally represented as concepts in an editable generic coding scheme called ETSClassification. In this example, the classification, cardiovascular diseases is stored as a concept in the coding scheme ETSClassification.

But you could also create another classification such as heart diseases and make it a child classification of the cardiovascular diseases classification. Thereafter, any concept in the heart diseases classification, such as congestive heart failure, would be considered by the Classifications interface to be implicitly contained in the cardiovascular diseases classification.

Figure 2-5 Parent and Child ETS Classification


cardiovascular diseases
Arterial Disease (SNOMED-CT concept)
Disorder of Vein not elsewhere classified (ICD-9-CM concept)
heart diseases
Heart Failure (SNOMED concept)
Congenital Cardiac Failure (SNOMED concept)
Congestive Heart Failure (SNOMED concept)

ETS Classifications are basically containers of ETS concepts. In this chart, the concepts in each classification node could derive from different coding schemes.

ETS classification interfaces are oblivious to relationships between the classification concepts in their native terminologies. For example, in this chart, congestive heart failure and congenital cardiac failure are children of heart failure in their native terminology, but the heart disease ETS Classification views its contents as a flat list of concepts. There is no classification interface that is aware of relationships between the classification concepts in their native terminologies. However, those relationships can be navigated by querying the individual concepts, using the ETS generic terminology interface or a terminology-specific interface.

Classifications are themselves ETS Concepts – components of a special, pre-defined editable terminology called ETSClassifications. Creating a new classification actually is creating a new concept in the ETSClassifications terminology. Since classifications are concepts:

- A classification may have multiple descriptions, one of which is designated preferred
- Local descriptions may be associated with usage contexts
- A classification has an ETS ID

Classifications can be Linked Hierarchically

Classifications may be linked in strict or multiple hierarchies (a classification may have multiple parent classifications) to form a network of linked classifications forming an acyclic graph.

Testing Containment

ETS classifications support testing of containment across levels of a hierarchy. For example, given the hierarchy in the chart Sample ETS Classification, if the concept congestive heart failure is in the heart disease classification, and the heart disease classification is a child of the cardiovascular diseases classification, you can test if congestive heart failure is a cardiovascular disease, and the answer will be Yes.

Equivalence is incorporated into Classifications.

The concepts contained in an ETS Classification retain their equivalence information. For example, what is actually considered as included in cardiovascular diseases are groups of concepts that are equivalent to heart failure, congenital cardiac failure, congestive heart failure, and arterial anuerysm. Consequently, if concept X is equivalent to congestive heart failure, and the question “Does ‘cardiovascular diseases' contain X?" is asked, the answer will be “yes".

Creating and Populating Classifications

Classifications are created by generating a hierarchical network of classification nodes and populating those nodes with individual ETS concepts. An individual classification node may contain concepts from multiple terminologies.

A single concept can be used in multiple classifications, including multiple sub-classifications of the same parent classification, but it cannot appear more than once in the same sub-classification. However, equivalent concepts can be inserted into the same sub-classification. As for all ETS concepts, concepts in a classification can have multiple text representations.

Classifications may be created and populated by two methods:

- Specifying classification data in files, and loading the files using HDR Terminology Jobs
- Using ETS APIs directly

A new classification will have a pending state when it is created and loaded. It will become effective only when the HDR Maintenance job is run. Until then, the new classification will be unusable. This enables new classifications to be created and readied for use without requiring a downtime.

Whenever changes are made that could affect classification contents (declarations are added or removed, a version of a terminology that contains classification concepts is loaded, or a mapping involving a version that contains classification concepts is loaded), the classification moves from the active state to a dirty state. In this state the classification in its former active state is still usable. The effects of the changes that placed the classification into the dirty state will not be usable until the classification is moved from the dirty state to the active state. Classifications are moved from the dirty state to the active state by running the HDR Maintenance Job.

Note:

Classification contents are defined declaratively. This enables a *short-cut* for adding concepts declared a terminology to be children of another concept. For example, the concepts *heart failure*, *congenital cardiac failure*, and *congestive heart failure* could have been added to *heart diseases* by a single statement *(add heart failure and its descendents to heart diseases).* This adds *heart failure* and its children as defined within its native terminology. A declaration can also add a concept and only those concepts deemed to be direct children in its native terminology, add a concept's descendents but not the concept itself, or add a concept's direct children but not the concept itself. The various insertion choices are called *insert options*.

See also:

- *Oracle Healthcare Data Repository Javadoc*
  A declaration with an insert option of *concept only* adds a single concept ((the concept with none of its children). The contents of a classification amount to a series of declarations. Classification contents are augmented or reduced by adding or removing declarations.

  To retrieve a concept's children when implementing a declaration, the ETS classification build process must know which relationships in the concept's native terminology represent parent-child relationships. Accordingly, for each core terminology, HDR has identified certain relationships as defining parent-child associations. The ETS classification build process queries for such relationships when called upon to find a concept's children. For generic terminologies, the build process queries for relationships in which the relationship type concept is identified as type *IS_PARENT_OF* or *IS_CHILD_OF.*

See also:

- Documentation at the Apelon, Inc. web site for descriptions of the treatment of specific terminologies.
- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme /Terminology_File_Formats_Generic.txt from hdr-1.0.0-8.0.0.zip

Note:

Each ETS concept defined in a generic terminology must be identified as being valid or invalid for use as a relationship type, and is indicated by the *RELATIONSHIPTYPEFLAG* column in the Concepts fie. This flag can contain one of the following values (Table):

Table 2-6 Concept File: RELATIONSHIPTYPEFLAG Legal Values

Value	Description
N	The concept cannot be used as a relationship type.
Y	The concept can be used as a relationship type.
IS_PARENT_OF	The concept can be used as a relationship type, and the type indicates that the source concept is the *parent* of the target.
IS_CHILD_OF	The concept can be used as a relationship type, and the type indicates that the source concept is the *child* of the target.

Those concepts designated as valid relationship types in the concepts file can subsequently be used in the relationships file in the *RELATIONSHIPTYPECONCEPTCODE* column.

Use the following HDR interfaces to define and use ETS classifications:

- *ETSAdministrationService*
- *ETSService*
- *Classification*

Creating a Classification

To create and populate a classification via the loader, perform the following steps:

Steps

1. Create a Classifications file. This file lists the classifications to be created and their properties.
2. Create a Classifications descriptions file: This file lists the descriptions to be associated with the classifications and their properties. Just as multiple descriptions for a concept can be listed in a terminology descriptions file, multiple descriptions can be created for a classification.
   See also:
   - Creating New Generic Coding Schemes
3. Create a Classifications declarations file. This file lists the declarations (each declaration consisting of a concept and an insert option) that will be added to classifications. The classifications referenced in this file can be new classifications listed in the classifications file, or pre-existing classifications.

   Note:

   Declarations must be removed through an API call; they cannot be removed through the loader.
4. Create a control file specifying the Classifications, Classification Descriptions, and Classification Declarations files.
   See also:

- /ets/hdr-ets-1.0.0-8.0.0/db/execute/readme/Terminology_File_Formats_Classifications.txt from hdr-1.0.0-8.0.0.zip for file formats.

5. Load the classification. The procedure for loading a classification is the same as that for loading a Coding Scheme Version. Enter *ETSClassifications* in the Coding Scheme Name field on the Parameters page. Enter any text for the Coding Scheme Version Name (this field must contain text, but the actual text is ignored by the loader).
   See also:
   - Loading a Coding Scheme Version
6. Import the classification. The procedure for importing a classification is the same that for importing a Coding Scheme Version.
   See also:
   - Publishing a Coding Scheme Version
7. Run the Healthcare HDR Maintenance Job to build the classification.

Building a Classification with the HDR Maintenance Job

To build a classification run the HDR Maintenance Job with CLASSIFICATIONS in run mode.

Note:

Run the HDR Maintenance job in the *full* mode whenever a significant amount of new classification data is created (including the first time classifications are created and populated in ETS.

Updating Published Coding Scheme Versions

After a coding scheme version is imported (published), you can update its properties (description, status, and default status) through the ETS API.

Running the HDR Maintenance Job

The HDR maintenance Job performs several database tasks. These tasks include:

- Maintaining the ETS stage tables. For example, cleanup of incomplete or obsolete data in the staged tables.
- Maintaining the ETS data in the active tables. For example, cleanup of failed imports in the active tables.
- Building classifications by processing data for classifications in the pending or dirty state.
- Building and synchronizing intermedia indexes.
- Gathering statistics for the Cost-based Optimizer.
- Ensuring that there is an entry in the language mapping table for each combination of coding scheme version and installed languages.

The maintenance job must be run in either *FULL* mode or *CLASSIFICATIONS* mode to move a classification from the 'dirty' or pending state to the active state. In general, it is a good idea to run the maintenance job periodically to keep ETS running optimally.

Note:

You should run the HDR Maintenance Job in the *FULL* mode:

- Each time you apply a patch to ETS.
- Whenever a significant amount of new classification data is created (including the first time classifications are created and populated in ETS).

Scheduling the Maintenance Job

Steps

Refer to Running HDR maintenance Job from Appendix D: Running HDR Terminology Jobs

Note:

In job arguments, select the desired Run Mode. The available choices are:

- *CLASSIFICATIONS:* Builds classifications by matching definitions of classifications in the *pending* or *dirty* state. Classifications that have been created since the last time the maintenance job was run in CLASSIFICATIONS or FULL mode will be in the pending state. Classifications that have been modified since the last time the maintenance job was run in CLASSIFICATIONS or FULL mode will be in the dirty state. Successfully processed classifications obtain the active state.
- *CLEAN_ACTIVE:* Performs maintenance on the ETS data in the active tables. Removes data from failed imports in the active tables, and rebuilds the intermedia indexes, if necessary. Ensures that there is an entry in the language mapping table for each combination of coding scheme version and installed languages.
- *CLEAN_STAGE:* Performs maintenance on ETS stage tables. Removes incomplete or obsolete data from the staged tables, and rebuilds the intermedia indexes, if necessary.
- *DEFAULT:* Performs maintenance on ETS stage, active, and language mapping tables, but does not build classifications. This mode is composed of CLEAN_ACTIVE and CLEAN_STAGE modes of the maintenance job.
- *FULL:* Performs all operations, including maintenance of stage, active and language mapping tables, and building of classifications. This mode is composed of CLEAN_ACTIVE, CLEAN_STAGE, and CLASSIFICATION modes of the maintenance job.
- *TRUNCATE_STAGE:* Removes all staged contents, regardless of their status. This mode is faster than CLEAN_STAGE for large datasets.

Caution:

The TRUNCATE_STAGE option can cause data that could have been used by the importer to be lost.

See also:

- *Oracle Applications System Administrator's Guide*
- Appendix D: Running HDR Terminology Jobs

Implement Audit Services

HDR Auditing Services lets you log and monitor all HDR activities, to monitor security policy and regulation compliance-by recording actions taken by users during sessions. Such actions could include invoking an API, performing a custom function, or other defined events.

HDR Configuration Manager, a GUI tool, lets security administrators define auditing policies. Implementation of HDR Audit Services includes the following steps:

- Enabling HDR Audit Services

- Initializing existing audit event types
- Creating new audit event types
- Invoking HDR Audit Services

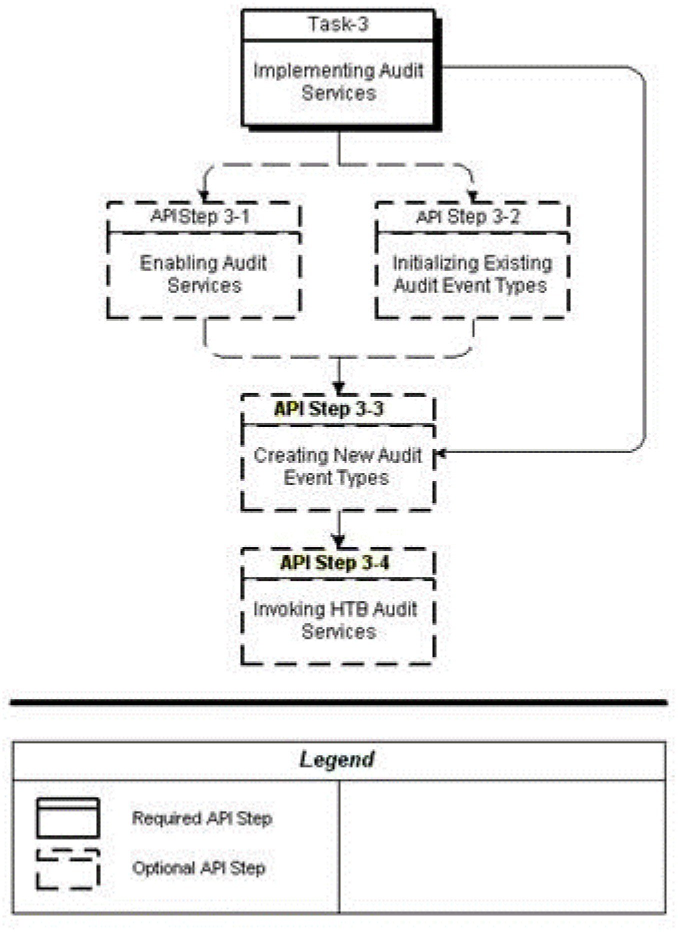
Prerequisites

- Implementing Enterprise Terminology Services

Procedures

The following chart provides an overview of the implementation process for Audit Services:

Figure 2-6 Implementation Process: Audit Services


Task-3
Implementing Audit Services
APIStep 3-1
Enabling Audit Services
API Step 3-2
Initializing Existing Audit Event Types
API Step 3-3
Creating New Audit Event Types
API Step 3-4
Invoking HTB Audit Services
Legend
Required API Step
Optional API Step

To implement Audit Services, refer to the following procedure table:

Task-Step	Description	Optional?
3-1	Enabling Audit Services	Yes

Task-Step	Description	Optional?
3-2	Initializing Existing Audit Event Types	Yes
3-3	Creating New Audit Event Types	Yes
3-4	Invoking HDR Audit Services	Yes

Enabling Audit Services

HDR Audit Services can be enabled (turned on) or disabled (turned off) globally. When enabled, audit events of all seeded and user-defined audit event types can be audited. When disabled, Audit Services is not operative.

Auditing is turned on or off by setting the profile option CTB: Auditing ON to Y or N respectively. By default, CTB: Auditing ON is set to Y on install. Use the ProfileOptionService to update this value. The profile option service API to update this profile option is:

```
ProfileOptionService.setProfileOptionValue
```

Initializing Existing Audit Event Types

Audit event types can selectively be turned on or off. When both the global auditing flag and a particular audit event type are turned on, events of this particular type are audited by HDR Audit Service.

Following is the list of HDR audit event types is seeded for HDR use. By default, these event types are turned on.

1. CTB: Audit Receive Message
2. CTB: Audit Update OID
3. CTB: Audit Query on Personal Health Information
4. CTB: Audit Insert/Update of Personal Health Information

Creating New Audit Event Types

Applications developed on the HDR Platform can define business audit event types in addition to the seeded event types.

For example, an Admitting application might define an audit event type asAdmit Patient, and monitor events of this type.

Note:

Although HDR provides the mechanism to audit business events, it is your responsibility to implement the appropriate audit calls to log such events.

To create a new audit event type, use ProfileOptionService.createProfileOption to create a new profile option with the new audit event type as the profile option code.

Invoking HDR Audit Services

After defining new audit event types, applications can log audit events of these types by calling the Audit Services interface.

Reference

Oracle Healthcare Data Repository Javadoc

Table 2-7 Service and Methods: Audit Services

Level	Detail
Package	oracle.hsgbu.hdr.auditing
Class	AuditService
Methods	createEventLog

Prerequisite

Creating New Audit Event Types

Responsibility

Any responsibility.

Navigation

This is an API-based implementation procedure.

Steps

1. Turn on HDR Audit Services and the audit event type.
   - Enabling Audit Services
   - Initializing Existing Audit Event Types
2. In the application code, call the createEventLog method with the new event type as the value of the EventType attribute. This can be found in *Oracle Healthcare Data Repository Javadoc*.

Attribute Values in Audit Events

Every entry in the audit trail has the attributes listed by the attributes table included in oracle.hsgbu.hdr.auditing.EventLog. This can be found in the *Oracle Healthcare Data Repository Javadoc*.

Implement HDR Object Identifiers

HDR generates a unique identifier for each Act, Entity, and Role persisted through RIMServices. Each identifier is an Instance Identifier data type, consisting of a root that uniquely identifies the HDR instance and an extension that is unique within the HDR instance. Together, they uniquely identify the HDR act, entity, or role. The value used for the root is configured during the setup of HDR and needs to be unique to a given HDR instance to support the sending of HL7 messages between different HDR instances and other systems. If an organization has multiple HDR instances, each instance must have a separate OID. The root is considered the namespace for the HDR instance's identifiers and guarantees the uniqueness of the identifier among all HL7 compliant systems. That is, two objects created in two different systems may have the same extension but can be uniquely identified due to different roots.

Note:

User-defined or externally-supplied instance identifiers may also be persisted for an object. These are in addition to the system generated identifier. The system generated identifier guarantees that each object has at least one unique identifier.

This section describes how to configure the root object identifiers (OIDs) used for the various identifiers created by HDR.

A set of root OIDs must be configured during implementation to enable HDR to generate identifiers. Use the HDR Object Identifiers window to configure the root OID values for various identifier types. The various root OIDs defined using this window are used by HDR as a default root for the identifier, for the object being persisted.

Root OID *values* are not seeded because they are unique to each HDR installation. They are owned by the organization implementing HDR; they are not Oracle registered OIDs. Accordingly, the organization must obtain the OIDs from an appropriate issuing authority, such as HL7 or the standards authority relevant to their country. Alternatively, if the organization already owns an OID, they can use it or define a sub-OID to represent the instance of HDR. In order to enable interoperability with external systems, the root OID values must uniquely identify the HDR instance. Oracle will not supply the OIDs.

See also:

- HL7 Web site: http://www.hl7.org/. Refer to the current version 3 ballot documentation for details on the OID and II data types.
- ISO/IEC 8824 standard: http://www.iso.org/iso/en/ISOOnline.frontpage. Refer to the ISO standard for further details on OIDs.

The set of root OIDs requiring configuration at implementation apply to specific HDR features and need not be configured if the associated features are not required. However, all parts of HDR make use of the RIM Services feature and therefore, the INTERNAL_ROOT must be configured in order to persist HDR objects. It is the root for any HDR internal identifier, which is essentially the primary key of an object. The other root OIDs must be configured if certain EMPI, TCA, Messaging, Financials, Identification, or Migration features are to be used. See the table Optional Root Object Identifiers for further information about how each OID is used.

The user-defined or externally-supplied instance identifiers are not pre-configured in HDR. The root and extension for these external OIDs are sent from external systems or created via an application built on HDR.

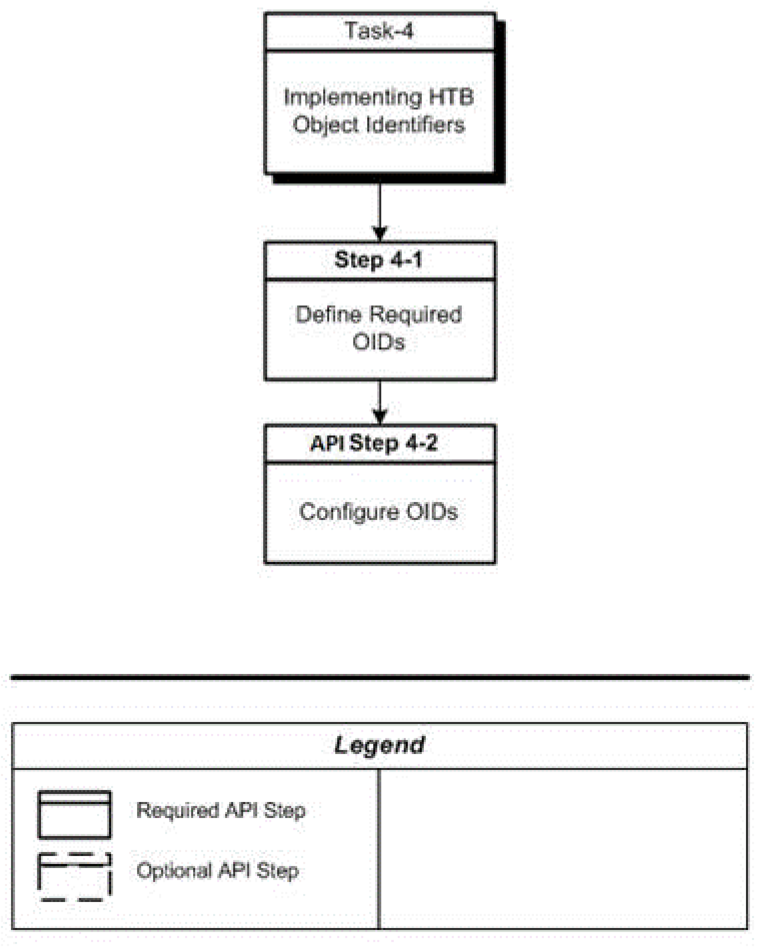
Prerequisites

Obtain the relevant OIDs from the appropriate issuing authority, such as HL7 or the relevant standards authority. Alternatively, if the organization already owns an appropriate root OID (that represents the organization), they can use this OID or issue a sub-OID to represent the instance of HDR.

Procedures

The following chart provides an overview of the implementation process for HDR Object Identifiers:

Figure 2-7 Implementation process for HDR Object Identifiers



Use the OIDService.registerOID API to register new OIDs in HDR.

Implement Profile Option Services

Profile options are configurable preferences that affect the way an Oracle application looks and behaves. System administrators can control HDR behavior by setting profile option values. Application developers can control application behavior by programming their applications to perform in accordance with customized profile option values.

Examples of typical profile options include the following:

- Language: Determines the language in which the application is displayed to users.
- Date Format: Determines the format (mmddyyyy, ddmmyy...) for date displays.

System administrators can set profile options at the following levels:

- User (highest level)
- Site (lowest level)

The profile option values set at each level define runtime values for each user's profile options. An option's runtime value is the highest level setting for that option.

A profile option can be set at more than one level. When a profile option is set to more than one level, an order of precedence applies: Site has the lower priority, which is superseded by User. A profile option value entered at the Site level can be overridden by the value entered at the User level.

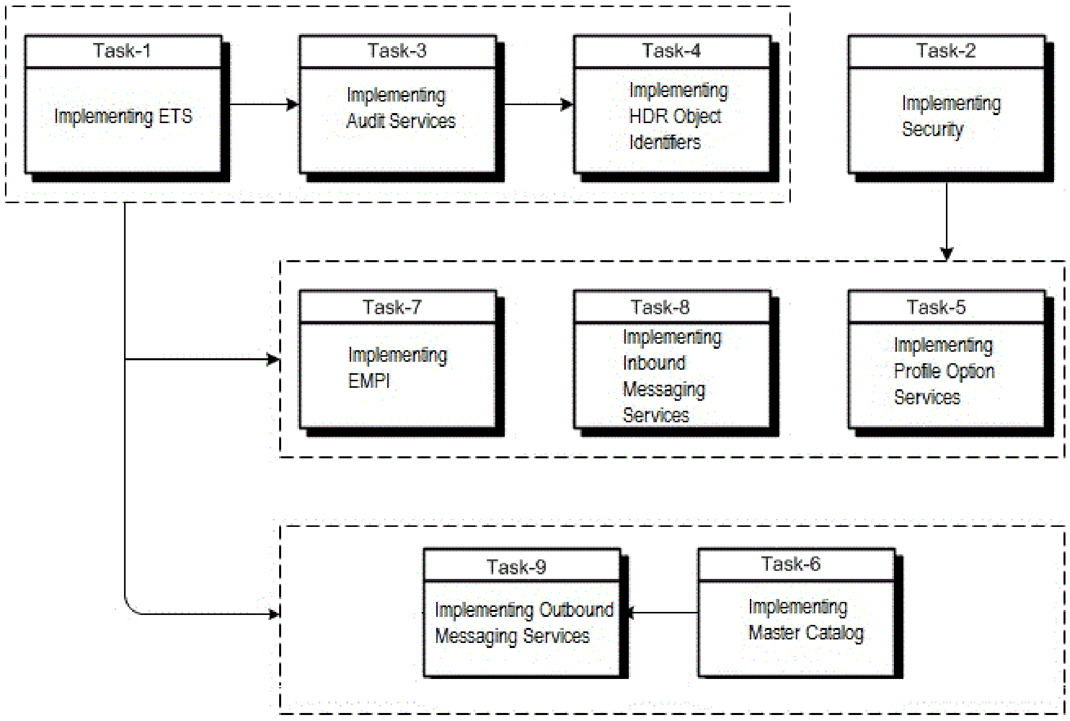
Prerequisites

User Accounts must exist before setting profile options.

Procedures

The following chart provides an overview of the implementation process for Profile Option Services:

Figure 2-8 Implementation Process: Profile Option Services


Task-1
Implementing ETS
Task-3
Implementing Audit Services
Task-4
Implementing HDR Object Identifiers
Task-2
Implementing Security
Task-7
Implementing EMPI
Task-8
Implementing Inbound Messaging Services
Task-5
Implementing Profile Option Services
Task-9
Implementing Outbound Messaging Services
Task-6
Implementing Master Catalog

Implement Profile Options

The HDR ProfileOptionService enables users to manager profile options and its associated values at various profile levels.

Creating Profile Options

Create new profile options using the ProfileOptionService.createProfileOption API.

For example, a physician order entry application could define a profile option that defines the default sort order of patient problem lists (by date, severity,...).

Updating Profile Options

Update profile options using the ProfileOptionService.updateProfileOption API.

Implement Profile Option Values

Create new profile option values for the already created profile options using the ProfileOptionService.setProfileOptionValue API.

Updating Profile Option Values

Update existing profile option values for the already created profile options using the ProfileOptionService.setProfileOptionValue API.

Implement Master Catalog and Focal Class State Transitions

The Healthcare Data Repository (HDR) incorporates two configuration components that provide a repository of Act, Entity, and Role metadata. These two configuration components are the Master Catalog and the Focal Class State Transitions.

You can use a MasterCatalogService APIs to create, update, and search for Master Catalog and Focal Class State Transition entries.

Master Catalog

The Master Catalog defines specific combinations of the *principal* attributes on the Act, Entity, and Role classes.

These principal attributes are sometimes referred to as the *structural* attributes. For acts, the principal attributes are classCode, moodCode, and code. For entities, the principal attributes are the classCode, determinerCode, and code. For roles, the principal attributes are classCode and code, and the principal attributes of the player and scoper entities.

The following are the three main reasons for creating a Master Catalog entry:

- Defining which types of Acts, Entities, and Roles can be persisted to the HDR repository by the Reference Information Model (RIM) Service interfaces. The Enterprise Terminology Services (ETS) repository defines the valid concepts allowed for the coded attributes on the Act, Entity, and Role classes (for example, classCode, moodCode, determinerCode, and code). The Master Catalog provides additional validation, beyond that provided by ETS, by defining the valid *combination* of the Act, Entity, and Role classCode, with the other principal attributes on the object.

See also:

  – Implementing Enterprise Terminology Services
  – Oracle Healthcare Data Repository Programmer's Guide (RIM Services > HDR RIM Services > Using Master Catalog API)

- Distinguishing between multiple similar Acts, Entities, or Roles to provide a unique Master Catalog Id based on the unique concept assigned to the code attribute of the Act, Entity, or Role. The unique Master Catalog Id is referenced by the Act Concept Configuration.
- Defining the specific combinations of the principal attributes of the Acts, Entities, or Roles to enable definition of the side effect processing rules for each unique combination. The unique Master Catalog Id is referenced by the Side Effect Configuration, used in Inbound Message Processing (IMP).
  See also:
  – *Oracle Healthcare Data Repository Programmer's Guide* (HDR Messaging > HDR Inbound Message Processor > IMP Configuration API Usage > Sender Side Effect Configuration Attributes)

The Master Catalog is a required HDR component and must be installed and configured before the use of the HDR RIM Services, IMP, and OMP interfaces. Each combination of the principal attributes for the Acts, Entities, and Roles required by an HDR solution must have a corresponding entry in the Master Catalog.

Master Catalog is supplied with seeded entries that address a wide range of Act, Entity, and Role requirements. HDR solutions can use the seeded Master Catalog entries, and you can also define new Master Catalog entries that allow additional Acts, Entities, and Roles to be used.

There are three types of Master Catalog entries that are described in the following sections

- Master Catalog Acts
- Master Catalog Entities
- Master Catalog Roles

Master Catalog Acts

In RIM, the Class Code, Mood Code, and Code attributes on an Act specify the type of Act that instance represents. As in the RIM, these attributes in the Master Catalog specify the types of Acts that are used in HDR. Act Master Catalog entries define the specific combinations of the Class Code, Mood Code and Code, which are required by the HDR solution.

The following table describes the Master Catalog metadata that can be defined for Acts:

Table 2-8 Act Master Catalog Attributes

Attribute	Content	Description	Mandatory
Master Catalog Id	Internal identifier generated by the system.	Uniquely identifies each Master Catalog entry. This attribute is not displayed on the Master Catalog - Act, Entity, or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)

Table 2-8 (Cont.) Act Master Catalog Attributes

Entry Type Code	Defaulted to ACT for all Act Master Catalog entries.	Specifies which type of class-Act, Entity, or Role-this Master Catalog entry relates to. This attribute is defaulted for the Master Catalog entry, depending on which screen is used to create the entry. This attribute is not displayed on the Master Catalog - Act, Entity, or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)
Class Code	A valid ETS concept from the ActClass code system.	Specifies the value for the Act.classCode attribute.	Yes
Mood Code	A valid ETS concept from the ActMood code system	Specifies the value for the Act.moodCode attribute.	Yes
Active Flag	Valid values are: • ACTIVE • INACTIVE	Determines if this Master Catalog entry is active or inactive.	Yes
Code Type	Valid values are: • ANY • ID • NULL	Indicates how the Code attribute of the Master Catalog entry is used. • ANY indicates that the Act.code attribute can be any valid code. • ID indicates that the Act.code attribute must be the specified code (as defined in the following Code attribute). • NULL indicates that the Act.code attribute must be null.	Yes
Code System Name, Version Name, Code	A valid ETS code, version, and code system if the Code Type is ID, or null if the Code Type is NULL or ANY.	Specifies the value for the Act.code attribute. The value defined for the Act.code includes the specific code system, version, and code that have been defined, and any equivalence defined for that code system/version/ code.	Yes, if Code Type is ID. Should be null if Code Type is NULL or ANY.
Confidentiality Code	A valid ETS concept or concepts from the Confidentiality code system.	Allows you to assign one or more confidentiality codes to a Master Catalog entry, to specify the type of confidentiality associated with these types of Acts. The Confidentiality Code may be used when querying the HDR repository, to mask specific information on the act. **Note:** The Master Catalog Confidentiality Code attribute is independent of the Act.confidentialityCode attribute. Both attributes can be used to store the same or different confidentiality codes. Both attributes can be used when querying the HDR repository, to mask specific information on the act.	No

See also:

- *Oracle Healthcare Data Repository Javadoc*, MasterCatalog interface
- Sample Act Master Catalog entries (from the seeded data supplied with HDR) are given in the Sample Act Entries table.

Concept Equivalence Support

The Master Catalog incorporates concept equivalence. Concept equivalence considers all concepts that are equivalent to the concept used in the Master Catalog entry, as also being covered by that Master Catalog entry. For example, an OBS.EVN Master Catalog entry is defined with the Code attribute valued with 364075005 (SNOMED-CT, Version 0607CORE) for heart rate. Subsequently, a new version of SNOMED-CT is released with a different concept representing heart rate. When using concept equivalence, it is not necessary to modify or append the existing Master Catalog entry. If the concept equivalence defines the equivalence between the two concepts for heart rate, Master Catalog will consider as valid any OBS.EVN with either of the two concepts, though the Master Catalog entry itself only contains the Version 0607CORE concept. This approach eliminates the need to update the Master Catalog when a new coding scheme or version is loaded in ETS, provided equivalence information is also loaded with the new coding scheme or version.

Using the Code Type Attribute for Acts

The Code Type attribute in the Master Catalog specifies the values permitted for the Code attribute on the Act instance. You can choose to do any one of the following:

- Allow a specific Code value for a given Class Code and Determiner Code (Code Type ID)
- Allow any Code value for a given Class Code and Determiner Code (Code Type ANY)
- Constrain the Code attribute so it must be null (Code Type NULL)

Code Type ID

If the Code Type attribute of the Act Master Catalog entry is ID, the Class Code, Mood Code, and Code attributes of the entry specify which Acts are covered by this Master Catalog entry. The Code attribute of the Act (Act.code) must have the specified value, or its equivalent.

The following table displays a sample Act entry with Code Type attribute as ID:

Table 2-9 Sample Act Entry with Code Type ID

Class Code	Mood Code	Code Type	Code/Code System	Code Description
PCPR	EVN	ID	000928/HDR Supplemental	PCP Assignment

This Act Master Catalog entry encompasses Acts representing a patient care provision event, specifically a primary care provider (PCP) assignment. That is, any Act whose Class Code attribute is PCPR, Mood Code attribute is EVN, and Code attribute is 000928 or an equivalent ETS Concept.

Code Type NULL

If the Code Type attribute of the Act Master Catalog entry is NULL, the Class Code, Mood Code, and Code attributes of the entry specify which Acts are covered by this Master Catalog entry. The Code attribute of the allowed Act (Act.code) must be null.

The following table displays a sample Act entry with Code Type attribute as NULL:

Table 2-10 Sample Act Entry With Code Type NULL

Class Code	Mood Code	Code Type	Code/Code System	Code Description
PCPR	RQO	NULL	-	-

This Act Master Catalog entry encompasses Acts representing a request for patient care provision, with no further specification. That is, any Act whose Class Code attribute is PCPR, Mood Code attribute is RQO, and Code attribute is null.

Code Type ANY

If the Code Type attribute of the Act Master Catalog entry is ANY, the Class Code, Mood Code, and Code attributes of the entry specify which Acts are covered by this Master Catalog entry. The Code attribute of the allowed Act (Act.code) can have any value except null.

The following table displays a sample Act entry with Code Type attribute as ANY:

Table 2-11 Sample Act Entry With Code Type ANY

Class Code	Mood Code	Code Type	Code/Code System	Code Description
PCPR	RQO	ANY	-	-

This Act Master Catalog entry encompasses Acts representing a request for patient care provision, with further specification. That is, any Act whose Class Code attribute is PCPR, Mood Code attribute is RQO, and Code attribute is any valid ETS concept. Acts that have a null Act.code are not covered by this Master Catalog entry.

The following table displays sample Act Master Catalog entries from the HDR seed data:

Table 2-12 Sample Act Entries

Class Code	Mood Code	Code Type	Code/Code System	Code Description
PCPR	RQO	ANY	-	-
PCPR	RQO	NULL	-	-
PCPR	EVN.CRT	NULL	-	-
PCPR	EVN.CRT	ANY	-	-
PCPR	EVN	NULL	-	-
PCPR	EVN	ANY	-	-
PCPR	EVN	ID	000928/HDR Supplemental	PCP Assignment
INC	EVN	NULL	-	-
INC	EVN	ANY	-	-

Note:

The Confidentiality Code column has not been displayed as no data is seeded for this attribute.

Master Catalog Entities

In the RIM, the Class Code, Determiner Code, and Code attributes on an Entity specify the type of Entity that instance represents. As in the RIM, these attributes in the Master Catalog specify the types of Entities that are used in HDR. Entity Master Catalog entries define the specific combinations of the Class Code, Determiner Code and Code, which are required by the HDR solution.

The following table describes the Master Catalog metadata that can be defined for Entities:

Table 2-13 Entity Master Catalog Attributes

Attribute	Content	Description	Mandatory
Master Catalog Id	Internal identifier generated by the system.	Uniquely identifies each Master Catalog entry. This attribute is not displayed on the Master Catalog - Act, Entity or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)
Entry Type Code	Defaulted to ENTITY for all Entity Master Catalog entries.	Specifies which type of class-Act, Entity, or Role-this Master Catalog entry relates to. This attribute is defaulted for the Master Catalog entry, depending on which screen is used to create the entry. This attribute is not displayed on the Master Catalog - Act, Entity, or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)
Class Code	A valid ETS concept from the EntityClass code system.	Specifies the value for the Entity.classCode attribute.	Yes
Determiner Code	A valid ETS concept from the EntityDeterminer code system	Specifies the value for the Entity.determinerCode attribute.	Yes
Active Flag	Valid values are: • ACTIVE • INACTIVE	Determines if this Master Catalog entry is active or inactive.	Yes

Table 2-13 (Cont.) Entity Master Catalog Attributes

Code Type	Valid values are: • ANY • ID • NULL	Indicates how the Code attribute of the Master Catalog entry is used. • ANY indicates that the Entity.code attribute can be any valid code. • ID indicates that the Entity.code attribute must be the specified code (as defined in the following Code attribute). • NULL indicates that the Entity.code attribute must be null.	Yes
Code System Name, Version Name, Code	A valid ETS code, version, and code system if the Code Type is ID, or null if the Code Type is NULL or ANY.	Specifies the value for the Entity.code attribute. The value defined for the Entity.code includes the specific code system, version, and code that have been defined, and any equivalence defined for that code system/version/code.	Yes, if code type is ID. Should be null if Code Type is NULL or ANY.

See also:

- *Oracle Healthcare Data Repository API Documentation*, MasterCatalog interface
- Sample Entity Master Catalog entries (from the seeded data supplied with HDR) are given in the Sample Entity Entries table.

Using the Code Type Attribute for Entities

The Code Type attribute in the Master Catalog specifies the values permitted for the Code attribute on the Entity instance. You can choose to do any one of the following:

- Allow a specific Code value for a given Class Code and Determiner Code (Code Type ID)
- Allow any Code value for a given Class Code and Determiner Code (Code Type ANY)
- Constrain the Code attribute so it must be null (Code Type NULL)

Code Type ID

If the Code Type attribute of the Entity Master Catalog entry is ID, the Class Code, Determiner Code, and Code attributes of the entry specify which Entities are covered by this Master Catalog entry. The Code attribute of the Entity (Entity.code) must have the specified value or its equivalent.

The following table displays a sample Entity entry with Code Type attribute as ID:

Table 2-14 Sample Entity Entry with Code Type ID

Class Code	Determiner Code	Code Type	Code/Code System	Code Description
ORG	INSTANCE	ID	RELIG/EntityCode	Religious Institution

This Entity Master Catalog entry encompasses Entities representing an organization instance, where the organization is specifically a religious organization. That is, any Entity whose Class Code attribute is ORG, Determiner Code attribute is INSTANCE, and Code attribute is RELIG or an equivalent ETS Concept.

Code Type NULL

If the Code Type attribute of the Entity Master Catalog entry is ANY, the Class Code, Determiner Code, and Code attributes of the entry specify which Entities are covered by the entry. The Code attribute of the allowed Entity (Entity.code) can have any value except null.

The following table displays a sample Entity entry with Code Type attribute as NULL:

Table 2-15 Sample Entity Entry with Code Type NULL

Class Code	Determiner Code	Code Type	Code/Code System	Code Description
ORG	INSTANCE	NULL	-	-

This Entity Master Catalog entry encompasses Entities representing an organization instance, with no further specification. That is, any Entity whose Class Code attribute is ORG, Determiner Code attribute is INSTANCE, and Code attribute is null.

Code Type ANY

If the Code Type attribute of the Entity Master Catalog entry is ANY, the Class Code, Determiner Code, and Code attributes of the entry specify which Entities are covered by the entry. The Code attribute of the allowed Entity (Entity.code) can have any value except null.

The following table displays a sample Act entry with Code Type attribute as ANY:

Table 2-16 Sample Entity Entry with Code Type ANY

Class Code	Determiner Code	Code Type	Code/Code System	Code Description
ORG	INSTANCE	ANY	-	-

This Entity Master Catalog entry encompasses Entities representing an instance of an organization, with further specification. That is, any Entity whose Class Code attribute is ORG, Determiner Code attribute is INSTANCE, and Code attribute is any valid ETS Concept. Entities that have a null Entity.code are not allowed by this entry.

The following table displays sample Master Catalog Entity entries from the HDR seed data:

Table 2-17 Sample Entity Entries

Class Code	Determiner Code	Code Type	Code/Code System	Code Description
PLC	INSTANCE	ANY	-	-
PLC	INSTANCE	NULL	-	-
PSN	INSTANCE	ANY	-	-
PSN	INSTANCE	NULL	-	-
ORG	INSTANCE	ANY	-	-
ORG	INSTANCE	NULL	-	-
ORG	INSTANCE	ID	RELIG/EntityCode	Religious Institution
CONT	INSTANCE	ANY	-	-
CONT	INSTANCE	NULL	-	-
CONT	KIND	ANY	-	-
CONT	KIND	NULL	-	-

Master Catalog Roles

In the RIM, the Class Code and Code attributes on a Role, and the Class Code, Determiner Code, and Code attributes on the player and scoper Entity, specify the type of Role that instance represents. As in the RIM, these attributes in the Master Catalog specify the types of Roles that are used in HDR. Role Master Catalog entries define the specific combinations of the Class Code and Code, and Entity Class Code, Determiner Code, and Code, which are required by the HDR solution.

The following table describes the Master Catalog metadata that can be defined for Roles:

Table 2-18 Role Master Catalog Attributes

Attribute	Content	Description	Mandatory
Master Catalog Id	Internal identifier generated by the system.	Uniquely identifies each Master Catalog entry. This attribute is not displayed on the Master Catalog - Act, Entity, or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)
Entry Type Code	Defaulted to ROLE for all Entity Master Catalog entries.	Specifies which type of class-Act, Entity, or Role-this Master Catalog entry relates to. This attribute is defaulted for the Master Catalog entry, depending on which screen is used to create the entry. This attribute is not displayed on the Master Catalog - Act, Entity, or Role screens on the user interface.	Not Applicable (This is a system-generated attribute.)
Class Code	A valid ETS concept from the RoleClass code system.	Specifies the value for the Role.classCode attribute.	Yes

Table 2-18 (Cont.) Role Master Catalog Attributes

Role Owner Code	Valid values: • Player • Scoper	Specifies if the Role is owned by an Entity, and if so, whether that owning Entity is the player Entity or the scoper Entity. • Player: Indicates that the owning Entity is the player Entity. The player Entity details should be specified. • Scoper: Indicates that the owning Entity is the scoper Entity. The scoper Entity details should be specified. **Note:** It is valid to specify the Player Entity and/or the Scoper Entity, even if Role Owner Code is NULL. NULL indicates that there is no owning Entity.	No
Active Flag	Valid values are: • ACTIVE • INACTIVE	Determines if this Master Catalog entry is active or inactive.	Yes
Code Type	Valid values are: • ANY • ID • NULL	Indicates how the Code attribute of the Master Catalog entry is used. • ANY indicates that the Role.code attribute can be any valid code. • ID indicates that the Role.code attribute must be the specified code (as defined in the following Code attribute). • NULL indicates that the Role.code attribute must be null.	Yes
Code System Name, Version Name, Code	A valid ETS code, version, and code system if the Code Type is ID, or null if the Code Type is NULL or ANY.	Specifies the value for the Role.code attribute. The value defined for the Role.code includes the specific code system, version, and code that have been defined, and any equivalence defined for that code system/version/ code.	Yes, if code type is ID. Should be null if Code Type is NULL or ANY.
Player Master Catalog	A valid Master Catalog Entity classCode, determinerCode, codeType, and code.	Specifies the player Entity for Roles allowed by this Master Catalog entry. The value must identify an Entity entry in the Master Catalog. If this attribute is null, the Role must not have a player Entity.	Yes, if Role Owner Code is 'Player'
Scoper Master Catalog	A valid Master Catalog Entity classCode, determinerCode, codeType, and code.	Specifies the scoper Entity for Roles allowed by this Master Catalog entry. The value must identify an Entity entry in the Master Catalog. If this attribute is null, then the Role must not have a scoper Entity.	Yes, if Role Owner Code is 'Scoper'

Table 2-18 (Cont.) Role Master Catalog Attributes

Confidentiality Code	A valid ETS concept or concepts from the Confidentiality code system.	Allows you to assign one or more confidentiality codes to a Master Catalog entry, to specify the type of confidentiality associated with these types of Roles. The Confidentiality Code may be used when querying the HDR repository, to mask specific information on the Role. **Note:** The Master Catalog Confidentiality Code attribute is independent of the Role.confidentialityCode attribute. Both attributes can be used to store the same or different confidentiality codes. Both attributes can be used when querying the HDR repository, to mask specific information on the Role.	No

See also:

- *Oracle Healthcare Data Repository Javadoc*, MasterCatalog interface

Using the Code Type Attribute for Roles

The Code Type attribute in the Master Catalog specifies the values permitted for the Code attribute on the Role instance. You can choose to do any one of the following:

- Allow a specific Code value for a given Class Code, player, and scoper Entity (Code Type ID)
- Allow any Code value for a given Class Code, player, and scoper Entity (Code Type ANY)
- Constrain the Code attribute so it must be null (Code Type NULL)

Code Type ID

If the Code Type attribute of the Role Master Catalog entry is ID, the Class Code and Code attributes of the Role entry, and the Class Code, Determiner Code, and Code attributes of the player Entity and scoper Entity, specify which Roles are covered by this Master Catalog entry. The Code attribute of the Role (Role.code) must have the specified value or its equivalent.

The following table displays a sample Role entry with Code Type attribute as ID:

Table 2-19 Sample Role Entry with Code Type ID

Class Code	Code Type	Code/Code System	Code Description	Role Owner	Player Master Catalog Scoper Master Catalog

Table 2-19 (Cont.) Sample Role Entry with Code Type ID

GUAR	ID	001899/HDR Supplemental	Worker's Compensation Guarantor	Scoper	ORG.INSTANCE.ANY PSN.INSTANCE.ANY

This Role Master Catalog entry encompasses Roles representing a guarantor, specifically a workers compensation guarantor, with a playing organization Entity and a scoping person Entity, where the scoping person Entity owns the role. That is, any Role whose Class Code attribute is GUAR and Code attribute is 001899 or an equivalent ETS Concept, which is played by an ORG.INSTANCE with any valid Entity.code, and scoped by a PSN.INSTANCE with a NULL Entity.code, where the PSN.INSTANCE is the owning Entity.

Code Type NULL

If the Code Type attribute of the Role Master Catalog entry is NULL, the Class Code and Code attributes of the Role entry, and the Class Code, Determiner Code, and Code attributes of the player Entity and scoper Entity, specify which Roles are covered by this Master Catalog entry. The Code attribute of the allowed Role (Role.code) must be null.

The following table displays a sample Role entry with Code Type attribute as NULL:

Table 2-20 Sample Role Entry with Code Type NULL

Class Code	Code Type	Code/Code System	Code Description	Role Owner	Player Master Catalog Scoper Master Catalog
BIRTHPL	NULL	-	-	Scoper	PLC.INSTANCE.ANY -

This Role Master Catalog entry encompasses Roles representing a birthplace, with a playing place Entity and a scoping person Entity, where the scoping person Entity owns the role. That is, any Role whose Class Code attribute is BIRTHPLC and Code attribute is null, which is played by a PLC.INSTANCE with any valid Entity.code and scoped by a PSN.INSTANCE with a null Entity.code, where the PSN.INSTANCE is the owning Entity.

Code Type ANY

If the Code Type attribute of the Entity Master Catalog entry is ANY, the Class Code and Code attribute of the Role entry, and the Class Code, Determiner Code, and Code attributes of the player Entity and scoper Entity, specify which Roles are covered by this Master Catalog entry. The Code attribute of the allowed Role (Role.code) may have any value except null.

The following table displays a sample Role entry with Code Type attribute as ANY:

Table 2-21 Sample Role Entries with Code Type ANY

Class Code	Code Type	Code/Code System	Code Description	Role Owner	Player Master Catalog Scoper Master Catalog
COVPTY	ANY	-	-	-	PLC.INSTANCE.NULL -

This Role Master Catalog entry encompasses Roles representing a covered party, with a playing person Entity and a null scoping Entity, where the role is not owned. That is, any Role whose Class Code attribute is COVPTY and Code attribute is any valid ETS Concept, which is played by a PSN.INSTANCE with a null Entity.code, with no scoper entity and no owning entity.

The following table displays sample Master Catalog Role entries from the HDR seed data:

Table 2-22 Sample Role Entries

Class Code	Code Type	Code/ Code System	Code Description	Role Owner	Player Master Catalog Scoper Master Catalog
BIRTHPL	NULL	-	-	Scoper	PLC.INSTANCE.ANY PSN.INSTANCE.NULL
BIRTHPL	NULL	-	-	Scoper	PLC.INSTANCE.NULL PSN.INSTANCE.NULL
BIRTHPL	NULL	-	-	Scoper	- PSN.INSTANCE.NULL
COVPTY	ANY	-	-	-	PSN.INSTANCE.NULL -
COVPTY	NULL	-	-	-	- -
CRINV	NULL	-	-	-	PSN.INSTANCE.NULL -
CRINV	NULL	-	-	-	- -
GUAR	ID	001899/HDR Supplemental	Worker's Compensation Guarantor	Scoper	ORG.INSTANCE.ANY PSN.INSTANCE.NULL
GUAR	ID	001899/HDR Supplemental	Worker's Compensation Guarantor	Scoper	ORG.INSTANCE.NULL PSN.INSTANCE.NULL
GUAR	ANY	-	-	Scoper	ORG.INSTANCE.ANY PSN.INSTANCE.NULL
GUAR	ANY	-	-	Scoper	ORG.INSTANCE.NULL PSN.INSTANCE.NULL
NOK	NULL	-	-	Player	PSN.INSTANCE.NULL PSN.INSTANCE.NULL

Table 2-22 (Cont.) Sample Role Entries

NOK	ANY	-	-	Player	PSN.INSTANCE.NULL PSN.INSTANCE.NULL

Note:

The Confidentiality Code column has not been displayed as no data is seeded for this attribute.

Focal Class State Transitions

The Focal Class State Transitions is a repository of state transitions for the Acts, Entities, and Roles defined in the Master Catalog. The following are the main purposes of the Focal Class State Transitions:

- To define which state transitions can be applied to the HDR repository by the RIM Service interfaces. A state transition is a change in the statusCode (from a beginning state to an ending state) of an Act, Entity, or Role. The ETS repository defines the valid concepts allowed for Act, Entity, and Role statusCode. The HDR Generic State Transitions define the valid transitions for Act, Entity, and Role statusCode. The Focal Class State Transitions provides additional validation, beyond that provided by ETS and the Generic State Transitions, by defining the valid *combination* of the start state and end state for the specific combination of the Master Catalog Act, Entity, or Role and the specific Control Act.
  See also:
  - *Oracle Healthcare Data Repository Javadoc*, for more information on Act, Entity, and Role.
  - *Oracle Healthcare Data Repository Programmer's Guide* (RIM Services > HDR RIM Services > Using Master Catalog)
- To define any business events that should be initiated in the event of that unique state transition.

The repository of Focal Class State Transitions is a required HDR component and must be installed and configured before the use of the HDR RIM Services, IMP, and OMP interfaces. Each unique combination of Control Act, Focal Class, begin state, and end state required for HDR processing must have a corresponding entry in the Focal Class State Transition table. The Focal Class State Transition entries required for Inbound Messaging and other solution areas are supplied as seed data as part of the base HDR platform. You can add entries to this table to meet additional requirements.

Note:

As HDR conforms to the RIM standard, all entries in the Focal Class State Transition table must be a subset of the state transitions prescribed by the RIM.

Each record in the Focal Class State Transition table references the relevant Master Catalog Control Act and Master Catalog Focal Class. Therefore, the relevant Act, Entity, and Role entries must exist in the Master Catalog before the Focal Class State Transition entries are defined for them.

Note:

The Focal Class State Transition table is dependent upon the Master Catalog as it contains references to Master Catalog entries.

See also:

- Overview

Business Events

In the Focal Class State Transition table there is a Business Event attribute, which refers to a process that will be raised or initiated in the event of that state transition. (This attribute is named TriggerEventCode in the HDR API). HDR Applications can use these business events to initiate processes that are dependent on specific focal class state transitions.

For example, an encounter discharge may be defined in the Focal Class State Transition table with the relevant Control Act and Focal Class, and begin state *active* and end state *completed*. The business event code for the discharge business event will then be populated in the Focal Class State Transition table to cross reference the state transition to the business event details.

No default values for the business events are seeded with the base HDR platform data. You must add your own business event codes if you wish to utilize this functionality.

The following table describes the metadata that can be defined for Focal Class State Transitions:

Table 2-23 Focal Class State Transition Attributes

Attribute	Content	Description	Mandatory
Focal Class	Defaulted to the relevant Act, Entity, or Role Master Catalog entry.	Specifies which focal class - Act, Entity or Role - this Focal Class State Transition applies to. This attribute is defaulted for the Focal Class State Transition, depending upon the Master Catalog entry being updated.	Yes
Control Act Code Type and Code	A valid Code and Code Type for a Master Catalog control act. ETS code, version, and code system if the Code Type is ID or null if the Code Type is NULL or ANY.	The Focal Class State Transition must be linked to an existing Master Catalog control act. A control act is represented in the Master Catalog by Class Code CACT and Mood Code EVN. You must identify the unique control act by defining the Code Type and Code attributes of the control act.	Yes

Table 2-23 (Cont.) Focal Class State Transition Attributes

Start State	A valid ETS concept from the ActStatus, EntityStatus, or RoleStatus code systems (depending on the Entry Type of the Master Catalog entry).	The object status at the beginning of the create or update process. Valid inputs for this attribute are specific RIM-defined states (such as 'active', 'completed', and 'suspended'), or generic values of 'null' or 'any'.	Yes
End State	A valid ETS concept from the ActStatus, EntityStatus, or RoleStatus code systems (depending on the Entry Type of the Master Catalog entry).	The object status at the end of the create or update process. Valid inputs for this attribute are specific RIM-defined states (such as 'active', 'completed', and 'suspended'), or generic values of 'null' or 'any'.	Yes
Business Event Code System Name and Code	A valid ETS concept from the defined code system used for business events.	The unique code that is used to define the business event associated with this state transition. This attribute informs HDR to raise a specific business event.	No
Active Flag	Valid values are: • ACTIVE • INACTIVE	Determines if this Focal Class State Transition is active or inactive.	Yes

See also:

- Oracle Workflow

Start State and End State

The start state and end state of the Focal Class State Transitions can be defined with specific values (RIM-defined states, such as *active* and *completed*) or generic values (*null* or *any*). This allows you to define which state transitions are allowed. For example, an entry can allow a start state of *active* and an end state of *completed* for a given control act and focal class. Alternatively, an entry can broadly define the state transitions allowed for a given control act and focal class. For example, from *any* start state to *any* end state, or from a *null* start state to a specific end state of *active*.

The following combinations of start state and end state are valid:

- *any* to *any*
- *null* to *null*
- specific state to specific state
- *null* to specific state

The following combinations of start state and end state are invalid:

- specific state to *any*
- specific state to *null*
- *any* to specific state

The following table displays sample Focal Class State Transitions from the HDR seed data:

Control Act Details	Focal Class Details	Start State	End State
CACT.EVN.ID. REPC_TE002001	Act: PCPR.RQO.ANY	null	active
CACT.EVN.ID. REPC_TE002002	Act: PCPR.RQO.ANY	null	completed
CACT.EVN.ID. REPC_TE002002	Act: PCPR.RQO.ANY	active	completed
CACT.EVN.ID. REPC_TE002003	Act: PCPR.RQO.ANY	active	aborted
CACT.EVN.ID. REPC_TE002003	Act: PCPR.RQO.ANY	suspended	aborted
CACT.EVN.ID. PRPA_TE000001	Entity: PSN.INSTANCE.NULL	null	active
CACT.EVN.ID. PRPA_TE000002	Entity: PSN.INSTANCE.NULL	active	active
CACT.EVN.ID. PRPA_TE000003	Entity: PSN.INSTANCE.NULL	active	inactive
CACT.EVN.ID. PRPA_TE000004	Entity: PSN.INSTANCE.NULL	active	inactive
CACT.EVN.ID. MFFI_TE000101	Role: EMP.ID.001895 Player PSN.INSTANCE.NULL Scoper ORG.INSTANCE.ANY	null	active
CACT.EVN.ID. MFFI_TE000102	Role: EMP.ID.001895 Player PSN.INSTANCE.NULL Scoper ORG.INSTANCE.ANY	active	active
CACT.EVN.ID. MFFI_TE000103	Role: EMP.ID.001895 Player PSN.INSTANCE.NULL Scoper ORG.INSTANCE.ANY	active	nullified
CACT.EVN.ID. MFFI_TE000103	Role: EMP.ID.001895 Player PSN.INSTANCE.NULL Scoper ORG.INSTANCE.ANY	terminated	nullified
CACT.EVN.ID. MFFI_TE000104	Role: EMP.ID.001895 Player PSN.INSTANCE.NULL Scoper ORG.INSTANCE.ANY	active	terminated

Creating Master Catalog Entries

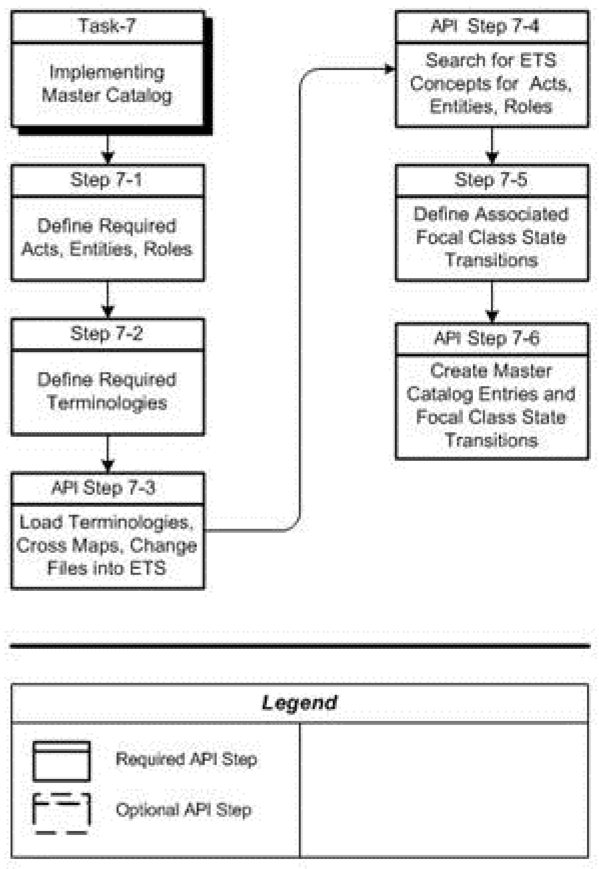
Prerequisites

- Implementing HDR Object Identifiers: The following seeded object identifier must be configured:
  - INTERNAL_ROOT
- Implementing Enterprise Terminology Services

Procedure

The following chart provides an overview of the implementation process for Master Catalog:

Figure 2-9 Implementing Master Catalog


Task-7
Implementing Master Catalog
Step 7-1
Define Required Acts, Entities, Roles
Step 7-2
Define Required Terminologies
API Step 7-3
Load Terminologies, Cross Maps, Change Files into ETS
API Step 7-4
Search for ETS Concepts for Acts, Entities, Roles
Step 7-5
Define Associated Focal Class State Transitions
API Step 7-6
Create Master Catalog Entries and Focal Class State Transitions
Legend
Required API Step
Optional API Step

To implement Master Catalog, refer to the following procedure table:

Table 2-24 HDR Implementation Procedures: Master Catalog

Task-Step	Description	Optional?	Responsibility	Interface
7-1	Define Required Acts, Entities, Roles	No	NA	Analytical Step
7-2	Define Required Terminologies	No	NA	Analytical Step

Table 2-24 (Cont.) HDR Implementation Procedures: Master Catalog

7-3	Load Terminologies, Cross Maps, Change Files into ETS	No	NA	HDR Terminology Jobs
7-4	Search for ETS Concepts for Acts, Entities, Roles	Yes	NA	ETSService APIs
7-5	Define Associated Focal Class State Transitions	No	NA	Analytical Step
7-6	Create Master Catalog Entries and Focal Class State Transitions	No	NA	MasterCatalogService

Setting Up Master Catalog

Perform the following steps to make entries into the Master Catalog:

Steps

For a new implementation of HDR, perform the following steps:

1. Create a comprehensive list of all Acts, Entities, and Roles required by the entire healthcare enterprise.
   This may include Acts, Entities, and Roles that will need to be persisted by RIM Services, and any required to support Act Concept Configuration and Side Effect Configuration.

   Many generic Act, Entity, and Role entries are provided in the Master Catalog as seed data. These can be a starting point for this analysis. Identify any required Act, Role, and Entity classes that are not provided as seed data.

   See also:

   - *Oracle Healthcare Data Repository, Seeded Master Catalog Entries for Version 5*, an Oracle White Paper available on *My Oracle Support*
   - *Oracle Healthcare Data Repository Programmer's Guide* (HDR Messaging > HDR Inbound Message Processor > IMP Configuration > Side Effect Configuration)
2. Analyze the identified acts, roles and entities to determine which terminology best meets enterprise requirements.
   - In some cases, a single standard terminology may be sufficient for a logical group. For example, the LOINC terminology can be used for all laboratory results.
   - In other cases, you may have to extend a standard terminology. For example, LOINC and a local vocabulary could be used for laboratory orders.
   - Less frequently, an unmapped local vocabulary scheme may be required or created for unique needs, such as patient education.
3. Load the core, standard, and local terminologies into ETS that are required for creating the enterprise Master Catalog. Also load the relevant Cross Maps and Change files (which contain the equivalence data).

See also:

- Implementing Enterprise Terminology Services
- Implementing Interterminology and Intraterminology Equivalence

4. Search for the ETS Concepts that are to be used for specific Act, Entity, and Role Master Catalog entries with a Code Type ID and identify their Concept Code, Code System, and Version. For example, to create a Master Catalog Act entry for a specific observation (Code Type is ID) with Act.code defined as SNOMED-CT concept 364075005 (heart rate), you must determine the Concept Code, Code System, and Version for this concept.
5. Determine the Master Catalog attribute values for each Act, Entity, and Role.
   See also:
   - Act, Entity, and Role Master Catalog Attribute tables for information about Act, Role, and Entity attributes
6. Identify focal class state transitions (clinical, administrative and core) relevant to the defined Act, Entity, and Role Master Catalog entries. All focal class state transitions required to support the HDR messaging solutions are provided as seed data. These can be used as a starting point. Identify any additionally required focal class state transitions, which are not already provided as seed data.
   To identify the focal class state transitions do the following:
   - Identify the focal class object, which must be an Act, an Entity, or a Role from the Master Catalog.
   - Identify the appropriate Control Act from the Master Catalog.
   - Identify the required state transitions.

   Caution:

   These state transitions must be a complete set or subset of the valid state transitions defined by HL7. You cannot extend the valid state transitions defined by HL7.
7. Create the Master Catalog (Act, Entity, and Role) entries and Focal Class State Transition entries using the MasterCatalogService API.
   See also:
   - Act Master Catalog Attributes table, for information about Act attributes.
   - Entity Master Catalog Attributes table, for information about Entity attributes.
   - Role Master Catalog Attributes table, for information about Role attributes.

Master Catalog Object Factory

The Master Catalog object factory provides methods for creating two types of objects:

- Master Catalog entries
- Master Catalog focal class state transition entries

Both object types are intended to be passed to the MasterCatalogService.

Implement Inbound Messaging Services

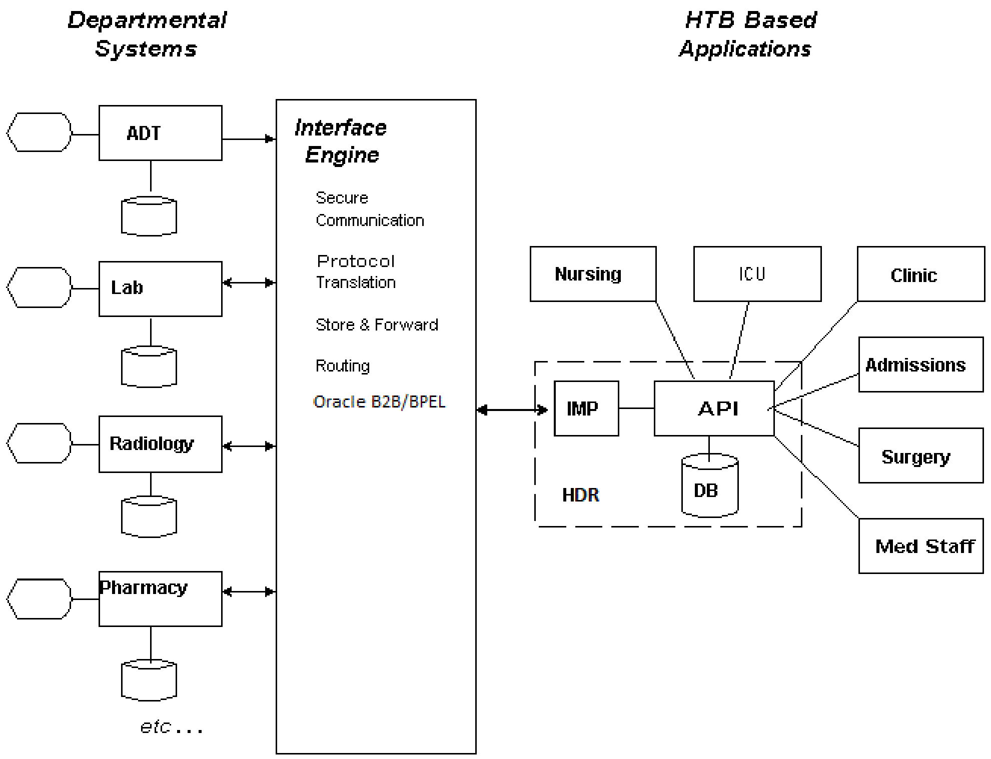
Healthcare enterprises typically operate a number of departmental systems such as ADT, diagnostic departments, pharmacy, and others that may be acquired from multiple vendors. Such systems require messaging services to communicate events and request actions from applications throughout the enterprise.

The two principal components of system messaging are Inbound Messaging Services, described by this section. *Note that you can elect to implement inbound messaging services separately or jointly.*

To route the message from the source system, an external interface engine that handles HL7 message translation and routing must be implemented for IMP (Inbound Message Processor) to function. Although a single interface engine is typically required, multiple interface engines can be implemented. An interface engine is not included with HDR. However, Oracle B2B/BPEL can be used as an Interface Engine.

In the following chart, the ADT system registers and admits patients. After updating its own database, ADT sends an HL7 message to an interface engine that in turn routes the message to HDR and to other systems within the enterprise. HDR maintains this patient data in a clinical data repository, available to HDR-based applications.

Figure 2-10 Typical Application Topology (IMP)



Messaging Schema

HDR includes messaging schemas for all supported message types. Messaging schema includes the following for each message type:

- Schema (XSD Files) for the Payload of the Message Type
- Composite Message Schema (XSD File) for each Interaction ID of the Message Type
- Model Interchange Files (MIF files) for the Payload of the Message Type

In addition, messaging schema contains a common Vocabulary schema and data type schema for all message types.

The Composite Message Schema (CMS) has three parts: Message Wrapper, Control Act Wrapper, and Payload Reference. If there are three Interaction IDs seeded for the same Payload, there will be three composite message schemas; one for each Interaction ID and all of them will refer to the same Payload.

For samples, refer to the schemas for Lab Result available at the following locations:

- Payload Schema for a Message Type (Lab Result)
  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/schemas/
  POLB_MT004000HT01.xsd
- Composite Message Schema for Interaction Ids POLB_IN004003, POLB_IN004004 (Lab Result)
  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/schemas/POLB_IN004003.xsd

  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/schemas/POLB_IN004004.xsd
- Common Data Type Schemas
  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/coreschemas/datatypes.xsd

  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/coreschemas/datatypes-base.xsd
- Common Vocabulary Schemas
  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/coreschemas/datatypes.xsd

  <HDR_DOMAIN>/config/hdr/message/defs/rim214101/coreschemas/datatypes-base.xsd

For more information on message types supported, refer to the *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*.

Acknowledgement Processing

Upon receipt of a message from the sending application (the source of the message), IMP synchronously processes the message into HDR, and returns an Application Acknowledgment (AA), an Application Error (AE), or an Application Reject (AR).

- **Application Acknowledgement (AA):** An AA response indicates that the message was successfully processed and persisted in HDR.
- **Application Error (AE):** An AE response indicates an error reported by HDR, including error information in message content or format (error type code, error detail code, free text). It is the responsibility of the interface engine to determine if the acknowledgement message is returned to the sending system or if the message should be resent to HDR or skipped (abandoning the message). IMP does not support sequence number protocol--the interface engine is responsible for assuring that messages are delivered in order.
- **Application Reject (AR):**An AR response indicates that the message is rejected, for reasons unrelated to its content or format (system or network down, network transmission errors). For most such problems, the receiving system may be able to accept the message at a later time. The sending system or interface engine must decide on an application-specific basis whether the message should be sent again. Ultimately, the AR

is resolved to either an AA (upon successful retransmission) or an AE--which thence generates a call to error processing.

The acknowledgement message contains the following XML segments:

Table 2-25 XML Segments in an Acknowledgement Message

Components	XPATH	Sample values
Acknowledgement Type	MCCI_MT002300HT01.Message/ acknowledgement/typeCode/ @code	`<typeCode code="AA"/> , <typeCode code="AE"/>, <typeCode code="AR"/>`
Acknowledgement Detail Code	MCCI_MT002300HT01.Message/acknowledgement/ acknowledgementDetail/ code/ @code	`<code code="NS250" codeSystemName="AcknowledgementDetailCode"/>`
Acknowledgement Error Text	MCCI_MT002300HT01.Message/acknowledgement/ acknowledgementDetail/ text	`<text mediaType="text/plain" encoding="TXT"> Application: CTB, Message Name: HDR_MS_INVALID_PROCESS_MD_CD. Tokens: PROCESSING_MODE_CODE = T1; </text>`
Acknowledgement Error Location	MCCI_MT002300HT01.Message/acknowledgement/ acknowledgementDetail/ location	`<location> HDR_MS_IMP_EXCEPTION_LOCATION2 :Error occurred while processing XML data located at line 6, column 30. XPATH: /PRPA_IN400000[1] COMPLEX_TYPE: MCCI_MT000100HT04.Message</location>`
HDR Error Code	MCCI_MT002300HT01.Message/acknowledgement/ acknowledgementDetail/text	`CTB_MS_INVALID_PROCESS_MD_CD`

Table 2-25 (Cont.) XML Segments in an Acknowledgement Message

Responder Information	MCCI_MT002300HT01.Message/sender/device/id	`<sender type="CommunicationFunction"> <typeCode code="SND"/> <device type="Device" classCode="DEV" determinerCode="INSTANCE"> <id root="9.989898.5.100" extension="ORG1000"/> <asAgent type="RoleHeir" classCode="AGNT"> <representedOrganization type="Organization" classCode="ORG" determinerCode="INSTANCE"> <id root="9.989898.5.100" extension="ORG1000"/> </representedOrganization> </asAgent> </device> </sender>`

Sender Configuration

Before processing a message, the message type must be configured for the sender. IMP extracts Sender, Receiver, and Interaction Id available in the message, and picks up the associated side-effect configuration. If Interaction ID is not configured for the Sender and Receiver, IMP rejects the message.

Based on the side-effect configuration, IMP sets the reference modifier on RIM objects available in the message. If a particular RIM object is not configured for side-effect, IMP defaults the value of reference modifier for the RIM object to MUST_EXIST. There are certain side-effect rules in IMP that influences the value of reference modifier of a RIM Object. These rules are illustrated in *Oracle Healthcare Data Repository Programmer's Guide*.

Message Validation

In addition to being compliant with messaging schema, IMP imposes certain validations on messages before processing the message. Major validations that affect messages are described in this section.

Identified Object Processing

All RIM objects containing ids are identified objects. If a message instance contains repeating objects with same ids, IMP merges the information of repeating objects and persists union of data from different instances into HDR Repository. This is called *Identified Object Processing*. If the repeating objects in the message contain inconsistent information, IMP rejects the message. For example, if the age of a particular person (having same II) has different values at different segments of the message, IMP rejects the message. For information on complete set of rules to merge information of repeating objects, refer to the *Oracle Healthcare Data Repository Programmer's Guide* and *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*.

Media Type and MIME Type Validation for CDA Messages

For CDA Message Types, IMP supports only certain Media Type and MIME Type. Refer to the CDA Message Type section of the *Oracle Healthcare Data Repository Message Conformance Specification V6.1*.

Master Catalog Validation

Master Catalog entries must exist in HDR Repository for all Acts, Entities, and Roles submitted to HDR for persistence.

Vocabulary Validation

Code System Names used in the message must be loaded into ETS and the Codes used should be part of Coding Scheme.

State Transition Validation

All Acts, Entities, and Roles submitted to HDR for persistence is subjected to Generic State Transition validation. The Focal object in the message is subjected to focal class state transition.

Immutable Attributes Validation

An update message cannot modify values of structural attributes and code (example, act.ClassCode) of an already persisted object.

RIM Service Validation

Every message is persisted as a control act graph in HDR Repository and subjected to the validations done by RIM Persistence Service.

Messaging Metadata

To process a message, IMP needs the following RMIM schematic information about the message elements:

- Name of RIM Foundation Class of the RIM Object available in the message element.
- Type of RIM Association.
- Constrained RMIM Data Type of the attribute.
- If the association is a choice.

The RMIM schematic information is not available in the schemas for message types, but present in the MIF files for the same message type. The information is extracted from the MIF file and loaded into the database after installing HDR. This information is known as *Messaging Metadata*.

To load Messaging Metadata, use ConcurrentProgService.loadMessagingMetadata() API.

Profile Options and System Properties

Use the *CTB: Store Incoming Message* profile option to indicate whether the incoming message has to be stored or not. The valid values are *Y* and *N*. If the value is *Y*, the incoming message is stored in the submission unit table. If the value is *N*, the incoming message is not stored.

The following system properties impacts behavior of the IMP engine:

Table 2-26 IMP System Properties

Property Name	Valid Values	Description
IgnoreUnrecognizedElements	*Y* or *N*	With value 'N' throws an exception when an unrecognized RIM attribute is encountered. With 'Y', just skips it. If *N*, IMP throws an exception when an unrecognized RIM attribute is encounters. If *Y*, IMP skips the validation. The default value is *N*.
IMP_NONDESTRUCTIVE_MODE	*Y* or *N*	If *Y*, IMP rollbacks all transactions and does not update the audit log. The default value is *N*.
IMP_BUNDLED_MODE	*Y* or *N*	If *Y*, IMP collects all non-runtime exceptions, and continues processing. If *N*, each exception aborts processing immediately. The default value is *N*.

See also:

- *Oracle Healthcare Data Repository Programmer's Guide*
- *Oracle Healthcare Data Repository Javadoc*
- *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*

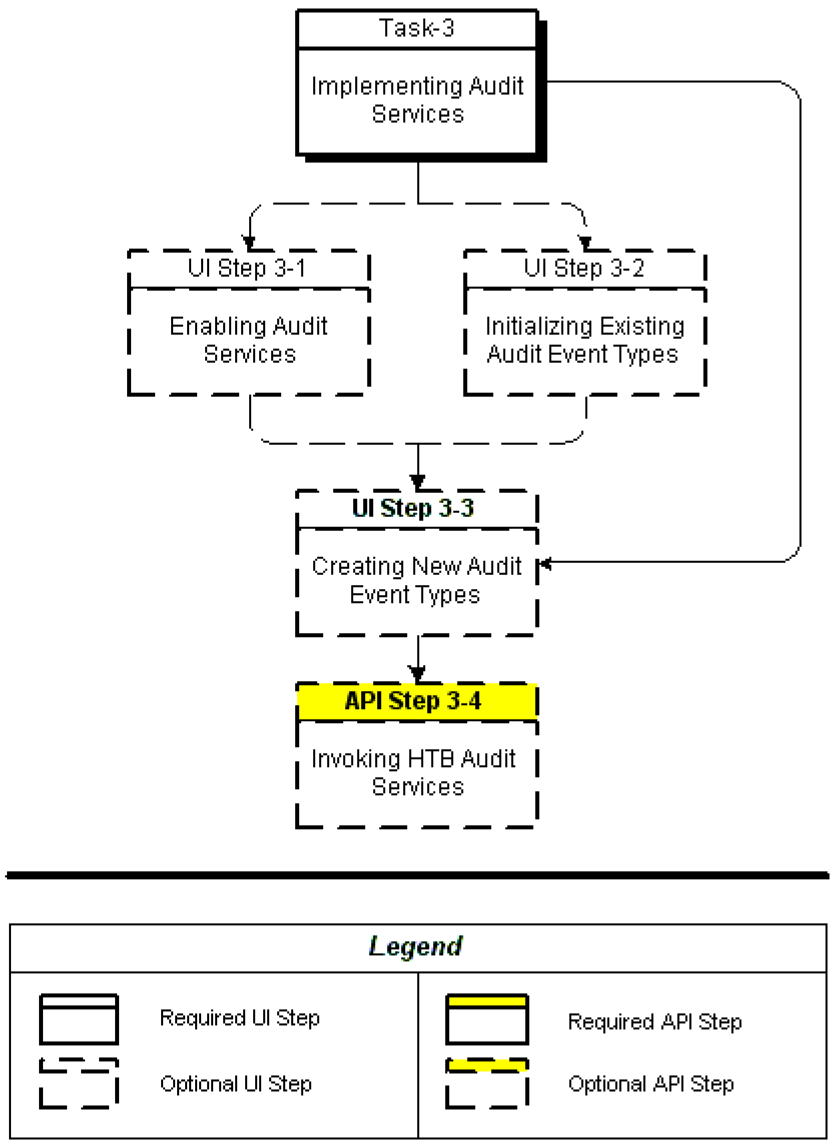
Prerequisites

- Implement HDR Object Identifiers: The following HDR Object Identifiers must be configured:
  – *INTERNAL_ROOT*
  – *CDA_MMID*
- Implement Master Catalog and Focal Class State Transitions: Master Catalog must be implemented before assigning them in Sender and Receiver configuration, and every act, role and entity referenced in an incoming message must have a pre-existing entry in a Master Catalog table.
- Enterprise Terminology Services: Terminologies used in messages should be loaded. Inbound messages can contain coded data types that include a codeSystemName, and code but no codeSystemVersion. In the absence of a version, IMP searches for the code in the default version of the terminology as configured in ETS. If no default version is found the message is rejected. Configuring default versions is critical because some terminologies may reuse codes across versions. To use the concept with the intended meaning, ETS must know explicitly which version to use for a certain terminology.

Procedures:

The following chart provides an overview of the implementation process for Inbound Messaging Services:

Figure 2-11 Implementation Process: Inbound Messaging Services



To implement Inbound Messaging Services, refer to the following procedure table:

Table 2-27 HDR Implementation Procedures: Inbound Messaging Services

Task-Step	Description	Optional?	Interface
9-1	Configuring Interactions	Yes	API
9-2	Configuring Sender, Sender Interaction, and Side Effect	No	API
9-3	Invoke Inbound Messaging Services	Yes	API

Configuring Interactions

IMP extracts Interaction Id and Trigger Event Code from the incoming message and checks whether it is configured or not. The following table lists the location of the parameters in the message

Table 2-28 Location of the Parameters in the Message

Parameter	XPath
Interaction Id	Top Level Element in the message. Example, PRPA_IN400000
Trigger Event Code	PRPA_IN400000/controlActProcess/code/@code

Interaction Ids for all supported message types are seeded. Refer to the *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification* for the list of seeded interaction ids. You can also configure new Interactions Id for supported messages. Use the Interactions window to configure new Interaction Id. For more information on the Interactions window, refer to *Oracle Healthcare Data Repository User Interface Guide*.

When you configure a new Interaction Id, an Interaction schema is generated by the Healthcare Data Repository User Interface and stored at the following location with the name of {InteractionId}.xsd::

<HDR_DOMAIN>/config/hdr/message/defs/customSchema/newMessageType/interaction.

Configuring Sender, Sender Interaction, and Side Effect

IMP extracts the following information (in the table) from the message and validates them for the configuration:

Table 2-29 Information Extracted and Validated by IMP

Parameter	XPath
Interaction Id	Top Level Element in the message. Example, PRPA_IN400000
Sender Root	PRPA_IN400000/sender/device/ id/@root
Sender Extension	PRPA_IN400000/sender/device/id/@extension
Receiver Root	PRPA_IN400000/receiver/device/asAgent/representedOrganization /id/@root
Receiver Extension	PRPA_IN400000/receiver/device/as Agent/representedOrganization/id/@extension
Trigger Event Code	PRPA_IN400000/controlActProcess/code/@code

If the Sender Root and Extension and Receiver Root and Extension is not configured, IMP rejects the message. This configuration thus controls a valid sender and HDR enterprises authorized to send messages. This is called the *Sender Configuration*.

Important: You must only use Organization's external II while creating sender configuration. You must not use any of the Internal IIs that are automatically generated in HDR.

Upon validation of the Sender Configuration, IMP uses its configuration to determine if the Interaction Id is valid for the Sender Configuration. If it is not configured for that Sender Configuration, IMP rejects the message. This configuration thus controls which types of

Interaction Id a sender is permitted to send to a receiver. This is called the *Sender Interaction Configuration*.

Upon validation of the Sender Configuration and Sender Interaction Configuration combination, IMP processes the message payload. The focal object is created or updated in the HDR Repository. However, for non-focal objects, IMP inspects its side effect configuration to determine its behavior. You can configure IMP to let each non-focal object type create or not create the object if *it is not* present in the repository, and to update or overlay or not update or overlay the object *if it is* present in the repository. This configuration of side effects is called the *Side Effect Configuration*.

Use the IMPConfigAdminIntrService to configure sender and side effects.

See also:

- *Oracle Healthcare Data Repository Programmer's Guide*, for more information on Side Effect Processing Rules.
- *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification* for a list of side effect configuration records required for each message type.

Invoke Inbound Messaging Services

Reference

- *Oracle Healthcare Data Repository Javadoc*
- *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*

The following table lists the principal IMP service and methods:

Table 2-30 Service and Methods: IMP

Level	Detail
Package	oracle.hsgbu.hdr.message.improcessor
Class	IMPService
Methods	• processMessage
Class	RawIMPService
Methods	• processMessage
Class	Result
Methods	• getResponseXML • getStatus • getControlActId • getTriggerEvent

Login

This is an API-based implementation procedure.

Navigation

This is an API-based implementation procedure.

Steps

1. Use the Service Locator to access the IMP Service.

> **Note:**
>
> `RawIMPService` is implemented as a container-managed transactions (CMT) bean, and does not create `SubmissionUnit`. Use `RawIMPService` if you want to use the Java Transaction API (JTA) support of IMP.

2. Use the processMessage method with an HDR-compliant message (see following Note) as a parameter to invoke message processing services; a Result object is returned.
3. Use the following methods to inspect the result of processing the message:
   - getResponseXML
   - getStatus

> **Note:**
>
> IMP supports XML formatted inbound messages that conform to the HL7 version 3 messaging standard. The messages must conform to the messaging schema for the message types supported in HDR. The schemas for all supported message type is available at the following location:
>
> - <HDR_DOMAIN>/config/hdr/messge/defs/rim214101/schemas
>
> The list of supported message types is provided in *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*. Using Messaging Tool Kit, additional message types can be supported. For more information, refer to *Oracle Healthcare Data Repository Messaging Tool Kit User Guide*.

See also:

- *Oracle Healthcare Data Repository HL7 Version 3 Conformance Specification*, for information about message types supported by IMP.
- *Oracle Healthcare Data Repository Messaging Tool Kit User Guide.*
- *Oracle Healthcare Data Repository Programmer's Guide.*

Implement Concurrent Program Service

HDR provides a JMS queue based job scheduler service called ConcurrentProgramService, which is exposed on ServiceLocator as a stateless session bean. ConcurrentProgramService is used to schedule the following jobs.

Loading Messaging Metadata

Messaging metadata load can be scheduled as a background job using the ConcurrentProgramService.loadMessagingMetadata() API call. The API returns a request ID that can be used to monitor the job status using the ConcurrentProgramService.getJobStatus() API.

Loading MTK Schema to HDR Server

MTK schemas from a local directory specified in the profile option value for profile code CTB_MTK_SCHEMA_DIR_PATH, can be loaded to HDR server location using the ConcurrentProgramService.loadMTKCustomSchema() API.

Loading MTK Interaction Schema to HDR Server

Composite message schema can be created using the MTK custom interaction schemas using the ConcurrentProgramService.loadMTKCustomInteractionSchema() API.

Implement RIM Service

The RIMService is the main persistence and query service for persistence and retrieval of HL7 V3 RIM data. The HDR RIM Java APIs allows customers to build any HL7 V3 RIM object model to represent any clinical information model and persist clinical data based on the RIM clinical models into HDR.

The Java APIs include implementation of the standard HL7 RIM standard classes, associations and data types based on RIM version 2.14.1.01.

HDR RIM Service supports generic object models constructed using the supported RIM standard and does not constrain implementations to use any specific RIM models.

All RIM objects to be updated must follow the allowed state transitions specified in the supported RIM standard.

Before implementing the RIM Service, the following must be implemented before data can be successfully persisted using the RIM Service APIs:

1. **Load terminologies using ETS**: All coded terminologies to be used in the data to be persisted must be first loaded into the ETS. Please refer to Implementing "Enterprise Terminology Services" section to learn how to load terminologies into ETS.
   See also:
   - *Programmer's Guide* for more details about the ETS Services.
2. **Master Catalog Configuration**: All of the RIM Act/Role/Entity objects must resolve to a valid master catalog id to be successfully persisted in HDR. This master catalog configuration allows implementers to have a fine grained control over what type of clinical data can be persisted in HDR. For example, users can allow only Lab and Medication data to be persisted into HDR and disallow any radiology reports to be persisted in HDR. For example, this will be helpful in building solutions where HDR is one component that is designed to store only Lab and Medication data whereas Radiology reports are designed to be processed by another non-HDR component. Similar master catalog configuration also permits configuring allowed state transitions on focal classes. For more on master catalog configuration, please refer to "Implement Master Catalog and Focal Class State Transitions".

There are several types of update behaviors that can be configured when persisting RIM objects. RIM objects can be configured only to be created OR only updated OR created or updated OR only referenced with no updates. Please refer to `ReferenceModifier.java` in *Javadoc* for more details on the update behaviors supported and their usage.

RIM Service APIs provide a fine grained query API where the users can define their queries in terms of the RIM model and retrieve data out of HDR. The query results will be returned in the RIM Java object models. For more on the RIM Service API usage please refer to the HDR *Javadoc* on the Oracle Help Center (https://docs.oracle.com/health-sciences/health-hdr-81/HDRAP/index.html). The query API supports querying any part of the persisted model, and also supports querying aggregated clinical data based on the query filter criteria, like patient identifier.

There are several types of update behaviors that can be configured when persisting RIM objects. RIM objects can be configured only to be created OR only updated OR created or updated OR only referenced with no updates. Please refer to `ReferenceModifier.java` in *Javadoc* for more details on the update behaviors supported and their usage.

RIM Service APIs provide a fine grained query API where the users can define their queries in terms of the RIM model and retrieve data out of HDR. The query results will be returned in the RIM Java object models. For more on the RIM Service API usage please refer to the HDR *Javadoc* on the Oracle Help Center (https://docs.oracle.com/health-sciences/health-hdr-81/HDRAP/index.html). The query API supports querying any part of the persisted model and also support querying aggregated clinical data based on the query filter criteria like patient identifier.

Generate SQL Queries

The SQL queries generated by the RIM query API can be optimized in three different ways. By default, in all nested select sub queries, the where conditions will be generated as a SQL IN condition. This can be modified by configuring the HDR managed server start-up JVM argument CTB_SUBQRY_OPT_METHOD with one of the values NONE, EXISTS, or JOIN. Based on the database version and configuration, you can choose an option that results in the best database SQL execution plans for the HDR generated SQL queries.

-DCTB_SUBQRY_OPT_METHOD=NONE

This is the default behavior where all nested select sub queries in the where condition will be generated as the SQL IN condition.

-DCTB_SUBQRY_OPT_METHOD=EXISTS

By setting sub-query optimization method to EXISTS, all nested select sub queries in the where condition will be generated as the SQL EXISTS condition.

-DCTB_SUBQRY_OPT_METHOD=JOIN

By setting sub-query optimization method to JOIN, all nested select sub queries in the where condition will be converted to the SQL JOIN condition.

3

Implement the Healthcare Enterprise XDS.b Web Service

- Introducing Cross-Enterprise Document Sharing-b
- Configure Oracle HDR as a Document Repository

HDR implements the Cross-Enterprise Document Sharing-b (XDS.b) specification of the Information Technology Infrastructure (ITI) profile from Integrating the Healthcare Enterprise (IHE) standard.

IHE IT Infrastructure Integration Profiles, offer a common language that healthcare professionals and vendors can use to discuss integration needs of healthcare enterprises and the integration capabilities of information systems in precise terms. Integration Profiles specify implementations of standards that are designed to meet identified clinical needs. They enable users and vendors to state which IHE capabilities they require or provide, by reference to the detailed specifications of the IHE IT Infrastructure Technical Framework.

Integration profiles are defined in terms of IHE Actors and Transactions. Actors are information systems or components of information systems that produce, manage, or act on information associated with clinical and operational activities in the enterprise. Transactions are interactions between actors that communicate the required information through standards-based messages.

For a detailed description of the IHE standard and the various profiles in IHE standard, refer to `http://www.ihe.net`.

Introducing Cross-Enterprise Document Sharing-b

Cross-Enterprise Document Sharing (XDS) enables a number of healthcare delivery organizations belonging to an XDS Affinity Domain (for example, a community of care) to co-operate in the care of a patient by sharing clinical records in the form of documents as they proceed with their patients' care delivery activities. Federated document repositories and a document registry create a longitudinal record of information about a patient within a given XDS Affinity Domain. This profile is based upon Electronic Business using eXtensible Markup Language (ebXML) Registry standards, Simple Object Access Protocol (SOAP), Hypertext Transfer Protocol (HTTP), and Simple Mail Transfer Protocol (SMTP). It describes the configuration of an ebXML Registry in sufficient detail to support Cross Enterprise Document Sharing.

As an XDS.b *Document Repository* actor, HDR is capable of storage and retrieval of electronic health record documents through web services.

The Web services implemented in HDR for supporting Document Repository actor are:

- Provide and Register Document Set-b - ITI-41
- Retrieve Document Set - ITI-43
- Asynchronous Provide and Register Document Set-b
- Asynchronous Retrieve Document Set

Configure Oracle HDR as a Document Repository

- Configure Registry for Accepting Register Document Set-b Web Service Calls
- Configure Repository for Syslog Audit Messages
- Configure Repository Unique ID
- User Creation in 'myrealm' WebLogic Server realm for IHE XDS.b Web Services
- Configure HDR as a Time Client Actor
- Configure HDR a Secure Node Actor
- Configure ITI-42 Timeouts for IHE XDS Registry Web Service Call
- Configure Registry for Accepting Delete Document Set ITI-62 Web Service Calls
- Asynchronous IHE XDS.b Web Services
- Configure HDR To Accept Web Services Invocation Over http Protocol
- Configure Credential Store
- Configure MTOM/ XOP for the ITI-43 Transaction (Retrieve Document Set)

HDR exposes a set of configuration APIs as EJB session beans, to configure the profile options used by HDR as a Document Repository actor. Use the `IHEXDSConfigService` API and its following methods for this purpose:

- `configureRegistryServerDetails(String registryURL)`
- `configureSyslogAuditServerDetails(String syslogServerHost, String syslogServerPort, String transportProtocol)`
- `configureRepositoryUniqueId(String repositoryUniqueId)`
- `configureRegistryAsyncURL(String registryAsyncURL)`

Configure Registry for Accepting Register Document Set-b Web Service Calls

The registry server URL has to be configured for enabling HDR to send Register Document Set-b request web service calls to external document registries.

The registry URL should be a valid web service endpoint URL implementing XDS.b Register Document Set-b specification.

The web service end point URL could be an *http* URL or secure *https* URL. In case of https connections to registry server, the necessary truststore and keystore files need to be generated and configured in <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml under the REGISTRY_AUDIT_SERVER_SSL_CONFIG configuration name.

where, <weblogic_install_dir> is the file path where the WebLogic server has been installed. <weblogic_domain_name> represents the name of WebLogic domain.

Refer to: `http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html` to learn more about digital certificates and using jdk's keytool to generate keystore and truststore.

Configure Repository for Syslog Audit Messages

HDR generates the following Syslog audit event messages:

- Actor Start Audit Event
- Actor Stop Audit Event
- Local user authentication success or failure audit event
- Security Alert Audit Event for change in System configurations
- Register Document Set Audit Event
- Provide and Register Document Set Audit Event
- Retrieve Document Audit Event

HDR can be configured to send audit messages to syslog audit server over UDP or TLS protocol. The Syslog audit messages are per RFC 5424 standard.

You must configure the Syslog audit server host, port number, and transport protocol (UDP or TLS) profile options to enable HDR to send audit messages to Syslog audit server. The Syslog server host and port number should be a valid server host and port number.

Configure Repository Unique ID

You must configure the HDR Document Repository's Unique Id. This id will be added to the Register Document Set-b requests sent out by HDR to external registry. The Repository Unique Id should be a valid OID.

User Creation in 'myrealm' WebLogic Server realm for IHE XDS.b Web Services

Create the user IHE_XDS_USER in WebLogic default server realm'myrealm' for IHE XDS.b Web services.

Configure HDR as a Time Client Actor

Time synchronization is important for security and auditing purpose to provide a synchronized time trail in the logs. It is also important in Web service security where the client can send a timestamp. In this way, no one snooping the traffic on the wire can affect a replay of the packet being sent, as the server will report an error once the timestamp has expired.

The IHE Consistent Time Integration Profile provides a means to ensure that the system clocks and timestamps of many computers in a network are well synchronized. It specifically means the servers and clients in the system must have their system time synchronized with a Time Server. This can be achieved in a Linux server by setting up the Network Time Protocol (NTP) service to synchronize with the Time Server.

Once the machine starts, identify a Time Server with which you need to synchronize your machine, and perform the following steps:

1. Log in as **root**.
2. Change the time zone to your required location. For example, to change to a Central Time Zone, run the command:`ln -sf /usr/share/zoneinfo/CST6CDT /etc/localtime`

3. Edit /etc/ntp.conf by adding the following line, and save it:`server 10.1.1.1` #Any Time Server you want to synchronize with.
4. Check whether the ntpd service is running using the following command:`service ntpd status`
5. If the ntpd service is not stopped, run the command:`service ntpd stop`
6. Make the first-time synchronization using the following command:`ntpdate 10.1.1.1`
7. Make the second and subsequent synchronizations (if required) by using the same command above until the offset shows 0.xxxx or -0.xxxx.
8. Now start the ntpd service using the following command:`service ntpd start`
9. To enable the ntpd service to run all the time even after restart, run the following command:`chkconfig --level 2345 ntpd on`
10. To query the time synchronization status, issue the following command:`ntpq -p -n`
11. Perform the above steps both on the HDR host OS.
12. Shut down the WebLogic Managed Server on which HDR is deployed to, and bring it up after the setup.

Configure HDR a Secure Node Actor

- Set HDR for TLS Communication with Document Source and Document Consumer
- Set Up HDR for TLS Communication with Document Registry and Syslog Audit Server
- Generate Audit Event - OS Level Authentication Events

A Secure Node is a system unit that validates the identity of any user as well as any other node, and determines whether this user is allowed to access the system and exchange information with other nodes or not.

A Secure Application provides security features only for the application features. HDR's IHE XDS.b is a Secure Application. The difference between the Secure Node and the Secure Application is the extent to which the underlying operating system and other environments are secured. A Secure Node includes all aspects of user authentication, file system protections, and operating environment security. The Secure Application is a product that does not include the operating environment.

Set HDR for TLS Communication with Document Source and Document Consumer

HDR requires certificates to be loaded into the Keystore & Truststore of WebLogic Managed Server for TLS communication with Document Source actors and Document Consumer actors.

To configure the Identity and Trust for WebLogic Server, follow the steps provided in the link `http://docs.oracle.com/middleware/1212/wls/SECMG/identity_trust.htm#i1196575`.

Enable SSL to secure communication between client and the HDR application. For configuring the SSL, follow the steps provided in the link `http://docs.oracle.com/middleware/1212/wls/SECMG/ssl.htm#i1194343`. Under Advanced section of SSL configuration, set Hostname Verification to None, enable Use Server Cert, and set Two Way Client Cert Behaviour option to Client Certs Requested and Enforced.

Set Up HDR for TLS Communication with Document Registry and Syslog Audit Server

Edit `<weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml` and enter the absolute paths to trustStore, keyStore and values for trustStorePassword, keyStorePassword, keyStoreType and cipherSuites that will be used for secure TLS communication with Document Registry and Syslog Audit Server under configuration item with name REGISTRY_AUDIT_SERVER_SSL_CONFIG. Enter comma seperated names of the cipher suites to be used for TLS communication for *cipherSuites*.

Configuration parameter *netDebug* can be set to any of the valid values application for JVM argument javax.net.debug

These configuration parameters will be used to set the java runtime arguments javax.net.ssl.trustStore, javax.net.ssl.keyStore, javax.net.ssl.trustStorePassword, javax.net.ssl.keyStorePassword, javax.net.ssl.keyStoreType, javax.net.debug and https.cipherSuites.

Generate Audit Event - OS Level Authentication Events

HDR provides the capability of logging OS Level Authentication Events to qualify as a Secure Node actor.

Perform the following steps on the remote machine where HDR is deployed:

1. Log in to the machine as root user and edit the file /etc/syslog.conf to add the following entry:`authpriv.* |/var/log/syslog_auth.pipe`
2. Create a named pipe 'syslog_auth.pipe' under the /var/log directory by using the mkfifo command, as follows:`> cd /var/log > mkfifo syslog_auth.pipe`
3. Change the owner and group of the pipe to <OS User>. Since the shell script that would read the content from this file will be somewhere in <OS User> home directory, change the owner and group of this file using the following commands:`> chown <OS User> syslog_auth.pipe > chgrp <OS User> syslog_auth.pipe`
4. Restart the syslog service using the following command:`> service syslog restart`
5. „Log in to the machine as <OS User>, and create a script, send_audit_event_for_user_authentication.sh. Copy the following content to that file. Also, ensure that Java 1.7 executable is in the path

```
#!/bin/sh

#update the log file location

  logFile=/home/hiauser/atna.log



  #update the logging.properties file location
```

```
log4JFile=/home/hiauser/logging.properties


transportProtocol=""

wlHost=""

wlPort=""

wlUserName=""

wlPassword=""

keystore=""

keystore_password=""

truststore=""

truststore_password=""


tmpPswd=somePswd

tmpPswdCnfm=somePswd

tmpInput=""




function getPassword {

read -s -p "Please enter $1: " tmpPswd

echo

read -s -p "Confirm password: " tmpPswdCnfm

echo

if [ "$tmpPswd" =  "$tmpPswdCnfm" ]

then

echo

else

echo "Entered password did not match, exiting."

tmpPswd=""

tmpPswdCnfm=""

exit 1;
```

```
  fi

  }


  function getInput {

  for i in {1..3}

  do

  read -p "Please enter $1:" tmpInput

  echo

  if [ "$tmpInput" != "" ]

  then

  break

  fi


  if [ "$i" -eq 3 ]

  then

  echo "Did not get the input, exiting."

  exit 1;

  fi

  done


  }


  #How do you want to send the audit messages to Audit Repository server, over TLS
or UDP?

  getInput "TransportProtocol (UDP or TLS)"

  transportProtocol=$tmpInput


  # WebLogic Host

  getInput "Weblogic Managed Server Host"

  wlHost=$tmpInput
  # WebLogic Port

  getInput "Weblogic Managed Server Port"

  wlPort=$tmpInput


  # WebLogic Admin UserName

  getInput "Weblogic UserName"

  wlUserName=$tmpInput


  # WebLogic Adin User Password

  getPassword "Weblogic Password"

  wlPassword=$tmpPswd


  if [ "$transportProtocol" == "TLS" ]; then

  #The keystore file location.

  getInput "Absolute Path to Keystore File"

  keystore=$tmpInput


  #The keystore password

  getPassword "Keystore Password"

  keystore_password=$tmpPswd


  #The truststore file location.

  getInput "Absolute Path to Truststore File"

  truststore=$tmpInput


  #The truststore password

  getPassword "Truststore Password"

  truststore_password=$tmpPswd

  fi

  #<Weblogic home> - represents the WebLogic home and should be replaced
with the actual file path where WebLogic is installed.
```

```
  #<HDR product install home> - represents the HDR product install home and should
be replaced with the actual file where HDR product is installed.

  CLASSPATH=<Weblogic home>/oracle_common/modules/javax.ejb_3.2.0.jar:<HDR product
install home>/weblogic/jars/hdrclnt.jar:<HDR product install home>/weblogic/jars/
wlfullclient.jar



  if [ "$transportProtocol" == "UDP" ]; then

  JAVA_OPTIONS="-classpath $CLASSPATH -DLogFile=$logFile -
Djava.util.logging.config.file=$log4Jfile -Dwl.user.name=$wlUserName  -
Dwl.user.pass=$wlPassword -Dwl.host.address=$wlHost -Dwl.listen.port=$wlPort"

  else

  JAVA_OPTIONS="-classpath $CLASSPATH -DLogFile=$logFile -
Djava.util.logging.config.file=$log4Jfile -Dwl.user.name=$wlUserName -
Dwl.user.pass=$wlPassword -Dwl.host.address=$wlHost -Dwl.listen.port=$wlPort -
Dkeystore=$keystore -Dkeystore_password=$keystore_password -
Dtruststore=$truststore -Dtruststore_password=$truststore_password"

  fi

  (

  cat < /var/log/syslog_auth.pipe | while read entry

  do

  LoggedInUser=`echo $entry | grep 'Accepted' | awk '{print $9 }'`

  if [ "${LoggedInUser}" != "" ]; then

  java $JAVA_OPTIONS
oracle.apps.ctb.ihe.xdsb.logger.audit.server.TLSSecureNodeAuditLogger $LoggedInUser
 Login



  fi



  LoggedOutUser=`echo $entry | grep 'session closed for user' | awk '{print $11 }'`

  if [ "${LoggedOutUser}" != "" ]; then

  java $JAVA_OPTIONS
oracle.apps.ctb.ihe.xdsb.logger.audit.server.TLSSecureNodeAuditLogger $LoggedOutUse
r Logout



  fi

  done

  ) &
```

6. In the script, update the log4JFile and CLASSPATH variables with appropriate values, and then save it.
7. Change the permissions of the send_audit_event_for_user_authentication.sh file.> `chmod 744 send_audit_event_for_user_authentication.sh`
8. Start the following script and provide the input by following the prompts:> `sh send_audit_event_for_user_authentication.sh`.

   Note:

   The script runs a set of commands in the background mode once you start the script. The script prompts for the following values:

   a. WebLogic Managed Server Host
   b. WebLogic Managed Server Port
   c. WebLogic Admin User Name
   d. WebLogic Admin User Password

   In case the user chooses tranportProtocol as TLS then the script prompts the following additional values:

   a. Absolute Path to Keystore File
   b. Keystore password
   c. Absolute Path to Truststore File
   d. Truststore password

Configure ITI-42 Timeouts for IHE XDS Registry Web Service Call

The IHE XDS configuration file /user_projects/domains//config/hdr/ihe_xdsb_config.xml contains two timeout configurations specific to IHE XDS registry web service call.

The name of the configuration is "WS_CLIENT_CONFIG" which has two components as follows:

- httpConnTimeout
- httpReadTimeout

httpConnTimeout

httpConnTimeout value is configured in milliseconds and this configuration controls how long the HDR will wait for the network connection to be made with the document registry. If a connection to the document registry cannot be established within this time, HDR will fail the entire PnR transaction and return a failure response to the client. The default value for this configuration is 3000 milliseconds. Set this value to a optimum value so that HDR can make a successful network connection with the document registry when available.

httpReadTimeout

httpReadTimeout value is configured in milliseconds and this configuration controls how long the HDR will wait for the document registry to respond with a success or failure. If the document registry does not respond with a response within this time, HDR will fail the entire PnR transaction and return a failure response to the client. The default value for this configuration is 5000 milliseconds. Set this value to a value greater than the configured document registry transaction timeout value (or JTA timeout) and less than the HDR's JTA transaction timeout value.

Add the following configuration in the IHE XDS configuration file:

```
<config name="WS_CLIENT_CONFIG"> <component name="httpConnTimeout"
value="3000" /> <component name="httpReadTimeout" value="5000" /> </config>
```

Configure Registry for Accepting Delete Document Set ITI-62 Web Service Calls

The registry update server URL has to be configured for enabling HDR to send ITI-62 Delete Document Set request web service calls to external document registries.

HDR uses this configuration to delete the ITI-41 document entries from registries if there is any failure in committing the transaction in HDR after getting the SUCCESS response from registries. To do this, HDR uses the RegistryStoredQuery ITI-18 request to get all the registry objects associated with the XDSSubmissionSet uniqueId, and then HDR constructs the ITI-62 Delete Document Set request using all the registry objects received from ITI-18 response and call the registry for deleting them using the XDS registry update URL. Here HDR receives all the registry objects (associations, external identifiers, extrinsic objects, and so on) that are associated with the corresponding XDSSubmissionSet uniqueId using the ITI-18 call. This request will be sent from HDR only when there is a failure in committing the ITI-41 document after getting the SUCCESS call from the registry.

The web service end point URL could be an http URL or secure https URL. In case of https connections to registry server, the necessary truststore and keystore files need to be generated and configured in <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml under the REGISTRY_AUDIT_SERVER_SSL_CONFIG configuration name.

where, <weblogic_install_dir> is the file path where the WebLogic server has been installed. <weblogic_domain_name> represents the name of the WebLogic domain.

For information about digital certificates and using JDK's keytool to generate keystore and truststore, visit `http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html`.

Configure Profile Option for ITI-62 Delete Document Set Transaction

As in synchronous XDS.b web services, HDR exposes a set of configuration APIs to configure profile options. Use the ProfileOptionService EJB session bean to configure the profile options. Before deploying and running synchronous web services, configure the External Document Registry update server endpoint. To send Delete Document Set Transaction ITI-62 request to an external Document Registry actor, HDR should be configured with the Document Registry Update endpoint's URL. This URL must be a valid web service endpoint URL implementing XDS.b UpdateDocumentSet specification.

ProfileOptionService API: `createProfileOption(ProfileOption profileOptionObject)`, `setProfileOptionValue(ProfileOptionValue profileOptionValueObject)`

ProfileOption Configuration Values: ProfileOptionCode: "CTB_XDS_B_REGISTRY_UPDATE_URL" ProfileOptionValue: "http://DOCUMENT_REGISTRY_HOST:PORT/UpdateServiceXYZ" ProfileOptionLevelCode: "SITE" ProfileOptionLevelValue: null

Asynchronous IHE XDS.b Web Services

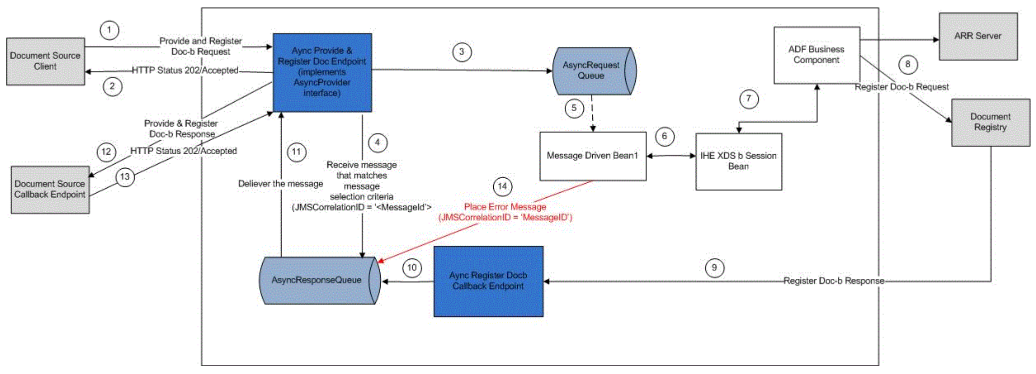
- Configure Profile Options for Asynchronous Web Services
- Configure WLS for Asynchronous Web Services
- Configure Message Receipt Timeout Value for Asynchronous Web Services

HDR provides support for the Asynchronous Web Services Exchange option of the IHE XDS.b specification, specifically for the Document Repository actor. The asynchronous XDS.b profile uses the same set of transactions specified in the XDS.b profile. However, any transaction between two XDS.b actors is now decoupled into two separate one-way transactions - one for request and another for response. Please note the difference that in the synchronous XDS.b profile, both request and response are exchanged between two XDS.b actors as part of a single transaction.

HDR implements the following Asynchronous XDS.b web services:

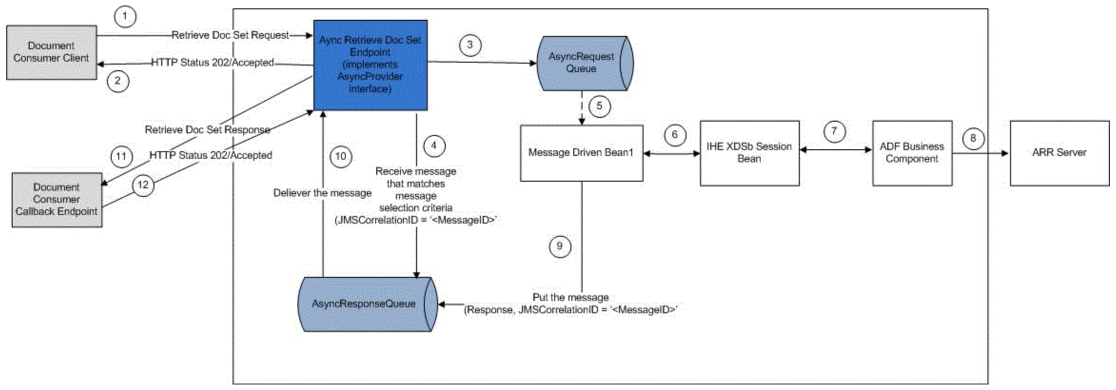
- Asynchronous Provide and Register Document Set-b
- Asynchronous Retrieve Document Set

Asynchronous Provide and Register Document Set-b


Document Source Client
1
Provide and Register Doc-b Request
HTTP Status 202/Accepted
2
Aync Provide & Register Doc Endpoint (implements AsyncProvider interface)
3
AsyncRequest Queue
5
Message Driven Bean1
6
IHE XDS b Session Bean
7
ADF Business Component
ARR Server
8
Register Doc-b Request
Document Registry
Provide & Register Doc-b Response
12
HTTP Status 202/Accepted
13
Document Source Callback Endpoint
11
Deliever the message
4
Receive message that matches message selection criteria (JMSCorrelationID = '<MessageId'>
14
Place Error Message (JMSCorrelationID = 'MessageID')
AsyncResponseQueue
10
Aync Register Docb Callback Endpoint
9
Register Doc-b Response

The following figure illustrates the Asynchronous Provide & Register Doc-b Web Service:

Asynchronous Retrieve Document Set


1
Retrieve Doc Set Request
Document Consumer Client
HTTP Status 202/Accepted
2
Aync Retrieve Doc Set Endpoint (implements AsyncProvider interface)
3
AsyncRequest Queue
5
Message Driven Bean1
6
IHE XDSb Session Bean
7
ADF Business Component
8
ARR Server
Retrieve Doc Set Response
11
HTTP Status 202/Accepted
12
Document Consumer Callback Endpoint
10
Deliever the message
4
Receive message that matches message selection criteria (JMSCorrelationID = '<MessageID>'
9
AsyncResponseQueue
Put the message (Response, JMSCorrelationID = '<MessageID>'

This section contains the following topics:

Configure Profile Options for Asynchronous Web Services

As in synchronous XDS.b web services, HDR exposes a set of configuration APIs, to configure profile options. Use the IHEXDSConfigService EJB session bean to configure the profile options. Before deploying and running Asynchronous web services, configure the Asynchronous External Document Registry endpoint: In order to send asynchronous Register Document Set-b request web service calls to an external Document Registry actor, HDR should be configured with the Document Registry endpoint's URL. This URL must be a valid web service endpoint URL implementing XDS.b Asynchronous Register Document Set-b specification.

API: `configureRegistryAsyncURL(String registryAsyncURL)`

Sample value: http://DOCUMENT_REGISTRY_HOST:PORT/ServiceXYZ

Configure WLS for Asynchronous Web Services

HDR leverages WLS JMS queues to implement Asynchronous XDS.b web services. The WL server where HDR is to be deployed must create the required JMS queues that are used by the asynchronous web services.

HDR's Asynchronous Provide and Register Document Set-b web service uses two JMS Queues:

- **AsyncXDS_PnRbRequestQueue**: All inbound 'Asynchronous Provide and Register Document Set-b' requests are saved in this queue, before getting dequeued and processed.
- **AsyncXDS_PnRbResponseQueue**: All outbound 'Asynchronous Provide and Register Document Set-b' responses are saved in this queue before getting dequeued and transmitted to callback endpoints of respective Document Sources.

HDR's Asynchronous Retrieve Document Set uses two JMS Queues:

- **AsyncXDS_RetrieveDocbRequestQueue**: All inbound Asynchronous Retrieve Document Set requests are saved in this queue, before getting dequeued and processed.
- **AsyncXDS_RetrieveDocbResponseQueue**: All outbound Asynchronous Retrieve Document Set responses are saved in this queue before getting dequeued and transmitted to callback endpoint of the client (An XDS.b Document Consumer).

Configure Message Receipt Timeout Value for Asynchronous Web Services

Edit <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml and enter the timeoutValue under configuration item with name JMS_MESSAGE_TIMEOUT. This timeoutValue represents how long the JMS Message Consumer will wait to receive response message from the AsyncXDS_PnRbResponseQueue or AsyncXDS_RetrieveDocbResponseQueue destinations before the timeout expires. The timeoutValue is in milliseconds, and default value is 180000 milliseconds.

Configure HDR To Accept Web Services Invocation Over http Protocol

By default, HDR installation enables client to invoke Web services over https protocol only. In case the user wants HDR to accept Web services invocation over http protocol, then a custom JVM argument EnableHTTPForWS=true has to be included to the value of the JAVA_OPTIONS variable in <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/bin/startManagedWebLogic.sh. where, <weblogic_install_dir> is the file path where the WebLogic server has been installed. <weblogic_domain_name> represents the name of the WebLogic domain.

Configure Credential Store

The password of IHE_XDS_USER is stored in SecretStore of Oracle Wallet. IHE Service uses the IHE_XDS_USER credentials for user authentication. Edit <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml and enter the absolute path to Oracle Wallet as a value for the credentialStore under configuration item with name CREDENTIAL_STORE_CONFIG.

Configure MTOM/ XOP for the ITI-43 Transaction (Retrieve Document Set)

This configuration controls how HDR returns the MTOM/XOP response for the ITI-43 transaction. Edit <weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/config/hdr/ihe_xdsb_config.xml and enter the optimizeMessage value under configuration item with name MTOM_XOP_CONFIG. Set optimizeMessage value to true to permit HDR to return optimized MTOM/XOP message. Else, set optimizeMessage value to false. For more information on optimized and non-optimized MTOM/XOP messages, refer to http://www.ihe.net/uploadedFiles/Documents/ITI/IHE_ITI_TF_Vol2b.pdf.

4 Implement CDA (Clinical Document Architecture) Persistence Service

By default, the CDA Persistence Web service accepts only https connections. To make it accept client requests over http protocol, a custom JVM argument EnableHTTPForWS=true has to be included to the value of JAVA_OPTIONS variable in `<weblogic_install_dir>/user_projects/domains/<weblogic_domain_name>/bin/startManagedWebLogic.sh`. where, `<weblogic_install_dir>` is the file path where the WebLogic server has been installed. `<weblogic_domain_name>` represents the name of the WebLogic domain.

To configure Transport Layer Security (TLS), refer to the "Configure HDR a Secure Node Actor" section in Implement the Healthcare Enterprise XDS.b Web Service chapter.

A

Appendix A: ETS Supported Terminologies and Cross Maps

ETS Supported Terminologies and Cross Maps

The following table lists terminologies and cross maps supported by ETS, and the information is current as at the publication date of this implementation guide.

Note:

- The Apelon, Inc. website is password protected. Obtain a user ID and password from your Oracle customer representative.
- Apelon, Inc. is an independent company and is not affiliated with Oracle. Contact Apelon directly for further information about available terminologies, mappings and versions.
- Table: ETS Supported Terminologies defines supported terminology or vocabulary standards.
- Table: ETS Supported Cross Maps lists cross maps that have been assured by Apelon, Inc. The mappings are available to HDR customers from Apelon, subject to prior licensing requirements for source and target terminologies.

Table A-1 ETS Supported Terminologies

Coding Scheme Name	OID	Versions	Description	Source	Core?	Seeded?
ANSI ASC X12 Healthcare Provider Taxonomy	2.16.840.1.113883.6.101	2.1, 3.1	HIPAA Provider Taxonomy	Apelon, Inc. from Washington Publishing Company	No12	Yes2
CCI	2003	Canadian Classifications of Health Interventions	Apelon, Inc. from Canadian Institute for Health Information (CIHI)	No12	No	
CDT	v4, v5	Current Dental Terminalogy	Apelon, Inc. from Captiva (from ADA)	No12	No	

Table A-1 (Cont.) ETS Supported Terminologies

CPT4	2.16840.1.113883.6.101	2002, 2003, 2004, 2005	Current Procedural Terminology, Fourth Edition	Apelon, Inc. from Captiva (from AMA)	No12	No
DSM-IV	2.16.840.1.113883.6.126	1994, 2000	Diagnostic and Statistical Manual of Mental Disorders	Apelon, Inc. from American Psychiatric Association	No12	No
ETSClassifications3	2.16.840.1.113894.1004.100.100.2.6	1	Oracle Healthcare Data Repository editable coding scheme for ETS Classifications	Oracle Healthcare	No12	Yes
FDB4	2.16.840.1.113894.1004.100.100.2.1	Monthly, from July, 2004	US FirstDataBank (FDB) US National Drug Data File Plus (NDDF Plus)	Apelon, Inc. from FDB	Yes	No
FDB Diseases	Monthly, from December 2004	US FirstDataBank (FDB) US National Drug Data File Plus (NDDF Plus)	Apelon, Inc. from FDB	No12	No	
HCPCS-2	2.16.840.1.113883.3.41	2002, 2003, 2004, 2005	Healthcare Common Procedure Coding System Level II Alphanumeric Codes (excluding CDT)	Apelon, Inc. from CMS Public Use Files1	No12	No
HL7 v3	1.16	HL7 RIM v3 1.16 coding scheme	HL7	Yes	Yes	
HL7, v35	various	2.01	HL7 RIM v3 coding schemes	HL7	No12	Yes
HDR Supplemental	2.16.840.1.113894.1004.100.100.2.5	various	HDR Supplemental terminology (internal Oracle terminology)	Oracle Healthcare	No12	Yes

Table A-1 (Cont.) ETS Supported Terminologies

ICD-10	2.16.840.1.113883.6.3	1999	International Statistical Classification of Diseases and Related Health Problems, Twentieth Revision	Apelon, Inc. from WHO	No12	No
ICD-9-CM-DRG	2.16.840.1.113883.12.53	2002, 2003, 2004, 2005	US Diagnosis Related Groups	Apelon, Inc. from CMS Public Use Files1	No12	No
ICD-9-CM-MDC	2.16.840.1.113883.12.118	2002, 2003, 2004, 2005	US Major Diagnostic Categories	Apelon, Inc. from CMS Public Use Files1	No12	No
ICD-9-CM-V1	2.16.840.1.113883.6.103	2002, 2003, 2004, 2005	International Classification of Diseases, Ninth Revision, Clinical Modification, volume 1	Apelon, Inc. from Captiva	No12	No
ICD-9-CM-V3	2.16.840.1.113883.6.104	2002, 2003, 2004, 2005	International Classification of Diseases, Ninth Revision, Clinical Modification, volume 3	Apelon, Inc. from Captiva	No12	No
IETF RFC 17666	2.16.840.1.113883.6.84	As of 3/03; as of 11/04	Internet Task Force Request for Comments 1766	Apelon, Inc. from IETF / ISO	No12	Yes 7
ISO 11073-101016	As of 11/03; as of 12/04	International Standards Organization 11073-10101 Point of Care Device Codes	Apelon, Inc. from ISO1	No12	No	
ISO 3166-1 alpha26	2.16.1	As of 2/03; as of 4/04	International Standards Organization 3166-1, 2-character country codes	Apelon, Inc. from ISO1	No12	Yes8

Table A-1 (Cont.) ETS Supported Terminologies

LOINC	2.16.840.1.113883.6.1	2.05, 2.10, 2.12, 2.13, 2.149	Logical Observation Identifiers Names and Codes	Apelon, Inc. from Regenstrief Institute	Yes	No
MedDRA	6.0, 7.1	Medical Dictionary for Regulatory Activities	Apelon, Inc. from Northrup Grumman	No12	No	
NUBC-UB929,10	2.16.840.113883.6.21	As of 5/02	National Uniform Committee Uniform Bill-92	Apelon, Inc. from National Uniform Billing Committee1	No12	Yes
OPCS4	10/02	UK Classification of Operative Procedures (OPCS-4)	Apelon, Inc. from UK National Health Service Information Authority (NHSIA)	No12	No	
RxNorm	2003AC, 2004AB, 2005AA	RxNorm, a standardized nomenclature for clinical drugs	Apelon, Inc. from National Library of Medicine's UMLS	No12	No	
SNOMED CT	2.16.840.1.113883.6.96	1/03,7/03,1/04, 7/04, 1/05	SNOMED Clinical Terms	Apelon, Inc. from College of American Pathologists	Yes	No
TCA.CommunicationType11	2.16.840.1.113894.1004.100.100.2.29	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA COMMUNICATION_TYPE values	Oracle Healthcare	No12	Yes
TCA.ContactPointPurpose11	2.16.840.1.113894.1004.100.100.2.28	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA CONTACT_POINT_PURPOSE values	Oracle Healthcare	No12	Yes

Table A-1 (Cont.) ETS Supported Terminologies

TCA.HZGender11	2.16.840.1.113894.1004.100.100.2.27	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA HZ_GENDER values	Oracle Healthcare	No12	Yes
TCA.MaritalStatus11	2.16.840.1.113894.1004.100.100.2.26	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA MARITAL_STATUS values	Oracle Healthcare	No12	Yes
TCA.PartySiteUseCode11	2.16.840.1.113894.1004.100.100.2.25	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA PARTY_SITE_USE_CODE values	Oracle Healthcare	No12	Yes
TCA.PhoneLineType11	2.16.840.1.113894.1004.100.100.2.24	HZ.M	Oracle Healthcare Data Repository coding scheme for representation of TCA PHONE_LINE_TYPE values	Oracle Healthcare	No12	Yes

Note:

1. Where an OID is available from HL7, that OID is listed. In nine cases (HDR Supplemental, ETS Classification, the TCA terminologies and FDB), an OID has been created under Oracle's root. Where OIDs are missing, petition HL7.
2. v2.1 seeded; v3.1 available from Apelon, Inc.
3. This terminology is for internal use only. Each concept represents an ETS classification. When a new classification is created, ETS adds a new concept to this editable terminology.
4. A subset of FDB's NDDF Plus offering is supported (contains drug identifiers).
5. Implemented as multiple generic coding schemes.
6. Referenced by HL7 as an external vocabulary domain.
7. IETF 1766 references ISO 639-1; a subset of the 3/03 version of ISO 639-1 codes is loaded into ETS. The 4/04 version of ISO 639-1 is available from Apelon, Inc.
8. the 2/03 version is seeded; the 4/04 version is available from Apelon, Inc.
9. Versions 2.12 and later must be obtained from Apelon, Inc. for loading into ETS.
10. Certain portions referenced by HL7.
11. These editable terminologies are for Oracle's Trading Community Architecture (TCA) values and are used in the integration of HDR and TCA.
12. Supported as generic terminology.
13. The following is the format of the LOINC main data file: tab delimited, no header, all fields double quoted. It must have 59 columns with 58 tabs in between. If there is no data, there should be tabs between each column. If you provide more or less number of columns, it will raise an exception. LOINC data is loaded with a SHORT description, which is taken from the SHORTNAME column in the file and the Terminology Preferred description in English, which is actually a concatenation of the 2nd through 7th columns. They are the Component, Property, Time Aspect, System, Scale Type, and Method Type columns. After each column is included there is a colon appended after it. This has indicate other languages. In columns 39, 40, 41, and 42 there are place holders for the French, German, Spanish, and Italian names, respectively. You can load English, Spanish, French, German, and Italian descriptions all in one single load.

    The LOINC data must be truncated at the 300 byte mark when converted to "UTF-8" format (due to DB column size restriction) for the COMMENT field in order for the loader to succeed.

    How Languages are Handled:

LOINC coding scheme by default has descriptions in English, French, German, Spanish and Italian languages. If none of these languages are installed in the Oracle E-Business Suite instance, the load will fail.

As the territory variants of LOINC languages are not provided, the descriptions of these languages will be replicated for all the territory variants installed in E-Business Suite instance. For example, if multiple French language territories have been installed in the E-Business Suite instance, the loader will create descriptions for all of these territories using the same French language LOINC data.

The language mappings will be created by mapping the supported languages to themselves and mapping all other languages to English.

Table A-2 Table: ETS Supported Cross Maps

Mapping	Source
FDB Diseases (monthly versions since 2004) to/from ICD-9-CM (2005)	FDB, as formatted by Apelon, Inc.
HL7 routes and forms1 (released with RIM 2.01) to/ from *FDB routes and forms1* (monthly versions since 2004)	Apelon, Inc.
LOINC to/from CPT-4 (v2.10 - 2005, v2.12 - 2005, v2.13 - 2005))	Apelon, Inc.
SNOMED-CT to/from OPCS4 (v200407 - 2002)	College of American Pathologists, as formatted by Apelon, Inc.
SNOMED-CT to/from CPT-4 (v200301 - 2003, v200407 - 2004, v200407 - 2005, v200501 - 2005)	Apelon, Inc.)
SNOMED-CT to/from ICD-10 (v200407 - 1999, v200501 - 1999)	College of American Pathologists, as formatted by Apelon, Inc.
SNOMED-CT to/from ICD-9-CM (v200301 - 2003, v200307 - 2003, v200407 - 2004, v200407 - 2005, v200501 - 2005)	College of American Pathologists, as formatted by Apelon, Inc.

Note:

1. *HL7 routes and forms* refers to the HL7 codeSystems RouteOfAdministration and OrderableDrugForm, respectively, seeded in HDR as generic *CodingSchemes*. *FDB routes and forms* refers to the concepts within the FDB coding scheme that represent routes and forms.

B

Appendix B: Extensible Concept Lists

Extensible Concept Lists

This Appendix documents seeded concept lists used by the HDR Call Interface, all of which are extensible, and some of which are empty. The following table includes all of the seeded lists, indicating which are shipped empty. You must review and populate the empty lists as appropriate prior to using the associated functionality. Because all of these lists are extensible, you can also add concepts to any list in the table.

See also:

- Adding Concepts to a Concept List
- *HDR Concept Lists Index, Oracle Healthcare Data Repository Javadoc (click HDR Concept Lists link at bottom of Javadoc page),* for a list of all concept lists and their values.

Note:

Lookup Types are synonymous with concept lists.

Table B-1 Seeded Extensible Concept Lists

Concept List Name	Description	Empty List?
CL_ACCESS_APPROACH_SITE_CODE	HDR concept list for values of Access.approachSiteCode	-
CL_ACCESS_TARGET_SITE_CODE	HDR concept list for values of Access.targetSiteCode	-
CL_ACCT_CURRENCY_CODE	HDR concept list for values of Account.currencyCode	-
CL_ACT_CONFIDENTIALITY_CODE	HDR concept list for values of Act.confidentialityCode	-
CL_ACT_LEVEL_CODE	HDR concept list for values of Act.levelCode	Yes
CL_ACT_PRIORITY_CODE	HDR concept list for values of Act.priorityCode	-
CL_ACT_REASON_CODE	HDR concept list for values of Act.reasonCode	-
CL_CASE_DETECTION_METHOD_CODE	HDR concept list for values of PublicHealthCase.detectionMethodCode	Yes
CL_CASE_DISEASE_IMPORTED_CODE	HDR concept list for values of PublicHealthCase.diseaseImportedCode	Yes

Table B-1 (Cont.) Seeded Extensible Concept Lists

CL_CASE_TRANSMISSION_MODE_CODE	HDR concept list for values of PublicHealthCase.transmissionModeCode	Yes
CL_CONT_CAP_TYPE_CODE	HDR concept list for values of Container.capTypeCode	-
CL_CONT_SEPARATOR_TYPE_CODE	HDR concept list for values of Container.separatorTypeCode	-
CL_DEV_ALERT_LEVEL_CODE	HDR concept list for values of Device.alertLevelCode	-
CL_DEV_LOCAL_REMOTE_CONTROL_STATE_CODE	HDR concept list for values of Device.localRemoteControlStateCode	-
CL_DEV_MANUFACTURER_MODEL_NAME	HDR concept list for values of Device.manufacturerModelName	Yes
CL_DEV_SOFTWARE_NAME	HDR concept list for values of Device.softwareName	Yes
CL_DGIMG_SUBJECT_ORIENTATION_CODE	HDR concept list for values of DiagnosticImage.subjectOrientationCode	Yes
CL_DOC_COMPLETION_CODE	HDR concept list for values of Document.completionCode	-
CL_DOC_STORAGE_CODE	HDR concept list for values of Document.storageCode	-
CL_EMP_JOB_CLASS_CODE	HDR concept list for values of Employee.jobClassCode	-
CL_EMP_JOB_CODE	HDR concept list for values of Employee.jobCode	Yes
CL_EMP_JOB_TITLE_NAME	HDR concept list for values of Employee.jobTitleName	Yes
CL_EMP_SALARY_TYPE_CODE	HDR concept list for values of Employee.salaryTypeCode	Yes
CL_ENC_ACUITY_LEVEL_CODE	HDR concept list for values of PatientEncounter.acuityLevelCode	Yes
CL_ENC_ADMISSION_REFERRAL_SOURCE_CODE	HDR concept list for values of PatientEncounter.admissionReferralSourceCode	-
CL_ENC_DISCHARGE_DISPOSITION_CODE	HDR concept list for values of PatientEncounter.dischargeDispositionCode	-
CL_ENC_SPECIAL_ACCOMMODATION_CODE	HDR concept list for values of PatientEncounter.specialAccommodationCode	-
CL_ENC_SPECIAL_COURTESIES_CODE	HDR concept list for values of PatientEncounter.specialCourtesiesCode	-
CL_ENT_HANDLING_CODE	HDR concept list for values of Entity.handlingCode	-
CL_ENT_RISK_CODE	HDR concept list for values of Entity.riskCode	-
CL_FCNTRCT_PAYMENT_TERMS_CODE	HDR concept list for values of FinancialContract.paymentTermsCode	-

Table B-1 (Cont.) Seeded Extensible Concept Lists

CL_INVE_MODIFIER_CODE	HDR concept list for values of InvoiceElement.modifierCode	Yes
CL_LANGCOM_MODE_CODE	HDR concept list for values of LanguageCommunication.modeCode	-
CL_LANGCOM_PROFICIENCY_LEVEL_CODE	HDR concept list for values of LanguageCommunication.proficiencyLevelCode	-
CL_LIST_OWNERSHIP_LEVEL_CODE	HDR concept list for values of WorkingList.ownershipLevelCode	Yes
CL_LIV_ADMINISTRATIVE_GENDER_CODE	HDR concept list for values of LivingSubject.administrativeGenderCode	-
CL_MAT_FORM_CODE	HDR concept list for values of Material.formCode	Yes
CL_NLIV_GENDER_STATUS_CODE	HDR concept list for values of NonPersonLivingSubject.genderStatusCode	-
CL_OBS_INTERPRETATION_CODE	HDR concept list for values of Observation.interpretationCode	-
CL_OBS_METHOD_CODE	HDR concept list for values of Observation.methodCode	-
CL_OBS_TARGET_SITE_CODE	HDR concept list for values of Observation.targetSiteCode	-
CL_ORG_STANDARD_INDUSTRY_CLASS_CODE	HDR concept list for values of Organization.standardIndustryClassCode	Yes
CL_PAT_CONFIDENTIALITY_CODE	HDR concept list for values of Patient.confidentialityCode	-
CL_PAT_VERY_IMPORTANT_PERSON_CODE	HDR concept list for values of Patient.veryImportantPersonCode	-
CL_PROC_APPROACH_SITE_CODE	HDR concept list for values of Procedure.approachSiteCode	-
CL_PROC_METHOD_CODE	HDR concept list for values of Procedure.methodCode	Yes
CL_PROC_TARGET_SITE_CODE	HDR concept list for values of Procedure.targetSiteCode	-
CL_PRTCPN_AWARENESS_CODE	HDR concept list for values of Participation.awarenessCode	-
CL_PRTCPN_FUNCTION_CODE	HDR concept list for values of Participation.functionCode	-
CL_PRTCPN_MODE_CODE	HDR concept list for values of Participation.modeCode	-
CL_PRTCPN_SUBSTITUTION_CONDITION_CODE	HDR concept list for values of Participation.substitutionConditionCode	-
CL_PSN_DISABILITY_CODE	HDR concept list for values of Person.disabilityCode	-

Table B-1 (Cont.) Seeded Extensible Concept Lists

CL_PSN_EDUCATION_LEVEL_CODE	HDR concept list for values of Person.educationLevelCode	-
CL_PSN_ETHNIC_GROUP_CODE	HDR concept list for values of Person.ethnicGroupCode	-
CL_PSN_LIVING_ARRANGEMENT_CODE	HDR concept list for values of Person.livingArrangementCode	-
CL_PSN_MARITAL_STATUS_CODE	HDR concept list for values of Person.maritalStatusCode	-
CL_PSN_RACE_CODE	HDR concept list for values of Person.raceCode	-
CL_PSN_RELIGIOUS_AFFILIATION_CODE	HDR concept list for values of Person.religiousAffiliationCode	-
CL_SBADM_APPROACH_SITE_CODE	HDR concept list for values of SubstanceAdministration.approachSiteCode	-
CL_SBADM_ROUTE_CODE	HDR concept list for values of SubstanceAdministration.routeCode	-
CTB_AU_EVENT_OUTCOME	HDR concept list for auditing event outcomes	-
CTB_ID_TYPES	HDR concept list of identification types	-
CTB_OS_ORG_SYNCHRONIZATION_ROLES	HDR concept list of organization types that are eligible for organization synchronization	-
CTB_SEC_PURP	HDR concept list for security purposes (for backward compatibility)	-

C

Appendix C: Concept List Equivalents

Concept Lists Equivalents

This Appendix contains the following reference table that documents concept lists that are the functional equivalent of TCA attribute validation:

Table C-1 Concept List and FND Lookup Equivalents

TCA Attribute	ETS Concept List	Extensible?
communication_type	CL_URL_SCHEME	No
contact_point_purpose	CL_TEL_USE	No
hz_gender	CL_LIV_ADMINISTRATIVE_GENDER_CODE	Yes
marital_status	CL_PSN_MARITAL_STATUS_CODE	Yes
party_site_use_code	CL_AD_USE	No
phone_line_type	CL_TEL_USE	No

See also:

- Adding Concepts to a Concept List
- *HDR Concept Lists Index, Oracle Healthcare Data Repository Javadoc (click HDR Concept Lists link at bottom of Javadoc page),* for a list of all concept lists and their values.

D

Appendix D: Running HDR Terminology Jobs

- Running HDR Loader Job
- Running HDR Importer Jobs
- Running HDR Maintenance Job
- Running HDR ETS Programs Using Scripts

Note:

Before or after running HDR Terminology Jobs, always call the ETSAdministrationService.invalidateETSCache() API to refresh the cached ETS data. Otherwise, ETS service will contain stale data in cache.

Running HDR Loader Job

You can find the database scripts in $HDR_HOME/ hdr_db/ets/db, if the HDR database is installed using Oracle Universal Installer.

To run a HDR loader job, perform the following:

1. Log in to the database machine as ORACLE_OWNER user and navigate to the $HDR_HOME/hdr_db/ets/db/execute folder.
   Execute the following command:
2. `su $ORACLE_OWNER`

   Use the preceding command if you are not logged in as $ORACLE_OWNER user.

   ```
   cd $HDR_HOME/hdr_db/ets/db/execute
   ```

   Update the run_ets_loader.sql script with required job arguments.

   Substitute **&1** with absolute path of the control file, **&2** with the coding scheme name, and **&3** with the version name.
3. Log in to the database as sysdba user.
   Execute the following command:

   ```
   sqlplus sys@$SID as sysdba
   ```

   Provide the password when prompted.
4. Execute the following script:

   ```
   @ run_ets_loader.sql;
   ```

   Check the log file and execution report for status.

   The log file is generated when the job starts and the execution report is generated when the job ends. These files are located in the $HDR_HOME/ hdr_db/ets/log folder.

Running HDR Importer Jobs

To run a HDR importer job, perform the following:

1. Log in to the database machine as ORACLE_OWNER user and navigate to the $HDR_HOME/ hdr_db/ets/db/execute folder.
2. Execute the following commands:

   ```
   su $ORACLE_OWNER
   ```

   Use the preceding command if you are not logged in as $ORACLE_OWNER user.

   ```
   cd $HDR_HOME/hdr_db/ets/db/execute
   ```

3. Update the run_ets_importer.sql script with required job arguments. Substitute **&1** with the load sequence number, which is available from execution report of the HDR loader job.
4. Log in to the database as sysdba user.
5. Execute the following command:

   ```
   sqlplus sys@$SID as sysdba
   ```

   Provide the password when prompted.
6. Execute the following script:

   ```
   @ run_ets_importer.sql;
   ```

7. Check the log file and execution report for status.

   ```
   The log file is generated when the job starts and the execution report is
   generated when the job ends. These files are located in the $HDR_HOME/
   hdr_db/ets/log folder.
   ```

Running HDR Maintenance Job

To run a HDR maintenance job, perform the following:

1. Log in to the database as ORACLE_OWNER user and navigate to the $HDR_HOME/hdr_db/ets/db/execute folder.
2. Execute the following commands:

   ```
   su $ORACLE_OWNER
   ```

   Use the preceeding command if you are not logged in as $ ORACLE_OWNER user.

   ```
   cd $HDR_HOME/hdr_db/ets/db/execute
   ```

3. Update the run_ets_maintenance.sql script with required job arguments. Substitute **FULL** with the desired mode, if you are not running in Full mode.

   The available modes are FULL, DEFAULT, CLEAN_STAGE, TRUNCATE_STAGE, CLEAN_ACTIVE, and CLASSIFICATIONS.
4. Log in to the database as sysdba user.
5. Execute the following command:

```
sqlplus sys@$SID as sysdba
```

Provide the password when prompted.

6. Execute the following script:

```
@ run_ets_maintenance.sql;
```

7. Check the log file and execution report for status.
   The log file is generated when the job starts and the execution report is generated when the job ends. These files are located in the $HDR_HOME/ hdr_db/ets/log folder.

Running HDR ETS Programs Using Scripts

You can run the ETS loader, importer, and maintenance programs from the middle tier or from the database tier using the following scripts:

Middle Tier

If you want to run the ETS programs from the middle tier, ensure that ETS_HOME and JAVA_HOME are set properly. Before running the following scripts, edit the <HDR_HOME>/ weblogic/hdr/ets/lib/connection.properties file to pass the database host name, port number, database service name, and ETS schema user name.

ETS_HOME = <HDR_HOME>/weblogic/hdr/ets

1. HDR Loader Script

```
#!/bin/sh

        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location
where HDR is installed.
                exit
        fi

        if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
        then
                echo JAVA_HOME is not set or not accessible, set to the location
where JDK is installed.
                exit
        fi

        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_loader_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_loader_program.log
        LOG_FORMAT="%5\$s%n"

        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
           then
               ROLL_OFF_TIMESTAMP=$(date"+%Y.%m.%d-%H.%M.%S")
               echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
               mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi

        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        echo $JAVA_HOME
        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
                echo Configuration files not readable.
                echo $CONNECTION_PROPERTIES
                echo $LOG_CFG
                exit
        fi

        echo HDR ETS Loader Options:
        echo ETS_HOME : $ETS_HOME
        echo Report file: $EXEC_REPORT
        echo Log file: $LOG_FILE
        echo

        #Default ETS process options, can be editable
        #SNOMED-CT 20130901
        #CNTRL_FILE=$ETS_HOME/snomed/SCT_RF2_Snapshot_US1000124_20130901.ctl
        #CODING_SCHEME=SNOMED-CT
        #VERSION_NAME=v20130901
        #LOINC v219
        CNTRL_FILE=$ETS_HOME/loinc/LOINC_for_ETS_v219_control.ctl
        CODING_SCHEME=LOINC
        VERSION_NAME=v219

        echo -en "Use Default ETS Loader Options[Y/N] (default=Y): "
        read etsDefaultOptions
        if [ -z "$etsDefaultOptions" ]
        then
                etsDefaultOptions=Y
        fi

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        if [ "$etsDefaultOptions" != "Y" ]; then
        read -p "Enter Coding Scheme Control File Path: " CNTRL_FILE
        read -p "Enter Coding Scheme Name: " CODING_SCHEME
        read -p "Enter Coding Scheme Version Name: " VERSION_NAME
        fi

        echo "Coding Scheme Control File Path: "$CNTRL_FILE
        echo "Coding Scheme Name: "$CODING_SCHEME
        echo "Coding Scheme Version Name: "$VERSION_NAME

        if [ -z "$CNTRL_FILE" ] || [ ! -f $CNTRL_FILE ];
        then
                echo Coding Scheme Control File Not Readable.
                echo $CNTRL_FILE
                exit
        fi

        if [ -z "$CODING_SCHEME" ] || [ -z "$VERSION_NAME" ];
        then
```

```
                echo Coding Scheme Name and Version Name Must Be Specified.
                exit
        fi
        # Get all of the files in the classpath and convert to the correct UNIX
format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo "CLASSPATH:" $CLASSPATH
        echo
        echo Running ETS Loader Process ...
        echo
        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
        DExecutionReport=$EXEC_REPORT -DLogFile=$LOG_FILE -DDB_USER=ETS -
        DDB_USER_PASSWORD=$DB_PWD -Djava.util.logging.config.file=$LOG_CFG
        oracle.hsgbu.ets.util.LoadManagerLauncher "$CNTRL_FILE" "$CODING_SCHEME"
        "$VERSION_NAME" > /dev/null 2>&1 &

        echo Process ID for ETS Loader: $!

        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

2. HDR Importer Script

```
#!/bin/sh

        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location
where HDR is installed.
                exit
        fi

       if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
       then
                echo JAVA_HOME is not set or not accessible, set to the location
where JDK is installed.
                exit
        fi

        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_importer_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_importer_program.log
        LREPORT_FILE=$ETS_HOME/log/hdr_ets_loader_program.rpt
        LOG_FORMAT="%5\$s%n"
        ACTIVATE_VERSION="N"
        DEFAULT_VERSION="N"
        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
           then
                ROLL_OFF_TIMESTAMP=$(date "+%Y.%m.%d-%H.%M.%S")
                echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
                mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi

        if [ -f $LREPORT_FILE ]
        then
                SEQNO=`grep "Load sequence number" $LREPORT_FILE | cut -d
':' -f2-2 | sed -e 's/^[ ]*//g' | sed
        's/[ ]*$//g'`
                echo Load sequence no from loader report file: $SEQNO
        fi

        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
                echo Configuration files not readable.
                echo $CONNECTION_PROPERTIES
                echo $LOG_CFG
                exit
        fi

        echo HDR ETS Importer Options:
        echo ETS_HOME : $ETS_HOME
        echo ETS Loader Report file: $LREPORT_FILE
        echo ETS Importer Report file: $EXEC_REPORT
        echo ETS Importer Log file: $LOG_FILE
        echo

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        read -p "Enter ETS Loader Sequence Number (default=$SEQNO): "SEQ_N

        if [ ! -z "$SEQ_N" ]
        then
                SEQNO=$SEQ_N
        fi

        if [ -z "$SEQNO" ]
        then
                echo Load sequence no is needed to run the importer program.
The load sequence no is \"$SEQNO\"
                exit
        fi

        read -p "Enter Y/N to specify if the newly imported version should
be set to 'active' after import
        (default=$ACTIVATE_VERSION): " ACTIVATE_VERSION_P
        if [ ! -z "$ACTIVATE_VERSION_P" ]then
              ACTIVATE_VERSION=$ACTIVATE_VERSION_P
        fi
        read -p "Enter Y/N to specify if the newly imported version should be set
to 'default' after import
       (default=$DEFAULT_VERSION): " DEFAULT_VERSION_P
        if [ ! -z "$DEFAULT_VERSION_P" ]
        then
              DEFAULT_VERSION=$DEFAULT_VERSION_P
        fi

        # Get all of the files in the classpath and convert to the correct UNIX
format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo
        echo Running ETS Importer Process ...
        echo

        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
DExecutionReport=$EXEC_REPORT -
        DLogFile=$LOG_FILE -DDB_USER=ETS -DDB_USER_PASSWORD=$DB_PWD -
        Djava.util.logging.config.file=$LOG_CFG
oracle.hsgbu.ets.util.ImportManagerLauncher $SEQNO
        $ACTIVATE_VERSION $DEFAULT_VERSION > /dev/null 2>&1 &
        echo Process ID for ETS Importer: $!
        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

3. HDR Maintenance Script

> **Note:**
>
> ETS Maintenance Job prompts for the run mode. For run mode details, see Scheduling the Maintenance Job in the *Oracle Healthcare Data Repository Implementation Guide*.

```
#!/bin/sh
        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location
where HDR is installed.
                exit
        fi

        if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
        then
                echo JAVA_HOME is not set or not accessible, set to the location
where JDK is installed.
                exit
        fi

        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_maintenance_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_maintenance_program.log
        LOG_FORMAT="%5\$s%n"
        RUNMODE=FULL

        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
          then
                ROLL_OFF_TIMESTAMP=$(date "+%Y.%m.%d-%H.%M.%S")
                echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
                mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi

        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
               echo Configuration files not readable.
               echo $CONNECTION_PROPERTIES
               echo $LOG_CFG
               exit
        fi

        echo HDR ETS Maintenance Options:
        echo ETS_HOME : $ETS_HOME
        echo Report file: $EXEC_REPORT
        echo Log file: $LOG_FILE
        echo

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        read -p "Enter ETS Maintenance Run Mode (default mode: FULL): "
RUN_MOD

        if [ ! -z "$RUN_MODE"]
        then
              RUNMODE=$RUN_MODE
        fi

        # Get all of the files in the classpath and convert to the correct
UNIX format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo
        echo Running ETS Maintenance Process
        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
DExecutionReport=$EXEC_REPORT -
        DLogFile=$LOG_FILE -DDB_USER=ETS -DDB_USER_PASSWORD=$DB_PWD -
        Djava.util.logging.config.file=$LOG_CFG
oracle.hsgbu.ets.util.MaintenanceManagerLauncher
        $RUNMODE > /dev/null 2>&1 &
        echo Process ID for ETS Maintenance: $!
        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

Database Tier

If you want to run the HDR ETS programs from the database tier, ensure that ETS_HOME and JAVA_HOME are set properly. Before running the following scripts, edit the <HDR_HOME>/ hdr_db/ets/lib/connection.properties file to pass the database host name, port number, database service name, and ETS schema user name.

ETS_HOME = <HDR_HOME>/hdr_db/ets

1. HDR Loader Script

```
#!/bin/sh

        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location
where HDR is installed.
                exit
        fi

        if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
        then
                echo JAVA_HOME is not set or not accessible, set to the location
where JDK is installed.
                exit
        fi

        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_loader_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_loader_program.log
        LOG_FORMAT="%5\$s%n"

        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
           then
               ROLL_OFF_TIMESTAMP=$(date"+%Y.%m.%d-%H.%M.%S")
               echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
               mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi
        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        echo $JAVA_HOME
        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
                echo Configuration files not readable.
                echo $CONNECTION_PROPERTIES
                echo $LOG_CFG
                exit
        fi

        echo HDR ETS Loader Options:
        echo ETS_HOME : $ETS_HOME
        echo Report file: $EXEC_REPORT
        echo Log file: $LOG_FILE
        echo

        #Default ETS process options, can be editable
        #SNOMED-CT 20130901
        #CNTRL_FILE=$ETS_HOME/snomed/SCT_RF2_Snapshot_US1000124_20130901.ctl
        #CODING_SCHEME=SNOMED-CT
        #VERSION_NAME=v20130901
        #LOINC v219
        CNTRL_FILE=$ETS_HOME/loinc/LOINC_for_ETS_v219_control.ctl
        CODING_SCHEME=LOINC
        VERSION_NAME=v219

        echo -en "Use Default ETS Loader Options[Y/N] (default=Y): "
        read etsDefaultOptions
        if [ -z "$etsDefaultOptions" ]
        then
                etsDefaultOptions=Y
        fi

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        if [ "$etsDefaultOptions" != "Y" ]; then
        read -p "Enter Coding Scheme Control File Path: " CNTRL_FILE
        read -p "Enter Coding Scheme Name: " CODING_SCHEME
        read -p "Enter Coding Scheme Version Name: " VERSION_NAME
        fi

        echo "Coding Scheme Control File Path: "$CNTRL_FILE
        echo "Coding Scheme Name: "$CODING_SCHEME
        echo "Coding Scheme Version Name: "$VERSION_NAME

        if [ -z "$CNTRL_FILE" ] || [ ! -f $CNTRL_FILE ];
        then
                echo Coding Scheme Control File Not Readable.
                echo $CNTRL_FILE
                exit
        fi

        if [ -z "$CODING_SCHEME" ] || [ -z "$VERSION_NAME" ];
        then
                echo Coding Scheme Name and Version Name Must Be Specified.
                exit
        fi
        # Get all of the files in the classpath and convert to the correct UNIX 
format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo "CLASSPATH:" $CLASSPATH
        echo
        echo Running ETS Loader Process ...
        echo
        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
        DExecutionReport=$EXEC_REPORT -DLogFile=$LOG_FILE -DDB_USER=ETS -
        DDB_USER_PASSWORD=$DB_PWD -Djava.util.logging.config.file=$LOG_CFG 
        oracle.hsgbu.ets.util.LoadManagerLauncher "$CNTRL_FILE" "$CODING_SCHEME" 
        "$VERSION_NAME" > /dev/null 2>&1 &

        echo Process ID for ETS Loader: $!

        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

2. HDR Importer Script

```
#!/bin/sh

        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location 
where HDR is installed.
                exit
        fi

       if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
       then
                echo JAVA_HOME is not set or not accessible, set to the location 
where JDK is installed.
                exit
        fi

        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_importer_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_importer_program.log
        LREPORT_FILE=$ETS_HOME/log/hdr_ets_loader_program.rpt
        LOG_FORMAT="%5\$s%n"
        ACTIVATE_VERSION="N"
        DEFAULT_VERSION="N"
        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
           then
                ROLL_OFF_TIMESTAMP=$(date "+%Y.%m.%d-%H.%M.%S")
                echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
                mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi

        if [ -f $LREPORT_FILE ]
        then
                SEQNO=`grep "Load sequence number" $LREPORT_FILE | cut -d
':' -f2-2 | sed -e 's/^[ ]*//g' | sed
        's/[ ]*$//g'`
                echo Load sequence no from loader report file: $SEQNO
        fi

        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
                echo Configuration files not readable.
                echo $CONNECTION_PROPERTIES
                echo $LOG_CFG
                exit
        fi

        echo HDR ETS Importer Options:
        echo ETS_HOME : $ETS_HOME
        echo ETS Loader Report file: $LREPORT_FILE
        echo ETS Importer Report file: $EXEC_REPORT
        echo ETS Importer Log file: $LOG_FILE
        echo

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        read -p "Enter ETS Loader Sequence Number (default=$SEQNO): "SEQ_N

        if [ ! -z "$SEQ_N" ]
        then
                SEQNO=$SEQ_N
        fi

        if [ -z "$SEQNO" ]
        then
                echo Load sequence no is needed to run the importer program.
The load sequence no is \"$SEQNO\"
                exit
        fi

        read -p "Enter Y/N to specify if the newly imported version should
be set to 'active' after import
        (default=$ACTIVATE_VERSION): " ACTIVATE_VERSION_P
        if [ ! -z "$ACTIVATE_VERSION_P" ]then
              ACTIVATE_VERSION=$ACTIVATE_VERSION_P
        fi
        read -p "Enter Y/N to specify if the newly imported version should be set
to 'default' after import
       (default=$DEFAULT_VERSION): " DEFAULT_VERSION_P
        if [ ! -z "$DEFAULT_VERSION_P" ]
        then
              DEFAULT_VERSION=$DEFAULT_VERSION_P
        fi

        # Get all of the files in the classpath and convert to the correct UNIX
format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo
        echo Running ETS Importer Process ...
        echo

        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
DExecutionReport=$EXEC_REPORT -
        DLogFile=$LOG_FILE -DDB_USER=ETS -DDB_USER_PASSWORD=$DB_PWD -
        Djava.util.logging.config.file=$LOG_CFG
oracle.hsgbu.ets.util.ImportManagerLauncher $SEQNO
        $ACTIVATE_VERSION $DEFAULT_VERSION > /dev/null 2>&1 &
        echo Process ID for ETS Importer: $!
        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

3. HDR Maintenance Script

> **Note:**
>
> ETS Maintenance Job prompts for the run mode. For run mode details, see Scheduling the Maintenance Job in the *Oracle Healthcare Data Repository Implementation Guide*.

```
#!/bin/sh
        if [ -z "$ETS_HOME" ] || [ ! -d "$ETS_HOME" ];
        then
                echo ETS_HOME is not set or not accessible, set to the location
where HDR is installed.
                exit
        fi

        if [ -z "$JAVA_HOME" ] || [ ! -d "$JAVA_HOME" ];
        then
                echo JAVA_HOME is not set or not accessible, set to the location
where JDK is installed.
                exit
        fi
        #Default ETS process options, can be editable
        JAVA=$JAVA_HOME/bin/java
        ETS_LIB="$ETS_HOME/lib/*"
        CONNECTION_PROPERTIES=$ETS_HOME/lib/connection.properties
        LOG_CFG=$ETS_HOME/lib/logging.properties
        EXEC_REPORT=$ETS_HOME/log/hdr_ets_maintenance_program.rpt
        LOG_FILE=$ETS_HOME/log/hdr_ets_maintenance_program.log
        LOG_FORMAT="%5\$s%n"
        RUNMODE=FULL

        # ER 29190730 - Roll off previous run's log file (if it exists)
        if [ -f $LOG_FILE ]
          then
                  ROLL_OFF_TIMESTAMP=$(date "+%Y.%m.%d-%H.%M.%S")
                  echo rolling off existing log file $LOG_FILE with
date $ROLL_OFF_TIMESTAMP
                  mv $LOG_FILE $LOG_FILE.$ROLL_OFF_TIMESTAMP
        fi

        if [ ! -x $JAVA ]
        then
                echo $JAVA is not executable
                exit
        fi

        if [ ! -f $CONNECTION_PROPERTIES ] || [ ! -f $LOG_CFG ];
        then
               echo Configuration files not readable.
               echo $CONNECTION_PROPERTIES
               echo $LOG_CFG
               exit
        fi

        echo HDR ETS Maintenance Options:
        echo ETS_HOME : $ETS_HOME
        echo Report file: $EXEC_REPORT
        echo Log file: $LOG_FILE
        echo

        read -s -p "Enter ETS Database User Password (not echoed to screen):
" DB_PWD && echo
        read -p "Enter ETS Maintenance Run Mode (default mode: FULL): "
RUN_MOD

        if [ ! -z "$RUN_MODE"]
        then
              RUNMODE=$RUN_MODE
        fi

        # Get all of the files in the classpath and convert to the correct
UNIX format
        if [[ "$(uname)" == "CYGWIN"* ]];
        then
        CLASSPATH=$ETS_LIB
        else
        CLASSPATH=""
        for j in $ETS_LIB
        do
                CLASSPATH=$j:$CLASSPATH
        done
        fi

        echo
        echo Running ETS Maintenance Process
        $JAVA -cp $CLASSPATH -
Djava.util.logging.SimpleFormatter.format="$LOG_FORMAT" -
        DCONNECTION_PROPERTIES=$CONNECTION_PROPERTIES -
DExecutionReport=$EXEC_REPORT -
        DLogFile=$LOG_FILE -DDB_USER=ETS -DDB_USER_PASSWORD=$DB_PWD -
        Djava.util.logging.config.file=$LOG_CFG
oracle.hsgbu.ets.util.MaintenanceManagerLauncher
        $RUNMODE > /dev/null 2>&1 &
        echo Process ID for ETS Maintenance: $!
        tail --retry -q -n 0 -f $LOG_FILE --pid=$!
```

E

Appendix E: Act Configuration Artifacts

- Act Configuration Artifacts
- Act Configuration Artifact Tables
- Act Configuration Artifact Tables for HDR Version 5

Act Configuration Artifacts

Because some RMIMs and CMETs tolerate different conformance profiles for the same object type, and because there is no information in the RMIM that determines for OMP which conformance to use , OMP uses the information in the Act Concept configuration table to define the appropriate conformance. The tables listed in the following sections define all occurrences in OMP-supported models that either have seed data or require additional client configuration, for both RMIMs and CMETs.

Act Configuration Artifact Tables

This section includes the following tables:

- Act Configuration Artifacts: RMIMs, Partially Seeded
- Act Configuration Artifacts: RMIMs, Client Configured
- Act Configuration Artifacts: CMETs, Seeded
- Act Configuration Artifacts: CMETs, Partially Seeded
- Act Configuration Artifacts: CMETs, Client Configured

Table E-1 Act Configuration Artifacts: RMIMs, Partially Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
Condition Problem	POPR_RM930000HT01	RSON	POPR_RM930000HT01.ConditionProblem POPR_MT930000HT01.Reason	ObservationEventGeneral
Condition Problem	POPR_RM930000HT01	RSON	POPR_RM930000HT01.ConditionProblem POPR_MT930000HT01.Reason	ObservationEventDx
Diagnostic Report Observation Event	POXX_RM112000HT01	PERT	POXX_RM112000HT01.DiagnosticReportObservationEvent POXX_MT112000HT01.PertinentInformation2	ObservationEventAnnotation
Diet Request	PODI_RM9410000HT01	RSON	PODI_MT941000HT01.DietRequest PODI_MT941000HT01.Reason	ObservationEventGeneral

Table E-1 (Cont.) Act Configuration Artifacts: RMIMs, Partially Seeded

Diet Request	PODI_RM941000HT01	RSON	PODI_MT941000HT01.DietRequest PODI_MT941000HT01.Reason	ObservationEventDx
EncounterAppointment	PRPA_RM410001HT01	PERT	PRPA_MT410001HT01.EncounterAppointment	ObservationEventSupporting
EncounterAppointment	PRPA_RM410001HT01	PERT	PRPA_MT410001HT01.EncounterAppointment	ObservationEventAnnotation
EncounterAppointment	PRPA_RM410001HT01	RSON	PRPA_MT410001HT01.EncounterAppointment	ObservationEventGeneral
EncounterAppointment	PRPA_RM410001HT01	RSON	PRPA_MT410001HT01.EncounterAppointment	ObservationEventDx
EncounterEvent	PRPA_RM400001HT02	PERT	PRPA_MT400001HT02.EncounterEvent PRPA_MT400001HT02.PertinentInformation3	ObservationEventSupporting
EncounterEvent	PRPA_RM400001HT02	RSON	PRPA_MT400001HT02.EncounterEvent	ObservationEventGeneral
EncounterEvent	PRPA_RM400001HT02	RSON	PRPA_MT400001HT02.EncounterEvent PRPA_MT400001HT02.PertinentInformation4	ObservationEventDx
Intolerance Observation Event	PRPA_RM420001HT01	PERT	PRPA_RM420001HT01.ObservationEventIntolerance	ObservationEvent
Intolerance Observation Event	PRPA_RM420001HT01	PERT	PRPA_RM420001HT01.ObservationEventIntolerance	ObservationEventSupporting
Intolerance Observation Event	PRPA_RM420001HT01	PERT	PRPA_RM420001HT01.ObservationEventIntolerance	ObservationEventAnnotation
Observation Event	POXX_MT110000HT02	PERT	POXX_MT110000HT02.ObservationEvent	ObservationEventSupporting
Observation Event	POXX_MT110000HT02	PERT	POXX_MT110000HT02.ObservationEvent	ObservationEventAnnotation
Observation Order	POXX_RM120000HT02	RSON	POXX_MT120000HT02.ObservationOrder	ObservationEventGeneral
Observation Order	POXX_RM120000HT02	RSON	POXX_MT120000HT02.ObservationOrder	ObservationEventDx
Patient Supply Request	POSP_RM940000HT01	RSON	POSP_MT940000HT01.SupplyRequest	ObservationEventGeneral
Patient Supply Request	POSP_RM940000HT01	RSON	POSP_MT940000HT01.SupplyRequest	ObservationEventDx
Procedure Order	POXX_RM130000HT02	RSON	POXX_MT130000HT02.ProcedureOrder	ObservationEventGeneral
Procedure Order	POXX_RM130000HT02	RSON	POXX_MT130000HT02.ProcedureOrder	ObservationEventDx

Table E-1 (Cont.) Act Configuration Artifacts: RMIMs, Partially Seeded

Specimen Observation Event	POXX_RM111000HT01	PERT	POXX_MT111000HT01.SpecimenObservationEvent	ObservationEventAnnotation
Specimen Observation Event	POXX_RM111000HT01	PERT	POXX_MT111000HT01.SpecimenObservationEvent	ObservationEventGeneral
Substance Administration Order	POSA_RM920000HT01	PERT	POSA_MT920000HT01.SubstanceAdministrationOrder	ObservationEventAnnotation
Substance Administration Order	POSA_RM920000HT01	RSON	POSA_MT920000HT01.SubstanceAdministrationOrder	ObservationEventGeneral
Substance Administration Order	POSA_RM920000HT01	RSON	POSA_MT920000HT01.SubstanceAdministrationOrder	ObservationEventDx

Table E-2 Act Configuration Artifacts: CMETs, Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
E_Person	COCT_RM030200HT02	IDENT	COCT_MT030200HT02.Person	playedPersonDomain

Table E-3 Act Configuration Artifacts: CMETs, Partially Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
A_DiagnosticReportObservationEvent	COCT_RM122500HT01	PERT	COCT_MT122500HT01.DiagnosticReportObservationEvent	ObservationEventAnnotation
A_DietRequest	COCT_RM941010HT01	RSON	COCT_MT941010HT01.DietRequest	ObservationEventGeneral
A_DietRequest	COCT_RM941010HT01	RSON	COCT_MT941010HT01.DietRequest	ObservationEventDx
A_ObservationGeneral	COCT_RM120500HT03	PERT	COCT_MT120500HT03.ObservationEventGeneral	ObservationEventAnnotation
A_ObservationOrder	COCT_RM970000HT03	RSON	COCT_MT970000HT03.ObservationOrder	ObservationEventGeneral
A_ObservationOrder	COCT_RM970000HT03	RSON	COCT_MT970000HT03.ObservationOrder	ObservationEventDx
A_PatientSupplyRequest	COCT_RM940000HT01	RSON	COCT_MT940000HT01.SupplyRequest	ObservationEventGeneral
A_PatientSupplyRequest	COCT_RM940000HT01	RSON	COCT_MT940000HT01.SupplyRequest	ObservationEventDx
A_ProcedureOrder	COCT_RM950000HT02	RSON	COCT_MT950000HT02.ProcedureOrder	ObservationEventGeneral

Table E-3 (Cont.) Act Configuration Artifacts: CMETs, Partially Seeded

A_Procedure Order	COCT_RM950000 HT02	RSON	COCT_MT950000HT02. ProcedureOrder	ObservationEventDx
A_Specimen Observation Event	COCT_RM121500 HT02	PERT	COCT_MT121500HT02. SpecimenObservationEv ent	ObservationEventAnnot ation

Table E-4 Act Configuration Artifacts: CMETs, Client Configured

Artifact Name	ID	AR/Part/ Role	Complex Type	Element Name
A_Diagnostic ReportObserv ationEvent	COCT_RM12250 0HT01	PERT	COCT_MT122500HT0 1.DiagnosticReportObs ervationEvent	ObservationEventSup porting
A_Observatio nGeneral	COCT_RM12050 0HT03	PERT	COCT_MT120500HT0 3.ObservationEventGe neral	ObservationEventSup porting
A_Specimen ObservationE vent	COCT_RM12150 0HT02	PERT	COCT_MT121500HT0 2.SpecimenObservatio nEvent	ObservationEventSup porting
E_Person	COCT_RM03020 0HT02	IDENT	COCT_MT030200HT0 2.Person	playedIdentifications

Act Configuration Artifact Tables for HDR Version 5

This section includes the following tables, for use by HDR Version 5 users, or for users migrating from Version 4 to Version 5:

- Act Configuration Artifacts: RMIMs, Seeded
- Act Configuration Artifacts: RMIMs, Partially Seeded
- Act Configuration Artifacts: RMIMs, Client Configured
- Act Configuration Artifacts: CMETs, Seeded
- Act Configuration Artifacts: CMETs, Partially Seeded
- Act Configuration Artifacts: CMETs, Client Configured

Table E-5 Act Configuration Artifacts: RMIMs, Seeded

Artifact Name	ID	AR/Part/ Role	Complex Type	Element Name
CT Lab Observation Periodic Report	PORT_RM0 30001HL01	PERT	PORT_MT030001HL01.Pertine ntInformation2	pertinentToxicityGr ade
CT Lab Observation Periodic Report	PORT_RM0 30001HL01	PERT	PORT_MT030001HL01.Pertine ntInformation1	pertinentTestCom ment

Table E-5 (Cont.) Act Configuration Artifacts: RMIMs, Seeded

CT Lab Observation Periodic Report	PORT_RM030001HL01	PERT	PORT_MT030001HL01.PertinentInformation	pertinentAgeAtVisit
CT Lab Observation Periodic Report	PORT_RM030001HL01	PERT	PORT_MT030001HL01.PertinentInformation3	pertinentFastingStatus
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation5	pertinentPriorInsurancePlanID
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation4	pertinentCoverageType
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation28	pertinentReleaseInformationCode
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation24	pertinentNoticeOfAdmissionFlag
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation26	pertinentNoticeOfAdmissionDate
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation22	pertinentAssignmentOfBenefits
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation20	pertinentCoordinationOfBenefits
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation18	pertinentReportOfEligibilityFlag
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation15	pertinentReportOfEligibilityDate
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation12	pertinentBillingStatus
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation7	pertinentDelayBeforeLRDay
IndividualCaseSafetyReport (ICSR)	PORR_RM040000HT01	PERT	PORR_MT040000HT01.PertinentInformation1	pertinentObservationEvent
IndividualCaseSafetyReport (ICSR)	PORR_RM040000HT01	PERT	PORR_MT040000HT01.PertinentInformation5	ObservationEventIntolerance
NotifiableCondition	PORR_RM100001HT01	COMP	PORR_MT100001HT01.Component	ObservationEventIntolerance
NotifiableCondition	PORR_RM100001HT01	PERT	PORR_MT100001HT01.PertinentInformation3	ObservationEventIntolerance

Table E-6 Act Configuration Artifacts: RMIMs, Partially Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
Admit Order	PRPA_RM420001HT01	PERT	PRPA_MT420001HT01.PertinentInformation1	ObservationEventSupporting

Table E-6 (Cont.) Act Configuration Artifacts: RMIMs, Partially Seeded

Admit Order	PRPA_RM420001HT01	PERT	PRPA_MT420001HT01.PertinentInformation3	ObservationEventAnnotation
Admit Order	PRPA_RM420001HT01	RSON	PRPA_MT420001HT01.Reason	ObservationEventGeneral
Admit Order	PRPA_RM420001HT01	RSON	PRPA_MT420001HT01.Reason	ObservationEventDx
Admit Order	PRPA_RM420001HT01	RSON	PRPA_MT420001HT01.Reason2	ObservationEventGeneral
Admit Order	PRPA_RM420001HT01	RSON	PRPA_MT420001HT01.Reason2	ObservationEventDx
Diet Request	PODI_RM9410000HT02	RSON	PODI_MT941000HT02.Reason	ObservationEventGeneral
Diet Request	PODI_RM9410000HT02	RSON	PODI_MT941000HT02.Reason	ObservationEventDx
Discharge Order	PRPA_RM430001HT01	RSON	PRPA_MT430001HT01.Reason	ObservationEventGeneral
Discharge Order	PRPA_RM430001HT01	RSON	PRPA_MT430001HT01.Reason	ObservationEventDx
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation3	ObservationEventSupporting
EncounterEvent	PRPA_RM400001HT03	PERT	PRPA_MT400001HT03.PertinentInformation1	ObservationEventAnnotation
EncounterEvent	PRPA_RM400001HT03	RSON	PRPA_MT400001HT03.Reason	ObservationEventGeneral
EncounterEvent	PRPA_RM400001HT03	RSON	PRPA_MT400001HT03.Reason	ObservationEventDx
IndividualCaseSafetyReport (ICSR)	PORR_RM040000HT01	PERT	PORR_MT040000HT01.PertinentInformation5	ObservationEventGeneral
NotifiableCondition	PORR_RM100001HT01	COMP	PORR_MT100001HT01.Component	ObservationEventGeneral
NotifiableCondition	PORR_RM100001HT01	COMP	PORR_MT100001HT01.Component	ObservationEventDx
NotifiableCondition	PORR_RM100001HT01	PERT	PORR_MT100001HT01.PertinentInformation3	ObservationEventDx
Observation Order	POXX_RM120000HT03	RSON	POXX_MT120000HT03.Reason	ObservationEventGeneral
Observation Order	POXX_RM120000HT03	RSON	POXX_MT120000HT03.Reason	ObservationEventDx
Patient Care Referral Order	REPC_RM002000HT01	RSON	REPC_MT0020000HT01.Reason	ObservationEventGeneral
Patient Care Referral Order	REPC_RM002000HT01	RSON	REPC_MT0020000HT01.Reason	ObservationEventDx

Table E-6 (Cont.) Act Configuration Artifacts: RMIMs, Partially Seeded

Patient Supply Request	POSP_RM940000HT02	RSON	POSP_MT940000HT02.Reason	ObservationEventGeneral
Patient Supply Request	POSP_RM940000HT02	RSON	POSP_MT940000HT02.Reason	ObservationEventDx
Procedure Order	POXX_RM130000HT03	RSON	POXX_MT130000HT03.Reason	ObservationEventGeneral
Procedure Order	POXX_RM130000HT03	RSON	POXX_MT130000HT03.Reason	ObservationEventDx
Specimen Observation Order	POXX_RM121000HT02	RSON	POXX_MT12100HT02.Reason	ObservationEventDx
Specimen Observation Order	POXX_RM121000HT02	RSON	POXX_MT12100HT02.Reason	ObservationEventGeneral
Substance Administration Order	POSA_RM920000HT02	PERT	POSA_MT920000HT02.PertinentInformation1	ObservationEventAnnotation
Substance Administration Order	POSA_RM920000HT02	RSON	POSA_MT920000HT02.Reason	ObservationEventGeneral
Substance Administration Order	POSA_RM920000HT02	RSON	POSA_MT920000HT02.Reason	ObservationEventDx
Transfer Order	PRPA_RM302001HT01	RSON	PRPA_MT302001HT01.Reason	ObservationEventGeneral
Transfer Order	PRPA_RM302001HT01	RSON	PRPA_MT302001HT01.Reason	ObservationEventDx

Table E-7 Act Configuration Artifacts: RMIMs, Client Configured

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
CT Lab Observation Periodic Report	PORT_RM030001HL01	RCT	PORT_MT030001HL01.RecordTarget	EnrolledSubject
CT Lab Observation Periodic Report	PORT_RM030001HL01	RCT	PORT_MT030001HL01.RecordTarget	ScreeningSubject
CT Lab Observation Periodic Report	PORT_RM030001HL01	RESBJ	PORT_MT030001HL01.Person	playedScreeningSubject2

Table E-7 (Cont.) Act Configuration Artifacts: RMIMs, Client Configured

CT Lab Observation Periodic Report	PORT_RM030001HL01	RESBJ	PORT_MT030001HL01.Person	playedSpareSubject
IndividualCaseSafetyReport (ICSR)	PORR_RM040000HT01	PERT	PORR_MT040000HT01.PertinentInformation5	DiagnosticReportObservationEvent
NotifiableCondition	PORR_RM100001HT01	COMP	PORR_MT100001HT01.Component	DiagnosticReportObservationEvent
NotifiableCondition	PORR_RM100001HT01	PERT	PORR_MT100001HT01.PertinentInformation3	ObservationEventGeneral
NotifiableCondition	PORR_RM100001HT01	PERT	PORR_MT100001HT01.PertinentInformation3	DiagnosticReportObservationEvent
PostFinancial Transaction	FIAB_RM021000HT01	PERT	FIAB_MT020000HT01.PertinentInformation	ObservationEventGeneral
PostFinancial Transaction	FIAB_RM021000HT01	PERT	FIAB_MT020000HT01.PertinentInformation	DiagnosticReportObservationEvent
Substance Administration Order	POSA_RM920000HT02	PERT	POSA_MT920000HT02.PertinentInformation2	ObservationEventGeneral

Table E-8 Act Configuration Artifacts: CMETs, Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
A_ConditionProblem	COCT_RM030200HT03	IDENT	COCT_MT030200HT03.Person	playedPersonDomain
A_DiagnosticReportObservationEvent	COCT_RM120300HT01	PERT	COCT_MT120300HT01.PertinentInformation1	pertinentObservationEvent
A_Substance Administration Order	COCT_RM960000HT03	PERT	COCT_MT080000HT03.PertinentInformation1	pertinentObservationEvent
R_Specimen	COCT_RM080000HT03	PERT	COCT_MT080000HT03.PertinentInformation1	pertinentObservationEvent

Table E-9 Act Configuration Artifacts: CMETs, Partially Seeded

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
A_DietRequest	COCT_RM120300HT01	PERT	COCT_MT120300HT01.PertinentInformation3	ObservationEventAnnotation
A_DietRequest	COCT_RM120300HT01	PERT	COCT_MT120500HT03.PertinentInformation2	ObservationEventAnnotation
A_DietRequest	COCT_RM120500HT03	PERT	COCT_MT120500HT03.PertinentInformation1	ObservationEvent

Table E-9 (Cont.) Act Configuration Artifacts: CMETs, Partially Seeded

A_ObservationGeneral	COCT_RM120600HT01	PERT	COCT_MT120600HT01.PertinentInformation1	ObservationEvent
A_ObservationGeneral	COCT_RM120600HT01	PERT	COCT_MT120600HT01.PertinentInformation2	ObservationEventGeneral
A_ObservationIntolerance	COCT_RM121500HT02	PERT	COCT_MT121500HT02.PertinentInformation2	ObservationEventAnnotation
A_ObservationIntolerance	COCT_RM122500HT02	PERT	COCT_MT122500HT02.PertinentInformation2	ObservationEventAnnotation
A_ObservationOrder	COCT_RM920000HT01	RSON	COCT_MT920000HT01.Reason	ObservationEventGeneral
A_ObservationOrder	COCT_RM920000HT01	RSON	COCT_MT920000HT01.Reason	ObservationEventDx
A_PatientSupplyRequest	COCT_RM920000HT01	PERT	COCT_MT920000HT01.PertinentInformation1	ObservationEventAnnotation
A_ProcedureEvent	COCT_RM920100HT01	PERT	COCT_MT920100HT01.PertinentInformation2	ObservationEvent
A_ProcedureEvent	COCT_RM930000HT01	RSON	COCT_MT930000HT01.Reason	ObservationEventGeneral
A_ProcedureEvent	COCT_RM930000HT01	RSON	COCT_MT930000HT01.Reason	ObservationEventDx
A_ProcedureOrder	COCT_RM940000HT01	RSON	COCT_MT940000HT01.Reason	ObservationEventGeneral
A_ProcedureOrder	COCT_RM940000HT01	RSON	COCT_MT940000HT01.Reason	ObservationEventDx
A_SpecimenObservationEvent	COCT_RM941010HT01	RSON	COCT_MT941010HT01.Reason	ObservationEventGeneral
A_SpecimenObservationEvent	COCT_RM941010HT01	RSON	COCT_MT941010HT01.Reason	ObservationEventDx
A_SpecimenObservationOrder	COCT_RM950000HT03	RSON	COCT_MT950000HT03.Reason	ObservationEventGeneral
A_SpecimenObservationOrder	COCT_RM950000HT03	RSON	COCT_MT950000HT03.Reason	ObservationEventDx
A_SubstanceAdministrationEvent	COCT_RM951000HT01	RSON	COCT_MT951000HT01.Reason	ObservationEventGeneral
A_SubstanceAdministrationEvent	COCT_RM951000HT01	RSON	COCT_MT951000HT01.Reason	ObservationEventDx
A_SubstanceAdministrationOrder	COCT_RM951000HT01	PERT	COCT_MT951000HT01.PertinentInformation2	ObservationEvent

Table E-9 (Cont.) Act Configuration Artifacts: CMETs, Partially Seeded

A_SubstanceAdministrationOrder	COCT_RM960000HT03	PERT	COCT_MT080000HT03.PertinentInformation2	ObservationEventAnnotation
E_Person	COCT_RM970000HT03	RSON	COCT_MT970000HT03.Reason	ObservationEventGeneral
E_Person	COCT_RM970000HT03	RSON	COCT_MT970000HT03.Reason	ObservationEventDx
R_OrderSpecimen	COCT_RM971000HT02	RSON	COCT_MT971000HT02.Reason	ObservationEventGeneral
R_OrderSpecimen	COCT_RM971000HT02	RSON	COCT_MT971000HT02.Reason	ObservationEventDx

Table E-10 Act Configuration Artifacts: CMETs, Client Configured

Artifact Name	ID	AR/Part/Role	Complex Type	Element Name
A_ConditionProblem	COCT_RM030200HT03	IDENT	COCT_MT030200HT03.Person	playedIdentifications
A_DiagnosticReportObservationEvent	COCT_RM120300HT01	PERT	COCT_MT120300HT01.PertinentInformation2	ObservationEventSupporting
A_ObservationEventReaction	COCT_RM120500HT03	PERT	COCT_MT120500HT03.PertinentInformation3	ObservationEventSupporting
A_ObservationGeneral	COCT_RM121500HT02	PERT	COCT_MT121500HT02.PertinentInformation1	ObservationEventSupporting
A_ObservationIntolerance	COCT_RM122500HT02	PERT	COCT_MT122500HT02.PertinentInformation1	ObservationEventSupporting
A_PatientSupplyRequest	COCT_RM920000HT01	PERT	COCT_MT920000HT01.PertinentInformation2	ObservationEventGeneral
A_ProcedureEvent	COCT_RM920100HT01	PERT	COCT_MT920100HT01.PertinentInformation	ObservationEventGeneral
A_SubstanceAdministrationOrder	COCT_RM951000HT01	PERT	COCT_MT951000HT01.PertinentInformation1	ObservationEventGeneral
R_Specimen	COCT_RM080000HT03	PERT	COCT_MT080000HT03.PertinentInformation2	ObservationEventGeneral

F

Appendix F: Abbreviations and Acronyms

Abbreviations and Acronyms

The following table defines abbreviations and acronyms used in the Oracle Healthcare Data Repository Implementation Guide:

Table F-1 Abbreviations and Acronyms

Abbreviation / Acronym	Description
ABG	Arterial blood gasses
ADT	Admit, discharge and transfer
AGS	Administrative grouping of services
AMA	American Medical Association
API	Applications Programming Interface
B2B	Business to Business
B2C	Business to Customer
CA	Certificate Authority
CAT	Computer-Assisted Tomography; See also: CT
CBC	Complete blood count
CDA	Clinical Document Architecture
CDT	Current Dental Terminology
CDT-2	Current Dental Terminology, 2nd Revision
CLIA	Clinical Laboratories Improvement Act
CMS	Centers for Medicare & Medicaid Services; formerly called HCFA
CPT	Current Procedural Terminology
CPT4	Current Procedural Terminology, 4th Revision
CT	Computerized Tomography; See also: CAT
CTB	Oracle Clinical Transaction Base; now HDR
DES	Data Encryption Standard (U.S.)
DHHS	U.S. Department of Health and Human Services
DMIM	Domain Message Information Model
DN	Distinguished Name
DNS	Domain Naming Service
DRG	US Diagnosis Related Group
DRS	Designated Record Set
E&M	Evaluation and Management Guidelines
ECG	Electrocardiogram; electrocardiography

Table F-1 (Cont.) Abbreviations and Acronyms

EEG	Electroencephalogram; Electroencephalography
EHR	Electronic health record
EMPI	Enterprise Master Person Index
EMS	Emergency Medical Services
ER	Emergency Room
ETS	Oracle Enterprise Terminology Services
ETSID	An internal identifier for an ETS entity or structure.
FDB	First Data Bank
GTS	General Timing Specification
GUI	Graphic User Interface
HCFA	Health Care Financing Administration; now CMS
HCPCS	Healthcare Financing Administration Common Procedural Coding System
HCPCS Level II	Healthcare Financing Administration Common Procedural Coding System, Level II
HCSM	Oracle Healthcare Staff Management
HHS	U.S. Department of Health and Human Services
HIPAA	Health Insurance Portability and Accountability Act of 1996
HL7	Health Level 7
HMD	Hierarchical Message Description
HPI	History of present illness
HR	Human Relations; relates to Oracle Human Resource Management.
HRMS	Oracle Human Resource Management.
HSS	U.S. Department of Health and Human Services
HDR	Oracle Healthcare Data Repository; was Oracle Clinical Transaction Base (CTB)
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP combined with underlying SSL layer
ICD-9-CM	International Classification of Diseases-9th Revision, Clinical Modification
ICD-10	International Statistical Classification of Diseases and Health-related problems, 10th Revision
ICU	Intensive Care Unit
IETF	Internet Engineering Task Force
IETF RFC 1766	Internet Engineering Task Force Request for Comments 1766: Tags for the Identification of Languages
II	Instance Identifier
ISO	International Standards Organization
ISO 3166-1	International Standards Organization 3166-1: Country Codes
ISP	Internet Service Provider

Table F-1 (Cont.) Abbreviations and Acronyms

ISV	Independent Software Vendor; Independent Service Vendor
JAR	Java archive file; contains compressed Java classes
JDBC	Java Database Connectivity
JDK	Java Development Kit
JRE	Java Runtime Environment
JSP	Java Server Pages
JVM	Java Virtual Machine
LAN	Local Area Network
LDS	Limited Data Set
LOINC	Logical Observation Identifier of Names and Codes
LOV	List of Values
MDC	US Major Diagnostic Categories
MDF	Message Development Framework (RIM)
MLS	Multiple Language Support
MRI	Magnetic Resonance Imaging; Medical Records Information
MRN	Medical Record Number
MT	Message Type
NDDF	US National Drug Data File
NLS	National Language Support
OHCA	Organized Healthcare Arrangements
OID	Object Identifier; Oracle Internet Directory
PCP	Primary Care Provider; Primary Care Physician
PE	Physical Exam
PHI	Personal Health Information
PIN	Personal Identification Number
PKE	Public Key Encoding
PKI	Public Key Infrastructure
RIM	Reference Information Model (HL7)
RMIM	Refined Message Information Model
SCHIP	State Children's Health Insurance Program (U.S.)
SDO	Standards Developing Organization
SNOMED	Systematized Nomenclature of Medicine
SNOMED CT	Systematized Nomenclature of Medicine Clinical Terms
SQL	Structured Query Language
SSL	Secure Sockets Layer
SSO	Single Sign-on
TCP/IP	Transmission Control Protocol / Internet Protocol
TPO	Treatment, Payment, or Healthcare Operation
UB92	Universal Billing Document [1992]

Table F-1 (Cont.) Abbreviations and Acronyms

UML	Unified Modeling Language
USAM	Unified Service Action Model
WAN	Wide Area Network
XML	Extensible Markup Language