

iCARDEA

“An Intelligent Platform for Personalized Remote Monitoring of the Cardiac Patients with Electronic Implant Devices”

SPECIFIC TARGETED RESEARCH PROJECT

PRIORITY Objective ICT-2009.5.1: Personal Health Systems - a) Minimally invasive systems and ICT-enabled artificial organs: a1) Cardiovascular diseases

iCARDEA Deliverable D6.4.1 Code Mapping API

Due Date: May 31, 2011
Actual Submission Date: May 31, 2011
Project Dates: Project Start Date : February 01, 2010
 Project End Date : January 31, 2013
 Project Duration : 36 months
Leading Contractor Organization: SRDC

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Document History:

Version	Date	Changes	From	Review
V01	September 6, 2010	Initial Document	SRDC	Internal Review
V02	May 11, 2011	Draft Deliverable	SRDC	All partners
V03	May 31, 2011	Final Deliverable	SRDC	All partners

iCARDEA Consortium Contacts:

SRDC	Asuman Dogac	+90-312-2101393	+90(312)2101837	asuman@srdc.com.tr
OFFIS	Wilfried Thoben	+49-441-9722131	+49-441-9722111	thoben@offis.de
SRFG	Manuela Plößnig	+43-662-2288-402	-	manuela.ploessnig@salzburgresearch.at
FORTH	Catherine Chronaki	+302810391691	+302810391428	chronaki@ics.forth.gr
SALK	Bernhard Strohmer	+43-6624482-3481	+43-6624482-3486	b.strohmer@salk.at
SJM	Karl Eberhardt	+43-16073067	-	keberhardt@sjm.com
Medtronic	Alejandra Guillén	34916250361	+34913346453	alejandra.guillen@medtronic.com
HCPB	Josep Brugada	+34932275703	+34932275459	jbrugada@clinic.ub.es

Table of contents

1	PURPOSE	5
1.1	Scope	5
1.2	Definitions and Acronyms	5
2	INTRODUCTION	7
3	UNIFIED MEDICAL LANGUAGE SYSTEM (UMLS)	9
3.1	Metathesaurus	10
3.1.1	Semantic Network	11
3.1.2	SPECIALIST Lexicon and Lexical Tools	11
3.1.3	UMLS Knowledge Source Server	12
3.1.4	MetamorphoSys: The UMLS Installation and Customization Program	13
4	CODE SYSTEMS	13
4.1	LOINC	14
4.1.1	Structure and Purpose of LOINC Codes	15
4.2	SNOMED CT	17
4.2.1	Overview	18
4.3	ICD-10	20
4.4	Rx-NORM	23
4.5	MeSH	24
5	COMMON TERMINOLOGY SERVICES (CTS)	25
5.1	CTS Modules	25
5.1.1	The Message and Vocabulary API	25
6	IMPLEMENTATION OF CTS	27
6.1	Preparing UMLS Knowledge Sources	27
6.2	Used Modules of CTS	29
6.3	Implementation of CTS Web Services	31
6.4	Client Application	32
6.4.1	translateCode	32
6.4.2	validateCode	34
6.4.3	getSupportedCodeSystems	36
6.4.4	getSupportedMaps	39
7	CONCLUSION	41
8	APPENDIX A – WSDL’s of CTS MODULES	41
8.1	Message Runtime	41
8.2	Vocabulary Runtime	56
8.3	Code Mapping	64

1 PURPOSE

1.1 SCOPE

In the healthcare domain, semantic encoding of the data elements is achieved by using coded terms from code systems like LOINC, ICD10 or SNOMED-CT. Even in a single EHR document, several code systems can appear at the same time. In the iCARDEA project, EHR and PHR systems may be using different code systems. This task handles automatic mapping of coded terms from different code systems. To be able to map different code systems, HL7 Common Terminology Services (CTS) is preferred. In order to access the existing terminology servers using CTS, CTS interfaces must be mapped to the interfaces of the terminology servers. For this purpose, one of the most prominent and free terminology servers, namely Unified Medical Language System Knowledge Source Server (UMLS) is used. The implementation is exposed as Web Services as defined by the CTS.

This document describes the achievements in Task 6.4 “Code Mapping among Healthcare Code Systems”. Within the scope of this task, iCARDEA’s Code Mapping API is developed. This API provides the interoperability among the code systems mentioned above.

1.2 DEFINITIONS AND ACRONYMS

Table 1 List of Abbreviations and Acronyms

Abbreviation/ Acronym	DEFINITION
API	Application Programming Interface
AUI	Atom Unique Identifier
CAP	College of American Pathologists
CDC	Centers for Disease Control
CDISC	Clinical Data Interchange Standards Consortium
CIED	Cardiovascular Implantable Electronic Device
CPT	Current Procedural Terminology
CTS	Common Terminology Services
CUI	Concept Unique Identifier
DICOM	Digital Imaging and Communication in Medicine
EHR	Electronic Health Record
FSN	Fully Specified Name
HIPAA	Health Insurance Portability and Accountability Act
HL7	Health Level Seven
ICD	International Statistical Classification of Diseases and Related Health Problems
ICDO	International Classification of Diseases for Oncology
IHTSDO	Health Terminology Standards Development Organisation
LOINC	Logical Observation Identifiers Names and Codes

LUI	Lexical Unique Identifier
MedDRA	Medical Dictionary for Regulatory Activities
MeSH	Medical Subject Headings
NHS	National Health Service
NLM	National Library of Medicine
PHR	Personal Health Record
RELMA	Regenstrief LOINC Mapping Assistant
SNOMED	Systematized Nomenclature of Medicine
SNOMED-CT	Systematized Nomenclature of Medicine – Clinical Terms
SNOP	Systemized Nomenclature of Pathology
SQL	Structured Query Language
SUI	String Unique Identifier
UMLS	Unified Medical Language System
UMLSKS	UMLS Knowledge Source Server
WHO	World Health Organization

2 INTRODUCTION

The iCARDEA system aims to automate and personalize the follow-up of cardiac arrhythmia patients with implanted CIED devices with computer interpretable clinical guideline models using standard device interfaces and integrating patient EHRs.

In iCARDEA, a care plan is personalized to a patient by also accessing her medical history from the EHR systems. For example, in executing iCARDEA care plans for monitoring CIED patients with Atrial Fibrillation (AF), the history of the non-cardiac conditions, detailed information about severity of each condition (e.g., record of prior hospitalizations or specifics of therapy for the condition), the medications being taken at the time of spontaneous arrhythmia occurrence or the non-cardiac conditions denoting contraindications to the proposed therapies need to be accessed from the patient EHRs. The major challenge addressed in accessing the EHR systems is the interoperability problem of communicating with very many heterogeneous EHR systems.

To be able to avoid routinely monitoring a wide variety of clinical data from disparate systems, and developing ad hoc interfaces to access heterogeneous systems, IHE has specified the “Care Management Profile”¹ and this profile is used in the iCARDEA system. Figure 1 shows the overall architecture and the environment in which iCARDEA needs to provide interoperation services.

The major components of the system are as follows:

- Personalized Adaptive Care Planner for the CIED Recipients: In the iCARDEA project, the personalized follow-up of CIED patients is coordinated through a “care plan” which is an executable definition of computer interpretable clinical guideline models. The care plans are represented in GLIF, and the Care Plan Engine is capable of semi-automatically executing the care plan by processing its machine processable definition. The control flow of the care plan is dynamically adapted based on the patient’s context derived from the data coming from CIEDs and the medical context obtained from the EHRs. Through a graphical monitoring tool, the physicians are allowed to follow the execution of the care plan in detail, and coordinate the flow of actions when consultations to physicians are required.
- The CIED Data Exposure Module uses “IHE Implantable Device Cardiac Observation Profile (IDCO)” to expose the CIED data from different vendors in a machine processable format to be used in the care plan of the patients. For this, it has a component that allows accessing the CIED Portal of the vendor and triggers the CIED data export automatically from the CIED Data Center (periodically every x hours or each morning at a defined time). The CIED Data Listener Component waits for the exported data. For this it either scans a configurable directory in case of the data is exported directly to a vendor system in the clinic, alternatively it listens a pre-configured port for the exported data using the IHE IDCO/HL7 v2.5 protocol in case of direct network retrieval. In both cases the PDF file(s) need to be processed to extract the CIED data and the Data Translation Service sub-system creates a valid IHE IDCO format (HL7 v2.5 ORU Message) and makes the CIED data available to the iCARDEA Adaptive Care Planner through PCD-09 Send Observation message.
- EHR Interoperability Infrastructure: To execute the clinical guidelines, it is also necessary to have access to medical history of the patients in the EHR systems. Considering that there are very many EHR systems with proprietary interfaces, in iCARDEA, “IHE Care Management (CM) Profile” is used. In our system, the proprietary hospital information systems export “Discharge Summary” and also “Laboratory Report Summary” CDA documents in conformance to IHE CDA Document templates² to an EHR Server which is implemented as an IHE XDS Repository³. This EHR Server also acts as a “Clinical Data Source” by implementing the IHE CM Profile. In this way, Adaptive Care Manager can subscribe to

¹ IHE Patient Care Coordination (PCC) Technical Framework Supplement, 2008-2009, Care Management (CM), Draft for Trial Implementation, August 22, 2008

² IHE Care Coordination Framework, Content Modules, [http://wiki.ihe.net/index.php?title=1.3.6.1.4.1.19376.1.5.3.1.1#Medical Documents Specification 1.3.6.1.4.1.19376.1.5.3.1.1.1](http://wiki.ihe.net/index.php?title=1.3.6.1.4.1.19376.1.5.3.1.1#Medical_Documents_Specification_1.3.6.1.4.1.19376.1.5.3.1.1.1)

³ IHE Cross Enterprise Document Sharing (XDS) Profile, http://www.ihe.net/Technical_Framework/index.cfm#IT

receive update notifications for the clinical data that is necessary to execute the care plans. IHE Care Management Profile specifies standard interfaces to extract this data that is needed by the care plans from the EHR systems. The two standardized transactions used in the iCARDEA system are as follows:

- “PCC-09 Care Management Data Query” allows querying the clinical data sources such as the EHR systems for the data required to execute the care plan.
- “PCC-10-V3 Care Management Update” allows the clinical data sources (EHR systems) to send the updated clinical data to the subscribed Care management systems as an HL7 V3 messages.

Additionally, IHE has specified “Content Modules” to be used as the payloads of these transactions to transfer clinical data in terms of CDA Sections and Entries. The HL7 Clinical Document Architecture (CDA)⁴ is a document markup standard that specifies the structure and semantics of "clinical documents" for the purpose of exchange and each CDA document is made up of CDA Sections and each Section is made up of CDA Entries.

Different content module templates for CDA Documents such as Discharge Summary, Referral Summary; CDA Sections such as History of Present Illness, Medications, and CDA Entries such as Problem Entry, Vital Signs Observation have been specified.

While a Care manager queries a clinical data source, it specifies the type of the clinical data required through a code specified in the “careProvisionCode” field, such as “LABCAT”, meaning all lab results. For each code specified in this controlled code list, the IHE content module template (for example “Simple Observations” template is specified for reporting lab results) is also specified through which the clinical data update is sent. The clinical data sources send the updated clinical data to the iCARDEA care plan engine by conforming to these content module templates. In this way the interoperability of the transactions among clinical data sources and care managers is guaranteed.

- There is also a Patient Empowerment component that aims to provide active and informed involvement of patients in management of their own health. Through the web based PHR, patients will be able to view their medical history, CIED data, and manage their medication summaries, daily nutrition information.

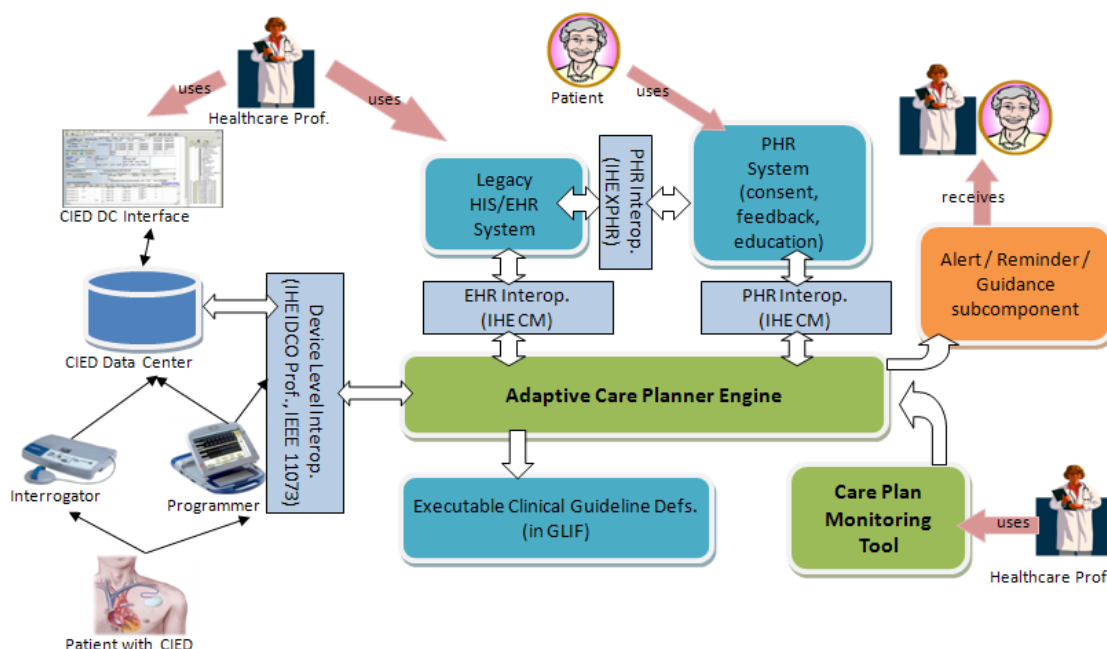


Figure 1 iCARDEA Architecture Overview

⁴ HL7 Clinical Document Architecture (CDA), <http://hl7.org/library/Committees/structure/CDA.ReleaseTwo.CommitteeBallot03.Aug.2004.zip>

Clinical practice guidelines present and formalize medical knowledge required for clinical decision-making and try to standardize the patient care delivery by guiding the healthcare practitioners regarding next actions to be performed. By modeling and converting paper-based follow-up care guidelines into an electronic and executable format, healthcare processes, which need to be achieved as routine follow-ups or remote monitoring of patients with CIEDs, can be automated and hence the workload of healthcare professionals can be decreased by reduced hospitalization and overall efficiency and accuracy of the process can be improved by minimizing costs. Despite the potential benefits of the clinical guidelines, at the moment they are underutilized in clinical practice due to interoperability problems of healthcare data sources. iCARDEA exposes healthcare resources such as electronic healthcare records (EHR), personal healthcare records (PHR) and CIED data of a patient through standard interfaces. Furthermore, the integration problem does not occur only between EHR and PHR systems. Care providers may provide care plans using different types of codes systems, and these code systems may not be used in patients' EHR and PHR systems.

EHR includes patient medical history in coded form, which is needed in decision steps of clinical guidelines. Current clinical healthcare follow-up systems cannot employ available EHR data in their process flows due to the interoperability problems with legacy EHR systems.

The iCARDEA platform provides EHR interoperability so that information about patients' medical history such as history of non-cardiac conditions; more detailed information about severity of each condition (e.g., record of prior hospitalizations or specifics of therapy for the condition); the medications being taken at the time of spontaneous arrhythmia occurrence or the non-cardiac conditions denoting contraindications to the proposed therapies can be obtained from the patient EHR data and used in the clinical workflow.

One of the interoperability challenges in iCARDEA system is the possibility of the use of different code systems by the distinct iCARDEA components. For example, the code system used in the PHR component may be different than the one used in EHR. Even the code system used in the medical careplan definition may be different than the others, because each user use the code system s/he is familiar with. Furthermore, there are many coding systems to express the same clinical data such as ICD10⁵, CPT⁶ or SNOMED⁷.

In the implementation of the iCARDEA Code Mapping API, Unified Medical Language System is used as the terminology server. However, UMLS provides a proprietary API to access its code mappings. Accessing the code mappings that UMLS provided in a standard way would enhance the interoperability. For this purpose, we have used HL7 v3 Common Terminology Services (CTS)⁸. In order to access the existing terminology servers using CTS, CTS interfaces must be mapped to the interfaces of the terminology servers. In this respect, we provide the implementation of CTS functions in terms of the Web Service operations. It should be noted that CTS itself recommends the use of UMLS.

The organization of this deliverable is as follows. First the UMLS is described in detail. After that the supported code systems is presented in Section 4. After introducing the HL7 v3 CTS in Section 5, how the CTS is implemented is described in Section 6, in detail. Finally, Section 7 concludes the deliverable.

3 UNIFIED MEDICAL LANGUAGE SYSTEM (UMLS)

The Unified Medical Language System (UMLS)⁹ is a compendium of many controlled vocabularies in the biomedical sciences. It provides a mapping structure among these vocabularies and thus allows one to translate among the various terminology systems; it may also be viewed as a comprehensive thesaurus and ontology of biomedical concepts. The project was initiated in 1986.

UMLS consists of the following components:

5 <http://www.who.int/classifications/icd/en/>

6 <http://www.ama-assn.org/ama/pub/category/3113.html>

7 <http://www.snomed.org>

8 HL7 Common Terminology Services (CTS), Release 1, <http://www.hl7.org/v3ballot/html/infrastructure/cts/cts.htm>

9 <http://www.nlm.nih.gov/research/umls/>

- Metathesaurus, the core database of the UMLS, a collection of concepts and terms from the various controlled vocabularies, and their relationships;
- Semantic Network, a set of categories and relationships that are being used to classify and relate the entries in the Metathesaurus;
- SPECIALIST Lexicon, a database of lexicographic information for use in natural language processing;
- Web based Knowledge Source Server, UMLS Terminology Services (UTS)¹⁰ that uses the above mentioned components for finding clinical concepts, retrieving concept details and mapping clinical terms.
- a number of supporting software tools.

The UMLS was designed and is maintained by the US National Library of Medicine, is updated quarterly and may be used for free. The users of the system have to sign a "UMLS agreement" and to file brief annual reports on their use. Academic users can employ the UMLS free of charge for research. Commercial or production use requires copyright licenses for some of the incorporated source vocabularies. In the following subsections, the components of the UMLS are described.

3.1 METATHESAURUS

The Metathesaurus is a large, multi-purpose, and multi-lingual vocabulary database that contains information about biomedical and health-related concepts, their various names and the relationships among them. It is built from the electronic versions of numerous thesauri, classifications, code sets, and lists of controlled terms used in patient care, health services billing, public health statistics, indexing biomedical literature, and/or basic, clinical, and health services research. In this deliverable, these are referred to as the "source vocabularies" of the Metathesaurus. In the Metathesaurus, all the source vocabularies are available in a common, fully-specified database format.

The Metathesaurus forms the base of the UMLS and is organized by concept. In essence, it links alternative names and views of the same concept and identifies useful relationships between different concepts. The Metathesaurus comprises over 1 million biomedical concepts and 5 million concept names, all of which stem from the over 100 incorporated controlled vocabularies and classification systems. Some examples of the incorporated controlled vocabularies are ICD-10, MeSH, SNOMED CT, LOINC, WHO Adverse Drug Reaction Terminology, UK Clinical Terms, RxNORM, Gene Ontology, and OMIM.

Each concept in the UMLS represents a distinct meaning and they are identified by concept identifiers, called Concept Unique Identifier (CUI). The concepts have terms that are coded meanings in source vocabularies. As an example, the "Addison's disease" concept has the CUI C0001403 and it has the following terms:

Table 2 Code of Addison's Disease in different source vocabularies

Term name	Source vocabulary	Code in the Source Vocabulary
Addison's disease	SNOMED CT	363732003
Addison's Disease	MedlinePlus	T1233
Addison Disease	MeSH	D000224
Bronzed disease	SNOMED Intl	DB-70620
Primary Adrenal Insufficiency	MeSH	D000224
Primary hypoadrenalism syndrome, Addison	MedDRA	10036696

All concepts in the Metathesaurus are assigned at least one Semantic Type from the Semantic Network to provide consistent categorization at the relatively general level represented in the Semantic Network. Many of the words and multi-word terms that appear in concept names or strings in the Metathesaurus also appear in the SPECIALIST Lexicon. The Lexical Tools are used to generate

10 "Unified Medical Language System (UMLS) Terminology Services (TS)," <https://uts.nlm.nih.gov/home.html>

the word, normalized word, and normalized string indexes to the Metathesaurus. MetamorphoSys¹¹ is used to install the UMLS Knowledge Sources and customize the Metathesaurus.

3.1.1 Semantic Network

The Semantic Network provides a consistent categorization of all concepts represented in the Metathesaurus and provides a set of useful relationships between these concepts. All information about specific concepts is found in the Metathesaurus; the Network provides information about the set of basic Semantic Types, or categories, which may be assigned to these concepts, and it defines the set of relationships that may hold between the Semantic Types. The Semantic Network contains 133 Semantic Types and 54 relationships. The Semantic Network serves as an authority for the Semantic Types that are assigned to concepts in the Metathesaurus. The Network defines these types, both with textual descriptions and by means of the information inherent in its hierarchies. As an example, Addison's Disease's Semantic Type is "Disease or Syndrome", and there is a "causes" relationship between "Virus" and "Disease or Syndrome" Semantic Types. In Figure 2, a part of the Semantic Network is depicted.

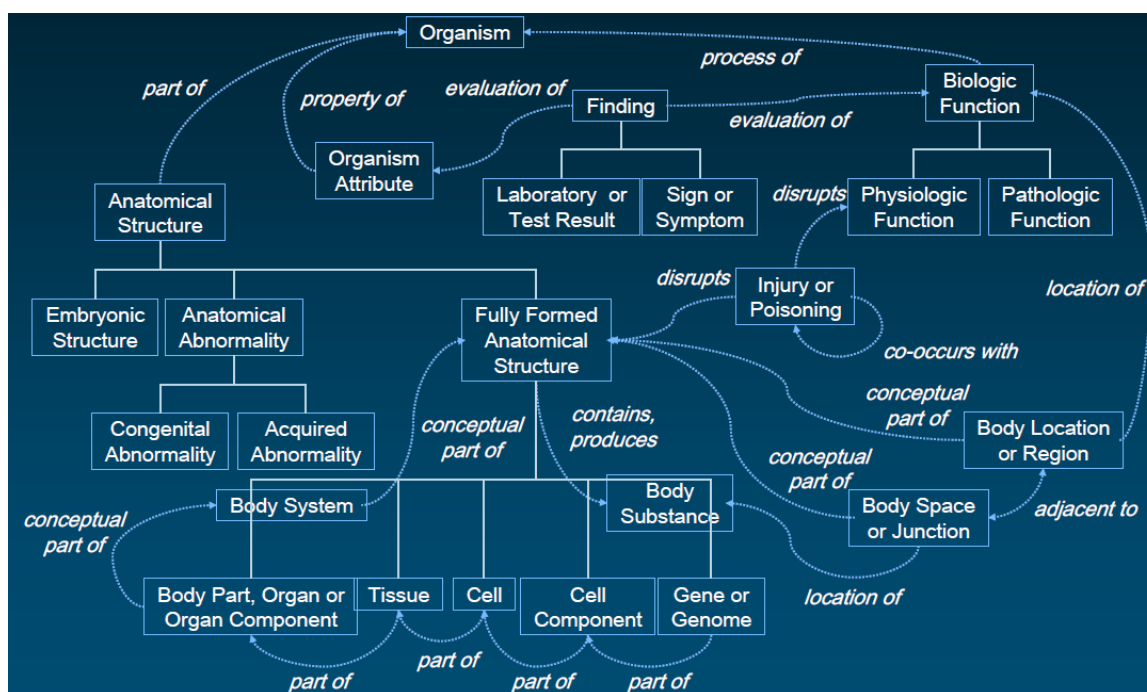


Figure 2 A part of Semantic Network

The Semantic Types are the nodes in the Network, and the Semantic Relations between them are the links. There are major groupings of Semantic Types for organisms, anatomical structures, biologic function, chemicals, events, physical objects, and concepts or ideas. The current scope of the UMLS Semantic Types is quite broad, allowing for the semantic categorization of a wide range of terminology in multiple domains.

3.1.2 SPECIALIST Lexicon and Lexical Tools

The SPECIALIST Lexicon contains information about common English vocabulary, biomedical terms, terms found in MEDLINE and in the UMLS Metathesaurus. Each entry contains syntactic (how words are put together to create meaning), morphological (form and structure) and orthographic (spelling) information. The tools are mainly used to generate the word, normalized word, and normalized string indexes to the Metathesaurus. An example is provided in **Figure 3** and **Figure 4**. There are a number of terms for Hodgkin Disease as shown in **Figure 3**. In order to unify them to one

¹¹ <http://www.ncbi.nlm.nih.gov/books/NBK9683/>

entry the steps in **Figure 4** are applied to the terms. In this way all of the terms are normalized to “disease hodgkin” which may be used the name for the concept in the metathesaurus.

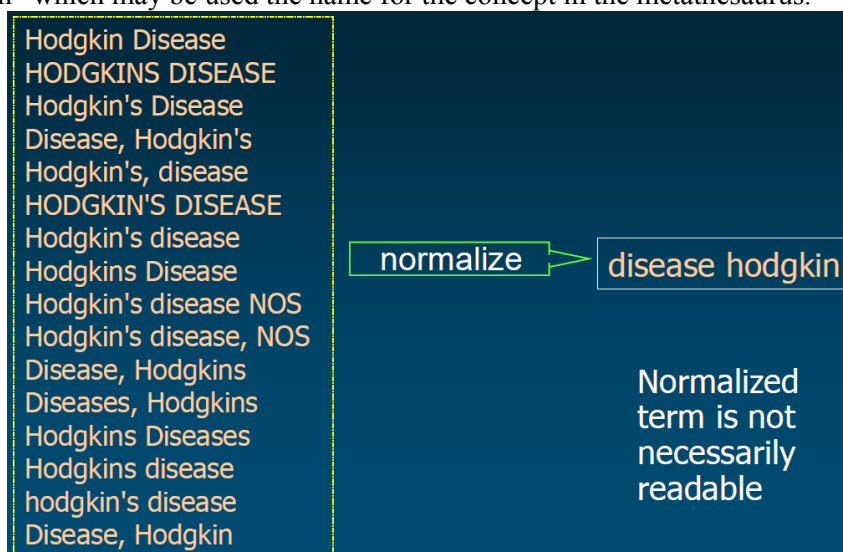


Figure 3 Normalization Example

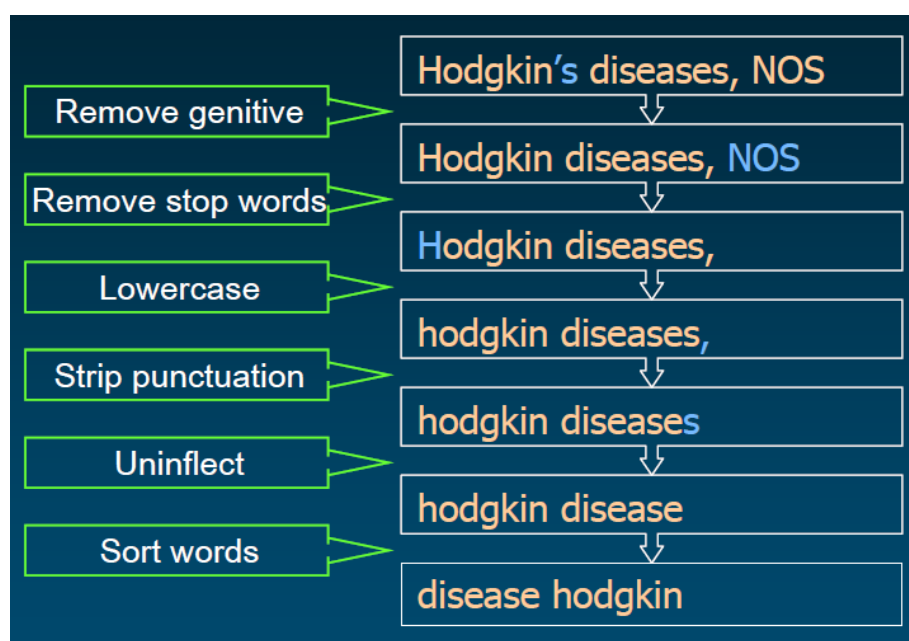


Figure 4 Normalization Steps

3.1.3 UMLS Terminology Services (UTS)

UMLS Terminology Services (UTS) provide both web interfaces as well as Web Services to search and retrieve UMLS data. UTS replaces the UMLS Knowledge Source Server (UMLSKS) with new features and improved access to UMLS and other NLM terminology resources. The UMLSKS was a set of Web-based interactive tools and a programmer interface that allows users and developers to access the UMLS components, including the vocabularies within the Metathesaurus. The UTS is a useful starting point for gaining an understanding of the content of the UMLS resources. Because it contains the complete Metathesaurus files, access to UTS is restricted to registered users who have signed the License Agreement for use of the UMLS Metathesaurus. Below, a screenshot of a query from UTS source view screen is presented in **Figure 5**.

The screenshot shows the UMLS Terminology Services Metathesaurus Browser. The search bar contains the term 'headache'. The search results list 522 items, including 'Headache', 'ENT surgical result nose headache', 'No headache', 'Aural headache', 'Cluster Headache', 'Dental headache', 'Nasal headache', 'Ocular Headache', 'Postseizure headache', 'Post-Traumatic Headache', 'Tension Headache', 'Sinus headache', 'Vascular Headaches', 'Migraine Disorders', 'Chronic Headache', 'Occipital headache', 'Allergic headache', and 'Headache dull'. The main view shows the 'Basic View' for the concept [C0018681] Headache, including semantic types, definitions, and atoms.

Search Results (522)

- C0018681 Headache
- C2096315 ENT surgical result nose headache
- C0423624 No headache
- C0004317 Aural headache
- C0009088 Cluster Headache
- C0011364 Dental headache
- C0027427 Nasal headache
- C0028838 Ocular Headache
- C0032810 Postseizure headache
- C0032816 Post-Traumatic Headache
- C0033893 Tension Headache
- C0037195 Sinus headache
- C0042376 Vascular Headaches
- C0149931 Migraine Disorders
- C0151293 Chronic Headache
- C0231613 Occipital headache
- C0234240 Allergic headache
- C0239885 Headache dull

Basic View

Concept: [C0018681] Headache

Semantic Types

[Sign or Symptom \[T184\]](#)

Definitions

CSP/PT | pain in the cranial region that may occur as an isolated and benign symptom or as a manifestation of a wide variety of conditions.

MSH/MH | The symptom of PAIN in the cranial region. It may be an isolated benign occurrence or manifestation of a wide variety of HEADACHE DISORDERS.

NCI/CTCAEPT | A disorder characterized by a sensation of marked discomfort in various parts of the head, not confined to the area of distribution of any nerve.

NCI/PT | Pain in various parts of the head, not confined to the area of distribution of any nerve.

Atoms (307) string [AUI / RSAB / TTY / Code]

- cephalalgia [A1304573/AOD/DE/0000006116]
- headache [A0480505/AOD/DE/0000006115]
- headache, cephalalgia [A1389175/AOD/DE/0000006117]
- ha [A1412440/BI/AB/BI00020]
- headache [A0480506/BI/PT/BI00020]
- headaches [A1412439/BI/RT/BI00020]
- HA [A1619730/CCPSS/MP/U001223]
- HEADACHE [A0418070/CCPSS/PT/1014497]
- Cephalgia [A0400088/COSTAR/PT/NOCODE]
- HEADACHE [A0418071/COSTAR/PT/343]
- Headaches [A0066006/COSTAR/PT/NOCODE]
- cephalalgia [A1304574/CSP/ET/2056-6161]
- cephalgia [A1304575/CSP/ET/2056-6161]

Figure 5 A query from UMLS and Source View

3.1.4 MetamorphoSys: The UMLS Installation and Customization Program

MetamorphoSys¹² is a cross-platform Java application that must be used if the UMLS Knowledge Sources (Metathesaurus, Semantic Network, and SPECIALIST Lexicon) are installed locally. MetamorphoSys also supports the creation and refinement of customized subsets of the Metathesaurus. In general, the Metathesaurus must be customized to be used effectively in specific applications.

MetamorphoSys guides the users first through the installation of one or more UMLS Knowledge Sources, and then through customization of the Metathesaurus for local use. A variety of options are available, such as the inclusion or exclusion of specific source vocabularies, languages, and term types, specification of output character set (7-bit ASCII or Unicode UTF-8) and output format (Rich Release Format or Original Release Format) for the Metathesaurus files.

MetamorphoSys is used to install UMLS database locally to be used in Code Mapping API framework of iCARDEA Project.

4 CODE SYSTEMS

Code Systems are systematically organized computer processable collection of coded representation of meanings used in clinical information covering the areas such as diseases, findings, procedures, observations, pharmaceuticals, etc. They allow a consistent way to index, store, retrieve and aggregate clinical data across specialties and sites of care. They also help organizing the content of medical records, reducing the variability in the way data is captured, encoded and used for clinical care of patients and research. For example, the terms “heart attack” and “myocardial infarction” may mean the same thing. However, to make automated processing of the terms possible and to be able to

¹² <http://www.ncbi.nlm.nih.gov/books/NBK9683/>

exchange clinical information consistently between different health care providers, care settings, researchers and others, medical terminology systems are used. This deliverable documents prominent Code Systems and covers:

- Logical Observation Identifiers Names and Codes (LOINC)¹³,
- SNOMED CT (Systematized Nomenclature of Medicine -- Clinical Terms)¹⁴,
- International Statistical Classification of Diseases and Related Health Problems 10th Revision (ICD-10)¹⁵
- RxNorm¹⁶
- Medical Subject Headings (MeSH)¹⁷

They are described in the following subsections, respectively.

4.1 LOINC¹⁸

The Logical Observation Identifier Names and Codes¹⁹ (LOINC®) database provides a universal code system for reporting laboratory and other clinical observations. It was developed by the Regenstrief Institute Inc., a US non-profit medical research organization, in 1994. Its purpose is to identify observations in electronic messages such as Health Level Seven (HL7) observation messages, so that when hospitals, health maintenance organizations, pharmaceutical manufacturers, researchers, and public health departments receive such messages from multiple sources, they can automatically file the results in the right slots of their medical records, research, and/or public health systems. For each observation, the database includes a code (of which 25,000 are laboratory test observations), a long formal name, a “short” 30-character name, and synonyms. The database comes with a mapping program called Regenstrief LOINC Mapping Assistant (RELMA) to assist the mapping of local test codes to LOINC codes and to facilitate browsing of the LOINC results. Both LOINC and RELMA are available at no cost from <http://www.regenstrief.org/loinc/>. The LOINC medical database carries records for >30,000 different observations.

Before 1994, no universal pre-coordinated code system for laboratory test names existed, although considerable background work had been done within organizations such as the IFCC/IUPAC Committee/Commission on Properties and Units in Clinical Chemistry, and EUCLIDES. In 1994, a group of researchers met in Indianapolis at the Regenstrief Institute to begin the development of such a system, which they called the Logical Observation Identifiers Names and Codes (LOINC®) code system. The initial release, in the spring of 1995, included a 70-page Users’ Guide and identifiers and names for more than 6000 laboratory test results. Since this first release, the Regenstrief Institute and the LOINC Committee have delivered 17 releases, increased the size of the database fivefold, added codes for many clinical subjects beyond the laboratory and added short names for laboratory tests.

LOINC codes are now being used by large reference laboratories and federal agencies, e.g., the CDC and the Department of Veterans Affairs, and are part of the Health Insurance Portability and Accountability Act (HIPAA) attachment proposal. Internationally, they have been adopted in Switzerland, Hong Kong, Australia, and Canada, and by the German national standards organization, the Deutsches Institut für Normung.

¹³ Logical Observation Identifiers Names and Codes (LOINC) [Internet]. <http://www.regenstrief.org/loinc/>.

¹⁴ The Systematized Nomenclature of Medicine (SNOMED) [Internet]. International Health Terminology Standards Development Organization; Available from: <http://www.ihtsdo.org/snomed-ct/>

¹⁵ International Statistical Classification of Diseases and Related Health Problems, 10th Revision (ICD-10), Second Edition [Internet]. World Health Organization, Geneva, Switzerland, Tech. Rep., (2005). Available from: <http://www.who.int/whosis/icd10/>

¹⁶ <http://www.nlm.nih.gov/research/umls/rxnorm/>

¹⁷ <http://www.nlm.nih.gov/mesh/>

¹⁸ Clement J. McDonald et.al, LOINC, a Universal Standard for Identifying Laboratory Observations: A 5-Year Update, *Clinical Chemistry* 49:4, 624–633 (2003)

¹⁹ <http://loinc.org/>

4.1.1 Structure and Purpose of LOINC Codes

The initial purpose of the LOINC database was to provide universal identifiers for observations in HL7 messages. Specifically, LOINC provides a code system for the observation identifier field (OBX-3) of the HL7 observation reporting messages. However, LOINC is now being used in Digital Imaging and Communication in Medicine (DICOM) ultrasound messages and in Clinical Data Interchange Standards Consortium (CDISC) pharmaceutical industry messages to identify clinical and laboratory observations, respectively, and could well be used in clinical and research databases for the same purpose.

LOINC codes exist for laboratory observations such as partial pressure of arterial blood oxygen (Po₂) and percentage lymphocytes; electrocardiogram (EKG) measurements, such as PR interval; vital signs, such as pulse, body weight, and height; and for many other clinical domains. Laboratory result values (which are stored in the HL7 field OBX-5) are often reported as numbers, but depending on the nature of the test observation, they may also be reported as free text or as codes. Some laboratory systems report blood types, for example, as codes. The scope of the LOINC Committee includes the codes that identify the test observation per se, e.g., serum glucose or blood culture, not the codes that might be reported in the values of some test observations. If we consider the observation as a question and the observation values as answers, LOINC provides codes for the questions. Other code systems, e.g., International Classification of Diseases (ICD)-9, International Classification of Diseases for Oncology (ICDO)-3, Systemized Nomenclature in Medicine (SNOMED), MEDCIN, and the Medical Dictionary for Regulatory Activities (MedDRA), provide codes for the answers.

The LOINC database carried records for more than 30,000 different observations. Each record carries the formal six-part LOINC name; the LOINC code, a number with a check digit; the observation class (e.g., chemistry, hematology, and radiology); related names (to assist searches of the database); and other attributes (see **Table 3** for an example). For most classes of laboratory observations, the database also includes a “short” report name, which is <30 characters, is less formal and is more readable.

Table 3 Examples of laboratory LOINC codes and formal

LOINC code	LOINC name (<i>component:property:timing:specimen:scale</i>)
2951-2	SODIUM:SCNC:PT:SER/PLAS:QN
2955-3	SODIUM:SCNC:PT:UR:QN
2956-1	SODIUM:SRAT:24H:UR:QN
2164-2	CREATININE RENAL CLEARANCE:VRAT:24H:UR:QN
1514-9	GLUCOSE^2H POST 100 G GLUCOSE PO:MCNC:PT:SER/PLAS:QN
3665-7	GENTAMICIN^TROUGH:MCNC:PT:SER/PLAS:QN
17863-2	CALCIUM.IONIZED:MCNC:PT:SER/PLAS:QN
2863-9	ALBUMIN:MCNC:PT:SNV:QN:ELECTROPHORESIS

LOINC clinical observation names are defined in terms of six major axes. The formal LOINC name must include entries for the first six major axes shown in **Table 4**.

Table 4 LOINC axes

Number	Description
1	Component (analyte): e.g., potassium, hemoglobin, hepatitis B antigen
2	Property measured: e.g., a mass concentration, enzyme activity (catalytic rate)
3	Timing: i.e., whether the observation applies to a moment in time or is an average or amount taken

4	over a period of time, as is the case for a 24-h urine sodium concentration System: i.e., type of sample or organ examined: e.g., urine, blood, chest
5	Scale: e.g., whether the measurement is quantitative (a true measurement), ordinal (a ranked set of options), nominal, or narrative (e.g., dictation results from x-rays)
6	Method used to produce the observation, but only when different methods give clinically significant different result

The LOINC Committee divides the LOINC development into three divisions; the first of these is laboratory LOINC. The early years of the LOINC development were focused exclusively on laboratory observations and large adopters of LOINC laboratory continue to stimulate expansion of the laboratory terminology. Hence, the laboratory content, which is the focus of this report, is the best developed and largest of the three divisions. The categories of laboratory test procedures that are included in the LOINC database are identified in **Table 5** along with the number of LOINC terms defined for each category.

Table 5 Laboratory LOINC Scope

Class	# of terms	Class	# of terms
Antibiotic susceptibilities	1010	Hematology cell count	1082
Blood bank	658	Microbiology	5786
Chemistry	4817	Molecular pathology	245
Coagulation	346	Skin tests	25
Cytology	31	Pathology	124
Fertility	123	Drug & toxicology	3919
Flow cytometry cell markers	580	Urinalysis	136

The second division is the clinical LOINC division and it is concerned with non-laboratory diagnostic studies, critical care, and nursing measures, as well as the history, physical, and survey instruments. The clinical LOINC division includes several new projects for defining clinical notes, report titles, and dental observations. The categories and numbers of clinical LOINC terms per categories that are available in the LOINC database are described in **Table 6**.

Table 6 Clinical LOINC Scope

Class	# of terms	Class	# of terms
Body measurements	61	History and physical	324
Cardiac ultrasound	487	Obstetrical ultrasound	562
Clinical documents headers	19	Ophthalmology measurements	477
Colonoscopy/Endoscopy	72	Radiology reports	1177
EKG (ECG)	411	Respiratory therapy	33
Emergency department	34	Standard survey instruments (41)	460
Fluid intake/output	400	Tumor registry	246

The third division focuses on proposals for the Health Insurance Portability and Accountability Act (HIPAA) attachments and this division deals with payment, enrolment, and other purely administrative functions.

4.2 SNOMED CT

SNOMED CT²⁰ (Systematized Nomenclature of Medicine -- Clinical Terms), is a systematically organized computer processable collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, microorganisms, and pharmaceuticals.

The history of SNOMED CT is shown in **Figure 6**. The College of American Pathologists (CAP) introduced SNOMED® in 1978. SNOMED derived from SNOP (Systemized Nomenclature of Pathology), which was introduced in 1965 and expanded this lexicon to include all of general medicine²¹. Further significant revisions included SNOMED-II in 1979 and SNOMED-III (International) in 1993. After that SNOMED RT (Reference Terminology) emerged in 2001 and represented a significant update while remaining compatible with SNOMED International, presenting data in a completely machine-readable format. What had previously been a relatively flat, multi-axial system became a true semantic network. In January 2002, SNOMED CT was created by the merger, expansion, and restructuring of the SNOMED RT and the UK National Health Service (NHS) Clinical Terms (also known as the Read codes). On 26 April 2007, the Health Terminology Standards Development Organisation (IHTSDO®, also known as SNOMED SDO®) acquired the ownership of SNOMED Clinical Terms (SNOMED CT®) and will be responsible for future maintenance and development. In April 2002, the SNOMED CT Spanish Edition and in April 2003 the SNOMED CT German Edition were released.

The historical strength of the Systematized Nomenclature of Medicine (SNOMED) was its terminologies for specialty medicine, while the strength of Clinical Terms Version 3 was its terminologies for general practice. By combining these two systems, SNOMED CT is the most comprehensive clinical vocabulary available in any language, covering most aspects of clinical medicine with over 350,000 concepts. SNOMED CT crossmaps to other terminologies such as ICD-10 and LOINC as well.

SNOMED CT has been licensed in at least 30 countries worldwide. SNOMED CT has been adopted as the preferred terminology of the England and Wales NHS. Furthermore, a five year renewable license agreement between the CAP and the US National Library of Medicine (NLM) provides free access to the English and Spanish language editions of the SNOMED CT Core content and all version updates through the NLM's UMLS Metathesaurus, which is a knowledge source containing biomedical concepts and terms from approximately 100 source vocabularies and classifications.

²⁰ The Systematized Nomenclature of Medicine (SNOMED) [Internet]. International Health Terminology Standards Development Organization; Available from: <http://-www.ihtsdo.org/snomed-ct/>

²¹ <http://www.openclinical.org/medTermSnomedCT.html>

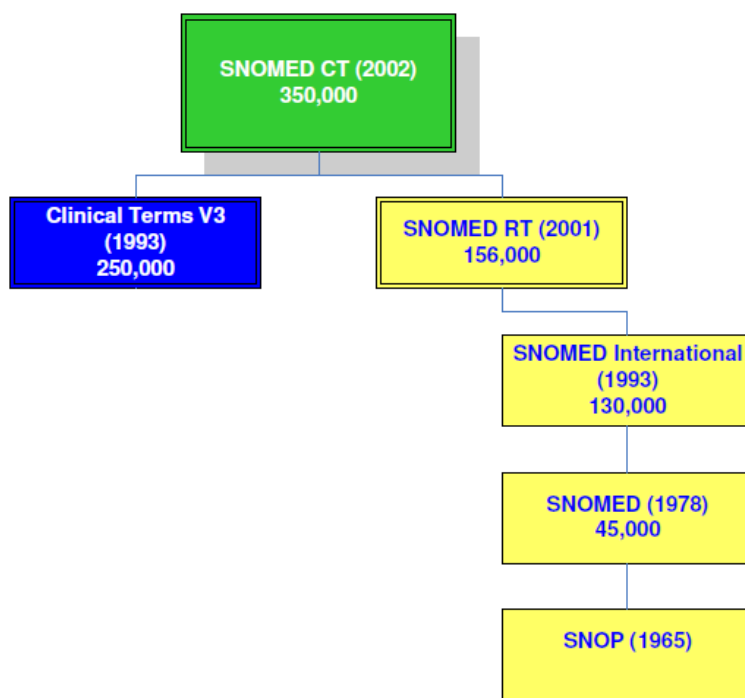


Figure 6 SNOMED CT History

4.2.1 Overview

SNOMED is the largest reference terminology covering clinical domain, developed in native description logics (DL) formalism. It contains approximately 350,000 active concepts, more than one million terms (incl. synonyms) and about 1.5 million relations between the concepts.

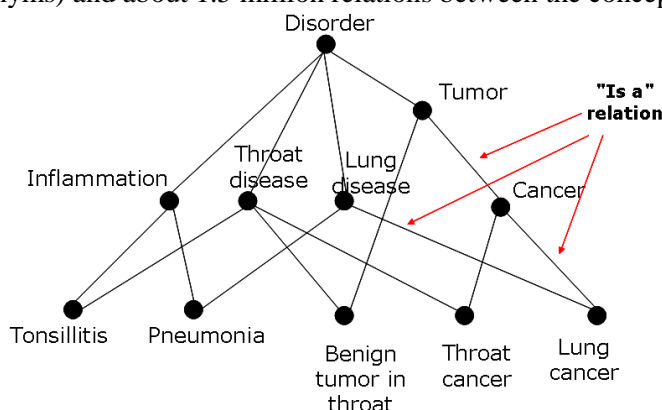


Figure 7 The architecture of SNOMED CT

The terminology is comprised of concepts, descriptions and relationships with the objective of precisely representing clinical information across the scope of health care.

A “concept” is a clinical meaning identified by a unique numeric identifier (ConceptID) that never changes. ConceptIDs do not contain hierarchical or implicit meaning. The numeric identifier does not reveal any information about the nature of the concept.

Concepts are divided into hierarchies, which include: *Clinical finding, Physical force, Procedure, Event, Observable entity, Environment or geographical location, Body structure, Social context, Organism, Situation with explicit context, Substance, Staging and scales, Pharmaceutical/biologic product, Physical object, Specimen, Qualifier value, Special concept, Record artifact and Linkage concept.*²²

²² http://storage.pardot.com/604/40385/SNOMED_CT_User_Guide.pdf

Concepts are represented by a unique human-readable Fully Specified Name (FSN). The concepts are formally defined in terms of their relationships with other concepts. These logical definitions give explicit meaning which a computer can process and query on. Every concept also has a set of terms that name the concept in a human-readable way. Concepts represent various levels of clinical detail (Figure 8). Concepts can be very general or they can represent increasingly specific levels of detail, also referred to as increasing granularity. Multiple levels of granularity improve the capability to code clinical data at the appropriate level of detail.

Concept descriptions are the terms or names assigned to a SNOMED CT concept. “Term” in this context means a phrase used to name a concept. Multiple descriptions might be associated with a concept identified by a ConceptID. As an example, some of the descriptions associated with ConceptID 22298006 is as follows:

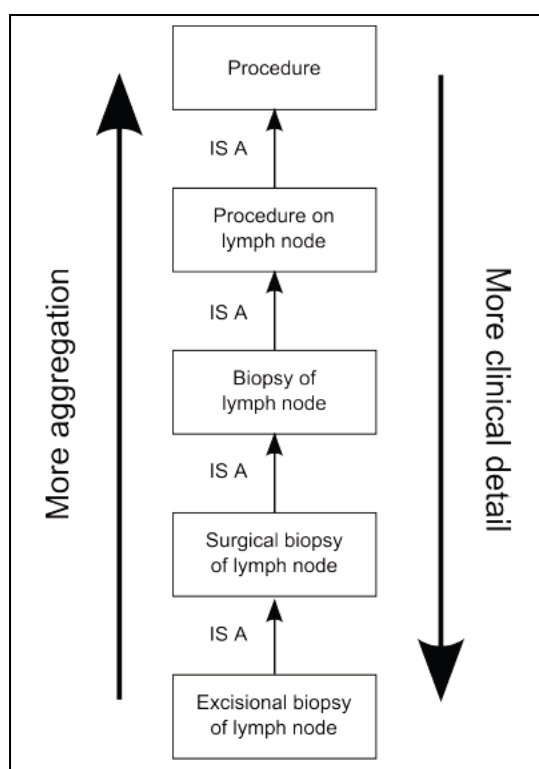


Figure 8 Multiple levels of granularity

- Fully Specified Name: Myocardial infarction (disorder)
- Preferred term: Myocardial infarction
- Synonym: Cardiac infarction
- Synonym: Heart attack
- Synonym: Infarction of heart

There are three types of descriptions:

- Fully Specified Name (FSN): Each concept has one unique FSN intended to provide an unambiguous way to name a concept. The purpose of the FSN is to uniquely identify a concept and clarify its meaning, not necessarily to present the most commonly used or natural phrase for that concept.

- Preferred Term: Each concept has one Preferred Term meant to capture the common word or phrase used by clinicians to name that concept. For example, the concept 54987000 Repair of common bile duct (procedure) has the Preferred Term “Cholelithotomy” to represent a common name clinicians use to describe the procedure.

- Synonym: Synonyms represent any additional terms that represent the same concept as the FSN.

Each concept in SNOMED CT is logically defined through its relationships to other concepts.

There are two types of defining relationships: (1) IS A

and (2) Attributes. Every active SNOMED CT concept (except the SNOMED CT Concept Root concept) has at least one IS A relationship to a supertype concept. Attributes relate two concepts and establish the type of relationship between them. Like concept, these attributes have a hierarchical relationship to one another known as “attribute hierarchies”. For example:

Fracture of tarsal bone (disorder)

- IS A Fracture of foot (disorder)
- FINDING SITE Bone structure of tarsus (body structure)
- ASSOCIATED MORPHOLOGY Fracture (morphologic abnormality)

SNOMED CT currently uses over 50 defining attributes to model concept definitions. Each SNOMED CT attribute can usually be applied to one hierarchy. The hierarchy or hierarchies to which an attribute can be applied are referred to as the “domain” of the attribute. Each attribute can be given a limited set of values; this set of values is called the “range” of the attribute. For example, the Domain of the attribute ASSOCIATED MORPHOLOGY is the Clinical finding hierarchy and a Procedure cannot have an ASSOCIATED MORPHOLOGY. Furthermore, Procedure has a PROCEDURE MORPHOLOGY.

On the other hand, the Range is the set of values allowed for each attribute. For example, the Range for ASSOCIATED MORPHOLOGY is Morphologically abnormal structure (morphologic abnormality) and its descendants, and the range for FINDING SITE is Anatomical or acquired body structure (body structure) and its descendants in the Body structure hierarchy.

Like concepts, there is an attribute hierarchy, where one general attribute is the parent of one or more specific subtypes of that attribute. The attributes are divided into following classifications:

- Attributes used to define Clinical Finding concepts
- Attributes used to define Procedure concepts
- Attributes used to define Evaluation Procedure concepts
- Attributes used to define Specimen concepts
- Attributes used to define Body structure concepts
- Attributes used to define Pharmaceutical/Biologic Product concepts
- Attributes used to define Situation with Explicit Context concepts
- Attributes used to define Event concepts
- Attributes used to define Physical Object concepts

In SNOMED, if concepts are referred to directly by an information model then they are considered to be 'pre-coordinated'. However, SNOMED CT also enables more complex descriptions to be used. For example, there might not be an explicit concept for a burn between the toes, but it could be described as:

```
284196006|Burn of skin|
 246112005|Severity|=24484000|severe,
363698007|Finding Site|=
(113185004|Structure of skin between fourth and fifth toes|:272741003|Laterality|=7771000|left)
```

Such expressions are said to have been 'post-coordinated'. When used, post coordinated conditions avoids the need to create large numbers of Defined Concepts within SNOMED. However, many systems only use pre-coordinated conditions.

Reliable analysis and comparison of any such post-coordinated expressions - with respect to both those concepts already within the SNOMED CT release dataset and any other ad hoc concepts created or yet to be created by its community of end users - properly requires the application of an appropriate description logic classification algorithm. In this way, it becomes possible to apply description logic reasoning to any new candidate post-coordinated expressions in order to assess whether it is a parent or ancestor of, a child or other descendent of, or semantically equivalent to any existing concept from the 350,000 pre-coordinated concepts which are already distributed worldwide.

4.3 ICD-10

The International Statistical Classification of Diseases and Related Health Problems 10th Revision (ICD-10)²³ is a coding of diseases and signs, symptoms, abnormal findings, complaints, social circumstances and external causes of injury or diseases, as classified by the World Health Organization (WHO)²⁴. The code set allows more than 155,000 different codes and permits tracking of many new diagnoses and procedures. It is used to classify diseases and other health problems recorded on many types of health and vital records including death certificates and health records. In addition to enabling the storage and retrieval of diagnostic information for clinical, epidemiological and quality purposes, these records also provide the basis for the compilation of national mortality and morbidity statistics by WHO Member States.

²³ International Statistical Classification of Diseases and Related Health Problems, 10th Revision (ICD-10), Second Edition [Internet]. World Health Organization, Geneva, Switzerland, Tech. Rep., (2005). Available from: <http://www.who.int/whosis/icd10/>

²⁴ <http://www.who.int/entity/classifications/icd/en/bluebook.pdf>

ICD-10 was endorsed by the forty-third World Health Assembly in May 1990 and came into use in WHO Member States as from 1994. The classification is the latest in a series which has its origins in the 1850s. The first edition, known as the International List of Causes of Death, was adopted by the International Statistical Institute in 1893. WHO took over the responsibility for the ICD at its creation in 1948 when the Sixth Revision, which included causes of morbidity for the first time, was published. The World Health Assembly adopted in 1967 the WHO Nomenclature Regulations that stipulate use of ICD in its most current revision for mortality and morbidity statistics by all Member States.

There are 22 chapters in the current version of ICD-10 as shown in **Table 7**.

Table 7 ICD-10 Chapters

International Statistical Classification of Diseases and Related Health Problems 10th Revision		
Chapter	Blocks	Title
I	A00-B99	Certain infectious and parasitic diseases
II	C00-D48	Neoplasms
III	D50-D89	Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism
IV	E00-E90	Endocrine, nutritional and metabolic diseases
V	F00-F99	Mental and behavioural disorders
VI	G00-G99	Diseases of the nervous system
VII	H00-H59	Diseases of the eye and adnexa
VIII	H60-H95	Diseases of the ear and mastoid process
IX	I00-I99	Diseases of the circulatory system
X	J00-J99	Diseases of the respiratory system
XI	K00-K93	Diseases of the digestive system

XII	L00-L99	Diseases of the skin and subcutaneous tissue
XIII	M00-M99	Diseases of the musculoskeletal system and connective tissue
XIV	N00-N99	Diseases of the genitourinary system
XV	O00-O99	Pregnancy, childbirth and the puerperium
XVI	P00-P96	Certain conditions originating in the perinatal period
XVII	Q00-Q99	Congenital malformations, deformations and chromosomal abnormalities
XVIII	R00-R99	Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified
XIX	S00-T98	Injury, poisoning and certain other consequences of external causes
XX	V01-Y98	External causes of morbidity and mortality
XXI	Z00-Z99	Factors influencing health status and contact with health services
XXII	U00-U99	Codes for special purposes

The ICD-10 codes uses five-character (one of which is the point character) scheme as shown in Figure 9. Generally, the first three characters are used to identify a disease and the fifth character after the point is used to further identify subdivisions if there is any.



Figure 9 ICD-10 Coding Scheme

For example, the codes for “Gastritis and duodenitis” and its subdivisions are as follows:

(K29.) Gastritis and duodenitis

- (K29.0) Acute haemorrhagic gastritis
- (K29.1) Other acute gastritis
- (K29.2) Alcoholic gastritis
- (K29.3) Chronic superficial gastritis
- (K29.4) Chronic atrophic gastritis

- (K29.5) Chronic gastritis, unspecified
- (K29.6) Other gastritis
- (K29.7) Gastritis, unspecified
- (K29.8) Duodenitis
- (K29.9) Gastroduodenitis, unspecified

4.4 RX-NORM

RxNorm, a standardized nomenclature for clinical drugs and drug delivery devices, is produced by the National Library of Medicine (NLM). In this context, a clinical drug is a pharmaceutical product given to (or taken by) a patient with a therapeutic or diagnostic intent²⁵. A drug delivery device is a pack that contains multiple clinical drugs or clinical drugs designed to be administered in a specified sequence. In RxNorm, the name of a clinical drug combines its ingredients, strengths, and/or form.

While ingredient and strength have straightforward meanings, clarification of what is meant by form may be needed. In RxNorm, the form is the physical form in which the drug is administered or is specified to be administered in a prescription or order. The RxNorm clinical drug name does not refer to the size of the package, the form in which the product was manufactured, its form when it arrived at the dispensary or the intended route.

RxNorm's standard names for clinical drugs and drug delivery devices are connected to the varying names of drugs present in many different controlled vocabularies within the Unified Medical Language System (UMLS) Metathesaurus, including those in commercially available drug information sources. These connections are intended to facilitate interoperability among the computerized systems that record or process data dealing with clinical drugs.

RxNorm is organized around normalized names for clinical drugs and drug delivery devices. These names contain information on ingredients, strengths, and dose forms. In the case of the drug delivery devices, the quantity is also listed. For example:

For generic drug name-
Acetaminophen 500 MG Oral Tablet

For a branded drug name-
Acetaminophen 500 MG Oral Tablet [Tylenol]

For a generic drug pack-
{5 (Aspirin 325 MG Oral Tablet) / 5 (Pravastatin 20 MG Oral Tablet) } Pack

For a branded drug pack-
{30 (Aspirin 325 MG Oral Tablet) / 30 (Pravastatin 20 MG Oral Tablet [Pravachol]) } Pack
[Pravigard 325/20]

Within RxNorm, generic and branded normalized forms are related to each other and to the names of their individual components by a well-defined set of named relationships. Thus, *Acetaminophen 500 MG Oral Tablet* is related to *Acetaminophen 500 MG Oral Tablet [Tylenol]*, and both have relationships to *Acetaminophen*, *Acetaminophen 500 MG*, and *Oral Tablet*. Within the UMLS Metathesaurus, *Acetaminophen 500 MG Oral Tablet* and *Acetaminophen 500 MG Oral Tablet [Tylenol]* will each be linked to different names that are used for these entities in other vocabularies.

RxNorm contains the names of prescription and many nonprescription formulations that exist in the United States, including the devices that administer the medications.

RxNorm is intended to cover all prescription medications approved for human use in the United States. Prescription medications from other countries may be included as opportunities allow, a principal consideration being that there be an authoritative source of information about these drugs. Over-the-counter (OTC) medications will be added and covered, as well, when reliable information

²⁵ <http://www.nlm.nih.gov/research/umls/rxnorm/overview.html>

about the medications can be found. Medications, whether prescription or OTC, with more than four ingredients are not fully represented at the present time.

Additions to the vocabulary will be made as new products are put on the market. Radiopharmaceuticals, because of the decay in strength over time and the requirement that they be ordered and prepared especially for a given time of administration, are listed only as ingredients. Contrast media such as barium compounds are also not represented.

4.5 MESH

The Medical Subject Headings (MeSH®) thesaurus is a controlled vocabulary produced by the National Library of Medicine and used for indexing, cataloging, and searching for biomedical and health-related information and documents²⁶.

2010 MeSH includes the subject descriptors appearing in MEDLINE®/PubMed®, the NLM catalog database, and other NLM databases.

Many synonyms, near-synonyms, and closely related concepts are included as entry terms to help users find the most relevant MeSH descriptor for the concept they are seeking. In NLM's online databases, many terms entered by searchers are automatically mapped to MeSH descriptors to facilitate retrieval of relevant information.

Various online systems provide access to MeSH and the vocabulary is available in several online systems. These include the MeSH Browser, containing the complete contents of the vocabulary; the MeSH Entrez databases, which are designed to assist those searching MEDLINE/PubMed; and the UMLS Metathesaurus® with links to many other controlled vocabularies. MeSH browser is shown in **Figure 10**.

The screenshot shows the MeSH Browser interface. At the top, there are logos for NLM (National Library of Medicine) and MeSH (Medical Subject Headings). Below the logos is a navigation bar with links: MeSH Home, Contact NLM, Site Index, Search Our Web Site, and NLM Home. A secondary navigation bar includes: Health Information, Library Services, Research Programs, New & Noteworthy, and General Information.

The main content area features the text: "MeSH Browser (2010 MeSH): The files are updated every week on Sunday. [Go to 2009 MeSH](#)".

Below this is a search input field with the text "headache". To the right of the input field is a button labeled "Navigate from tree top".

Underneath the search field are two columns of radio button options:

- Search for these record types:**
 - Main Headings
 - Qualifiers
 - Supplementary Concepts
 - All of the Above
 - Search as MeSH Unique ID
 - Search as text words in Annotation & Scope Note
- Search in these fields of chemicals:**
 - Heading Mapped To (HM) (Supplementary List)
 - Indexing Information (II) (Supplementary List)
 - Pharmacological Action (PA)
 - CAS Registry/EC Number (RN)
 - Related CAS Registry Number (RR)

At the bottom of the search area are three buttons: "Find Exact Term", "Find Terms with ALL Fragments", and "Find Terms with ANY Fragment".

Figure 10 Mesh Browser

²⁶ http://www.nlm.nih.gov/mesh/2009/introduction/intro_preface.html

5 COMMON TERMINOLOGY SERVICES (CTS)

The HL7 Common Terminology Services (HL7 CTS) is an Application Programming Interface (API) specification that is intended to describe the basic functionality that will be needed by HL7 Version 3 software implementations to query and access terminological content²⁷. It is specified as an API rather than a set of data structures to enable a wide variety of terminological content to be integrated within the HL7 Version 3 messaging framework without the need for significant migration or rewrite.

Instead of specifying what an external terminology must look like, HL7 has chosen to identify the common functional characteristics that an external terminology must be able to provide. As an example, an HL7 compliant terminology service will need to be able to determine whether a given concept code is valid within the particular resource. Instead of describing a table keyed by the resource identifier and concept code, the CTS specification describes an Application Programming Interface (API) call that takes a resource identifier and concept code as input and returns a true/false value. Each terminology developer is free to implement this API call in whatever way is most appropriate for them.

There are two layers between HL7 Version 3 message processing applications and the target vocabularies. The upper layer, the Message API communicates with in terms of vocabulary domains, realms, coded attributes and other artifacts of the RIM and HL7 messaging model. The lower layer, the Vocabulary API communicates in terms of coding system, concept codes, designations, and other vocabulary related entities.

5.1 CTS MODULES²⁸

5.1.1 The Message and Vocabulary API

There are two distinct layers between HL7 Version 3 message processing applications and the target vocabularies. The upper layer, the **Message API**, communicates with the messaging software, and does so in terms of vocabulary domains, contexts, value sets, coded attributes and other artifacts of the HL7 message model. The lower layer, the **Vocabulary API**, communicates with the terminology service software, and does so in terms of code systems, concept codes, designations, relationships and other terminology specific entities.

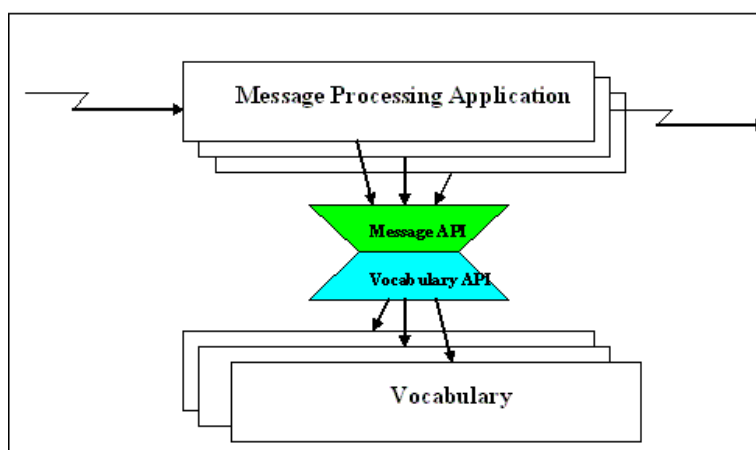


Figure 11 Message and Vocabulary APIs

²⁷ http://wiki.hl7.org/index.php?title=Product_CTS

²⁸

https://wiki.nci.nih.gov/download/attachments/20285848/CTS+2+SFM+DSTU+FINAL+20090322_v1+1_addenda001.doc

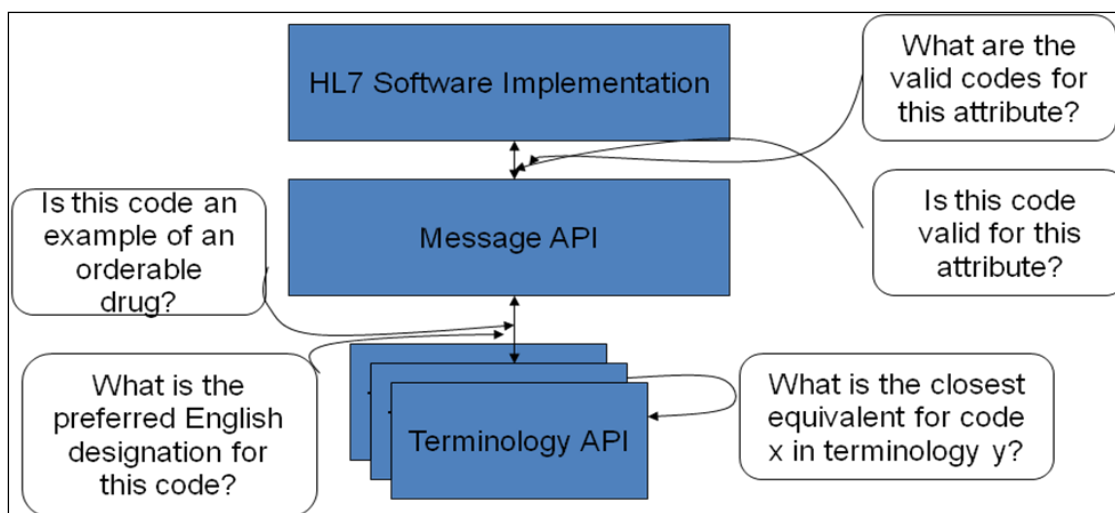


Figure 12 Message API vs. Terminology(Vocabulary) API

The Message API is specific to HL7. Its primary purpose is to allow a wide variety of message processing applications to create, validate and translate CD-derived data types in a consistent and reproducible fashion.

The Vocabulary API is intended to be generic. It allows applications to query different terminologies in a consistent, well-defined fashion. The Message API utilizes the Vocabulary API. The following figure shows an example of a Message/Vocabulary interaction sequence. In this example, an application requests that the Message Runtime Service fill out the details of a CD-derived attribute. The service, in turn, makes multiple calls to the Vocabulary API service in order to get concept code designations, code system names, release versions, etc.

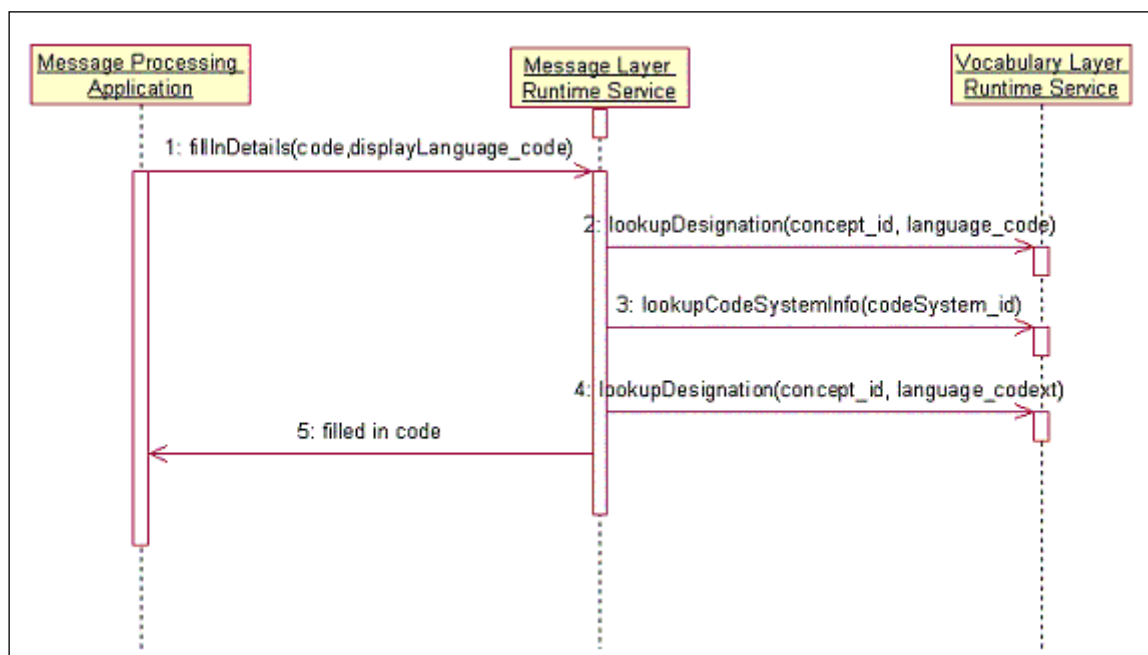


Figure 13 Sample Message/Vocabulary API interaction

The message/vocabulary split can be combined with the two requirements profiles to yield five separate modules:

Table 8 Specification Components

	Runtime	Browser
Message API	Message Runtime	Message Browser
Vocabulary API	Vocabulary Runtime	Vocabulary Browser
Translation (CodeMapping)	Vocabulary Translation	

6 IMPLEMENTATION OF CTS

6.1 PREPARING UMLS KNOWLEDGE SOURCES

While developing Code Mapping API of the iCARDEA Platform, a local version of UMLS Knowledge Sources (Metathesaurus, Semantic Network, and SPECIALIST Lexicon) is used instead of UMLS Knowledge Source Server API. To install these sources, MetamorphoSys tool of UMLS is used. MetamorphoSys guides the users through the installation of one or more UMLS Knowledge Sources. LOINC, SNOMEDCT, HL7 v2.5, HL7 v3.0, RxNORM, ICD10 and MeSH are chosen to be filled in database.

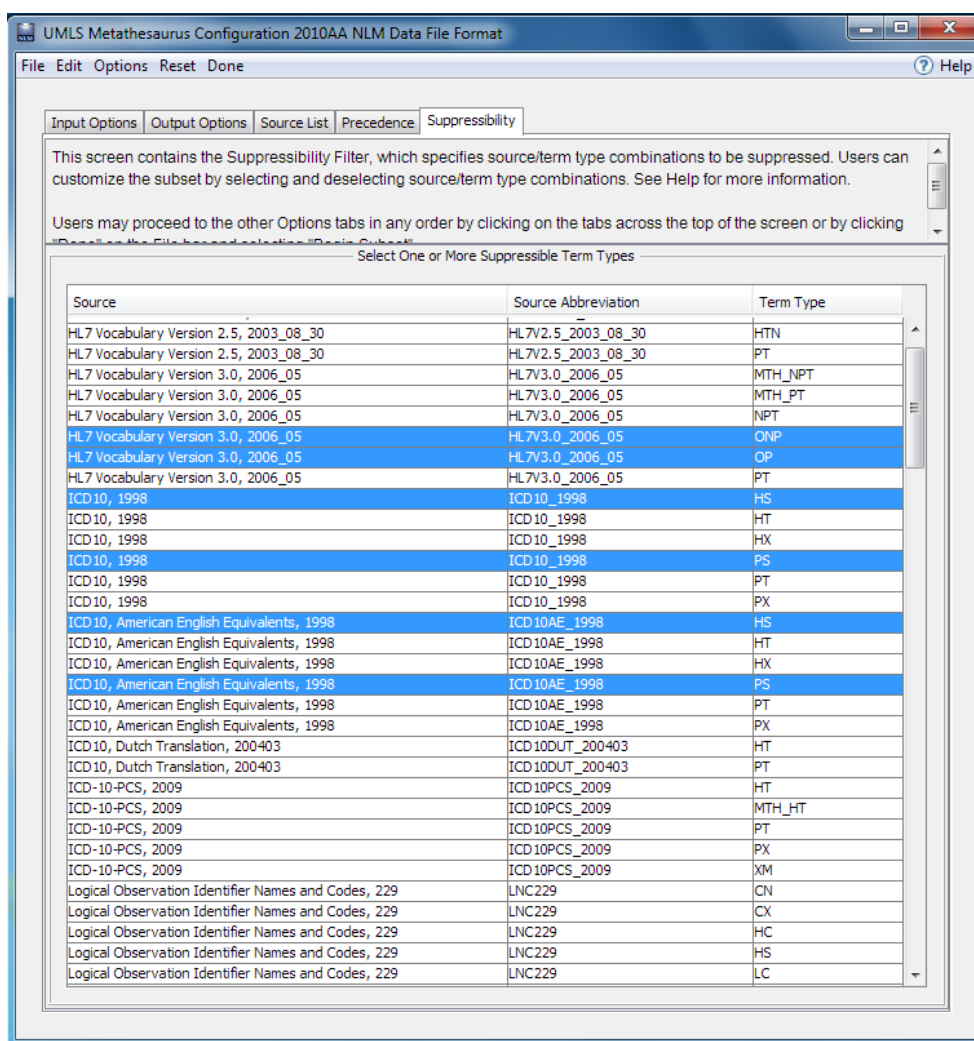


Figure 14Preparing UMLS Knowledge Sources

MetamorphoSys tool is able to export SQL scripts for MySQL, MS Access and Oracle to create database in one of those servers. We chose MySQL server. Beside SQL scripts, it also generates a .bat file to run scripts in server. We created database using this script. Size of the created database, even it has only information of the specified code systems, is larger than 20 GB. Some of the tables have million rows of data. Hence, we need to make some modifications to database configurations to shorten the required time to add indexes. Even with these modifications, adding indexes took more than one week. These indexes are obligatory, since some of the SQL queries takes too long with in tables which has more than 10 million rows.

There are 47 tables in UMLS database. Most important table is the one called 'mrconso'. The entire concept structure appears in this table. If we call the basic building blocks as atoms, this table contains content id, lexical id, atomical id, code system and code for every atom in UMLS. The atoms with the same content id, means that they have the same meaning. This provides mapping between codes of different code systems.

CUI	LUI	SUI	AUI	SAB	CODE	STR
C0000005	L0270109	S0007491	A0016458	MSH	D012711	(131)I-MAA
C0000005	L0000005	S0007492	A7755565	MSH	D012711	(131)I-Macroaggregated Albumin
C0000039	L0000035	S0007560	A0016511	MSH	D015060	1,2-Dihexadecyl-sn-Glycerophosphocholine
C0000039	L0000038	S0007563	A0016514	MSH	D015060	1,2-Dipalmitoyl-Glycerophosphocholine
C0000039	L0000039	S0007564	A0016515	MSH	D015060	1,2-Dipalmitoylphosphatidylcholine
C0000039	L6195414	S0033295	A0049234	MSH	D015060	Dipalmitoyl Phosphatidylcholine
C0000039	L0012508	S0033296	A0049235	MSH	D015060	Dipalmitoylglycerophosphocholine
C0000039	L0012509	S0033297	A0049236	MSH	D015060	Dipalmitoyllecithin
C0000039	L0012507	S0033298	A0049237	MSH	D015060	Dipalmitoylphosphatidylcholine
C0000039	L6195414	S0073244	A0100864	MSH	D015060	Phosphatidylcholine, Dipalmitoyl
C0000039	L0354989	S0464922	A0528280	MSH	D015060	3,5,9-Trioxa-4-phosphapentacosan-1-aminium, 4-hydr...
C0000039	L0012507	S0033298	A11900142	LNC	MTHU010538	Dipalmitoylphosphatidylcholine
C0000039	L6530188	S7593536	A12080359	MSH	D015060	DIPALMITOYLPHOSPHATIDYLCHOLINE 0102
C0000039	L6329835	S7252962	A12091182	MSH	D015060	DPPC
C0000039	L4091165	S4774635	A12180839	SCTSPA	102735002	dipalmitoifosfatidilcolina (sustancia)
C0000039	L4091164	S4774636	A12463599	SCTSPA	102735002	dipalmitoifosfatidilcolina
C0000039	L0000035	S1357276	A1317687	MSH	D015060	1,2 Dihexadecyl sn Glycerophosphocholine
C0000039	L0000038	S1357295	A1317707	MSH	D015060	1,2 Dipalmitoyl Glycerophosphocholine
C0000039	L0000039	S1357296	A1317708	MSH	D015060	1,2 Dipalmitoylphosphatidylcholine
C0000039	L0012507	S0033298	A15625537	LNC	LP15542-1	Dipalmitoylphosphatidylcholine
C0000039	L0012507	S0033298	A8380106	SNOMEDCT	102735002	Dipalmitoylphosphatidylcholine
C0000039	L3000054	S3260062	A8383517	SNOMEDCT	102735002	Dipalmitoylphosphatidylcholine (substance)
C0000052	L0000052	S0007578	A0016529	MSH	D015061	1,4 alpha Glucan Branching Enzyme
C0000052	L0000052	S0007584	A0016535	MSH	D015061	1,4-alpha-Glucan Branching Enzyme
C0000052	L0006129	S0020479	A0032514	MSH	D015061	Branching Enzyme
C0000052	L0038181	S0020482	A0032517	MSH	D015061	Branching Enzyme, Starch
C0000052	L0006130	S0020483	A0032518	MSH	D015061	Branching Glycosyltransferase
C0000052	L0006129	S0038167	A0054980	MSH	D015061	Enzyme, Branching
C0000052	L0038181	S0038183	A0054992	MSH	D015061	Enzyme, Starch Branching
C0000052	L0006130	S0045536	A0064292	MSH	D015061	Glycosyltransferase, Branching

Figure 15 A small part of the 'mrconso' table

CUI, LUI, SUI, AUI are the unique identifiers. SAB is the code system of the term. CODE is the code. STR is the term itself.

There are four Unique Identifiers in UMLS:

- Content UI
- Lexical UI

- String UI
- Atom UI
- All occurrences of a term in any code system has different AUI.
- If two term has the same characters, in other words they are written same, they have same SUI. For example in **Figure 15** Dipalmitoylphosphatidylcholine term is both found in LNC(LOINC) and in SNOMEDCT. Both occurrence of the term has same SUI 'S0033298', however they have different AUI.
- If two terms are lexically equal, in other words, they are written same after normalization, they have same LUI. For example, both 'Branching Glycosyltransferase' and 'Glycosyltransferase, Branching' becomes 'Branching Glycosyltransferase' after normalization. Hence they have same LUI 'L0006130' Since they are not written same without normalization, they have different SUI. They also have different AUI.
- If two terms has the same content or meaning, they have same CUI. The terms with CUI 'C0000039' have all the same meaning, despite many of them are in different code systems. The terms with same CUI and different code systems can be translated into each other.

6.2 USED MODULES OF CTS

As presented in **Table 8**, CTS has 5 different modules with 5 different Web services. CTS provides WSDL for all these modules. Since only three of them are used in Code Mapping API, these three modules will be described.

Four functions of CTS from three different modules are implemented to be used in four use cases which are presented below:

- UC32 - Get a code translated from one code system to another code system
- UC33 - Get supported mappings
- UC34 - Get supported code systems
- UC35 - Validate a given code

1. Message Runtime module describes the functions translateCode (for UC32) and validateCode (for UC35).

RuntimeOperations	
▶	↔ subsumes
▶	↔ getSupportedMatchAlgorithms
▶	↔ getSupportedVocabularyDomains
▶	↔ validateCode
▶	↔ validateTranslation
▶	↔ translateCode
▶	↔ areEquivalent
▶	↔ fillInDetails
▶	↔ lookupValueSetExpansion
▶	↔ expandValueSetExpansionContext
▶	↔ getServiceName
▶	↔ getServiceVersion
▶	↔ getServiceDescription
▶	↔ getCTSVersion
▶	↔ getHL7ReleaseVersion

Figure 16 Message Runtime Functions

All functions of the Message Runtime module are presented in **Figure 16**. Only two of the presented functions are implemented. Other functions are not necessary for the purposes of Code Mapping API. Below, details of the implemented functions are presented in **Table 9**.

Table 9 Message Layer Runtime Functions

Function	Inputs	Outputs	Description
translateCode	- From "Code System" - From "Code" - To "Code System"	-To "Code"	Translates the given code from former code system to latter code system
validateCode	- Code System - Code	- isValid	Returns TRUE if given code system has given code, FALSE otherwise

- Vocabulary Runtime module describes the function `getSupportedCodeSystem` (for UC33).

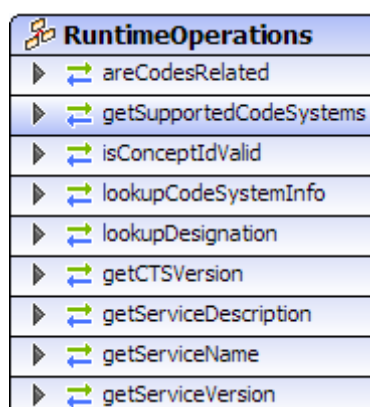


Figure 17 Vocabulary Runtime Functions

All functions of the Vocabulary Runtime module are presented in **Figure 17**. Only one of the presented functions are implemented. Other functions are not necessary for the purposes of Code Mapping API. Below, details of the implemented function are presented in **Table 10**.

Table 10 Vocabulary Layer Runtime Functions

Function	Inputs	Outputs	Description
getSupportedCodeSystem	- Limit (optional, 0 means no limit)	- List of code systems	Returns a list of a limited number of supported code systems

- Code Mapping module describes the function `getSupportedMaps` (for UC34).

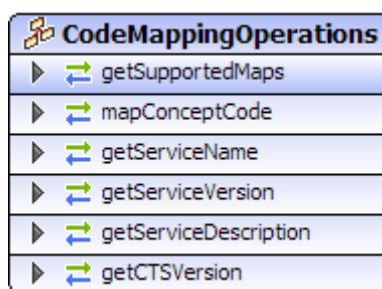


Figure 18 Code Mapping Functions

All functions of the Code Mapping module are presented in **Figure 18**. Only one of the presented functions are implemented. Other functions are not necessary for the purposes of Code Mapping API. Below, details of the implemented function are presented in **Table 11**.

Table 11 Mapping Layer Functions

Function	Inputs	Outputs	Description
getSupportedMaps		- List of mappings	Returns a list of supported mappings as pairs

6.3 IMPLEMENTATION OF CTS WEB SERVICES

At the first step of implementation, Axis2 WSDL2JAVA tool is used to generate data types and skeleton classes from the WSDL files of CTS. Skeleton classes are the server side classes that take input from client side and match the input with related methods. Those skeleton classes are implemented for connecting to database and performing requested operations. Below a part of the skeleton class of Message Runtime module is presented. This part contains the SQL query to translate a message.

```
rs = stat.executeQuery(
"SELECT DISTINCT CODE " +
    "FROM mrconso " +
    "WHERE SAB='"+ toCodeSystem +
        "' AND CUI = (" +
            "SELECT DISTINCT CUI " +
            "FROM mrconso " +
            "WHERE SAB = '"+ fromCodeSystem +"' " +
            "AND CODE = '"+ fromCode +"'");
if(rs.next()){
    toCode = rs.getString("CODE");
}else{
    UnableToTranslate utt = new UnableToTranslate();
    throw utt;
}
```

After filling the necessary parts in skeleton classes, web services are deployed into Axis2, an XML based Web Service framework. Axis2 can be deployed under a Tomcat installation by putting axis2.war file under 'webapps' folder of the Tomcat. To deploy web services into Axis2, we use the ANT build tool and the build.xml file, which is generated while generating the classes with WSDL2JAVA tool, by following the steps below.

- If not installed, install ANT and set ENVIRONMENT VARIABLES related with it.

- Open a command line.
- Go to the directory of the generated build.xml file.
- Run the command: ant jar.server
- Copy the .aar file located under build/lib folder.
- Paste .aar file under the axis2 deployment directory which is located at TOMCAT_HOME/webapps/axis2/WEB_INF/services
- Restart Tomcat

After following these steps for three modules, we have three web services deployed and ready to be used.

6.4 CLIENT APPLICATION

Since different functions are served as different web services, we have developed a client application to call those web services. This application takes several parameters to be used in requested operations. First parameter specifies the function to be called by giving the name of the function. Other parameters after the first one may vary according to requested function.

6.4.1 translateCode

To call “translateCode” function, three more parameters need to be given after the first parameter. First of these three parameters specifies the “code system” of the translation will be made from. Second one specifies the “code” to be translated. Finally, the last one specifies the “code system” of the translation will be made to.

Here are the arguments of an example call to “translateCode” function:

“translateCode ICD10 F79 SNOMEDCT”

This call automatically creates parameters for translateCode function, calls the function and prints the result in a readable format like “154978008”.

‘154978008’ is the SNOMEDCT code for CUI ‘C0025362’ where ‘F79’ is ICD10 code for the same CUI.

Below, the original messages that are sent and received at the background are shown.

Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cts="urn://hl7.org/CTSMAPI" xmlns:typ="urn://cts.hl7.org/types">
  <soapenv:Header/>
  <soapenv:Body>
    <cts:translateCode>
      <cts:parm1>
        <typ:v>ICD10</typ:v>
      </cts:parm1>
      <cts:parm2>
        <typ:code>F79</typ:code>
        <typ:codeSystem></typ:codeSystem>
        <typ:codeSystemName></typ:codeSystemName>
        <typ:codeSystemVersion></typ:codeSystemVersion>
        <typ:displayName></typ:displayName>
        <typ:qualifiers>
        </typ:qualifiers>
        <typ:originalText>
          <typ:_this>
            <typ:itemType></typ:itemType>
            <typ:binaryValue>cid:1052033184941</typ:binaryValue>
            <typ:textualValue></typ:textualValue>
          </typ:_this>
        </typ:originalText>
      </cts:parm2>
    </cts:translateCode>
  </soapenv:Body>
</soapenv:Envelope>
```

```

<typ:encoding></typ:encoding>
<typ:mediaType></typ:mediaType>
<typ:compression></typ:compression>
<typ:integrityCheck>cid:569831531427</typ:integrityCheck>
<typ:reference></typ:reference>
<typ:ingegrityCheckAlgorithm></typ:ingegrityCheckAlgorithm>
<typ:charset></typ:charset>
<typ:language></typ:language>
<typ:thumbnail>
</typ:thumbnail>
</typ:originalText>
<typ:translation>
</typ:translation>
<typ:codingRationale>
<typ:code></typ:code>
<typ:originalText>
  <typ:_this>
    <typ:itemType></typ:itemType>
    <typ:binaryValue>cid:162384140517</typ:binaryValue>
    <typ:textualValue></typ:textualValue>
  </typ:_this>
  <typ:encoding></typ:encoding>
  <typ:mediaType></typ:mediaType>
  <typ:compression></typ:compression>
  <typ:integrityCheck>cid:875803842357</typ:integrityCheck>
  <typ:reference></typ:reference>
  <typ:ingegrityCheckAlgorithm>?</typ:ingegrityCheckAlgorithm>
  <typ:charset></typ:charset>
  <typ:language></typ:language>
  <typ:thumbnail>
  </typ:thumbnail>
</typ:originalText>
<typ:codingRationale></typ:codingRationale>
</typ:codingRationale>
</cts:parm2>
<cts:parm3>
  <typ:v>SNOMEDCT</typ:v>
</cts:parm3>
</cts:translateCode>
</soapenv:Body>
</soapenv:Envelope>

```

Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns2:translateCodeResponse xmlns:ns2="urn://hl7.org/CTSMAPI">
      <ns2:translateCodeReturn>
        <ns1:code xmlns:ns1="urn://cts.hl7.org/types">154978008</ns1:code>
        <ns1:codeSystem xmlns:ns1="urn://cts.hl7.org/types">SNOMEDCT</ns1:codeSystem>
        <ns1:codeSystemName xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>

```

```

    <ns1:codeSystemVersion xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:displayName xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:qualifiers xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:originalText xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:translation xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:codingRationale xsi:nil="1" xmlns:ns1="urn://cts.hl7.org/types"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  </ns2:translateCodeReturn>
</ns2:translateCodeResponse>
</soapenv:Body>
</soapenv:Envelope>

```

6.4.2 validateCode

To call “validateCode” function, two more parameters need to be given after the first parameter. First of these two parameters specifies the “code system” of the validation that will be made in. Second one specifies the “code” to be validated.

Here are the arguments of an example call to “validateCode” function:
“validateCode MHS D015060”

This call automatically creates parameters for validateCode function, calls the function and prints the result in a readable format like “Valid” or “Invalid”.

‘F79’ is a valid code in ICD10, and it refers the term “Unspecified mental retardation”.

Below, the original messages that are sent and received at the background are shown. Example request given following returns with an error, since we give input code system as ‘MHS’ which should be ‘MSH’. Furthermore, response also contains an explanation in errorText field which is indicated as bold.

Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cts="urn://hl7.org/CTSMAPI" xmlns:typ="urn://cts.hl7.org/types">
  <soapenv:Header/>
  <soapenv:Body>
    <cts:validateCode>
      <cts:vocabularyDomain_name>
        <typ:v>MHS</typ:v>
      </cts:vocabularyDomain_name>
      <cts:codeToValidate>
        <typ:code>D015060</typ:code>
        <typ:codeSystem>?</typ:codeSystem>
        <typ:codeSystemName>?</typ:codeSystemName>
        <typ:codeSystemVersion>?</typ:codeSystemVersion>
        <typ:displayName>?</typ:displayName>
        <typ:qualifiers>
        </typ:qualifiers>
        <typ:originalText>
          <typ:_this>
            <typ:itemType>?</typ:itemType>

```

```

    <typ:binaryValue>cid:11135445159</typ:binaryValue>
    <typ:textualValue>?</typ:textualValue>
  </typ:_this>
<typ:encoding>?</typ:encoding>
<typ:mediaType>?</typ:mediaType>
<typ:compression>?</typ:compression>
<typ:integrityCheck>cid:576917804409</typ:integrityCheck>
<typ:reference>?</typ:reference>
<typ:ingegrityCheckAlgorithm>?</typ:ingegrityCheckAlgorithm>
<typ:charset>?</typ:charset>
<typ:language>?</typ:language>
<typ:thumbnail>
  </typ:thumbnail>
</typ:originalText>
<typ:translation>
</typ:translation>
<typ:codingRationale>
  <typ:code>?</typ:code>
  <typ:originalText>
    <typ:_this>
      <typ:itemType>?</typ:itemType>
      <typ:binaryValue>cid:84961271975</typ:binaryValue>
      <typ:textualValue>?</typ:textualValue>
    </typ:_this>
    <typ:encoding>?</typ:encoding>
    <typ:mediaType>?</typ:mediaType>
    <typ:compression>?</typ:compression>
    <typ:integrityCheck>cid:349798733963</typ:integrityCheck>
    <typ:reference>?</typ:reference>
    <typ:ingegrityCheckAlgorithm>?</typ:ingegrityCheckAlgorithm>
    <typ:charset>?</typ:charset>
    <typ:language>?</typ:language>
    <typ:thumbnail>
      </typ:thumbnail>
    </typ:originalText>
    <typ:codingRationale>?</typ:codingRationale>
  </typ:codingRationale>
</cts:codeToValidate>
<cts:activeConceptsOnly>
  <typ:v>>true</typ:v>
</cts:activeConceptsOnly>
<cts:errorCheckOnly>
  <typ:v>>true</typ:v>
</cts:errorCheckOnly>
</cts:validateCode>
</soapenv:Body>
</soapenv:Envelope>

```

Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns2:validateCodeResponse xmlns:ns2="urn://hl7.org/CTSMAPI">

```

```

<ns2:validateCodeReturn>
  <ns2:nErrors>
    <ns1:v xmlns:ns1="urn://cts.hl7.org/types">1</ns1:v>
  </ns2:nErrors>
  <ns2:nWarnings xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns2:detail>
    <ns2:item>
      <ns2:codeInError xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      <ns2:isError>
        <ns1:v xmlns:ns1="urn://cts.hl7.org/types">true</ns1:v>
      </ns2:isError>
      <ns2:error_id xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      <ns2:errorText>
        <ns1:v xmlns:ns1="urn://cts.hl7.org/types">Given code is not valid in given code system.</ns1:v>
      </ns2:errorText>
    </ns2:item>
  </ns2:detail>
</ns2:validateCodeReturn>
</ns2:validateCodeResponse>
</soapenv:Body>
</soapenv:Envelope>

```

6.4.3 getSupportedCodeSystems

To call “getSupportedCodeSystems” function, one more parameter may be given after the first parameter. This parameter specifies the limit of the number of the results. If this parameter is not given or given ‘0’, function returns all the results.

Here are the arguments of an example call to “getSupportedCodeSystems” function:
“getSupportedCodeSystems 11”

This call automatically creates parameters for getSupportedCodeSystems function, calls the function and prints the result in a readable format like the following:

```

ICD10
ICD10AE
DMDICD10
ICD10DUT
HL7V2.5
MTHHL7V2.5
HL7V3.0
ICD10PCS
SCTSPA
SNOMEDCT
LNC

```

Below, the original messages that are sent and received at the background are shown.

Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cts="urn://hl7.org/CTSVAPI">
  <soapenv:Header/>

```

```

<soapenv:Body>
  <cts:getSupportedCodeSystems>
    <cts:timeout>0</cts:timeout>
    <cts:sizeLimit>11</cts:sizeLimit>
  </cts:getSupportedCodeSystems>
</soapenv:Body>
</soapenv:Envelope>

```

Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:getSupportedCodeSystemsResponse xmlns:ns1="urn://hl7.org/CTSVAPI">
      <ns1:getSupportedCodeSystemsReturn>
        <ns1:codeSystem_id>ICD10</ns1:codeSystem_id>
        <ns1:codeSystem_name>ICD10, 1998</ns1:codeSystem_name>
        <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:codeSystem_versions>
          <ns1:item>1998</ns1:item>
        </ns1:codeSystem_versions>
      </ns1:getSupportedCodeSystemsReturn>
      <ns1:getSupportedCodeSystemsReturn>
        <ns1:codeSystem_id>ICD10AE</ns1:codeSystem_id>
        <ns1:codeSystem_name>ICD10, American English Equivalents,
1998</ns1:codeSystem_name>
        <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:codeSystem_versions>
          <ns1:item>1998</ns1:item>
        </ns1:codeSystem_versions>
      </ns1:getSupportedCodeSystemsReturn>
      <ns1:getSupportedCodeSystemsReturn>
        <ns1:codeSystem_id>DMDICD10</ns1:codeSystem_id>
        <ns1:codeSystem_name>German translation of ICD10, 1995</ns1:codeSystem_name>
        <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:codeSystem_versions>
          <ns1:item>1995</ns1:item>
        </ns1:codeSystem_versions>
      </ns1:getSupportedCodeSystemsReturn>
      <ns1:getSupportedCodeSystemsReturn>
        <ns1:codeSystem_id>ICD10DUT</ns1:codeSystem_id>
        <ns1:codeSystem_name>ICD10, Dutch Translation, 200403</ns1:codeSystem_name>
        <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:codeSystem_versions>
          <ns1:item>200403</ns1:item>
        </ns1:codeSystem_versions>
      </ns1:getSupportedCodeSystemsReturn>
      <ns1:getSupportedCodeSystemsReturn>
        <ns1:codeSystem_id>HL7V2.5</ns1:codeSystem_id>
        <ns1:codeSystem_name>HL7 Vocabulary Version 2.5, 2003_08_30</ns1:codeSystem_name>
        <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:codeSystem_versions>
          <ns1:item>2003_08_30</ns1:item>
        </ns1:codeSystem_versions>
      </ns1:getSupportedCodeSystemsReturn>
    </ns1:getSupportedCodeSystemsResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

```

</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>MTHHL7V2.5</ns1:codeSystem_id>
  <ns1:codeSystem_name>HL7 Vocabulary Version 2.5, 7-bit equivalents,
2003_08</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>2003_08</ns1:item>
  </ns1:codeSystem_versions>
</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>HL7V3.0</ns1:codeSystem_id>
  <ns1:codeSystem_name>HL7 Vocabulary Version 3.0, 2006_05</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>2006_05</ns1:item>
  </ns1:codeSystem_versions>
</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>ICD10PCS</ns1:codeSystem_id>
  <ns1:codeSystem_name>ICD-10-PCS, 2009</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>2009</ns1:item>
  </ns1:codeSystem_versions>
</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>SCTSPA</ns1:codeSystem_id>
  <ns1:codeSystem_name>SNOMED Terminos Clinicos (SNOMED CT), Edicion en Espanol,
Distribucion Internacional, Octubre de 2009, 2009_10_31</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>2009_10_31</ns1:item>
  </ns1:codeSystem_versions>
</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>SNOMEDCT</ns1:codeSystem_id>
  <ns1:codeSystem_name>SNOMED Clinical Terms, 2010_01_31</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>2010_01_31</ns1:item>
  </ns1:codeSystem_versions>
</ns1:getSupportedCodeSystemsReturn>
<ns1:getSupportedCodeSystemsReturn>
  <ns1:codeSystem_id>LNC</ns1:codeSystem_id>
  <ns1:codeSystem_name>Logical Observation Identifier Names and Codes,
229</ns1:codeSystem_name>
  <ns1:copyright xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
  <ns1:codeSystem_versions>
    <ns1:item>229</ns1:item>
  </ns1:codeSystem_versions>

```

```

    </ns1:getSupportedCodeSystemsReturn>
  </ns1:getSupportedCodeSystemsResponse>
</soapenv:Body>
</soapenv:Envelope>

```

6.4.4 getSupportedMaps

“getSupportedMaps” function does not use any parameters. To call this function, only its name needs to be given as only parameter.

This automatically calls the function and prints the result in a readable format like the following:

```

DMDICD10 -> HL7V2.5
DMDICD10 -> HL7V3.0
DMDICD10 -> ICD10
DMDICD10 -> ICD10AE
DMDICD10 -> ICD10DUT
...
...
...

```

Below, the original messages that are sent and received at the background are shown.

Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cts="urn://hl7.org/CTSVAPI">
  <soapenv:Header/>
  <soapenv:Body>
    <cts:getSupportedMaps/>
  </soapenv:Body>
</soapenv:Envelope>

```

Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:getSupportedMapsResponse xmlns:ns1="urn://hl7.org/CTSVAPI">
      <ns1:getSupportedMapsReturn>
        <ns1:map_name xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
        <ns1:mapDescription xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
        <ns1:fromCodeSystem_id>DMDICD10</ns1:fromCodeSystem_id>
        <ns1:fromCodeSystem_name>German translation of ICD10,
1995</ns1:fromCodeSystem_name>
        <ns1:fromCodeSystem_version>1995</ns1:fromCodeSystem_version>
        <ns1:toCodeSystem_id>HL7V2.5</ns1:toCodeSystem_id>
        <ns1:toCodeSystem_name>HL7 Vocabulary Version 2.5,
2003_08_30</ns1:toCodeSystem_name>
        <ns1:toCodeSystem_version>2003_08_30</ns1:toCodeSystem_version>
      </ns1:getSupportedMapsReturn>
    </ns1:getSupportedMapsResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

```

    <ns1:fromCodeSystem_id>DMDICD10</ns1:fromCodeSystem_id>
    <ns1:fromCodeSystem_name>German translation of ICD10,
1995</ns1:fromCodeSystem_name>
    <ns1:fromCodeSystem_version>1995</ns1:fromCodeSystem_version>
    <ns1:toCodeSystem_id>HL7V3.0</ns1:toCodeSystem_id>
    <ns1:toCodeSystem_name>HL7 Vocabulary Version 3.0, 2006_05</ns1:toCodeSystem_name>
    <ns1:toCodeSystem_version>2006_05</ns1:toCodeSystem_version>
  </ns1:getSupportedMapsReturn>
  <ns1:getSupportedMapsReturn>
    <ns1:map_name xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:mapDescription xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
    <ns1:fromCodeSystem_id>DMDICD10</ns1:fromCodeSystem_id>
    <ns1:fromCodeSystem_name>German translation of ICD10,
1995</ns1:fromCodeSystem_name>
    <ns1:fromCodeSystem_version>1995</ns1:fromCodeSystem_version>
    <ns1:toCodeSystem_id>ICD10</ns1:toCodeSystem_id>
    <ns1:toCodeSystem_name>ICD10, 1998</ns1:toCodeSystem_name>
    <ns1:toCodeSystem_version>1998</ns1:toCodeSystem_version>
  </ns1:getSupportedMapsReturn>
  <ns1:getSupportedMapsReturn>
    <ns1:map_name xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:mapDescription xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
    <ns1:fromCodeSystem_id>DMDICD10</ns1:fromCodeSystem_id>
    <ns1:fromCodeSystem_name>German translation of ICD10,
1995</ns1:fromCodeSystem_name>
    <ns1:fromCodeSystem_version>1995</ns1:fromCodeSystem_version>
    <ns1:toCodeSystem_id>ICD10AE</ns1:toCodeSystem_id>
    <ns1:toCodeSystem_name>ICD10, American English Equivalents,
1998</ns1:toCodeSystem_name>
    <ns1:toCodeSystem_version>1998</ns1:toCodeSystem_version>
  </ns1:getSupportedMapsReturn>
  <ns1:getSupportedMapsReturn>
    <ns1:map_name xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <ns1:mapDescription xsi:nil="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
    <ns1:fromCodeSystem_id>DMDICD10</ns1:fromCodeSystem_id>
    <ns1:fromCodeSystem_name>German translation of ICD10,
1995</ns1:fromCodeSystem_name>
    <ns1:fromCodeSystem_version>1995</ns1:fromCodeSystem_version>
    <ns1:toCodeSystem_id>ICD10DUT</ns1:toCodeSystem_id>
    <ns1:toCodeSystem_name>ICD10, Dutch Translation, 200403</ns1:toCodeSystem_name>
    <ns1:toCodeSystem_version>200403</ns1:toCodeSystem_version>
  </ns1:getSupportedMapsReturn>
  ...
  ...
  ...
</ns1:getSupportedMapsResponse>
</soapenv:Body>
</soapenv:Envelope>

```

7 CONCLUSION

By providing EHR interoperability, information about patients' medical history such as history of non-cardiac conditions; more detailed information about severity of each condition (e.g., record of prior hospitalizations or specifics of therapy for the condition); the medications being taken at the time of spontaneous arrhythmia occurrence or the non-cardiac conditions denoting contraindications to the proposed therapies can be obtained from the patient EHR data and used in the clinical workflow.

However, providing EHR interoperability has several challenges such as the legacy EHR system and interoperability of the code systems. To address these challenges, iCARDEA Project provides an easy to use API namely, Code Mapping API. Code Mapping API is developed and served to solve the interoperability of the code system used (semantic interoperability) to provide EHR interoperability. With this API, mapping of any code in any code system such as ICD10, SNOMEDCT, LOINC, RxNORM, etc. can be achieved. Moreover, this API also enables validating any code in a given code system. To be able to map different code systems, HL7 Common Terminology Services is used.

8 APPENDIX A – WSDL's of CTS MODULES

8.1 MESSAGE RUNTIME

```
<wsdl:definitions targetNamespace="urn://hl7.org/CTSMAPI" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="urn://hl7.org/CTSMAPI" xmlns:intf="urn://hl7.org/CTSMAPI" xmlns:tns3="urn://cts.hl7.org/types"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/" xmlns:wSDLsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema elementFormDefault="qualified" targetNamespace="urn://hl7.org/CTSMAPI"
xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="urn://cts.hl7.org/types"/>
      <element name="subsumes">
        <complexType>
          <sequence>
            <element name="parentCode" type="tns3:CD"/>
            <element name="childCode" type="tns3:CD"/>
          </sequence>
        </complexType>
      </element>
      <complexType name="ArrayOf_tns3_ED">
        <sequence>
          <element maxOccurs="unbounded" minOccurs="0" name="item" type="tns3:ED"/>
        </sequence>
      </complexType>
      <complexType name="ArrayOf_tns3_CR">
        <sequence>
          <element maxOccurs="unbounded" minOccurs="0" name="item" type="tns3:CR"/>
        </sequence>
      </complexType>
      <complexType name="ArrayOf_tns3_CD">
        <sequence>
          <element maxOccurs="unbounded" minOccurs="0" name="item" type="tns3:CD"/>
        </sequence>
      </complexType>
      <element name="subsumesResponse">
        <complexType>
          <sequence>
            <element name="subsumesReturn" type="tns3:BL"/>
          </sequence>
        </complexType>
      </element>
      <complexType name="SubsumptionNotSupported">
        <sequence>
          <element name="codeSystem_id" nillable="true" type="tns3:UID"/>
        </sequence>
      </complexType>
    </schema>
  </wsdl:types>

```

```

</sequence>
</complexType>
<element name="fault" type="impl:SubsumptionNotSupported"/>
<complexType name="QualifiersNotSupported">
  <sequence/>
</complexType>
<element name="fault1" type="impl:QualifiersNotSupported"/>
<complexType name="ConceptId">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="tns3:UID"/>
    <element name="concept_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<complexType name="UnknownConceptCode">
  <sequence>
    <element name="concept_id" nillable="true" type="impl:ConceptId"/>
  </sequence>
</complexType>
<element name="fault2" type="impl:UnknownConceptCode"/>
<complexType name="UnknownCodeSystem">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="tns3:UID"/>
  </sequence>
</complexType>
<element name="fault3" type="impl:UnknownCodeSystem"/>
<complexType name="UnrecognizedQualifier">
  <sequence>
    <element name="qualifier" nillable="true" type="tns3:CR"/>
  </sequence>
</complexType>
<element name="fault4" type="impl:UnrecognizedQualifier"/>
<complexType name="UnexpectedError">
  <sequence>
    <element name="possible_cause" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault5" type="impl:UnexpectedError"/>
<element name="getSupportedMatchAlgorithms">
  <complexType/>
</element>
<element name="getSupportedMatchAlgorithmsResponse">
  <complexType>
    <sequence>
      <element maxOccurs="unbounded" name="getSupportedMatchAlgorithmsReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<element name="getSupportedVocabularyDomains">
  <complexType>
    <sequence>
      <element name="matchText" type="tns3:ST"/>
      <element name="matchAlgorithm_code" type="tns3:ST"/>
      <element name="timeout" type="xsd:int"/>
      <element name="sizeLimit" type="xsd:int"/>
    </sequence>
  </complexType>
</element>
<element name="getSupportedVocabularyDomainsResponse">
  <complexType>
    <sequence>
      <element maxOccurs="unbounded" name="getSupportedVocabularyDomainsReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<complexType name="TimeoutError">
  <sequence/>
</complexType>

```

```

<element name="fault6" type="impl:TimeoutError"/>
<complexType name="UnknownMatchAlgorithm">
  <sequence>
    <element name="matchAlgorithm_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault7" type="impl:UnknownMatchAlgorithm"/>
<complexType name="BadlyFormedMatchText">
  <sequence>
    <element name="matchText" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault8" type="impl:BadlyFormedMatchText"/>
<element name="validateCode">
  <complexType>
    <sequence>
      <element name="vocabularyDomain_name" type="tns3:ST"/>
      <element name="codeToValidate" type="tns3:CD"/>
      <element name="applicationContext_code" type="tns3:ST"/>
      <element name="activeConceptsOnly" type="tns3:BL"/>
      <element name="errorCheckOnly" type="tns3:BL"/>
    </sequence>
  </complexType>
</element>
<element name="validateCodeResponse">
  <complexType>
    <sequence>
      <element name="validateCodeReturn" type="impl:ValidateCodeReturn"/>
    </sequence>
  </complexType>
</element>
<complexType name="ValidationDetail">
  <sequence>
    <element name="codeInError" nillable="true" type="tns3:CD"/>
    <element name="isError" nillable="true" type="tns3:BL"/>
    <element name="error_id" nillable="true" type="tns3:ST"/>
    <element name="errorText" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<complexType name="ArrayOfValidationDetail">
  <sequence>
    <element maxOccurs="unbounded" minOccurs="0" name="item" type="impl:ValidationDetail"/>
  </sequence>
</complexType>
<complexType name="ValidateCodeReturn">
  <sequence>
    <element name="nErrors" nillable="true" type="tns3:INT"/>
    <element name="nWarnings" nillable="true" type="tns3:INT"/>
    <element name="detail" nillable="true" type="impl:ArrayOfValidationDetail"/>
  </sequence>
</complexType>
<complexType name="UnknownApplicationContextCode">
  <sequence>
    <element name="applicationContext_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault9" type="impl:UnknownApplicationContextCode"/>
<complexType name="UnknownVocabularyDomain">
  <sequence>
    <element name="vocabularyDomain_name" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault10" type="impl:UnknownVocabularyDomain"/>
<complexType name="NoApplicableValueSet">
  <sequence>
    <element name="vocabularyDomain_name" nillable="true" type="tns3:ST"/>
    <element name="applicationContext_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>

```

```

</sequence>
</complexType>
<element name="fault11" type="impl:NoApplicableValueSet"/>
<element name="validateTranslation">
  <complexType>
    <sequence>
      <element name="parm1" type="tns3:ST"/>
      <element name="parm2" type="tns3:CD"/>
      <element name="parm3" type="tns3:ST"/>
      <element name="parm4" type="tns3:BL"/>
      <element name="parm5" type="tns3:BL"/>
    </sequence>
  </complexType>
</element>
<element name="validateTranslationResponse">
  <complexType>
    <sequence>
      <element name="validateTranslationReturn" type="impl:ValidateCodeReturn"/>
    </sequence>
  </complexType>
</element>
<element name="translateCode">
  <complexType>
    <sequence>
      <element name="parm1" type="tns3:ST"/>
      <element name="parm2" type="tns3:CD"/>
      <element name="parm3" type="tns3:ST"/>
      <element name="param5" type="tns3:UID" minOccurs="0"/>
    </sequence>
  </complexType>
</element>
<element name="translateCodeResponse">
  <complexType>
    <sequence>
      <element name="translateCodeReturn" type="tns3:CD"/>
    </sequence>
  </complexType>
</element>
<complexType name="UnableToTranslate">
  <sequence/>
</complexType>
<element name="fault12" type="impl:UnableToTranslate"/>
<element name="areEquivalent">
  <complexType>
    <sequence>
      <element name="parm1" type="tns3:CD"/>
      <element name="parm2" type="tns3:CD"/>
    </sequence>
  </complexType>
</element>
<element name="areEquivalentResponse">
  <complexType>
    <sequence>
      <element name="areEquivalentReturn" type="tns3:BL"/>
    </sequence>
  </complexType>
</element>
<element name="fillInDetails">
  <complexType>
    <sequence>
      <element name="codeToFillIn" type="tns3:CD"/>
      <element name="displayLanguage_code" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<element name="fillInDetailsResponse">
  <complexType>

```

```

    <sequence>
      <element name="fillInDetailsReturn" type="tns3:CD"/>
    </sequence>
  </complexType>
</element>
<complexType name="UnknownLanguage">
  <sequence>
    <element name="language_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault13" type="impl:UnknownLanguage"/>
<complexType name="NoApplicableDesignationFound">
  <sequence>
    <element name="codeToFillIn" nillable="true" type="tns3:CD"/>
    <element name="displayLanguage_code" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<element name="fault14" type="impl:NoApplicableDesignationFound"/>
<element name="lookupValueSetExpansion">
  <complexType>
    <sequence>
      <element name="vocabularyDomain_name" type="tns3:ST"/>
      <element name="applicationContext_code" type="tns3:ST"/>
      <element name="language_code" type="tns3:ST"/>
      <element name="expandAll" type="tns3:BL"/>
      <element name="timeout" type="xsd:int"/>
      <element name="sizeLimit" type="xsd:int"/>
    </sequence>
  </complexType>
</element>
<element name="lookupValueSetExpansionResponse">
  <complexType>
    <sequence>
      <element maxOccurs="unbounded" name="lookupValueSetExpansionReturn"
type="impl:ValueSetExpansion"/>
    </sequence>
  </complexType>
</element>
<complexType name="ValueSetDescriptor">
  <sequence>
    <element name="valueSet_id" nillable="true" type="tns3:UID"/>
    <element name="valueSet_name" nillable="true" type="tns3:ST"/>
  </sequence>
</complexType>
<complexType name="ValueSetExpansion">
  <sequence>
    <element name="pathLength" nillable="true" type="tns3:INT"/>
    <element name="nodeType_code" nillable="true" type="tns3:ST"/>
    <element name="valueSet" nillable="true" type="impl:ValueSetDescriptor"/>
    <element name="concept_id" nillable="true" type="impl:ConceptId"/>
    <element name="displayName" nillable="true" type="tns3:ST"/>
    <element name="isExpandable" nillable="true" type="tns3:BL"/>
    <element name="expansionContext" nillable="true" type="xsd:base64Binary"/>
  </sequence>
</complexType>
<element name="expandValueSetExpansionContext">
  <complexType>
    <sequence>
      <element name="expansionContext" type="xsd:base64Binary"/>
    </sequence>
  </complexType>
</element>
<element name="expandValueSetExpansionContextResponse">
  <complexType>
    <sequence>
      <element maxOccurs="unbounded" name="expandValueSetExpansionContextReturn"
type="impl:ValueSetExpansion"/>
    </sequence>
  </complexType>
</element>

```

```

    </sequence>
  </complexType>
</element>
<complexType name="InvalidExpansionContext">
  <sequence/>
</complexType>
<element name="fault15" type="impl:InvalidExpansionContext"/>
<element name="getServiceName">
  <complexType/>
</element>
<element name="getServiceNameResponse">
  <complexType>
    <sequence>
      <element name="getServiceNameReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<element name="getServiceVersion">
  <complexType/>
</element>
<element name="getServiceVersionResponse">
  <complexType>
    <sequence>
      <element name="getServiceVersionReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<element name="getServiceDescription">
  <complexType/>
</element>
<element name="getServiceDescriptionResponse">
  <complexType>
    <sequence>
      <element name="getServiceDescriptionReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
<element name="getCTSVersion">
  <complexType/>
</element>
<element name="getCTSVersionResponse">
  <complexType>
    <sequence>
      <element name="getCTSVersionReturn" type="impl:CTSVersionId"/>
    </sequence>
  </complexType>
</element>
<complexType name="CTSVersionId">
  <sequence>
    <element name="major" nillable="true" type="tns3:INT"/>
    <element name="minor" nillable="true" type="tns3:INT"/>
  </sequence>
</complexType>
<element name="getHL7ReleaseVersion">
  <complexType/>
</element>
<element name="getHL7ReleaseVersionResponse">
  <complexType>
    <sequence>
      <element name="getHL7ReleaseVersionReturn" type="tns3:ST"/>
    </sequence>
  </complexType>
</element>
</schema>
<schema elementFormDefault="qualified" targetNamespace="urn://cts.hl7.org/types"
xmlns="http://www.w3.org/2001/XMLSchema">
  <import namespace="urn://hl7.org/CTSMAPI"/>

```

```

<complexType name="binary_or_text">
  <sequence>
    <element name="itemType" nillable="true" type="xsd:anyType"/>
    <element name="binaryValue" nillable="true" type="xsd:base64Binary"/>
    <element name="textualValue" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="ED">
  <sequence>
    <element name="_this" nillable="true" type="tns3:binary_or_text"/>
    <element name="encoding" nillable="true" type="xsd:string"/>
    <element name="mediaType" nillable="true" type="xsd:string"/>
    <element name="compression" nillable="true" type="xsd:string"/>
    <element name="integrityCheck" nillable="true" type="xsd:base64Binary"/>
    <element name="reference" nillable="true" type="xsd:string"/>
    <element name="ingegrityCheckAlgorithm" nillable="true" type="xsd:string"/>
    <element name="charset" nillable="true" type="xsd:string"/>
    <element name="language" nillable="true" type="xsd:string"/>
    <element name="thumbnail" nillable="true" type="impl:ArrayOf_tns3_ED"/>
  </sequence>
</complexType>
<complexType name="CS">
  <sequence>
    <element name="code" nillable="true" type="xsd:string"/>
    <element name="originalText" nillable="true" type="tns3:ED"/>
    <element name="codingRationale" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="CV">
  <sequence>
    <element name="code" nillable="true" type="xsd:string"/>
    <element name="codeSystem" nillable="true" type="xsd:string"/>
    <element name="codeSystemName" nillable="true" type="xsd:string"/>
    <element name="codeSystemVersion" nillable="true" type="xsd:string"/>
    <element name="displayName" nillable="true" type="xsd:string"/>
    <element name="originalText" nillable="true" type="tns3:ED"/>
    <element name="codingRationale" nillable="true" type="tns3:CS"/>
  </sequence>
</complexType>
<complexType name="BL">
  <sequence>
    <element name="v" type="xsd:boolean"/>
  </sequence>
</complexType>
<complexType name="CR">
  <sequence>
    <element name="name" nillable="true" type="tns3:CV"/>
    <element name="value" nillable="true" type="tns3:CV"/>
    <element name="inverted" nillable="true" type="tns3:BL"/>
  </sequence>
</complexType>
<complexType name="CD">
  <sequence>
    <element name="code" nillable="true" type="xsd:string"/>
    <element name="codeSystem" nillable="true" type="xsd:string"/>
    <element name="codeSystemName" nillable="true" type="xsd:string"/>
    <element name="codeSystemVersion" nillable="true" type="xsd:string"/>
    <element name="displayName" nillable="true" type="xsd:string"/>
    <element name="qualifiers" nillable="true" type="impl:ArrayOf_tns3_CR"/>
    <element name="originalText" nillable="true" type="tns3:ED"/>
    <element name="translation" nillable="true" type="impl:ArrayOf_tns3_CD"/>
    <element name="codingRationale" nillable="true" type="tns3:CS"/>
  </sequence>
</complexType>
<complexType name="UID">
  <sequence>
    <element name="v" nillable="true" type="xsd:string"/>
  </sequence>

```

```

    </sequence>
  </complexType>
  <complexType name="ST">
    <sequence>
      <element name="v" nillable="true" type="xsd:string"/>
    </sequence>
  </complexType>
  <complexType name="INT">
    <sequence>
      <element name="v" type="xsd:int"/>
    </sequence>
  </complexType>
</schema>
</wsdl:types>
<wsdl:message name="lookupValueSetExpansionResponse">
  <wsdl:part name="parameters" element="impl:lookupValueSetExpansionResponse"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionRequest">
  <wsdl:part name="parameters" element="impl:getServiceDescription"/>
</wsdl:message>
<wsdl:message name="getSupportedVocabularyDomainsResponse">
  <wsdl:part name="parameters" element="impl:getSupportedVocabularyDomainsResponse"/>
</wsdl:message>
<wsdl:message name="validateCodeResponse">
  <wsdl:part name="parameters" element="impl:validateCodeResponse"/>
</wsdl:message>
<wsdl:message name="translateCodeResponse">
  <wsdl:part name="parameters" element="impl:translateCodeResponse"/>
</wsdl:message>
<wsdl:message name="UnknownConceptCode">
  <wsdl:part name="fault" element="impl:fault2"/>
</wsdl:message>
<wsdl:message name="expandValueSetExpansionContextRequest">
  <wsdl:part name="parameters" element="impl:expandValueSetExpansionContext"/>
</wsdl:message>
<wsdl:message name="BadlyFormedMatchText">
  <wsdl:part name="fault" element="impl:fault8"/>
</wsdl:message>
<wsdl:message name="getCTSVersionRequest">
  <wsdl:part name="parameters" element="impl:getCTSVersion"/>
</wsdl:message>
<wsdl:message name="UnknownLanguage">
  <wsdl:part name="fault" element="impl:fault13"/>
</wsdl:message>
<wsdl:message name="validateTranslationRequest">
  <wsdl:part name="parameters" element="impl:validateTranslation"/>
</wsdl:message>
<wsdl:message name="NoApplicableDesignationFound">
  <wsdl:part name="fault" element="impl:fault14"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionResponse">
  <wsdl:part name="parameters" element="impl:getServiceDescriptionResponse"/>
</wsdl:message>
<wsdl:message name="areEquivalentRequest">
  <wsdl:part name="parameters" element="impl:areEquivalent"/>
</wsdl:message>
<wsdl:message name="getSupportedVocabularyDomainsRequest">
  <wsdl:part name="parameters" element="impl:getSupportedVocabularyDomains"/>
</wsdl:message>
<wsdl:message name="UnableToTranslate">
  <wsdl:part name="fault" element="impl:fault12"/>
</wsdl:message>
<wsdl:message name="UnknownCodeSystem">
  <wsdl:part name="fault" element="impl:fault3"/>
</wsdl:message>
<wsdl:message name="QualifiersNotSupported">
  <wsdl:part name="fault" element="impl:fault1"/>

```

```

</wsdl:message>
<wsdl:message name="getHL7ReleaseVersionResponse">
  <wsdl:part name="parameters" element="impl:getHL7ReleaseVersionResponse"/>
</wsdl:message>
<wsdl:message name="UnknownApplicationContextCode">
  <wsdl:part name="fault" element="impl:fault9"/>
</wsdl:message>
<wsdl:message name="getServiceNameRequest">
  <wsdl:part name="parameters" element="impl:getServiceName"/>
</wsdl:message>
<wsdl:message name="InvalidExpansionContext">
  <wsdl:part name="fault" element="impl:fault15"/>
</wsdl:message>
<wsdl:message name="getServiceVersionResponse">
  <wsdl:part name="parameters" element="impl:getServiceVersionResponse"/>
</wsdl:message>
<wsdl:message name="getServiceVersionRequest">
  <wsdl:part name="parameters" element="impl:getServiceVersion"/>
</wsdl:message>
<wsdl:message name="fillInDetailsResponse">
  <wsdl:part name="parameters" element="impl:fillInDetailsResponse"/>
</wsdl:message>
<wsdl:message name="getSupportedMatchAlgorithmsResponse">
  <wsdl:part name="parameters" element="impl:getSupportedMatchAlgorithmsResponse"/>
</wsdl:message>
<wsdl:message name="UnexpectedError">
  <wsdl:part name="fault" element="impl:fault5"/>
</wsdl:message>
<wsdl:message name="UnrecognizedQualifier">
  <wsdl:part name="fault" element="impl:fault4"/>
</wsdl:message>
<wsdl:message name="getServiceNameResponse">
  <wsdl:part name="parameters" element="impl:getServiceNameResponse"/>
</wsdl:message>
<wsdl:message name="getSupportedMatchAlgorithmsRequest">
  <wsdl:part name="parameters" element="impl:getSupportedMatchAlgorithms"/>
</wsdl:message>
<wsdl:message name="expandValueSetExpansionContextResponse">
  <wsdl:part name="parameters" element="impl:expandValueSetExpansionContextResponse"/>
</wsdl:message>
<wsdl:message name="areEquivalentResponse">
  <wsdl:part name="parameters" element="impl:areEquivalentResponse"/>
</wsdl:message>
<wsdl:message name="subsumesRequest">
  <wsdl:part name="parameters" element="impl:subsumes"/>
</wsdl:message>
<wsdl:message name="validateCodeRequest">
  <wsdl:part name="parameters" element="impl:validateCode"/>
</wsdl:message>
<wsdl:message name="getHL7ReleaseVersionRequest">
  <wsdl:part name="parameters" element="impl:getHL7ReleaseVersion"/>
</wsdl:message>
<wsdl:message name="SubsumptionNotSupported">
  <wsdl:part name="fault" element="impl:fault"/>
</wsdl:message>
<wsdl:message name="UnknownVocabularyDomain">
  <wsdl:part name="fault" element="impl:fault10"/>
</wsdl:message>
<wsdl:message name="NoApplicableValueSet">
  <wsdl:part name="fault" element="impl:fault11"/>
</wsdl:message>
<wsdl:message name="TimeoutError">
  <wsdl:part name="fault" element="impl:fault6"/>
</wsdl:message>
<wsdl:message name="translateCodeRequest">
  <wsdl:part name="parameters" element="impl:translateCode"/>
</wsdl:message>

```

```

<wsdl:message name="fillInDetailsRequest">
  <wsdl:part name="parameters" element="impl:fillInDetails"/>
</wsdl:message>
<wsdl:message name="subsumesResponse">
  <wsdl:part name="parameters" element="impl:subsumesResponse"/>
</wsdl:message>
<wsdl:message name="UnknownMatchAlgorithm">
  <wsdl:part name="fault" element="impl:fault7"/>
</wsdl:message>
<wsdl:message name="getCTSVersionResponse">
  <wsdl:part name="parameters" element="impl:getCTSVersionResponse"/>
</wsdl:message>
<wsdl:message name="validateTranslationResponse">
  <wsdl:part name="parameters" element="impl:validateTranslationResponse"/>
</wsdl:message>
<wsdl:message name="lookupValueSetExpansionRequest">
  <wsdl:part name="parameters" element="impl:lookupValueSetExpansion"/>
</wsdl:message>
<wsdl:portType name="RuntimeOperations">
  <wsdl:operation name="subsumes">
    <wsdl:input name="subsumesRequest" message="impl:subsumesRequest"/>
    <wsdl:output name="subsumesResponse" message="impl:subsumesResponse"/>
    <wsdl:fault name="QualifiersNotSupported" message="impl:QualifiersNotSupported"/>
    <wsdl:fault name="SubsumptionNotSupported" message="impl:SubsumptionNotSupported"/>
    <wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
    <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
    <wsdl:fault name="UnrecognizedQualifier" message="impl:UnrecognizedQualifier"/>
  </wsdl:operation>
  <wsdl:operation name="getSupportedMatchAlgorithms">
    <wsdl:input name="getSupportedMatchAlgorithmsRequest" message="impl:getSupportedMatchAlgorithmsRequest"/>
    <wsdl:output name="getSupportedMatchAlgorithmsResponse"
message="impl:getSupportedMatchAlgorithmsResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getSupportedVocabularyDomains">
    <wsdl:input name="getSupportedVocabularyDomainsRequest"
message="impl:getSupportedVocabularyDomainsRequest"/>
    <wsdl:output name="getSupportedVocabularyDomainsResponse"
message="impl:getSupportedVocabularyDomainsResponse"/>
    <wsdl:fault name="BadlyFormedMatchText" message="impl:BadlyFormedMatchText"/>
    <wsdl:fault name="UnknownMatchAlgorithm" message="impl:UnknownMatchAlgorithm"/>
    <wsdl:fault name="TimeoutError" message="impl:TimeoutError"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="validateCode">
    <wsdl:input name="validateCodeRequest" message="impl:validateCodeRequest"/>
    <wsdl:output name="validateCodeResponse" message="impl:validateCodeResponse"/>
    <wsdl:fault name="UnknownApplicationContextCode" message="impl:UnknownApplicationContextCode"/>
    <wsdl:fault name="NoApplicableValueSet" message="impl:NoApplicableValueSet"/>
    <wsdl:fault name="UnknownVocabularyDomain" message="impl:UnknownVocabularyDomain"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="validateTranslation">
    <wsdl:input name="validateTranslationRequest" message="impl:validateTranslationRequest"/>
    <wsdl:output name="validateTranslationResponse" message="impl:validateTranslationResponse"/>
    <wsdl:fault name="UnknownApplicationContextCode" message="impl:UnknownApplicationContextCode"/>
    <wsdl:fault name="UnknownVocabularyDomain" message="impl:UnknownVocabularyDomain"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="translateCode">
    <wsdl:input name="translateCodeRequest" message="impl:translateCodeRequest"/>
    <wsdl:output name="translateCodeResponse" message="impl:translateCodeResponse"/>
    <wsdl:fault name="UnknownApplicationContextCode" message="impl:UnknownApplicationContextCode"/>
    <wsdl:fault name="UnknownVocabularyDomain" message="impl:UnknownVocabularyDomain"/>
    <wsdl:fault name="UnableToTranslate" message="impl:UnableToTranslate"/>
    <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
  </wsdl:operation>

```

```

    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="areEquivalent">
    <wsdl:input name="areEquivalentRequest" message="impl:areEquivalentRequest"/>
    <wsdl:output name="areEquivalentResponse" message="impl:areEquivalentResponse"/>
    <wsdl:fault name="QualifiersNotSupported" message="impl:QualifiersNotSupported"/>
    <wsdl:fault name="SubsumptionNotSupported" message="impl:SubsumptionNotSupported"/>
    <wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
    <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
    <wsdl:fault name="UnrecognizedQualifier" message="impl:UnrecognizedQualifier"/>
  </wsdl:operation>
  <wsdl:operation name="fillInDetails">
    <wsdl:input name="fillInDetailsRequest" message="impl:fillInDetailsRequest"/>
    <wsdl:output name="fillInDetailsResponse" message="impl:fillInDetailsResponse"/>
    <wsdl:fault name="UnknownLanguage" message="impl:UnknownLanguage"/>
    <wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
    <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
    <wsdl:fault name="NoApplicableDesignationFound" message="impl:NoApplicableDesignationFound"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="lookupValueSetExpansion">
    <wsdl:input name="lookupValueSetExpansionRequest" message="impl:lookupValueSetExpansionRequest"/>
    <wsdl:output name="lookupValueSetExpansionResponse" message="impl:lookupValueSetExpansionResponse"/>
    <wsdl:fault name="UnknownApplicationContextCode" message="impl:UnknownApplicationContextCode"/>
    <wsdl:fault name="UnknownLanguage" message="impl:UnknownLanguage"/>
    <wsdl:fault name="NoApplicableValueSet" message="impl:NoApplicableValueSet"/>
    <wsdl:fault name="UnknownVocabularyDomain" message="impl:UnknownVocabularyDomain"/>
    <wsdl:fault name="TimeoutError" message="impl:TimeoutError"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="expandValueSetExpansionContext">
    <wsdl:input name="expandValueSetExpansionContextRequest"
message="impl:expandValueSetExpansionContextRequest"/>
    <wsdl:output name="expandValueSetExpansionContextResponse"
message="impl:expandValueSetExpansionContextResponse"/>
    <wsdl:fault name="InvalidExpansionContext" message="impl:InvalidExpansionContext"/>
    <wsdl:fault name="TimeoutError" message="impl:TimeoutError"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceName">
    <wsdl:input name="getServiceNameRequest" message="impl:getServiceNameRequest"/>
    <wsdl:output name="getServiceNameResponse" message="impl:getServiceNameResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceVersion">
    <wsdl:input name="getServiceVersionRequest" message="impl:getServiceVersionRequest"/>
    <wsdl:output name="getServiceVersionResponse" message="impl:getServiceVersionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceDescription">
    <wsdl:input name="getServiceDescriptionRequest" message="impl:getServiceDescriptionRequest"/>
    <wsdl:output name="getServiceDescriptionResponse" message="impl:getServiceDescriptionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getCTSVersion">
    <wsdl:input name="getCTSVersionRequest" message="impl:getCTSVersionRequest"/>
    <wsdl:output name="getCTSVersionResponse" message="impl:getCTSVersionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getHL7ReleaseVersion">
    <wsdl:input name="getHL7ReleaseVersionRequest" message="impl:getHL7ReleaseVersionRequest"/>
    <wsdl:output name="getHL7ReleaseVersionResponse" message="impl:getHL7ReleaseVersionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="MessageRuntimeServiceSoapBinding" type="impl:RuntimeOperations">

```

```

<wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="subsumes">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="subsumesRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="subsumesResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="QualifiersNotSupported">
    <wsdlsoap:fault name="QualifiersNotSupported" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="SubsumptionNotSupported">
    <wsdlsoap:fault name="SubsumptionNotSupported" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownConceptCode">
    <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnrecognizedQualifier">
    <wsdlsoap:fault name="UnrecognizedQualifier" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getSupportedMatchAlgorithms">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getSupportedMatchAlgorithmsRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getSupportedMatchAlgorithmsResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getSupportedVocabularyDomains">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getSupportedVocabularyDomainsRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getSupportedVocabularyDomainsResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="BadlyFormedMatchText">
    <wsdlsoap:fault name="BadlyFormedMatchText" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownMatchAlgorithm">
    <wsdlsoap:fault name="UnknownMatchAlgorithm" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="TimeoutError">
    <wsdlsoap:fault name="TimeoutError" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="validateCode">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="validateCodeRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="validateCodeResponse">

```

```

    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownApplicationContextCode">
    <wsdlsoap:fault name="UnknownApplicationContextCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="NoApplicableValueSet">
    <wsdlsoap:fault name="NoApplicableValueSet" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownVocabularyDomain">
    <wsdlsoap:fault name="UnknownVocabularyDomain" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="validateTranslation">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="validateTranslationRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="validateTranslationResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownApplicationContextCode">
    <wsdlsoap:fault name="UnknownApplicationContextCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownVocabularyDomain">
    <wsdlsoap:fault name="UnknownVocabularyDomain" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="translateCode">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="translateCodeRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="translateCodeResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownApplicationContextCode">
    <wsdlsoap:fault name="UnknownApplicationContextCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownVocabularyDomain">
    <wsdlsoap:fault name="UnknownVocabularyDomain" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnableToTranslate">
    <wsdlsoap:fault name="UnableToTranslate" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="areEquivalent">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="areEquivalentRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="areEquivalentResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="QualifiersNotSupported">
    <wsdlsoap:fault name="QualifiersNotSupported" use="literal"/>
  </wsdl:fault>

```

```

</wsdl:fault>
<wsdl:fault name="SubsumptionNotSupported">
  <wsdlsoap:fault name="SubsumptionNotSupported" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnknownConceptCode">
  <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnknownCodeSystem">
  <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnexpectedError">
  <wsdlsoap:fault name="UnexpectedError" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnrecognizedQualifier">
  <wsdlsoap:fault name="UnrecognizedQualifier" use="literal"/>
</wsdl:fault>
</wsdl:operation>
<wsdl:operation name="fillInDetails">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="fillInDetailsRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="fillInDetailsResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownLanguage">
    <wsdlsoap:fault name="UnknownLanguage" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownConceptCode">
    <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="NoApplicableDesignationFound">
    <wsdlsoap:fault name="NoApplicableDesignationFound" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="lookupValueSetExpansion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="lookupValueSetExpansionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="lookupValueSetExpansionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownApplicationContextCode">
    <wsdlsoap:fault name="UnknownApplicationContextCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownLanguage">
    <wsdlsoap:fault name="UnknownLanguage" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="NoApplicableValueSet">
    <wsdlsoap:fault name="NoApplicableValueSet" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownVocabularyDomain">
    <wsdlsoap:fault name="UnknownVocabularyDomain" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="TimeoutError">
    <wsdlsoap:fault name="TimeoutError" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>

```

```

</wsdl:operation>
<wsdl:operation name="expandValueSetExpansionContext">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="expandValueSetExpansionContextRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="expandValueSetExpansionContextResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="InvalidExpansionContext">
    <wsdlsoap:fault name="InvalidExpansionContext" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="TimeoutError">
    <wsdlsoap:fault name="TimeoutError" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceName">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceNameRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceNameResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceDescription">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceDescriptionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceDescriptionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getCTSVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getCTSVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getCTSVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>

```

```

<wsdl:operation name="getHL7ReleaseVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getHL7ReleaseVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getHL7ReleaseVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="RuntimeOperationsService">
  <wsdl:port name="MessageRuntimeService" binding="impl:MessageRuntimeServiceSoapBinding">
    <wsdlsoap:address location="http://localhost:8080/axis/services/MessageRuntimeService"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

8.2 VOCABULARY RUNTIME

```

<wsdl:definitions targetNamespace="urn://hl7.org/CTSVAPI" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="urn://hl7.org/CTSVAPI" xmlns:intf="urn://hl7.org/CTSVAPI"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/" xmlns:wSDLsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema elementFormDefault="qualified" targetNamespace="urn://hl7.org/CTSVAPI"
xmlns="http://www.w3.org/2001/XMLSchema">
      <element name="areCodesRelated">
        <complexType>

```

```

          <sequence>
            <element name="codeSystem_id" type="xsd:string"/>
            <element name="source_code" type="xsd:string"/>
            <element name="target_code" type="xsd:string"/>
            <element name="relationship_code" type="xsd:string"/>
            <element maxOccurs="unbounded" name="relationQualifiers" type="xsd:string"/>
            <element name="directRelationsOnly" type="xsd:boolean"/>
          </sequence>
        </complexType>
      </element>
      <element name="areCodesRelatedResponse">
        <complexType>
          <sequence>
            <element name="areCodesRelatedReturn" type="xsd:boolean"/>
          </sequence>
        </complexType>
      </element>
      <complexType name="UnknownRelationshipCode">
        <sequence>
          <element name="relationship_code" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <element name="fault" type="impl:UnknownRelationshipCode"/>
      <complexType name="UnexpectedError">
        <sequence>
          <element name="possible_cause" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <element name="fault1" type="impl:UnexpectedError"/>
      <complexType name="UnknownConceptCode">
        <sequence>
          <element name="concept_code" nillable="true" type="xsd:string"/>
        </sequence>

```

```

</complexType>
<element name="fault2" type="impl:UnknownConceptCode"/>
<complexType name="UnknownCodeSystem">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault3" type="impl:UnknownCodeSystem"/>
<complexType name="UnknownRelationQualifier">
  <sequence>
    <element name="relationQualifier_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault4" type="impl:UnknownRelationQualifier"/>
<element name="getSupportedCodeSystems">
  <complexType>
    <sequence>
      <element name="timeout" type="xsd:int"/>
      <element name="sizeLimit" type="xsd:int"/>
    </sequence>
  </complexType>
</element>
<element name="getSupportedCodeSystemsResponse">
  <complexType>
    <sequence>
      <element maxOccurs="unbounded" name="getSupportedCodeSystemsReturn"
type="impl:CodeSystemIdAndVersions"/>
    </sequence>
  </complexType>
</element>
<complexType name="ArrayOf_xsd_string">
  <sequence>
    <element maxOccurs="unbounded" minOccurs="0" name="item" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="CodeSystemIdAndVersions">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="xsd:string"/>
    <element name="codeSystem_name" nillable="true" type="xsd:string"/>
    <element name="copyright" nillable="true" type="xsd:string"/>
    <element name="codeSystem_versions" nillable="true" type="impl:ArrayOf_xsd_string"/>
  </sequence>
</complexType>
<complexType name="TimeoutError">
  <sequence/>
</complexType>
<element name="fault5" type="impl:TimeoutError"/>
<element name="isConceptIdValid">
  <complexType>
    <sequence>
      <element name="concept_id" type="impl:ConceptId"/>
      <element name="activeConceptsOnly" type="xsd:boolean"/>
    </sequence>
  </complexType>
</element>
<complexType name="ConceptId">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="xsd:string"/>
    <element name="concept_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="isConceptIdValidResponse">
  <complexType>
    <sequence>
      <element name="isConceptIdValidReturn" type="xsd:boolean"/>
    </sequence>
  </complexType>

```

```

</element>
<element name="lookupCodeSystemInfo">
  <complexType>
    <sequence>
      <element name="codeSystem_id" type="xsd:string"/>
      <element name="codeSystem_name" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
<element name="lookupCodeSystemInfoResponse">
  <complexType>
    <sequence>
      <element name="lookupCodeSystemInfoReturn" type="impl:CodeSystemInfo"/>
    </sequence>
  </complexType>
</element>
<complexType name="CodeSystemInfo">
  <sequence>
    <element name="codeSystem" nillable="true" type="impl:CodeSystemIdAndVersions"/>
    <element name="fullName" nillable="true" type="xsd:string"/>
    <element name="codeSystemDescription" nillable="true" type="xsd:string"/>
    <element name="supportedLanguages" nillable="true" type="impl:ArrayOf_xsd_string"/>
    <element name="supportedRelations" nillable="true" type="impl:ArrayOf_xsd_string"/>
    <element name="supportedProperties" nillable="true" type="impl:ArrayOf_xsd_string"/>
    <element name="supportedMimeType" nillable="true" type="impl:ArrayOf_xsd_string"/>
    <element name="supportedRelationQualifiers" nillable="true" type="impl:ArrayOf_xsd_string"/>
  </sequence>
</complexType>
<complexType name="CodeSystemNameIdMismatch">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="xsd:string"/>
    <element name="codeSystem_name" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault6" type="impl:CodeSystemNameIdMismatch"/>
<element name="lookupDesignation">
  <complexType>
    <sequence>
      <element name="concept_id" type="impl:ConceptId"/>
      <element name="language_code" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
<element name="lookupDesignationResponse">
  <complexType>
    <sequence>
      <element name="lookupDesignationReturn" type="impl:StringAndLanguage"/>
    </sequence>
  </complexType>
</element>
<complexType name="StringAndLanguage">
  <sequence>
    <element name="text" nillable="true" type="xsd:string"/>
    <element name="language_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="UnknownLanguageCode">
  <sequence>
    <element name="language_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault7" type="impl:UnknownLanguageCode"/>
<complexType name="NoApplicableDesignationFound">
  <sequence>
    <element name="concept_id" nillable="true" type="impl:ConceptId"/>
    <element name="language_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>

```

```

</complexType>
<element name="fault8" type="impl:NoApplicableDesignationFound"/>
<element name="getCTSVersion">
  <complexType/>
</element>
<element name="getCTSVersionResponse">
  <complexType>
    <sequence>
      <element name="getCTSVersionReturn" type="impl:CTSVersionId"/>
    </sequence>
  </complexType>
</element>
<complexType name="CTSVersionId">
  <sequence>
    <element name="major" type="xsd:short"/>
    <element name="minor" type="xsd:short"/>
  </sequence>
</complexType>
<element name="getServiceDescription">
  <complexType/>
</element>
<element name="getServiceDescriptionResponse">
  <complexType>
    <sequence>
      <element name="getServiceDescriptionReturn" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
<element name="getServiceName">
  <complexType/>
</element>
<element name="getServiceNameResponse">
  <complexType>
    <sequence>
      <element name="getServiceNameReturn" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
<element name="getServiceVersion">
  <complexType/>
</element>
<element name="getServiceVersionResponse">
  <complexType>
    <sequence>
      <element name="getServiceVersionReturn" type="xsd:string"/>
    </sequence>
  </complexType>
</element>
</schema>
</wsdl:types>
<wsdl:message name="getServiceNameRequest">
  <wsdl:part name="parameters" element="impl:getServiceName"/>
</wsdl:message>
<wsdl:message name="UnknownCodeSystem">
  <wsdl:part name="fault" element="impl:fault3"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionResponse">
  <wsdl:part name="parameters" element="impl:getServiceDescriptionResponse"/>
</wsdl:message>
<wsdl:message name="NoApplicableDesignationFound">
  <wsdl:part name="fault" element="impl:fault8"/>
</wsdl:message>
<wsdl:message name="lookupDesignationRequest">
  <wsdl:part name="parameters" element="impl:lookupDesignation"/>
</wsdl:message>
<wsdl:message name="getCTSVersionRequest">
  <wsdl:part name="parameters" element="impl:getCTSVersion"/>

```

```

</wsdl:message>
<wsdl:message name="UnknownConceptCode">
  <wsdl:part name="fault" element="impl:fault2"/>
</wsdl:message>
<wsdl:message name="areCodesRelatedRequest">
  <wsdl:part name="parameters" element="impl:areCodesRelated"/>
</wsdl:message>
<wsdl:message name="lookupCodeSystemInfoRequest">
  <wsdl:part name="parameters" element="impl:lookupCodeSystemInfo"/>
</wsdl:message>
<wsdl:message name="isConceptIdValidResponse">
  <wsdl:part name="parameters" element="impl:isConceptIdValidResponse"/>
</wsdl:message>
<wsdl:message name="lookupDesignationResponse">
  <wsdl:part name="parameters" element="impl:lookupDesignationResponse"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionRequest">
  <wsdl:part name="parameters" element="impl:getServiceDescription"/>
</wsdl:message>
<wsdl:message name="getSupportedCodeSystemsResponse">
  <wsdl:part name="parameters" element="impl:getSupportedCodeSystemsResponse"/>
</wsdl:message>
<wsdl:message name="lookupCodeSystemInfoResponse">
  <wsdl:part name="parameters" element="impl:lookupCodeSystemInfoResponse"/>
</wsdl:message>
<wsdl:message name="getCTSVersionResponse">
  <wsdl:part name="parameters" element="impl:getCTSVersionResponse"/>
</wsdl:message>
<wsdl:message name="UnknownLanguageCode">
  <wsdl:part name="fault" element="impl:fault7"/>
</wsdl:message>
<wsdl:message name="isConceptIdValidRequest">
  <wsdl:part name="parameters" element="impl:isConceptIdValid"/>
</wsdl:message>
<wsdl:message name="TimeoutError">
  <wsdl:part name="fault" element="impl:fault5"/>
</wsdl:message>
<wsdl:message name="getSupportedCodeSystemsRequest">
  <wsdl:part name="parameters" element="impl:getSupportedCodeSystems"/>
</wsdl:message>
<wsdl:message name="UnknownRelationQualifier">
  <wsdl:part name="fault" element="impl:fault4"/>
</wsdl:message>
<wsdl:message name="areCodesRelatedResponse">
  <wsdl:part name="parameters" element="impl:areCodesRelatedResponse"/>
</wsdl:message>
<wsdl:message name="UnknownRelationshipCode">
  <wsdl:part name="fault" element="impl:fault"/>
</wsdl:message>
<wsdl:message name="getServiceNameResponse">
  <wsdl:part name="parameters" element="impl:getServiceNameResponse"/>
</wsdl:message>
<wsdl:message name="UnexpectedError">
  <wsdl:part name="fault" element="impl:fault1"/>
</wsdl:message>
<wsdl:message name="getServiceVersionResponse">
  <wsdl:part name="parameters" element="impl:getServiceVersionResponse"/>
</wsdl:message>
<wsdl:message name="getServiceVersionRequest">
  <wsdl:part name="parameters" element="impl:getServiceVersion"/>
</wsdl:message>
<wsdl:message name="CodeSystemNameIdMismatch">
  <wsdl:part name="fault" element="impl:fault6"/>
</wsdl:message>
<wsdl:portType name="RuntimeOperations">
  <wsdl:operation name="areCodesRelated">
    <wsdl:input name="areCodesRelatedRequest" message="impl:areCodesRelatedRequest"/>

```

```

<wsdl:output name="areCodesRelatedResponse" message="impl:areCodesRelatedResponse"/>
<wsdl:fault name="UnknownRelationshipCode" message="impl:UnknownRelationshipCode"/>
<wsdl:fault name="UnknownRelationQualifier" message="impl:UnknownRelationQualifier"/>
<wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
<wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
<wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="getSupportedCodeSystems">
  <wsdl:input name="getSupportedCodeSystemsRequest" message="impl:getSupportedCodeSystemsRequest"/>
  <wsdl:output name="getSupportedCodeSystemsResponse" message="impl:getSupportedCodeSystemsResponse"/>
  <wsdl:fault name="TimeoutError" message="impl:TimeoutError"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="isConceptIdValid">
  <wsdl:input name="isConceptIdValidRequest" message="impl:isConceptIdValidRequest"/>
  <wsdl:output name="isConceptIdValidResponse" message="impl:isConceptIdValidResponse"/>
  <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="lookupCodeSystemInfo">
  <wsdl:input name="lookupCodeSystemInfoRequest" message="impl:lookupCodeSystemInfoRequest"/>
  <wsdl:output name="lookupCodeSystemInfoResponse" message="impl:lookupCodeSystemInfoResponse"/>
  <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
  <wsdl:fault name="CodeSystemNameIdMismatch" message="impl:CodeSystemNameIdMismatch"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="lookupDesignation">
  <wsdl:input name="lookupDesignationRequest" message="impl:lookupDesignationRequest"/>
  <wsdl:output name="lookupDesignationResponse" message="impl:lookupDesignationResponse"/>
  <wsdl:fault name="UnknownLanguageCode" message="impl:UnknownLanguageCode"/>
  <wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
  <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
  <wsdl:fault name="NoApplicableDesignationFound" message="impl:NoApplicableDesignationFound"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="getCTSVersion">
  <wsdl:input name="getCTSVersionRequest" message="impl:getCTSVersionRequest"/>
  <wsdl:output name="getCTSVersionResponse" message="impl:getCTSVersionResponse"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="getServiceDescription">
  <wsdl:input name="getServiceDescriptionRequest" message="impl:getServiceDescriptionRequest"/>
  <wsdl:output name="getServiceDescriptionResponse" message="impl:getServiceDescriptionResponse"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="getServiceName">
  <wsdl:input name="getServiceNameRequest" message="impl:getServiceNameRequest"/>
  <wsdl:output name="getServiceNameResponse" message="impl:getServiceNameResponse"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
<wsdl:operation name="getServiceVersion">
  <wsdl:input name="getServiceVersionRequest" message="impl:getServiceVersionRequest"/>
  <wsdl:output name="getServiceVersionResponse" message="impl:getServiceVersionResponse"/>
  <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="VocabRuntimeServiceSoapBinding" type="impl:RuntimeOperations">
  <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="areCodesRelated">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="areCodesRelatedRequest">
      <wsdlsoap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="areCodesRelatedResponse">
      <wsdlsoap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="UnknownRelationshipCode">

```

```

    <wsdlsoap:fault name="UnknownRelationshipCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownRelationQualifier">
    <wsdlsoap:fault name="UnknownRelationQualifier" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownConceptCode">
    <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getSupportedCodeSystems">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getSupportedCodeSystemsRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getSupportedCodeSystemsResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="TimeoutError">
    <wsdlsoap:fault name="TimeoutError" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="isConceptIdValid">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="isConceptIdValidRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="isConceptIdValidResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="lookupCodeSystemInfo">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="lookupCodeSystemInfoRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="lookupCodeSystemInfoResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnknownCodeSystem">
    <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="CodeSystemNameIdMismatch">
    <wsdlsoap:fault name="CodeSystemNameIdMismatch" use="literal"/>
  </wsdl:fault>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="lookupDesignation">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="lookupDesignationRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>

```

```

</wsdl:input>
<wsdl:output name="lookupDesignationResponse">
  <wsdlsoap:body use="literal"/>
</wsdl:output>
<wsdl:fault name="UnknownLanguageCode">
  <wsdlsoap:fault name="UnknownLanguageCode" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnknownConceptCode">
  <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnknownCodeSystem">
  <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
</wsdl:fault>
<wsdl:fault name="NoApplicableDesignationFound">
  <wsdlsoap:fault name="NoApplicableDesignationFound" use="literal"/>
</wsdl:fault>
<wsdl:fault name="UnexpectedError">
  <wsdlsoap:fault name="UnexpectedError" use="literal"/>
</wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getCTSVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getCTSVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getCTSVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceDescription">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceDescriptionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceDescriptionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceName">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceNameRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceNameResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>

```

```

</wsdl:operation>
</wsdl:binding>
<wsdl:service name="VocabRuntimeOperationsService">
  <wsdl:port name="VocabRuntimeService" binding="impl:VocabRuntimeServiceSoapBinding">
    <wsdlsoap:address location="http://localhost:8080/axis/services/VocabRuntimeService"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

8.3 CODE MAPPING

```

<wsdl:definitions targetNamespace="urn://hl7.org/CTSVAPI" xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="urn://hl7.org/CTSVAPI" xmlns:intf="urn://hl7.org/CTSVAPI"
xmlns:wSDL="http://schemas.xmlsoap.org/wsdl/" xmlns:wSDLsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <schema elementFormDefault="qualified" targetNamespace="urn://hl7.org/CTSVAPI"
xmlns="http://www.w3.org/2001/XMLSchema">
      <element name="getSupportedMaps">
        <complexType/>
      </element>
      <element name="getSupportedMapsResponse">
        <complexType>
          <sequence>
            <element maxOccurs="unbounded" name="getSupportedMapsReturn" type="impl:CodeMap"/>
          </sequence>
        </complexType>
      </element>
      <complexType name="CodeMap">
        <sequence>
          <element name="map_name" nillable="true" type="xsd:string"/>
          <element name="mapDescription" nillable="true" type="xsd:string"/>
          <element name="fromCodeSystem_id" nillable="true" type="xsd:string"/>
          <element name="fromCodeSystem_name" nillable="true" type="xsd:string"/>
          <element name="fromCodeSystem_version" nillable="true" type="xsd:string"/>
          <element name="toCodeSystem_id" nillable="true" type="xsd:string"/>
          <element name="toCodeSystem_name" nillable="true" type="xsd:string"/>
          <element name="toCodeSystem_version" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <complexType name="UnexpectedError">
        <sequence>
          <element name="possible_cause" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <element name="fault" type="impl:UnexpectedError"/>
      <element name="mapConceptCode">
        <complexType>
          <sequence>
            <element name="in0" type="impl:ConceptId"/>
            <element name="in1" type="xsd:string"/>
            <element name="in2" type="xsd:string"/>
          </sequence>
        </complexType>
      </element>
      <complexType name="ConceptId">
        <sequence>
          <element name="codeSystem_id" nillable="true" type="xsd:string"/>
          <element name="concept_code" nillable="true" type="xsd:string"/>
        </sequence>
      </complexType>
      <element name="mapConceptCodeResponse">
        <complexType>
          <sequence>
            <element name="mapConceptCodeReturn" type="impl:MappedConceptCode"/>
          </sequence>
        </complexType>
      </element>
    </schema>
  </wsdl:types>

```

```

    </sequence>
  </complexType>
</element>
<complexType name="MappedConceptCode">
  <sequence>
    <element name="mappedConcept_id" nillable="true" type="impl:ConceptId"/>
    <element name="mapQuality_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="ArrayOf_xsd_string">
  <sequence>
    <element maxOccurs="unbounded" minOccurs="0" name="item" type="xsd:string"/>
  </sequence>
</complexType>
<complexType name="AmbiguousMapRequest">
  <sequence>
    <element name="possible_maps" nillable="true" type="impl:ArrayOf_xsd_string"/>
  </sequence>
</complexType>
<element name="fault1" type="impl:AmbiguousMapRequest"/>
<complexType name="MapNameSourceMismatch">
  <sequence>
    <element name="fromCodeSystem_id" nillable="true" type="xsd:string"/>
    <element name="mapSourceCodeSystem_id" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault2" type="impl:MapNameSourceMismatch"/>
<complexType name="MapNameTargetMismatch">
  <sequence>
    <element name="toCodeSystem_id" nillable="true" type="xsd:string"/>
    <element name="mapTargetCodeSystem_id" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault3" type="impl:MapNameTargetMismatch"/>
<complexType name="MappingNotAvailable">
  <sequence>
    <element name="fromCodeSystem_id" nillable="true" type="xsd:string"/>
    <element name="toCodeSystem_id" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault4" type="impl:MappingNotAvailable"/>
<complexType name="UnableToMap">
  <sequence/>
</complexType>
<element name="fault5" type="impl:UnableToMap"/>
<complexType name="UnknownConceptCode">
  <sequence>
    <element name="concept_code" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault6" type="impl:UnknownConceptCode"/>
<complexType name="UnknownMapName">
  <sequence>
    <element name="map_name" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault7" type="impl:UnknownMapName"/>
<complexType name="UnknownCodeSystem">
  <sequence>
    <element name="codeSystem_id" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
<element name="fault8" type="impl:UnknownCodeSystem"/>
<element name="getServiceName">
  <complexType/>
</element>
<element name="getServiceNameResponse">

```

```

    <complexType>
      <sequence>
        <element name="getServiceNameReturn" type="xsd:string"/>
      </sequence>
    </complexType>
  </element>
  <element name="getServiceVersion">
    <complexType/>
  </element>
  <element name="getServiceVersionResponse">
    <complexType>
      <sequence>
        <element name="getServiceVersionReturn" type="xsd:string"/>
      </sequence>
    </complexType>
  </element>
  <element name="getServiceDescription">
    <complexType/>
  </element>
  <element name="getServiceDescriptionResponse">
    <complexType>
      <sequence>
        <element name="getServiceDescriptionReturn" type="xsd:string"/>
      </sequence>
    </complexType>
  </element>
  <element name="getCTSVersion">
    <complexType/>
  </element>
  <element name="getCTSVersionResponse">
    <complexType>
      <sequence>
        <element name="getCTSVersionReturn" type="impl:CTSVersionId"/>
      </sequence>
    </complexType>
  </element>
  <complexType name="CTSVersionId">
    <sequence>
      <element name="major" type="xsd:short"/>
      <element name="minor" type="xsd:short"/>
    </sequence>
  </complexType>
</schema>
</wsdl:types>
<wsdl:message name="MapNameSourceMismatch">
  <wsdl:part name="fault" element="impl:fault2"/>
</wsdl:message>
<wsdl:message name="UnknownMapName">
  <wsdl:part name="fault" element="impl:fault7"/>
</wsdl:message>
<wsdl:message name="getServiceNameRequest">
  <wsdl:part name="parameters" element="impl:getServiceName"/>
</wsdl:message>
<wsdl:message name="UnableToMap">
  <wsdl:part name="fault" element="impl:fault5"/>
</wsdl:message>
<wsdl:message name="getCTSVersionResponse">
  <wsdl:part name="parameters" element="impl:getCTSVersionResponse"/>
</wsdl:message>
<wsdl:message name="UnknownCodeSystem">
  <wsdl:part name="fault" element="impl:fault8"/>
</wsdl:message>
<wsdl:message name="MappingNotAvailable">
  <wsdl:part name="fault" element="impl:fault4"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionResponse">
  <wsdl:part name="parameters" element="impl:getServiceDescriptionResponse"/>

```

```

</wsdl:message>
<wsdl:message name="getSupportedMapsRequest">
  <wsdl:part name="parameters" element="impl:getSupportedMaps"/>
</wsdl:message>
<wsdl:message name="getSupportedMapsResponse">
  <wsdl:part name="parameters" element="impl:getSupportedMapsResponse"/>
</wsdl:message>
<wsdl:message name="mapConceptCodeRequest">
  <wsdl:part name="parameters" element="impl:mapConceptCode"/>
</wsdl:message>
<wsdl:message name="MapNameTargetMismatch">
  <wsdl:part name="fault" element="impl:fault3"/>
</wsdl:message>
<wsdl:message name="getCTSVersionRequest">
  <wsdl:part name="parameters" element="impl:getCTSVersion"/>
</wsdl:message>
<wsdl:message name="AmbiguousMapRequest">
  <wsdl:part name="fault" element="impl:fault1"/>
</wsdl:message>
<wsdl:message name="UnknownConceptCode">
  <wsdl:part name="fault" element="impl:fault6"/>
</wsdl:message>
<wsdl:message name="mapConceptCodeResponse">
  <wsdl:part name="parameters" element="impl:mapConceptCodeResponse"/>
</wsdl:message>
<wsdl:message name="getServiceNameResponse">
  <wsdl:part name="parameters" element="impl:getServiceNameResponse"/>
</wsdl:message>
<wsdl:message name="getServiceDescriptionRequest">
  <wsdl:part name="parameters" element="impl:getServiceDescription"/>
</wsdl:message>
<wsdl:message name="UnexpectedError">
  <wsdl:part name="fault" element="impl:fault"/>
</wsdl:message>
<wsdl:message name="getServiceVersionResponse">
  <wsdl:part name="parameters" element="impl:getServiceVersionResponse"/>
</wsdl:message>
<wsdl:message name="getServiceVersionRequest">
  <wsdl:part name="parameters" element="impl:getServiceVersion"/>
</wsdl:message>
<wsdl:portType name="CodeMappingOperations">
  <wsdl:operation name="getSupportedMaps">
    <wsdl:input name="getSupportedMapsRequest" message="impl:getSupportedMapsRequest"/>
    <wsdl:output name="getSupportedMapsResponse" message="impl:getSupportedMapsResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="mapConceptCode">
    <wsdl:input name="mapConceptCodeRequest" message="impl:mapConceptCodeRequest"/>
    <wsdl:output name="mapConceptCodeResponse" message="impl:mapConceptCodeResponse"/>
    <wsdl:fault name="AmbiguousMapRequest" message="impl:AmbiguousMapRequest"/>
    <wsdl:fault name="UnableToMap" message="impl:UnableToMap"/>
    <wsdl:fault name="MapNameTargetMismatch" message="impl:MapNameTargetMismatch"/>
    <wsdl:fault name="UnknownMapName" message="impl:UnknownMapName"/>
    <wsdl:fault name="MapNameSourceMismatch" message="impl:MapNameSourceMismatch"/>
    <wsdl:fault name="UnknownConceptCode" message="impl:UnknownConceptCode"/>
    <wsdl:fault name="MappingNotAvailable" message="impl:MappingNotAvailable"/>
    <wsdl:fault name="UnknownCodeSystem" message="impl:UnknownCodeSystem"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceName">
    <wsdl:input name="getServiceNameRequest" message="impl:getServiceNameRequest"/>
    <wsdl:output name="getServiceNameResponse" message="impl:getServiceNameResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceVersion">
    <wsdl:input name="getServiceVersionRequest" message="impl:getServiceVersionRequest"/>
    <wsdl:output name="getServiceVersionResponse" message="impl:getServiceVersionResponse"/>
  </wsdl:operation>

```

```

    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getServiceDescription">
    <wsdl:input name="getServiceDescriptionRequest" message="impl:getServiceDescriptionRequest"/>
    <wsdl:output name="getServiceDescriptionResponse" message="impl:getServiceDescriptionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
  <wsdl:operation name="getCTSVersion">
    <wsdl:input name="getCTSVersionRequest" message="impl:getCTSVersionRequest"/>
    <wsdl:output name="getCTSVersionResponse" message="impl:getCTSVersionResponse"/>
    <wsdl:fault name="UnexpectedError" message="impl:UnexpectedError"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="CodeMappingServiceSoapBinding" type="impl:CodeMappingOperations">
  <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="getSupportedMaps">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getSupportedMapsRequest">
      <wsdlsoap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="getSupportedMapsResponse">
      <wsdlsoap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="UnexpectedError">
      <wsdlsoap:fault name="UnexpectedError" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="mapConceptCode">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="mapConceptCodeRequest">
      <wsdlsoap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="mapConceptCodeResponse">
      <wsdlsoap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="AmbiguousMapRequest">
      <wsdlsoap:fault name="AmbiguousMapRequest" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnableToMap">
      <wsdlsoap:fault name="UnableToMap" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="MapNameTargetMismatch">
      <wsdlsoap:fault name="MapNameTargetMismatch" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnknownMapName">
      <wsdlsoap:fault name="UnknownMapName" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="MapNameSourceMismatch">
      <wsdlsoap:fault name="MapNameSourceMismatch" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnknownConceptCode">
      <wsdlsoap:fault name="UnknownConceptCode" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="MappingNotAvailable">
      <wsdlsoap:fault name="MappingNotAvailable" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnknownCodeSystem">
      <wsdlsoap:fault name="UnknownCodeSystem" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="UnexpectedError">
      <wsdlsoap:fault name="UnexpectedError" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <wsdl:operation name="getServiceName">
    <wsdlsoap:operation soapAction=""/>
    <wsdl:input name="getServiceNameRequest">
      <wsdlsoap:body use="literal"/>
    </wsdl:input>
  </wsdl:operation>

```

```

</wsdl:input>
<wsdl:output name="getServiceNameResponse">
  <wsdlsoap:body use="literal"/>
</wsdl:output>
<wsdl:fault name="UnexpectedError">
  <wsdlsoap:fault name="UnexpectedError" use="literal"/>
</wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getServiceDescription">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getServiceDescriptionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getServiceDescriptionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getCTSVersion">
  <wsdlsoap:operation soapAction=""/>
  <wsdl:input name="getCTSVersionRequest">
    <wsdlsoap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="getCTSVersionResponse">
    <wsdlsoap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="UnexpectedError">
    <wsdlsoap:fault name="UnexpectedError" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="CodeMappingOperationsService">
  <wsdl:port name="CodeMappingService" binding="impl:CodeMappingServiceSoapBinding">
    <wsdlsoap:address location="http://localhost:8080/axis/services/CodeMappingService"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```