An Ontology Model for Clinical Documentation Templates

by

Joyce George

M.B.B.S

Saint Johns National Academy of Health Sciences, 2002

Submitted to the Department of Health Sciences and Technology
in Partial Fulfillment of the Requirement for the Degree of
Masters of Science in Medical Informatics

at the

Massachusetts Institute of Technology

August 2005
[September 2005]

Signature of Author: [illegible]
Department of Health Sciences and Technology
August 14, 2005

Certified By: [illegible]
Aziz Boxwala
Clinical Assistant Professor of Radiology, HMS
Affiliated Faculty, Division of Health Sciences and Technology
Thesis Supervisor

Accepted By: [illegible]
Martha L. Gray
Director, HST
Edward Hood Taplin Professor of Medical and Electrical Engineering, MIT

An Ontology Model for Clinical Documentation Templates

by

Joyce George

Submitted to the Department of Health Sciences and Technology
on August 14, 2005 in Partial Fulfillment of the
Requirements for the Degree of Masters of Science in
Medical Informatics

ABSTRACT

There are various kinds of clinical documents used in a hospital or clinic setting. With the emergence of Electronic Medical Records, efforts are being made to computerize these documents in a structured fashion in order to enable decision support. With structured data entry, because each fact about the patient is stored discretely and can be retrieved separately, information can be organized and presented in different ways, depending on the needs of the user. A typical structured clinical document contains a range of findings recorded by a physician, nurse or other care. These findings can be thought of as discrete pieces of information, called observations. These observations can be grouped together to form observation sets that can be placed under relevant headers within the document. When building information systems that support structured clinical documentation, these observations and sets are created and stored in catalogs. My thesis addresses the issue of building an ontology model for clinical documentation that supports the creation and management of an observations catalog, observation sets catalog and a clinical document catalog. The ontology can be used as an organizational tool for efficient maintenance of these catalogs. By tagging observations and observation sets with relevant attributes, it is possible to generate intelligent displays of data that are more flexible and dynamic.

Thesis Supervisor: Aziz Boxwala
Title: Clinical Assistant Professor of Radiology, HMS

Acknowledgments

I would like to thank my advisor Aziz Boxwala for providing insightful feedback, criticism and direction to my work, without whose guidance this thesis would not have been possible.

Thanks to Davide for all the help, I needed with Protégé. Thanks to all my friends, both here, in India and in other parts of the world, for always being there and lending a listening ear even at the most odd hours.

Of course, none of this would have been possible without the constant support and understanding of my family, so thank you Mom, Dad, Priya and Kiran.

TABLE OF CONTENTS

1 Introduction

There are various kinds of clinical documents used during patient care – clinician encounter notes, discharge summaries, referral letters, and vital signs flowsheets, to name a few. With the emergence of electronic medical records, efforts are being made to computerize these documents in an intelligent fashion. Implementation of an information system with databases, structured data entry, and message standardization results in patient records that are far more flexible and harness more of the computer's ability as an analytical tool. Because each fact about the patient is stored discretely and can be retrieved separately, information can be organized and presented in different ways, depending on the needs of the user. The records can easily incorporate information from laboratory and administrative systems and the information systems of other institutions, and they can be supplemented by decision support systems.

The clinical documentation section of the patient record comprises the medical history, physical examination, progress notes, and reports on additional tests and interventions. A typical structured clinical document in an electronic medical record contains a range of such findings recorded by a care provider. These findings can be thought of as discrete pieces of information, called observations. These observations are grouped together in logical collections to form observation sets. For example, one might construct an observation set for physical examination of the abdomen that contains among others observations to record tenderness and distension. The observation sets are grouped together in document templates constructed for specific clinical situations such as abdominal pain. In many clinical information systems, these observations and sets are created and stored in catalogs for use within clinical document templates.

Observations stored in an unorganized catalog are difficult to search and retrieve. For example, while building a vital signs flowsheet template one will want to put into the flowsheet only observations that are of type "vital signs". Unless observations are tagged with attributes such as vital signs or body system involved, it is not easy to retrieve only the required observations from a catalog that contains thousands of diverse observations. Another example would be while building a template for Urinary Tract Infections, the creator of the template might want to retrieve all observations that deal with fever and the genitourinary system. Moreover, in many present day systems, clinical documents have a non-customizable, rigid style of display. For example, most notes have pre-defined headers like chief complaints (CC), history of present illness (HPI), review of systems (ROS), physical examination (PE). It is not possible to generate sections dynamically, for example replacing CC, HPI and ROS by subjective findings and PE by objective findings or having a section that displays both subjective and objective findings in a pithy paragraph or displaying a PE with more emphasis on cardiovascular system rather than psychiatric examination.

My thesis addresses the issue of building an ontology model for clinical documentation that supports the creation and management of an observations catalog, observation sets catalog and a clinical document template catalog. The ontology can be used as an organizational tool for classifying and retrieving observations from these catalogs and thus help in the building of templates. Additionally, by tagging observations and observation sets with relevant attributes, the ontology provides a framework to render different views (i.e. displays of data that are more flexible and dynamic) since observations can be classified into different hierarchies.

2 Background

2.1 The Medical Record

A medical record is a record that is kept for each patient by a healthcare professional or organization. It contains the patient's demographic details (such as name, address, date of birth), a summary of the patient's medical history, and documentation of each event, including symptoms, diagnosis, treatment and outcome. Relevant documents and correspondence are also included. Traditionally, each healthcare provider involved in a patient's care has kept an independent record, usually on paper. The main purpose of the medical record is to provide a summary of a patient's health history to ensure appropriate healthcare. Information from medical records also provides the data for monitoring the quality of care, and for assessing patterns of care and service delivery.

Considerable effort is invested in writing, filing, sorting, searching, retrieving, issuing and recovering the medical record, completely or in part. The ready availability of well-organized, legible, accurate and comprehensive clinical notes can play a very significant role in the clinical decision making process and in the provision of quality healthcare.

The uses of the medical record are to:

- enable health professionals to review previous care events, to reach timely and appropriate clinical decisions, and to develop treatment plans that minimize the risks and maximize the potential benefits to the patient
- provide an archival and legally acceptable record of the steps that were taken - when, why and by whom - in the care of an individual
- provide material for researchers studying the etiology, natural history and cost-effective approaches to treatment of specific conditions
- act as a source of information which will enable various administrative functions of the healthcare service unit to be carried out automatically as a by-product of the clinical data collected. For example, alert the intensive care unit of an admission when a head trauma victim is brought to the emergency department.

2.1.1 Limitations of the Paper-Based Medical Record

The paper-based medical record has many shortcomings limiting its usefulness to support the roles described above. Some of these shortcomings are discussed in this section.

Finding and re-using data items

It is difficult to find information in a paper record because:

1. Information is usually captured in an un-coded, free-text format.
2. Important information may be hidden amongst the clutter of trivia.
3. Information may simply not have been collected or recorded, or may have been misplaced.

Thus, it hampers the optimal treatment of patients because a clinician might miss an important finding. Similarly, when billing is associated with case complexity, the billing data might be incorrect if important findings are not extracted. Finally, for aggregate analyses, data has to be abstracted. The abstraction process can be expensive and error-prone.

Fragmentation of the record

The growing need to share the care of patients between healthcare providers (for example, between general practitioner (GP) and consultant, between one clinic and another) is often poorly served by the paper record. It can only be in one place at a time and logistical issues make it difficult to move it around as fast as is needed. In practice, every healthcare unit has a separate record for each individual, thereby creating a serious problem of record fragmentation and disintegrity. This can lead to potentially serious problems of continuity of care for the patient.

Decision support systems

Interest is fast growing in the use of decision support systems as a means to improve care quality and decrease cost. These assume a very wide range of forms, but can, for example, provide expert critiques of clinical decisions, indicate the most cost-effective options for further work-up, investigation and management, or provide context-specific knowledge from the biomedical literature. Decision support is not possible with paper records since natural language is ambiguous and the computer's analytical powers are required to perform most of the decision support tasks.

In order to overcome all the above-mentioned shortcomings of paper-based records, electronic medical records (EMRs) are being built.

2.1.2 Electronic Medical Records

Some of the benefits of an EMR are

Availability, transfer and retrieval

Electronic files can be readily accessed from anywhere, local or remote, across a communications link or network. Data that are stored in electronic formats can be retrieved electronically. More than one user at a time can have access to them, and all service providers can share the same records.

Linkage

Records made by multiple providers in different locations and units can be linked and shared to create a single record for the individual. The problem of record fragmentation can be resolved, and patient care can be genuinely shared between providers. Further all the graphic data (for example, images), incoming letters (for example, referrals) and auditory data (for example, heart sounds, spoken notes) relating to a patient can be linked to their electronic record file using multimedia techniques.

Abstraction and reporting

Once in electronic format, records can be used more easily for reporting. Patients' treatments can be assigned to Diagnosis Related Groups (DRGs), and statistical reports can be generated for auditing and quality assurance,. Automatic audit reports can be prepared, for example of caseloads, services provided, lengths of stay, and costs of care. Data can be gathered more easily for research studies. In addition, all the data required for administration and contract management can be derived automatically from the medical records.

Data quality and standards

Data can be checked as they are entered to ensure adequacy and accuracy by querying entries that are unlikely (for example, heart rate 200-220) or rejecting those that are impossible (for example, plasma sodium outside the range 120-160 meq/L). Results and reports can be entered directly from other systems, eliminating the possibility of misfiling and of transcription errors.

Decision support

Direct links to knowledge-based tools can be built: the present development of the Arden syntax and a library of Medical Decision Logic Modules will make it possible for a clinical information system to incorporate intelligent alerting flags to users warning them of possible errors, and advising on the best way forward.

Data views

One of the special benefits of structured records, if designed appropriately is their ability to display different views - for example, all current medications, or problems; the last ten full blood counts in graphic display; test results for a specified admission or date range. The data in the record are no longer static and accessible only in the order and format determined by the writer, but can be dynamically displayed in any way that suits the needs of different viewers.

2.1.3 Content of the Medical Record

The data in the medical record can be represented in two forms:

a. 'Free text', that is an unstructured description of findings,

b. Structured form, often as a numeric, date-time, Boolean, or coded value. Structured data can have associated with them controlled medical vocabulary terms (for example, from SNOMED-CT or ICD-9 (ref)).

The reason for making this distinction is that each has to be treated in a quite different way, and each imposes different constraints on the capacity for and technology required to transfer, sort, search, and manipulate the record.

2.1.4 Free Text

One of the main purposes in storing much of the data in the record is in order to report upon it. Being able to find it with a search is therefore vitally important. Free text in the record can be searched for the occurrence of 'strings' of characters, that is, for a match with a specified sequence of letters or numbers. In this regard, text string searches have certain shortcomings:

- They are slow and cumbersome, the more so the larger the database.
- If the word has been mis-spelled, the search fails to find the entry.
- If a different word meaning the same (synonym) was entered, the search will not find it (for example, AMI or heart attack in place of myocardial infarct). Similarly, if an abbreviation has been used, the search will not detect it.
- If the entry has been made in the wrong field (see structured text below) of the database (for example, the diagnosis has been entered where the system expects the complaint to be stored), the search will fail to detect it.
- If a search is required for a group of entities (for example, all renal diseases, or all medications), it is a very long and difficult (probably near impossible) job to string search for each of the possible terms that might have been used to describe all renal conditions or medications.

In summary, free text can be useful in electronic records where the data will simply be retrieved for display or printing. Whilst some limited manipulation (searching, analysis) is possible, it is usually difficult and unreliable.

2.1.5 Structured Data

Structuring the text makes the problem of searching much more effective and efficient. Structured data entry can capture more complete information than comparable free-text records [1, 2], can facilitate computer-based decision support [3, 4], and may reduce the time and expense of dictating and transcribing reports. Structure can be introduced by:

1. providing a framework within which the data are to be entered (for example, checkboxes);
2. limiting the entries to a specified range (e.g., specifying a set of 'preferred' terms from a dictionary and rejecting any entries not in that dictionary, or specifying a valid numeric range);

3. displaying the alternatives from which to select for each box (a 'picking' list); or

4. requiring that entries are classified (and coded).

All data that will be used for analysis or as a key for searching and sorting the record are best captured in a structured, and preferably in a coded form.

Most current day electronic medical records contain clinical data in natural language (or free text). However, in order to obtain much of the utility mentioned in the EMR section above, one requires medical data to be structured in a pre-defined format. My ontology defines a model for structured data entry of clinical findings.

2.2 Issues with Medical Documentation

Developing a framework for structured capture of medical documentation is challenging. Various issues are characteristic of clinical data.

- Apparently, simple data might require complex data structures. For example, blood pressure (BP) measurement might require separate entries for systolic and diastolic pressures though sometimes it might be acceptable to record them as a combined entry. A patient's BP may be un-recordable in certain situations or in other cases it may not have been measured (due to oversight of the provider) and this distinction between an un-recordable BP and an unmeasured BP should be made. Alternatively, it might be the case that only the diastolic pressure is un-recordable while the systolic is. This distinction needs to be documented too. There are different arterial locations where the blood pressure can be measured using different methods and different positions. The patient's clinical condition during the recording of blood pressure might also have relevance [11].

- Diverse data types used in medicine - Medical data contain numeric values (e.g. height, blood sugar), coded text, date, times, and durations, clinical drawings to describe abnormalities, signals (e.g., electrocardiograms, electroencephalograms), images (e.g., radiographs, ultrasonographs) among other data types.

- There are many coding systems used in medicine, and a shared healthcare record must allow use of any one or all of these systems. These coding systems have limitations too – (1) a limited vocabulary constrains natural expressivity (2) it may artificially skew information (3) sometimes it is difficult to combine vocabulary terms together to represent the intended meaning.

- Templates for documentation need to be optimized for various features: speed, usability, comprehensiveness of documentation, accuracy. They need to allow default values to shorten documentation time. Templates should also be able to allow charting by exception (this would apply in a case where multiple findings have been grouped together. Here the system should default to the standard negative value except for the positive findings in the patient).
- The comprehensiveness of recording of consultations varies with medical specialty and clinical setting.
- The clinical document is used by different care providers who might require different views of the document. For example, a nurse might need to see only parts of the note that document triage details, patient's vital signs and input/output charts and might not need to see details of history and physical examination documented by a physician. The billing staff might want to see only the patient's orders and laboratory tests while insurance and coding staff might want to see a patient's problem list. Within clinical specialties, a cardiologist might not want a summary of a patient's psychiatric illness, but would like to review in detail findings related to the cardiac problem.
- There are different ways in which medical data can be grouped together based on the symptoms/diseases a patient suffers from. Certain history questions need to be asked only in the presence of other findings. For example, a patient is not asked about the color and quantity of sputum produced unless he/she complains of cough first.

2.3 Efforts in Building Data Models for Electronic Model Records

There have been many efforts to formalize a data model for efficient storage and manipulation of structured clinical data. The following are some of the systems/representations that have dealt with the issue of my project's objective.

2.3.1 Health Level 7 Clinical Document Architecture

Health Level Seven (HL7), an American National Standards Institute (ANSI) accredited Standards Developing Organization (SDO) has formulated clinical document architecture (CDA), a document markup standard for the structure and semantics of exchanged clinical documents (such as discharge summaries and progress notes). CDA documents are encoded in Extensible Markup Language (XML) and can include text, images, sounds, and other multimedia content. They derive their structure and meaning from underlying vocabularies embedded in the HL7 Reference Information Model (RIM). RIM is an extensive object-oriented representation of clinical data. It specifies the actual symbols and form (referred to as syntax) as well as the semantic and lexical connections between the information carried in the fields of HL7 messages. HL7 messages are packets of clinical information that are tagged with appropriate semantics to render them meaningful and thereby help in the communication of medical data between systems.

A CDA document comprises a Header and a Body. There are four components of the CDA *Header*: (1) *Document information*: identifies the document, defines confidentiality status, and describes relationships to other documents and orders. (2) *Encounter data*: describes the setting in which a documented encounter occurred (3) *Service actors*: include those who authenticate the document, those intended to receive a copy of the document, document originators and transcriptionists, and health care providers who participated in the service(s) being documented (4) *Service targets*: include the patient and other significant participants (such as family members). The CDA *Body* contains either nested structures/containers or non-xml blobs (encapsulated data). Structures can be sections, paragraphs, lists or tables and have (optionally coded) captions. Structures contain "entries" (which can be links, coded entry, content, observation media or local markup).

The CDA specification defines a three-level architecture where each level is derived from a more basic level and iteratively adds greater markup to clinical documents. The levels refer to varying degrees of required markup granularity and specificity. Level 1 is the root of the hierarchy, most general document specification and is largely structural. Level 2 defines a set of allowable structures and semantics based on document type code. Level

3 adds additional markup capability enabling clinical content to be expressed with more granularity (RIM). Domain specializations or differences starts at this level [5, 6, 7, 8, 9].

2.3.2 Good European Health Record

The Good European Health Record (GEHR) project was established within the European Union (EU) Telematics Research and Technology Development Program to develop comprehensive multi-media data architecture for using and sharing electronic healthcare records within Europe. It is the product of a project involving 21 participating organizations in seven European countries.

Principal architectural components described in the GEHR system: The electronic health care record (EHCR) is the electronic record for one patient. The EHCR is the top-level containment structure, and is composed of one or more Transactions, together with some data enabling the record to be identified. *Transaction* represents the data from a patient record entered in one interactive session. This could result from a consultation or other contact with a patient, or perhaps from the 'filing' of a test result or letter. Seven different types of Transaction have been identified – Contact, Administration, Report, Summary, Continuing Care, Nota Bene, Trigger.

The information within a Transaction is commonly a mixed grouping of clinical findings organized under *headings*. The actual clinical information under each Heading will vary considerably with different situations, and may include historic information from the patient, clinical findings, the results of tests (which may be multi-media), procedures performed, management plans and recall dates, medication intended or actually given, letters or reports, or the results of analyses in the form of tables or graphs. All of this actual clinical data (as opposed to the Headings) is referred to in GEHR as *Observations.* The clinical Observations under a Heading is handled by the architecture construct *Health Record Item (HRI) or HRI Collection.* The Health Record Item (HRI) provides the mechanism for expressing the content value of Entries made in the record. A HRI can be regarded as the unit of information that can be obtained as the result of one specific measurement, question, observation, discussion, or other investigation mechanism. The HRI is composed of an Item Name, its primary content value, and other associated identifiers, properties and attributes. "Weight - 78 kg" and "Diagnosis - Hypertension" are simple examples. The content of a HRI can be of a wide range of data types, including dates, text strings, longer narrative comments, numeric values, and multimedia data types such as images and biosignals. The *Heading* provides a means of grouping or labeling combinations of Collections/HRIs [10, 11].

2.3.3 OpenEHR

OpenEHR is an international not-for-profit Foundation working on a formal specification for the representation and communication of electronic health record information. The *open*EHR reference models are designed within the framework of "two-level" modeling that allows the separation of generic features of any EHR from the domain-specific (usually clinical) features needed for specific EHR instances, i.e. the information and knowledge layers of systems are fully separated out. The reference models constitute a first level of modeling, while formal structured constraint definitions of clinical concepts, the knowledge models - *archetypes* - are the second level.

OpenEHR health record consists of 'container classes', which contain information about the patient or data subject:

- EHR - this is the top-level class and contains all information about the data subject.
- Extract - this class contains all information that is to be transferred to another EHR.
- Folder - this class allows information within an EHR to be organized.

A *Composition* is a container class in the *open*EHR reference model. It holds the information committed to the EHR by a clinician. Compositions contain one or more organizing *Sections* - which themselves contain *Entries*. The purpose of a section is to bring together information in a way that is useful for clinicians. A Section class can consist of further section classes. The *Entry* class of the *open*EHR reference model is like the leaf node of the document and contains the information - the blood pressure, the assessment, the diagnosis, the medication order. Importantly, entries' can be interpreted independent of the composition or the section within which they are located. All *Entry* classes contain: (1) a *Data* part which contains the core information e.g. the systolic and diastolic pressures when measuring a blood pressure. (2) a *State* part which contains information about the subject of data at the time the information was collected, for example is the position of the patient at the time of measuring a blood pressure. (3) a *Protocol* part which contains information on how the information was gathered or measured. *ENTRY*, has 3 concrete subtypes: *Evaluation* - ideas, labels or views which arise within the clinician's mind, example, diagnosis, assessment, family history, adverse reaction. *Observation* – are the clinical observations, example, blood pressure, height, weight. *Instructions* - are statements about what should happen in the future [12, 13].

2.3.4 European Committee for Standardization (CEN) ENV 13606-2

Electronic Healthcare Record Support Action (EHCR-SupA) is a consortium of European countries that produce illustrative materials to demonstrate the CEN EHCR architecture features. CEN stands for the European Committee for Standardization. CEN ENV 13606-2 information models provide a means to represent the organizational structure of one or more nested electronic healthcare record entries. This standard proposes sub-categorization of the EHR into four specializations:

- *Folder*: High-level subdivisions of the entire EHR for a patient, usually grouping entries over long time-spans within one organization or department, or for a particular health problem.
- *Composition*: A set of record entries relating to one time and place of care delivery; grouped contributions to an aspect of health care activity; composed reports and overviews of clinical progress.
- *Headed Section*: Sub-divisions used to group entries with a common theme or derived through a common healthcare process.
- *Cluster*: Low-level aggregations of elementary entries (Record Items) to represent a compound clinical concept [14, 15, 16].

2.3.5 OpenSDE

OpenSDE is an application that helps clinicians record medical findings and patient history in a structured manner. It was a project started in the medical informatics division at Erasmus University Medical Center Rotterdam. OpenSDE supports structured recording of medical narrative data in the form of an application that allows tailoring to specific medical domains and individual preferences without the need for technical adaptation [17].

Data entry is enabled using forms that are generated using domain-specific trees of medical concepts. For data storage, the authors have expanded the traditional row modeling methodology with additional columns that allow structured representation of medical narratives including descriptions of findings, multiple occurrences of findings, and the progression of findings over time. In OpenSDE, metadata are represented as discipline-specific domain models. Domain models vary in content but not in structure. The content consists of concepts and constraints organized in a rooted tree structure. The nodes of the tree structure represent the concepts and are connected to each other via one-directional arcs; a node at the end of an arc represents a descriptor of the node at the beginning of an arc. For every node, one path extends from the root to the particular node [14, 15, 16].

2.3.6 Open Record for Care: Structured Data Entry

Open Record for Care (ORCA) was developed was Erasmus University, Rotterdam. In ORCA, there is a thesaurus of concepts, in which each concept has a unique identifier, and a name. Structured data entry employs this knowledge base. The information is used in an interactive user interface for navigating through the set of medical concepts, for creating the right data entry forms, for storing and retrieving patient data in the form of instances of medical concepts and for presenting them to the user.

The structured data entry window contains functionality to browse through the knowledge base domains and their hierarchies: one can select a domain from a list, a concept in the concept path to the current concept, or search by entering a search string. Data can be entered by focusing on a medical concept and selecting the proper option from a popup menu: present, absent, or normal. The knowledge base contains medical concepts and their descriptors and is used by a data entry program to support menu-driven SDE. Depending on the contents of the knowledge base, physicians are free to determine the scope and detail of the data they enter in a structured format [17, 18, 19, 20, 21].

2.3.7 Comparison of above models with my Ontology Model

All these models, like the model to be presented in this paper deal with a way of displaying, storing and communicating structured clinical information. Some features common to these models and the ontology model to be discussed in this paper are:

- Individual data items are contained in some kind of placeholder called either a template or container or document.
- There are specialized structures or constructs within the placeholder – their definitions vary in the different models. In some models, the relationship between the various components of a document is recursive.
- Data items or the simplest components are contained either within the specialized structures or directly within the placeholder.
- Either the specialized structures or the data items or both are tagged with semantics that vary in the different models
- Relationships exist between the various structures in a document. The kinds of relationships are different in the different models. Some examples are is-a, part-of, has-specialization, has-feature and has-value relationships.

There is some similarity in the general approach between my model and the first four models described below. OpenSDE and Open Record for Care (ORCA), on the other hand use a different approach – these systems rely on an extensive knowledge base of medical concepts that is viewable by a physician or user, who then choose from these concepts to customize a form according to his needs.

My model differs from these models by being more flexible in terms of its ability to generate clinical documents of different types or different views of a document with the help of simple queries, since tagging of data elements is done at a more basic level, thus giving room for greater manipulation of data.

2.4 Description Logics

Description Logics (DLs) is a family of knowledge representation (KR) formalisms that represent the knowledge of an application domain (the "world") by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain (the world description) [22]. DLs are equipped with formal, logic-based semantics. Another distinguishing feature of DL is the emphasis on reasoning as a central service: reasoning allows one to infer implicitly represented knowledge from the knowledge that is explicitly contained in the knowledge base. Classification of concepts determines subconcept/superconcept relationships (called subsumption relationships in DL) between the concepts of a given terminology, and thus allows one to structure the terminology in the form of a subsumption hierarchy. Classification of individuals determines whether a given individual is always an instance of a certain concept (i.e., whether this instance relationship is implied by the description of the individual and the definition of the concept). It thus provides useful information on the properties of an individual.

Description Logics provides facilities to set up knowledge bases, to reason about their content, and to manipulate them. A knowledge base (KB) comprises two components, which we can call the T-Component and the A-Component. The T-Component introduces the *terminology*, i.e., the vocabulary of an application domain, while the A-Component contains *assertions* about named individuals in terms of this vocabulary. The vocabulary consists of *concepts*, which denote sets of individuals, and *properties*, which denote binary relationships between individuals. A DL system not only stores terminologies and assertions, but also offers services that *reason* about them. Typical reasoning tasks for a terminology are to determine whether a description is *satisfiable* (i.e., non-contradictory), or whether one description is more general than another one, that is, whether the first s*ubsumes* the second. Important problems for an A-Component are to find out whether its set of assertions is *consistent,* that is, whether it has a model, and whether the assertions in the A-Component entail that a particular individual is an *instance* of a given concept description. Satisfiability checks of descriptions and consistency checks of sets of assertions are useful to determine whether a knowledge base is meaningful at all. A concept description can also be conceived as a query, describing a set of objects one is interested in. Thus, with instance tests, one can retrieve the individuals that satisfy the query.

DL uses open world assumption (OWA) where the absence of information only indicates lack of knowledge and is not interpreted as negative information. For example, if the only assertion about Peter is hasChild(PETER; HARRY), in DL, the assertion only expresses that, in fact, Harry is a child of Peter. However, DL has several models, some in which Harry is the only child and others in which he has brothers or sisters. The only way of stating Harry is the only child is by doing so explicitly.

2.5 Ontologies and Protégé

Protégé is an open source knowledge-modeling platform [23] that helps developers and domain experts build conceptual models and knowledge bases in the form of ontologies. It is a description logic based ontology building tool. An ontology is a collection of domain concepts (also called classes) and their relationships [24]. Protégé supports the development of ontologies in the Web Ontology Language (OWL) [25], a description logic based ontology language created by the World Wide Web Consortium (W3C). An ontology together with a set of individual *instances* of classes constitutes a *knowledge base*. Protégé enables one to

- Load and save Web Ontology Language (OWL) and Resource Description Framework (RDF) ontologies
- Define logical class characteristics as OWL expressions
- Create and edit OWL individuals or instances
- Classify ontologies with the help of reasoners such as RACER [26]

There are three tabs commonly used in Protégé. (1) The **Classes Tab** is an ontology editor that is used to define classes and class hierarchy, properties and property-value restrictions, relationships between classes and properties. (2) The **Properties Tab** can be used to create object or data type properties and to define the domain and range of these properties. Properties are the attributes of a class. An object property has a class as the domain as well as the range of the property while a data type property has a class as the domain and a data type (such as integer, string, Boolean) as its range. (3) The **Individuals Tab** is used to create instances of the classes defined in the ontology. Forms with pre-specified fields for creating instances are automatically generated based on the conditions asserted for classes in the Classes Tab.

2.6 Features to Note in Web Ontology Language

Individuals, Properties, Classes: Individuals also known as instances, represent objects in the domain that we are interested in. Properties are binary relations on individuals - i.e. properties link two individuals together. For example, the property hasSibling might link the individual John to the individual Matt. Properties can have inverses. For example, the inverse of hasOwner is isOwnedBy. Properties can be limited to having a single value – i.e., to being functional. They can also be either transitive or symmetric. OWL Classes are interpreted as sets that contain individuals. Classes may be organized into a superclass-subclass hierarchy, which is also known as taxonomy. In OWL classes are built of descriptions that specify the conditions that must be satisfied by an individual for it to be a member of the class. Classes can be made disjoint from each other to state that there is no intersection of attributes between the classes.

Operators: Logic statements are written using operators that link classes via restrictions. Protégé supports three operators – AND ($\sqcap$), OR ($\sqcup$) and negation ($\neg$). AND operator represents the intersection between individual classes and the OR operator represents union between classes.

Restrictions: In OWL properties are used to create restrictions. Restrictions are used to restrict the individuals that belong to a class. They actually describe an anonymous class (or unnamed class). Restrictions in OWL fall into three main categories: (1) Quantifier Restrictions (2) Cardinality Restrictions and (3) hasValue Restrictions.

Quantifier restrictions are composed of a quantifier, a property, and filler (which is a class). The two quantifiers that may be used are: (a) existential quantifier ($\exists$) - an existential restriction specifies the existence of a (i.e. at least one) relationship along a given property to an individual that is a member of a specific class. (b) universal quantifier ($\forall$) – universal restrictions also known as All restrictions constrain the relationships along a given property to individuals that are members of a specific class.

Universal restrictions state that if a relationship exists for the property then it must be to individuals that are members of a specific class.

Cardinality Restrictions describe the class of individuals that have at least, at most or exactly a specified number of relationships with other individuals or data type values. For a given property P, a Minimum Cardinality Restriction specifies the minimum number of P relationships that an individual must participate in. A Maximum Cardinality Restriction specifies the maximum number of P relationships that an individual can participate in. A Cardinality Restriction specifies the exact number of P relationships that an individual must participate in.

hasValue Restriction denoted by the symbol $\ni$, describes the set of individuals that have at least one relationship along a specified property to a specific individual.

Closure Axioms are used to overcome OWA used in Protégé. A closure axiom on a property consists of a universal restriction that acts along the property to say that it can only be filled by the specified fillers. The restriction has a filler that is the union of the fillers that occur in the existential restrictions for the property

Necessary and Sufficient Conditions: Classes can be described using only necessary conditions or both necessary and sufficient conditions. Necessary conditions can be read as, "If something is a member of this class then it is necessary to fulfill these conditions". With necessary conditions alone, we cannot say that, "If something fulfils these conditions then it must be a member of this class". Necessary and Sufficient conditions on the other hand mean that not only are the conditions necessary for membership of the class, they are also sufficient to determine that any (random) individual that satisfies them must be a member of that class. This converts the description of the class into a definition. Classes that have at least one set of necessary and sufficient conditions are known as defined classes. Classes that do not have any sets of necessary and sufficient conditions (only have necessary conditions) are know as primitive classes.

Definitions: An equality whose left-hand side is an atomic concept is a *definition.* Definitions are used to introduce *symbolic names* for complex descriptions. For instance, by the axiom

$$\mathsf{Mother} \equiv \mathsf{Woman} \sqcap \exists \mathsf{hasChild}.\mathsf{Person}$$

we associate to the description on the right-hand side the name Mother. If, for example, we have defined Father analogously to Mother, we can define Parent as

$$\mathsf{Parent} \equiv \mathsf{Mother} \sqcup \mathsf{Father}.$$

Racer: A reasoner called RACER [26] is used to perform the reasoning mentioned in the Description Logic section. RACER is typically started by double clicking on the RACER application icon, which opens a terminal/console window and starts the reasoner running with HTTP communication enabled. Having started RACER, or another reasoner, the ontology can be 'sent to the reasoner' to automatically compute the classification hierarchy, and also to check the logical consistency of the ontology. In Prot´eg´e-OWL the 'manually constructed' class hierarchy is called the asserted hierarchy. The class hierarchy that is automatically computed by the reasoner is called the inferred hierarchy [26].

2.7 Using Protégé

In order to create and test an ontology in Protégé, these are the general steps followed.

- create and describe classes using necessary conditions
- Arrange the classes in a taxonomic (subclass–superclass) hierarchy
- Define properties and assign the domain and range for these properties
- Define universal or existential restrictions on properties for the various classes.
- Use AND, OR, negation operators, cardinality rules where necessary.
- Check for inconsistencies in the ontology (using Racer) and the 'Classify Taxonomy' action button. When the inferred hierarchy has been computed, an inferred hierarchy window pops open next to the existing asserted hierarchy window. If a class has been reclassified (i.e. if its super classes have changed) then the class name will appear in a blue color in the inferred hierarchy. If a class has been found to be inconsistent, its icon will be circled in red.

- Demonstrate the use/application of the ontology by creating defined classes (using necessary and sufficient conditions) either in the same file or in a new file with the imported ontology
- Create Instances of these defined classes
- With the help of racer check to see if instances are being classified correctly using the 'Compute Inferred Types' action button.

In the next chapter, I describe how I have used Protégé to build an ontology in OWL using the sequence of steps listed above.

3 Ontology model

I have built a generic clinical document model that can be applied to various types of clinical documents. Simpler and more generic models are more robust to changing requirements and are cheaper to implement and maintain than complex and application-specific models.

3.1 A Conceptual View of Clinical Documents

Any clinical document can be considered an organized collection of observations. These observations can be sentences, paragraphs, coded text, images, numerical data, a string of words, etc. The way in which these observations are organized in a document may vary depending on the type of document. A Physician Encounter Note for example might have different sections like Chief Complaints, Past History, Family History, ROS, and Physical Examination. The content within these sections might be organized as discrete, coded data or free text paragraphs or a combination of both. On the other hand, a Referral Letter will contain information on referring and consulting providers, and a brief description of the patient's condition, diagnosis and reason for referral. Whatever the representation or layout of the document, the content captured within a document can be thought of as building blocks called *Observations*[1].

I used the above insights to build the model for clinical documentation in the form of an ontology. An OWL schema was created with classes, class restrictions and properties.

3.2 Model Architecture

The container for the content of a clinical document is called a *Template.* When the template is used for a particular patient to document clinical information, an instance of a *clinical document* is created. A *Template* is made up of two key components: *Observations* and *ObservationSets* (shown as *Observation Set* in Figure 1). *Observations* are the smallest bits of semantically meaningful data in the context of the patient. *ObservationSets* are collections of *Observations.* A *Template* may contain (one or more *Observations* OR one or more *ObservationSets*). *ObservationSets* can be made up of (one or more *Observations* OR one or more *ObservationSets*). The structure of a *Template* is depicted in Figure 1.

[1] *Italicized font* is used in this document to indicate names of classes and relationships in the model and ontology.

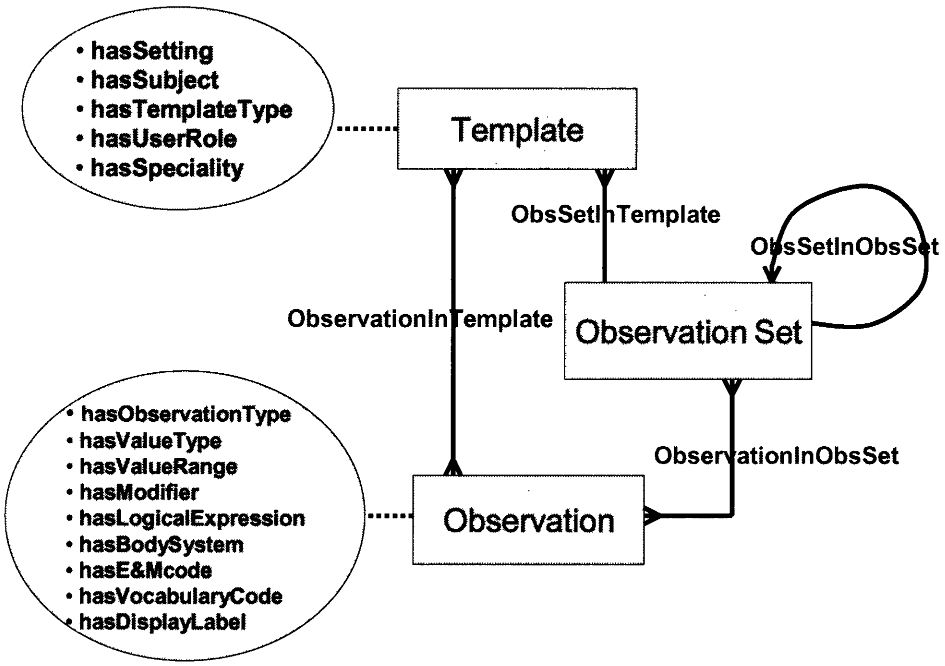


Figure 1: Clinical Document/Template Model

ObservationInTemplate, *ObsSetInTemplate*, *ObservationInObsSe*t and *ObsSetInObsSet* are used to depict the relationship between *Template*, *Observation* and *ObservationSet.* These relationships are modeled as classes in Protégé and are assigned attributes that specify the domain and range of each relationship and the sequence number of the *Observation* or *ObservationSet* within its container. The bubbles in the diagram show the properties assigned to *Template* and *Observation.*

Observation, *ObservationSet* and *Template* are the key entities in my model. They are tagged with various properties. These tags are descriptors of items within these classes and can be used for classifying the items. For example, *Observation* have these properties as tags: (1) *ObservationType* – whether it is a history observation or physical examination observation or laboratory test result. History observations can be further classified into chief complaints, history of present illness and past history to name a few. Furthermore history and physical examination observations can be classified based on body system affected; (2) *ObservationValueType* – whether the observation is a free text observation or coded observation or numeric data or image; and (3) *ObservationModifier* – whether the observation is a severity observation, location observation, for example.

Appendix 1 shows all the classes within the ontology in the asserted class- subclass hierarchy. This ontology was built in a file named document.owl. There are 20 classes in the first level of the hierarchy, 89 classes in the second level, 46 classes in the third level, 22 in the fourth, eight classes in the fifth and 64 classes in the sixth level of the hierarchy making a sum of 249 classes in the entire ontology schema.

Appendix 2 shows all the object and data type properties defined in the ontology seen in Table 2 and Table 3. There are 13 data type properties and 18 object properties making a sum of 31 properties.

Appendix 3 shows the asserted conditions widget of classes in the ontology with necessary and sufficient conditions along with the narrative version of description logic statements.

4 Results

The OWL schema file (named as document.owl) described in the previous chapter contains the ontology with classes that defined the components of a clinical document. This ontology was imported into a new OWL file called cd_instances.owl where defined classes of type *Observation*, *ObservationSet* and *Template* were created. This was followed by the creation of instances of the *Observation*, *ObservationSet* and *Template* classes. Inference rules embedded in the definition of the defined classes were used to classify automatically the instances into the correct defined classes. This was done to check the functionality of the document ontology, refine it where required based on use cases, test the consistency of logic rules embedded in the model (ontology schema) and demonstrate the overall utility of the model.

4.1 Consistency Check of Clinical Document Ontology

All the classes in the OWL schema were defined individually and placed in a particular super class-subclass hierarchy. This is known as an asserted hierarchy. The intention was to ensure that on running Racer, the reasoning engine, the class hierarchy would still be maintained – in other words, the inferred hierarchy would be the same as the asserted hierarchy. In other words, the ontology was defined such that there should be no subsumption of classes. Since OWL follows open world assumption, it was important to make sure that descriptions of classes were accurate and were interpreted by Protégé as intended. The ontology was classified by using the "classify taxonomy" feature in Protégé, which in turn uses Racer. A comparison of the resulting inferred hierarchy with the asserted hierarchy showed that there were no inconsistencies in the ontology.

4.2 Creation of Defined Classes to Demonstrate Utility of Ontology

New defined classes were created in the cd_instances.owl file to demonstrate the utility of the data model. We created defined classes to demonstrate the following uses of the ontology

- It can be used as a classification tool – to classify different kinds of observations, observation sets and documents.

- It can be used as a querying tool to retrieve only a particular group/set of observations based on the tags defined in the document ontology.
- It can be used to generate different display views of documents, for example nurse views (only the observations that a nurse would like to see, filtering out others), or specialist views (only what a cardiologist or psychiatrist would like to see).
- Generate forms/documents on the fly based on observations in the template. For example, a physician encounter document requires history observations, physical examination observations, probably laboratory tests, and a problem list.

Nineteen sub-classes of type *Observation*, four sub-classes of type *Observation Set* and four sub-classes of type *Template* were created.

4.2.1 Creating Sub-Classes of *Observation*

The following subclasses of Observation were created:

HistoryObservation: all history observations

PastHistoryObservation: all past history observations

HPILocationObservation: history of present illness (HPI) observations that relate to location of the presenting complaint

ROSObservation: observations related to the review of systems.

CVSHPIObservation: HPI observations related to the cardiovascular system

RSHPIObservation: HPI observations related to the respiratory system

AnyObservationByBodySystem: observations related to a body system

CVSObservation: observations related to the cardiovascular system

CVSPEObservation: PE observations related to the cardiovascular system

PEObservation: any PE observation

PalpatoryObservation: Any PE observation which is a palpatory observation. In the ontology, palpation is defined as a modifier for PE observations

AuscultatoryObservation: Any PE observation, which is an auscultatory observation

CardiacAuscultatoryObservation: Any PE observation that is an auscultatory observation and is related to the cardiovascular system

LabResultObservation: Laboratory test results

FreeTextObservation: Observations that are documented as sentences /phrases in free text format (i.e. information that is not marked up)

NumericDataObservation: Observations that document numeric values. For example laboratory results, hemoglobin value of 15.

StructuredObservation: Observations that are recorded as discrete meaningful data that can be coded using standardized vocabularies.

Appendix 4 lists the various defined classes of type *Observation.* Definitions are stated in logic statements followed by English definitions.

4.2.2 Creating Sub-Classes of Type *ObservationSet*

As the names, suggest

LabTestSet: is an observation set that contains only laboratory test observations

ProcedureObservationSet: is an observation set that contains only operative procedure observations

PastHistorySet: is an observation set that contains only past history observations

ProblemListSet: is an observation set that contains only problems/diseases a patient is suffering from.

Appendix 5 lists the defined classes of type *ObservationSet.* Definitions are stated in logic statements followed by English definitions.

4.2.3 Creating Sub-Classes of Type *Template*

EncounterNote: is a clinical document or template that is filled out by a physician after questioning and examining a patient who comes in with disease complaints or for a routine physical. In some cases, it might contain just the patient's demographics, history, PE and problem list or diagnosis. At other times, it might contain orders, laboratory results in addition.

ProcedureNote: is a clinical document that is generated after a clinical procedure. It could be a major operative procedure like a cardiac bypass surgery, or a minor outpatient procedure like abscess drainage, a diagnostic procedure like angiography, an interventional procedure like renal stent insertions, or therapeutic procedure like radiotherapy for cancer.

PrimaryCareNote: is a clinical documentation note used in a primary care setting. These notes are in practice used by primary care providers.

EDNote: is a clinical documentation note created in an emergency department (ED). Though the main framework of documenting clinical findings is the same between ED physicians and primary care physicians, there are still subtle differences in the nature and style of documentation adopted in these two settings. For example, ED is more concise and skeletal while primary care documentation is more elaborate, covering many ancillary questions that are barely skimped by ED physicians. These differences occur across specialties and other clinical settings too.

Appendix 6 lists the defined classes of type *Template.* Definitions are stated in logic statements followed by English definitions.

Though not demonstrated in these examples, the ontology also supports the retrieval of other observations. For example, all observations with a severity indicator, all family history or social history findings, review of system observations, patient's demographic data, orders, provider information to name a few. Notes can be classified based on specialty, location where used, type of note (referral letter, discharge summary), subject of the note (whether the note is about a specific disease condition or a routine visit note).

4.3 Validation of Instances Ontology

Racer was run on the cd_instances.owl file (also called validation file) using "classify taxonomy" action to check for inconsistencies in the definitions of the new classes. The inferred hierarchy generated by Protégé is seen in Figure 2. In the figure, the asserted hierarchy is depicted on the left and the inferred hierarchy on the right as seen by the labels. This screenshot displays how the classes of type *Observation* have been re-arranged by Protégé (in the inferred hierarchy) based on the definitions of those classes. As seen in the display window, Racer correctly classifies the observation classes into distinct types – *AnyObservationByBodySystem, FreeTextObservation, LabResultObservation, NumericDataObservation,* and *StructuredObservation.*

AnyObservationByBodySystem subsumed the classes *CVSObservation, HistoryObservation* and *PEObservation.* These three classes were classified under *AnyObservationByBodySystem*, since all of them are observations that can have a body system associated with them.

CVSObservation further subsumed-classified correctly *CVSHPIObservation* and *CVSPEObservation* since both of them are observations related to the cardiovascular system. *CVSPEObservation* is further sub-classified into *CardiacAuscultatoryObservation* since the latter is a kind of cardiac PE observation.

HistoryObservation on the other hand is sub-classified into *CVSHPIObservation, HPILocationObservation, PastHistoryObservation, ROSObservation* and *RSHPIObservation* since all of them are types of History Observations.

PEObservation was sub-classified into *AuscultatoryObservation, CVSPEObservation* and *PalpatoryObservation. CardiacAuscultatoryObservation* was correctly classified both under *AuscultatoryObservation, CVSPEObservation.*

In the case of *ObservationSet* and *Template* classes, no subsumption of classes was performed by the reasoner since these defined classes were primarily standalone classes.

Since Racer classified these classes correctly, the definitions of the classes stated earlier were validated.

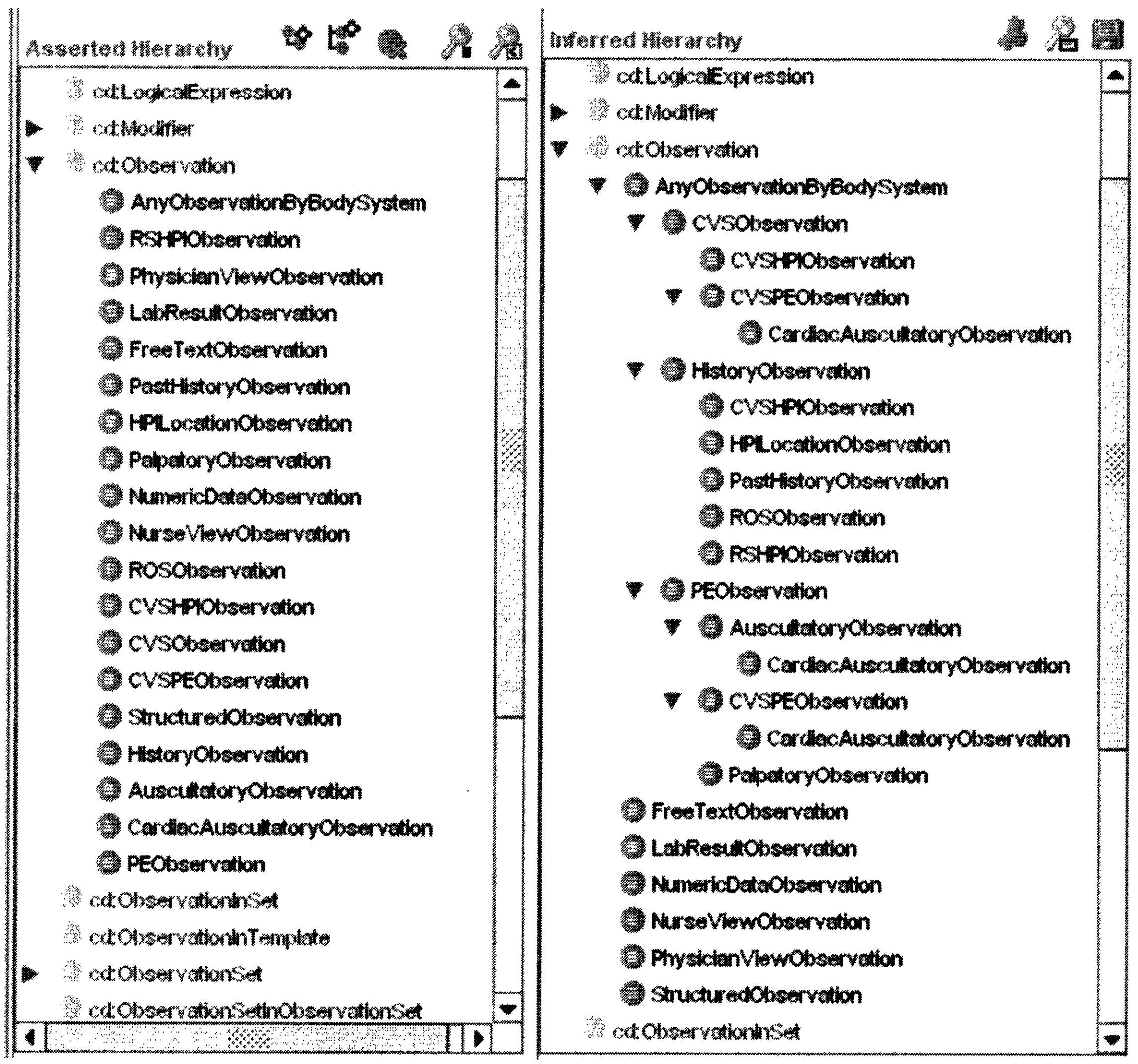


Figure 2: Inferred Hierarchy of Defined sub-classes of type OBSERVATION

4.4 Classification of Instances

Instances of three classes (*Observation*, *ObservationSet* and *Template*) from the document ontology were created in the cd_instances.owl file (validation file). The instances are shown in Table 1. These instances were created by filling fields on instance editing forms created by Protégé. The field values in the forms are constrained by restriction conditions described for the classes in the document ontology. In the case of *CVS_HPI_Observation1* (seen in Table 1), the definition of the instance and the narrative section of the instance definition reads as shown in Figure 3. Creation of the instances listed in Table 1 required creation also of instances of classes linked to these three classes via properties shown in their conditions widget (see Appendix 3).

Instances Created in cd_instances.owl (validation file)		
***Observation* Instances**	***ObservationSet* Instances**	***Template* Instances**
Chief_Complaint	Lab_Test_Set_1	Encounter_Note_Template_1
CVS_HPI_Observation1	Past_History_Set_1	Procedure_Note_Template_1
CVS_HPI_Observation2	Problem_List_Set_1	
CVS_HPI_Observation3	Procedure_Set_1	
CVS_PE_Observation1		
CVS_PE_Observation2		
CVS_PE_Observation4		
Lab_Test_1		
Lab_Test_2		
Lab_Test_3		
NumericData_Observation1		
Past_history_Observation2		
Past_history_Observation3		
Problem_List_Observation_1		
Problem_List_Observation_2		
Procedure_Observation_1		
Procedure_Observation_2		
ROS_Observation1		
RS_HPI_Observation1		
RS_HPI_Observation2		
RS_HPI_Observation3		
RS_PE_Observation		

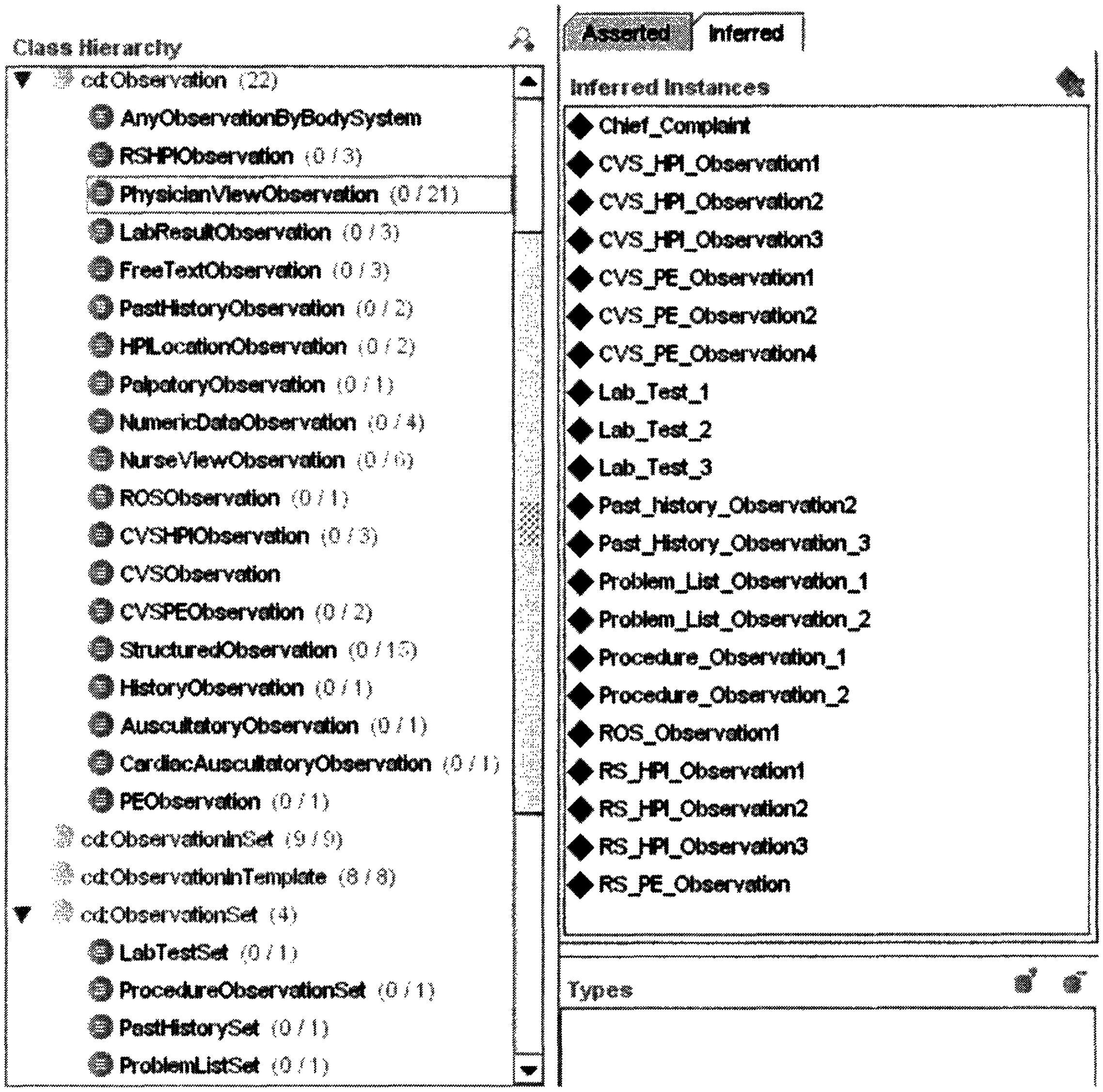
Table 1: List of Instances Created in the validation file

CVS_HPI_Observation1 • has DisplayLabel Cardiac Dyspnea • hasType HPI • hasValueType List	*CVS_HPI_Observation 1*is a thing that belongs to the *Observation* class, has DisplayLabel Cardiac Dyspnea, is of type HPI and has ValueType List (i.e. it is a discrete, coded or structured observation)

Figure 3: Definition of CVS_HPI_Observation1 shown on the left and narrative form of the definition shown on the right.

In order to demonstrate that these instances were classified correctly, Racer was employed and the "compute inferred types" action was chosen. All instances were classified correctly under the appropriate defined class for that instance (seen in Figure 4 and Figure 5). In these figures, the inferred instances for a particular class are shown on the right hand side while the classes are shown on the left hand side. Figure 4shows that the class *Observation* has 22 instances (shown in parenthesis) defined under it. If the *Observation* class is selected, these instances are seen under the asserted tab of the instance browser. On running racer, these instances are classified under the appropriate class and can be viewed in the inferred tab of the instance browser. The numbers in parenthesis against each class indicate the number of instances classified under that class. For example, *LabResultObservation* has (0/3) against it. This indicates that zero instances were asserted under this class but three instances were inferred to belong to this class based on the definition of the instance. Figure 5 shows that four instances were created under *ObservationSet* and two instances under *Template*, which were then appropriately classified. The following are examples instances and their classification by Racer:

- The blood glucose level instance (Lab_Test_1) was classified under *LabResultObservation.*
- Past history of Cancer and myocardial infarction instances (Past_history_Observation2 and Past_history_Observation3) were classified under *PastHistoryObservation* which was further classified under *HistoryObservation.*
- Heart murmur instance (CVS_PE_Observation1) was classified under *CardiacAuscultatoryObservation*, which was further classified under *CVSPEObservation.*
- Cough, wheeze and pleuritic pain instances (RS_HPI_Observation1, RS_HPI_Observation2, RS_HPI_Observation3) were classified under *RSHPIObservation* and further under *HistoryObservation.*
- *LabTestSet* contained blood glucose level (Lab_Test_1), blood cholesterol level (Lab_Test_2) and hemoglobin levels (Lab_Test_3), i.e. all the laboratory instances.
- The *EDNote* contained only ED instances.
- The *ProcedureNote* only contained procedure note sets that contained only procedure note instances.



Class Hierarchy

- cd:Observation (22)
  - AnyObservationByBodySystem
  - RSHPIObservation (0 / 3)
  - PhysicianViewObservation (0 / 21)
  - LabResultObservation (0 / 3)
  - FreeTextObservation (0 / 3)
  - PastHistoryObservation (0 / 2)
  - HPILocationObservation (0 / 2)
  - PalpatoryObservation (0 / 1)
  - NumericDataObservation (0 / 4)
  - NurseViewObservation (0 / 6)
  - ROSObservation (0 / 1)
  - CVSHPIObservation (0 / 3)
  - CVSObservation
  - CVSPEObservation (0 / 2)
  - StructuredObservation (0 / 13)
  - HistoryObservation (0 / 1)
  - AuscultatoryObservation (0 / 1)
  - CardiacAuscultatoryObservation (0 / 1)
  - PEObservation (0 / 1)
- cd:ObservationInSet (9 / 9)
- cd:ObservationInTemplate (8 / 8)
- cd:ObservationSet (4)
  - LabTestSet (0 / 1)
  - ProcedureObservationSet (0 / 1)
  - PastHistorySet (0 / 1)
  - ProblemListSet (0 / 1)

Asserted | Inferred

Inferred Instances

- Chief_Complaint
- CVS_HPI_Observation1
- CVS_HPI_Observation2
- CVS_HPI_Observation3
- CVS_PE_Observation1
- CVS_PE_Observation2
- CVS_PE_Observation4
- Lab_Test_1
- Lab_Test_2
- Lab_Test_3
- Past_history_Observation2
- Past_History_Observation_3
- Problem_List_Observation_1
- Problem_List_Observation_2
- Procedure_Observation_1
- Procedure_Observation_2
- ROS_Observation1
- RS_HPI_Observation1
- RS_HPI_Observation2
- RS_HPI_Observation3
- RS_PE_Observation

Types

Figure 4: Classification of Instances Under Defined Classes

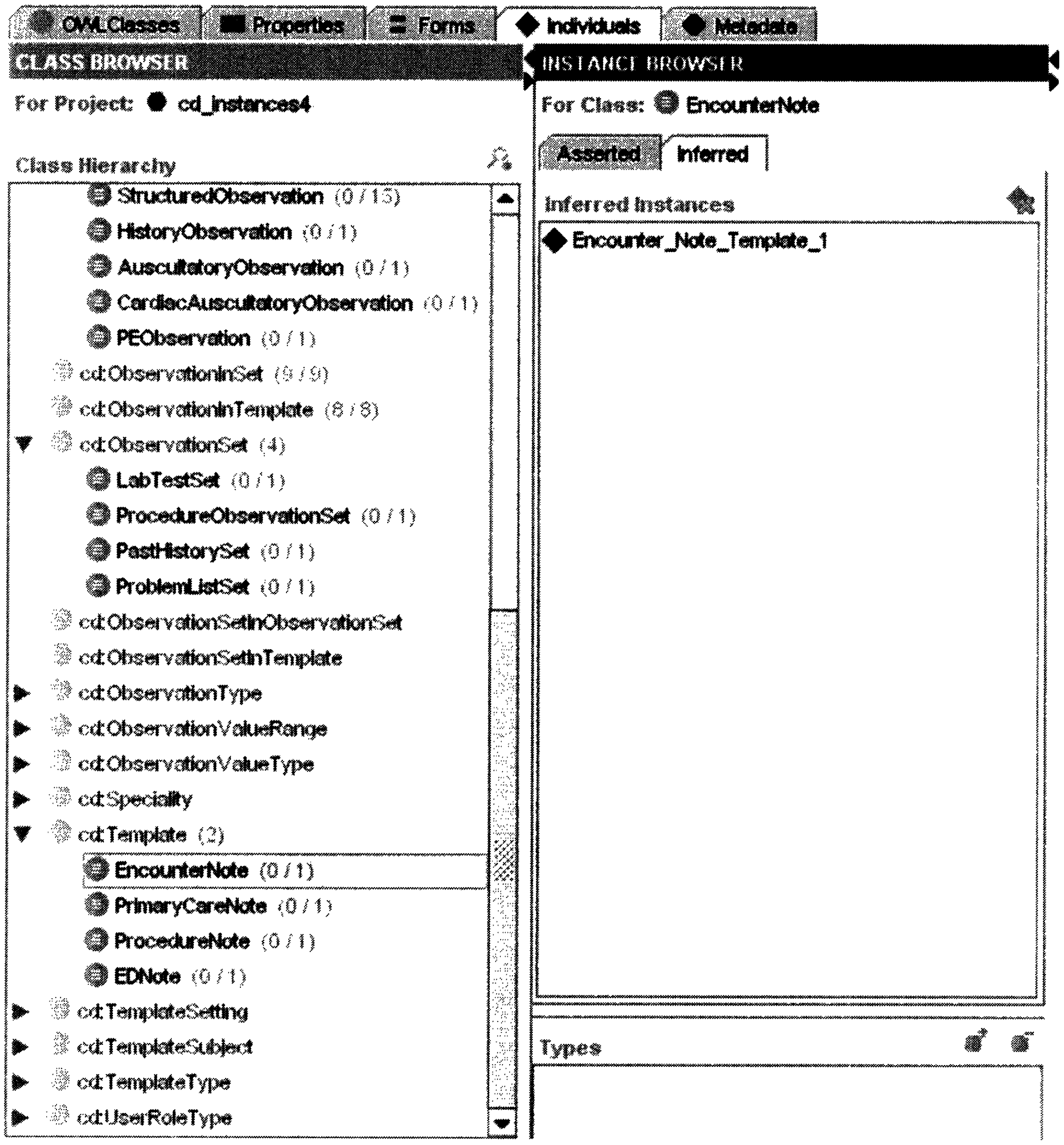


Figure 5: Classification of Instances Under Defined Classes

4.5 Display of Different Views

The ontology is able to classify observation instances into different "bins" based on the logic embedded in the defined classes. This ability can be used to generate different views of a clinical document. Since this ability cannot be easily demonstrated in Protégé without building a special application, Figure 6 below is usd to depict a conceptual representation. This figure shows two different ways in which observations in a catalog can be organized and displayed on the screen.

Unorganized Observations	**View 1 ObservationsByBodySystem**	**View 2 ObservationsByType**
Bowel Sounds	**CVS**	**CC**
Skin Symptoms	Chest Pain Location	RS Chief Complaints
Chest Pain Location	Apical Impulse	
Cancer in the past	Systolic Murmur	**HPI**
Blood Glucose		Chest Pain Location
Apical Impulse	**RS**	Cough
Past Surgeries	Cough	Localized Paralysis
Abscess drainage	Breath Sounds	
Cough	RS Chief Complaints	**PH**
RS Chief Complaints		Cancer in the past
Size of Pelvic Mass	**CNS**	Past Surgeries
Hemoglobin	Localized Paralysis	
Breath Sounds		**ROS**
Systolic Murmur	**GI**	Skin Symptoms
Length of Incision	Bowel Sounds	
Localized Paralysis		**PE**
	GU	**Palpation**
	Size of Pelvic Mass	Apical Impulse
		Size of Pelvic Mass
		Auscultation
		Systolic Murmur
		Breath Sounds
		Bowel Sounds
		LabTest
		Blood Glucose
		Hemoglobin
		Procedure Observation
		Abscess drainage
		Length of Incision

Figure 6: Shows an unorganized observations catalog on the left. The observations are organized to fit into appropriate bins and then displayed in different ways as shown in View 1 and View 2

5 Discussion

5.1 Significance

5.1.1 Utility of Ontology for Flexible Displays of Data

The document ontology I have developed helps in building an observations catalog, observation sets catalog and a catalog of templates. It can be used also to create flexible displays of clinical information. Since observations are tagged with various properties, manipulation of observations can be done in different ways as required by the application. *Observations* can be classified based on (1) *Observation Type* (for example, *History* vs. *Examination* vs. *Laboratory Results*) (2) by *Body System* (3) by *Observation Modifiers* (such as *acuity scale*, *anatomic location*, *duration*, *time*, *laterality* etc (4) the data type of *Observation* - for example, *free text*, *coded data, numerical data*. This information can be used to render different displays of clinical data. It can also be used for classification/manipulation of observation catalogs. For example, if we want to just group together Cardiovascular (CVS) history and physical examination (PE) findings for viewing by a cardiologist, it should be possible to create a defined class that will pull into its bin all instances of Observations that are CVS history and PE findings. Alternatively, if we want only the *Past History* findings of a patient, this could be done similarly too.

The different properties associated with a *Template* such as its *Setting*, *Type*, *Subject*, *Specialty*, *User Role* can be used to generate the Header or name of the Template. For example, if the *Template Setting* is *Outpatient*, *Type* is *Consultation Note*, *Subject* is *Dyisfunctional Uterine Bleeding* and *Specialty* is *Obstetrics*, a header for the template can be generated of the following kind "DUB, OB, Outpatient". It could also be possible that different sections of the template are used by different care providers, for example, *Vital Signs* may be documented by the nursing staff. In which case the template can be given two or more User Roles or different sections of the document may be made editable by different Users.

In addition, if the clinical document needs to be partitioned into sub-sections, it should be possible to do so. For example, to group together all observations of *Type HPI* (History of Present Illness) and put them under a header called *HPI* (bold it and increase the font) and have headers like this for *Social History*, *ROS*, *General Examination*, *Respiratory*

Examination, *Path Statement Section* to name a few. Alternatively, it is possible to have all the observations irrespective of type to be grouped together in a paragraph without any headers. Hence, this model makes for flexible display and manipulation of data to be represented in different clinical documents.

5.1.2 Utility of Ontology as an Organizational Tool

This ontology can be used to classify observations into different bins based on the attributes assigned to an *Observation* and define different kinds of *ObservationSets*. For example, we could have observations like- *no eye opening, eye opening to pain, no verbal response, no motor response, extension to pain* etc. which can exist as stand alone observations in a document or maybe spread out in a document and be used in different observation sets. However, these observations when grouped together in a single set or made to appear consecutively in a template can be used as the basis of the Glasgow Coma Scale. Alternatively, there could be observations that are grouped together by virtue of their semantics. For, example Pain Scale observations are made to appear together. In addition, an observation set may contain all kinds of detailed questions about the *nature of cough, sputum production, color of sputum*, which the user is required to see only if the patient has cough in the first place.

5.2 Limitations

- This ontology is created as a prototype to explore the application of this technology for managing observations technology. In its current state, the ontology is not complete or exhaustive. For example, it does not stipulate observations that are required in a particular template or need to be suppressed in certain observations. *ObservationSets* are not tagged with classifiable attributes. *ObservationSets* can be classified only based on the *Observations* they contain. The ontology may not be usable by specific users whose needs are not met by it.

- Not every single application of the ontology could be validated. A complete robust testing of all possible display views of observations or all possible groupings of observations within sets was not tried. It is quite possible that the ontology does not support every possible representation/mix of observations within sets or sets within templates.

- Quantitative reasoning is not supported well in OWL. Therefore, it is difficult to model a scenario where current history findings become past history observations after 'x' amount of time.

5.2.1 Future Directions:

- One application of this ontology, with further refinement is in the interchange of pre-created medical content across different versions of the same EMR or across different EMRs by mapping the content of different clinical documentation systems on to this ontology. This helps in the sharing of clinical data and the re-usability of content across different versions of a product or across different systems/schemas. For example, if a clinical document built in one system needs to be used by another system, the content of the clinical document to be transferred can be first broken down and mapped onto the key classes in the ontology. Once the *Observations* catalog and *ObservationSets* catalog have been populated in the ontology, this content can then be mapped into the corresponding clinical document schema of the recipient clinical information system.
- Perform a more robust testing of all aspects of the ontology. For example having the same observations within a document displayed in two or three different ways, probably within different observation sets.
- Evaluation and Management (E&M) codes [28] are billing codes that are used in the process of reimbursement to physicians for patient visits. depending on the amount of time a physician has spent with the patient and the level of examination documented. E&M Coding follows certain rules such as the level of detail of the examination, whether the required number of subjective and objective findings was documented. These rules for E&M coding can be written within the ontology such that the correct numeric points are assigned to observations or list values when new observations or lists are created.
- Develop the ability to pull in the same observations or sets into different kinds of documents. For example, a patient going in for an ultrasound or biopsy requires relevant history and physical examination documented in the ultrasound or biopsy report. Or a patient going in for an obstetric scan might require all obstetric related

observations displayed in the scan report. So, my ontology can be used to pull in and display relevant information in different documents.

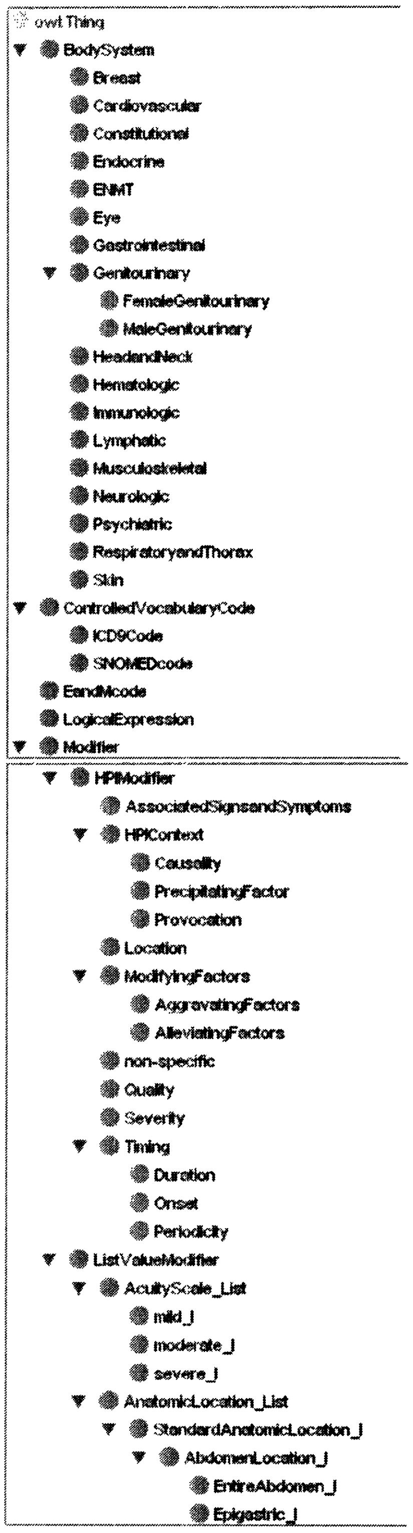
References

1. Kuhn K, Gaus W, Wechsler JG, et al. Structured reporting of medical findings: evaluation of a system in gastroenterology. *Methods Inf Med* 1992;31:268-274.
2. Bell DS, Greenes RA. Evaluation of UltraSTAR: performance of a collaborative structured data entry system. In: Ozbolt JG, ed. *Proceedings of the 18th Annual Symposium on Computer Applications in Medical Care*. Philadelphia: Hanley-Belfus, 1994: 216-221.
3. Campbell KE, Wieckert K, Fagan LM, Musen MA. A computer-based tool for generation of progress notes. In: Safran C, ed. *Proceedings of the 17th Annual Symposium on Computer Applications in Medical Care*. New York: McGraw-Hill, Inc., 1993: 284- 288.
4. Kuhn K, Zemmler T, Reichert M, Rosner D, Baumiller O, Knapp H. An integrated knowledge-based system to guide the physician during structured reporting. *Methods Inf Med* 1994;33:417-422.
5. Lloyd D. GEHR Architecture Documents available at http://www.chime.ucl.ac.uk/work-areas/ehrs/EUCEN/gehr.htm. 1993. Accessed on 05-15-05.
6. Health Level 7. HL7 Web site. Available at: http://www.hl7.org. Accessed May 9, 2005.
7. HL7 RIM [www.hl7.org/libary/datamodel/RIM/modelpage_non.htm]
8. Dolin RH, Alschuler L, Beebe C, Biron PV. The HL7 Clinical Document Architecture. *J Am Med Inform Assoc.* 2001;8:552–569.
9. Dolin RH, Alschuler L, Boyer S, Beebe C. An update on HL7's XML-based document representation standards. Proc AMIA Annu Symp. 2000:190–4.
10. Dolin R, Alschuler L, Behlen F, Biron P, Boyer S, Essin D, Harding L, Lincoln T, Mattison J, Rishel W, Sokolowski R, Spinosa J, Williams J. HL7 Document Patient Record Architecture: an XML document architecture based on a shared information model. Fall AMIA, November 1999.
11. GEHR. UCL Chime. http://www.chime.ucl.ac.uk/work-areas/ehrs/GEHR/index.htm . Accessed on 05-15-05.
12. OpenEHR: University College London. http://www.openehr.org/ . Accessed on 04-20-05.
13. Tzelepis TZ, Tsiknakis M. Foundation for Research and Technology, Hellas. Design and Implementation of "Two-Level" Clinical Information Systems, Based on Archetypes. 2004. http://www.ics.forth.gr/eHealth/publications/papers/2004/icth2004.pdf
14. EHCR-SupA and CEN ENV. University College London. Accessed on 04-20-05. http://www.chime.ucl.ac.uk/work-areas/ehrs/EHCR-SupA/13606-1.htm
15. European Committee for Standardization, Technical Committee for Health Informatics (CEN TC251). Four-part EHCR Message Standard. Brussels, Belgium: CEN TC251. Publication CEN ENV 13606.
16. European Committee for Standardization, Technical Committee for Health Informatics (CEN TC251). Investigation of Syntaxes for Existing Interchange Formats to be Used in Healthcare. Brussels, Belgium: CEN TC251, Jan 1993.
17. van Ginneken AM, de Wilde M. A New Approach to Structured Data Collection. In: Waegemann CP, (ed). Proc of TEPR 2000. San Francisco: Med Rec Inst. 2000; 627–35.
18. Renske K. Los, Astrid M. Van Ginneken, Marcel De Wilde, Johan Van Der Lei. OpenSDE: Row Modeling Applied to Generic Structured Data Entry. J Am Med Inform Assoc. 2004;11:162–165. DOI 10.1197/jamia.M1375.

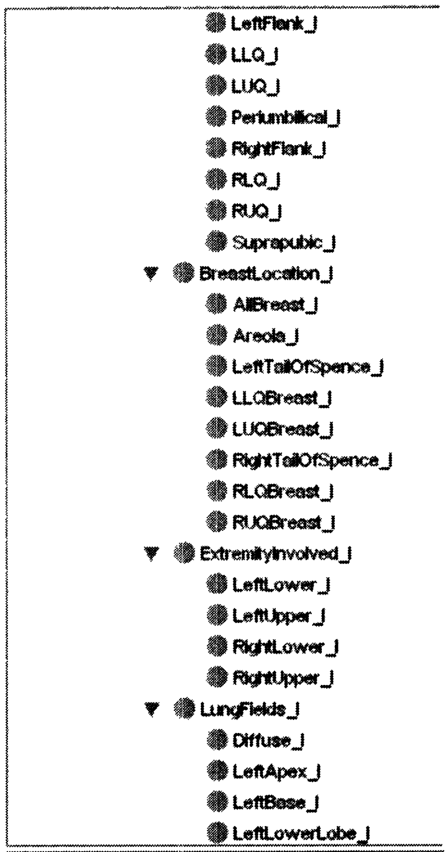
19. Astrid M. van Ginneken, Peter W. Moorman. Self-contained Patient Data in ORCA to Cope with an Evolving Vocabulary. Proc AMIA Symposium. 1998;:190-4.
20. A.M. van Ginneken, M. de Wilde, E.M. van Mulligen, H.Stam. Can Data Representation and Interface Demands be reconciled? Approach in ORCA. Proc AMIA Annual Fall Symposium. 1997;:779-83.
21. Astrid M. van Ginneken, Michiel J. Verkoijen. A Multi-Disciplinary Approach to a User Interface for Structured Data Entry. Medinfo. 2001;10(Pt 1):693-7.
22. R. Möller and V. Haarslev. Description Logic Systems. In F. Baader and D. Nardi P. Patel-Schneider D. Calvanese, D. McGuinness, editors, *The Description Logic Handbook*, chapter 8, pages 282–305. Cambridge University Press, 2003.
23. N. F. Noy, R. W. Fergerson, & M. A. Musen. The knowledge model of Protege-2000: Combining interoperability and flexibility. 2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, France, . 2000. Protégé (2000). The Protege Project. http://protege.stanford.edu
24. T. Gruber. *A Translation Approach to Portable Ontology Specifications*. Technical Report KSL 92-71. Knowledge Systems Laboratory, Stanford University, Revised April 1993.
25. F. Baader, I. Horrocks, and U. Sattler. *Description logics as ontology languages for the semantic web*. In D. Hutter and W. Stephan, editors, Festschrift in honor of Jorg Siekmann. LNAI. Springer-Verlag, 2003.
26. Horridge M, Knublauch H, Rector A, Stevens R, Wroe C. A Practical Guide To Building OWL Ontologies Using The Prot´eg´e-OWL Plugin and CO-ODE Tools. The University of Manchester. http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf. Accessed on 07/15/05.
27. Spackman KA, Campbell KE, Cote RA. SNOMED RT: a reference terminology for health care. Proc AMIA Annu Fall Symp. 1997;:640-4.
28. Evaluation and Management (E&M) codes http://www.hospitalist.net/e&mcodes1.htm. Accessed on 07/15/05.

6 Appendices

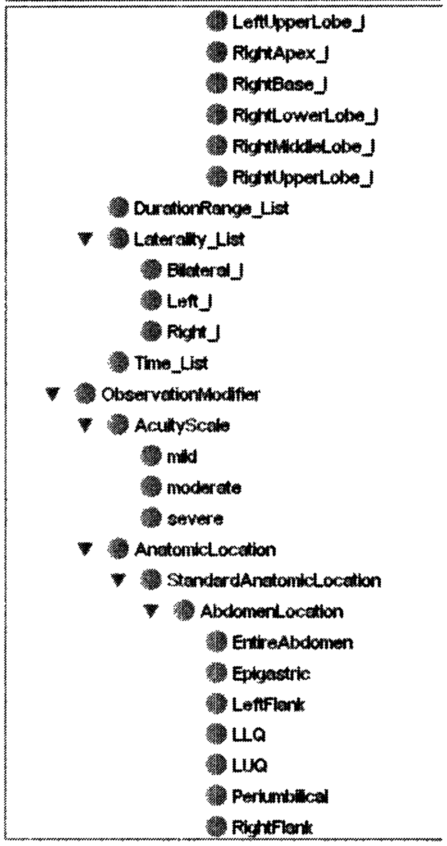
6.1.1 Appendix 1: Ontology - Asserted Hierarchy: Exploded Tree



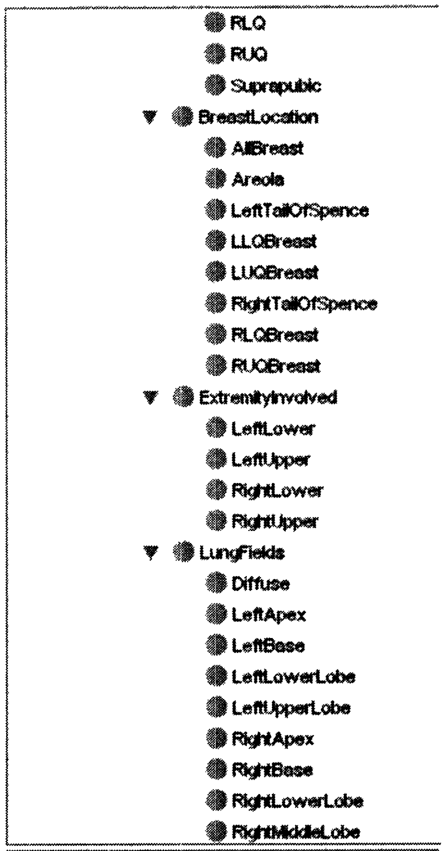
```
owl:Thing
▼ BodySystem
    Breast
    Cardiovascular
    Constitutional
    Endocrine
    ENMT
    Eye
    Gastrointestinal
  ▼ Genitourinary
      FemaleGenitourinary
      MaleGenitourinary
    HeadandNeck
    Hematologic
    Immunologic
    Lymphatic
    Musculoskeletal
    Neurologic
    Psychiatric
    RespiratoryandThorax
    Skin
▼ ControlledVocabularyCode
    ICD9Code
    SNOMEDcode
  EandMcode
  LogicalExpression
▼ Modifier
  ▼ HPIModifier
      AssociatedSignsandSymptoms
    ▼ HPIContext
        Causality
        PrecipitatingFactor
        Provocation
      Location
    ▼ ModifyingFactors
        AggravatingFactors
        AlleviatingFactors
      non-specific
      Quality
      Severity
    ▼ Timing
        Duration
        Onset
        Periodicity
  ▼ ListValueModifier
    ▼ AcuityScale_List
        mild_I
        moderate_I
        severe_I
    ▼ AnatomicLocation_List
      ▼ StandardAnatomicLocation_I
        ▼ AbdomenLocation_I
            EntireAbdomen_I
            Epigastric_I
```



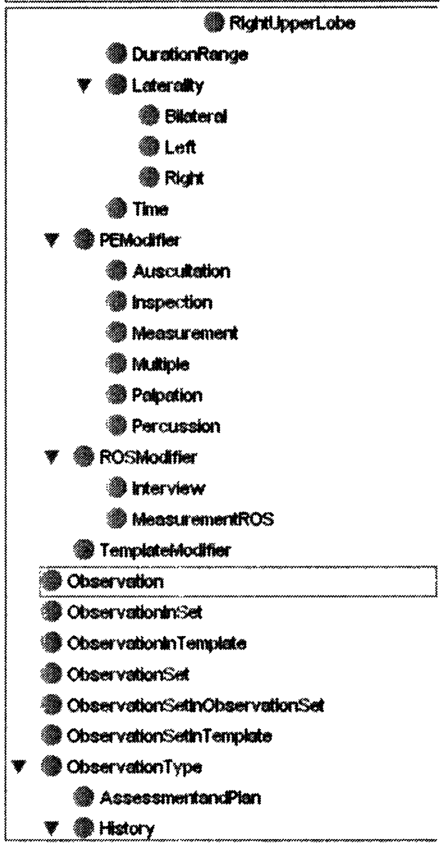
```
              ● LeftFlank_l
              ● LLQ_l
              ● LUQ_l
              ● Periumbilical_l
              ● RightFlank_l
              ● RLQ_l
              ● RUQ_l
              ● Suprapubic_l
          ▼ ● BreastLocation_l
              ● AllBreast_l
              ● Areola_l
              ● LeftTailOfSpence_l
              ● LLQBreast_l
              ● LUQBreast_l
              ● RightTailOfSpence_l
              ● RLQBreast_l
              ● RUQBreast_l
          ▼ ● ExtremityInvolved_l
              ● LeftLower_l
              ● LeftUpper_l
              ● RightLower_l
              ● RightUpper_l
          ▼ ● LungFields_l
              ● Diffuse_l
              ● LeftApex_l
              ● LeftBase_l
              ● LeftLowerLobe_l
```



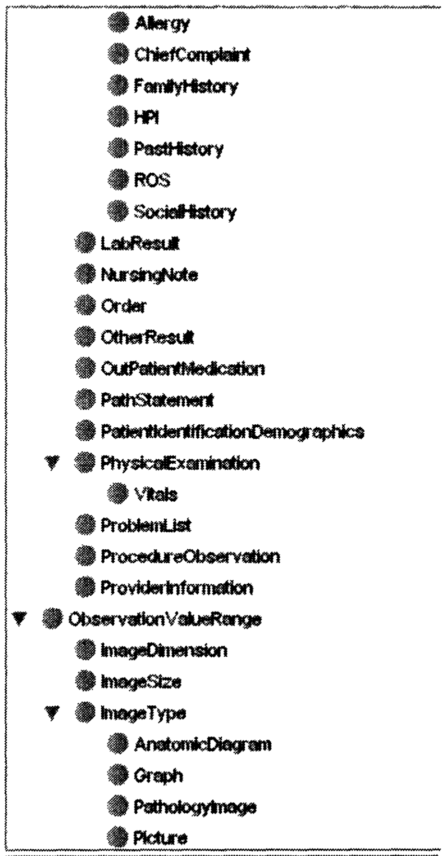
```
              ● LeftUpperLobe_l
              ● RightApex_l
              ● RightBase_l
              ● RightLowerLobe_l
              ● RightMiddleLobe_l
              ● RightUpperLobe_l
      ● DurationRange_List
    ▼ ● Laterality_List
          ● Bilateral_l
          ● Left_l
          ● Right_l
      ● Time_List
▼ ● ObservationModifier
    ▼ ● AcuityScale
          ● mild
          ● moderate
          ● severe
    ▼ ● AnatomicLocation
        ▼ ● StandardAnatomicLocation
            ▼ ● AbdomenLocation
                  ● EntireAbdomen
                  ● Epigastric
                  ● LeftFlank
                  ● LLQ
                  ● LUQ
                  ● Periumbilical
                  ● RightFlank
```



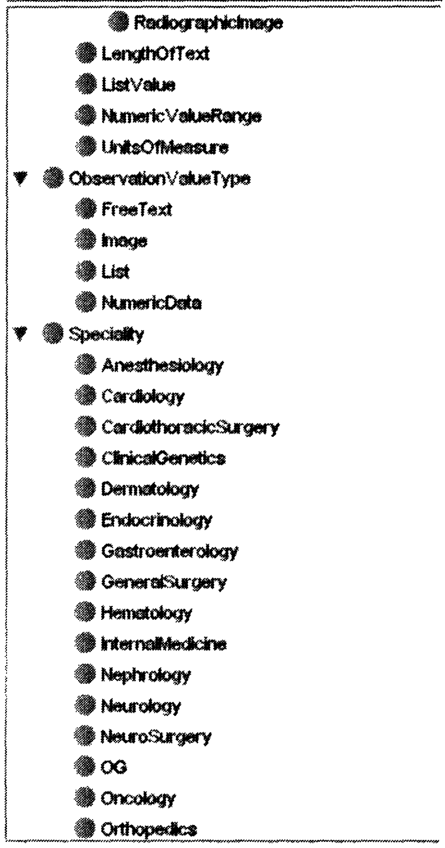
```
                ● RLQ
                ● RUQ
                ● Suprapubic
          ▼ ● BreastLocation
                ● AllBreast
                ● Areola
                ● LeftTailOfSpence
                ● LLQBreast
                ● LUQBreast
                ● RightTailOfSpence
                ● RLQBreast
                ● RUQBreast
          ▼ ● ExtremityInvolved
                ● LeftLower
                ● LeftUpper
                ● RightLower
                ● RightUpper
          ▼ ● LungFields
                ● Diffuse
                ● LeftApex
                ● LeftBase
                ● LeftLowerLobe
                ● LeftUpperLobe
                ● RightApex
                ● RightBase
                ● RightLowerLobe
                ● RightMiddleLobe
```



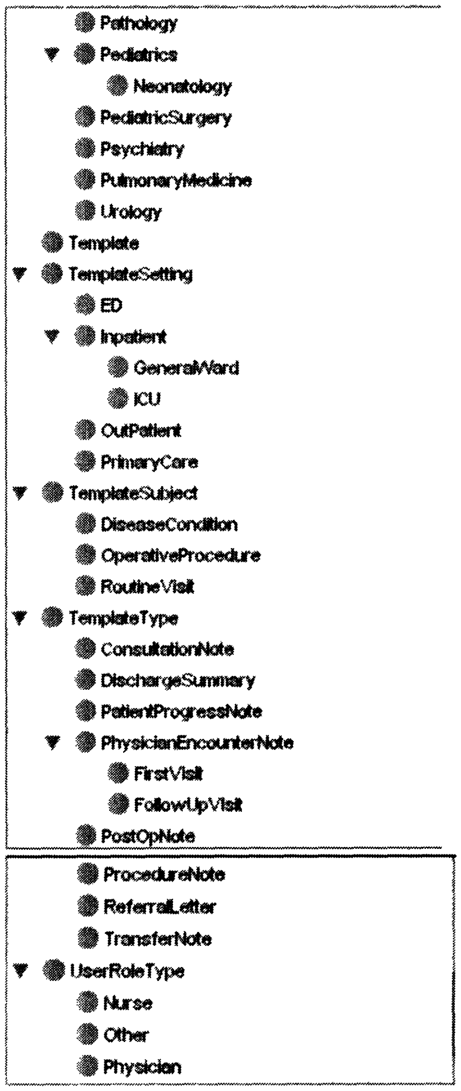
```
                ● RightUpperLobe
        ● DurationRange
      ▼ ● Laterality
            ● Bilateral
            ● Left
            ● Right
        ● Time
  ▼ ● PEModifier
        ● Auscultation
        ● Inspection
        ● Measurement
        ● Multiple
        ● Palpation
        ● Percussion
  ▼ ● ROSModifier
        ● Interview
        ● MeasurementROS
    ● TemplateModifier
● Observation
● ObservationInSet
● ObservationInTemplate
● ObservationSet
● ObservationSetInObservationSet
● ObservationSetInTemplate
▼ ● ObservationType
    ● AssessmentandPlan
  ▼ ● History
```



```
            ● Allergy
            ● ChiefComplaint
            ● FamilyHistory
            ● HPI
            ● PastHistory
            ● ROS
            ● SocialHistory
        ● LabResult
        ● NursingNote
        ● Order
        ● OtherResult
        ● OutPatientMedication
        ● PathStatement
        ● PatientIdentificationDemographics
    ▼   ● PhysicalExamination
            ● Vitals
        ● ProblemList
        ● ProcedureObservation
        ● ProviderInformation
▼   ● ObservationValueRange
        ● ImageDimension
        ● ImageSize
    ▼   ● ImageType
            ● AnatomicDiagram
            ● Graph
            ● PathologyImage
            ● Picture
```



```
            ● RadiographicImage
        ● LengthOfText
        ● ListValue
        ● NumericValueRange
        ● UnitsOfMeasure
▼   ● ObservationValueType
        ● FreeText
        ● Image
        ● List
        ● NumericData
▼   ● Speciality
        ● Anesthesiology
        ● Cardiology
        ● CardiothoracicSurgery
        ● ClinicalGenetics
        ● Dermatology
        ● Endocrinology
        ● Gastroenterology
        ● GeneralSurgery
        ● Hematology
        ● InternalMedicine
        ● Nephrology
        ● Neurology
        ● NeuroSurgery
        ● OG
        ● Oncology
        ● Orthopedics
```



- Pathology
- ▼ Pediatrics
  - Neonatology
- PediatricSurgery
- Psychiatry
- PulmonaryMedicine
- Urology

Template

▼ TemplateSetting
- ED
- ▼ Inpatient
  - GeneralWard
  - ICU
- OutPatient
- PrimaryCare

▼ TemplateSubject
- DiseaseCondition
- OperativeProcedure
- RoutineVisit

▼ TemplateType
- ConsultationNote
- DischargeSummary
- PatientProgressNote
- ▼ PhysicianEncounterNote
  - FirstVisit
  - FollowUpVisit
- PostOpNote
- ProcedureNote
- ReferralLetter
- TransferNote

▼ UserRoleType
- Nurse
- Other
- Physician

6.1.2 Appendix 2: Object and Data type Properties

Table 2: Object Properties

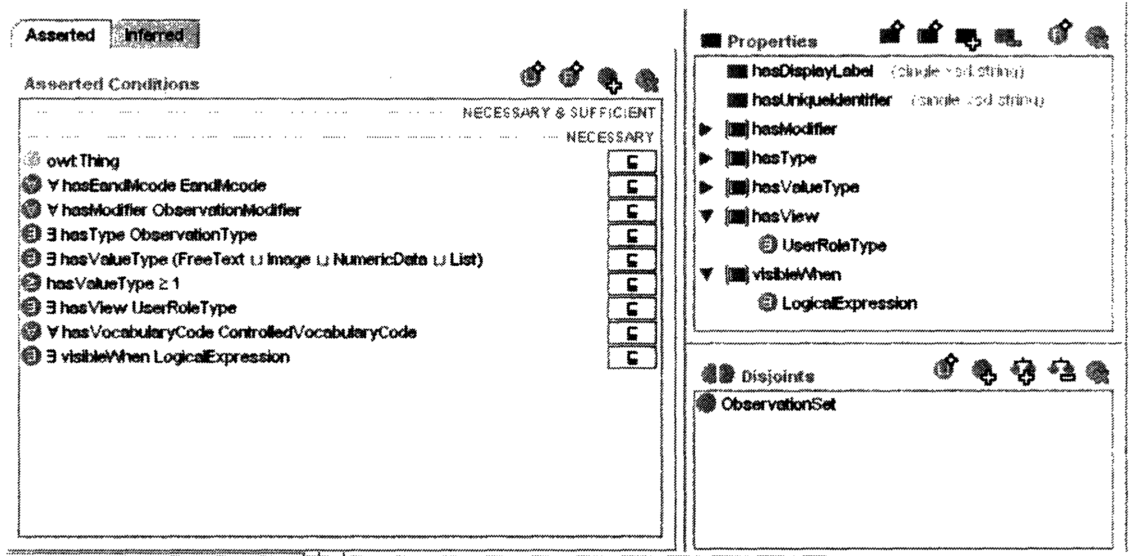
Object Property Name	Domain	Range
belongsTo	ObservationInSet	ObservationSet
	ObservationInTemplate	Template
	ObservationSetInObservationSet	ObservationSet
	ObservationSetInTemplate	Template
hasDimension	Image	ImageDimension
hasEandMcode	Observation	EandMcode
	List	EandMcode
hasImageType	Image	ImageType
hasModifier	Observation	ObservationModifier
	Template	Template Modifier
	List	ListValueModifier
hasPart	ObservationSet	ObservationInSet, ObservationSetInObservationSet
	Template	ObservationInTemplate, ObservationSetInTemplate
hasSetting	Template	TemplateSetting
hasSubject	PhysicianEncounterNote	TemplateSubject
	PostOpNote	TemplateSubject
hasType	Observation	ObservationType
	Template	TemplateType
hasUOM	NumericData	UnitsOfMeasure
hasValueRange	ObservationInSet	Observation
	ObservationInTemplate	Observation
	ObservationSetInObservationSet	ObservationSet
	ObservationSetInTemplate	ObservationSet
	ObservationValueType	ObservationValueRange
	Image	ImageSize
	Free Text	LengthOfText
	NumericData	NumericValueRange
	List	ListValue
hasValueType	Observation	FreeText, Image, NumericData, List
hasView	Observation	UserRoleType
hasVocabularyCode	Observation	ControlledVocabularyCode
	List	ControlledVocabularyCode
regardingBodySystem	History	BodySystem
	PhysicalExamination	BodySystem
usedBySpeciality	Template	Speciality
usedByUserType	Template	UserRoleType
visibleWhen	Observation	LogicalExpression
	ObservationSet	LogicalExpression

Table 3: Datatype Properties

Datatype Property Name	Domain	Range
evaluatesTo	LogicalExpression	xsd: boolean
hasAnnotationRange	AnatomicDiagram	xsd: string
hasDisplayLabel	Observation, ObservationSet	xsd: string
hasDuration	DurationRange	xsd: duration
hasLength	LengthOfText	xsd: int
hasNumericRange	NumericValueRange	xsd: float
hasSequenceNumber	ObservationInSet ObservationInTemplate ObservationSetInTemplate ObservationSetInObservationSet	xsd: int
hasSize	ImageSize	xsd: byte
hasSource	PathologyImage RadiographicImage Picture	xsd: anyURI
hasTemporality	Graph	xsd: boolean
hasUniqueIdentifier	Observation ObservationSet Template	xsd: string
hasVocabularyCodeName	ControlledVocabularyCode	xsd: string
hasVocabularyCodeNumber	ControlledVocabularyCode	xsd: string

6.1.3 Appendix 3: Asserted conditions widget of classes in the ontology

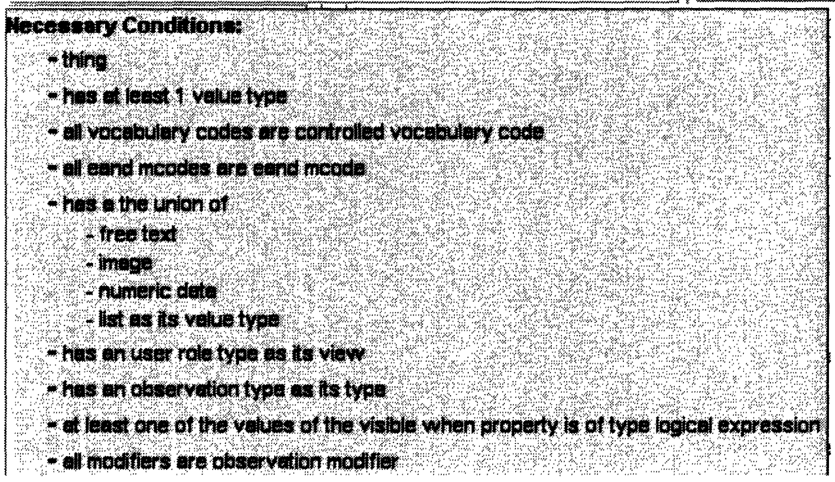
1. Observation



owl:Thing ⊑
∀ hasEandMcode EandMcode ⊑
∀ hasModifier ObservationModifier ⊑
∃ hasType ObservationType ⊑
∃ hasValueType (FreeText ⊔ Image ⊔ NumericData ⊔ List) ⊑
hasValueType ≥ 1 ⊑
∃ hasView UserRoleType ⊑
∀ hasVocabularyCode ControlledVocabularyCode ⊑
∃ visibleWhen LogicalExpression ⊑

hasDisplayLabel (single xsd:string)
hasUniqueIdentifier (single xsd:string)
▶ hasModifier
▶ hasType
▶ hasValueType
▼ hasView
  UserRoleType
▼ visibleWhen
  LogicalExpression

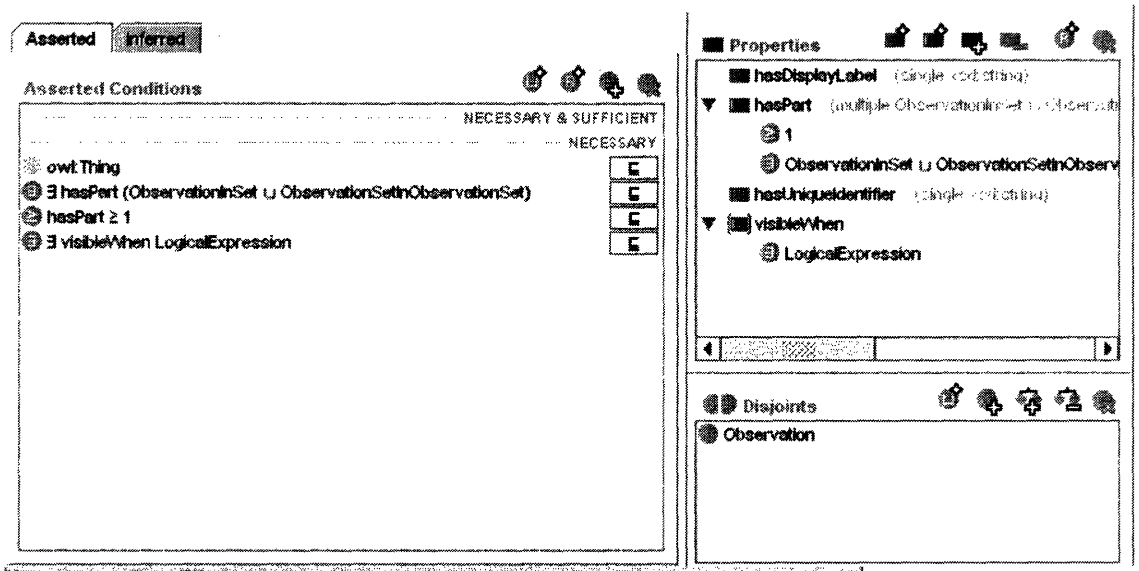
ObservationSet



Necessary Conditions:

- thing
- has at least 1 value type
- all vocabulary codes are controlled vocabulary code
- all eand mcodes are eand mcode
- has a the union of
  - free text
  - image
  - numeric data
  - list as its value type
- has an user role type as its view
- has an observation type as its type
- at least one of the values of the visible when property is of type logical expression
- all modifiers are observation modifier

2.ObservationSet



owl:Thing ⊑
∃ hasPart (ObservationInSet ⊔ ObservationSetInObservationSet) ⊑
hasPart ≥ 1 ⊑
∃ visibleWhen LogicalExpression ⊑

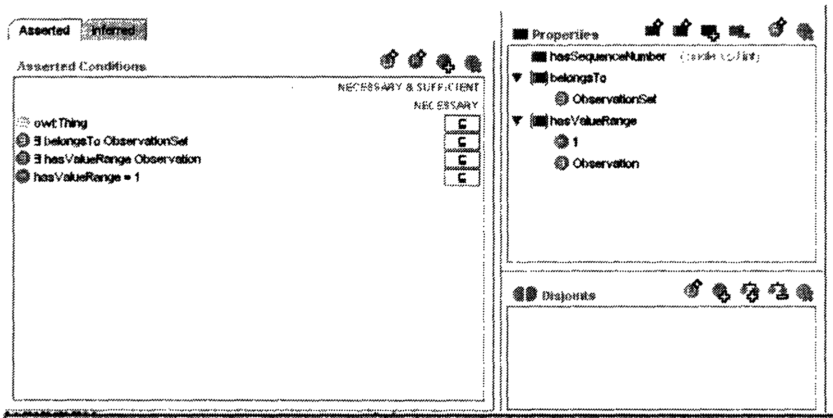
hasDisplayLabel (single xsd:string)
hasPart (multiple ObservationInSet ⊔ Observati
1
ObservationInSet ⊔ ObservationSetInObserv
hasUniqueIdentifier (single xsd:string)
visibleWhen
LogicalExpression

Observation

Necessary Conditions:

- has at least 1 part
- has a the union of
  - observation in set
  - observation set in observation set as its part
- thing
- at least one of the values of the visible when property is of type logical expression

3. ObservationInSet



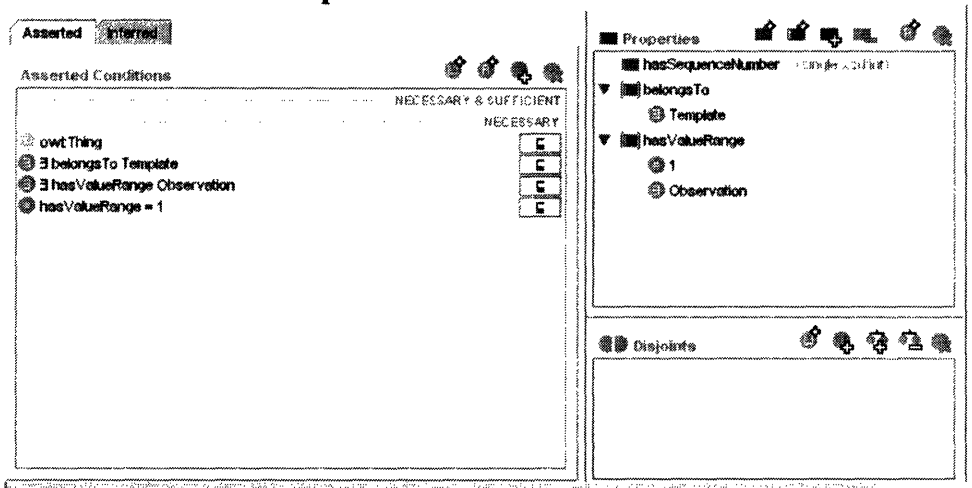
owl:Thing ⊑
∃ belongsTo ObservationSet ⊑
∃ hasValueRange Observation ⊑
hasValueRange = 1 ⊑

hasSequenceNumber (single xsd:int)
belongsTo
ObservationSet
hasValueRange
1
Observation

Necessary Conditions:

- at least one of the values of the belongs to property is of type observation set
- thing
- has exactly 1 value range
- has an observation as its value range

4. ObservationInTemplate



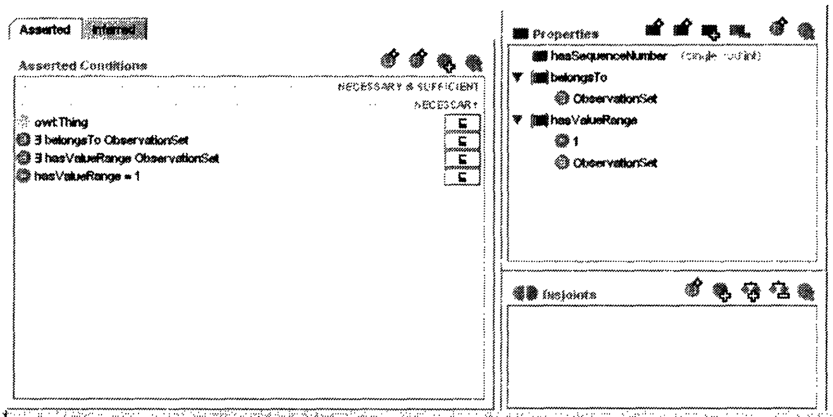
owl:Thing
∃ belongsTo Template
∃ hasValueRange Observation
hasValueRange = 1

hasSequenceNumber (single xsd:int)
belongsTo
Template
hasValueRange
1
Observation

Necessary Conditions:

- at least one of the values of the belongs to property is of type template
- has exactly 1 value range
- has an observation as its value range
- thing

5. ObservationSetInObservationSet



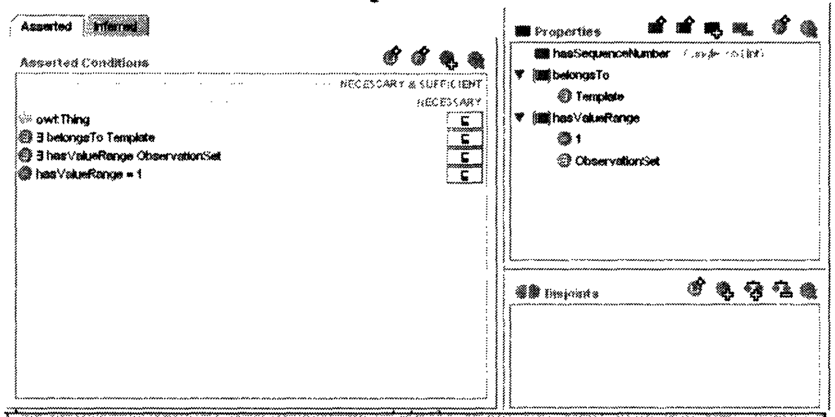
owl:Thing
∃ belongsTo ObservationSet
∃ hasValueRange ObservationSet
hasValueRange = 1

hasSequenceNumber (single xsd:int)
belongsTo
ObservationSet
hasValueRange
1
ObservationSet

Necessary Conditions:

- thing
- has exactly 1 value range
- has an observation set as its value range
- at least one of the values of the belongs to property is of type observation set

6. ObservationSetInTemplate



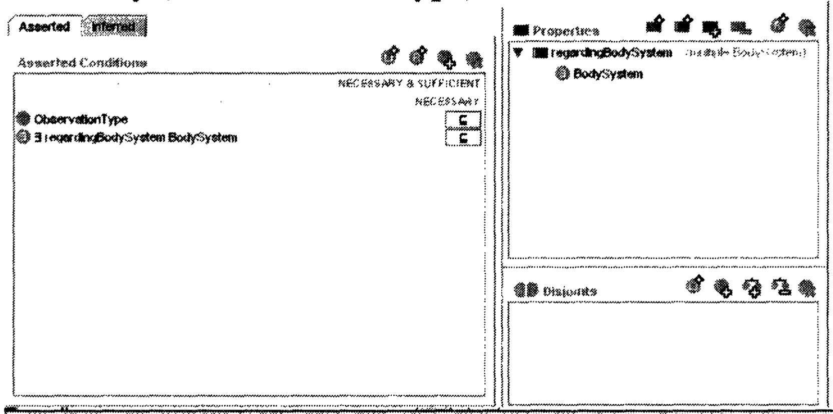
owl:Thing ⊑
∃ belongsTo Template ⊑
∃ hasValueRange ObservationSet ⊑
hasValueRange = 1 ⊑

hasSequenceNumber (single int)
belongsTo
Template
hasValueRange
1
ObservationSet

Necessary Conditions:

- at least one of the values of the belongs to property is of type template
- has exactly 1 value range
- has an observation set as its value range
- thing

7. History (in ObservationType)



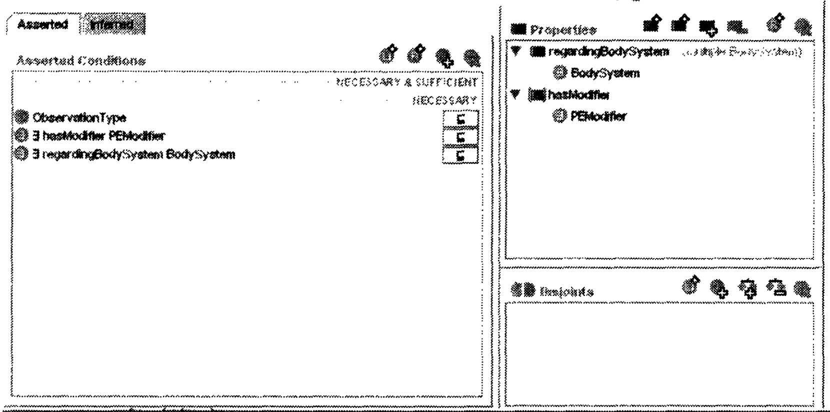
ObservationType ⊑
∃ regardingBodySystem BodySystem ⊑

regardingBodySystem (multiple BodySystem)
BodySystem

Necessary Conditions:

- observation type
- at least one of the values of the regarding body system property is of type body system

8. PhysicalExamination (in ObservationType)



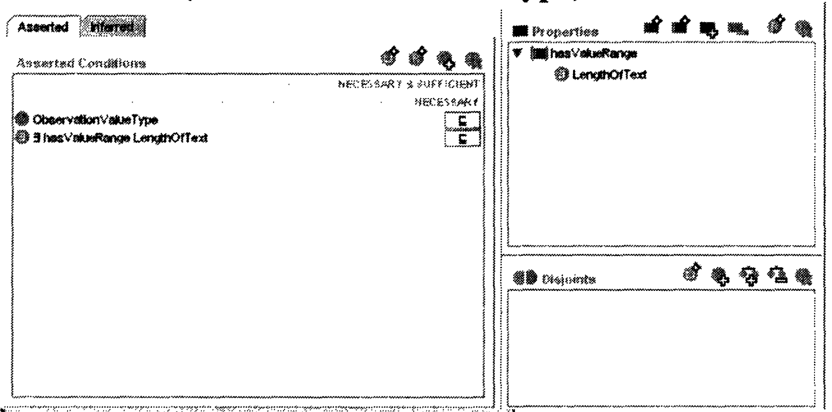
ObservationType ⊑
∃ hasModifier PEModifier ⊑
∃ regardingBodySystem BodySystem ⊑

▼ regardingBodySystem
BodySystem
▼ hasModifier
PEModifier

Necessary Conditions:

- has a PEModifier as its modifier
- observation type
- at least one of the values of the regarding body system property is of type body system

9. FreeText (in ObservationValueType)



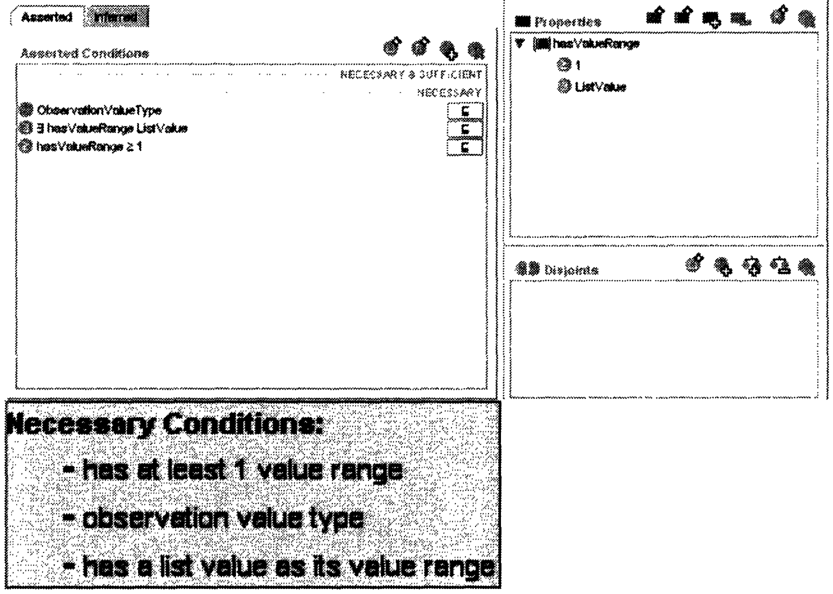
ObservationValueType ⊑
∃ hasValueRange LengthOfText ⊑

▼ hasValueRange
LengthOfText

Necessary Conditions:

- observation value type
- has a length of text as its value range

10. List (in ObservationValueType)



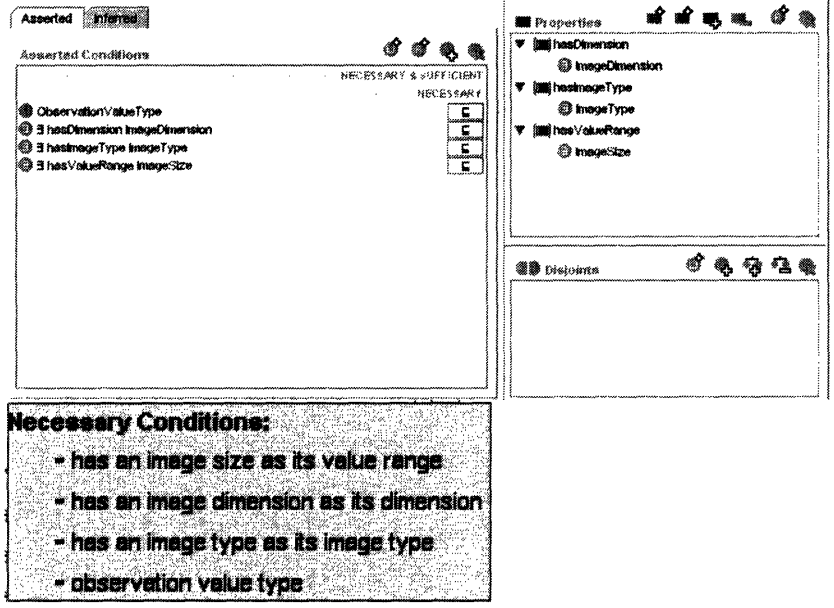
ObservationValueType ⊑
∃ hasValueRange ListValue ⊑
hasValueRange ≥ 1 ⊑

▼ hasValueRange
1
ListValue

Necessary Conditions:

- has at least 1 value range
- observation value type
- has a list value as its value range

11. Image (in ObservationValueType)



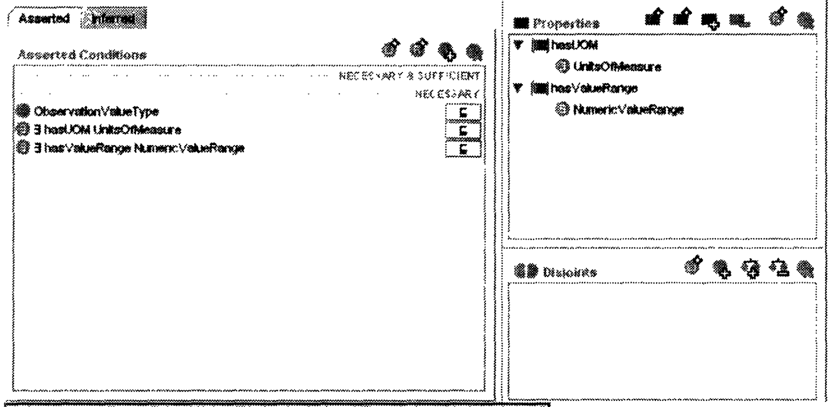
ObservationValueType ⊑
∃ hasDimension ImageDimension ⊑
∃ hasImageType ImageType ⊑
∃ hasValueRange ImageSize ⊑

▼ hasDimension
ImageDimension
▼ hasImageType
ImageType
▼ hasValueRange
ImageSize

Necessary Conditions:

- has an image size as its value range
- has an image dimension as its dimension
- has an image type as its image type
- observation value type

12. NumericData (in ObservationValueType)



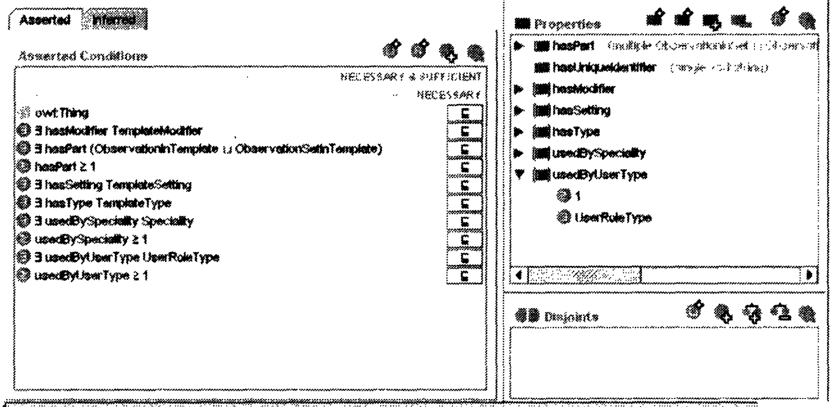
ObservationValueType ⊑
∃ hasUOM UnitsOfMeasure ⊑
∃ hasValueRange NumericValueRange ⊑

▼ hasUOM
  UnitsOfMeasure
▼ hasValueRange
  NumericValueRange

Necessary Conditions:

- has a numeric value range as its value range
- observation value type
- has an units of measure as its UOM

13. Template



owl:Thing ⊑
∃ hasModifier TemplateModifier ⊑
∃ hasPart (ObservationInTemplate ⊔ ObservationSetInTemplate) ⊑
hasPart ≥ 1 ⊑
∃ hasSetting TemplateSetting ⊑
∃ hasType TemplateType ⊑
∃ usedBySpecialty Specialty ⊑
usedBySpecialty ≥ 1 ⊑
∃ usedByUserType UserRoleType ⊑
usedByUserType ≥ 1 ⊑

▶ hasPart (multiple ObservationSet ⊔ Observati
hasUniqueIdentifier (single xsd:string)
▶ hasModifier
▶ hasSetting
▶ hasType
▶ usedBySpecialty
▼ usedByUserType
  1
  UserRoleType

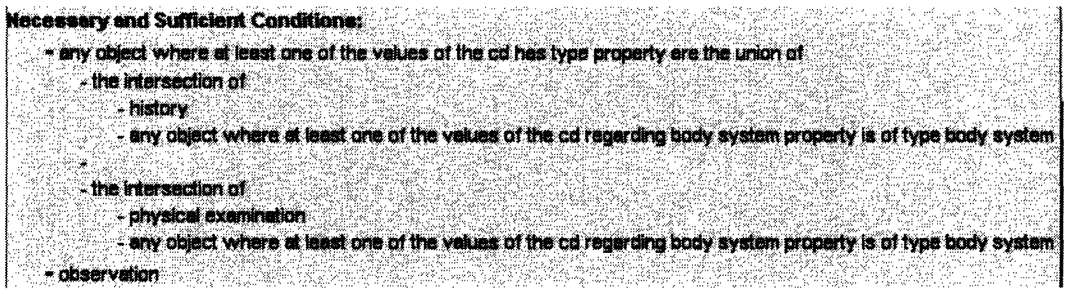
Necessary Conditions:

- has at least 1 part
- has a template setting as its setting
- used by user type has at least 1 value
- at least one of the values of the used by user type property is of type user role type
- thing
- used by specialty has at least 1 value
- has a the union of
    - observation in template
    - observation set in template as its part
- has a template modifier as its modifier
- has a template type as its type
- at least one of the values of the used by specialty property is of type specialty

6.1.4 Appendix 4: Defined Classes of Type OBSERVATION

1. AnyObservationByBodySystem

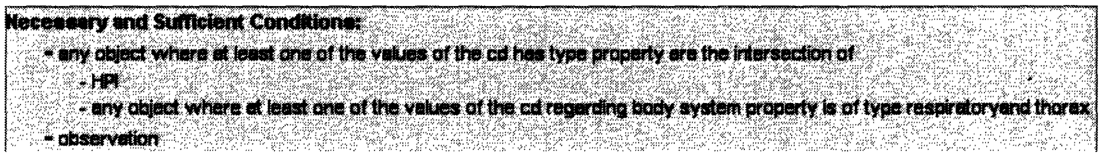
$\exists$ cd:hasType ((cd:History $\sqcap$ ($\exists$ cd:regardingBodySystem cd:BodySystem)) $\sqcup$ (cd:PhysicalExamination $\sqcap$ ($\exists$ cd:regardingBodySystem cd:BodySystem)))



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property are the union of
        - the intersection of
            - history
            - any object where at least one of the values of the cd regarding body system property is of type body system
        -
        - the intersection of
            - physical examination
            - any object where at least one of the values of the cd regarding body system property is of type body system
    - observation
```

2. RSHPIObservation

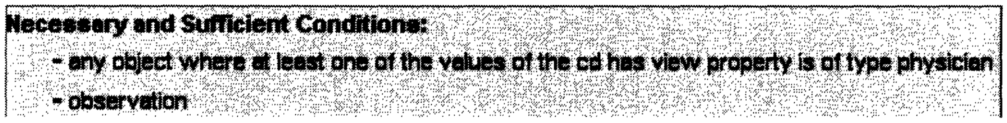
$\exists$ cd:hasType (cd:HPI $\sqcap$ ($\exists$ cd:regardingBodySystem cd:RespiratoryandThorax))



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property are the intersection of
        - HPI
        - any object where at least one of the values of the cd regarding body system property is of type respiratoryand thorax
    - observation
```

3. PhysicianViewObservation

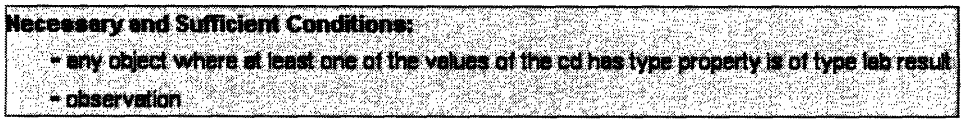
$\exists$ cd:hasView cd:Physician



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has view property is of type physician
    - observation
```

4. LabResultObservation

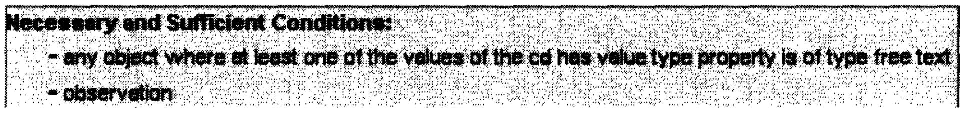
$\exists$ cd:hasType cd:LabResult



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property is of type lab result
    - observation
```

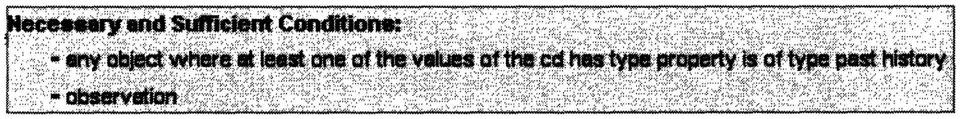
5. FreeTextObservation

∃ cd:hasValueType cd:FreeText


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has value type property is of type free text
- observation

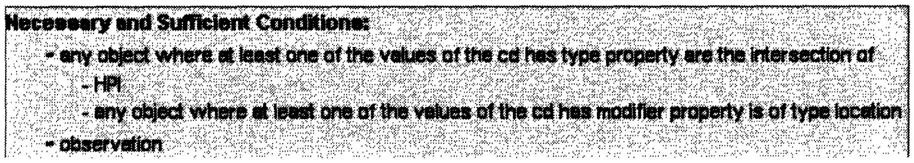
6. PastHistoryObservation

∃ cd:hasType cd:PastHistory


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property is of type past history
- observation

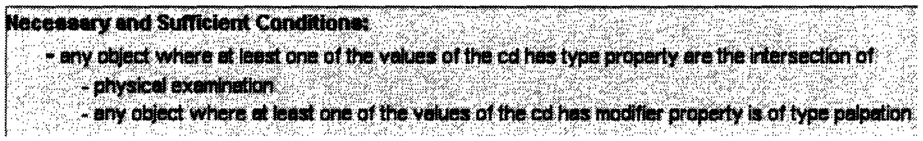
7. HPILocationObservation

∃ cd:hasType (cd:HPI ⊓ (∃ cd:hasModifier cd:Location))


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property are the intersection of
  - HPI
  - any object where at least one of the values of the cd has modifier property is of type location
- observation

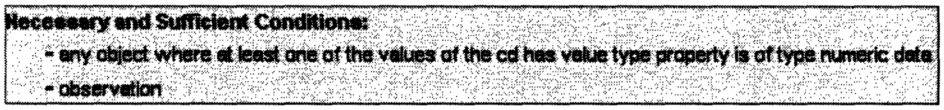
8. PalpatoryObservation

∃ cd:hasType (cd:PhysicalExamination ⊓ (∃ cd:hasModifier cd:Palpation))


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property are the intersection of
  - physical examination
  - any object where at least one of the values of the cd has modifier property is of type palpation

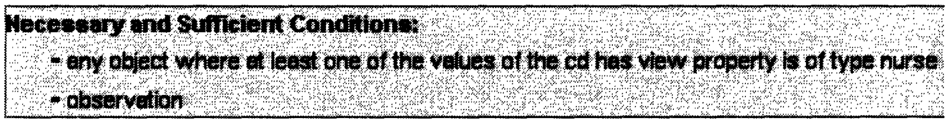
9. NumericDataObservation

$\exists$ cd:hasValueType cd:NumericData


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has value type property is of type numeric data
- observation

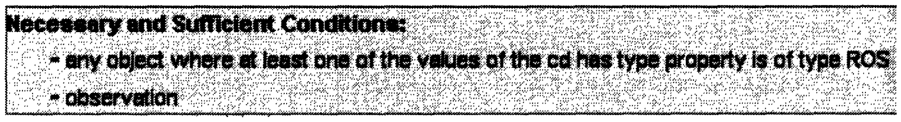
10. NurseViewObservation

$\exists$ cd:hasView cd:Nurse


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has view property is of type nurse
- observation

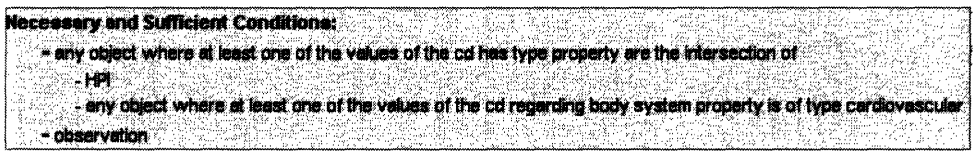
11. ROSObservation

$\exists$ cd:hasType cd:ROS


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property is of type ROS
- observation

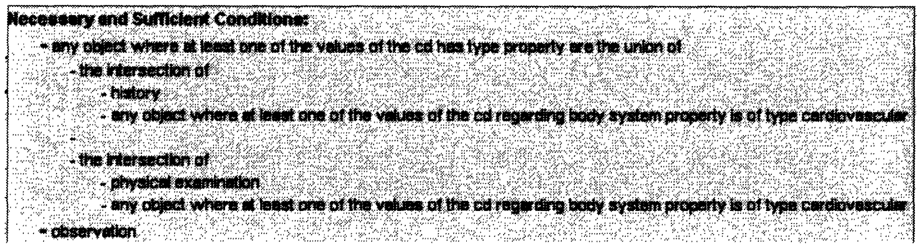
12. CVSHPIObservation

$\exists$ cd:hasType (cd:HPI $\sqcap$ ($\exists$ cd:regardingBodySystem cd:Cardiovascular))


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property are the intersection of
    - HPI
    - any object where at least one of the values of the cd regarding body system property is of type cardiovascular
- observation

13. CVSObservation

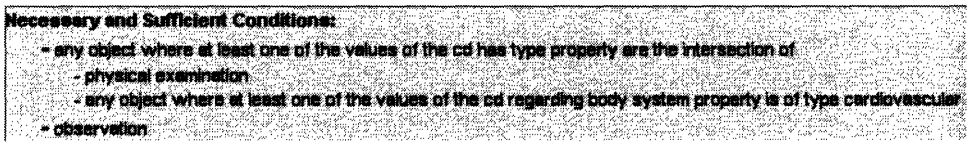
$\exists$ cd:hasType ((cd:History $\sqcap$ ($\exists$ cd:regardingBodySystem cd:Cardiovascular)) $\sqcup$ (cd:PhysicalExamination $\sqcap$ ($\exists$ cd:regardingBodySystem cd:Cardiovascular)))



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property are the union of
        - the intersection of
            - history
            - any object where at least one of the values of the cd regarding body system property is of type cardiovascular
        -
        - the intersection of
            - physical examination
            - any object where at least one of the values of the cd regarding body system property is of type cardiovascular
    - observation
```

14. CVSPEObservation

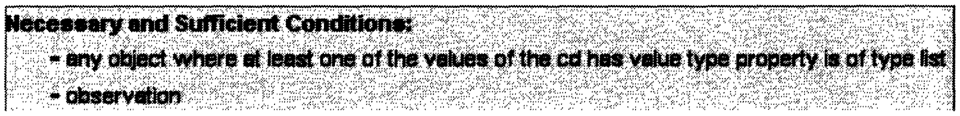
∃ cd:hasType (cd:PhysicalExamination ⊓ (∃ cd:regardingBodySystem cd:Cardiovascular))



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property are the intersection of
        - physical examination
        - any object where at least one of the values of the cd regarding body system property is of type cardiovascular
    - observation
```

15. StructuredObservation

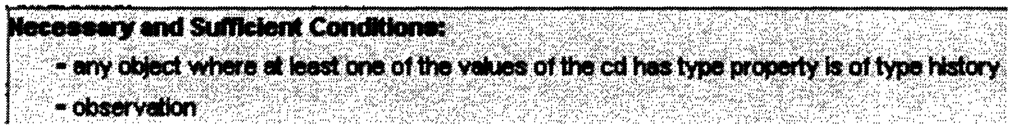
∃ cd:hasValueType cd:List



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has value type property is of type list
    - observation
```

16. HistoryObservation

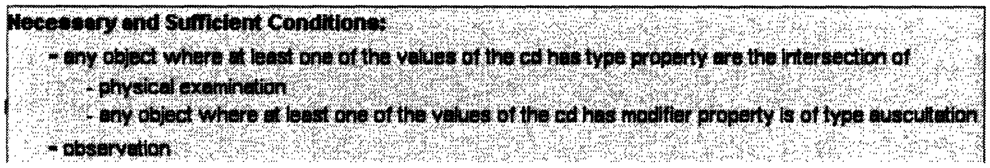
∃ cd:hasType cd:History



```
Necessary and Sufficient Conditions:
    - any object where at least one of the values of the cd has type property is of type history
    - observation
```

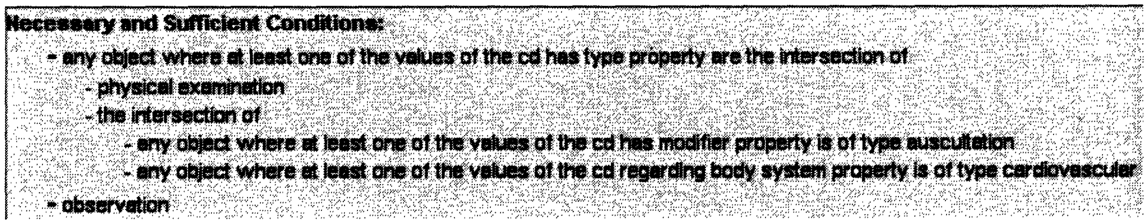
17. AuscultatoryObservation

∃ cd:hasType (cd:PhysicalExamination ⊓ (∃ cd:hasModifier cd:Auscultation))


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property are the intersection of
    - physical examination
    - any object where at least one of the values of the cd has modifier property is of type auscultation
- observation

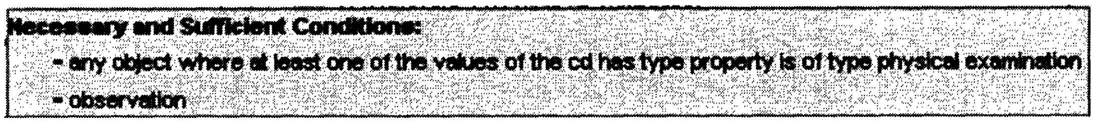
18. CardiacAuscultatoryObservation

∃ cd:hasType (cd:PhysicalExamination ⊓ ((∃ cd:hasModifier cd:Auscultation) ⊓ (∃ cd:regardingBodySystem cd:Cardiovascular)))


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property are the intersection of
    - physical examination
    - the intersection of
        - any object where at least one of the values of the cd has modifier property is of type auscultation
        - any object where at least one of the values of the cd regarding body system property is of type cardiovascular
- observation

19. PEObservation

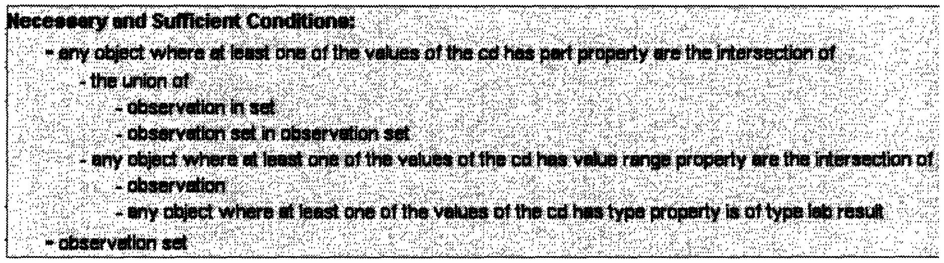
∃ cd:hasType cd:PhysicalExamination


Necessary and Sufficient Conditions:
- any object where at least one of the values of the cd has type property is of type physical examination
- observation

6.1.5 Appendix 5: Defined Classes of Type OBSERVATIONSET

1. LabTestSet

$\exists$ cd:hasPart ((cd:ObservationInSet $\sqcup$ cd:ObservationSetInObservationSet) $\sqcap$ ($\exists$ cd:hasValueRange (cd:Observation $\sqcap$ ($\exists$ cd:hasType cd:LabResult))))

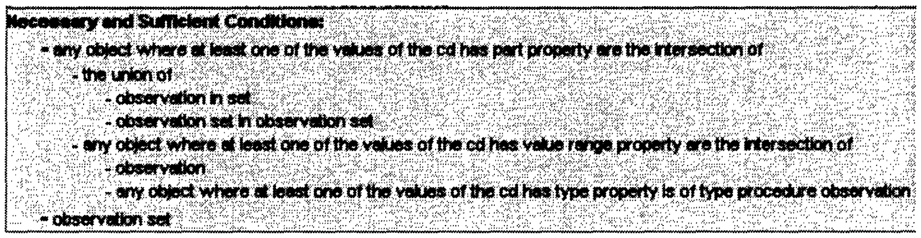


Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has part property are the intersection of
    - the union of
        - observation in set
        - observation set in observation set
    - any object where at least one of the values of the cd has value range property are the intersection of
        - observation
        - any object where at least one of the values of the cd has type property is of type lab result
- observation set

2. ProcedureObservationSet

$\exists$ cd:hasPart ((cd:ObservationInSet $\sqcup$ cd:ObservationSetInObservationSet) $\sqcap$ ($\exists$ cd:hasValueRange (cd:Observation $\sqcap$ ($\exists$ cd:hasType cd:ProcedureObservation))))

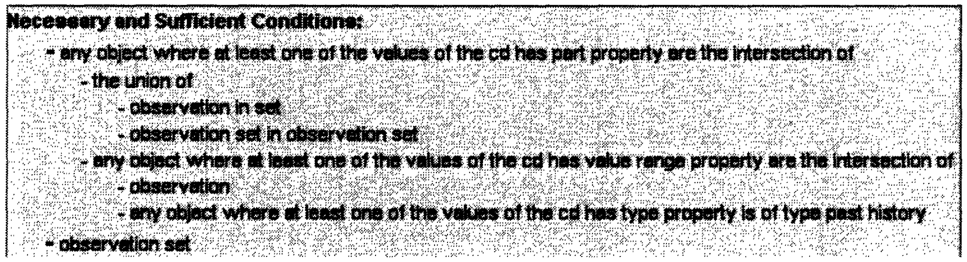


Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has part property are the intersection of
    - the union of
        - observation in set
        - observation set in observation set
    - any object where at least one of the values of the cd has value range property are the intersection of
        - observation
        - any object where at least one of the values of the cd has type property is of type procedure observation
- observation set

3. PastHistorySet

$\exists$ cd:hasPart ((cd:ObservationInSet $\sqcup$ cd:ObservationSetInObservationSet) $\sqcap$ ($\exists$ cd:hasValueRange (cd:Observation $\sqcap$ ($\exists$ cd:hasType cd:PastHistory))))

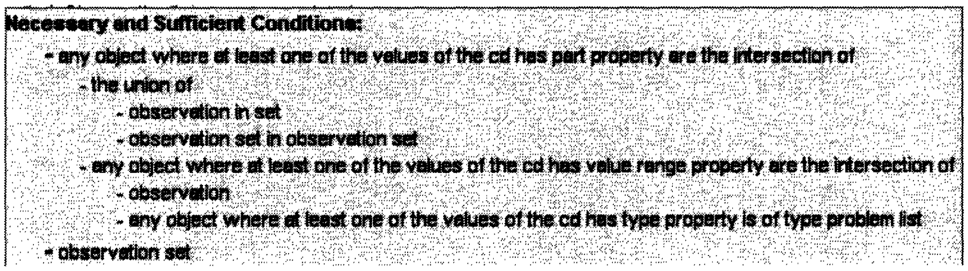


Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has part property are the intersection of
    - the union of
        - observation in set
        - observation set in observation set
    - any object where at least one of the values of the cd has value range property are the intersection of
        - observation
        - any object where at least one of the values of the cd has type property is of type past history
- observation set

4. ProblemListSet

$\exists$ cd:hasPart ((cd:ObservationInSet $\sqcup$ cd:ObservationSetInObservationSet) $\sqcap$ ($\exists$ cd:hasValueRange (cd:Observation $\sqcap$ ($\exists$ cd:hasType cd:ProblemList))))



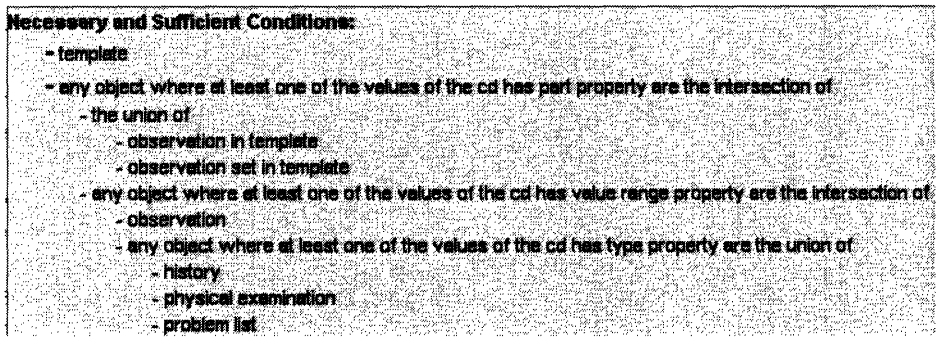
Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has part property are the intersection of
  - the union of
    - observation in set
    - observation set in observation set
  - any object where at least one of the values of the cd has value range property are the intersection of
    - observation
    - any object where at least one of the values of the cd has type property is of type problem list
- observation set

6.1.6 Appendix 6: Defined Classes of Type TEMPLATE

1. EncounterNote

∃ cd:hasPart ((cd:ObservationInTemplate ⊔ cd:ObservationSetInTemplate) ⊓ (∃ cd:hasValueRange (cd:Observation ⊓ (∃ cd:hasType (cd:History ⊔ cd:PhysicalExamination ⊔ cd:ProblemList)))))

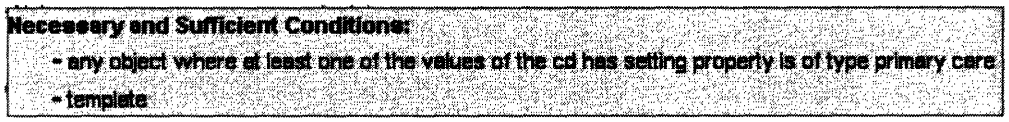


Necessary and Sufficient Conditions:

- template
- any object where at least one of the values of the cd has part property are the intersection of
  - the union of
    - observation in template
    - observation set in template
  - any object where at least one of the values of the cd has value range property are the intersection of
    - observation
    - any object where at least one of the values of the cd has type property are the union of
      - history
      - physical examination
      - problem list

2. PrimaryCareNote

∃ cd:hasSetting cd:PrimaryCare

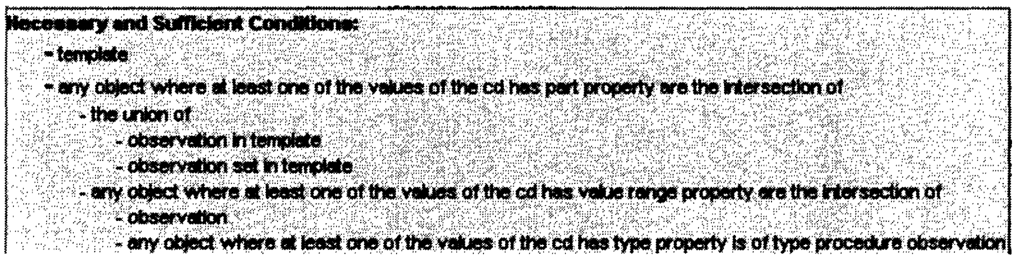


Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has setting property is of type primary care
- template

3. ProcedureNote

∃ cd:hasPart ((cd:ObservationInTemplate ⊔ cd:ObservationSetInTemplate) ⊓ (∃ cd:hasValueRange (cd:Observation ⊓ (∃ cd:hasType cd:ProcedureObservation))))

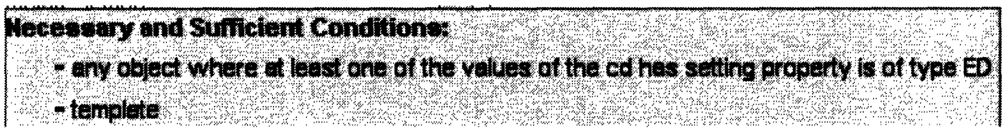


Necessary and Sufficient Conditions:

- template
- any object where at least one of the values of the cd has part property are the intersection of
  - the union of
    - observation in template
    - observation set in template
  - any object where at least one of the values of the cd has value range property are the intersection of
    - observation
    - any object where at least one of the values of the cd has type property is of type procedure observation

4. EDNote

∃ cd:hasSetting cd:ED



Necessary and Sufficient Conditions:

- any object where at least one of the values of the cd has setting property is of type ED
- template