A Software Engineering Approach to Ontology Modeling, Design, and Development with Lifecycle Process

Candidate: **Rishi Kanth Saripalle**

Major Advisor: **Prof. Steven A. Demurjian**

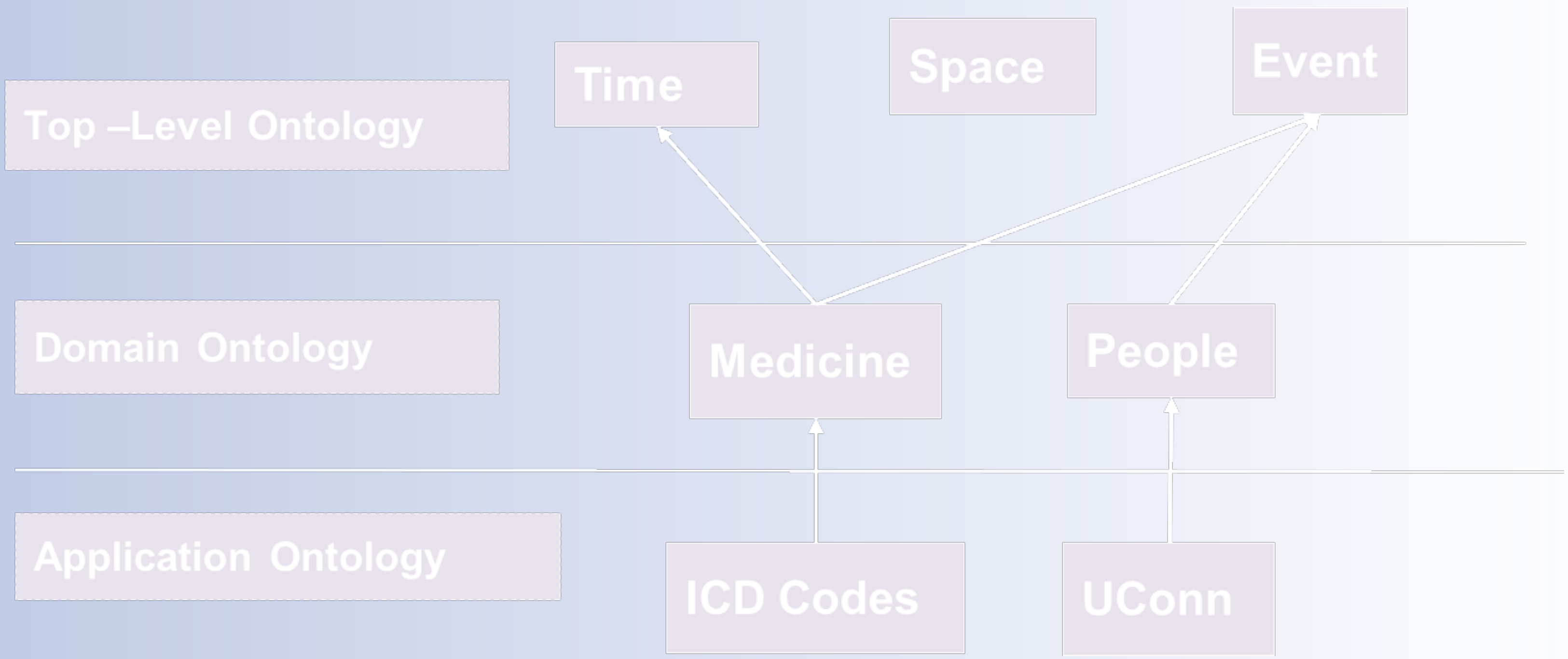
Associate Advisors: **Prof. Dong-Guk Shin**
Prof. Xiaoyan Wang
Prof. Michael Blechner

Ontologies

- The term Ontology is defined in:
  - *Philosophy* as particular system of categories accounting for a certain vision of the world.
  - *Computer science* as a possibly (complete or incomplete) consensus semantic agreement about a domain conceptualization
- In Abstract, Ontologies are Knowledge Containers, complied using Classes, Attributes and Associations
- Knowledge Represented can vary Based on the Domain, Application and User Requirements
- Ontologies Utilized in Health Care Systems to:
  - Represent Knowledge About the Data
  - Diagnosis, Treatment, Symptoms, Medications

How are Ontologies Categorized?

- In General, we can Categorize Ontologies into Three Types:
  - Top-Level Ontology – e.g. Time, Space, Event
  - Domain Ontology – e.g. Medicine, People.
  - Application Ontology – e.g. ICD, UConn



How are Ontologies Used in Computing?

- Attach Semantics to Digital Information → Converting into Knowledge
  - XML Concepts, HTML Documents, Database Records, etc.
- Represented in Formats
  - IS-A Hierarchy
  - Resource Definition Framework (RDF)
    - Represent Data in the form of Subject-Predicate-Object Expressions
  - Web ontology language (OWL)
    - Extends RDF with Description Features
    - Knowledge Representation Framework to capture Domain Knowledge
- Examples
  - Friend of a Friend (FOAF)
  - Foundational Model of Anatomy (FMA)

A Sample Ontology

- Anatomical entity
  - Physical anatomical entity
    - Material anatomical entity
      - Anatomical structure
        - Postnatal anatomical structure
          - Body
            - Body of vertebrate
          - Cardinal body part
          - Organ system
          - Subdivision of cardinal body part
          - Organ system subdivision
          - Organ
          - Cardinal organ part
            - Organ component
            - Region of organ component
            - Organ region
              - Organ segment
                - Segment of neuraxis
                  - Brain
                  - Segment of brain
                    - Cardinal segment of brain
                      - Forebrain
                      - Midbrain

- Sample Hierarchy from FMA
  - Human Anatomy Concepts Hierarchal Organized

How are Ontologies Used in BMI?

CSE 5810

- Preserve Semantics of the Clinical Information Encoded in Medical Records
  - Standard ontologies include UMLS, ICD, MeSH, SNOMED, and LONIC
- Intended to be Utilized in order to:
  - Structure and Semantics – digitalize clinical information in the form of HER, PHR , CCD, etc.
  - Share the information
    - Health Information Exchange (HIE) to Integrate Data
    - Virtual Chart (VC) to Present Integrated View to Users
- Proposed Standards Include I2B2, HL7 CDA R1, etc.
- Numerous EHRs, e.g., AllScripts, Centricity, Vista

What are the Issues with Ontology?

- Current Ontologies are:
  - Instance Based and Data Intensive
  - Developed for Specific Domain Applications
  - Can Represent Same Information in Conflicting Ways
- Current Ontology Frameworks/Tools are Non-Design Oriented
  - Construct a Specific Ontology for Particular Need
  - Difficult to Reuse Ontology in Different Setting
  - Difficult to Query Ontology form Inside Out
  - Tools (Protégé, Swoop, OntoStudio) Aren’t Design Oriented
- Ontologies Must be Able to be:
  - Designed Akin to Software or Database Design Process
  - Syntactically and Semantically Unified
  - Aim Towards Semi-Automated Integration Approach

Motivation

- In support of HIE and VC, Ontologies must be Integrated from Multiple sources
- Ontologies are Inherently:
  - Instance Based
  - Developed for Specific Applications
  - Can Represent same medical information on conflicting ways in different systems
    - Disease, Symptom, Treatment in one EMR
    - Symptom, Disease, Treatment in Another EMR
- Ontologies Must be Able to be:
  - Syntactically and Semantically Unified
  - Currently, a hands-on semi-automated approach
- Can Ontologies be More Design Oriented and Influenced by Software Engineering Models/Processes?

What Modeling Approaches to be Leveraged?

- UML is a de facto standard with
  - Diagrams- Class Diagram, Use-case Diagrams, Activity Diagrams, Sequence Diagrams
  - Provides profile extension mechanism to build domain specific metamodel elements,
  - Supports for design patterns that generalize and apply one template to many applications
- ER Diagrams entities can be transitioned to
  - Formal Relational Database Schema
  - Tables, Dependencies, Keys, Referential Integrity
- XML Focuses on Data Representation/Exchange with
  - Schema Definition for Structure
  - Schema Instances for Representation

What is Disconnect in Modeling?

- Ontologies are:
  - Application Based
  - Proceed from
    - Objects
    - Classes
    - Links
    - Hierarchy
  - Completely Data Focused
  - Bottom up Approach
  - Build a Specific Ontology for a Application
  - Inability to Reuse
  - Difficult to Integrate with one another
  - No Design Focus

- UML, ER:
  - Class/Type Based
  - Construct design artifacts:
    - Entities
    - Schemas
    - Classes
    - Relationships
    - Patterns
  - Top-Down Approach
  - Solution can apply to multiple applications
  - Emphasize Reuse
  - Components Interact with one another
  - Predominate Design Focus

What Problems are We Trying to Solve?

- How Can We Define Abstract Solutions for Ontologies?
  - Develop Abstract Solutions at Various Levels
    - Software Concepts: Meta-Models, Domain Models, Design Patterns, etc.
  - Reusable Across Multiple Domain Applications
- How to Extend Ontology Tools with Design Oriented Concepts?
  - Ability to Develop Reusable Abstract Solutions
  - Syntactically and Semantically be Unified
- Can we Develop a Software Development Process for Ontologies?
  - Top-Bottom Design and Development Strategy
  - Development Process to Design Ontologies Similar to Software or Database Design Process

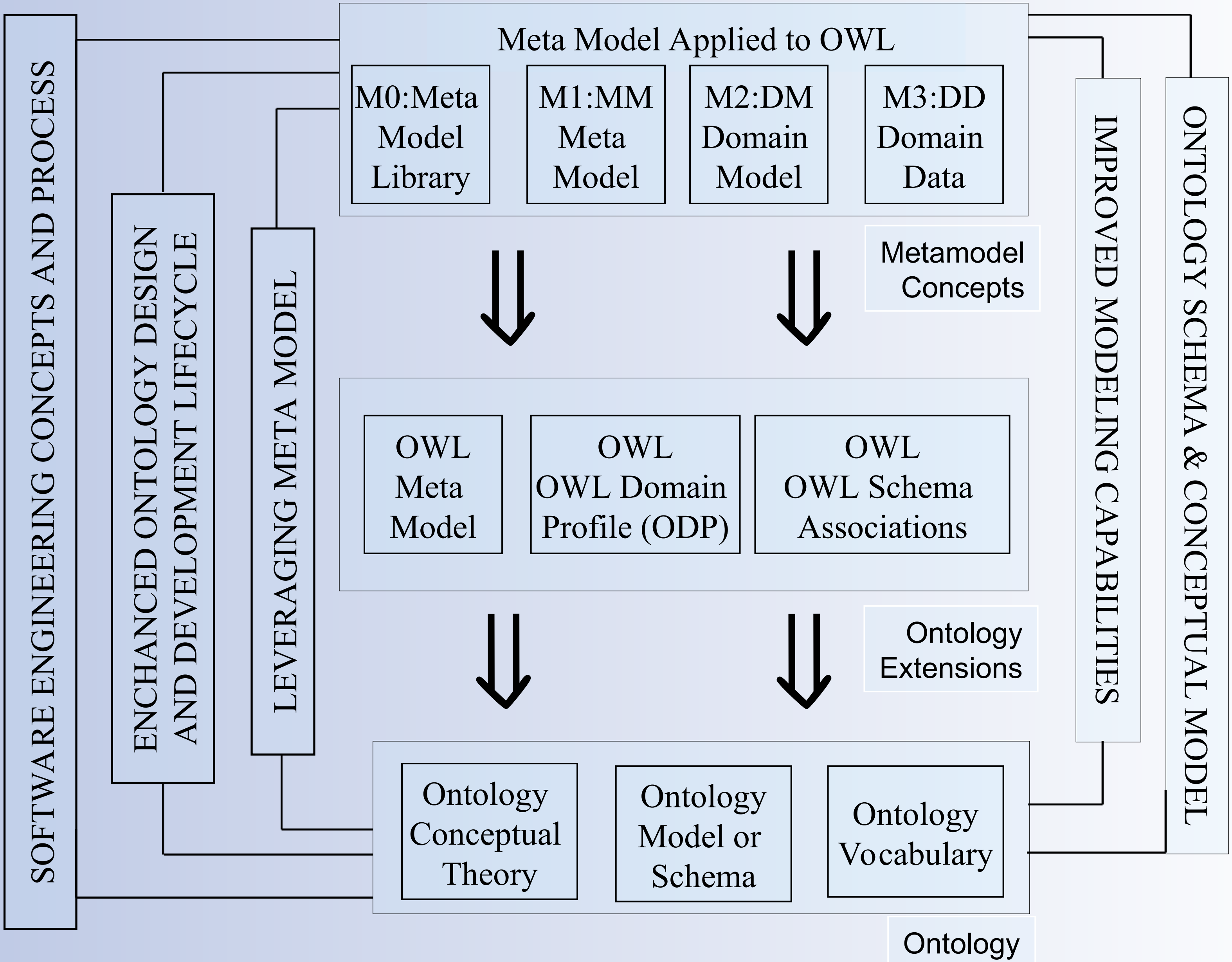
How are these Problems Addressed?

- Provide Object-Oriented Modeling Concepts to Ontology Frameworks
  - Leverage OO Based Frameworks UML, ERD, XML
    - Shifting Ontology Development From Instance Based → Design Oriented Approach
  - Provides the Ability to Design Models at Various Levels
    - Meta-Models and Domain Models
- Enhance Ontology Tools with New Modeling Concepts
  - Provides Software Engineering Based Usage
  - Developed Models are Reusable for Multiple Domain Environments
- Leverage Software Development Process Concepts:
  - Adapt Software Development Methodology for Ontologies
    - Agile Methodology, Meta Process Modeling, etc.

What is the Big Picture ?

- A Software Engineering Framework for Ontology Design and Development
  - Provides a Software Centric Work-Flow for Ontologies
    - Promotes a Design-Oriented Approach
- Define Ontology Design and Development Process
  - Employs the Leveraged Modeling Concepts
  - Adopts a Agile Development Methodology
- Improve Ruse Potential and Interoperation
  - Reusable Ontology Models and Ontology Vocabulary (i.e. instances) in Multiple Health Settings
- Apply the to Biomedical Informatics (BMI) Domain

The Big Picture: Ontology Framework



Research Emphases

- A. Meta Model for Ontologies
  - Applying UML Metamodel to OWL
  - Extending OWL with Domain Profile
- B. Ontology Model and Schema:
  - Extensions – Class & Attribute, Domain Profile, Ontology Schema Associations
  - Extending OWL and ODM
- C. Improved Abstraction for Knowledge Representation
  - Capturing Domain Abstract Theory with Domain Profile Extension
- D. Hybrid Ontology Design and Development Lifecycle
  - Ontology Design and Development Process employing A, B, and C
  - Leveraging Software Design and Development Process

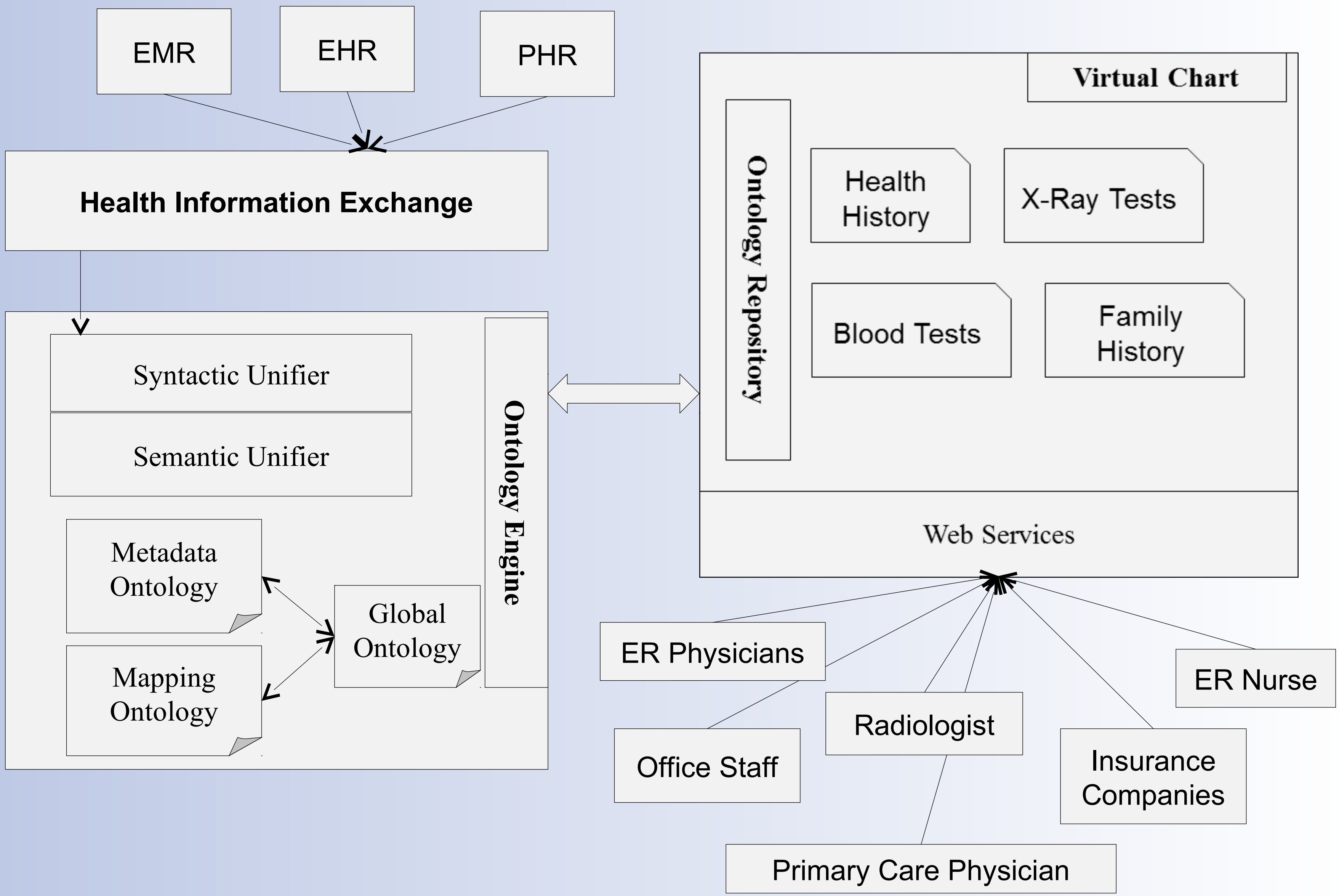
Overview of Presentation

- Background on Biomedical Informatics and its Relevance in Proposed Work
- Sample Clinical Scenario
- Background
  - UML Meta-Model, RDF, RDFS, and OWL
  - Protégé Ontology Editor
- Compare and Contrast Models
  - Evaluation of Modeling Features against UML, ERD, XML and RDF/RDFS/OWL
- Proposed OWL Extensions
  - Attribute, Domain Profile and Schema Associations
- Hybrid Ontology Design and Development Model with Lifecycle Process
- Summary
  - Research Contributions
  - Ongoing and Future Work

Biomedical Informatics and Role of Ontologies

- Biomedical Informatics (BMI) is:
  - Collecting/Managing/Processing of All Types of Health Care Data
  - Primary Objective:
    - Improved Patient Health Care
    - Reduce Medical Errors
    - Reduce overall Medical Costs
- Intended to be Utilized in order to:
  - Digitalize clinical information in the form of EHR, CCD, etc.
    - Standards Include HL7 CDA R1 and R2, RIM Model, etc.
  - Share the information
    - Health Information Exchange (HIE) to Integrate Data
    - Virtual Chart (VC) to Present Integrated View to Users
- Ontologies Preserve Semantics of the Clinical Data
  - Standards - UMLS, ICD, MeSH, LONIC, etc.

Role of Ontologies in Health Information Exchange



Example of Conflicting Ontologies

- Ontology 1:
  - Disease References Symptoms which References Treatments
  - Hierarchy of:
    - Disease
      - Respiratory Disease
      - Cardio Disease
      - Nervous Disease
    - Symptoms
      - General Symptoms
      - Behavioral Symptoms
    - Treatment
      - General Treatment
      - Surgical Treatments
- Ontology 2:
  - Symptoms References Diseases which References Treatments
  - Hierarchy of:
    - Symptoms
      - General Symptoms
      - Behavioral Symptoms
    - Disease
      - Respiratory Disease
      - Cardio Disease
      - Nervous Disease
    - Treatment
      - General Treatment
      - Surgical Treatments

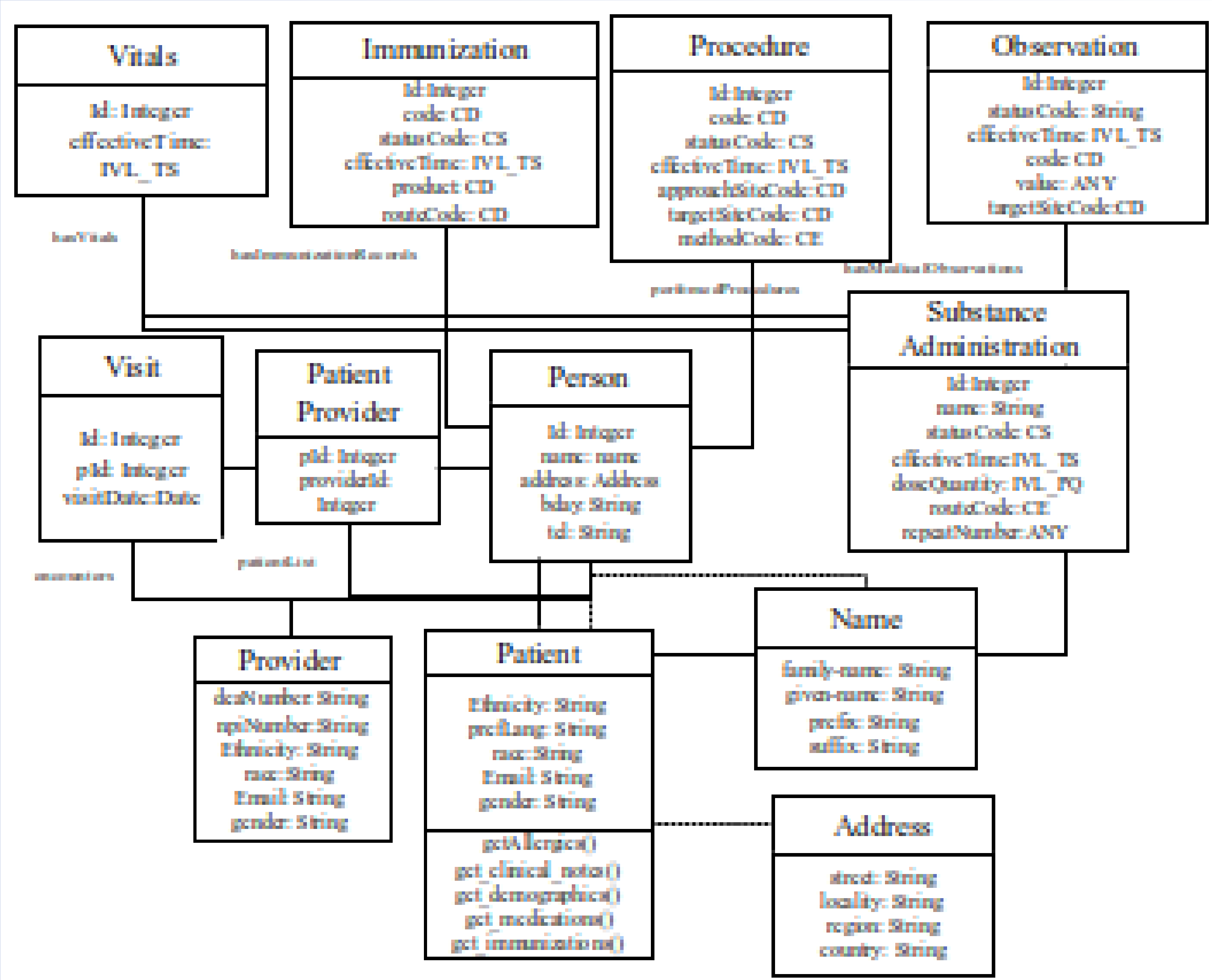
- Previously Discussed Issues:
  - How do you Integrate Ontologies Across HIT to Support HIE and VC?
  - How do you Merge Data Intensive Conflicting Ontologies?
  - How do you query from Inside Out?

Scenario in Clinical Domain

- Sample Clinical Scenario
  - Current Status
    - Mr. Jones Arrives with Shortness of Breadth, Occasional Chest Pain, etc.
    - Physician Performs Tests (XRay, EKG, Blood work, etc.) and Collects Test Results
    - Discharged – Recorded in EHR_2
  - Previously Suffered From CHF
    - Discharged with Lasix – Obtained From EHR1
- Clinical Researcher
  - Perform Queries Across Multiple Resources (EHR_1, EHR_2, EHR_3…….)
  - For Example,
    - What is the patient's profile with CHF and associated medications involved for diabetic therapies?
    - Are there patterns of laboratory test results seen in type 2 diabetes patients that are associated with increased risk of developing CHF or Stable Angina?
    - Does metformin affect the utility of the BNP test for the diagnosis and monitoring of CHF?

Sample UML Diagram

CSE 5810



Background on UML

CSE 5810

- UML Provides Diagrammatic Abstractions
  - Concepts: Actors, Use Cases, Class, Object, etc.
  - Diagrams: Class, Use Case, Sequence, etc.
- Underlying OMG Meta-Model Provides
  - Building Blocks to Construct and Extend UML
  - Employ's UML Meta Object Facility (MOF)
- Four Layers:
  - M3 Meta-Meta Library (MML)
  - M2 Meta-Model (MM)
  - M1 Domain Model (DM)
  - M0 Domain Data (DD)
- Align Concepts to Ontology Definition Model(ODM)

Background – RDF, RDFS and OWL

- Numerous Knowledge Representation Frameworks:
  - KIF, LOOM, DAML, DAML+OIL, RDF/RDFS and OWL
  - Facilitates binding *semantics* to information
- OWL is built on Resource Description Framework (RDF) to leverage Triple Structure or RDF Statement, which is of the form:

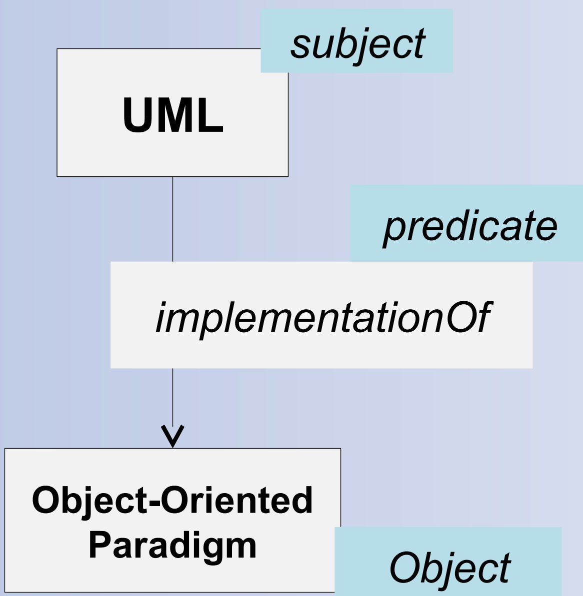
- For Example,
  - Heart Attack(Subject) hasSymptom (predicate) Stroke (Object)
- Endorsed by W3C: Web Ontology Language (OWL) and OWL DL is built on SHOIN description language

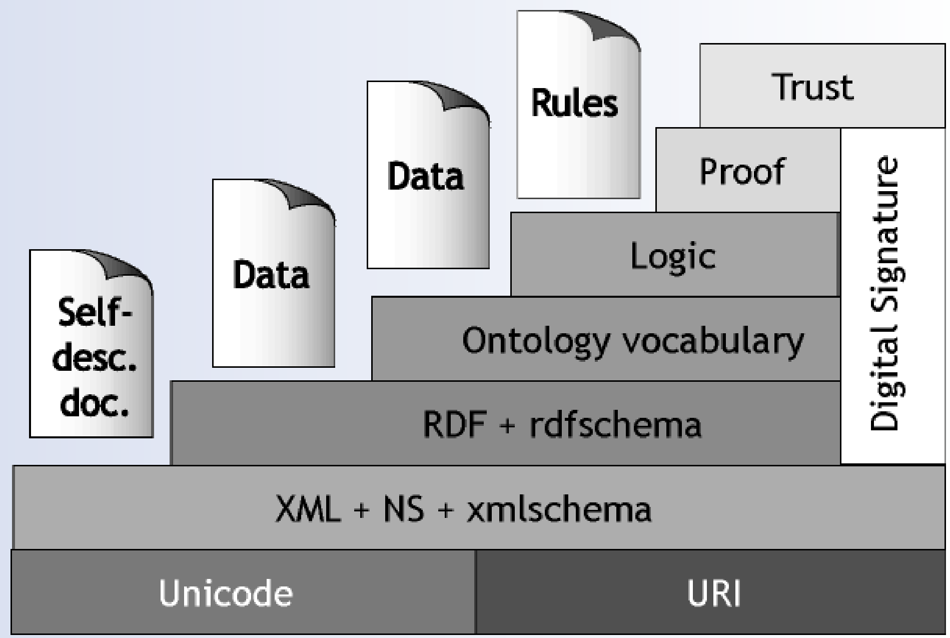
Background – RDF, RDFS and OWL

- OWL is more Expressive than RDF/RDFS
  - Axioms, Role Hierarchy, Transitive Roles, Inverse Roles, and Qualified Restrictions
- OWL DL (Description Logic) is Popular as it Supports Inference/Reasoning
- OWL DL provides Schema Modeling Elements:
  - OWL:Class : a set of Objects or Individual
  - OWL:ObjectProperty: captures Binary Relationship between Classes, e.g., Associations in Software Models
  - OWL:DatatypeProperty: capture Datatype Properties (e.g., integer, double, string, etc.)
  - OWL:AnnotationProperty: provides Annotation Mechanism to Concepts such as rdfs:seeAlso, rdfs:comment etc.

Background on RDF and RDFS

- Numerous Knowledge Representation Frameworks
  - KIF, LOOM, DAML, DAML+OIL, RDF/RDFS & OWL
  - Facilitates Binding Semantics to Information
- Resource Description Framework (RDF) and RDF Schema (RDFS) – Knowledge Expressed as *Triple Statement*





Background on OWL

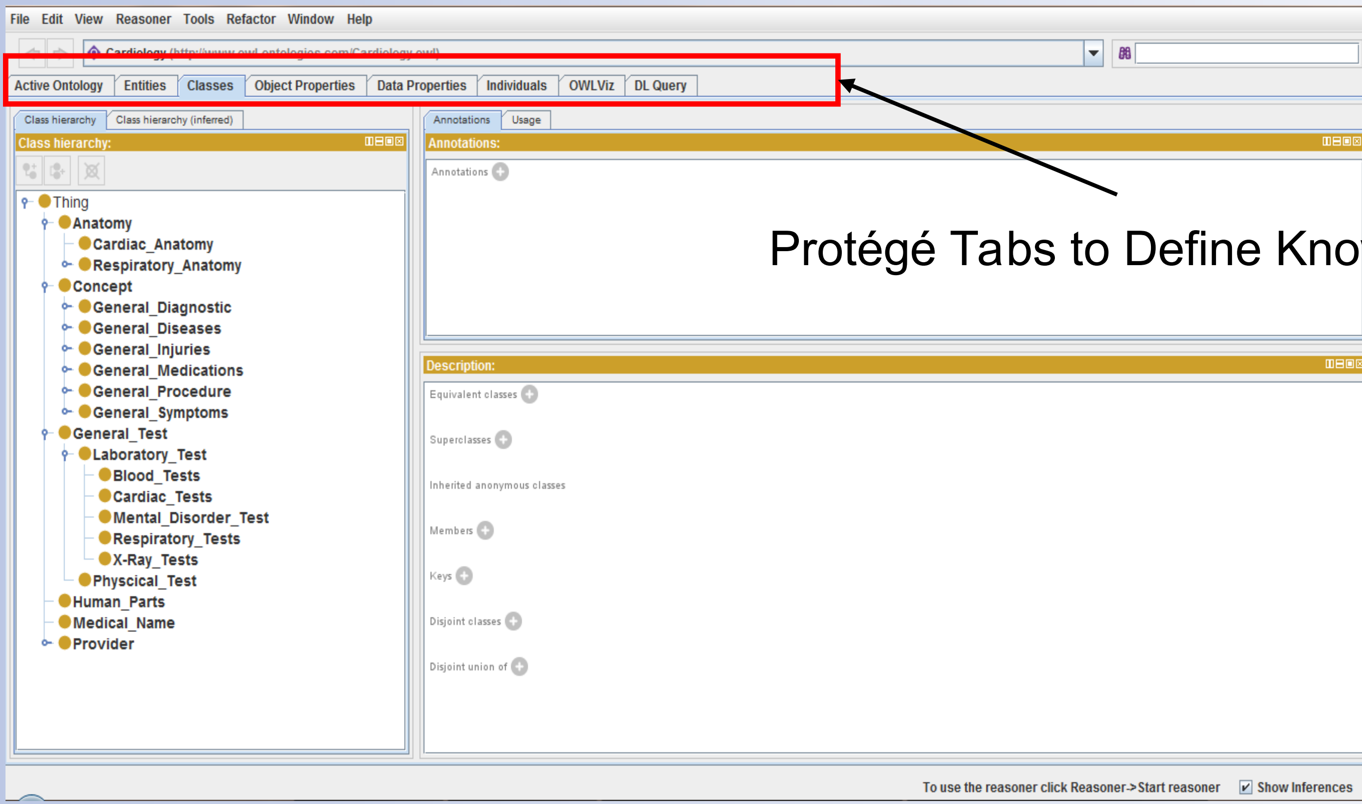
CSE 5810

- OWL Exploits RDF Triple Structure
- OWL is more Expressive than XML, RDF/RDFS
  - Axioms, Role Hierarchy, Transitive Roles, Inverse Roles, Qualified Restrictions, Reflexive Roles, Symmetric Roles etc.
- OWL DL (Description Logic) is Popular as it Supports Inference/Reasoning
- OWL DL Provides Schema Modeling Elements:
  - owl:Class: a set of Objects or Individual
  - owl:ObjectProperty: captures interactions between Classes,
    - Similar to Associations in Domain Modeling
  - owl:DatatypeProperty: capture Datatype Properties
  - owl:AnnotationProperty: provides Annotation to Concepts.

Protégé Editor – Ontology Editor

CSE 5810

- Standard Editor for Developing OWL Ontologies
  - Also Supports RDF, RDFS, Frames
- Architecture
  - Open Source, Extendable Java Swing Based UI
  - Ontology Editing using HP Jena API
  - Plugin-Play Architecture (e.g., Eclipse IDE)
  - Protégé 4.x is Current Version Support OWL 1



Protégé Tabs to Define Knowledge Concepts

Why Protégé Ontology Editor?

- Protégé Editor
  - Most Popular OWL Ontology Editor
    - Biologist, CS Researchers, Finance, etc.
  - Supports Multiple Formats and Allows Database Connections
  - Open Source and Flexible Architecture
- Leverage Existing Tools
  - Promote OO Concepts and Design Based Approach
- Benefits
  - Connect to Standard Ontology Repositories
  - Well-Defined API Allow Extension with New Capabilities

Compare and Contrast Models

- Available Models/Frameworks:
  - Unified Modeling Language (UML)
  - Entity Relationship Diagrams (ERD)
  - eXtensible Markup Language (XML)
  - Web Ontology Language (OWL)
- Compared in terms of
  - Basic Building Blocks
  - Abstraction Levels
  - Modeling Capabilities/Features
- Two Step Process
  - Define Object-Oriented Modeling Concepts
  - Compare/Contrast Against: UML, ERD, XML & OWL
- Intent
  - Identify Capabilities Lacking in OWL

What is the Disconnect?

- Ontologies are:
  - Application Based
  - Proceed from
    - Objects
    - Classes
    - Links
    - Hierarchy
  - Completely Data Focused
  - Bottom up Approach
  - Build a Specific Ontology for a Application
  - Inability to Reuse
  - Difficult to Integrate with one another
  - No Design Focus
- Modeling Frameworks:
  - Class/Type Based
  - Construct design artifacts:
    - Entities
    - Schemas
    - Classes
    - Relationships
    - Patterns
  - Top-Down Approach
  - Solution can apply to multiple applications
  - Emphasize Reuse
  - Components Interact with one another
  - Predominate Design Focus

Modeling Capabilities/Features

- Schema Definition: A Conceptual Model that Describes and Represents the Structure, and Behavior of a System
  - Classes in UML, XML Schema in XML, ERD in DB Design
- Schema Association: Relationship between the Schemas
  - A design can be separated into logical pieces
- Classes: A Structural Representation (aggregation) at a Design Level
  - Objects which share common attributes or properties into a named entity
- Attribute: Features for the Class
  - Describe characteristics of the class and owned by the class
- Association: Ability to Relate two or more Classes There are Three Types:
  - Qualified Association: Based on a Look-up or Key Value
  - Association Class: Properties describe Association
  - N-Array Association: three or more classes

Modeling Capabilities/Features

- Inheritance: Ability to Relate Classes based on Common (different) Information/Functionality
  - Extension: child adds functionality to parent
  - Specialization: child specializes parent
  - Generalization: common attributes from multiple classes form parent
  - Combination: child inherits from more than one parent class
- Constraints: Ability to Impose Constrain on Classes, Associations, etc.
  - OCL Language for UML
  - Cardinality Constraints on Associations
- Profile: Ability to Extend the Meta-Model to Define Domain Specific Meta-Model Concepts.
  - UML Profile – Extends UML Meta-Modeling features.

Compare and Contrast

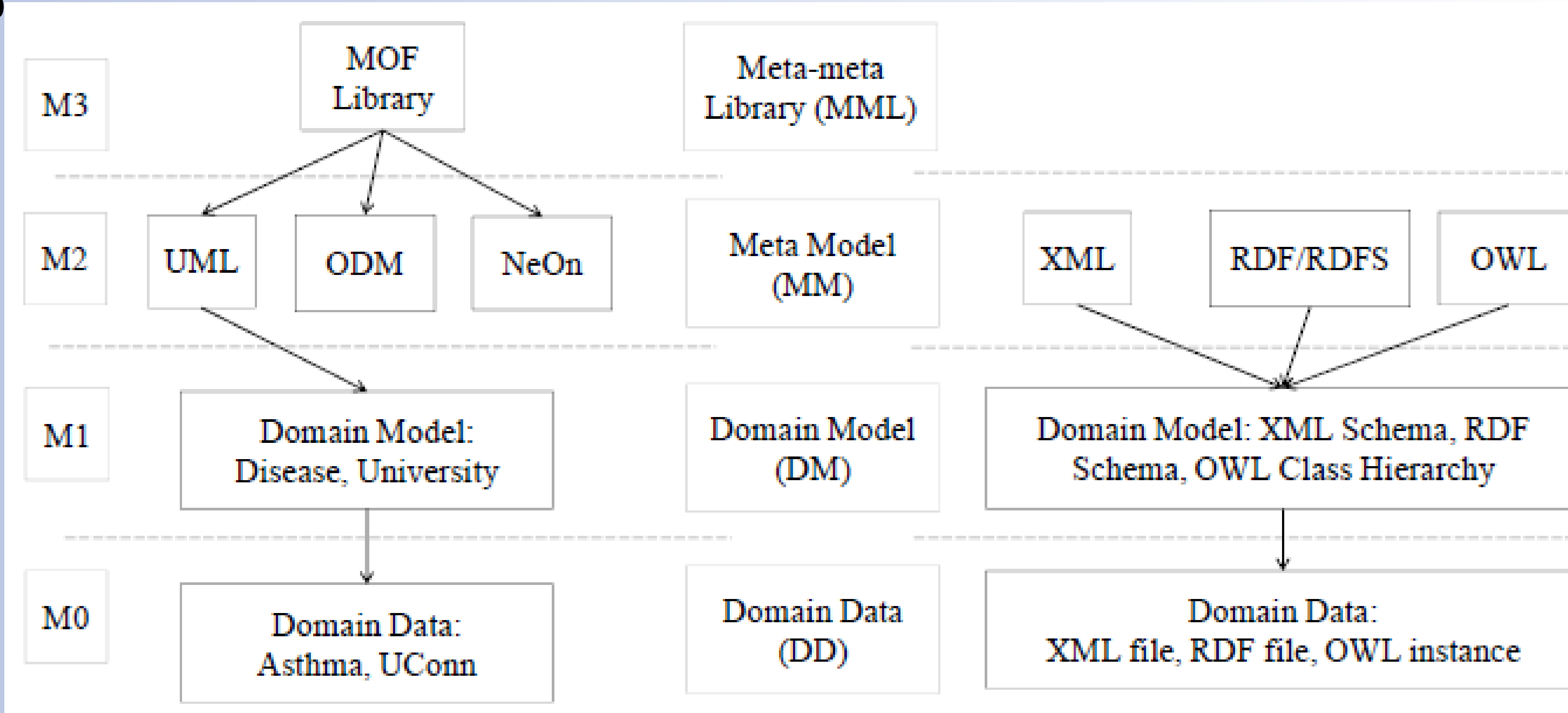
Modeling Element	UML	ERD	XML	OWL
Schema Definition	Full	None	Full	***Partial***
Schema Associations	Full	None	Partial	***Partial***
Class	Full	Partial	Full	***Partial***
Associations	Full	Full	Full	***Partial***
Qualified Association	Full	Full	Full	***None***
Association Class	Full	Full	Full	Full
N-ary Association	Full	Full	Full	Full
Cardinality	Full	Full	Full	Full
Inheritance	Full	Full	Full	Full
Extension	Full	Full	Full	Full
Specialization	Full	Full	Full	Full
Generalization	Full	Full	Full	Full
Combination	Full	Full	Full	Full
Constraints	Full	Full	Full	Full
Profile	Full	None	None	***None***

How Will OWL Change?

- Changing in two ways:
  - Align to MOF UML Meta-Model
  - Extend OWL with Modeling Features
    - Extension at the Meta-Model Level (M2)
      - Class & Attribute
      - Domain Profile
    - Extensions at the Model Level (M1)
      - Ontology Model/Schema Associations
- Secure Position in Modeling Hierarchy
  - Meta-Model Layer – OWL Meta-Model
  - Domain Model Layer – OWL Model/Schema
  - Instance Layer – OWL Instances

Applying Modeling Perspective

CSE 5810

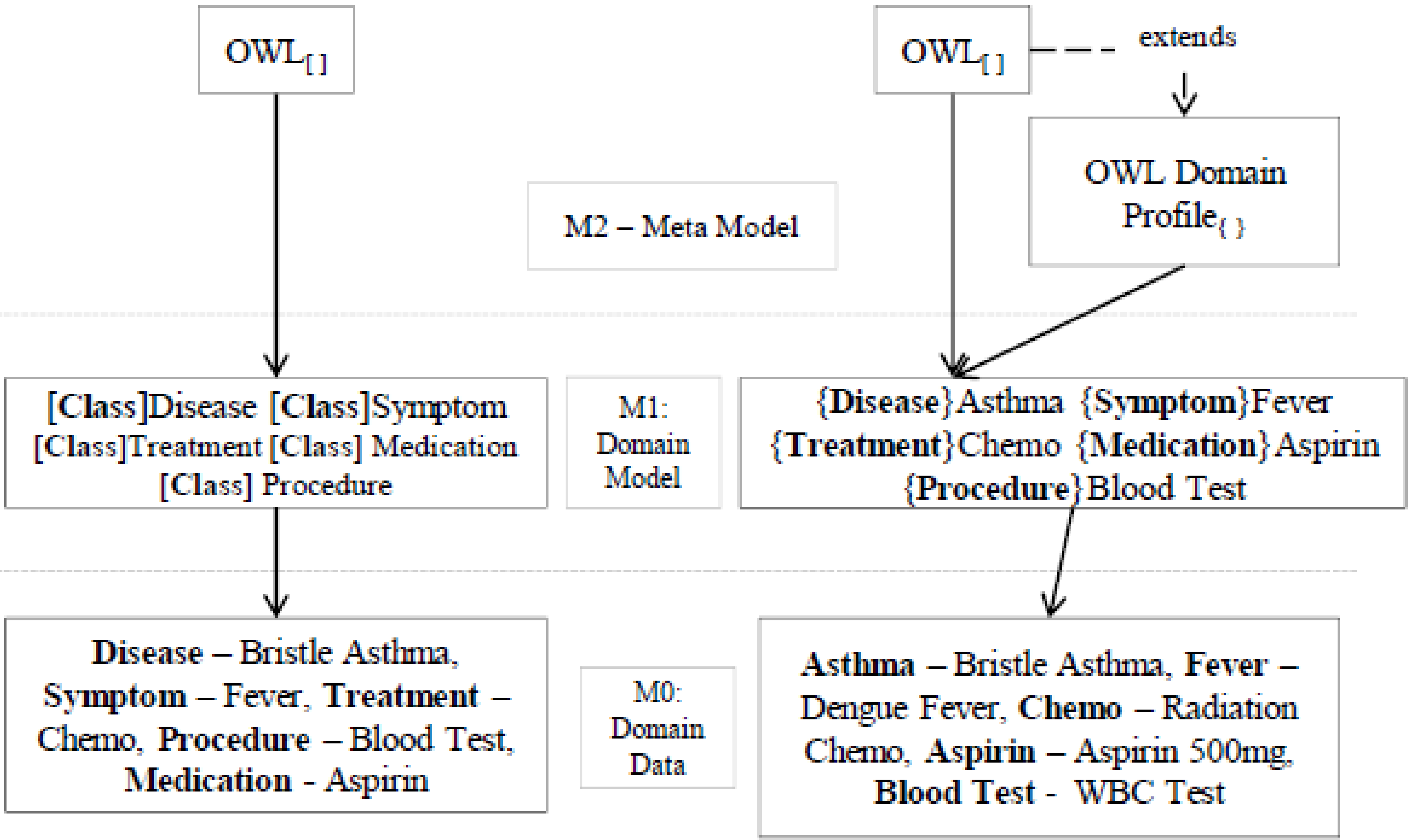


Hierarchical Organization of **UML, ODM** & **NeOn**

Applying Layered Approach to **XML, RDF/RDFS** & **OWL**

OWL Extension – Domain Profile

CSE 5810

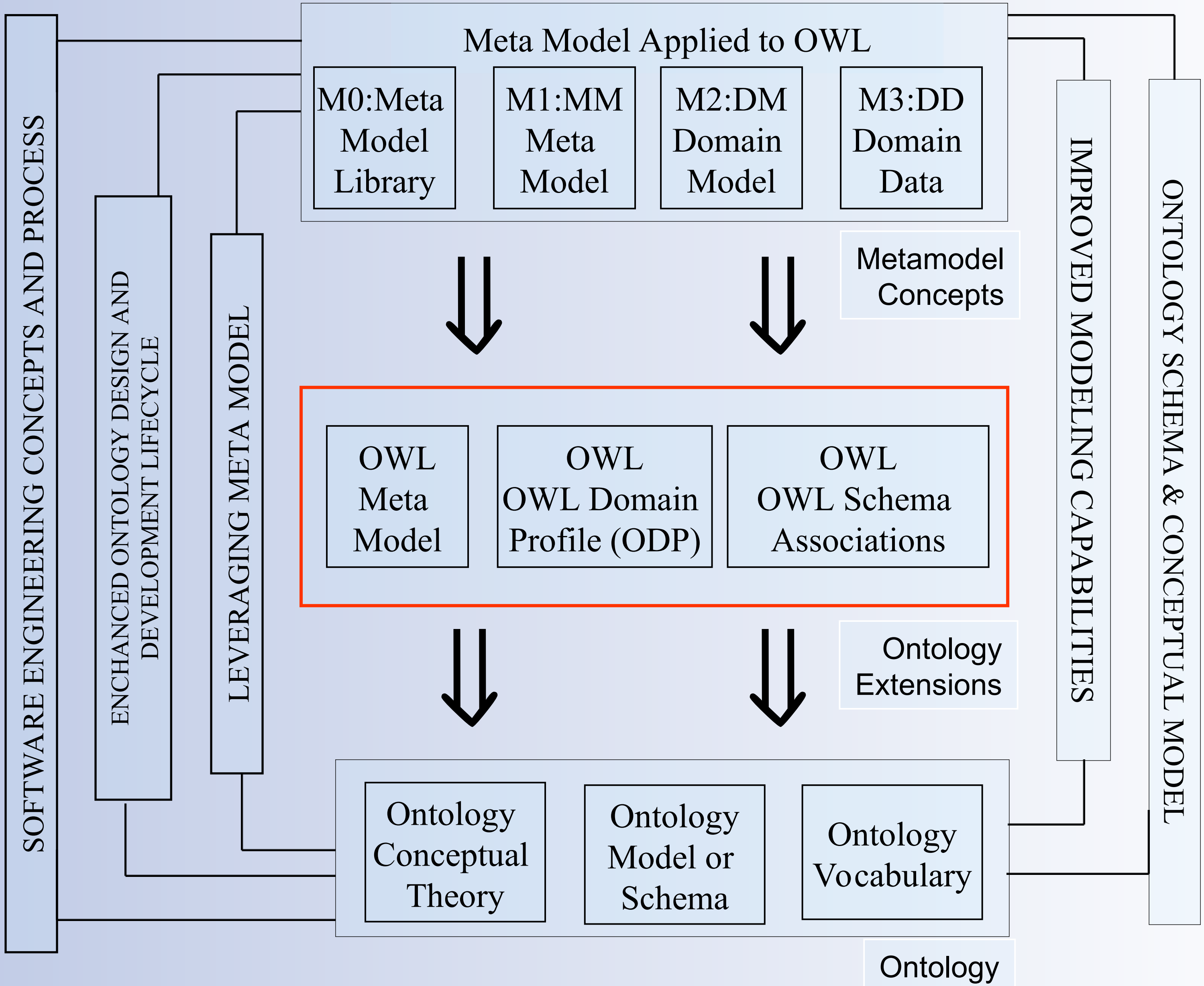


(A) Developing Ontology using OWL

(B) Developing Ontology using OWL and ODP

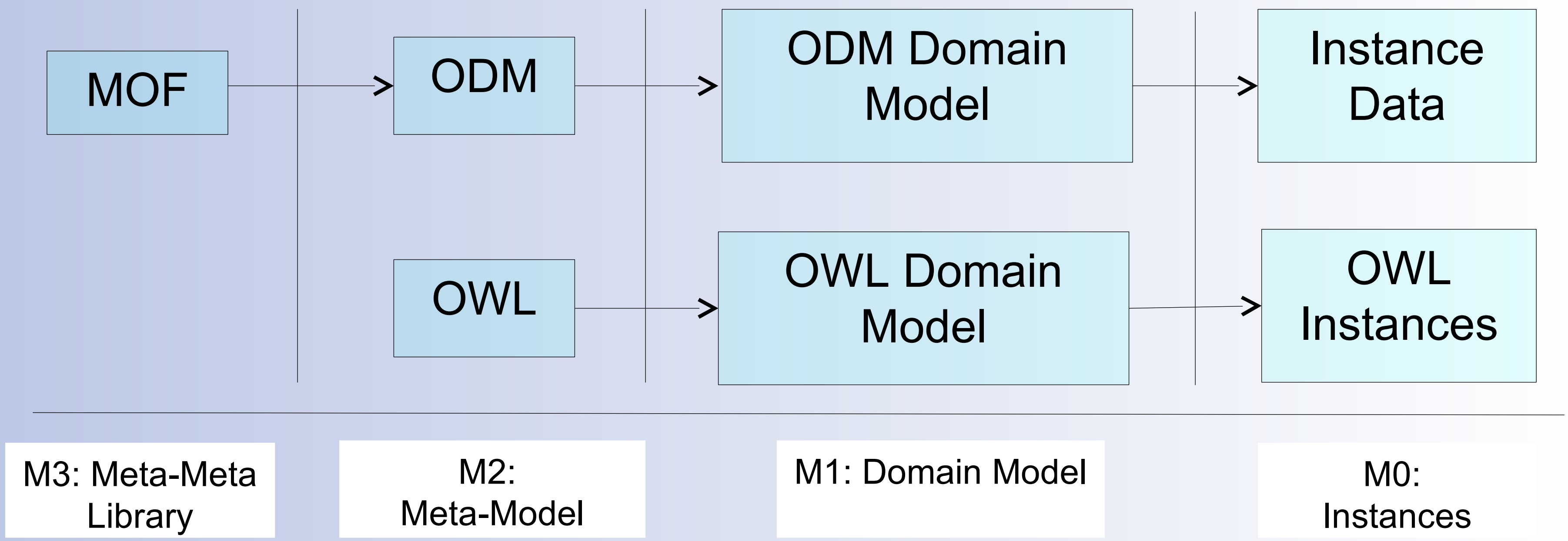
Where are we in Overall Process?

CSE 5810



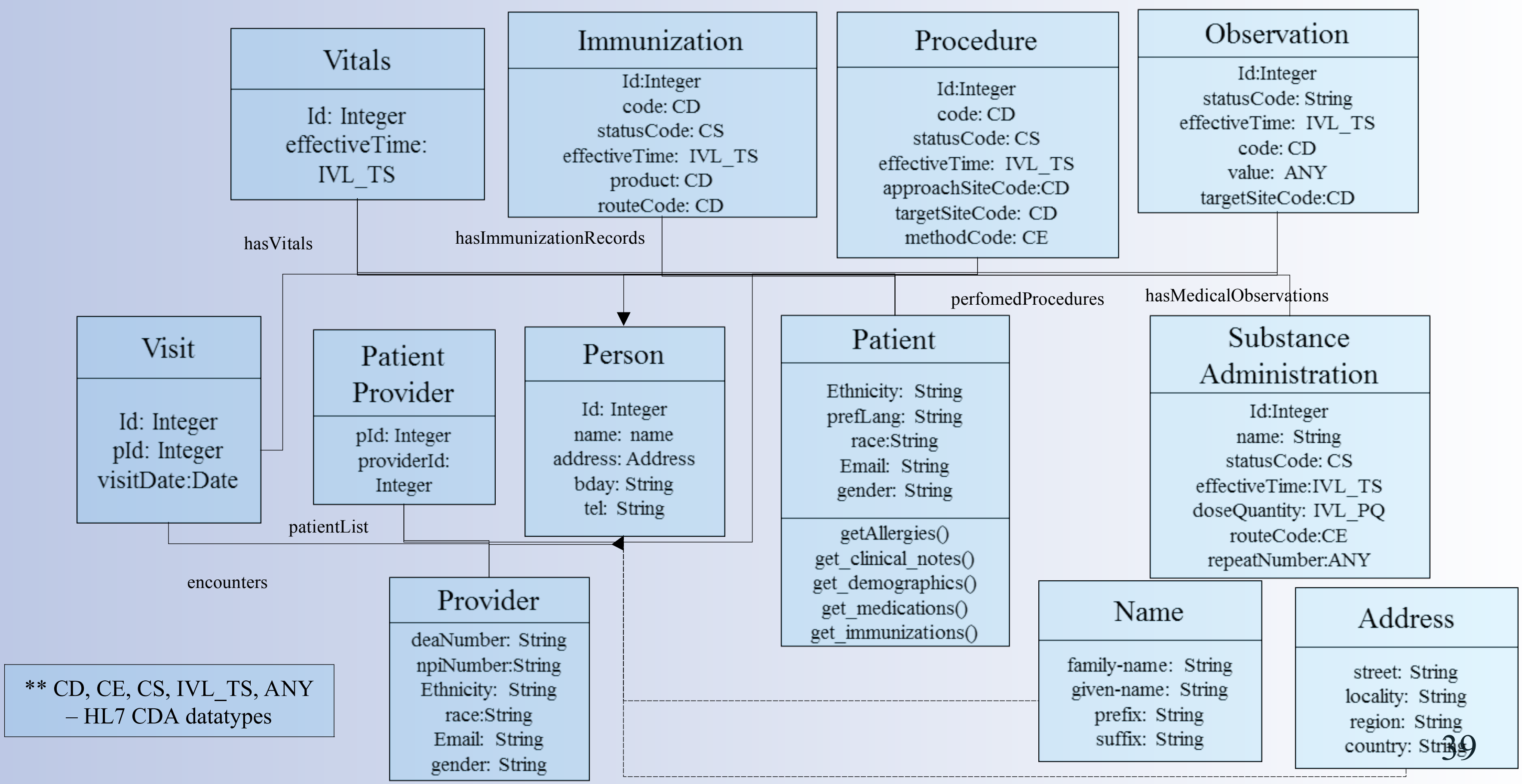
Proposed OWL Extensions

- Extension at the Meta-Model Level (M2)
  - Class & Attribute
  - Domain Profile
- Extensions at the Model Level (M1)
  - Ontology Model/Schema Associations



OWL Extension – Class & Attribute

- "A Class is formed by grouping a set of *Objects* or *Instance*" – OWL DL Semantics
  - Conflicts with Software Modeling Definition of a Class
    - Aggregation of Attributes



OWL Extension - Class & Attribute

- UML Class Diagram is converted into OWL:
  - Attributes "id, gender, email, race" are mapped to owl:DatatypeProperty
  - Association
    - hasObservations, vitals, perfomedProcedures
    - Mapped to owl:ObjectProperty.
  - Attributes
    - hasName, hasAddress, hasStatusCode, hasValue hasEffectiveTime, etc.
    - Mapped to owl:ObjectProperty.
- Mapping both Association and Attribute to the Same Modeling Entity owl:ObjectProperty Causes Semantic Ambiguity in Representing the Link
  - Also Resulting in a lack of a true concept of a "***Class***" in OWL and Ontologies

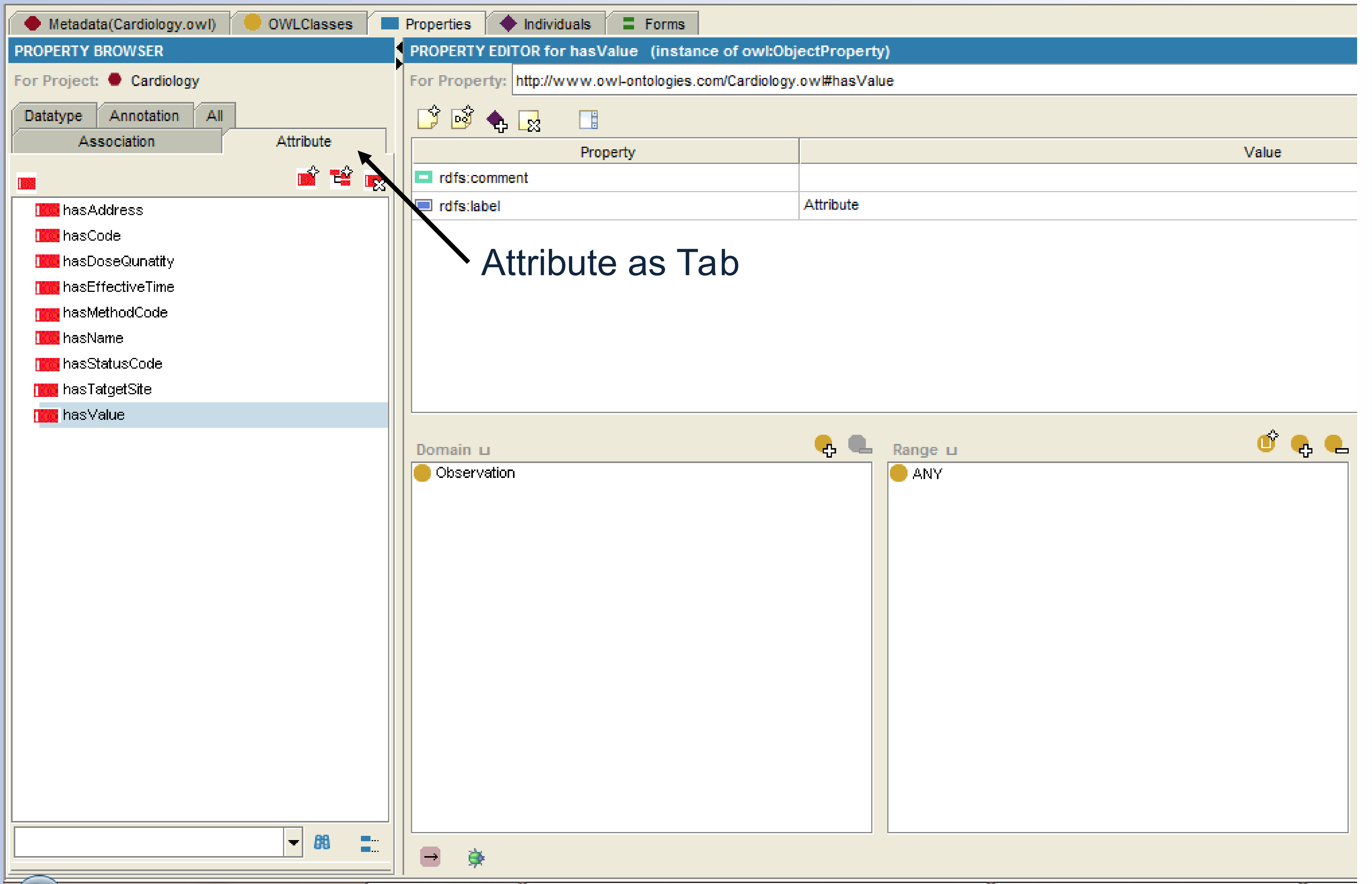
OWL Extension - Class & Attribute

- Define Attribute to capture feature of a class.
  - Semantics:
    - $C\{At_1, At_2 \ldots At_n\}$, where each At_i is the Attribute
    - Attribute is a *Role* in the Domain $\{At_i \subseteq \Delta^I \times \Delta^I\}$, which is Owned by the Class
  - Syntax:
    - <owl:Attribute rdf:id="hasEffectiveTime">
      <rdfs:Domain rdf:id="Observation"/>
      <rdfs:Range rdf:id="IVL_TS"/>
      </owl:Attribute>

    <owl:Attribute rdf:id="hasStatusCode"/>
    <owl:Attribute rdf:id="hasAddress"/>

Protégé Implementation - Attribute

- Attribute Tab in the Protégé Property Browser

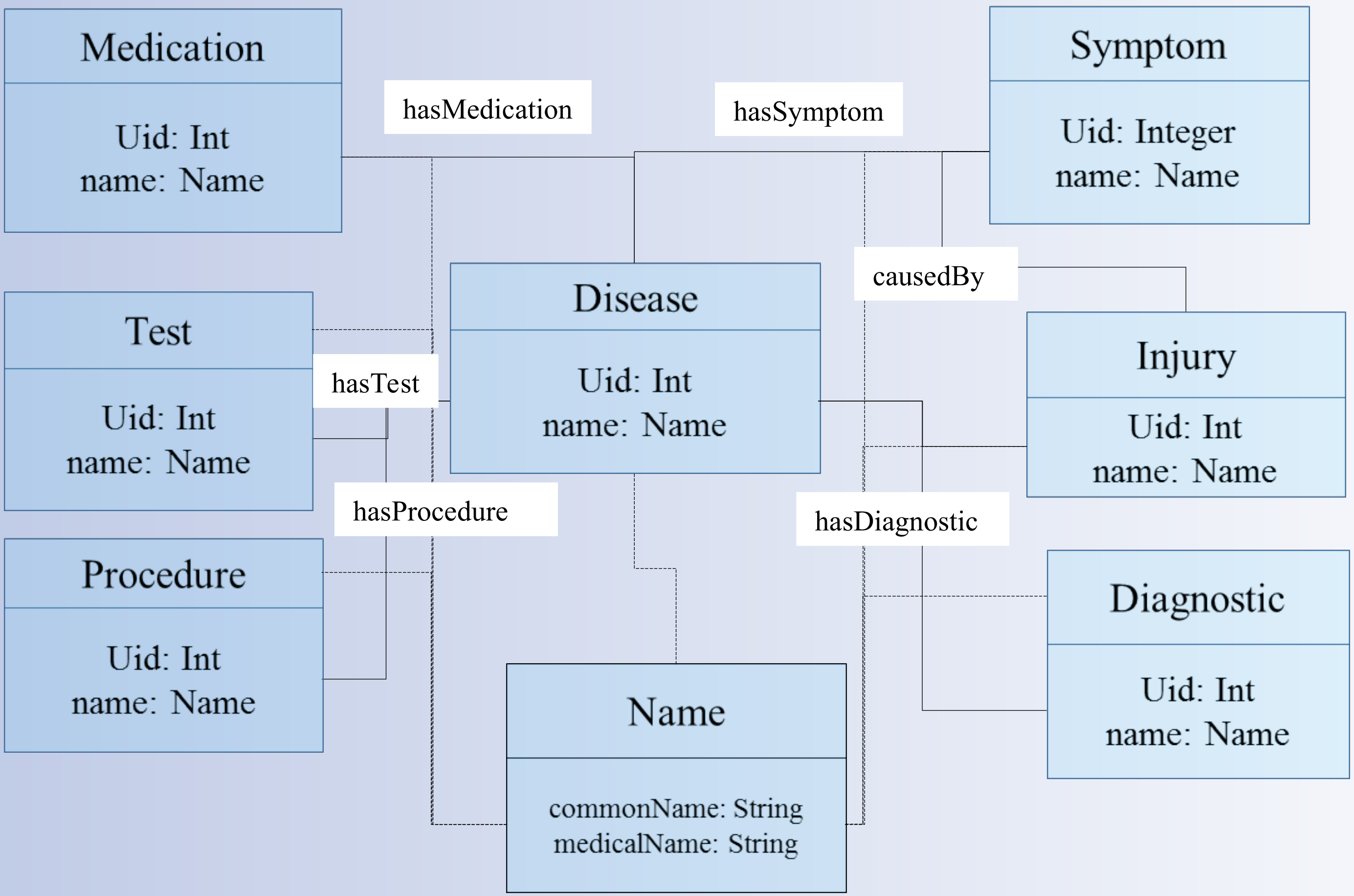


OWL Extension - Domain Profile

- Domain Profile – is an Abstract Theory Agreed by the Stakeholders before Developing Domain Models
  - Provides High-level Conceptual Perspective of the Domain Model
- OWL Domain Profile (ODP) extension, Captures the Concepts of the *Abstract Theory*
  - Extends OWL Meta-Modeling Concepts
- For Example, in BMI
  - Type Concepts: Disease, Symptom, Injury, Diagnostic, Procedure, Test, Medication, Name
  - Type Associations: hasMedication, hasTest, hasSymptom, isCausedBy, etc.
  - Type Attributes: hasName, hasUid, etc.

OWL Extension - Domain Profile

- Abstract Theory - Defined using Identified Type Concepts.



OWL Extension - Domain Profile

- OWL Domain Profile (ODP) is comprised of :
  - *ProfileClass* extends OWLClass
    - Syntax: <odp:ProfileClass odp:id="Disease"/>
  - *ProfileAttribute* extends OWLAttribute
    - Syntax: <odp:ProfileAttribute odp:id="hasName"/>
  - *ProfileObjectProperty* extends OWLObjectProperty
    - Syntax: <odp:ProfileObjectProperty odp:id="hasTest"/>
  - *ProfileDatatypeProperty* extends OWLDatatypeProperty
    - Syntax: <odp:ProfileDatatyeProperty odp:id="id"/>
- ODP Entities extend the OWL Core Modeling Entities
  - Obey the Semantics of OWL Meta-Model Elements
- ODP Profile Entities are Imposed onto the Domain Model

OWL Extension - Domain Profile

CSE 5810

- For the Sample Abstract Theory,
  - At the metamodel level:

    <odp:ProfileClass odp:ID="Disease"/>

    <odp:ProfileClass odp:ID="Symptom"/>

    ……

    <odp:ProfileObjectProperty dp:ID="hasMedication"/>

    <odp:ProfileObjectProperty dp:ID="hasSymptom"/>

    ……..

    <odp:ProfileAttribute odp:ID="hasName"/>

    <odp:ProfileDatatypeProperty rdf:ID="hasUId"/>

    <odp:ProfileDatatypeProperty rdf:ID="hasCommonName"/>

    ………

OWL Extension - Domain Profile

- Imposing the Profile Type Concepts onto the Domain Model Concepts
  - Domain Model Concepts:

    <odp:Disease *odp:isOfType*="Cardiac Diseases"/>

    <odp:Disease *odp:isOfType*="Respiratory Diseases"/>

    ………

    <odp:hasSymptom *odp:isOfType* ="hasCardiacSymptoms"/>

    <odp:hasTest *odp:isOfType* ="hasBloodTest"/>

    ……

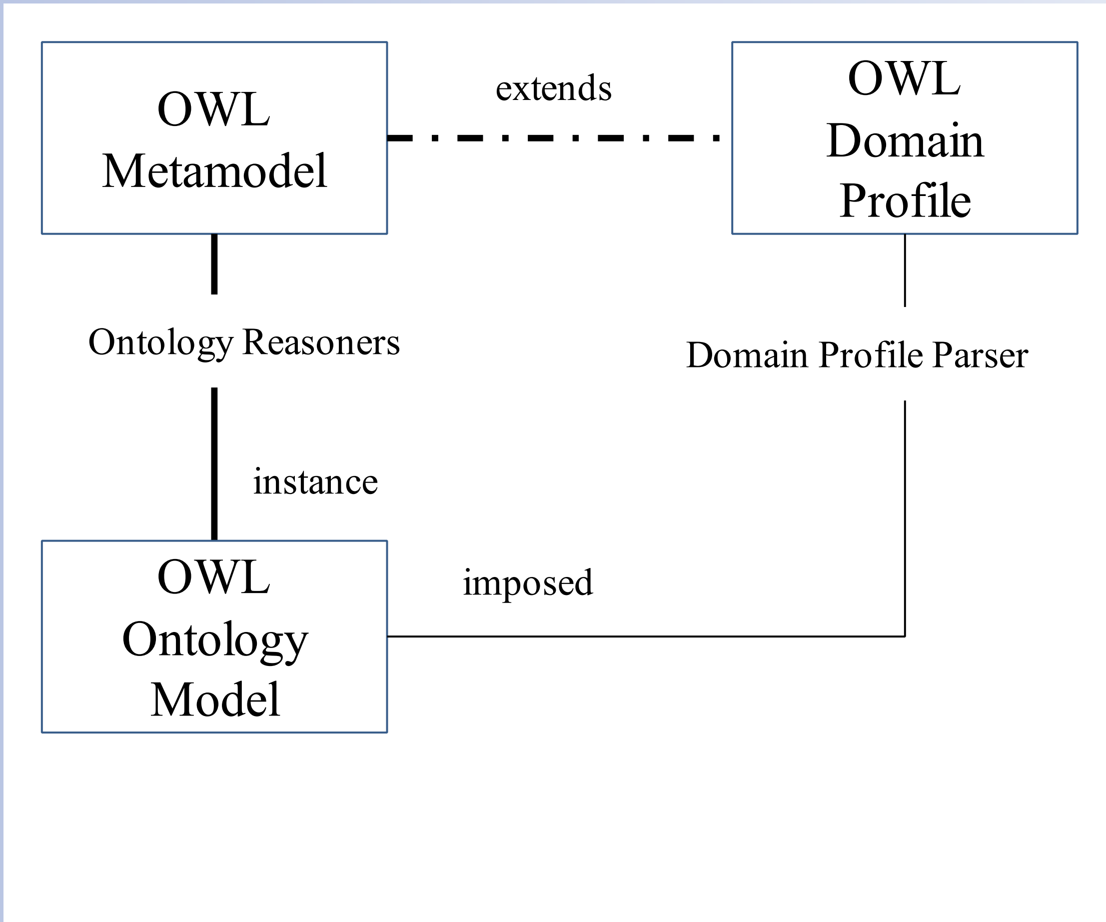
    <odp:hasUId *odp:isOfType* ="hasSSN"/>

    <odp:hasUId *odp:isOfType* ="hasTaxId"/>

    …….

OWL Extension – Domain Profile

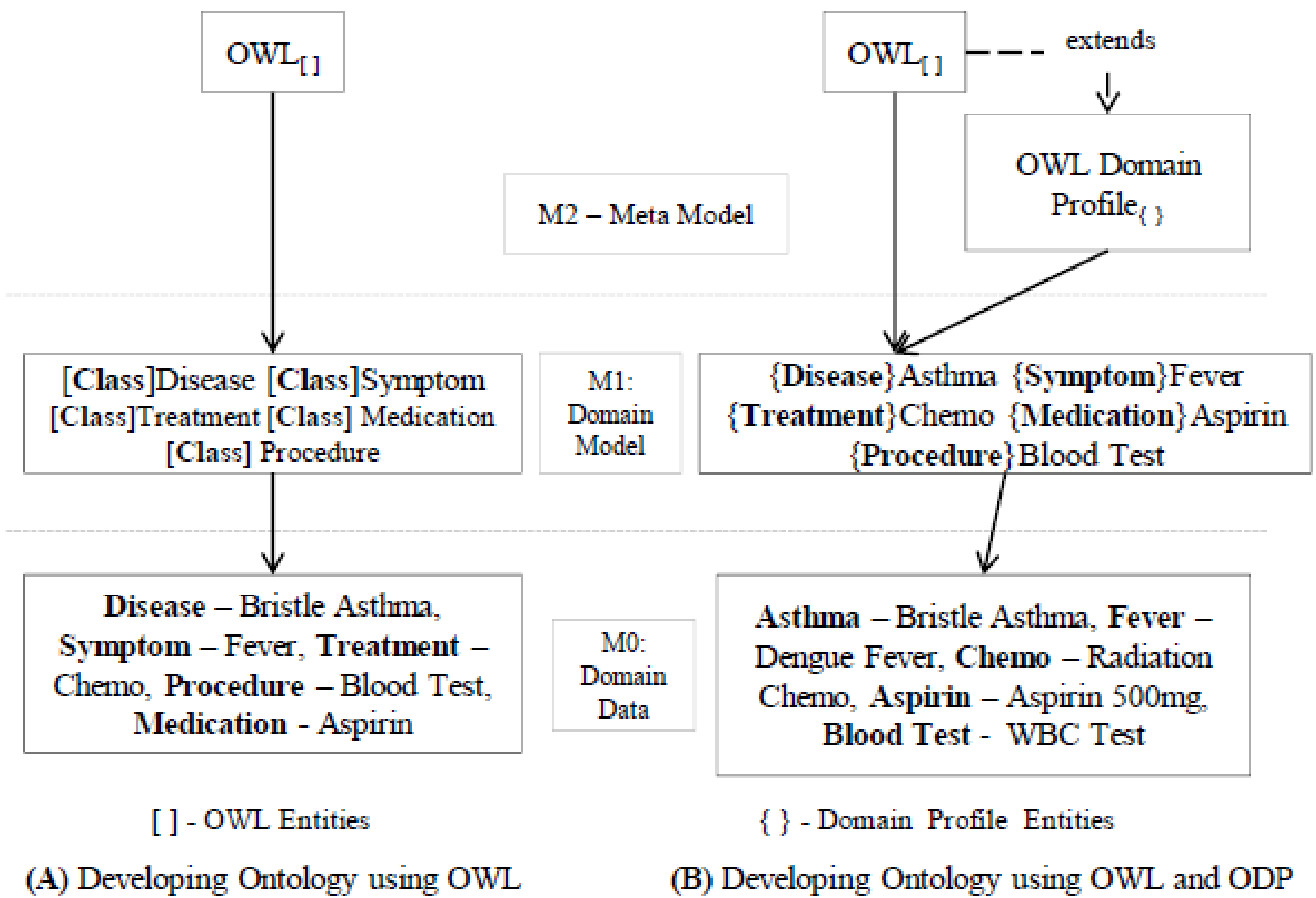
CSE 5810



- **DomainProfileParser** A Custom Parser to *Impose* and *Validate* the Profile (theory) onto the Ontology Model.
- ODP provides Structural and Semantics to the profile apart from OWL Ontology Model.

OWL Extension – Domain Profile

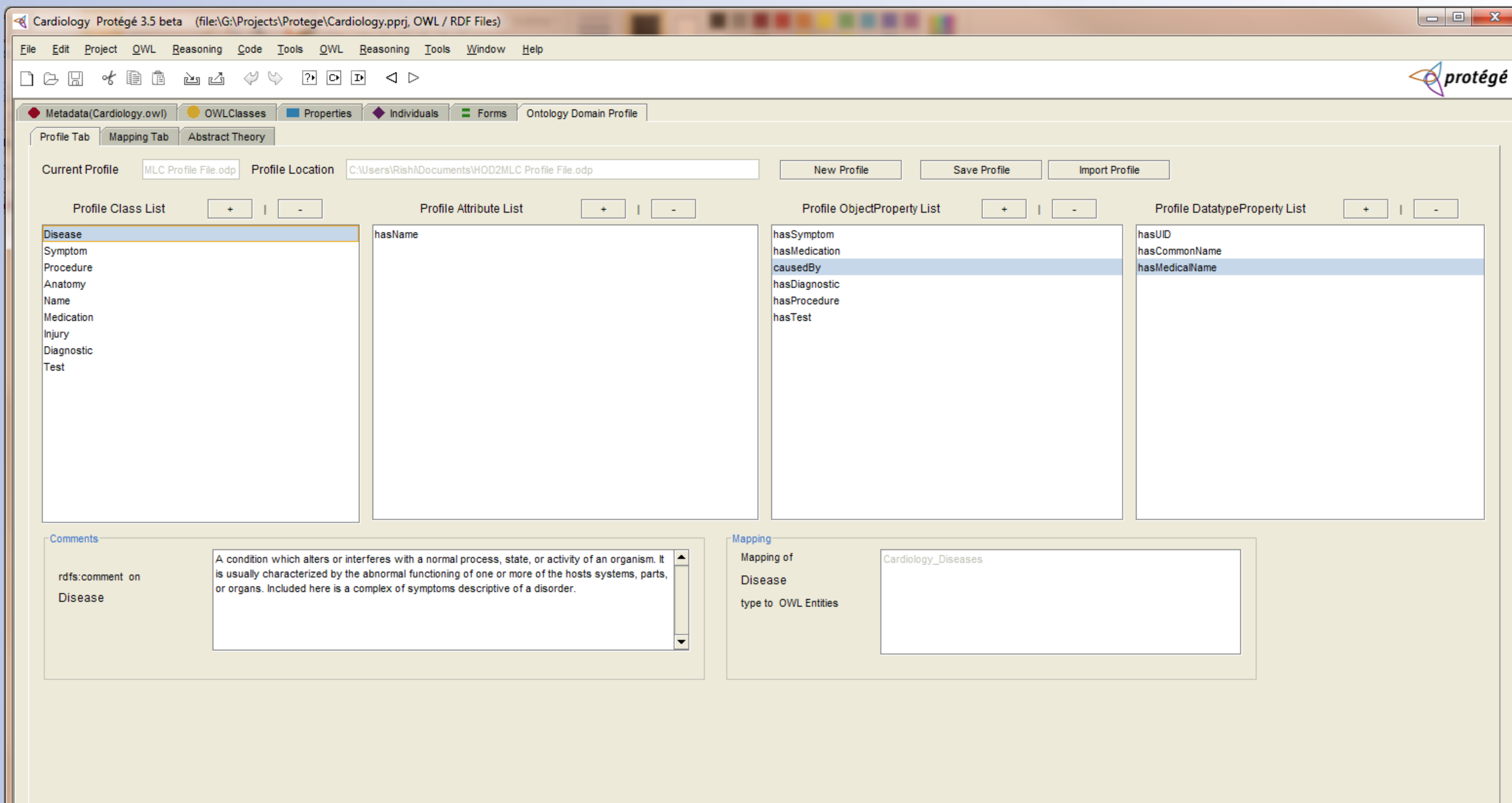
CSE
581



Protégé Implementation – ODP

CSE 5810

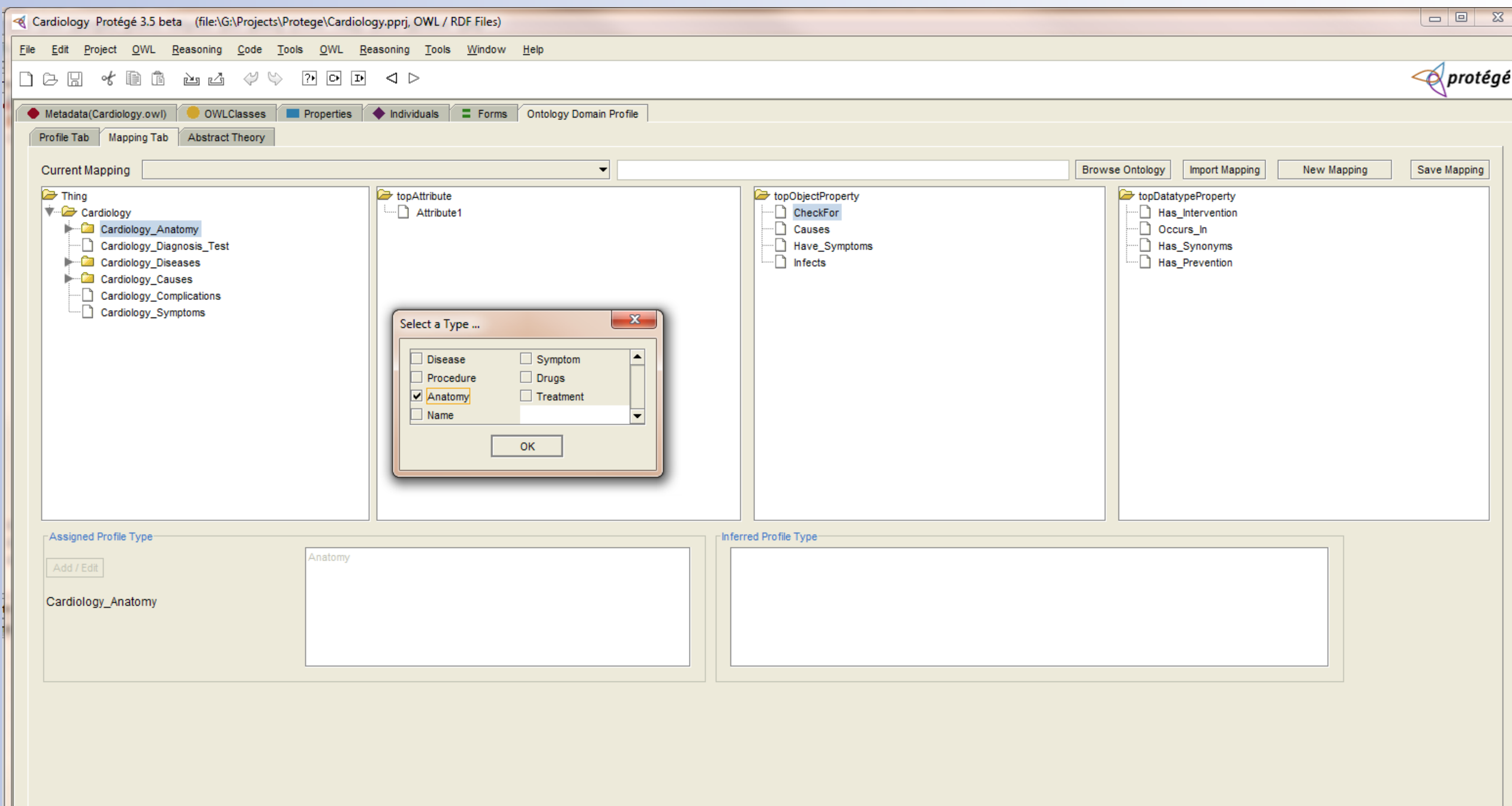
- Profile Tab - ODP Plugin-in for Protégé editor.
  - Define Domain Type Concepts.



Protégé Implementation – ODP

CSE 5810

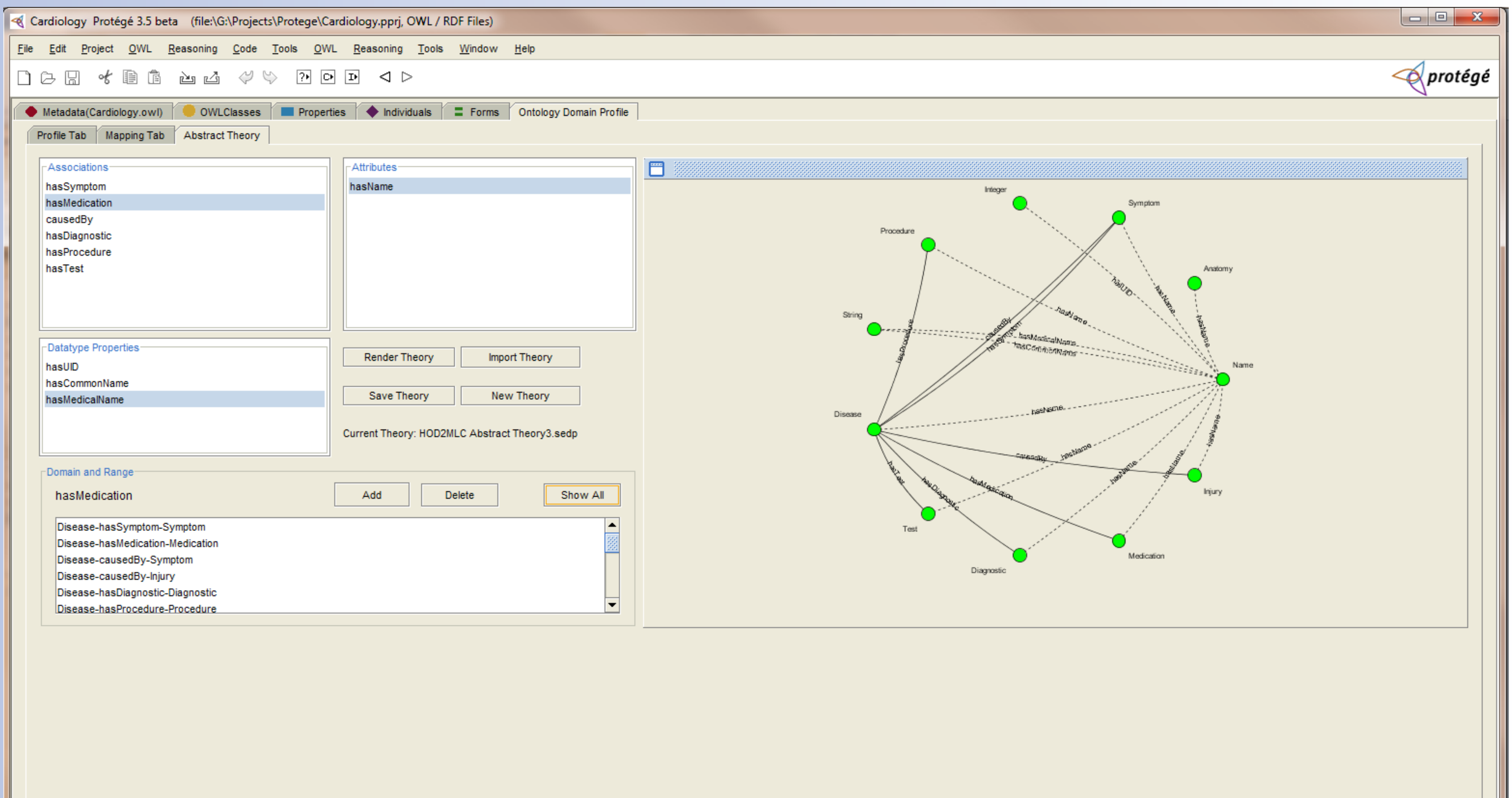
- Mapping Tab - ODP Plugin-in for Protégé editor.
  - Impose Type Concepts onto Domain Model Concepts



Protégé Implementation – ODP

CSE 5810

- Abstract Theory Tab - ODP Plugin-in for Protégé editor.
  - Construct the Abstract Theory.



Ontology Schema Associations

- OWL (with proposed extensions) provides *Structure* and *Semantics* for Representing Knowledge
- Meta Information about the Ontology itself is provided by Ontology Meta Vocabulary (OMV) Model:
  - Intended to Capture Meta-Data About the Ontology
- OMV provides Meta Information on
  - Domain – Domain Represented in the Ontology
  - Organization – Party Responsible for the Ontology
  - Knowledge Level – Formalness of the Ontology
  - Framework – Formal Language Used
  - Time – Time of Development
  - Location - Place of Development
  - Person etc. – Person Responsible for Development

OMV Model

CSE 5810

- Ontology
  - Meta Information of the Ontology
- Ontology Type
  - Category of the Ontology. E.g. Catalogues, Glossaries, Frames etc.
- Ontology EngineeringTool
  - Tool used for Development. E.g. Protégé, Swoop etc.
- Ontology Domain
  - Domain Represented E.g. Disease, Symptoms, Injuries
- Ontology Task
  - Usage of the Ontology
- Organization
  - Who has Developed the Ontology. E.g. NIH, WFO, UCHC etc.
- Location
  - Where the Ontology has been Developed E.g. MD, CT etc.
- Ontology Syntax
  - Formal Language Syntax used for Implementing the Ontology

OMV Model – Part 1

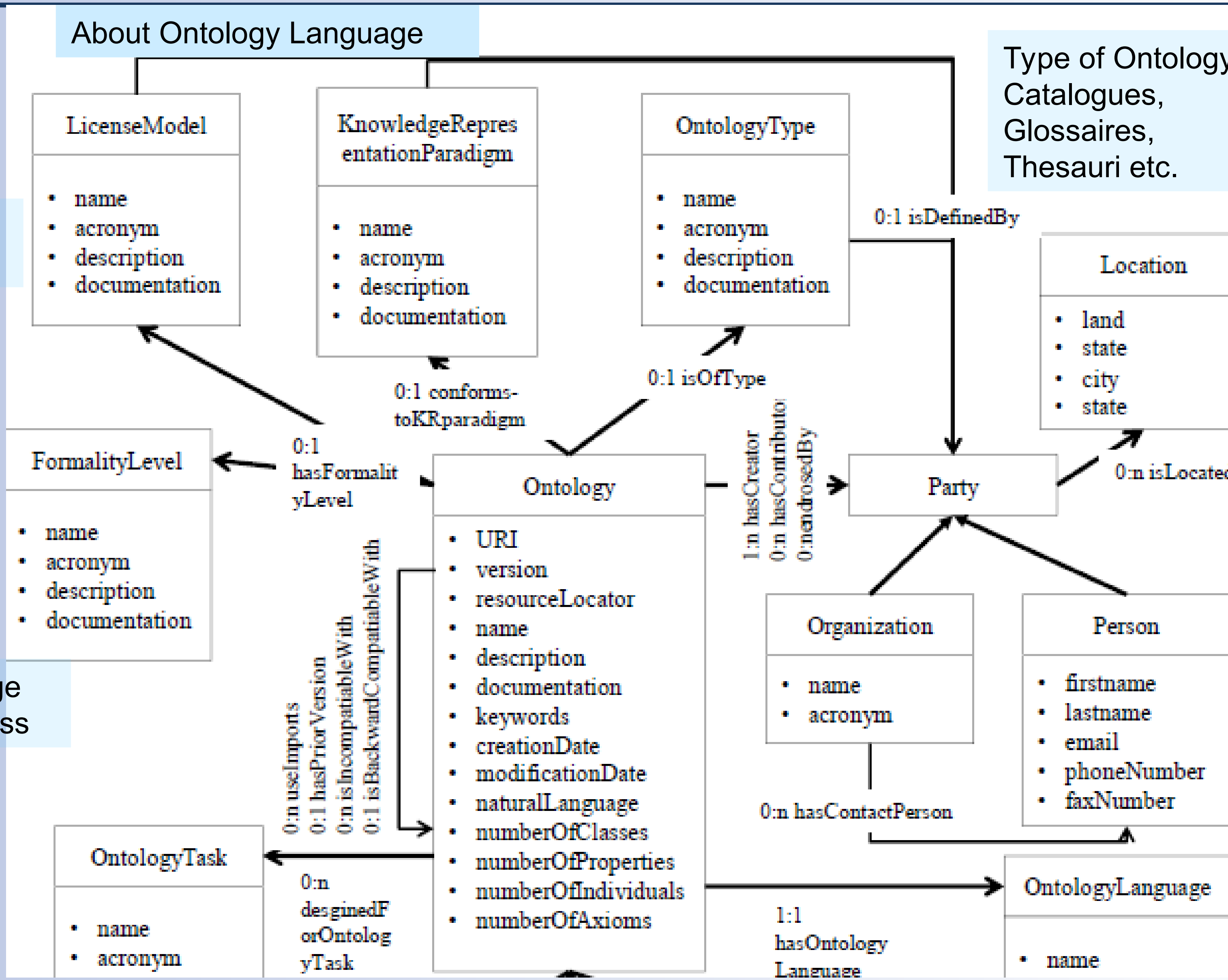
CSE 5810

About Ontology Language

Type of Ontology: Catalogues, Glossaires, Thesauri etc.

Usage Model

Knowledge Formalness



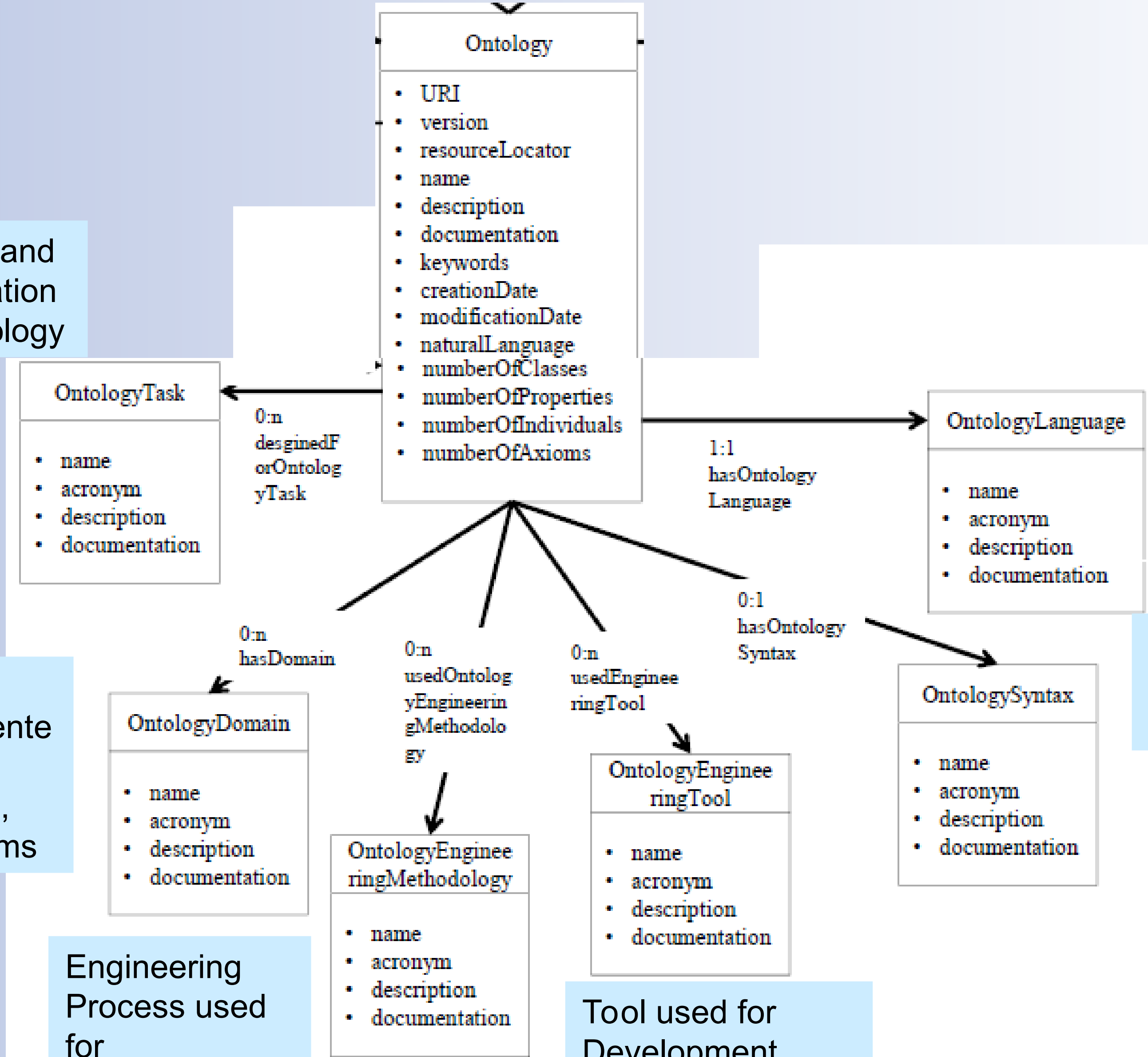
Meta Information About the Ontology

Party Responsable for Development

OMV Model – Part 2

CSE
5810

Usage and Application of Ontology



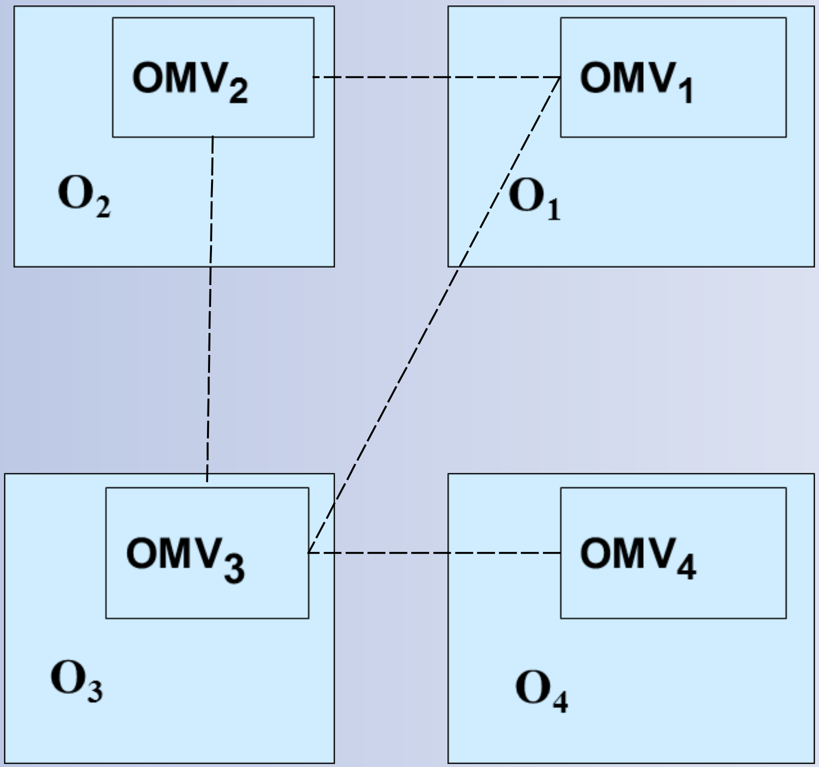
Language used for Implementation

Domain Represented. E.g. Disease, Symptoms

Engineering Process used for Development

Tool used for Development. E.g. Protégé, Swoop

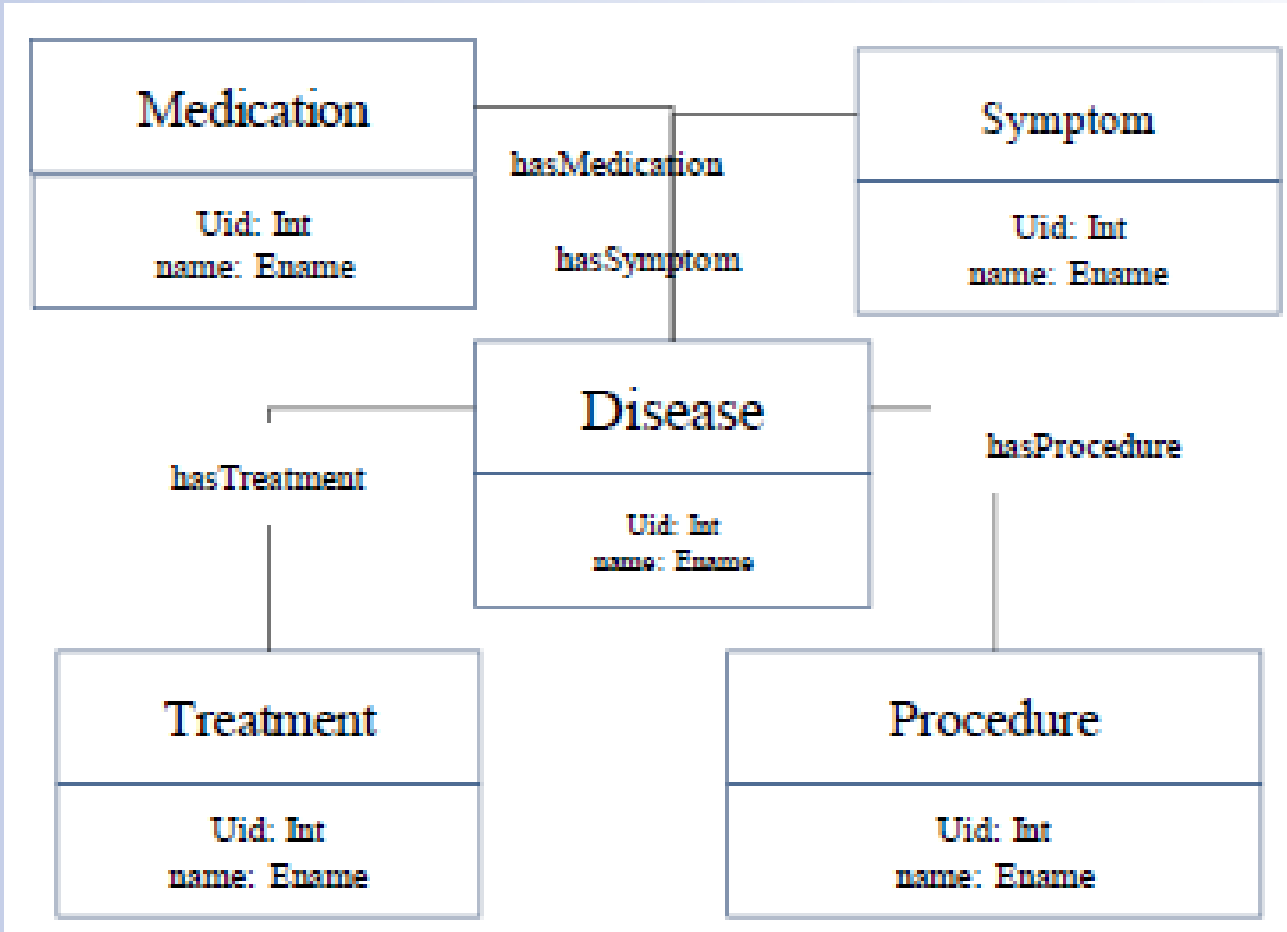
Schema Associations Using OMV



- Concepts of OMV1, OMV2 and OMV3 are Interconnected to form Ontology Schema Associations.
- OMV is Instantiated and Attached to Each Ontology
  - OMV2 and OMV3 can be Imported into Ontology OMV1 to build Ontology Schema Associations

How Does this Work?

- Recall UML/OWL Classes and Domain Profile



- How Do these Get Realized at Schema Level?

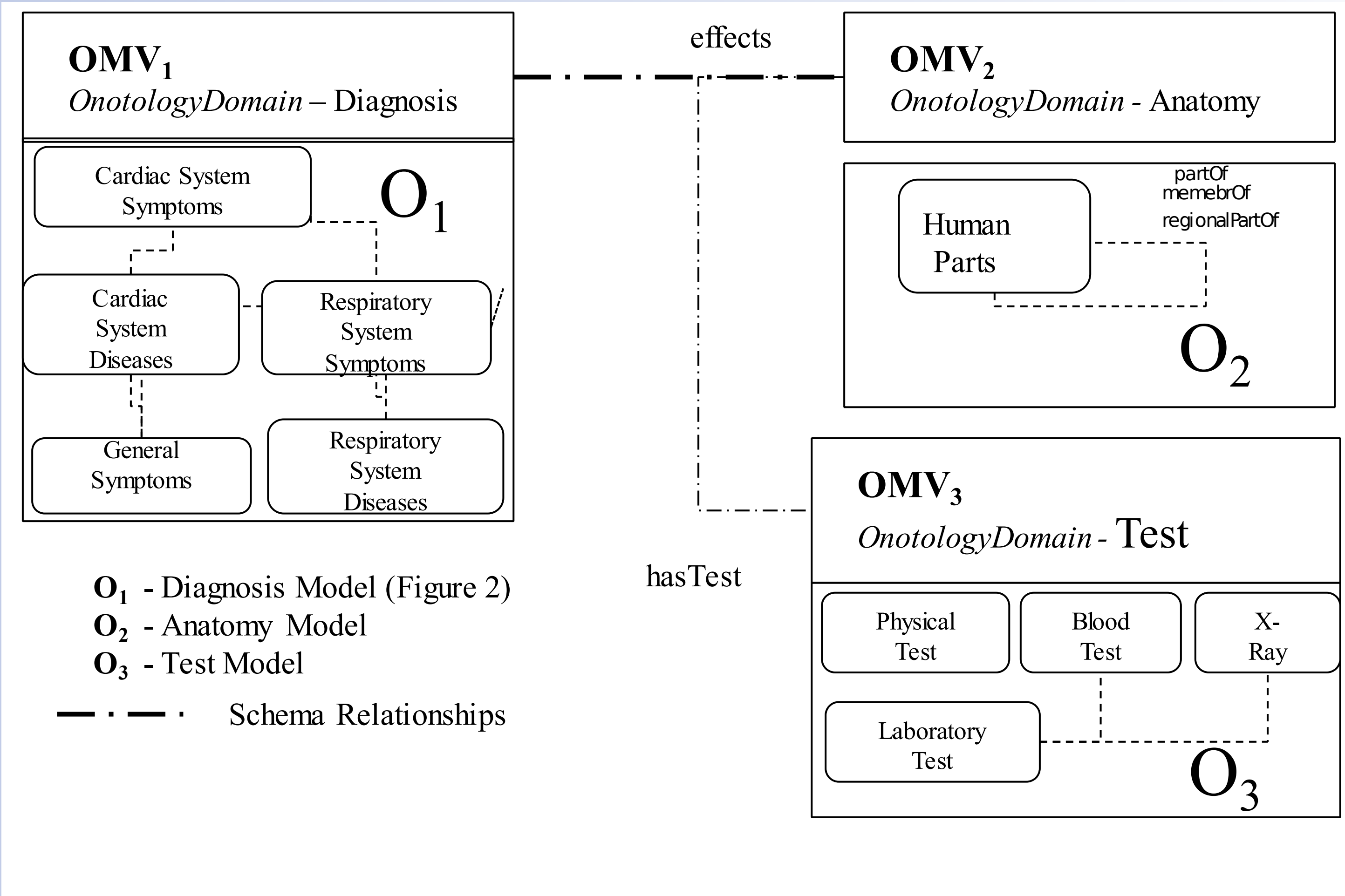
Schema Associations Using OMV

CSE 5810

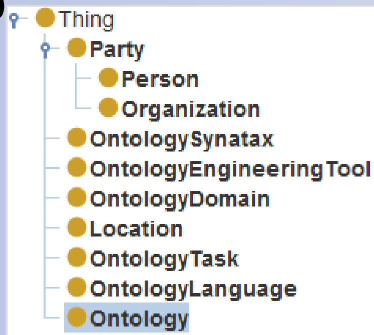
- Objectives:
  - Separate the Abstractions
  - Related the Ontologies
- Consider Three Different Ontologies
  - Diagnosis Ontology (O_1):
    - Defined from Perspective of Diagnosis
    - OMV_1 :OntologyDomain – Diagnosis_Ontology
  - Anatomy Ontology (O_2):
    - Designed from Perspective of Human Body Structure
    - OMV_2: OntologyDomain – Anatomy_Ontology
  - Test Ontology (O_3):
    - Designed from Perspective of Tests to be Ordered
    - OMV_3: OntologyDomain –Test_Ontology

Schema Associations Using OMV

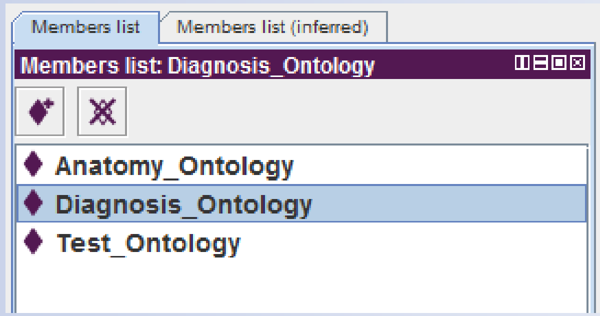
CSE
5810



Implementation: Schema Associations

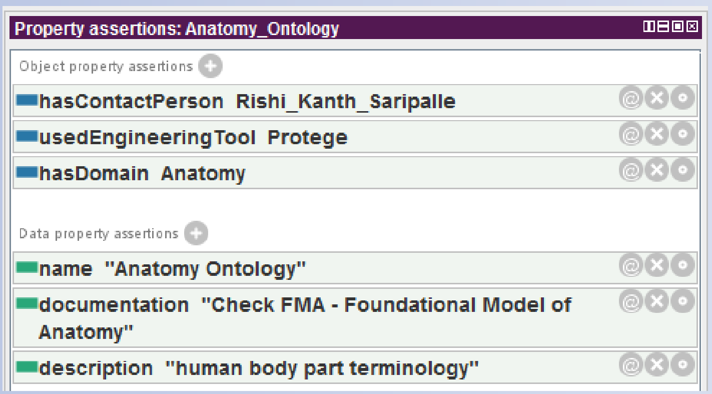


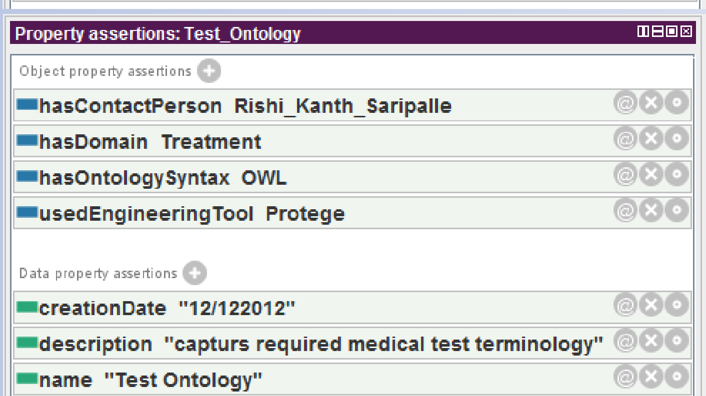
Step - 1



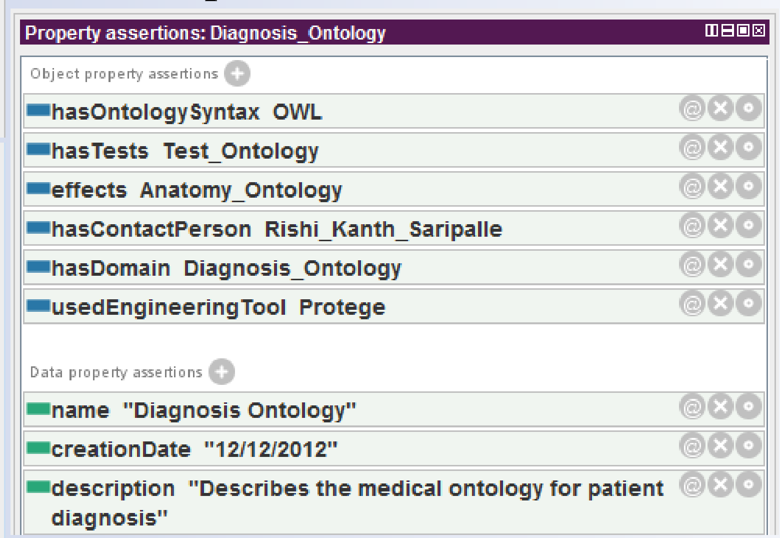
Step - 2

- Procedure
  - Step -1: Realize OMV model in OWL using Protégé
  - Step -2: Initialize OMV model for each Ontology Model
  - Step-3: Interconnect the defined OMV Concepts





Step - 3

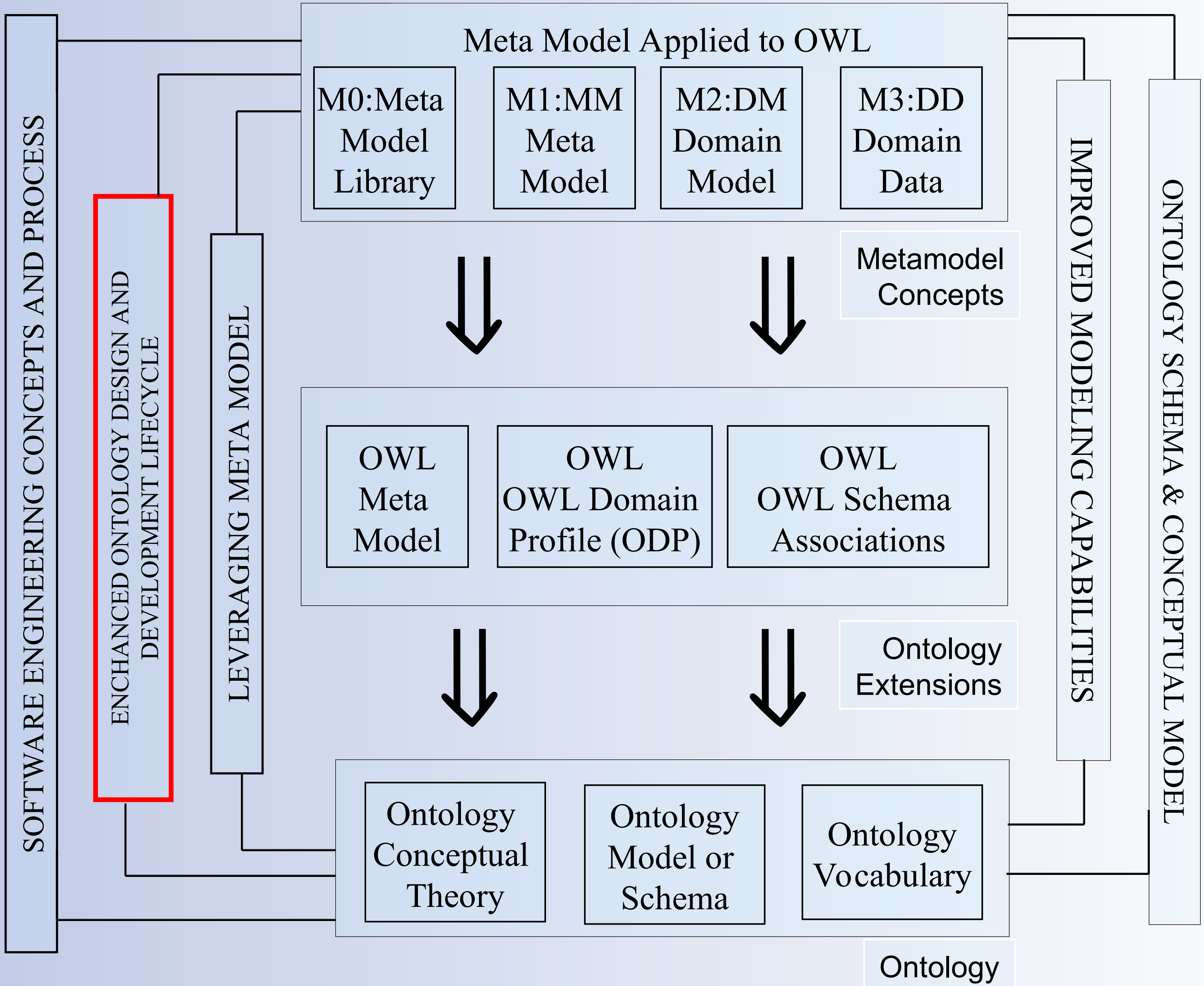


Related Work – Ontology Modeling

- Horrocks, I., Sattler, U., & Tobies, S. (1999) Practical reasoning for expressive description logics. *Proc. of the 6th Intl. Conf. on Logic for Programming and Automated Reasoning*, 161–180.
  - Hints that Ontology Vocabulary are Represented as Class to Exploit Reasoning Algorithm
- D. Djuric, D. Gaševic, V. Devedžic, Ontology Modeling and MDA, Journal of Object Technology, Vol. 4, pp. 109-128, 2005
  - Proposes the ODM, which is an instance of MOF and equivalent to UML
- K. Baclawski., M.M. Kokar, A.P. Kogut, L. Hart, E.J. Smith, J. Letkowski, and P. Emery: Extending the Unified Modeling Language for ontology development, Software and System Modeling, Vol. 1, pp. 142-156, 2002
  - Illustrates the mapping between OWL and UML ignoring semantics
- B. Motik: On Properties of Metamodeling in OWL, Proc. Of the 4th Intl. Semantic Web Conf., 2005
  - Proposes Metamodeling of ontologies using OWL DL with extended semantic
- Kuhn, W. (2010). Modeling vs Encoding for semantic web, IOS Semantic Web-Interoperability, Usability, Applicability,1(1), 11-15.
- Gruber, R.T. (2005). Toward principles for the design of ontologies used for knowledge sharing. Intl. Journal Human Computer Studies, Vol. 43, pp. 907-928.
  - Both Authors Emphasize that Ontologies Lack Formal Modeling Approach

Where are we in Overall Process?

CSE 5810



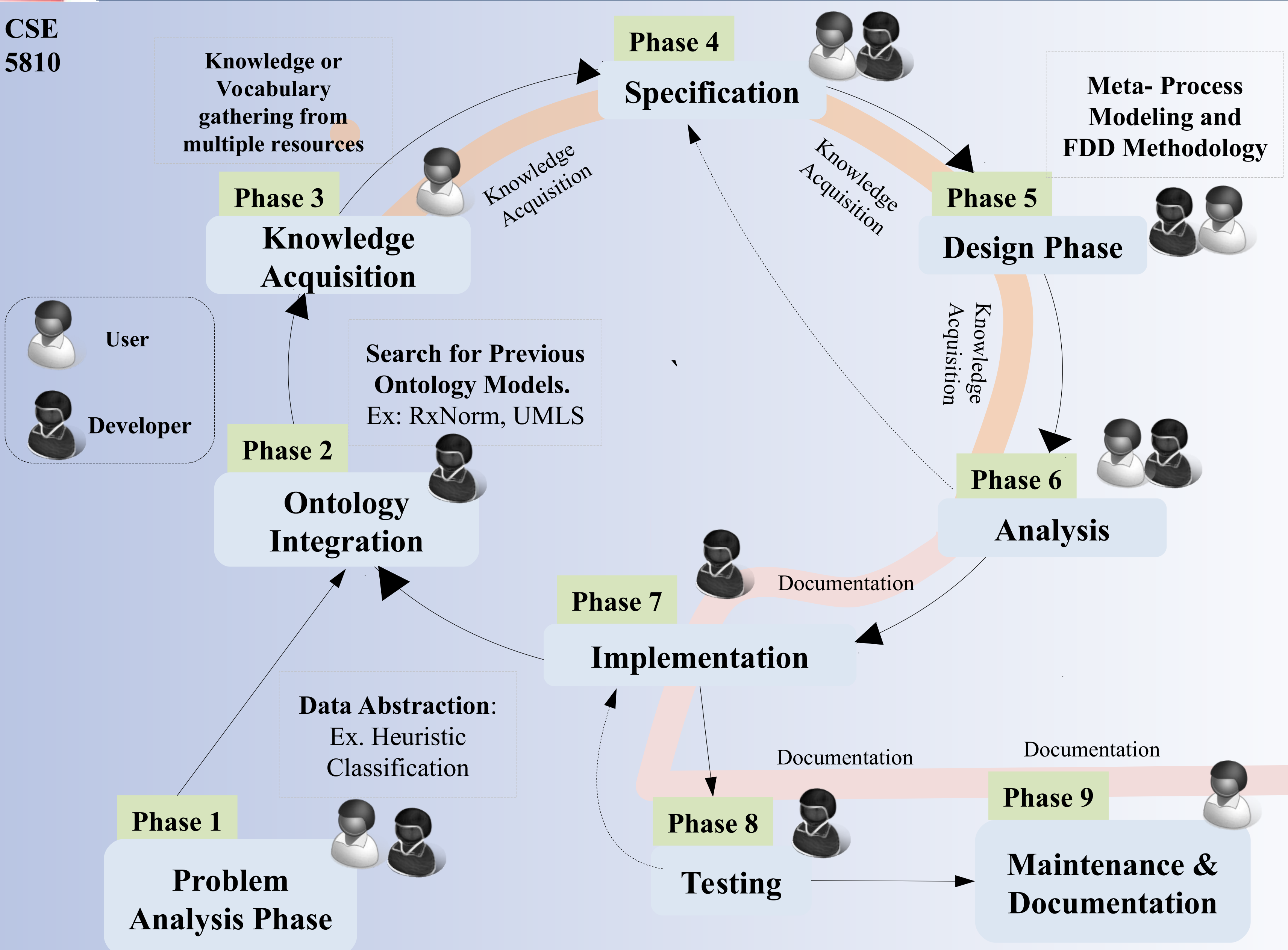
CSE 5810

Hybrid Ontology Design & Development Model with Lifecycle – HOD²MLC

- Objective
  - Narrow the Gap Between Ontology Design and Software Engineering
    - Define a Ontology Design and Development Model (ODDM) by Leveraging Software Development Process (SDP)
  - Final Outcome - Ontology Abstract Theory, Ontology Domain Model(s) and Ontology Vocabulary
    - Employing the Proposed OWL Extensions
- **HOD²MLC**
  - Agile Methodology – Iterative and Incremental Process
  - 9 Phases and 2 Feedback Loops
  - Represents the Required Stakeholder (Ontology Designer, Physician, Clinical Researcher, etc.) for Each Phase

HOD²MLC Model

CSE 5810

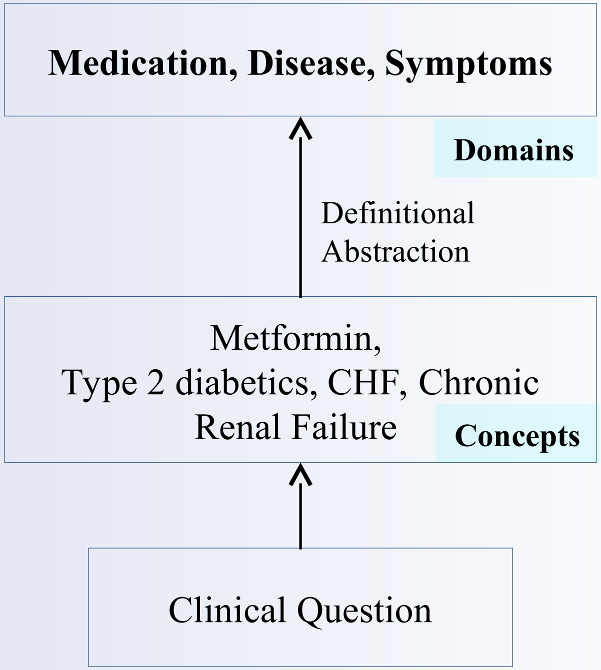


HOD²MLC – Problem Analysis Phase I

CSE 5810

- Objective
  - Identify Problem, Domains involved and Reason to Develop Ontology Models
  - Similar to Requirements Phase in many SDP
- Methodology
  - Employs Abstraction Techniques - Identify
    - Concepts, Domains & Associations
- Result- Identify
    - Domains, Concepts and Relationships between them

- Sample Clinical Question
  - How does metformin used for glucose control in type 2 diabetics effect the incidence and natural history of CHF and Chronic Renal Failure or stable Angina?



HOD^{2}MLC – Integration Phase II

- Objective
  - Identify Reusable Ontology Meta-Models, Domain Models and Ontology Vocabulary Methodology
- Methodology
  - Automated or Manual Search for Ontology Repositories
- Result
  - Reusable Ontology Modules
    - Abridge Semantic Interoperability
- Example
  - Reusable Vocabulary in BMI:
    - Standard Terminologies
      - LOINC – Vocabulary for Laboratory Codes
      - RxNorm – Medications
      - ICD – Vocabulary for Diseases, Symptoms, etc.

HOD^{2}MLC – Knowledge Acquisition Phase III

- Objective
  - Identify Modeling Concepts
    - Type Concepts, Domain Modeling Concepts and Vocabulary
- Methodology
  - Build Glossary of Terms (GT) Table holding the Concepts
  - Executed in Parallel until Design/Implementation Phase
- Result
  - Centralized GT Table Comprising of Concepts on Domains Involved
- Example
  - Sample GT Table:

Concepts	Definition
Anatomy	"A part of structural …."
Procedure	"A procedure, method, or technique…….. "
…….	……………..
…….	……………..

HOD^{2}MLC – Specification Phase IV

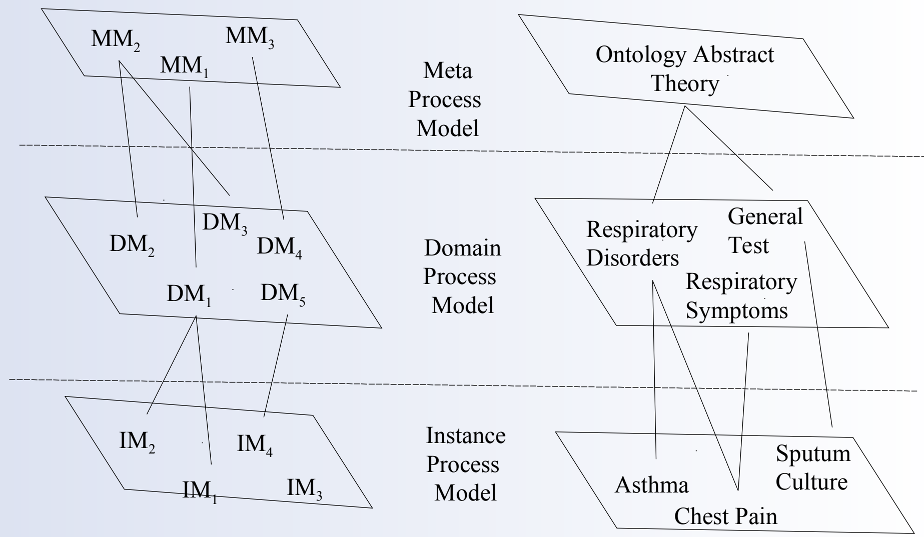
- Objective
  - Identify Boundaries on the Domains Involved and Concept Coverage
  - Similar to Specification Phase in any SDP
- Methodology
  - Collaboration and Cooperation between Stakeholders
- Result
  - Set of Constraints on Domains and its Concepts

- Sample Specifications for BMI
  - Capture Diseases of Mental Disorders, Respiratory System, Cardiac System, etc.
  - All concepts must have a *UID* and *MedicalName*
  - Concepts of Type Medication, Symptom, Procedure must be *disjoint*

HOD²MLC – Design Phase V

- Objective
  - Develop Domain Model(s) based on the Identified Domains and Specifications.
- Methodology
  - Implement Meta Process Modeling (MPM) Approach
  - Provides Abstraction Between Modeling Layers
    - Meta Models (MM) - hold Meta-Models
      - Ontology Abstract Theory
    - Domain Process Models (DM) – hold Ontology Domain Models
      - Ontology Domain Models
    - Instance Models (IM) – hold Instance Data
      - Ontology Vocabulary

- Hierarchical Representation of MPM Software Technique.

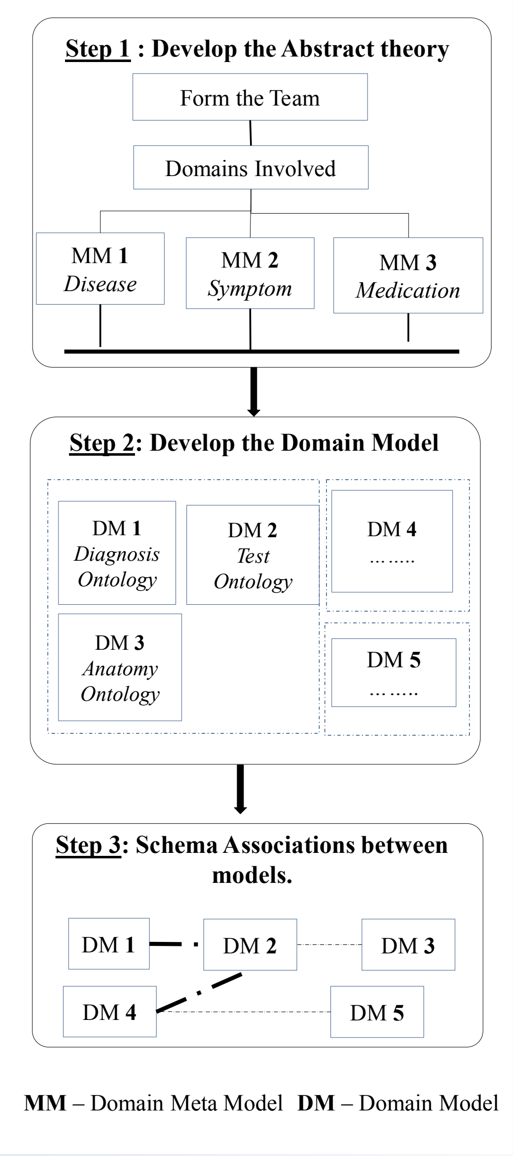


HOD^{2}MLC – Design Phase V

CSE 5810

- Methodology
  - Employ Feature Driven Development (FDD) to Achieve MPM
    - Top-Bottom Approach with Incremental and Iterative Process
    - Procedure
      - Identify Domains & Define Abstract Theory
      - Divide Theory to define *modular* and *reusable* Domain Model(s)
      - Interconnect Domain Model(s) – Schema Associations
- Result
  - Design Oriented Ontology Development
    - Ontology Abstract Theory, Domain Model(s)
    - Promote *Modularity, Adaptability, Reusability*

- Feature Driven Development



HOD²MLC – Analysis Phase VI

- Objective
  - Verify the Developed Domain Model(s) in Design Phase with *Specification* and User Requirements
- Methodology
  - Collaboration and Cooperation between Stakeholders
  - Feedback Loop Provides Flexibility
    - Accounting any Unexpected Changes
- Result
  - Well-Defined Structural and Semantic Domain Model

HOD²MLC – Implementation Phase VII

CSE 5810

- Objective
  - Implement Designed Ontology Abstract Theory, Ontology Domain Model/Schema(s), Ontology Vocabulary
- Methodology
  - Employ Modeling Framework
    - UML Profile or OWL+ODP for MPM Support.
    - Other Languages such as Frames, RDF, etc.
      - Based on Application Requirements
- Result
  - Realized Domain Model(s)

- Sample Implementation

(a) Diagnosis Ontology Model (b) Test Ontology Model (c) Anatomy Ontology Model

HOD²MLC – Testing Phase VIII

- Objective
  - Check for Consistency and Correctness of Realized Ontology Model
- Methodology
  - Employ Proven Frameworks and Methodologies
    - OWL Inference and Reasoner Algorithms
    - OWL Debugger
    - OWL Verification and Validation Framework
  - Rectify any Identified Bugs through Feedback Loop
- Result
  - Verified Domain Model(s) ready for Deployment

- Sample SPARQL Query

```
PREFIX hod2mlc:
<http://xmlns.com/foaf/0.1/>;
SELECT ?name
FROM
<http://www.ldodds.com/hod2mlc.owl>;
WHERE
{
        ?x
hod2mlc:hasMedicalName        ?
name.
}
```

HOD²MLC – Maintenance and Documentation Phase IX

- Objective
  - Documentation about the Methodology, Specification, Concepts
    - Source Citation, Definition, Version, etc.
- Methodology
  - Documentation
    - Use Conventional Approaches (e.g., Database, Text Notes, etc.)
  - Maintenance
    - Version Control using Existing Methodologies
      - Protégé Collaborative, SVoNT, etc.
  - Regular Performance Checks Similar to Software Applications.
- Result
  - Deployed Ontology Model(s) ready for Application Usage
- GT Table Documentation
  - Word Documents
  - Ontology Comments

HOD²MLC vs. Related Work

CSE 5810

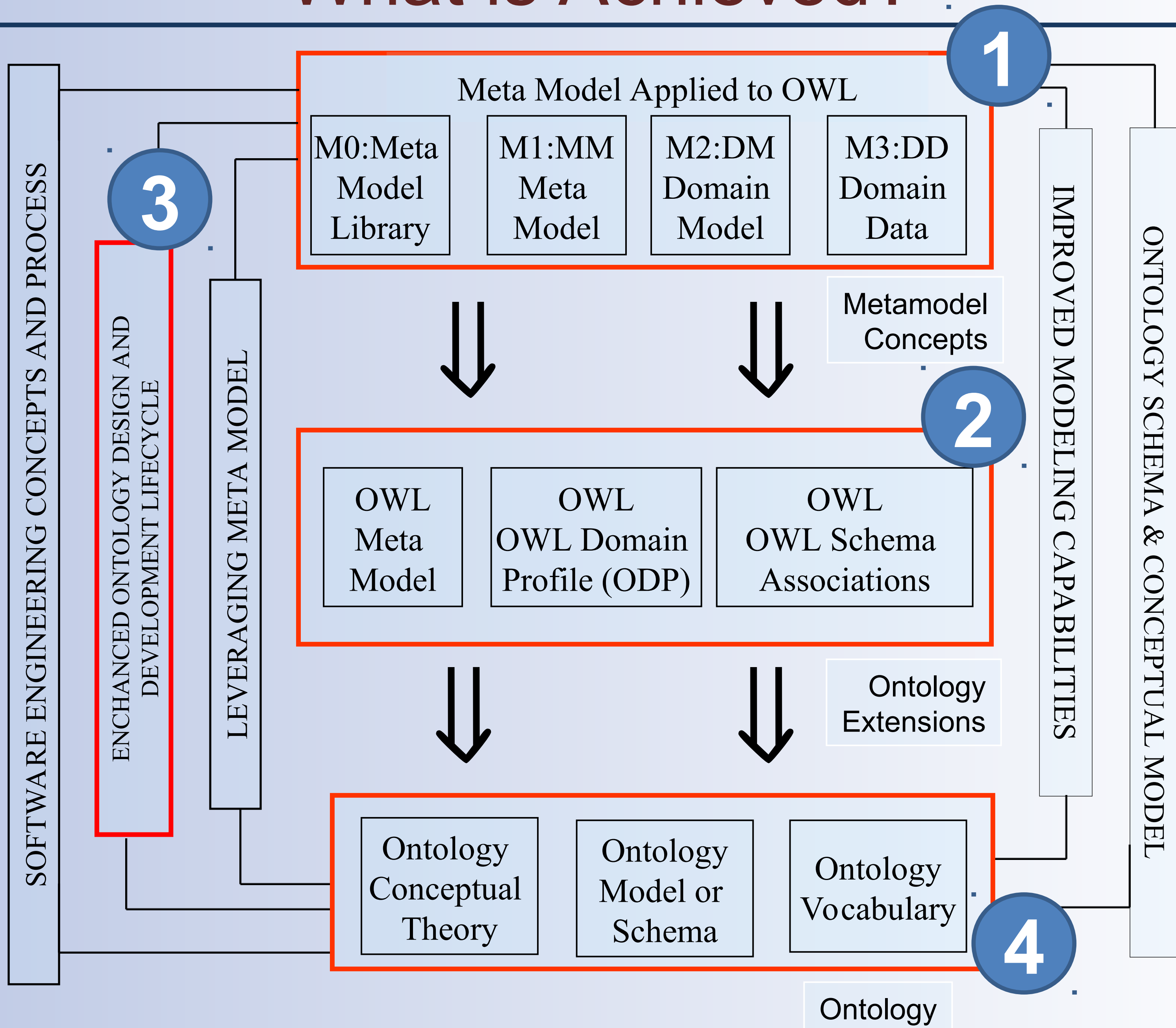
Phases	Ontology Life Cycle Models							
	Methontology	Fernandaz	EO Project	TOVE	Uschold	Noy	UPON	HOD²MLC
Problem Analysis	Partial	Full	Full	Full	Full	Full	Full	Full
Ontology Integration	Partial	None	Partial	Full	None	Partial	None	Full
Knowledge Acquisition	Full	Full	Full	Full	Full	Full	None	Full
Specifications	Full	None	Partial	Partial	Partial	Partial	Full	Full
Design	Partial	Partial	Full	Full	Full	Full	Full	Full
Analysis	None	None	Partial	None	None	None	Full	Full
Implementation	Full	None	Full	Partial	Partial	Partial	Full	Full
Testing	None	None	None	None	None	None	Full	Full
Maintenance / Documentation	Partial	None	Partial	Partial	None	None	None	Full
Model Adopted	Evolutionary	None	None	None	None	Iterative	Unified Process	Agile Process

HOD2MLC – Related Work

- M. Grüninger, M. Fox, “Methodology for the Design and Evaluation of Ontologies”, Proc. of Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI-95), August 1995.
- A. Gómez-Pérez, M. Fernández and A. J. de Vicente, “Towards a Method to Conceptualize Domain Ontologies”, Proc. of 12th European Conference on Artificial Intelligence Workshop on Ontological Engineering, August 1996.
- M. Uschold, “Building Ontologies: Towards a Unified Methodology”, Proc. of. 16th Annual Conf. of the British Computer Society Specialist Group on Expert Systems, September 1996.
- M. Fernández-Lopez, A. Gomez-Perez and N. Juristo, “METHONTOLOGY: from Ontological Art towards Ontological Engineering”, Proc. of AAAI Spring Symposium, pp. 33-40, 1997.
- Uschold M, “The Enterprise Ontology”, Journal of The Knowledge Engineering Review, Vol. 13, No. 1, pp. 31-89,March 1998.
- N. Noy and L. McGuinness, “Ontology Development 101: A Guide to Creating Your First Ontology”, Technical Report - Stanford Knowledge Systems Laboratory, March 2001.
- A. D. Nicola, M. Missikoff, and R. Navigli, “A Proposal for a Unified Process for Ontology building: UPON”, Proc. of 16th Intl. Conf. on Database and Expert Systems Applications (DEXA05), August 2005.

What is Achieved?

CSE
5810



Summary- Research Contributions

- UML Meta-Model to OWL
  - Addition of Abstraction Capabilities
  - Facilitate Early Stakeholder Interaction
  - Promote Domain Semantics Adaptability and Reusability
- Ontology Model and Schema: OWL Extensions
  - Aligns OWL with Object-Oriented Standards
  - Facilitate Model/Schema Level Design
  - Promote Model Based Ontology Integration
- Ontology Design and Development
  - Software Design Process For Ontologies
  - Comprehensive Ontology Development Methodology

CSE 5810

Ongoing and Future Work

- Ongoing Work
  - Integrate HOD2MLC into Protégé
  - Improve Performance of ODP UI and DomainProfileParser for Enhanced Performance
- Future Work
  - Encapsulate Contextual Knowledge
    - Capture the Context of the Knowledge Represented in the Ontology Models
      - For Example, Heart Attack hasCardiacSymtom Stroke
        - Is this Knowledge True for All Cases ?
        - Dependent on Patient Condition, Medications, History, etc.?
  - Need for Domain Meta-Model
    - Require Domain Specific Dedicated Meta-Model for Developing modular and reusable Health Care Ontology Domain Model(s)
      - For Example,
        - SQL Schema Language for Databases
        - UML for Object-Oriented Design and Modeling

Publications

CSE 5810

- **Published**
  - Rishi Saripalle, and S. Demurjian, “Towards a Hybrid Ontology Design and Development Life Cycle”. Proc. of Intl. Conf. Semantic Web and Web Services (SWWS), July, 2012.
  - Rishi Saripalle, and Steven A Demurjian, “Semantic Design Patterns using the OWL Domain Profile”, Intl. Conf. on Information Knowledge Engineering (IKE), July, 2012.
  - Michael Blechner, Rishi Kanth Saripalle and Steven A Demurjian, “A Proposed Star Schema and Extraction Process to Enhance the Collection of Contextual and Semantic Information for Clinical Research Data Warehouses”, Intl. Workshop on Biomedical and Health Informatics (BHI), October, 2012.
  - Timoteus B. Ziminski, Alberto De la Rosa Algarín, Rishi Saripalle, Steven A. Demurjian, Eric Jackson, “Towards Patient-Driven Medication Reconciliation Using the SMART Framework”, Intl. Workshop on Biomedical and Health Informatics, October, 2012.
  - Rishi Saripalle, S. Demurjian, S. Behre, “Towards a Software Design Process for Ontologies”, Proc. 2nd Intl. Conf. on Software and Intelligent Information, October, 2011.
  - Berhe, S., Demurjian, S., Gokhale, S., Maricial-Pavlich, J., Saripalle, R. “Leveraging UML for Security Engineering and Enforcement in a Collaboration on Duty and Adaptive Workflow Model that Extends NIST RBAC,” in Research Directions in Data and Applications Security XXV, July 2011, pp. 293-300.
  - Berhe, S., Demurjian, S., Saripalle, R., Agresta, T., Liu, J., Cusano, A., Fequiere, A, and Gedarovich, J., “Secure, Obligated and Coordinated Collaboration in Health Care for the Patient-Centered Medical Home,” Proc. of AMIA, November 2010.
  - Demurjian, S., Saripalle, R., and Berhe, S., “An Integrated Ontology Framework for Health Information Exchange,” Proc. of 21st Conf. Software Engineering and Knowledge Engineering (SEKE), July 2009.

- **Submitted**
  - Rishi Saripalle, Steven Demurjian and Alberto De La Rosa Algarin, “A Software Engineering Process for Ontology Design and Development through Extensions to ODM and OWL”, *in review*, Journal of SWIS, 2012.
  - Rishi Saripalle, Steven Demurjian, Micheal Blechner and Thomas Agresta, “HOD^{2}MLC – Hybrid Ontology Design and Development Model with LifeCycle”, *in review*, 2013.
  - Rishi Saripalle and Steve Demurjian, “Attaining Knowledge Interoperability using Ontology Architectural Patterns”, Book Chapter for Revolutionizing Enterprise Interoperability through Scientific Foundations, 2013.