Universidad Autónoma Metropolitana Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Proyecto Tecnológico
Reporte Final

Diseño de un sistema ontológico para la gestión de información académica universitaria

Francisco Pavón Gutiérrez
Matricula: 208200656

Asesora:
Dra. Maricela Claudia Bravo Contreras

Trimestre 17I

21 de abril de 2017

Yo, Maricela Claudia Bravo Contreras, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Firma

Maricela Claudia Bravo Contreras

Yo, Francisco Pavón Gutiérrez, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Firma

Francisco Pavón Gutiérrez

Resumen

En este reporte de proyecto de integración se describe la construcción de un sistema ontológico y su respectivo programa de alimentación, con la finalidad de tener una herramienta que lleve a cabo la gestión de información académica universitaria. El sistema ontológico se diseñó y construyó siguiendo una metodología desarrollada por la asesora del presente proyecto, la Dra. Maricela Bravo. Esta metodología enfatiza tres principios del diseño y construcción de ontologías: la claridad, la coherencia y la modularidad. Bajo estos fundamentos se obtuvieron resultados satisfactorios en la construcción del conjunto de ontologías.

En la primera parte del reporte, se detallan los pasos efectuados para el diseño de las ontologías. La herramienta utilizada para su construcción fue el Protege.

En la segunda parte, se explica el diseño y desarrollo del programa de carga que alimenta al sistema ontológico. El programa fue construido en el IDE Eclipse Neon, bajo el paradigma de la programación orientada a objetos en lenguaje Java. Este programa, en un primer paso, ejecuta la lectura de archivos Excel que contienen la información a procesar y, en un segundo paso, crea las instancias y sus propiedades para insertarlas en las ontologías.

Los archivos en Excel contienen la información sobre los horarios de los cursos regulares de las diferentes materias de la carrera de Ingeniería en Computación de la UAM Azcapotzalco. La información de los horarios incluye datos de: salones, grupos, materias, horas, días y profesores que imparten dichas materias.

El proyecto desarrollado consiste de un conjunto de ontologías y los programas de carga de datos, los cuales permiten contar con un sistema integral a través del cual es posible explotar de manera más precisa y efectiva la información de horarios. En la sección dedicada a la evaluación del proyecto, se puede apreciar el enorme potencial que conlleva la explotación de información usando un sistema ontológico.

Tabla de contenido

Índice de ilustraciones

1. Introducción

En esta sección se presenta el contexto del tema del proyecto terminal y, como motivación, el problema que se abordó con este proyecto, se explican brevemente algunos antecedentes de dicho tema y se justifica la elaboración del proyecto como solución al problema planteado.

a. Motivación

En toda universidad, el acceso a la información referente a las materias que cursan los alumnos de una carrera, como son horarios, salones, profesores, grupos, etc., es indispensable para el desarrollo de las actividades académicas diarias. Esta información, por lo general, se encuentra a disposición de los interesados, alumnos y personal académico, tanto en las páginas web de las universidades como en portales específicos. En la mayoría de los casos, estos portales solicitan la identificación previa del alumno para ingresar, ya que contienen información sensible del usuario como, por ejemplo, calificaciones o datos personales. Los alumnos a nivel licenciatura de la Universidad Autónoma Metropolitana Unidad Azcapotzalco (UAM Azcapotzalco) cuentan con esta información en el Módulo de Información Escolar de la página web de la universidad [1], al que pueden entrar tras identificarse proporcionando su matrícula y una contraseña. La matrícula identifica de manera única a cada alumno y se le asigna al quedar inscrito en la universidad.

Esta información se presenta ordinariamente de forma estática, es decir, que no se tiene la posibilidad de interactuar con ella para obtener datos más elaborados que los mostrados de forma directa. Para los alumnos, habitualmente, se expiden comprobantes de horarios de inscripción en los que se señalan las materias que van a cursar en un determinado periodo y los datos relacionados con estas materias. Para los profesores habrá, también, un horario con las materias que deben impartir y los salones, días y horas respectivos. Sin embargo, si se desea saber, por ejemplo, en qué salón se encuentra tal profesor dando clases a una hora precisa o la lista de materias que se imparten en tal aula un día concreto de la semana, puede ser difícil encontrar esta información de forma inmediata y habrá que buscarla indirectamente, dando rodeos que harían perder tiempo justo cuando se puede estar en una situación apresurada.

En el caso específico del Módulo de Información Escolar a nivel licenciatura de la página web de la UAM Azcapotzalco, se puede conocer la siguiente información acerca de cada una de las materias de la carrera en la que se encuentra inscrito el alumno: días, horas, salones, grupos y profesores que las imparten, así como los créditos, nombres y claves de cada UEA (en la UAM, “materia” es denominada Unidad de Enseñanza-Aprendizaje, UEA). Hay que hacer hincapié en que, para obtener esta información, el portal solicita la clave única de cada UEA y si se desconoce esta clave es imposible acceder los datos. En la ilustración 1 tenemos un ejemplo de la información que entrega dicho portal.

Horario

HORARIOS DE CURSOS DE EVALUACIÓN GLOBAL DEL TRIMESTRE 10I

Nombre de la UEA / Profesor	UEA	Grupo	Lunes	Martes	Miércoles	Jueves	Viernes
ESTRUCTURA DE LOS MATERIALES JOSE GONZALO HERNANDEZ CORTEZ	1113060	CBT83		19:00 - 20:30 E211		19:00 - 20:30 E211	
LABORATORIO DE ESTRUCTURA DE LOS MATERIALES ARTURO ALVAREZ GARCIA	1113061	CBT85			16:00 - 19:00 LQUI		
DISEÑO LOGICO I JOSE LUIS ALCANTARA NAVA	1121026	CFEL82	19:00 - 20:30 E109		19:00 - 20:30 E109		19:00 - 20:30 E109
LABORATORIO DE DISEÑO LOGICO I JOSE LUIS ALCANTARA NAVA	1121030	CFEL81				16:00 - 19:00 F302	
ESTRUCTURA DE DATOS CON ORIENTACION A OBJETOS IRMA FERNANDA ARDON PULIDO	1151008	CCT82	20:30 - 22:00 BABBAGE		20:30 - 22:00 BABBAGE		20:30 - 22:00 BABBAGE

Ilustración 1: Horario

Conviene agregar que, en el Módulo de Información Escolar, no se dispone únicamente de la información indicada, el alumno cuenta además con un historial académico y otros datos de tipo académico y personal que no tienen relación con el presente trabajo.

Ahora bien, supongamos que un alumno desea conocer el horario de un profesor determinado, es decir, la lista de materias que imparte con la respectiva información de salones, horas y días en los que da estas materias. A menos que el alumno se la pregunte personalmente al profesor, le será casi imposible conseguir esta información a través del módulo. Tendrá que ingresar las claves de todas las UEA de su carrera, guardar los datos y buscar al profesor en cada una de ellas.

Asimismo, si un alumno no tiene a la mano un plan de estudios con las claves de las UEA, no podrá obtener ningún dato ya que el portal sólo proporciona información por medio de dichas claves. No existe la posibilidad de realizar una búsqueda mediante los nombres de las UEA o de alguna palabra contenida o relacionada con dicho nombre, por si se desconoce el nombre completo, o cualquier otro dato relacionado con ellas, como podrían ser las horas en las que se imparten o el tronco (agrupamiento) al que pertenecen.

Cada periodo de estudios, que en el caso de la UAM es un trimestre, los alumnos deben armar sus horarios de cursos de manera previa a su inscripción. Estos horarios dependen de multitud de factores tanto personales como académicos, entre estos se encuentran las materias a las que se puedan inscribir, las horas a las que se imparten y la disponibilidad de cupo. Se recomienda armar más de un esquema de horario ya que, en el momento de la inscripción, como se da preferencia a los alumnos con promedios de calificaciones más altos, los grupos pueden agotar su cupo y un modelo de horario con una UEA “agotada” puede quedar inservible.

Los esquemas de horarios de cursos, como dijimos, dependen de muchos factores. En cuanto al aspecto personal, el alumno puede contar con un tiempo fijo para asistir a sus cursos debido, por

ejemplo, al tiempo de traslado a la universidad. En lo referente al ámbito académico, no todos los trimestres se imparten las mismas UEA, además de que la oferta de materias queda restringida a ciertas horas y cupos de los grupos. Por lo que, tomando en cuenta el mayor número de factores implicados, el propio Módulo de Información Escolar podría elaborar cierta cantidad de pronósticos de horarios de cursos y proponerlos al alumno antes de que éste se inscriba. El alumno revisaría estos pronósticos y podría pedir al módulo que considere diferentes factores y, quizás, distintas posibilidades por cada factor, a fin de generar los pronósticos que mejor se adapten a sus requerimientos personales. Para esto, se necesitaría que el módulo contara con una aplicación informática que, de forma interactiva, llevara a cabo esta tarea. Aún más, esta aplicación podría elaborar los pronósticos de horarios en el momento mismo de la inscripción, cuando los cupos de los grupos se encuentren variando.

La información que se necesita para tener estas y otras muchas facilidades en la gestión de la información académica, no difiere de la que se tiene actualmente en el Módulo de Información Escolar. En otras palabras, se cuenta con las bases de datos que tienen la información solicitada, pero no se tiene la posibilidad de explotarla como se ha propuesto en los ejemplos anteriores. Se requiere de una herramienta que pueda efectuar esta explotación. Esta herramienta debe tener, entre sus características principales, la posibilidad de hacer pronósticos con base en cierto número de datos previos, además de gran flexibilidad, precisión y especificidad en el manejo de la información. La tecnología que proporciona estas propiedades, y otras más, es la ontología informática. Un sistema ontológico puede generar información, es decir, producir conocimiento gracias a sus poderosas características.

En las próximas secciones de este trabajo, se describirá con detalle el alcance de un sistema ontológico y la forma en que se puede explotar la información contenida en él.

b. Antecedentes

A continuación, se presentan los documentos revisados durante la elaboración de la propuesta para el proyecto de integración, así como algunos otros examinados mientras se desarrolló el proyecto.

Proyectos de integración o terminales

1. Aplicación web de administración de horarios para estudiantes [2]

El objetivo principal consiste en procesar información académica que el usuario ingresaría previamente (ciclos escolares, materias y actividades de cada materia), para realizar una calendarización (programar las actividades del usuario) de forma posterior, utilizando el modelo Model-View-Controller (MVC) como patrón de diseño de software. Es una buena guía alterna para el desarrollo del presente proyecto porque recaba información académica y la procesa, sin embargo, la obtención de nuevos conocimientos a partir de datos suministrados con anterioridad no está incluida en el trabajo; además, el patrón de diseño difiere porque no utiliza ontologías.

2. Sistema de comparación de mapas curriculares basado en ontologías [3]

Tiene muchas coincidencias con el presente trabajo, pues diseña e implementa un modelo ontológico para la representación de mapas curriculares de planes de estudio a nivel licenciatura, a fin de conseguir varios objetivos, entre los que se destaca, por ser de interés para nuestro proyecto, ofrecer una orientación sobre el perfil de final de carrera. No obstante, nuestro trabajo centra su atención en la axiomatización de los componentes de las ontologías, a fin de permitir una posterior explotación de la información más orientada al uso cotidiano por parte de los alumnos.

3. Sistema para la comparación de documentos basado en ontologías [4]

Los objetivos finales están muy alejados de nuestro proyecto, ya que se dirigen al diseño e implementación de un sitio web que realice la comparación de documentos de tipo ontológico, para obtener porcentajes de similitud entre ellos. Sin embargo, existen fuertes coincidencias en la utilización del paradigma ontológico como fundamento para la explotación efectiva de la información, toda vez que las ontologías están diseñadas para estructurar y proporcionar un *significado* a los simples datos.

Tesis

1. Ontologías para servicios web semánticos de información de tráfico: descripción y herramientas de explotación [5]

La tesis aborda los problemas semánticos generados por la comunicación de las diversas tecnologías implicadas en el monitoreo del tráfico vial, así como la búsqueda de un lenguaje unificado por medio

de un modelo ontológico. Nuestro proyecto coincide en el uso de un conjunto de ontologías para generar información en un área del conocimiento en específico, pero no toca el tema central de la tesis. El desarrollo del modelo ontológico de la tesis es una buena guía para la construcción de ontologías.

2. Búsqueda Web basada en ontologías de dominio [6]

Esta tesis presenta una detallada guía para la construcción de ontologías orientadas a la búsqueda de conocimiento en la web, proponiendo dar un paso más allá de lo que realizan los motores de búsqueda cotidianos, los cuales no efectúan, entre otras cosas, un análisis semántico de las páginas web. Nuestro proyecto persigue, también, el objetivo de construir ontologías para obtener no sólo datos explícitos sino información generada por el uso de inferenciadores y razonadores, aunque el campo del conocimiento no sea tan amplio como la web.

c. Justificación

Como se mencionó en la Introducción, las ontologías son herramientas muy poderosas en el procesamiento del conocimiento. Son esquemas conceptuales rigurosos y exhaustivos que definen de manera muy precisa ámbitos del conocimiento; especifican sus relaciones, procesos y restricciones [7]. Gracias al rigor axiomático con que se construyen estos esquemas conceptuales, la entrega de información se vuelve mucho más precisa, si bien, su mayor ventaja reside en la posibilidad de efectuar inferencias a partir de datos previos, con lo cual se deja abierto el camino para generar más información conforme se vayan presentando nuevos requerimientos [6].

En el presente trabajo, se decidió diseñar y construir un conjunto de ontologías relacionadas entre sí, es decir, un pequeño sistema ontológico, para poder explotar de manera más efectiva la información académica relativa a la licenciatura de Ingeniería en Computación de la UAM Azcapotzalco. Los ejemplos de requerimientos de información mencionados párrafos arriba, aunque modestos, no se podrían cubrir con prontitud si no se usara un modelo ontológico. En otras palabras, se pretende que las preguntas planteadas en la Motivación sean contestadas por medio del uso de ontologías, a fin de evidenciar la facilidad del manejo del conocimiento cuando se tiene una herramienta vigorosa.

La estricta axiomatización es la clave para que en proyectos posteriores se pueda hacer uso de este esquema de ontologías. El proyecto está pensado para servir, por ejemplo, de cimiento sobre el que construir futuras capas de servicios que aprovechen otros aspectos de la información, o bien, que sirva de pieza en un conjunto ontológico modular más grande. Como se verá más adelante, los principios de diseño de la metodología escogida para la construcción del sistema apuntan precisamente a la modularidad, es decir, a que este conjunto de ontologías forme un módulo que se pueda reutilizar en otros proyectos.

Para que el sistema ontológico pueda ser aprovechado convenientemente se requiere que su diseño y construcción se realice con esmero, y atendiendo a ciertos principios que procuren una rigurosa definición de sus componentes. Los beneficios de lograr un conjunto de ontologías bien estructurado, claro y coherente se hacen manifiestos al momento de explotar la información que contiene. En el presente trabajo, se procederá a desarrollar, también, el programa de carga que alimente el modelo ontológico a fin de evaluar su consistencia con datos fidedignos.

2. Objetivos

Objetivo general

Diseñar y construir un sistema ontológico para la consulta y procesamiento de información académica relativa la licenciatura de Ingeniería en Computación.

Objetivos específicos

1. Diseñar y construir un conjunto de ontologías que definan los campos y las relaciones entre estos, relativos a la información de las UEA, créditos, horarios, salones, profesores y grupos de la licenciatura de Ingeniería en Computación.

2. Desarrollar un programa de carga, en lenguaje Java, para poblar las ontologías utilizando una base de datos por cada ontología.

3. Diseñar e implementar un conjunto de reglas de inferencia que soporten la consulta y el procesamiento de los datos contenidos en la estructura semántica de las ontologías, para evaluar el sistema anterior.

3. Marco teórico

a. Ontologías

La definición clásica de ontología se debe al científico Tom Gruber, presentada en 1993: “I use the term ontology to mean a specification of a conceptualization. That is, an ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents” [8].

Ilustrando lo anterior, se puede decir que, al enfocar la atención en un ámbito del conocimiento, o sea, un dominio específico de interés, se puede modelar este dominio describiendo los conceptos que lo componen y las relaciones que hay entre estos conceptos. La descripción de los conceptos debe ser rigurosa, se les debe definir exhaustivamente a fin de que no haya confusión entre un concepto y otro dentro del dominio que se está modelando. Cuando se tiene definido cada uno de los conceptos, se procede a definir las relaciones que existen entre ellos, también con la mayor precisión posible, de forma que al final se tenga dicho ámbito de interés del conocimiento completamente estructurado y listo para ser utilizado por agentes de software para explotar la información que contiene [9].

Las ontologías fueron creadas precisamente con la intención de llevar a cabo la tarea de definición conceptual de un dominio específico del conocimiento. Tal como lo asienta Gruber, la ontología es esa definición de los conceptos y sus relaciones.

En el presente proyecto, nuestro dominio de interés de conocimiento consiste en la información académica relativa a las materias de la carrera de Ingeniería en Computación de la UAM Azcapotzalco, si bien no se toma en cuenta toda la información referente a las UEA sino solamente los horarios en los que se imparten y los datos relacionados con estos horarios, como son los salones, profesores, grupos, etc. De esta forma, tendríamos que describir, en primer lugar, lo que significa “materia”, “horario”, “profesor”, etc., de modo que no haya viso de confusión entre un concepto y otro; y, posteriormente, definir las relaciones entre estos conceptos.

Después de definir la ontología, conviene detallar sus componentes básicos, los cuales son los individuos, las propiedades y las clases [10]:

- Individuos: Objetos concretos dentro del dominio de interés; por ejemplo, “Sor Juana Inés”, “Octavio Paz” o “Fernando del Paso” hablando de literatos. A los individuos también se les denomina *instancias*. Otros ejemplos de instancias son: “Primero Sueño”, “El laberinto de la soledad” y “Noticias del imperio” que son las obras respectivas de los literatos anteriores.

- Clases: Conceptos dentro del dominio de interés; se refieren a agrupaciones de individuos o, más específicamente, conjuntos que contienen individuos. Un ejemplo puede ser el concepto “Literato” que representa al conjunto de literatos concretos que se hallan dentro del dominio de interés que nos ocupe. Otro ejemplo puede ser “Persona”, el cual agrupa a

los seres humanos concretos. En este caso, la clase “Persona” contiene a la clase “Literato”, por lo que la clase “Persona” es una *superclase* para la clase “Literato” y ésta es una *subclase* de “Persona”. En este sentido, las clases pueden formar jerarquías a las que se les denomina *taxonomías*. Estas relaciones entre subclases y superclases pueden ser examinadas de forma automática por un programa llamado *razonador* como veremos más adelante.

- Propiedades: Relaciones binarias entre individuos. Las propiedades enlazan únicamente dos individuos, por ejemplo, **esAutorDe** puede unir al individuo “Octavio Paz” con el individuo “El laberinto de la soledad”. Si la propiedad se limita a enlazar individuos uno a uno, se tiene una propiedad *funcional*, tal como las funciones matemáticas en las que a un elemento del dominio le corresponde uno y sólo un elemento del rango. Un ejemplo de una propiedad funcional puede ser **tieneCURP**, la cual enlaza al individuo “Araceli Perez Diaz” con el individuo “PEDA971122MCSRZR06” (esta dato es ficticio). Además de funcionales, las propiedades pueden ser *simétricas*, *transitivas* o *reflexivas* (las explicaciones de estas cualidades exceden el presente trabajo toda vez que no se utilizaron para el mismo).

Como se puede apreciar, las clases y las propiedades se refieren a elementos de la lógica de conjuntos o matemática y, precisamente, las definiciones de los conceptos y sus relaciones deben estar definidas con precisión matemática, a fin de que los programas razonadores puedan efectuar inferencias (utilizando reglas del razonamiento lógico) para obtener información nueva a partir de los datos dados.

Las ontologías requieren de un lenguaje especial para describirlas, este lenguaje se llama OWL, Web Ontology Language, que es el estándar en el desarrollo de lenguajes para ontologías; está diseñado para las aplicaciones que necesitan procesar la información contenida en las ontologías [11].

Por otro lado, para el desarrollo de las ontologías es común utilizar un IDE denominado *Protege* [12], el cual permite el diseño y la construcción de las ontologías paso a paso con relativa sencillez. La siguiente imagen puede dar idea de algunos de los elementos que conforman una ontología; en este caso se refiere a la ontología “Persona” utilizada en el presente proyecto.

En el lado izquierdo de la imagen se puede apreciar el árbol taxonómico de las clases contenidas en la ontología, con la clase “Profesor” resaltada en azul. En la sección derecha, se encuentra la descripción de la clase “Profesor” o, mejor dicho, los elementos que conforman su definición exhaustiva y precisa a fin de evitar confusiones frente a los conceptos de las otras clases.

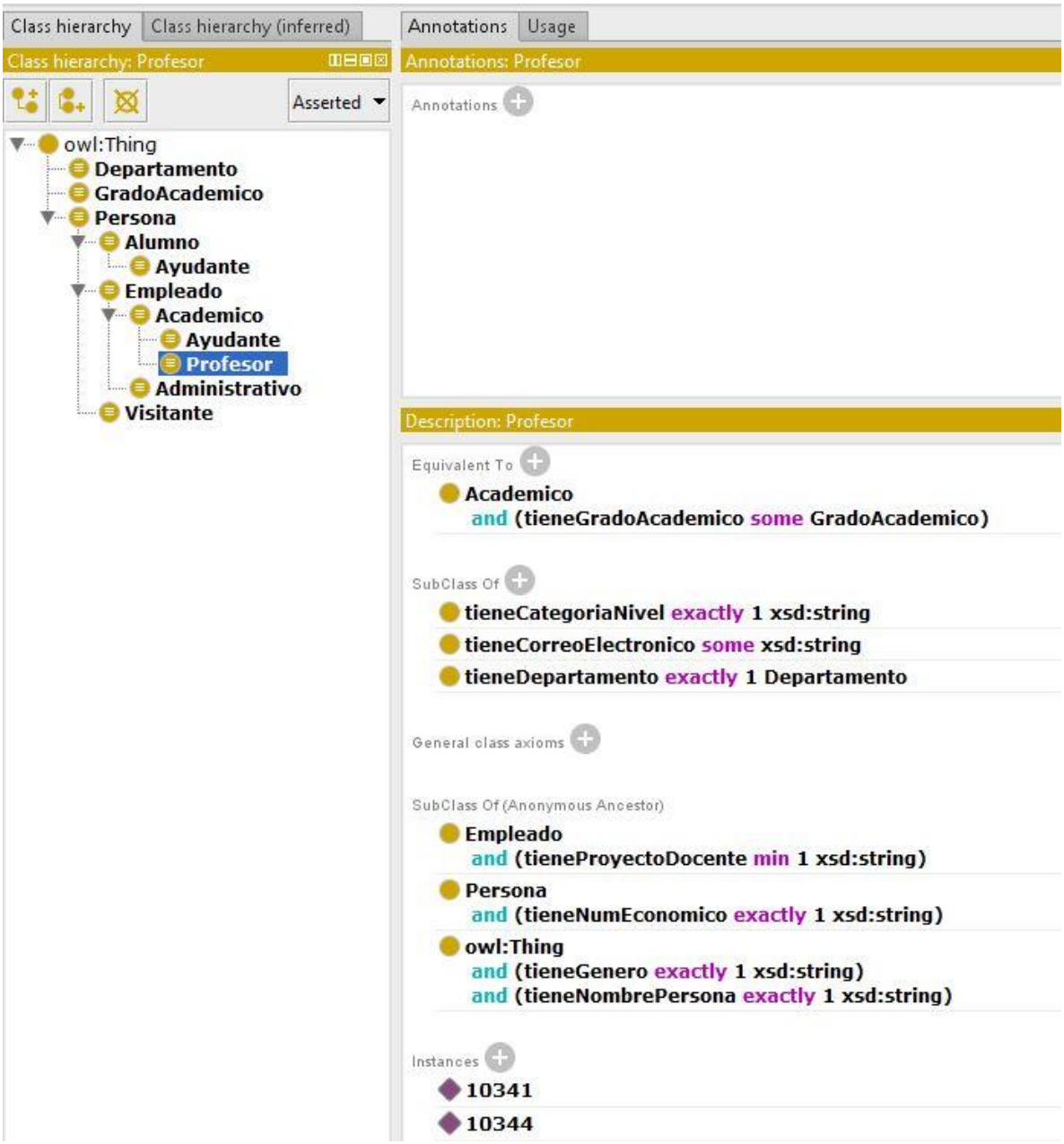


Ilustración 2: Clase "Profesor" en Protege

En la siguiente imagen, también de Protege, se muestra, en la parte inferior derecha, una de las instancias de la clase Profesor con sus propiedades. Los individuos de esta clase fueron *alimentados* a la ontología con el programa de carga desarrollado, también, en el presente proyecto.

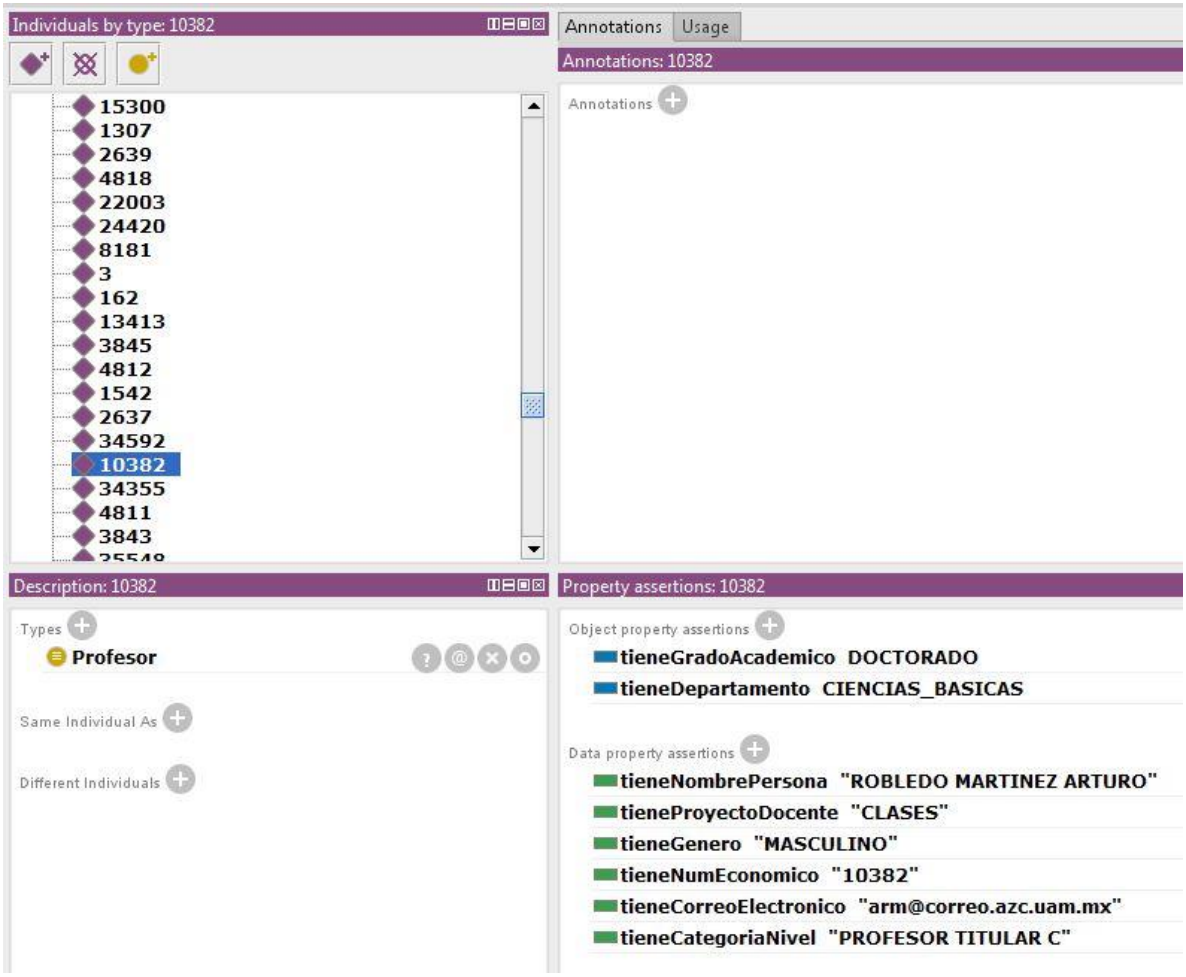


Ilustración 3: Instancia de la clase Profesor en Protege

b. Metodología

Para el diseño del conjunto de ontologías en este proyecto, se eligió la metodología de construcción de ontologías desarrollada por la asesora del mismo proyecto, la Dra. Maricela Bravo [13].

Las metodologías tradicionalmente usadas hacen poco énfasis en la consistencia final de las ontologías o lo toman como un componente a la par de otros elementos importantes, sin embargo, la consistencia es la clave principal para el diseño ontológico: “A key factor to achieve efficiency during ontology design is the principle of coherence… the design of ontologies is primarily focused on coherence, considering the creation of consistent ontology modules as an objective from the beginning of the design process” (cita del artículo al que se hace referencia).

La construcción de ontologías de manera consistente se logra siguiendo dos principios de diseño básicos: el principio de claridad y el principio de coherencia.

Principio de claridad	El propósito fundamental de las ontologías es comunicar los significados que están definiendo; esto se logra elaborando definiciones objetivas, declaradas en axiomas formales. En este punto, se prefiere la elaboración de definiciones “completas”, es decir, aquellas que incluyen las condiciones necesarias y suficientes, en vez de definiciones “parciales” que toman en cuenta sólo condiciones necesarias o sólo suficientes, pero no ambas.

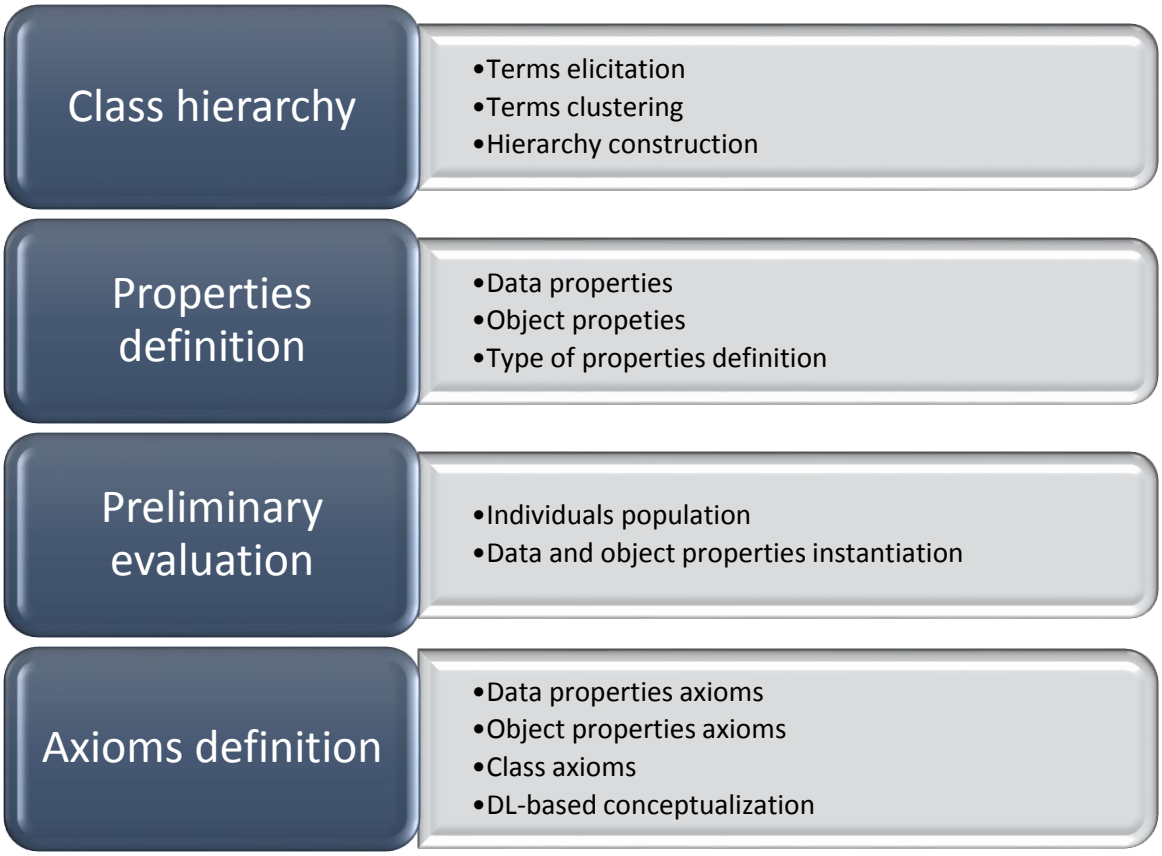
Principio de coherencia	Especifica que las definiciones deben ser sólidas o consistentes individualmente para no contradecir a los demás conceptos.

A los anteriores principios, se podría agregar uno más referente a la modularización:

Principio de Modularización	Enfatizar la construcción de ontologías modulares, en el sentido de tener ontologías “pequeñas” en lugar de enormes y complejas ontologías, que sería difícil manejar y casi imposible volver a utilizar. Dividir grandes ontologías en módulos operables, origina que estos módulos puedan ser reutilizados con facilidad posteriormente.

Es pertinente mencionar que, por regla general, cuando se van a construir sistemas de ontologías, se integran grupos de trabajo en los que hay expertos en el dominio de conocimiento sobre el que se va a trabajar, así como programadores y analistas con experiencia en el desarrollo e implementación de ontologías y aplicaciones que van a explotar la información de las ontologías.

La metodología seleccionada contempla cuatro fases principales, como se puede apreciar en el siguiente diagrama:



La elaboración de la jerarquía de clases se lleva a cabo, por ejemplo, planteando una serie de preguntas que se quieren responder con la información de las ontologías. A estas preguntas se les denomina “Preguntas de competencia”. A partir de estos cuestionamientos, se obtienen los términos importantes y se agrupan en conjuntos según algún grado de similitud. Las clases son los términos más importantes y a partir de los cuales se puede obtener la información dentro del dominio de conocimiento de que se trate.

En el caso que nos ocupa, la segunda y cuarta fases fueron realizadas una después de la otra. Después de que se definieron las propiedades y se construyó, también, una jerarquía de propiedades, se procedió a axiomatizarlas. Como dijimos en el apartado anterior, las propiedades son relaciones que enlazan a dos individuos. Específicamente, si la relación enlaza a dos instancias de clases se llama Object property; pero, si enlaza a una instancia con algún valor propio de esa instancia, se denomina Data property.

Como ejemplo de la Object property podemos tener a un alumno y su profesor: una instancia de la clase “Profesor” **esProfesorDe** una instancia de la clase “Alumno”. Aquí la Object property enlaza dos individuos, objetos, de clases diferentes, “Profesor” y “Alumno”; aunque, no es necesario que sean de clases diferentes, como en este otro ejemplo: una instancia de la clase “Alumno” **esCompañero** de otra instancia de la misma clase “Alumno”.

En cambio, como ejemplo de Data property, podemos tener el enlace entre un alumno y su edad (específicamente el *valor* de su edad): la instancia de la clase “Alumno” **tieneEdad** un valor numérico tipo entero comprendido entre 0 y 120. Aquí la instancia de la clase “Alumno” está enlazada con un valor tipo Integer que se encuentra dentro de un rango determinado.

La axiomatización de las Data y Object properties se refiere a definir las restricciones de cada propiedad. ¿Cuántas *edades* puede tener un alumno? O, ¿puede no tener *edad*? Entonces, la propiedad **tieneEdad** debe ser “exactly 1 Integer”, es decir, ni más de un valor ni menos de uno y debe ser tipo entero ya que no decimos “tiene 3.27 años”.

En relación con la axiomatización de las clases, se debe precisar si las clases pueden o no intersectarse o ser una subconjunto de otra. En otras palabras, debe quedar claro cuáles clases son “disjoint” y cuáles no.

La DL-based conceptualization (DL se refiere a Description Language) se lleva a cabo definiendo cada concepto de manera exhaustiva, es decir, considerando las condiciones necesarias y suficientes que se requieren para tener dicho concepto. Por ejemplo, para que un empleado sea precisamente un “empleado” se requiere que sea una “persona” y que “tengaUnNúmeroDeEmpleado”, es decir que, si tenemos una persona, la condición necesaria y suficiente para que esa persona sea empleado es que “tengaUnNúmeroDeEmpleado”. Esta propiedad evita que “empleado” pueda confundirse con otros conceptos que formen parte de la jerarquía de “persona”.

Por lo que respecta a la evaluación de la ontología, es necesario ejecutar esta fase con algunas instancias de cada clase. La evaluación es efectuada por un programa razonador que lleva a cabo numerosas tareas de clasificación taxonómica y de inferencia y consistencia de clases. Este programa se puede ejecutar desde la herramienta Protege, el cual nos avisará si hubiera algún error en la ontología.

4. Desarrollo del proyecto

a. Diseño del sistema ontológico

El primer paso para el diseño del sistema ontológico residió en el planteamiento del problema que se buscó resolver con dicho sistema. El problema consistió en una serie de cuestionamientos, algunos de los cuales se presentaron en la Motivación de la Introducción del presente trabajo, que demandaban información más precisa relacionada con los horarios de las UEA de la Ingeniería en Computación de la UAM Azcapotzalco. La siguiente tabla muestra algunos ejemplos de las preguntas referidas:

PREGUNTAS DE COMPETENCIA
¿En qué salón se encuentra tal profesor ahorita? **¿Cuáles son las UEA que se imparten en el salón F305 los días martes?** **¿Qué horario tiene tal profesor?** **¿Cuáles son las UEA que tienen en su nombre la palabra "programación"?** **¿Cuáles son las UEA que se imparten los martes y jueves de 7 pm a 8 pm?** **¿Cuáles son las UEA que pertenecen al Tronco General?** **¿Cuáles son las UEA que pertenecen al Tronco Básico Profesional?** **¿Cuántos créditos tiene tal UEA?** **¿Cuáles son las UEA que pueden llevar corregistro?** **¿Qué profesores imparten la materia de "Compiladores"?** **¿Cuántos créditos suman todas las UEA del área de Sistemas de Información?** **¿Cuáles son las claves de las UEA del área de Sistemas Embebidos?** **¿Cuáles son las horas y los días en que se imparten las UEA del tronco Inter y Multidisciplinar?** **¿Cuáles son las UEA que requieren autorización para inscribirse a ellas?** **¿Cuáles son todos los profesores que pertenecen a la División de Ciencias Básicas (ordenados alfabéticamente por el primer nombre)?** **¿Qué materias puedo inscribir en el siguiente trimestre si sólo cuento con tales horas, tales días de la semana?**

Después de elaborar las preguntas de competencia, se procedió a extractar los conceptos relevantes de entre dichas preguntas y a agruparlos según características de similitud, a fin de formar las clases diferenciando entre subclases y superclases. Asimismo, se buscó asociar las clases parecidas en conjuntos específicos. Los conjuntos de clases formaron, posteriormente, ontologías propias.

Al final de este proceso, se diseñaron diagramas con los grupos de clases similares. Es importante mencionar que en este paso, como en todas las etapas del desarrollo del proyecto, hubo que ser reiterativo y corregir una y otra vez los diagramas hasta obtener una versión final. La siguiente tabla muestra las clases resultantes:

CLASES (ontologías)
Grupo **Horario** **Persona** **Salón** **UEA**

El diseño, en el Protege, de las superclases y subclases fue el siguiente paso del proyecto. En esta herramienta, se llevaron a cabo las definiciones de cada una de las clases y el establecimiento de sus propiedades. En las siguientes imágenes, se presentan las taxonomías de cada una de las ontologías anteriores:

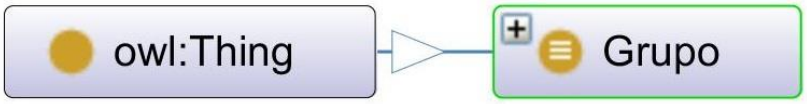


Ilustración 4: Ontología "Grupo"

Ilustración 5: Ontología "Persona"

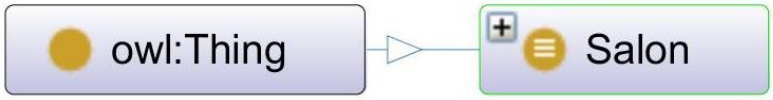


Ilustración 6: Ontología "Salón"

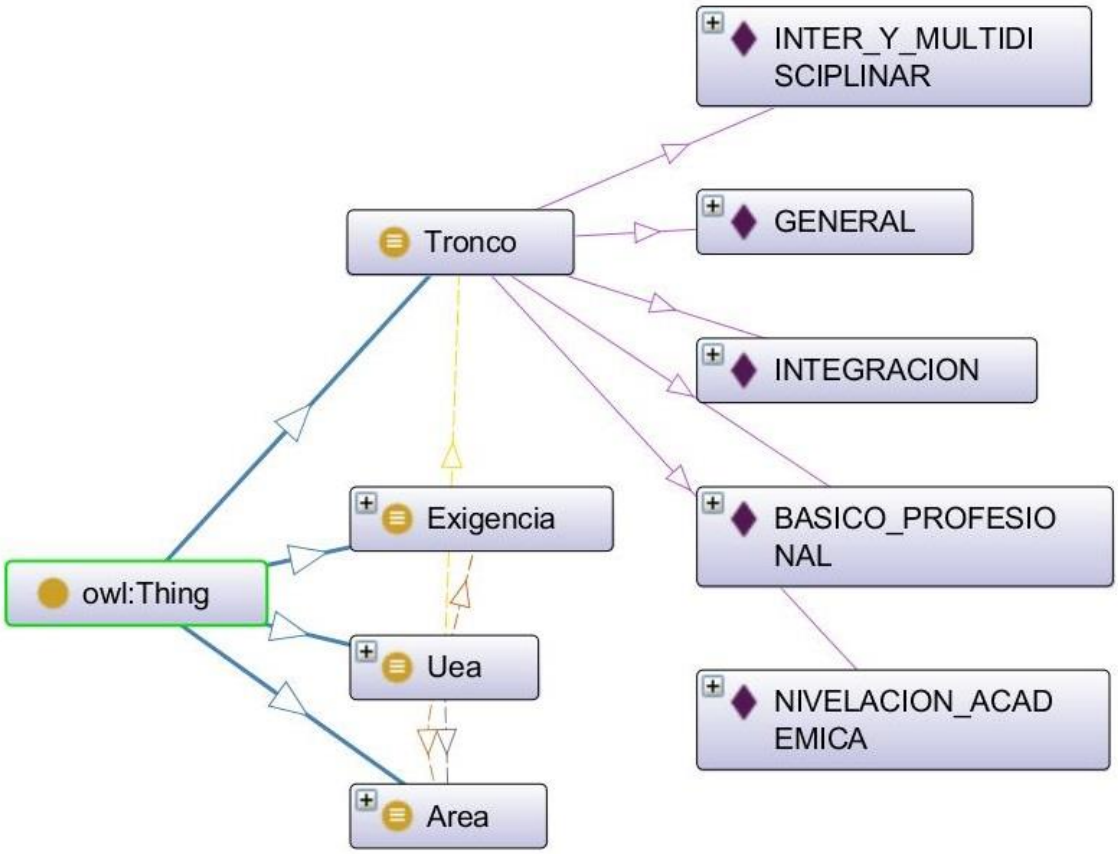


Ilustración 7: Ontología "UEA"

La imagen anterior, con la ontología UEA, muestra las instancias de la clase Tronco, que se refieren a los diferentes troncos en los que se agrupan las materias de la carrera. Casi todas las clases de las ontologías tienen su propia población, pero es imposible presentarlas en imágenes debido a la gran cantidad de instancias que incluyen. La base de datos de las UEA tiene 180 instancias y la de los profesores 416.

La ontología Horario engloba a las otras ontologías. Para elaborarla se realizó una importación de las otras ontologías de manera que, Horario acabó teniendo sus propias clases y propiedades además de las clases y propiedades de las ontologías importadas. Esta ontología, por sí sola, conformó el sistema ontológico completo. A continuación, la imagen con la taxonomía de sus clases:

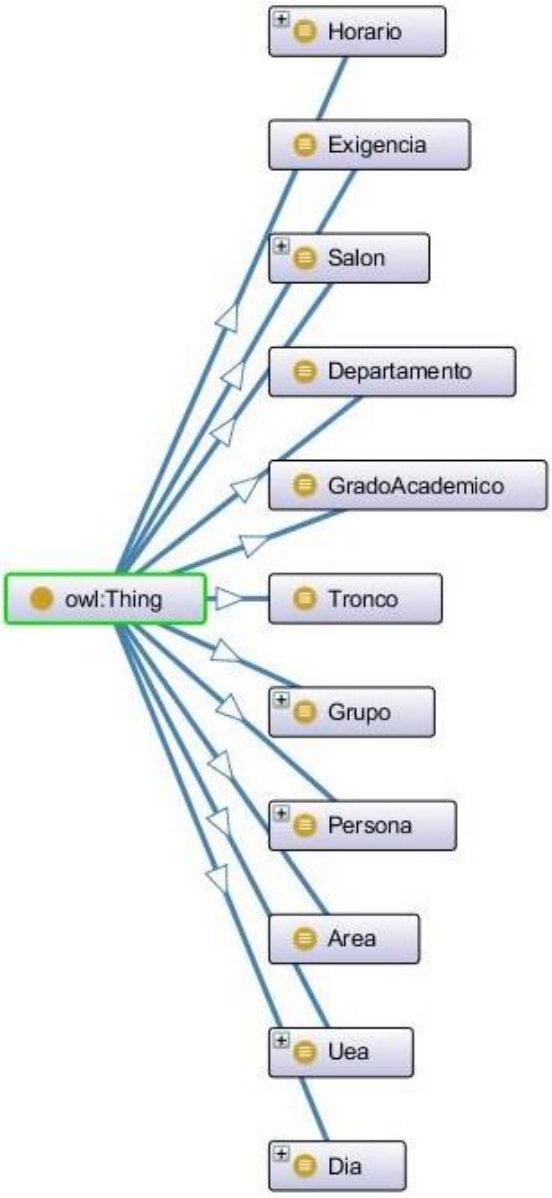


Ilustración 8: Ontología "Horario"

La intención de separar varias de estas clases en ontologías aparte (por ejemplo, “Salón” y “Grupo”), obedece al principio de modularidad. Estas ontologías, aunque muy simples y con poco contenido,

pueden ser reutilizadas en otro conjunto de ontologías debido a que son piezas desacopladas del todo, es decir, módulos independientes dentro del sistema.

A continuación, se identificaron las propiedades entre clases o, propiamente dicho, las relaciones semánticas entre los objetos de las clases. A modo de ilustración se muestran en las siguientes imágenes algunas Object properties de las ontologías Persona y Horario. En Persona, la Object property **tieneDepartamento** es una relación semántica entre la clase "Profesor" y "Departamento".

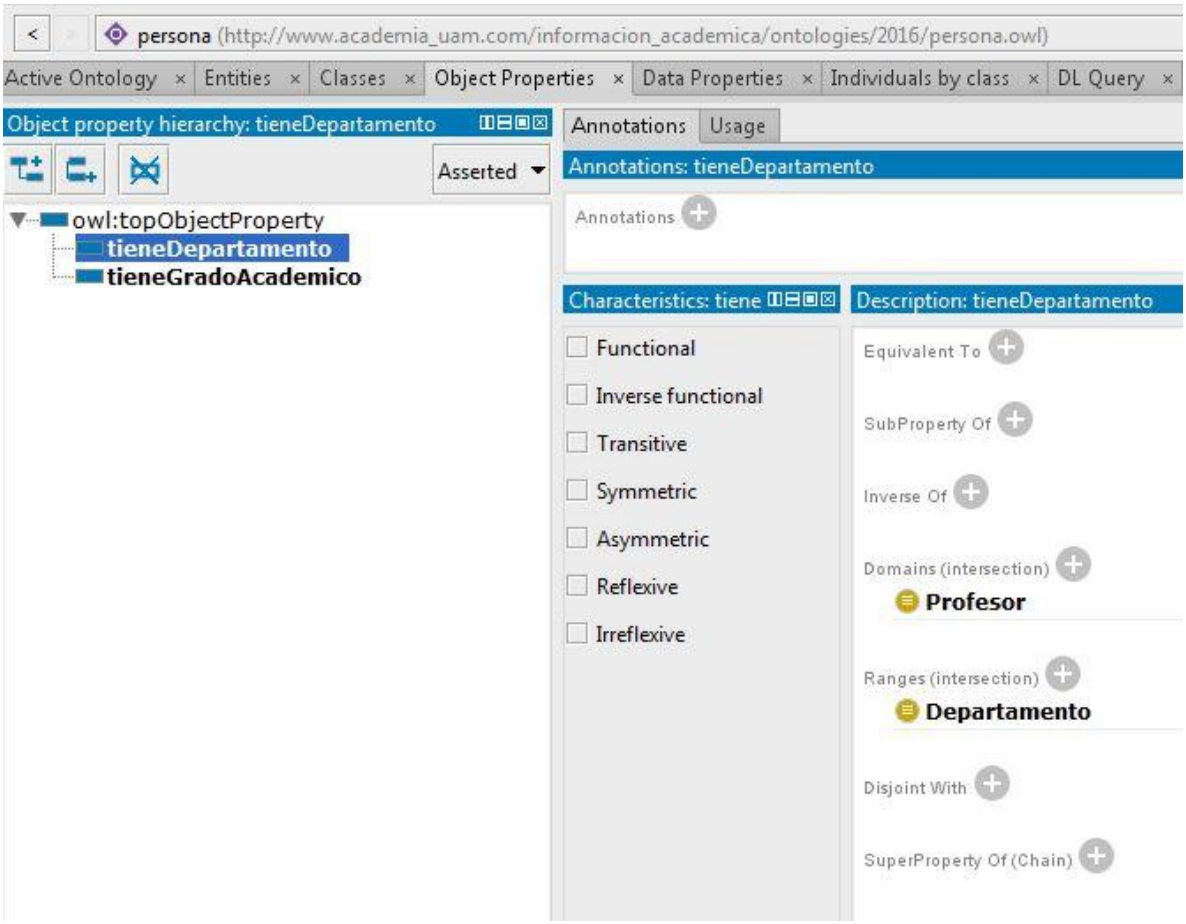


Ilustración 9: Object property en ontología Persona

Como se puede apreciar, esta propiedad tiene como "dominio" a la clase Profesor y como "rango" a la clase Departamento, es decir, que una instancia de la clase *Profesor* puede estar enlazada con una instancia de la clase *Departamento* por medio de la relación semántica **tieneDepartamento**.

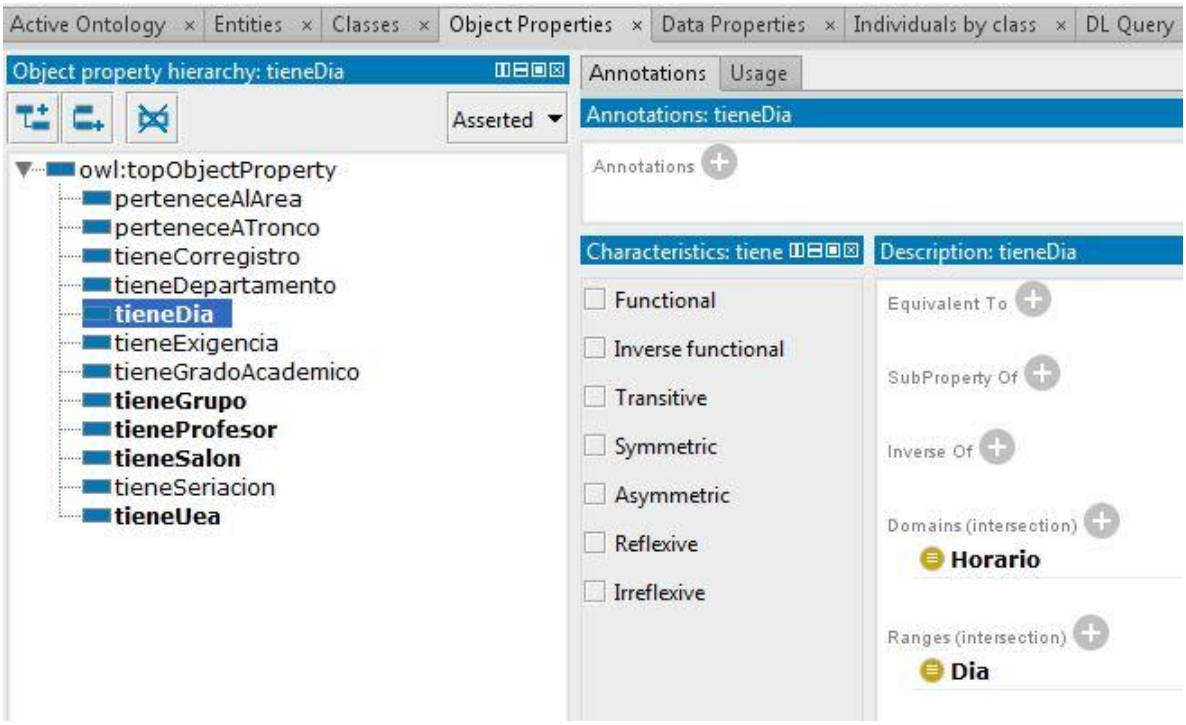


Ilustración 10: Object property en ontología Horario

Del mismo modo, en la imagen de la ontología Horario, se puede apreciar la taxonomía de las Object properties y el dominio y rango de la propiedad resaltada en azul.

Por último, se definieron y axiomatizaron las Data properties de todas las ontologías; así como las clases, estableciendo las clases disjuntas cuando fue necesario, y definiendo de manera completa, con las condiciones necesarias y suficientes, cada clase.

En resumen, para la definición de cada ontología se llevaron a cabo los siguientes pasos:

1. Identificar las clases de cada ontología
2. Establecer la jerarquía de clases
3. Definir las relaciones semánticas entre las clases
4. Establecer los axiomas de cada clase (Data properties)
5. Establecer los axiomas de las relaciones semánticas (Object properties)
6. Establecer los axiomas de las clases (Disjoint)
7. Definir de manera completa cada clase

En las siguientes imágenes se presentan las Data properties de las ontologías Persona y Horario.

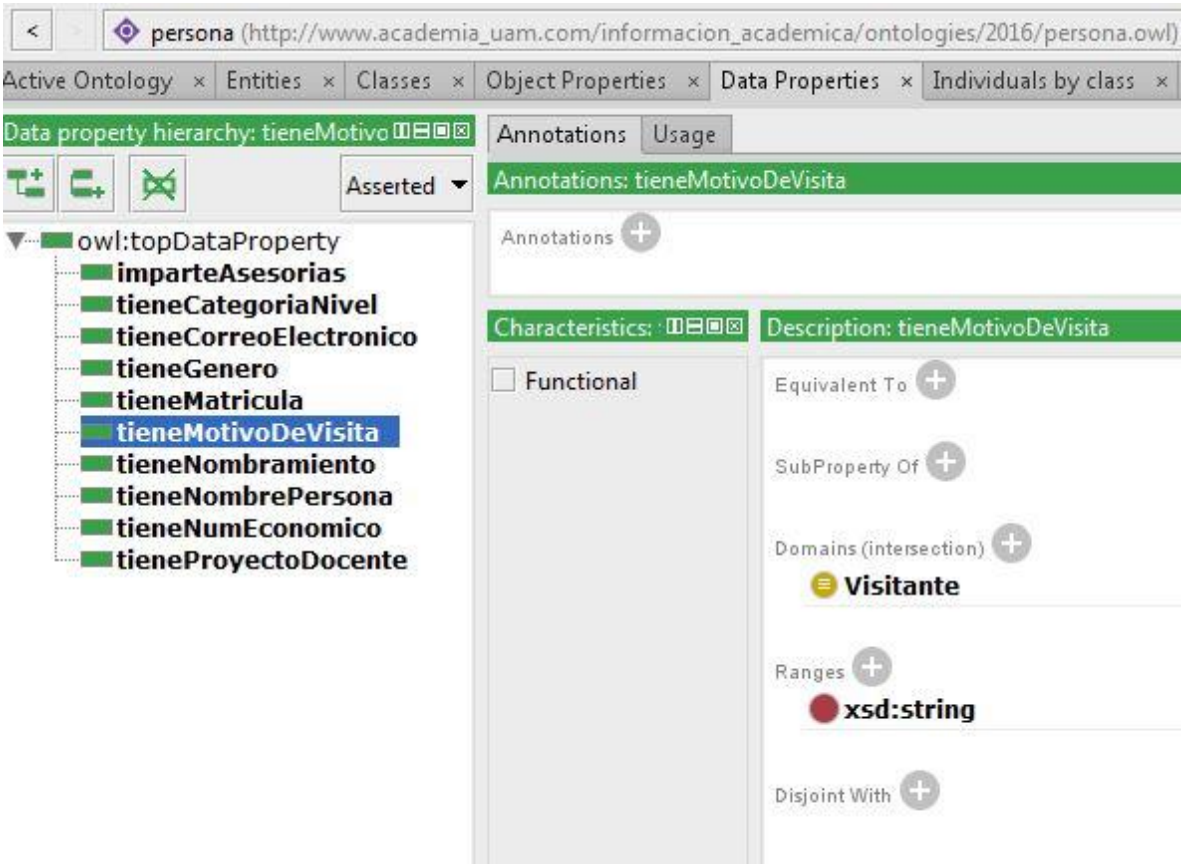


Ilustración 11: Data properties de ontología Persona

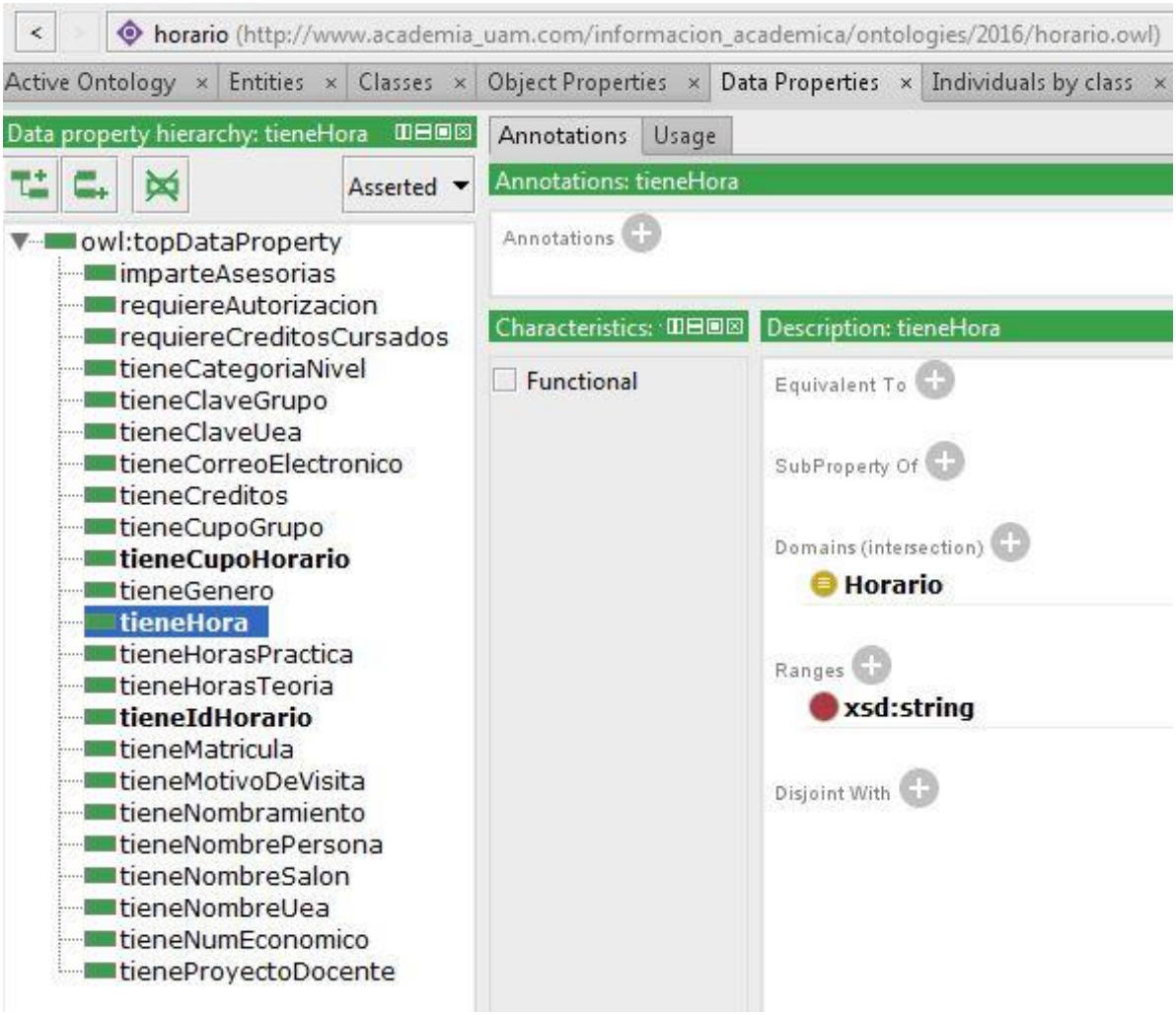


Ilustración 12: Data properties de la ontología Horario

En los archivos “Entregables” que se adjuntan con este reporte, se encuentran las ontologías completas en archivos “owl” tal como las puede leer el Protege.

Para concluir este apartado, falta comentar que la ontología Horario importó las demás ontologías para formar el sistema ontológico completo. En la siguiente imagen se muestran las importaciones realizadas en Horario:

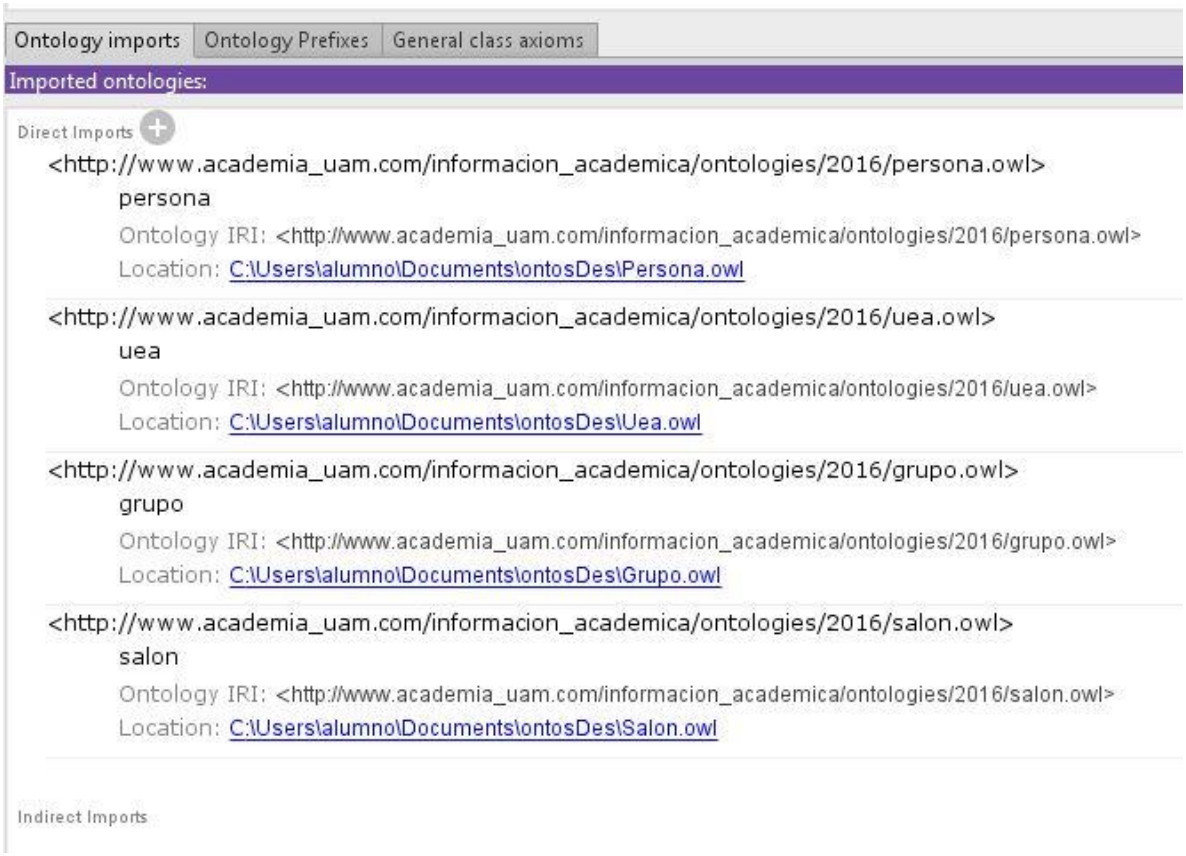


Ilustración 13: ontologías importadas en Horario

Al llevar a cabo una importación de una ontología en otra, todas las clases y propiedades de la ontología “hija” quedan incluidas en la ontología “madre”. Y, cualquier modificación en la ontología “hija” afectará forzosamente a la ontología “madre”.

b. Diseño e implementación del programa de carga de datos

Para el desarrollo del programa de carga de las ontologías, se utilizó el IDE Eclipse, versión JEE Neon 3 [14], con la perspectiva de trabajo en Java, programación orientada a objetos. Se procedió en dos fases, toda vez que por un lado había que desarrollar un programa de lectura de los archivos Excel y, por el otro, un programa de escritura para efectuar el poblado de las ontologías.

1ª fase: Desarrollo del programa de lectura de los archivos Excel

Los archivos Excel fueron considerados como las "bases de datos" con la información que se deseaba explotar. Estos archivos fueron proporcionados por la Coordinación de la carrera y contienen los horarios provisionales de las UEA del plan de estudios de la Ingeniería en Computación referente al trimestre 15º; incluyen los datos de los salones, grupos, días, horas y profesores que imparten las materias. En la captura de la información en estos archivos hubo algunos errores y repeticiones de información, esto es importante mencionarlo porque durante el desarrollo del programa, se tuvieron que tomar en cuenta estos incidentes: se crearon formas de verificación de los datos de cada celda de la hoja Excel y de las claves que sirvieron como identificadores de las instancias. Al evaluar los datos de cada celda se evitaron los fallos en la información y, con el examen de las claves, se impidió la repetición de información lo cual hubiera generado la duplicación de instancias en las ontologías. Asimismo, se procuró diseñar un programa que no requiriera modificar los archivos Excel o que la preparación de cada archivo para su lectura fuera mínima.

El programa de lectura fue conformado por los siguientes elementos: a) clases relativas a cada ontología, b) clases referentes a cada instancia que se debía insertar en las ontologías; y, c) los métodos de lectura de los archivos Excel y de creación de objetos.

a) Clases representando a cada ontología. Estas clases contienen los datos de los archivos Excel y las direcciones **url** de las ontologías y sus componentes. Se utilizó un archivo Excel por cada ontología, con excepción del archivo Excel "Ayudantes" que constituyó una prueba con datos ficticios. En la siguiente imagen se muestra lo anterior:

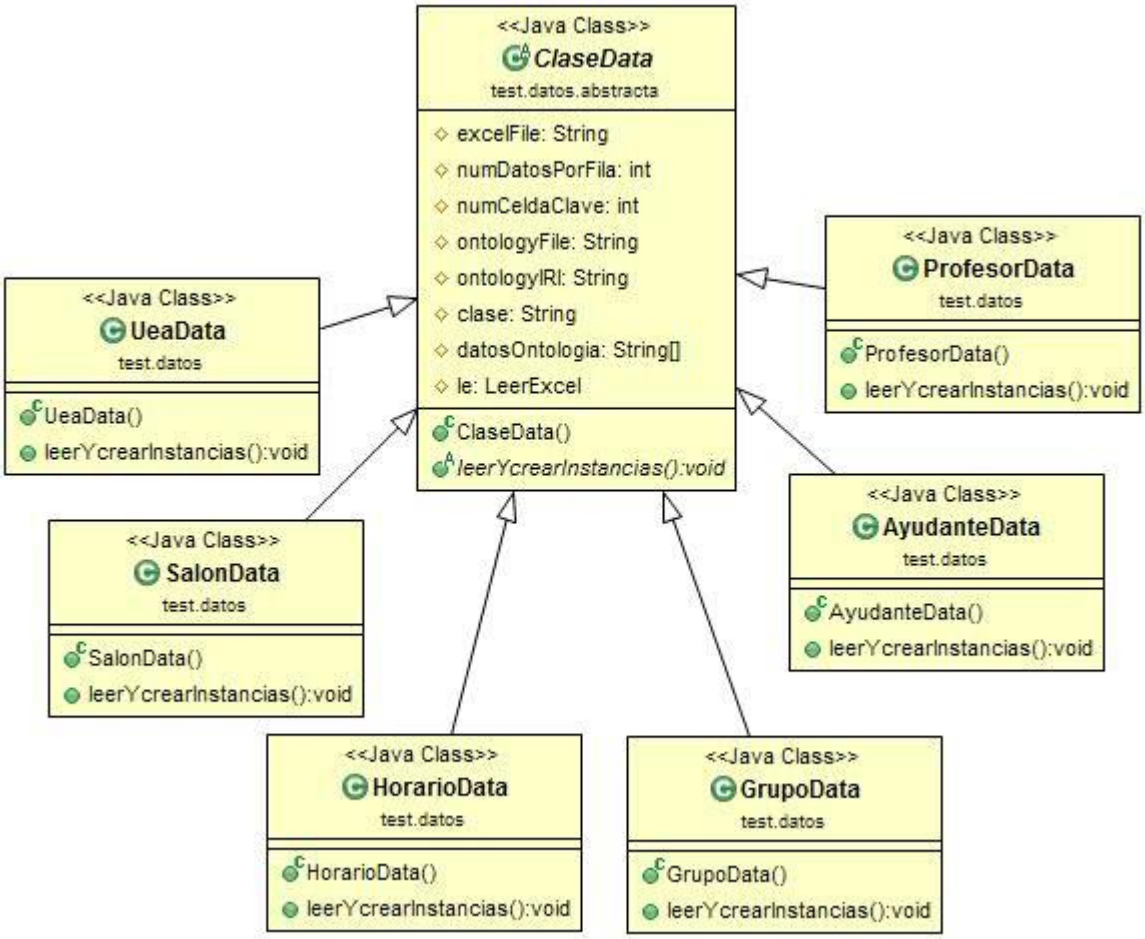


Ilustración 14: Clases con datos iniciales

b) Clases relativas a cada una de las instancias que se debían crear dentro de las ontologías. Como ejemplo de estas clases, respecto a la ontología Persona, se crearon todas las clases del árbol taxonómico de la clase *Persona*, es decir, se creó la clase *Persona* y, a partir de ésta, las clases hijas que heredan las características de persona: *Empleado*, *Alumno* y *Visitante*. Y, a partir de *Empleado*, las clases hijas *Académico* y *Administrativo*, que heredan los atributos de *Empleado*. Y, por último, de la clase madre *Académico*, las clases hijas *Ayudante* y *Profesor*. La finalidad de este paso fue crear primero las instancias como objetos Java y, posteriormente, desde cada objeto, crear las instancias y sus propiedades en las ontologías. Si bien este paso parece superfluo, permite desacoplar cada instancia y poder utilizar el código de cada una de ellas en otro programa, obedeciendo al principio de "encapsulamiento" de la programación orientada a objetos. Los diagramas de clases se muestran en seguida:

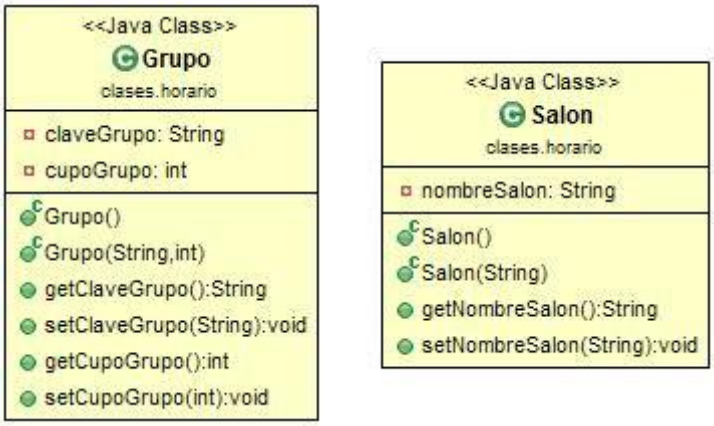


Ilustración 15: clases Grupo y Salon

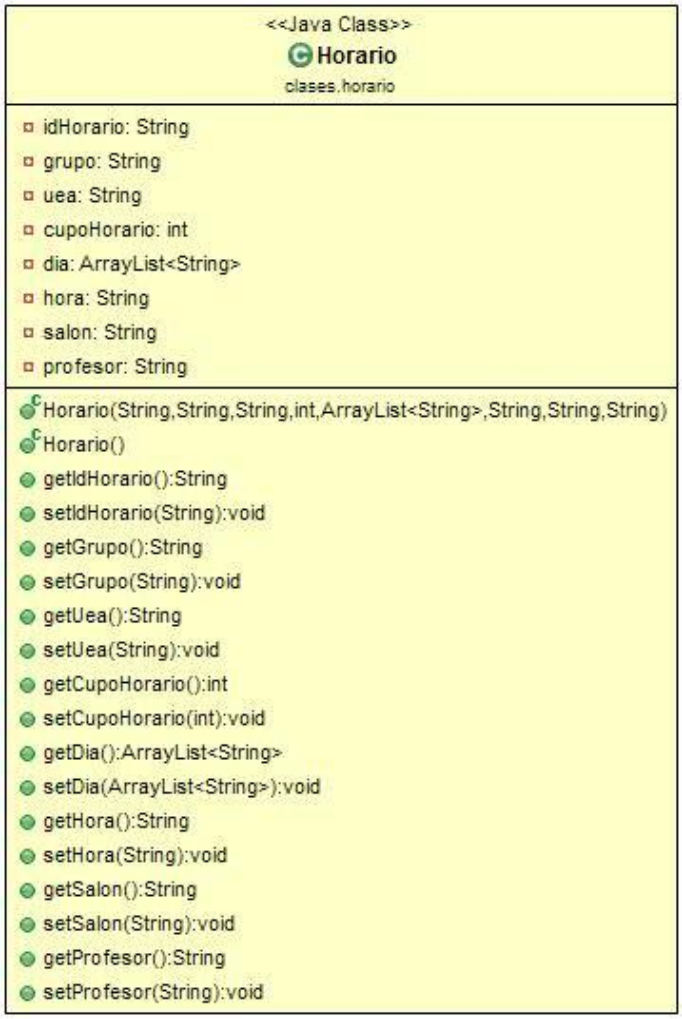


Ilustración 16: clase Horario

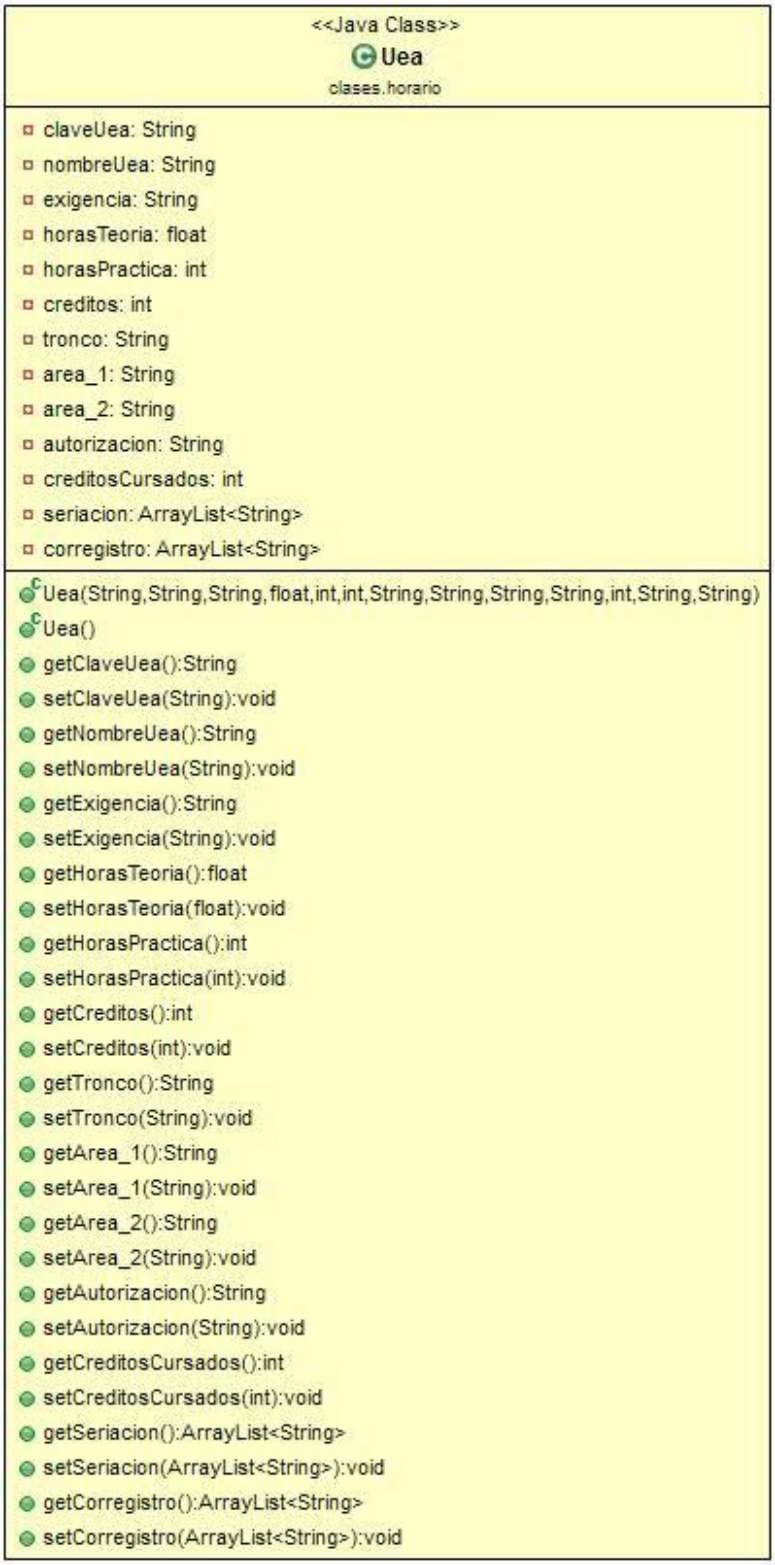


Ilustración 17: clase UEA

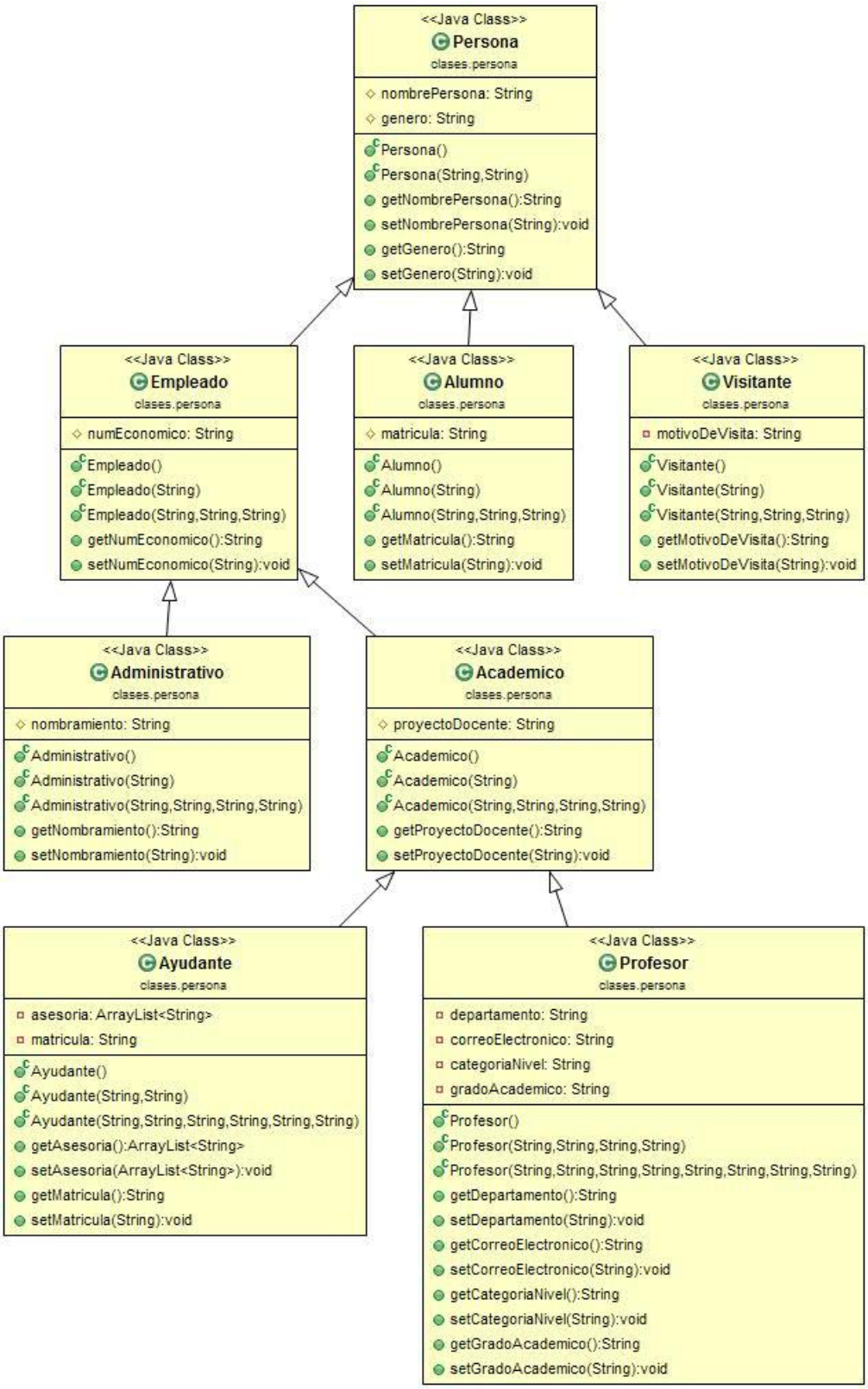


Ilustración 18: clase Persona

c) El método de lectura realizaba la lectura de una fila de la hoja Excel y llamaba al método que creaba el objeto respectivo con todos sus atributos, después volvía a realizar la misma operación hasta agotar todas las filas de la hoja. Para lo anterior, se utilizó un bucle.

2ª fase: Desarrollo del programa de escritura para efectuar el poblado de las ontologías

Para el poblado de las ontologías, se usó la API OWL [15]. Esta API permitió la conexión con las ontologías, así como la creación de las instancias y la inserción de sus Data properties y Object properties.

Aquí es importante destacar que el desarrollo del programa para poblar las ontologías "hijas" fue diferente del programa para poblar la ontología Horario. Esta ontología, al efectuar la importación de las otras ontologías, requirió de un código diferente, específico, ya que las instancias de las Object properties poseían direcciones url ya definidas en sus propias ontologías. La clase *OntologyOperationsWithImported*, da un ejemplo de esta situación. En ella aparecen las direcciones IRI (nombre adoptado por las ontologías) de las clases que hubo que manejar para realizar las inserciones, a diferencia de las clases que efectuaban el poblado de las ontologías sencillas. En las imágenes mostradas a continuación se puede apreciar la diferencia:

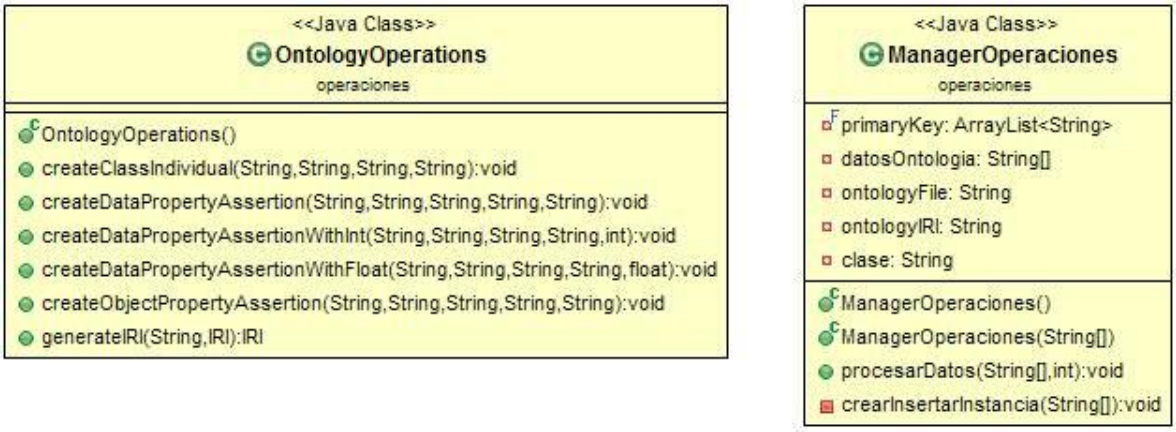


Ilustración 19: clases para poblar ontologías

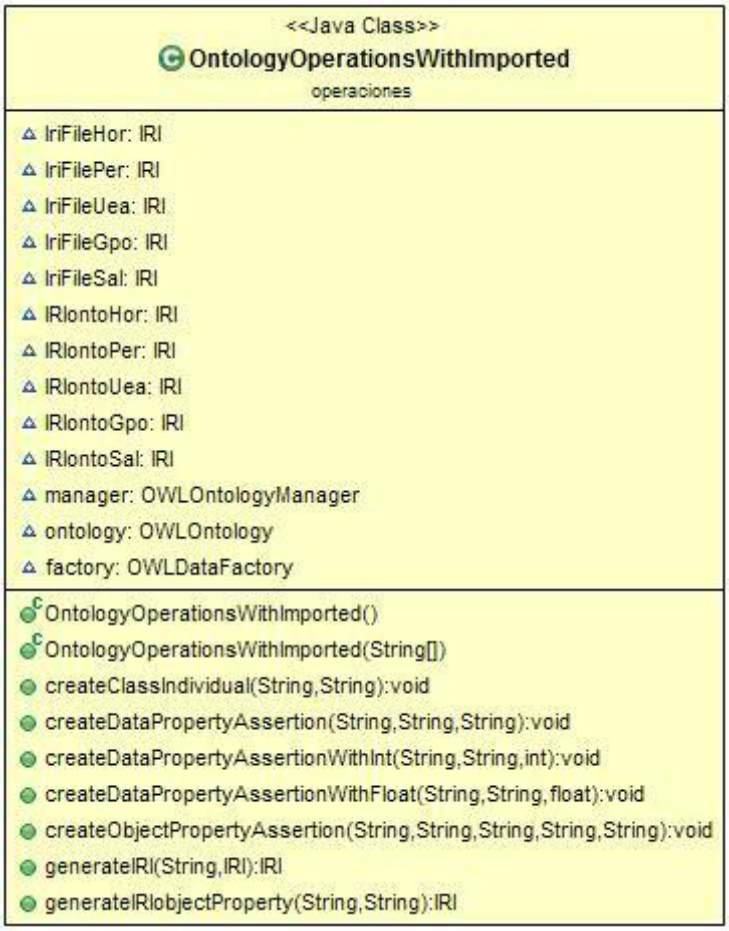


Ilustración 20: clase para poblar Horario

En esta parte del programa se tuvo que tomar en cuenta la axiomatización de las clases, así como de sus propiedades, a fin de no generar problemas con los tipos de datos insertados. Un ejemplo de esto fue la diferencia entre crear una Data property con un tipo de dato *Integer* como rango, a una que tenía un *String* o un *Float*.

Para cerrar este apartado, es conveniente agregar que el programa se dividió en tres conjuntos de paquetes:

- Conjunto de paquetes con las clases que tienen los datos de los archivos Excel y de las ontologías
- Conjunto de paquetes con las clases de las instancias
- Conjunto de paquetes con la lógica del programa: los métodos que realizan las lecturas y el poblado de las ontologías

c. Evaluación del sistema ontológico

La evaluación del sistema ontológico se llevó a cabo en tres etapas.

1ª etapa	Pruebas con el razonador Pellet antes y después de crear algunas instancias de prueba
En esta etapa, se probaron las ontologías cuando se concluyeron sin haber insertado ninguna instancia. El razonador pudo haber lanzado algún aviso de fallo al encontrar errores en la taxonomía de las clases, en las definiciones completas o en las axiomatizaciones de las Data properties. **En seguida, se crearon de forma manual algunas instancias para probar la axiomatización. Se crearon instancias que no cumplían con los datos especificados o que tenían propiedades equivocadas a propósito, con el objeto de observar la labor del razonador que, en estos casos, dio error.**	

2ª etapa	Pruebas con el razonador Pellet después de realizar el poblado completo de las ontologías
Este paso es muy parecido al anterior, sólo que se efectuó cuando se poblaron por completo las ontologías. Era de esperar que no hubieran errores, debido a que ya se había realizado este examen con algunas instancias, a menos de que existieran fallos en los datos de los archivos Excel, lo cual no sucedió. **Cabe aclarar que, durante el poblado de las ontologías, sí hubo que repetir las operaciones y modificar innumerables veces el programa de carga por la cantidad de errores que se presentaron.**	

3ª etapa	Pruebas con algunas de las preguntas de competencia
En esta etapa decisiva, se realizaron varias pruebas utilizando las preguntas de competencia presentadas al inicio de este reporte. Para efectuar este examen, las preguntas se diseñaron bajo el lenguaje de consulta SPARQL, el cual admite la herramienta Protege. Aunque SPARQL no es un lenguaje propio para la explotación de la información contenida en una ontología (como sí lo es SWRL), por no estar orientado a la lógica descriptiva, sí sirvió para el fin que se buscaba.	

La siguiente pregunta es un ejemplo de la 3ª etapa:

¿Qué UEA imparte la Dra. Maricela Bravo, en qué horas y qué días?

El código de la pregunta en lenguaje SPARQL quedó de la siguiente forma:

```
SELECT ?nomProf ?nomUea ?dia ?hora WHERE {
  ?idProf a per:Profesor. ?idHor a hor:Horario. ?idUea a uea:Uea.
  ?idHor hor:tieneProfesor ?idProf.
  ?idHor hor:tieneUea ?idUea.
  ?idProf per:tieneNombrePersona ?nomProf.
     FILTER regex(?nomProf, "MARICELA").
  ?idUea uea:tieneNombreUea ?nomUea.
  ?idHor hor:tieneDia ?dia.
  ?idHor hor:tieneHora ?hora
} ORDER BY ?nomProf
```

En la siguiente imagen se muestra la respuesta dada por el razonador Pellet del Protege:

```
PREFIX hor: <http://www.academia_uam.com/informacion_academica/ontologies/2016/horario.owl#>
PREFIX per: <http://www.academia_uam.com/informacion_academica/ontologies/2016/persona.owl#>
PREFIX uea: <http://www.academia_uam.com/informacion_academica/ontologies/2016/uea.owl#>
SELECT ?nomProf ?nomUea ?dia ?hora WHERE {
  ?idProf a per:Profesor. ?idHor a hor:Horario. ?idUea a uea:Uea.
  ?idHor hor:tieneProfesor ?idProf.
  ?idHor hor:tieneUea ?idUea.
  ?idProf per:tieneNombrePersona ?nomProf.
     FILTER regex(?nomProf, "MARICELA").
  ?idUea uea:tieneNombreUea ?nomUea.
  ?idHor hor:tieneDia ?dia.
  ?idHor hor:tieneHora ?hora
} ORDER BY ?nomProf
```

nomProf	nomUea	dia	
"BRAVO CONTRERAS MARICELA CLAUDIA"	"ARQUITECTURA E INTEGRACION DE APLICACIONES EMPRESARIALES"	JUEVES	"14:30 - 16:45"
"BRAVO CONTRERAS MARICELA CLAUDIA"	"ARQUITECTURA E INTEGRACION DE APLICACIONES EMPRESARIALES"	MARTES	"14:30 - 16:45"
"BRAVO CONTRERAS MARICELA CLAUDIA"	"PROGRAMACION ORIENTADA A SERVICIOS"@	MARTES	"16:45 - 19:00"
"BRAVO CONTRERAS MARICELA CLAUDIA"	"PROGRAMACION ORIENTADA A SERVICIOS"@	JUEVES	"16:45 - 19:00"

Ilustración 21: Sparql

Como se puede apreciar, la Dra. Bravo imparte dos UEA: Arquitectura e integración de aplicaciones empresariales los martes y jueves de las 14:30 a las 16:45 horas, y Programación orientada a servicios, también, los martes y los jueves pero de las 16:45 a las 19:00 horas.

5. Resultados

En el presente proyecto se desarrolló un sistema ontológico y un programa para poblar dicho sistema. Para el desarrollo del sistema se utilizó una metodología ampliamente probada. Los resultados satisfactorios muestran que la metodología de diseño y construcción de ontologías cumple a cabalidad con su cometido. En otras palabras, si queremos diseñar un conjunto de ontologías, la metodología seleccionada siempre nos dará resultados positivos. En este punto, es importante resaltar que los principios de diseño ontológico de la metodología (claridad, coherencia y modularidad), son esenciales para el correcto desarrollo de las ontologías.

Lo anterior no quiere decir que no se hayan tenido cuantiosos tropiezos en el desarrollo de las ontologías. Todo diseño e implementación de sistemas informáticos es recursivo, ya que aparecen correcciones una y otra vez en todos los pasos del desarrollo. Sin embargo, los resultados esperados demuestran la viabilidad del método usado.

Por otro lado, aunque el desarrollo del programa provocó, en algunos momentos, fuertes dolores de cabeza, también se obtuvieron buenos frutos. La API OWL facilitó en gran medida el trabajo con ontologías. Y, aunque en esta parte del trabajo no se adoptó ningún patrón de diseño de software, en todo momento se intentó cumplir con los postulados de la programación orientada a objetos, lo que también allanó el desarrollo del programa.

6. Conclusiones

Los objetivos planteados desde el comienzo del proyecto se lograron completamente, con lo que podemos concluir que los métodos y las herramientas utilizados en este proyecto son efectivos.

Por otro lado, el presente trabajo pone en evidencia el vasto potencial que encierra el uso de ontologías en el campo de los sistemas de información inteligentes. Aunque el proyecto fue relativamente modesto, se puede vislumbrar que las ontologías informáticas posibilitan el aprovechamiento de la información de forma más efectiva que muchas de las tecnologías tradicionales.

En este caso, se desarrolló un programa de carga para alimentar a las ontologías, pero no siempre debe ser así. Las ontologías pueden trabajar de la mano con bases de datos, sobre todo en los casos en que la información no es de cientos sino de miles o millones de registros.

7. Referencias bibliográficas

[1] UAM, «Módulo de Información Escolar de Alumnos de Licenciatura,» 2015. [En línea]. Available: https://ayamictlan.uam.mx:8443/sae/azc/aewbf001.omuestraframes?mod=1.

[2] A. M. Velázquez, Aplicación web de administración de horarios para estudiantes, Proyecto terminal, México: CBI, UAM Azcapotzalco, 2009.

[3] J. A. Romero, Sistema de comparación de mapas curriculares basado en ontologías, Proyecto terminal, México: CBI, UAM Azcapotzalco, 2014.

[4] F. Cortes, Sistema para la comparación de documentos basado en ontologías, Proyecto terminal, México: CBI, UAM Azcapotzalco, 2015.

[5] J. J. Samper, Ontologías para servicios web semánticos de información de tráfico: descripción y herramientas de explotación, Tesis doctoral, España: Departamento de informática, Universitat de Valencia, 2005.

[6] D. M. Aguilar, Búsqueda Web basada en ontologías de dominio, Tesis de maestría, México: Laboratorio de Tecnologías de la Información, CINVESTAV, Tamaulipas, 2008.

[7] A. E. Sandoval, «Uso de ontologías y web semántica para apoyar la gestión del conocimiento,» *Ciencia e Ingeniería Neogranadina,* vol. XVII, nº 2, pp. 111-129, 2007.

[8] T. R. Gruber, A Translation Approach to Portable Ontology Specifications, Technical Report KSL 92-71, USA: Knowledge Systems Laboratory, Computer Science Department, Stanford University, 1992.

[9] D. P. Sauras, Diseño de una ontología para aplicaciones en el dominio de la movilidad sostenible: el coche compartido, Tesis de maestría, España: ESCUELA TÉCNICA SUPERIOR, UNIVERSIDAD DE VALLADOLID, 2012.

[10] M. Horridge, A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools, USA: Universidad de Manchester, 2011.

[11] «OWL Web Ontology Language,» World Wide Web Consortium (W3C), 2004. [En línea]. Available: https://www.w3.org/TR/owl-features/.

[12] «Protege,» Stanford University, 2016. [En línea]. Available: http://protege.stanford.edu/.

[13] M. Bravo, «Addressing Clarity and Coherence during Ontology Construction,» *Tecnologías emergentes y avances de la computación en México,* pp. 177-182, 2016.

[14] «Eclipse,» 2017. [En línea]. Available: https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/3/eclipse-jee-neon-3-win32.zip.

[15] «The OWL API,» Sourceforge, [En línea]. Available: http://owlapi.sourceforge.net/. [Último acceso: 2017].

8. Entregables

Los entregables del presente proyecto de integración son los siguientes:

- Sistema ontológico con las cuatro ontologías “hijas” (Grupo, Persona, Salón y UEA) y la ontología “madre” (Horario) que incluye a las hijas, en archivos con extensión “owl” que pueden ser importados por la herramienta Protege. Todo lo anterior en un archivo comprimido llamado “Ontologías” con extensión zip.
- Programa de carga de datos a las ontologías, desarrollado en lenguaje Java, en archivo comprimido llamado “Codigo” con extensión zip. Este archivo puede ser importado por el IDE Eclipse para edición, lectura o ejecución.
- Archivos Excel con los datos de carga que representan las bases de datos para alimentar las ontologías. Estos archivos también están en un archivo comprimido llamado “Datos” con extensión zip.
- El presente reporte.