Instituto de Computación, Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

PROYECTO DE GRADO INGENIERÍA EN COMPUTACIÓN

Generación automática de formularios para el ingreso de datos en ontologías - Aplicación en la implementación de una ontología de percepciones de piezas de arte

Néstor Rocchetti y Gonzalo Labandera
Mayo de 2016

Tutora:
Regina Motz

Generación automática de formularios para el ingreso de datos en ontologías - Aplicación en la implementación de una ontología de percepciones de piezas de arte

Rocchetti, Néstor y Labandera, Gonzalo

Proyecto de Grado - Ingeniería en computación.
Instituto de Computación - Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay, mayo de 2016

Generación automática de formularios para el ingreso de datos en ontologías - Aplicación en la implementación de una ontología de percepciones de piezas de arte

Resumen

Las ontologías son utilizadas para conceptualizar aspectos de la realidad. La realidad se describe mediante conceptos y relaciones. Se utilizan junto con herramientas de razonamiento, las cuales sirven para inferir relaciones a partir de las ya definidas en la ontología. Un objetivo del proyecto de grado es la realización de una ontología de percepciones y medios de producción de piezas de arte (OPPA). Otro de los objetivos es el diseño e implementación de un generador automático de formularios y la construcción de una aplicación en la que se pruebe el funcionamiento del generador construido. La ontología OPPA se desarrolló en conjunto con el experto en el campo del arte Vladimir Muhvich, quien proveyó de documentación y también colaboró en el proceso de desarrollo validando los componentes. La documentación provista es un estudio de su autoría en el dominio de las piezas de arte con un enfoque en los sentidos y los medios de producción, el material es una taxonomía de su dominio de estudio. La ontología cuenta con cuatro componentes importantes asociados a las piezas de arte en exposiciones: percepciones, medios de producción, soportes de medios de producción y familias. Las piezas de arte en exposiciones se clasifican en familias según las relaciones definidas sobre los demás componentes importantes. Muhvich utiliza la taxonomía en el marco del proyecto *Engrama, investigación sobre modelos de visualización morfológica y evolutiva de Colecciones de Arte* para definir una terminología a seguir en su recolección de datos sobre piezas de arte en exposiciones. Con la finalidad de proveer una interfaz sencilla para el ingreso de datos en ontologías se desarrolló un generador de formularios automático. Se utilizó como caso de estudio la ontología de percepciones y medios de producción de piezas de arte. Se definieron dos tipos de formularios, un formulario solo de preguntas múltiple opción y otro para permitir ingreso de

datos. Los formularios se generan en tiempo de ejecución, de esta forma los cambios que se realicen sobre la ontología impactarán automáticamente en los formularios. Se permite el cambio a ontologías en la que se represente otra realidad, permitiendo que se reutilice el generador de formularios en otros dominios. Se implementó un prototipo de aplicación en la que se utiliza el generador de formularios automático y además se guardan los datos ingresados en dicho formulario en la ontología de origen. Esta aplicación será utilizada por el experto y por lo tanto tuvo el rol de cliente en el desarrollo de esta parte del proyecto. Se implementaron también funcionalidades específicas, entre ellas una funcionalidad para generar grafos a partir de un conjunto de respuestas por medio de una API en la que se implementan funciones de la plataforma Gephi. Los grafos generados son utilizados por el experto en el marco del proyecto Engrama. Como resultado, con respecto al desarrollo de la ontología se obtuvo una primera versión de la ontología de percepciones y medios de producción aplicados a piezas de arte en exposiciones. Con respecto al desarrollo del generador de formularios se logró insertarlo en un prototipo en el que se utiliza para ingresar datos en la ontología desarrollada y se probó su utilidad en ontologías de otro dominio. El prototipo fue validado por el cliente, quien lo integrará a su trabajo como herramienta para recolección de datos.

Palabras clave: ontologías, arte, OWL, formulario automático

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En la investigación en piezas de arte se utilizan un conjunto de conceptos y jerarquías que no se encuentran implementados en informática. La necesidad surge del proyecto *Engrama, investigación sobre modelos de visualización morfológica y evolutiva de Colecciones de Arte* implementado por el experto en arte Vladimir Muhvich y presentado en [25]. Engrama es: *La huella neurofisiológica en el cerebro que es la base de un recuerdo de la memoria* [1]. El proyecto Engrama se basa en la materialización por medio de grafos de engramas culturales. En el trabajo de Muhvich se presenta el desarrollo y los resultados del proyecto Engrama aplicado a la visualización de piezas de arte agrupadas en colecciones. En la Figura 1.1 se muestra el engrama perteneciente al Salón Nacional del año 1939, se representa como un grafo en el que los nodos representan piezas, experiencia sensorial, medios de producción, soportes, familias de producción y familias de preservación, a su vez las aristas representan las relaciones entre los nodos de piezas y los restantes tipos de nodos posibles. Como paso previo a la realización del proyecto Engrama, Muvich definió una taxonomía de conceptos y elementos involucrados en la percepción de una obra de arte, agrupándolos en: percepciones, soportes, medios de producción, familias de producción y familias de preservación. El proyecto Engrama muestra de forma visual las diferencias morfológicas existentes en los engramas realizados por cada Salón Oficial de Arte en Uruguay realizados entre 1939 y 2012. Muhvich plantea una discusión abierta sobre las causas de estas diferencias observando en algunos casos coincidencias de los cambios entre engramas de diferentes años y eventos históricos ocurridos en esos años, por ejemplo, el arte contemporáneo que se había comenzado a exponer desde 1968 deja de premiarse luego del comienzo de la dictadura en el Uruguay.

Para realizar la investigación presentada en el proyecto Engrama, Muhvich realiza un relevamiento de datos de 309 obras premiadas registrando el proceso de catalogación a través de notas en cuadernos. Para poder realizar el grafo, traduce los datos

[1] Enciclopedia de la salud -http://www.enciclopediasalud.com/definiciones/engrama

catalogados a nodos y relaciones, y finalmente los ingresa a una hoja de cálculo que sirve como punto de entrada de datos para la plataforma Gephi[2], en la que trabaja para dar forma a los grafos creados. Esta forma de trabajo manual para preparar los datos que procesa Gephi consume muchas horas hombre de trabajo rutinario y carece de posibilidades viables de extensión a corto plazo para el análisis de un número más amplio de exposiciones o de abarcar un período mayor de tiempo. Como solución a estas limitaciones, es objetivo de este proyecto proveer una herramienta informática capaz de brindar una representación de las percepciones de obras de arte con un nivel de abstracción adecuado para los expertos en arte y que permita su clasificación de forma automática.

El modelo de representación de conocimiento que cumple con estos dos objetivos es una ontología de percepciones de obras de arte. Las ontologías son utilizadas desde hace tiempo por distintas áreas, la primer ontología conocida es la de Aristóteles que clasifica a los seres vivos. Las ontologías se estudian en Filosofía y en Lingüística. Según Gruber [13] una Ontología en Computación es la conceptualización formal de una realidad compartida. La Ontología de Percepciones de Piezas de Arte (OPPA) que desarrolla este proyecto permite al experto en arte disponer de un medio estructurado para ingresar los datos de las piezas e inmediatamente de forma automática clasificarlas. Se utilizó un modelo de ontologías en las que se utiliza OWL 2 [3] (Web Ontology Language versión 2) como lenguaje de especificación. OWL 2 está basado en Lógica Descriptiva (DL en inglés) [27], que se encuentra compuesta por una TBox (i.e., donde se encuentran las definiciones de los conceptos), una RBox (i.e., donde se encuentran definidas las dependencias sobre los roles) y una ABox (i.e., es donde se definen aserciones sobre individuos). Las ontologías poseen dos mecanismos de razonamiento: satisfactibilidad del modelo y clasificación de instancias por medio de inferencias sobre datos existentes. El entorno de desarrollo para la implementación de la ontología es el editor de ontologías Protégé [4].
En la ontología planteada se conceptualizó la realidad de las piezas y su relación con percepciones, medios de producción, soportes y familias. Para la implementación de la ontología informática OPPA, se utilizó como insumo la taxonomía y la ontología (no informática) desarrollada por Muhvich. Se realizaron reuniones con el experto durante todo el proceso con la finalidad de validar la comprensión de la realidad y los resultados parciales en la construcción de la ontología. Cómo resultado de la implementación de la ontología se obtuvo: (i) una primera versión de la realidad planteada por el experto; (ii) un mecanismo de inferencia que permite la clasificación automática de las piezas ingresadas por exposición en las familias (i.e., producción bidimensional, producción tridimensional, nuevos soportes y medios inestables).

[2]Gephi - https://gephi.org/
[3]OWL 2 vista general del lenguaje - https://www.w3.org/TR/owl-overview/
[4]Sitio en línea de Protégé - http://protege.stanford.edu/products.php

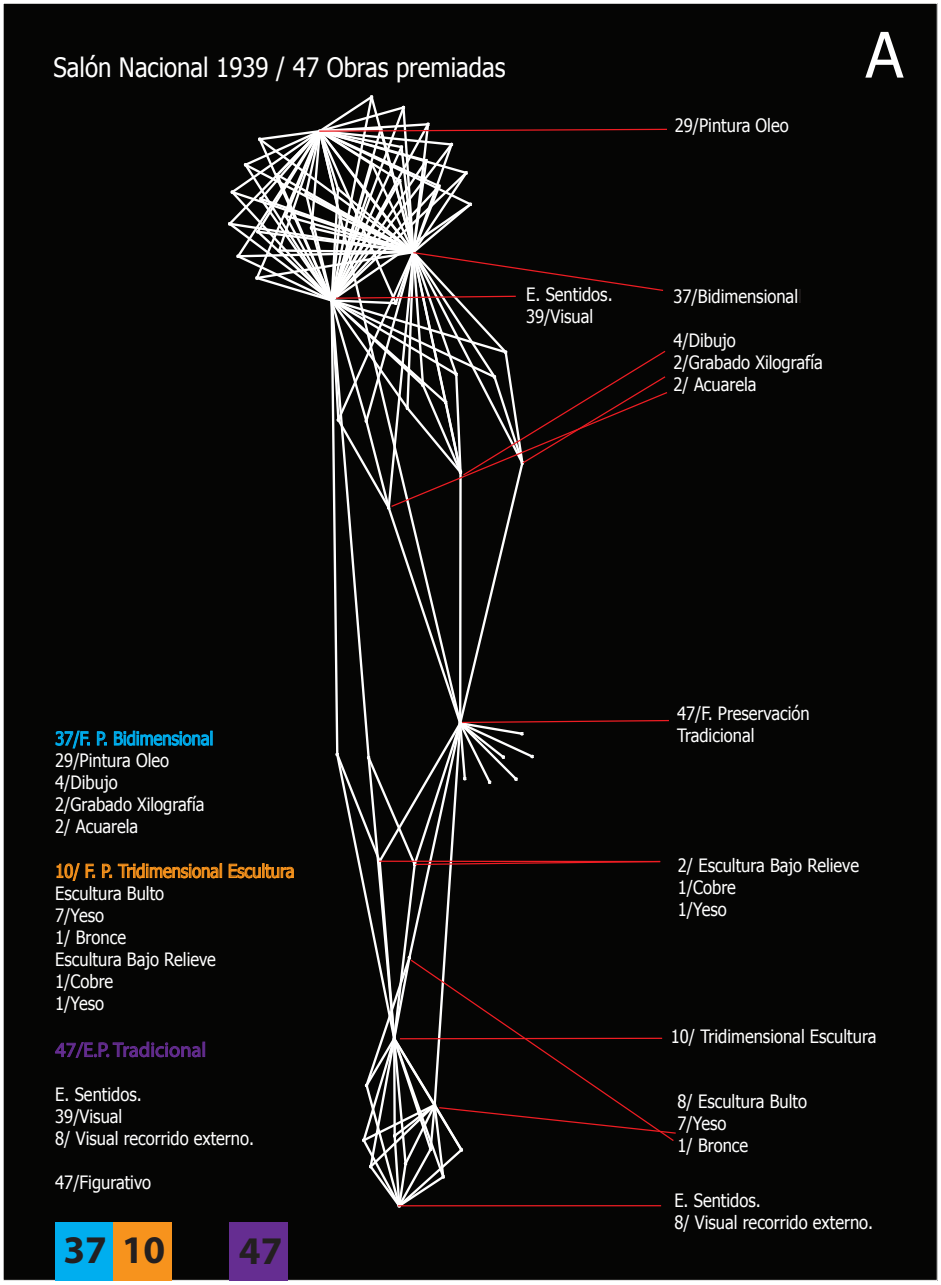

Figura 1.1: Engrama correspondiente al Salón Nacional de 1939. La exposición contó con 47 obras premiadas. En el grafo se representan los medios de producción, lo soportes, las características con enfoque en los sentidos, la familia de producción y la familia de preservación para todas las piezas premiadas de la exposición.

Adicionalmente, para evitar la necesidad de que el usuario (en este caso el experto en arte) deba interactuar directamente con la ontología, se desarrolla un formulario de ingreso de datos guiado por la ontología. Es relevante destacar que para desarrollar este formulario en este proyecto se diseña e implementa un **Generador de Formularios guiado por Ontologías** que permite disponer de forma semiautomática de un formulario web para el ingreso de datos para cualquier ontología. La aplicación es responsiva, esto significa que se adapta a distintos tamaños y formas de pantallas, permitiendo que pueda ser utilizada en diferentes dispositivos (e.g., computadora portátil, tablet, smart phone).
Se crearon perfiles de usuario que se componen de varios usuarios que tienen los mismos privilegios sobre la aplicación. En el caso particular de los formularios, todos los usuarios que pertenezcan al mismo perfil se encuentran habilitados a realizar el mismo tipo de formulario, por lo tanto la instancia de configuración se realiza una vez por perfil.
La construcción de los formularios implica definir cuales son los componentes que se deben agregar al formulario y la ubicación de estos, lo cual se realiza en parte en base a los datos ingresados en la etapa de configuración.
El generador de formularios implementado tiene la finalidad de proveer independencia de la aplicación sobre la ontología. Si se realizan cambios profundos en la estructura de la ontología es posible que se deba realizar la instancia de configuración nuevamente para todos los perfiles. Esto implica que es posible cambiar la ontología de percepción de piezas de arte por otra en la que se represente un dominio diferente al del campo del arte, y se genere también su formulario para ingreso de datos realizando nuevamente sólo la etapa de configuración.
Al proveer independencia de la aplicación sobre la ontología, se provee una capa de abstracción de la ontología al completar el formulario. De esta forma el usuario debe conocer el dominio (es decir que debe comprender el lenguaje utilizado en el contexto de las producciones del campo del arte en el caso de generar formularios a partir de la ontología OPPA) pero no es necesario que conozca el lenguaje en el que estas se implementan (i.e., OWL 2 en el caso del presente proyecto).

El generador se implementó en el marco de una aplicación web realizada a modo de prototipo. En el prototipo se proveen las funcionalidades de: configuración, generación de formularios y guardado de sus datos, y de exportación de los datos ingresados a un grafo en el formato adecuado para utilizar en la plataforma Gephi. Se proveen además las siguientes funcionalidades: historial de encuestas realizadas, creación de nuevos usuarios y edición de datos de usuario.

Con la finalidad de mejorar la experiencia del usuario al completar los formularios, en ciertos casos, la información se solicita por medio de preguntas expresadas en lenguaje natural tomando información del contexto del dato que se solicita de la ontología.

En resumen, los objetivos del proyecto son los siguientes: (i) el diseño e implementación de una ontología de piezas de arte en exposiciones y su relación con la experiencia sensorial, soportes, medios de producción, familias de producción y familias de preservación; (ii) diseño e implementación de un generador de formularios semiautomático para ingreso de datos basado en ontologías y que sea independiente del dominio que estas representan (el generador también incluye la funcionalidad de guardado de datos ingresados en la ontología utilizada); y (iii) la implementación de un prototipo de aplicación para múltiples dispositivos en el que se integre el generador y se provea la funcionalidad de generación de grafos para la plataforma Gephi.

El resto de este documento se organiza de la siguiente forma. En el Capítulo 2 se presenta un resumen de trabajos relacionados a las temáticas estudiadas. En el Capítulo 3 se describe la ontología diseñada e implementada, así como también el proceso de su construcción. En el Capítulo 4 se presenta la generación semiautomática de formularios para ingreso de datos. En el Capítulo 5 se describe la aplicación construida a modo de prototipo. En el Capítulo 6 se presenta la conclusión final y los lineamientos para el trabajo futuro. Luego, en el Anexo A se presenta un instructivo para la instalación de la aplicación. En el Anexo B se presenta el manual de usuario para realizar la configuración. Por último, en el Anexo C se presentan impresiones de pantalla de la aplicación.

Capítulo 2

Trabajos relacionados

En este capítulo se presentan los trabajos relacionados a las temáticas de ontologías en el arte, generación de estructuras y guardado de datos a través de ontologías, y generación de preguntas a partir de las clases de las ontologías y sus relaciones.

2.1. Ontologías en el Arte

En lo referente a Ontologías en el arte, se han realizado trabajos de investigación en el área de filosofía, tal es el caso del trabajo presentado por Davies et al [8], en donde se presenta una discusión sobre formas de clasificación de piezas de arte. Se explican dos formas de clasificar piezas de arte: según multiplicidad y universalidad, y se discute acerca de cuales tipos de piezas pertenecen a cada clasificación. En el caso de la multilpicidad, las piezas se clasifican como únicas, si su medio de producción es el mismo que su soporte (e.g. una pintura o una fotografía instantánea), o múltiples, si su soporte es diferente de su medio de producción (e.g. fotografías producidas a partir de negativos). También se realiza una discusión acerca de si las piezas de arte se pueden reconocer de forma universal o solo por un grupo de personas.
El punto de vista tratado en este trabajo difiere con el tratado en el proyecto planteado debido a que no se trata el tema de las percepciones asociadas a las piezas de arte, tampoco se describen los medios de producción de las piezas, y tampoco se nombran las familias de piezas planteadas por el experto junto al que se desarrolló el proyecto.

En el trabajo de Wolterstoff [34], se plantea un acercamiento a una ontología de tipos piezas de arte desde un punto de vista filosófico. Los tipos de piezas de arte están compuestos por *ejemplos* (i.e. instancias). Se plantean dos conjuntos de tipos de obras: objetos e interpretaciones (i.e. performance) y las piezas que pertenecen a estos tipos (e.g. escultura pertenece a objetos y canción pertenecen a performance). Luego la discusión se centra en definir formalmente las condiciones que cumplen las

piezas que pertenecen a cada tipo de pieza de arte, se utiliza la música como ejemplo para discutir las condiciones a cumplir por un tipo de obra para que pertenezca a un conjunto o a otro.

En el área de ontologías en informática y con respecto a la temática del arte, el libro publicado por Baka et al [4] explica la propuesta de una definición formal y unificada de *Categorías para la Descripción de Obras de Arte (CDWA en inglés)*. Por medio de esta herramienta se define la información necesaria para identificar y describir piezas de arte.
CDWA es una guía que se utiliza para organizar la descripción del contenido de piezas de arte, son metadatos que describen la información de la obra. Metadatos de piezas utilizados son *título*, *autor*, *clasificación*, entre otros. Existen también otros metadatos que relacionan piezas, por ejemplo si una pieza de arte se conforma de varias piezas se puede decir que estas piezas son *parte de*.
El objetivo del estándar es que se realice la correspondencia con otros estándares de metadatos, o con bases de datos, con tal fin se desarrolló un entorno de desarrollo. Actualmente se ha implementado la correspondencia con 14 estándares para catalogar y describir piezas de arte. El propósito de la creación de CDWA es dar los primeros pasos para hacer posible que la comunidad de investigadores pueda acceder a información en un único formato sin importar el formato de la fuente de información original.

2.2. Trabajos relacionados al ingreso de datos a ontologías

Con respecto a la generación de estructuras y guardado de datos en ontologías, se han creado herramientas por medio de las que se genera código tomando como punto de partida una ontología que represente un dominio. El objetivo en la utilización de estas herramientas es la generación de una estructura que se corresponda con la estructura de una ontología utilizada como insumo. Tal es el caso del trabajo presentado por Stevenson [29], en donde se presenta *Sapphire*, una herramienta de generación de software en forma dinámica, mediante la que es posible realizar una traducción de una ontología a definiciones de clases en java. Mediante este entorno es posible definir una realidad específica en una ontología informática con un lenguaje como OWL, y luego generar código en Java en base a la conceptualización realizada. Las clases que se generan implementan la herencia de clases, se crea una clase que corresponde al concepto *Thing*, todas las demás clases heredan de esta clase. También se encuentran modelados los atributos de instancia (i.e. propiedades de datos de los conceptos), las relaciones entre conceptos. Se provee soporte a lógica basada en mundo abierto mediante la definición de un tipo llamado *LogicValue* que adopta

los valores: verdadero, falso y desconocido. Se genera también código para realizar consultas a la ontología de una forma que requiere menos código que utilizando una API para manejo de ontologías directamente.
En [29] se realiza una comparación con otras herramientas mediante la cual se genera software en forma dinámica [9, 20, 31] con la finalidad de establecer a Sapphire como la herramienta más completa de generación de software basado en ontologías. En particular, los otros generadores carecen de la construcción de consultas a la ontología, para eso se apoyan directamente en la API de comunicación con la ontología (e.g., OWL API [2]).

Continuando con las herramientas, se encuentra el trabajo presentado por Kalyanpur et al [20], en donde se presenta una herramienta para la correlación automática de ontologías informáticas a clases Java. Es uno de los trabajos con los que se comparó a Sapphire en el artículo de Stevenson [29]. En este se relaciona a una clase Java con una clase de una ontología. A la clase generada se le realiza también la correlación con propiedades de objeto y de datos, también se respeta la relación de jerarquía definida en la ontología y las reglas aplicadas. La solución planteada se basa en utilizar interfaces de Java y herencia para lograr la herencia múltiple que java no posee, para representar por ejemplo la relación de jerarquía y de equivalencia entre clases, pero también se utiliza para realizar operaciones de unión, intersección, complemento, entre otras operaciones utilizadas en reglas de ontologías.

En cuanto a los editores de ontologías, estos proveen funcionalidades tales como soporte para creación de jerarquías de conceptos, relaciones, reglas e individuos. Algunos editores proveen funcionalidades adicionales como generación de formularios y soporte para el guardado de datos en ontologías, tal es el caso de la herramienta presentada en el trabajo de Weiten et al [33] en el que se presenta *OntoStudio*, un entorno para creación de ontologías propietario. Al comienzo se implementó como parte del entorno para desarrollo de ontologías *Protégé* [1] pero luego se separó y se comenzó a vender como un editor de ontologías completo. Está implementado como una extensión de Eclipse IDE [1] y su licencia es paga. Provee soporte para el guardado de ontologías, un mecanismo para corroborar satisfactibilidad y realizar inferencias sobre los datos ingresados, y soporte para establecer correspondencias entre conceptos y relaciones entre ontologías. Pero además provee una interfaz en la que se puede diseñar formularios para ingreso de datos en base a una ontología, los formularios se construyen utilizando todos los componentes de la ontología y se desarrollan en tiempo de diseño.

Con respecto a la generación de formularios para ingreso de datos en ontologías, además de OntoStudio existen otras herramientas que permiten realizar esta tarea.

[1] Página web de Eclipse IDE - `https://eclipse.org/` - Accedida: 2016-04-16

Tal es el caso de la herramienta presentada en el trabajo de Gonçalves et al [12], en donde se describe un método para generar formularios y guardar los datos ingresados utilizando ontologías como base para la creación de los formularios. Esta herramienta, a diferencia de OntoStudio, no es un editor de ontologías, sino que utiliza ontologías como insumo para realizar sus funciones. Se demuestra la aplicación de este método mediante el uso de una ontología médica en la que se describen los síntomas de los pacientes en el momento de realizar un diagnóstico cuando concurren a una consulta con un médico.
El proceso de creación de formularios se apoya en una ontología en la que se representa el dominio de los formularios, esta ontología se utiliza para modelar los formularios. Los formularios se especifican en la ontología y luego estos son generados a partir de esta. Para construir las preguntas se utiliza una ontología en la que se representa el dominio de la temática sobre la que se generan los formularios (en este caso la temática es síntomas). Para guardar los datos ingresados se utiliza una tercera ontología, en la que se almacenan todas las aserciones creadas como resultado del ingreso de datos en los formularios generados. Por lo tanto, para generar formularios sobre un dominio específico la herramienta presentada utiliza tres ontologías, una de ellas debe ser la ontología que representa el dominio, otra específica de la aplicación en la que se modelan los formularios y otra para guardar los datos ingresados al formulario.
En el trabajo no se menciona la realización de pruebas de esta herramientas con ontologías en las que se represente otro dominio que no sea el de síntomas de pacientes.

Continuando con los trabajos relacionados a la generación de formularios, en el libro presentado por Sachs et al [28] se presenta una guía de cómo utilizar Protégé-Frames, esta aplicación se encuentra basada en el paradigma de *frames* para modelado y desarrollo de ontologías.
Las principales características de *frames* se describen en el artículo de Wang et al [32] por medio de una comparación con el paradigma OWL para desarrollo de ontologías. Las características que *frames* tiene en común con OWL es que se modelan clases, instancias de dichas clases y relaciones entre clases. Por otro lado, difiere de OWL en que implementa el paradigma de *mundo cerrado*, es decir que una característica no se cumple hasta que explícitamente se exprese que se cumple. Un ejemplo de mundo cerrado es que dos individuos perteneciente a una misma clase se consideran diferentes salvo que explícitamente se aclare lo contrario. En el paradigma de mundo abierto sucede que dos individuos pertenecientes a una misma clase pueden ser potencialmente iguales salvo que se indique expresamente lo contrario. Otra diferencia fundamental es que en el paradigma de frames no se pueden realizar inferencias del conocimiento existente en la ontología, un individuo pertenece a las clases a las que se definió en forma explícita que lo hace. La capacidad de realizar inferencias se utiliza en OWL para clasificar individuos, debido a que un individuo puede pertenecer a cualquier clase salvo que se defina lo contrario con una restricción.

Protégé-Frames provee funcionalidades como crear jerarquías de clases, crear individuos y asociarlos a clases, crear relaciones, y también tiene la funcionalidad de creación de formularios para ingreso de datos en la ontología. Los formularios diseñados se pueden utilizar dentro de la aplicación y se crean con la finalidad de que la actividad de ingresar datos sea mas sencilla. Al comenzar el proceso de creación del formulario se muestra un formulario por defecto al cual se le pueden agregar, quitar, o cambiar de lugar los componentes. En la Figura 2.1 se muestra una captura de pantalla de Protégé-Frames en la que se encuentra un ejemplo de formulario creado a partir de la ontología implementada con el paradigma de frames. En el ejemplo se muestra la clase Editor, que tiene los atributos *nombre* y *salario*, y tiene una relación *responsable de*; los atributos y la relación se encuentran en el formulario como campos a completar al generar una instancia nueva de Editor.

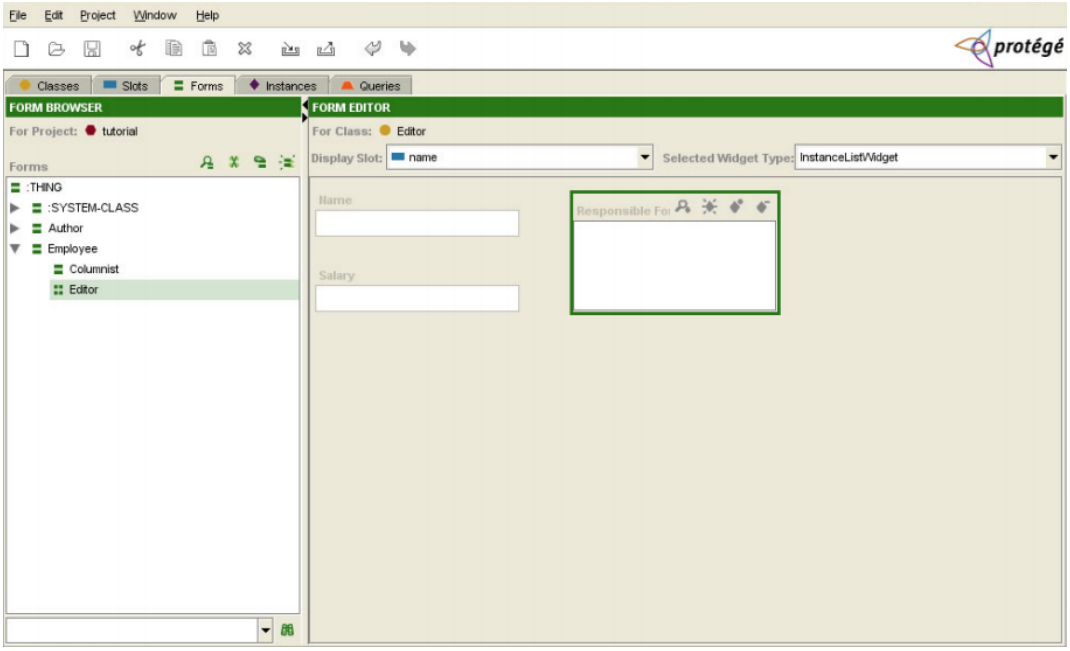

Figura 2.1: Ejemplo de formulario creado con Protégé-Frames. Tomado de Sachs et al [28].

Horridge et al [15], propone una forma de generar formularios para ingreso de datos para usuarios que conocen el dominio, pero no tienen conocimientos de ontologías. Se trabaja sobre ontologías ya implementadas, la propuesta no es un editor de ontologías completo, sino que solo se ingresan nuevos individuos a la misma. La especificación de los formularios es realizada por expertos en el dominio de la ontología a través de una aplicación. En la aplicación se crea la estructura de los formularios, en donde los campos de dichos formularios son clases y relaciones de la ontología.

Luego de realizada la etapa de creación de los formularios, estos se instancian cada vez que un usuario médico lo necesite. Los usuarios finales (doctores en este caso) se abstraen del uso de ontologías, es decir que no es necesario que tengan conocimiento de ontologías ni de ingeniería de conocimiento para utilizar los formularios. La solución propuesta se realizó para el dominio específico de documentación clínica, pero se aclara que puede ser utilizada para otros dominios.

Otro trabajo relevado sobre la generación de formularios más cercano a nuestras necesidades es el presentado en el articulo publicado por Hohmann y Fichtner [14] en el que se describe una herramienta para ayudar al ingreso de datos en ontologías mediante formularios. Estos formularios son generados en forma semiautomática con la ayuda de dicha herramienta. El proceso de generación permite crear una interfaz gráfica (el form en este caso) para el ingreso de datos a una ontología sin necesidad de simplificar o recortar la misma. El proceso de generación del formulario consta de elegir los conceptos y relaciones desde la ontología que se traducen a componentes del formularios, se identifica este proceso como definir la correspondencia entre el formulario y la ontología.
Se presenta en el artículo un caso de uso para una ontología de obras que se presentan en museos. A la fecha de la publicación del artículo varias aplicaciones existen que guardan y administran datos en bases de datos de tipo relacional de forma exitosa. Al utilizar la ontología como base de datos se elimina la necesidad de traducir los datos de la base de datos relacional a OWL, permitiendo por lo tanto que los datos se ingresen directamente a la ontología. Además de esto se pueden analizar las inferencias realizadas por OWL DL. Finalmente se destaca que al utilizar esta herramienta no es necesario incorporar conocimientos nuevos como lo es OWL, un lenguaje específico para desarrollo de ontologías.

Con respecto a la temática de generación de preguntas desde información presente en ontologías que representan dominios de conocimiento, en el trabajo presentado por Papasalouros et al. [26] se describe la generación de preguntas múltiple opción a partir de ontologías. La finalidad de este trabajo es la de crear pruebas de múltiple opción para estudiantes de forma automática. El generador de preguntas genera la sintaxis de la pregunta y además proporciona la opción correcta y un conjunto de opciones incorrectas. Estas opciones incorrectas están relacionadas con la pregunta que se está planteando, y por lo tanto son elementos de distracción. Se aclara que si bien se han presentado otros enfoques con respecto a este tema, el enfoque presentado es para ontologías que representan un dominio acotado y específico de la realidad. La construcción de las preguntas se basa en las relaciones binarias entre las clases y en la sentencia que define a qué clase pertenece un individuo. Además la construcción de las preguntas se apoya en una de las características más importantes de las ontologías que es que las relaciones son de tipo semánticas. Para lograr crear las preguntas y las opciones correspondientes se proponen varios enfoques.

Los autores construyeron un prototipo para corroborar el correcto funcionamiento del generador, dicho prototipo arrojó resultados exitosos, validando las estrategias de creación de preguntas propuestas por ellos.

Continuando en la generación de preguntas desde ontologías, en el trabajo de Tosic y Cubric [30] presenta una extensión para el editor de ontologías Protégé [21] creado con la finalidad de construir preguntas de tipo opción múltiple para cualquier dominio de conocimiento que se encuentre representado en una ontología. El sistema se creó con el objetivo de ayudar a docentes a construir pruebas de múltiple opción para estudiantes.
Se desarrolló un prototipo de sistema que permite la generación de preguntas de múltiple opción, la visualización de la ontología mediante una interfaz gráfica y también permite la realización de otras tareas (e.g.: gestionar grupos de estudiantes). Según Tosic y Cubric, el módulo de generación de preguntas múltiple opción se realizó basado en el trabajo de Papasalouros et al. (2008) [26].

Por último, en el trabajo relevado de Al-Yahya (2011) [3] se presenta OntoQue, un sistema que crea preguntas sobre un cierto dominio de conocimiento. La finalidad de estas preguntas es que sean utilizadas por docentes para las pruebas de los estudiantes. Según Al-Yahya, las preguntas propuestas por el sistema ayudan al docente en la tarea de realizar una prueba objetiva para evaluar el conocimiento de los estudiantes. La información acerca del dominio de conocimiento se obtiene por medio de ontologías.
Según Al-Yahya, el motor OntoQue es capaz de hacer tres tipos de preguntas diferentes: verdadero o falso, múltiple opción y completar el espacio en blanco. Para todos los tipos de preguntas se presentan tres enfoques diferentes: (i) uno se basa en una clase en particular y los individuos que pertenecen a ella, (ii) otro se basa en un individuo y las clases a las que este pertenece y (iii) el enfoque basado en propiedades que cumplen los individuos. El enfoque que consideramos relevante para nuestro proyecto es el (i), en dicho enfoque se pregunta por el *tipo* de una instancia, estos tipos aparecen como opciones.
El desempeño es medido por la cantidad de preguntas realizadas que son útiles para realizar evaluaciones, la utilidad de las preguntas fue decidida por expertos en el dominio de las ontologías utilizadas. Las pruebas realizadas por Al-Yahya sobre OntoQue han demostrado que tiene buen desempeño en ontologías basadas en individuos en lugar de ontologías de términos.

2.3. Conclusiones

En conclusión, en este capítulo se presentan trabajos relacionados con los temas abordados en el proyecto. En primera instancia se presentaron trabajos relacionados a la temática de la descripción del dominio de las piezas de arte por medio de ontologías en el campo de la filosofía, en este campo se desarrollan discusiones acerca de nuevas clasificaciones a tener en cuenta, en cambio en informática el objetivo es la construcción de una ontología concreta. Luego se presenta otra forma de representación formal del campo del arte en informática, un modelo de metadatos llamado CDWA, por medio del modelo planteado se afirma que es posible describir cualquier forma de arte conocida. Si bien el modelo permite identificar obras de arte, no se aproxima al modelo planteado por el experto en arte Vladimir Muhvich. El modelo del experto consiste en obtener datos de piezas de arte desde la experiencia sensorial, para luego poder clasificarlas en la familia de piezas de arte que corresponda. CDWA se plantea como un estándar de metadatos, por lo tanto los conceptos y relaciones pueden servir de referencia en el desarrollo de la ontología OPPA.

Luego, ingresando en la temática de correspondencia entre ontologías y otras estructuras, se presentan generadores de código a partir de ontologías. Los presentados generan código Java partiendo de ontologías ya especificadas. Por ejemplo en el caso de la jerarquía de conceptos esta se tradujo a una jerarquía de clases con herencia de atributos y métodos. Estas estructuras son generadas en tiempo de diseño y con el propósito de realizar la correspondencia para que luego pueda ser utilizada en el desarrollo de una aplicación. Es decir que se proponen generadores de código Java partiendo de una especificación en OWL. En nuestro proyecto los formularios son generados en tiempo real durante la ejecución de la aplicación y no es necesaria la generación automática de código.

En la linea de generadores de formularios se presentan dos editores de ontologías que permiten la generación de formularios especificados en tiempo de diseño de la ontología, pero para uso interno de la aplicación de edición. En el caso de Protégé-Frames se utiliza un paradigma de *frames* para la especificación de la ontología, un paradigma que difiere en varios aspectos con OWL. Una de las características de *frames* es que no es posible inferir conocimiento. En nuestro proyecto una razón de la realización de la ontología de percepciones de piezas de arte es la capacidad de inferir conocimiento para clasificar las piezas exhibidas en las familias que correspondan por lo tanto en nuestro proyecto no es posible la utilización de *frames*.

Se presentan también dos generadores de formularios que se crearon con la finalidad de abstraer al usuario final de saber utilizar ontologías. En ambos casos es necesario una instancia previa de configuración, que consiste en realizar una correspondencia entre conceptos, relaciones e instancias de la ontología, en campos en

el formulario, que son organizados en la etapa de configuración. Luego de la etapa de configuración, los formularios son creados automáticamente. En nuestro proyecto también se utiliza una etapa de configuración previo a la generación de formularios.

Finalmente se presentan trabajos relacionados a la temática de componer preguntas partiendo de texto inferido desde ontologías. Las estrategias son presentadas en el contexto de generar preguntas para pruebas de múltiple opción en universidades. Se basan en componer la pregunta a partir del nombre de relaciones y conceptos. El uso de estas estrategias se describe en el Capitulo 4.

En el siguiente capítulo se describe la generación de la ontología de Percepción de Piezas de Arte OPPA.

Capítulo 3

Ontología de características perceptibles de piezas de arte

En este capítulo se describe el proceso de diseño y construcción de una ontología que sirve para modelar las características perceptibles de piezas de arte en exposiciones, los medios de producción, los soportes y las familias a las que estas pertenecen. Se presentan los pasos seguidos en la aplicación de la metodología Menthontology para el diseño e implementación de ontologías.

3.1. Metodología utilizada para el desarrollo de la ontología

Para el desarrollo de la ontología, se aplicó METHONTOLOGY [10], un método sencillo y ampliamente utilizado para construir ontologías desde cero, como en este caso. En el diagrama presentado en la Figura 3.1 se muestran los estados en el ciclo de vida de una ontología. En primer lugar, en el recuadro inferior del diagrama, se encuentran las actividades de: *evaluación*, *documentación* y *adquisición de conocimiento*. Estas actividades se producen durante todo el ciclo de vida de la ontología. En particular, la etapa de adquisición de conocimiento implica la realización de entrevistas con expertos, la consulta de textos que tratan la temática a modelar, entre otros.
Luego, se encuentra el recuadro mayor del diagrama, es un diagrama de estados que representa los estados por lo que pasa una ontología durante su vida, dichos estados son: *especificación*, *conceptualización*, *formalización*, *integración*, *implementación* y *mantenimiento*. Por medio de estas etapas se logra cumplir con el objetivo de construir una ontología formal que modela una cierta realidad planteada. La etapa de mantenimiento de la ontología hace posible que a esta se le agreguen nuevos conceptos o que estos sean cambiados en la medida que sea necesario.

En la etapa de especificación se menciona el propósito de la ontología, la temática que trata, los usuarios finales y nivel de formalidad entre otros. El nivel de formalidad se refiere a la formalidad de los términos y relaciones utilizados para modelar la realidad.
Para la realización de la etapa de conceptualización en Corcho et al [7] se han definido una serie de pasos que guían la tarea. Tal como se muestra en la Figura 3.2, se dividió el proceso de conceptualización en 11 tareas, estas tareas, nombradas en orden de su ejecución, son: construir un glosario de términos, construir la taxonomía de conceptos, construir las relaciones binarias entre conceptos, construir un diccionario de conceptos, describir las relaciones binarias, describir los atributos de instancia, describir los atributos de clase, describir las instancias, describir los axiomas formales, describir las reglas y por último describir las instancias. Estas tareas se explican en detalle en las secciones posteriores mediante ejemplos de su utilización en la ontología desarrollada.
En la etapa de formalización se pretende que la realidad conceptualizada sea descrita en un lenguaje formal, como por ejemplo lógica de descripciones. La etapa de integración comprende la búsqueda de ontologías que se puedan utilizar para representar el dominio que se pretende modelar para evitar, si es posible, construir toda la ontología desde cero. La etapa de implementación consiste de escribir la ontología en un lenguaje que puede ser interpretado por computadoras (i.e., OWL, RDF, entre otros). Dicha implementación se puede hacer por medio de entornos de desarrollo de ontologías (i.e., Protégé). Por último la etapa de mantenimiento implica la actualización de la ontología, ya sea para agregar nuevos conceptos y relaciones, como para cambiar los existentes.

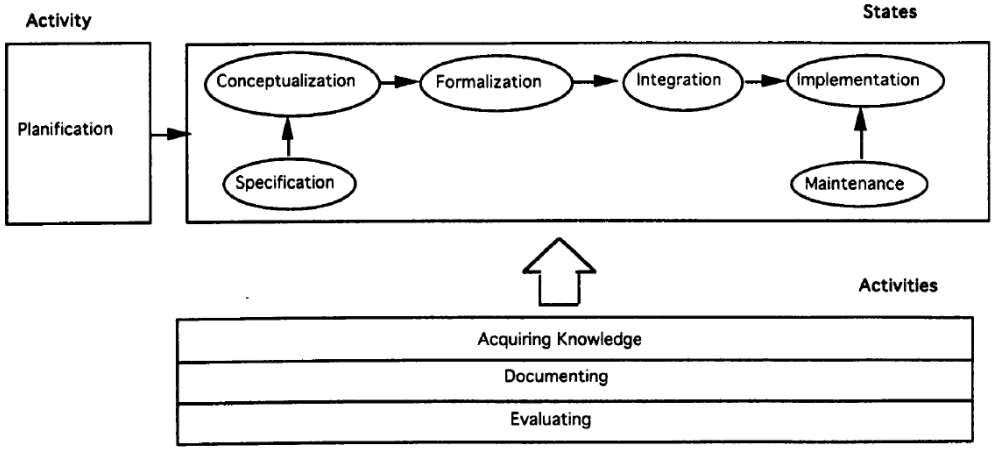

Figura 3.1: Proceso de desarrollo de una ontología según METHONTOLOGY. Obtenido de: Fernández et. al [10]

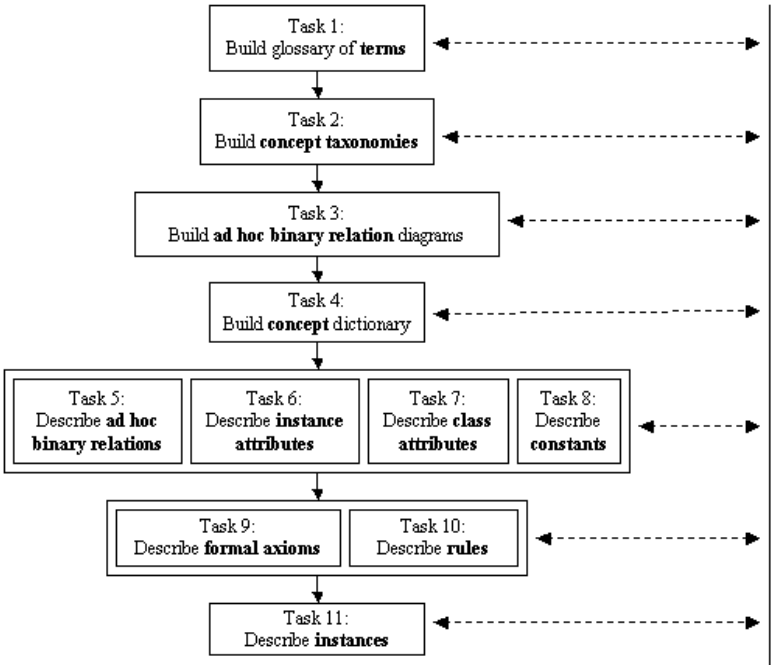

Figura 3.2: Pasos para el proceso de conceptualización de METHONTOLOGY. Obtenido de: Corcho et al [7]

Si bien el proceso de desarrollo se basó en METHONTOLOGY para la realización de la ontología, el mismo no se ha seguido al pie de la letra en todas las fases que se nombraron anteriormente. Las fases que no se han seguido son: etapa de formalización, integración y mantenimiento. Las demás etapas se considera que se siguieron de la misma manera que se especifica en la metodología.
Con respecto a la etapa de formalización, esta se realizó en forma parcial, dado que algunos componentes de la ontología no fueron traducidos a lógica de descripciones, por ejemplo la jerarquía de conceptos y los individuos que pertenecen a dichos conceptos.
La etapa de integración no se realizó debido a que no se encontró una ontología implementada en OWL que represente parcialmente el dominio representado.
La etapa de mantenimiento tampoco se realizó debido a que la ontología presentada es la primera versión, en las iteraciones siguientes se realizarán cambios para representar en forma mas ajustada el dominio que se representa.

En el resto del presente capítulo se explica el pasaje por las etapas de especificación, conceptualización e implementación.

3.2. Especificación de la realidad

El objetivo de la creación de la ontología es conceptualizar la realidad de las características perceptibles de las piezas de arte. Estas piezas se encuentran en *exposiciones*. Las piezas de arte pueden estar en exposiciones por tiempo limitado, por lo tanto una pieza puede estar en distintas exposiciones en diferentes momentos. Al cambiar el lugar en el que la pieza se encuentra expuesta se pueden afectar las características que se perciben desde lo sensorial. Además puede suceder que una misma obra se presente con cambios en diferentes exposiciones, esto también puede cambiar las características perceptibles de las piezas. Por lo tanto las características susceptibles a cambios en el contexto de la ubicación de la pieza estarán asociadas a las piezas en una exposición. A las piezas de arte en una exposición las llamamos en la ontología **PiezaExposición**.
Además de las percepciones, las piezas en una exposición tienen medios de producción, soportes de medios de producción y propiedades como nombre, autor y año de creación.

Las piezas de arte en exposiciones se clasifican en *Familias* de piezas de arte. A partir de reglas que definen la pertenencia a una familia de una obra ubicada en una exposición, el mecanismo de razonamiento de la ontología clasifica dicha pieza en la familia más especifica a la que pertenece. Las reglas utilizadas para la clasificación están basadas en las características de la pieza desde los sentidos junto con las relaciones que tiene la pieza en la exposición con respecto a otros conceptos de la ontología.

La ontología se desarrolló con el objetivo de que los usuarios finales sean expertos en el campo del arte, por lo tanto la terminología utilizada es comprendida por ellos. Además, la ontología se desarrolló con la finalidad de ser utilizada como medio de entrada para realizar formularios a ser completados por medio de una interfaz externa al editor de ontologías (uno de los requerimientos del experto para su aplicación), por lo tanto la forma que tiene la ontología es en parte afectada por la visión de formulario del experto en arte Vladimir Muhvich.

En la siguiente sección se describen los conceptos y las relaciones que ocurren en la ontología de percepciones sensoriales.

3.3. Proceso de conceptualización

Según Fernández et al [10], el proceso de construcción de una ontología consta de cuatro etapas diferenciadas: *especificación*, *conceptualización*, *formalización*, *integración*, *implementación* y *mantenimiento*. En la sección 3.1 se explicaron las etapas y

su utilización en el proceso de desarrollo de la ontología de percepciones de piezas de arte. En esta sección se muestra el proceso para realizar la etapa de conceptualización. La explicación del paso a paso para la aplicación de este método se encuentra explicada en Fernández et al (1997) [10]. Se utilizó una interpretación del método de Fernández et al propuesta por Corcho et al [7] como referencia para la realización de la etapa de conceptualización, en dicha interpretación se realiza una explicación mas detallada de las actividades a realizar en esta etapa y se la subdivide en pasos numerados.

Paso 1. El primer paso de la metodología fue definir los términos a utilizar. En este paso se debe escribir el nombre de todos los términos, y agregar sinónimos, acrónimos, una descripción y el tipo de término (i.e. concepto, constante, atributo de clase, atributo de instancia y relación) en donde corresponda. En la Tabla 3.1 se presenta la realización del paso 1 de una selección de términos de distinto tipo pertenecientes a la realidad conceptualizada, en dicha tabla se puede observar que se tiene en cuenta cómo términos a utilizar a los conceptos, atributos y relaciones. No se adicionó la columna de acrónimos debido a que en la realidad a modelar no hay elementos a los que se los conozca por un acrónimo.

Tabla 3.1: Paso 1 de methontology: definición de términos a utilizar.

nombre	*sinónimos*	*descripción*	*tipo*
pieza de arte	obra de arte	Pieza de producción artística.	concepto
exposición de piezas de arte	-	Espacio en el que se exhiben las piezas de arte.	concepto
percepción auditiva	-	Se refiere al término que representa la percepción auditiva.	concepto
altura	-	Altura del sonido que se percibe	concepto
pieza en exposición	-	pieza de arte se exhibe en una exposición	concepto
tiene percepción auditiva	-	Característica del sonido de una pieza en una exposición.	relación
tiene exposición	-	Exposición en la que se encuentra una pieza de arte.	relación
tiene autor	-	Autor de una pieza de arte	atributo de instancia
Pablo Picasso		Nombre de un autor de piezas de arte	instancia

Paso 2. El segundo paso de la metodología consta de construir una taxonomía con los conceptos identificados en el primer paso. En la Figura 3.3 se presenta un diagrama de una parte de la taxonomía, en dicha figura se muestra que se encuentran cuatro conceptos de mayor jerarquía, estos son: *Pieza, Encuesta, Pieza exposición*

y Percepción auditiva . Se señala que una clase es subclase de otra mediante la relación *subclase de*. De los cuatro conceptos de mayor jerarquía, únicamente Percepción auditiva tiene subclases.

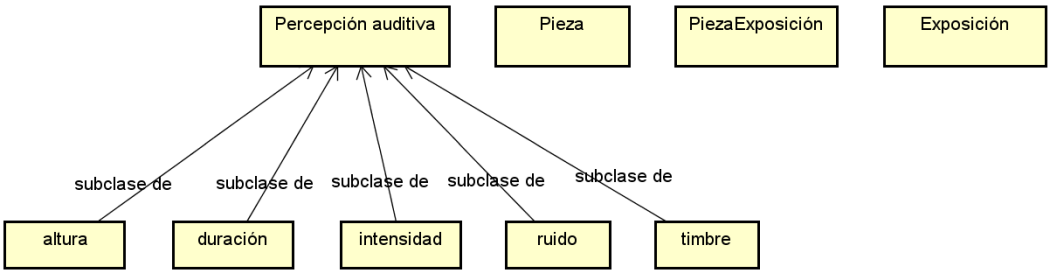

Figura 3.3: Paso 2 de METHONTOLOGY: realización de la taxonomía de conceptos.

Paso 3. El tercer paso es construir un diagrama de relaciones binarias. En este diagrama se definen las relaciones directas e inversas entre dos conceptos. En la Figura 3.4 se presenta el diagrama para el tercer paso de METHONGTOLOGY, se realizó el diagrama de relaciones binarias del sector de la taxonomía utilizada como ejemplo para el paso dos. En dicho diagrama se puede observar que PiezaExposición está relacionada con las subclases de Percepción auditiva. En dichas relaciones PiezaExposición es el dominio de la relación, mientras que las subclases de Percepción auditiva son el recorrido. Si bien Exposición y Pieza tienen relaciones con otras subclases, dichas relaciones se omitieron para mantener el ejemplo pequeño.

Paso 4. El paso cuatro de la metodología consiste en construir un diccionario de conceptos. En la Tabla 3.2 se muestra una parte del diccionario de conceptos desarrollado a modo de ejemplo. Para cada concepto se debe especificar, en caso de que corresponda: nombre, instancias, atributos de clase, atributos de instancia y las relaciones de las que es dominio. El campo de instancias de Pieza, Exposición y PiezaExposición se completó con instancias de ejemplo debido a que las instancias de dichas clases no están definidas en la etapa de desarrollo de la ontología, serán ingresadas por medio de usuarios de la aplicación al completar los formularios.

Paso 5. El paso cinco de la metodología es describir detalladamente las relaciones entre individuos. En la Tabla 3.3 se encuentran detalladas las relaciones descritas en el primer paso de la metodología. Las relaciones que se detallan son *tiene percepción auditiva* y *tiene exposición*. Los datos que se detallan para cada una de las relaciones son: el nombre de la relación, el concepto origen, la cardinalidad máxima de la relación, el concepto destino y la relación inversa.

Paso 6. El sexto paso de methontology es describir detalladamente los atributos

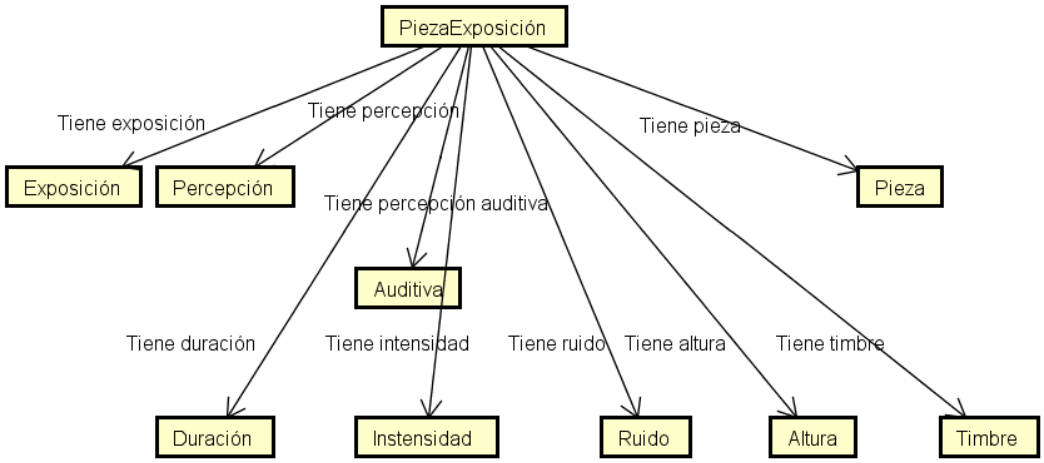

Figura 3.4: Paso 3 de METHONTOLOGY: realización del diagrama de relaciones binarias.

Tabla 3.2: Paso 4 de METHONTOLOGY: construcción del diccionario de conceptos.

nombre del concepto	*instancias*	*atributos de clase*	*atributos de instancia*	*relaciones*
pieza	Mona lisa, El grito	-	fecha de creación, URL imagen	tiene autor
exposición	Muestra de Persia antigua, Colección de artesanías de Brasil	-	fecha de comienzo	tiene curador, realizada en, tiene pieza en exposición
percepción auditiva	-	-	-	-
altura	alto, grave	-	-	-
pieza exposición	cuest_1, cuest_2	-	-	Tiene exposición, tiene percepción, tiene percepción auditiva, tiene altura, tiene pieza.

de instancias. Se ejemplificará mediante los atributos de instancia de la clase *Pieza*. La Tabla 3.4 es la correspondiente a este paso, en dicha tabla se encuentran los

Tabla 3.3: Paso 5 de METHONTOLOGY: descripción detallada de las relaciones binarias.

Nombre de la relación	*Concepto origen*	*Cardinalidad máxima*	*Concepto destino*	*Relación inversa*
tiene percepción auditiva	PiezaExposición	N	Auditiva	-
tiene exposición	PiezaExposición	N	Exposición	tiene pieza en exposición

detalles de los atributos: alto, ancho, largo, imagen y fecha. Para cada atributo se debe indicar (si corresponde) el nombre del atributo, el concepto, el tipo de valor, el rango de valores y la cardinalidad. En el caso de los enteros, debido a que indican la magnitud de una dimensión de la medida de una pieza, se indica que son positivos y mayores o iguales a 1 en el caso de alto y largo, mientras que en el caso de ancho es mayores o iguales a 0. En cuanto a las dimensiones, dado que varias piezas pueden tener igual valor en alguna (o todas) las dimensiones, entonces la cardinalidad es N-1. Lo mismo sucede en el caso de las URL de imágenes, varias Piezas podrán tener la misma imagen que las represente.

Tabla 3.4: Paso 6 de METHONTOLOGY: descripción detallada de los atributos de las instancias.

atributo de instancia	*concepto*	*tipo*	*rango*	*cardinalidad*
alto	pieza	entero	1..	(N,1)
ancho	pieza	entero	1..	(N,1)
largo	pieza	entero	1..	(N,1)
imagen	pieza	URI	-	(N,1)
fecha creación	pieza	fecha y hora	-	(N,1)

Paso 7. El paso siete es realizar una tabla que describa los atributos de clase. Los atributos de clase son aquellos que todos los individuos que integran la clase poseen, solo por el hecho de pertenecer a dicha clase. En la ontología de percepciones no se definió ningún atributo de clase, por lo tanto no se realizara la tabla correspondiente a este paso.

Paso 8. El paso ocho consta de detallar las características de las constantes, en

este paso de la metodología tampoco fue necesario realizar la tabla debido a que no hay constantes en la ontología.

Paso 9. El paso nueve consta de definir los axiomas formales. Los axiomas formales son expresiones que tienen la particularidad de ser siempre verdaderos y por lo general se utilizan para expresar restricciones. En el caso de la ontología que se desarrolló no se identificaron axiomas formales.

Paso 10. El paso número diez consiste en definir las reglas. Las reglas son utilizadas por el razonador para realizar inferencias sobre estas y generar conocimiento nuevo. En el caso de la ontología de percepciones, se utilizó para la clasificación de las piezas en las exposiciones en sus respectivas familias de producción artística.
En la Tabla 3.5 se encuentra un ejemplo de regla tomado de la ontología de percepciones, este ejemplo corresponde a la condición que tiene que cumplir una pieza en una exposición para pertenecer a la clase de las familias de producción bidimensional. Para pertenecer a dicha clase tiene que cumplir que el único medio de producción que puede tener es quirográfico, la pieza de arte tiene que ser una imagen fija, y el único movimiento perceptible debe ser el movimiento del espectador al circular frente a la obra.

Tabla 3.5: Paso 10 de methontology para el diseño de ontologías. Definición de las reglas.

nombre de la regla	*descripción*	*expresión*	*conc.*	*atr.*	*rel.*	*var.*
Piezas en exposición que pertenecen a la familia de producción bidimensional	Una pieza en una exposición que tiene medio de producción quirográfico, tiene movimiento real del espectador de tipo frontal y percepción visual aplicada a imagen fija pertenece a la familia de producción bidimensional	Si (existe(?X) y solo quirográfico(?X) y [tiene percepción visual aplicada a imagen fija](?X,?Y) y [tiene movimiento real del espectador](?X,?W)) entonces [producción bidimensional](?X)	solo quirográfico, producción bidimensional	-	tiene percepción visual aplicada a imagen fija, tiene movimiento real del espectador	?X, ?Y, ?W

Paso 11. El paso once es describir las instancias. En el primer paso de la metodología se han omitido las instancias en la Tabla 3.1 para evitar ocupar espacio

adicional. En este paso se detallan los individuos correspondientes a la clase *Ruido*, estos individuos son: *Constante*, *Fluctuante*, *Impulsivo* e *Intermitente*. La representación de estos individuos y sus características se encuentra en la Tabla 3.6. Algunas clases tendrán una cantidad de individuos que no estará definida en un comienzo (e.g., PiezaExposición, Pieza, Exposición) debido a que la asignación de individuos a estas clases se hará en forma progresiva y cooperativa por diferentes usuarios.
En la Tabla 3.6 también se encuentra otro caso, el de un individuo representativo de la clase Pieza. Antes de comenzar a utilizarse la aplicación no existirán individuos dentro de la clase Pieza debido a que estos serán creados e insertados cooperativamente por medio de los formularios. Por esta razón, se muestran los atributos sobre un individuo supuesto de la clase Pieza que existirá en el futuro. Este individuo es representante de todos los individuos que pertenecerán a la clase Pieza que serán creados e insertados por medio de la aplicación.
Para la clase Pieza se muestran dos de los atributos de instancias que posee: en primer lugar se muestra la fecha de realización de la pieza, y en segundo lugar un atributo de nombre imagen. El caso de los atributos de instancia es parecido al de la instancia en si: los valores de los atributos tampoco se encuentran definidos debido a que la instancia que los posee tampoco existe. Debido a que no tienen valores definidos, a modo de ejemplo se les otorgó valores. El atributo imagen no es un atributo en el que se guarda una imagen, es en realidad una URL a una imagen.

Tabla 3.6: Descripción detallada de las instancias. Onceavo paso de methontology para el diseño de percepción auditiva.

Nombre de la instancia	*Nombre del concepto*	*Atributo*	*Valores*
Constante	Ruido	-	-
Fluctuante	Ruido	-	-
Impulsivo	Ruido	-	-
Intermitente	Ruido	-	-
Pieza_ind	Pieza	Fecha de realización	23-09-2013
Pieza_ind	Pieza	Imagen	http://www.urlej.com/imagen_ej

Luego de utilizar Methontology se alcanzó un diseño básico de la ontología. Tomando este diseño preliminar como punto de partida, se realizaron entrevistas con el experto en el dominio para validar todos los componentes de la ontología y las relaciones entre ellos. Se realizó un total de 15 reuniones en el trascurso del proyecto. Al terminar el proceso de validación se constataron cambios con respecto al resultado luego de aplicar únicamente Methontology.

Ejemplo de cambio luego de finalizado el proceso. Un ejemplo represen-

tativo de estos cambios es la percepción visual aplicada a espacios. En este caso, en la versión luego de haber aplicado methontology, el espacio no era subclase de percepción visual, sino que era una clase al mismo nivel que Percepción. Espacios se encontraba relacionado con PiezaExposición, al igual que en la última versión de la ontología. El experto afirmó que el espacio se percibe desde lo visual, por lo tanto se ingresó a la jerarquía de percepciones como subclase de Visual, que es subclase de percepción. La jerarquía de subclases se puede apreciar en la Figura 3.5, en dicha figura se puede observar que espacios se encuentra dividido por su contenido. El Espacio puede estar formado por una o varias imágenes fijas, imágenes en movimiento, o puede contener objetos. Los objetos pueden ser uno o varios de tipo bajo relieve, bulto, seres vivos, u otros objetos.

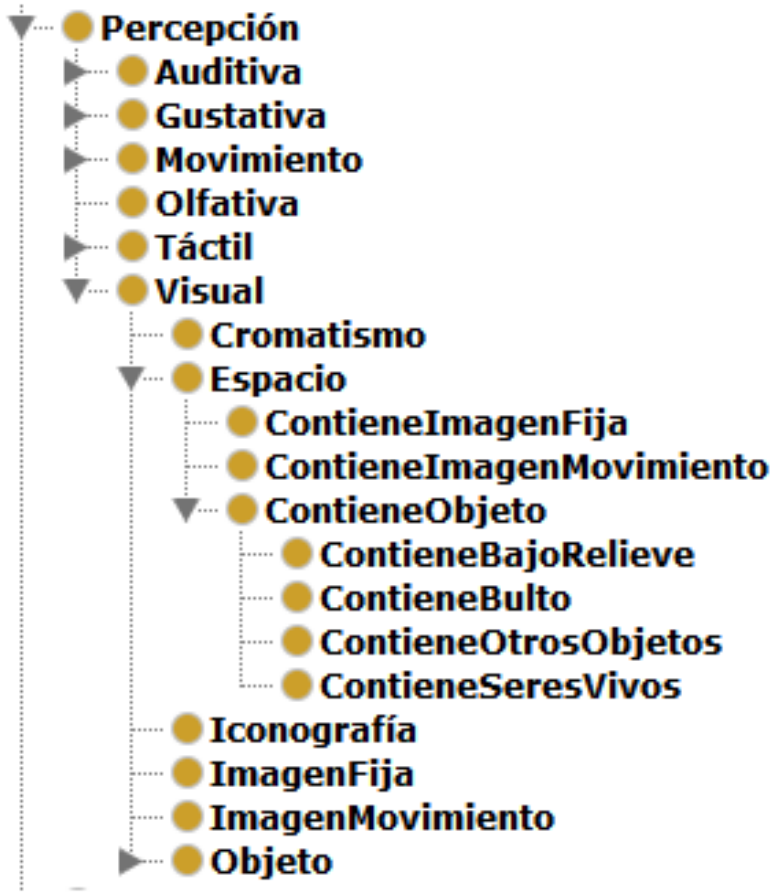

Figura 3.5: Jerarquía de clases para la clase *Percepción*, tomada de Protègè OWL 4.0.

3.4. Componentes de la ontología

En la ontología se conceptualizan tres grandes componentes de la realidad: (i) *Percepciones* sensoriales, información sobre características que se perciben en lo sensorial de una obra, (ii) *Medios de producción*, características de los medios de producción utilizados en las obras, y (iii) *Soportes de medios de producción*, características de los soportes utilizados en ellas.

Percepciones

Las percepciones describen las características de las piezas de arte desde el enfoque de lo que aprecian los sentidos del ser humano. Una clasificación primaria de la percepción es dividirla en seis tipos, las primeras cinco son las asociadas a los cinco sentidos del ser humano (i.e. vista, olfato, gusto, tacto, y oído), el sexto tipo de percepción es la percepción de movimiento, esta última se refiere a la descripción del movimiento de una pieza desde lo sensorial. A partir de esta separación se creó una jerarquía de conceptos de percepción en la que a su vez, cada tipo de percepción se clasifica en sub tipos de acuerdo a características especificas, por ejemplo el concepto de percepción de gusto tiene los subconceptos de Amargo, Dulce, Salado, etc. En la Figura 3.5, se muestra una imagen de la jerarquía de clases de percepción. El concepto *Percepción* se encuentra conformado por seis subclases, dichas subclases son los seis tipos de percepción. Los individuos de percepción se encuentran asociados a las clases que no tienen subclases, estos individuos representan en algunos casos, las características de percepción más específicas para una jerarquía (e.g. *Superficie lisa* es individuo de percepción visual aplicada a *imagen fija*), en otros casos representan formas de evaluar cierto aspecto de una percepción particular (e.g. varios componentes fuera del campo del arte es individuo de *percepción visual aplicada a espacios*).

La jerarquía de percepciones se encuentra relacionada con el concepto PiezaExposición. Cada concepto de la jerarquía de percepciones tiene una propiedad de objeto que lo relaciona con PiezaExposición, siendo PiezaExposición el dominio de la propiedad, mientras que el concepto que es subclase de percepciones es el recorrido. La correspondencia entre la jerarquía de conceptos y las relaciones, originó que se organizaran las propiedades de objeto para percepciones como una jerarquía con una estructura jerárquica análoga a la de percepciones.

La creación de la jerarquía de percepciones es necesaria debido a que en algunos casos un individuo de la clase PiezaExposición puede estar relacionado con un solo elemento de una subclase. Por lo tanto es necesario definir una propiedad de objeto que sea funcional entre PiezaExposición y el concepto con el que se puede relacionar solo por medio de un individuo a la vez. Ademas de definir que una re-

lación es funcional es necesario explicitar que todos los individuos que pertenecen al concepto son distintos. En caso de que no se explicite que los elementos son distintos, si se definen dos instancias de relaciones que involucran al mismo individuo como componente del dominio de ambas y a dos individuos (que a priori son diferentes) cada uno como componente del recorrido de una instancia de la relación, se infiere que los dos individuos del recorrido son el mismo (i.e. uno es un alias del otro).

Medios de producción

Las obras se pueden producir con dos tipos de medios de producción: *Quirográficos* (i.e. hecho a mano) o *Tecnográficos* (i.e. hecho con tecnología). Un obra que tiene un medio de producción quirográfico es aquella en la que se utilizaron herramientas no eléctricas para su realización, las obras con medio de producción tecnográfico son aquellas en las que se utilizó una o más herramientas eléctricas.
El medio de producción quirográfico se encuentra dividido en Bidimensional y Tridimensional dependiendo del uso que se le otorga a dicho medio. El medio tecnográfico se dividió en Analógico y Digital dependiendo de si el medio de producción es o no electrónico.

Soportes

La jerarquía desarrollada para soportes se divide en tres subclases principales: los soportes para producción quirográfica, para producción tecnográfica y los seres vivos como soporte para piezas de arte. Las clases de soporte para producción quirográfica y tecnográfica tienen a su vez una jerarquía de subclases asociadas.

3.5. Familias

Dependiendo de la relaciones que presente una pieza en una exposición con las percepciones y los medios de producción, se realiza la clasificación de obras de arte en familias. Las familias se componen de individuos de piezas de arte en exposiciones (i.e. PiezaEsxposición). Las familias en la ontología fueron creadas con la finalidad de clasificar en ellas a las obras ingresadas. En el campo del arte existen cuatro familias que se conceptualizaron en la ontología, estas son: producción bidimensional, producción tridimensional, nuevos soportes y medios inestables.

Las familias tienen condiciones que determinan que una pieza en una exposición pertenezca a ellas. Se identificaron las características necesarias y suficientes que hacen que una obra pertenezca a una familia en particular y se tradujeron dichas

características a *reglas* en la ontología. De esta manera se logró que las obras fueran clasificadas en las familias correspondientes.

El proceso de identificación de características necesarias y suficientes de pertenencia de una obra a una familia fue un trabajo multidisciplinario en conjunto con el experto en el campo del arte. Se llevaron a cabo varias reuniones en las que se logró comprender la realidad de las familias de piezas de arte. Como resultado de dichas reuniones se obtuvo una especificación de qué características debe cumplir una instancia de PiezaExposición para pertenecer a una familia en particular. Luego se tradujeron estas características a reglas que se puedan verificar en la ontología desarrollada. Dicho trabajo comprendió varias reuniones que se llevaron a cabo en el trascurso del proyecto.

El diseño inicial de la ontología consta de cuatro familias: producción bidimensional, producción tridimensional, nuevos soportes y medios inestables. En el caso de los nuevos soportes y los medios inestables además se definieron las características de tipos de obras que pertenecen a las familias. Para que una pieza pertenezca a uno de estos tipos de obra tiene que cumplir reglas definidas en la ontología. Estos tipos de obras fueron modelados como subclases de las familias a las que pertenecen.

Las reglas en una ontología se especifican en Lógica de Descripciones (DL). DL se creó como un lenguaje basado en lógica para la representación de dominios de conocimiento, es posible realizar una traducción de DL a lógica de primer orden. La teoría sobre DL se encuentra en Rudolph [27]. Según Rudolph, $SROIQ$ [17] es la lógica de descripciones en la que se basa OWL 2 DL [1]. $SROIQ$ se basa en tres elementos primarios: nombres de individuos (N_I), nombres de conceptos (N_C) y nombres de roles (N_R). Los elementos básicos anteriores son los que conforman las bases de conocimiento, estas bases de conocimiento se construyen con tres elementos llamados: RBox, TBox y ABox. En la RBox se encuentran definidas las dependencias entre los roles (i.e.: relaciones o propiedades de objeto en OWL 2), un ejemplo es $tienePercepciónVisual \sqsubseteq tienePercepción$. La TBox es donde se encuentra la definición de los conceptos (i.e.: clases), un ejemplo es la Ecuación 3.1. Todas las reglas definidas sobre las diferentes familias pertenecen a la TBox. En la ABox es donde se definen las afirmaciones sobre individuos en particular, ejemplos de aserciones de la ABox se encuentran en la Ecuación 3.5.

A continuación se presenta una descripción detallada de las condiciones (i.e.: reglas) que tiene que cumplir un individuo de PiezaExposición para pertenecer a una determinada Familia. Las reglas se explican en lenguaje natural y luego se muestra

[1] Owl 2 web ontology language document overview (second edition). `https://www.w3.org/TR/owl2-overview/`. Accedida: 2016-04-06.

su representación en DL.

Familia de producción bidimensional. Las piezas de arte en una exposición que se encuentran contenidas en la familia de producción bidimensional cumplen la condición de que son imágenes fijas, que el único movimiento es del espectador al visualizarlas, y que su medio de producción es solo quirográfico. La representación en DL de la familia de producción bidimensional de las piezas en una exposición se encuentra en la Ecuación 3.1. Según Rudolph [27], la regla se encuentra formada por roles, clases e individuos. Las relaciones de objeto son aquellas cuyos nombres comienzan en minúscula, los nombres de las clases comienzan con mayúscula y los nombres de los individuos comienzan con mayúscula de igual forma que las clases, pero además siempre tienen delimitadores de conjunto. Luego de ser expresada en DL, la regla se tradujo a lenguaje OWL 2 y se le asocia a la clase *ProducciónBidimensional* con la finalidad de ser utilizada para inferir la pertenencia de un individuo de PiezaExposición a la clase *ProducciónBidimensional*.

$$\begin{aligned} ProducciónBidimensional \equiv\ & Familia \\ & \sqcap \exists tieneMedioDeProducción.(Quirográfico) \\ & \sqcap \exists tienePercepción.(\{Circulaciónfrontal\}) \\ & \sqcap \exists tienePercepción.(ImagenFija) \end{aligned} \tag{3.1}$$

Familia de producción tridimensional. Para pertenecer a la familia de producción tridimensional, una pieza en una exposición tiene que cumplir que el único movimiento que se percibe es real por parte del espectador, que es un elemento de producción sólo quirográfico, ó sólo tecnográfico ó sólo quirográfico y tecnográfico analógico, y también que son piezas de arte que no tienen energía. Estas características se encuentran representadas en Lógica de Descripciones en la Ecuación 3.2.

$$\begin{aligned} ProducciónTridimensional \equiv\ & Familia \\ & \sqcap (\exists tieneMedioDeProducción.(Quirográfico) \\ & \sqcup \exists tieneMedioDeProducción.(TecnográficoAnalógico) \\ & \sqcup (\exists tieneMedioDeProducción.(TecnográficoAnalógico) \\ & \sqcap \exists tieneMedioDeProducción.(Quirográfico))) \\ & \sqcap \exists tienePercepción.(MovimientoEspectador) \\ & \sqcap \exists tienePercepción.(Objeto) \\ & \sqcap \exists tieneEnergía.(\{false\}) \end{aligned} \tag{3.2}$$

Familia de los nuevos soportes. La piezas en exposiciones que pertenecen a la familia de los nuevos soportes tienen la característica de que el único medio de producción que pueden tener es tecnográfico analógico. Todos los tipos de piezas que se definieron como subclases de nuevos soportes, por el hecho de ser subclase de nuevos soportes tienen definida dicha regla implícitamente. Cada tipo de pieza tiene definido un conjunto de reglas específicas además de la regla implícita de la clase padre a la que pertenecen. En la Ecuación 3.3 se encuentra representada la regla completa en lógica de descripciones.

$$\begin{aligned} NuevosSoportes \equiv\ & Familia \\ & \sqcap \exists tieneMedioDeProducción.(TecnográficoAnalógico) \end{aligned} \tag{3.3}$$

Familia de los medios inestables. En la familia de los medios inestables, todas las obras tienen la característica de que tienen energía y además tienen solamente medio de producción tecnográfico digital. La regla escrita en notación de lógica de descripciones se encuentra en la Ecuación 3.4.

$$\begin{aligned} NuevosSoportes \equiv\ & Familia \\ & \sqcap \exists tieneMedioDeProducción.(TecnográficoDigital) \\ & \sqcap \exists tieneEnergía.(\{true\}) \end{aligned} \tag{3.4}$$

Dentro de la familia de los medios inestables y la de los nuevos soportes existen algunos tipos de piezas de arte cuya pertenencia a alguna de estas dos familias fue definida de facto. Por definición del campo del arte, las piezas de arte de la familia de arte, animación, cine y video analógico, así como el arte sonoro de tipo analógico, pertenecen a la familia de los medios inestables. Estas familias constituyen excepciones a la familia de los medios inestables, esto es debido a que una de las reglas de la pertenencia de una pieza en una exposición a dicha clase es que ésta última tenga medio de producción tecnográfico digital únicamente, lo cual no se cumple.
La excepción dentro de la familia de los nuevos soportes se encuentra en el caso del *Body Art*, en este tipo de pieza de arte siempre se utiliza uno o varios medios de producción quirográficos. Según el experto del dominio, el body art es una rama del arte en la que el cuerpo de un ser humano vivo forma parte de la pieza creada. Si bien por definición del campo del arte esta rama del arte pertenece a la familia de los nuevos soportes, no cumple con la regla de haber sido producida únicamente con medios de producción tecnográfico analógico y por lo tanto constituye una excepción. La solución de diseño fue la de crear clases para las excepciones. En el caso de las excepciones a la familia de medios inestables, estas fueron declaradas como subclase

de una nueva clase llamada *MediosInestablesExcepciones*. Las excepciones a la familia de los nuevos soportes fueron declaradas como subclase de una clase llamada *NuevosSoportesExcepciones*.

Ejemplo de pieza en exposición y su clasificación

Se muestra mediante un ejemplo la clasificación de un individuo de PiezaExposición en una familia determinada. El ejemplo comprende la clasificación en la familia de producción bidimensional. Se denominará a la instancia a clasificar *PE_Ejemplo*. Con la intención de simplificar el ejemplo se destacarán solamente las instancias de relaciones relevantes para la clasificación. En la Ecuación 3.5 se puede observar las aserciones definidas sobre el individuo. Estas aserciones pertenecen a la ABox de $SROIQ$. La primera afirmación indica que $PE\_Ejemplo$ pertenece a $PiezaExposición$, dado que $PiezaExposición \equiv Familia$ entonces $PE\_Ejemplo$ también pertenece a $Familia$.

$$
\begin{aligned}
&RBox: \\
&tieneTécnicanGrabado \sqsubseteq tieneMedioDeProducción \\
\\
&ABox: \\
&PiezaExposición(PE\_Ejemplo)(i) \\
&tienePieza(PE\_Ejemplo, Mona\_lisa)(ii) \\
&tienePercepción(PE\_Ejemplo, Circulación\_frontal)(iii) \\
&tienePercepción(PE\_Ejemplo, Superficie\_lisa)(iv) \\
&tieneTécnicanGrabado(PE\_Ejemplo, Litografía)(v) \\
&ImagenFija(Superficie\_lisa) \\
&Quirográfico(Litografía)
\end{aligned}
\tag{3.5}
$$

La afirmación (i) indica que $PE\_Ejemplo$ pertenece a $PiezaExposición$, dado que $PiezaExposición \equiv Familia$ entonces $PE\_Ejemplo$ también pertenece a $Familia$. Luego, en (ii) se especifica que $PE\_Ejemplo$ se encuentra relacionado con la pieza de nombre $Mona\_Lisa$ por medio de la relación $tienePieza$.
Las aserciones (iii), (iv) y (v) son las que definen la pertenencia a la familia de producción bidimensional. En dichas aserciones se define respectivamente: que el movimiento percibido es la circulación frontal del espectador al observar la obra, que se percibe visualmente que la superficie es lisa y que la técnica de grabado utilizada

es litografía.
La aserción no numerada perteneciente a la RBox indica la dependencia entre los roles *tieneTécnicaGrabado* y *tieneMedioDeProducción*. Las aserciones no numeradas pertenecientes a la ABox indican que *Superficie_lisa* pertenece a la clase *ImagenFija* y que *Litografía* es un tipo de medio de producción *Quirográfico*. Se cumple entonces la regla de pertenencia a la *Familia de producción bidimensional* y por lo tanto será inferido que $ProducciónBidimensional(PE\_Ejemplo)$.

3.6. Implementación

Esta sección describe la implementacion de OPPA, la Ontología de Percepciones de Piezas de Arte.
La etapa de implementación se realizó utilizando como entorno de desarrollo al editor de ontologías Protégé [1] y OWL 2 [2]. En la Figura 3.6 se muestra una impresión de pantalla del editor. A la izquierda de la imagen, en la parte superior, se encuentra la jerarquía de clases implementada. Se encuentran las clases principales de las jerarquías de percepción (i.e. Percepción), medios de producción (i.e. MedioDeProducción), soportes de medios de producción (i.e. Soporte) y familia (i.e. Familia). En la parte inferior izquierda se encuentra la jerarquía de propiedades implementada.
Se encuentra seleccionado el concepto *ProducciónBidimensional*, en la parte central inferior se muestran las aserciones para dicho concepto y en la parte superior se muestran los usos de la clase en la ontología. Las aserciones definidas se encuentran divididas en dos partes: aserciones de equivalencia (i.e. Equivalent To) y aserciones de subclase (i.e. Subclass of). En las aserciones de equivalencia se definen las reglas que tiene que cumplir una pieza en una exposición para pertenecer (en este caso) a la familia de producción bidimensional, cuando hay aserciones de equivalencia la clase es una clase definida o *defined class*. Las aserciones de subclase son las definidas al construir la jerarquía de clases.

3.7. Conclusiones

Este capítulo muestra el proceso seguido para la producción de la ontología de percepciones y medios de producción de piezas de arte. Se explicó la realidad conceptualizada, los componentes de la ontología y METHONTOLOGY (la metodología de desarrollo utilizada) y su aplicación en el diseño e implementación de la ontología de percepciones de piezas de arte. Se explicó la jerarquía de percepciones, de medios de producción, de soportes de medios de producción y de familias de piezas de arte en exposiciones. Con respecto a las familias, se explica las condiciones que tiene que

[2] Owl 2 web ontology language document overview (second edition). `https://www.w3.org/TR/owl2-overview/`. Accedida: 2016-04-06.

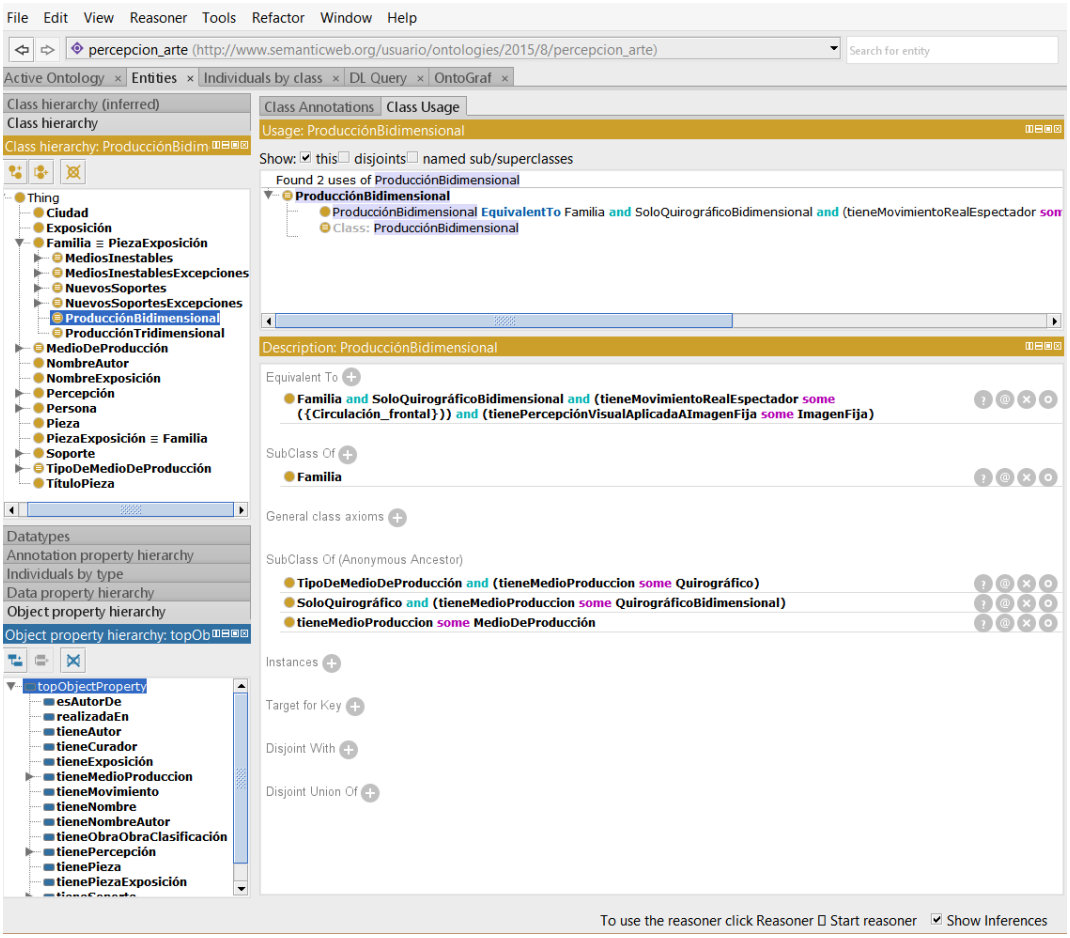

Figura 3.6: Impresión de pantalla de Protégé [1]. A la izquierda de la imagen, en la parte superior, se encuentra la jerarquía de clases implementada. En la parte inferior izquierda se encuentra la jerarquía de propiedades implementada. Se encuentra seleccionado el concepto *ProducciónBidimensional*, en la parte central inferior se muestran las aserciones para dicho concepto y en la parte superior se muestran los usos de la clase en la ontología.

cumplir una pieza en exposición para pertenecer a estas en lenguaje natural y en lógica de descripciones.

En cuanto al proceso de producción se explican las etapas del desarrollo de una ontología según METHONTOLOGY y su aplicación en el desarrollo de la ontología del proyecto. El proceso de especificación del diseño y la implementación de la ontología se desarrolló en el marco de un trabajo interdisciplinario junto con el especialista en el campo del arte Vladimir Muhvich. Este proceso involucró 15 reuniones

con el experto celebradas a lo largo de todo el transcurso del proyecto. Las sucesivas reuniones ayudaron a que se comprendiera la realidad planteada por el experto para lograr conceptualizarla. En estas reuniones se ajustó la representación de elementos existentes de la realidad y se agregaron representaciones de elementos que surgieron como resultado de la retroalimentación con el experto. De esta forma, se logró representar la realidad pretendida por el experto y realizar un diseño que facilitara a los usuarios a utilizar los formularios automáticos.

La ontología OPPA, que resultó del trabajo junto con el experto del dominio, se utilizó como ontología de referencia para la implementación del generador semiautomático de formularios para ingreso de datos en ontologías. Además, los datos ingresados en la ontología se utilizan para generar los grafos de Gephi que el experto utiliza para su investigación. El generador de formularios se describe en el siguiente capítulo.

Capítulo 4

Generación semiautomática de formularios para ingreso de datos en ontologías

En este capítulo se comienza por explicar la motivación de la creación de formularios. Se continúa explicando las estructuras creadas para la construcción de los subformularios que componen el formulario. Se exponen los tipos de subformulario que se pueden generar, así como también sus características. También se mencionan las restricciones que tiene que cumplir una ontología para poder generar a partir de ella un formulario con el generador propuesto.

4.1. ¿Por qué crear una formulario?

En una ontología, para ingresar datos es necesario saber utilizar un editor de ontologías (e.g. Protégé), se debe tener conocimientos de ontologías, y comprender el dominio que se conceptualiza en esta. Para atender la necesidad de que usuarios no familiarizados con ontologías puedan ingresarles datos, se necesita de una interfaz para ingresar datos en un entorno controlado. Estas razones motivaron la creación de un **formulario**, compuesto de subformularios que permiten el ingreso de datos en ontologías.

El formulario se crea con la finalidad de ingresar un individuo nuevo a un **concepto objetivo** previamente definido y relacionarlo con otros de la ontología. La ontología se construye a partir de las relaciones entre un concepto objetivo y otros conceptos de la ontología. El concepto objetivo puede tener relaciones con otros conceptos a través de múltiples propiedades de objeto, que a su vez pueden estar relacionados con otros conceptos. Puede suceder que no todos los conceptos relacio-

nados sean relevantes para el ingreso de datos mediante formularios. A los conceptos relevantes se los denomina **anclas**. Estas anclas definen los formularios a generar, cada uno de estos se construye basándose en la jerarquía de subclases del concepto ancla.

Para definir el concepto objetivo y las anclas es necesario una instancia de configuración previa a la creación de los formularios, por este motivo los formularios son semiautomáticos. Para poder realizar esta instancia es necesario conocer la ontología y la realidad conceptualizada en esta. Entonces existen dos tipos de usuarios, aquellos que solo completan la encuesta, y usuarios administradores que realizan la configuración. En la configuración se debe definir el concepto objetivo, además se deben definir otros conceptos con los que el concepto objetivo se relaciona para que sean incluidos en el formulario generado (e.g., anclas).

En las ontologías se modela una realidad, estas pueden ser modificadas debido a que la realidad que representan cambió o a mejoras en la descripción de la realidad, es importante que estos cambios se vean reflejados en el formulario. Para esto es de utilidad que los formularios se generen a partir de la estructura definida en la ontología en la que están basados en forma automática, sin necesidad de volver a compilar código. Este enfoque permite que se pueda cambiar a una ontología en la que se represente una realidad diferente y, luego de realizar una instancia de configuración para la nueva ontología, el formulario automáticamente refleja dichos cambios.

4.2. Tipos de formulario

Existen dos formas de ingresar datos a formularios, ingresar un valor y seleccionar un valor de un conjunto de opciones. Ingresar un valor significa crear un individuo nuevo o asociar un dato por medio de una propiedad al concepto objetivo. Mientras que, seleccionar un valor se traduce en relacionar individuos ya existentes de una clase al concepto objetivo. Estas dos formas de ingresar datos motivan la creación de dos tipos de formularios, uno para ingresar datos y otro para asociar datos previamente cargados.

Ejemplo selección de un valor. En la Figura 4.1 se muestra un ejemplo de selección de un conjunto de opciones. En el ejemplo se muestra una parte de un formulario múltiple opción creado para la inserción de un individuo de PiezaExposición (i.e., concepto objetivo) en la ontología. El objetivo de la pregunta es relacionar (si corresponde) un individuo de la clase Timbre ya existente, con uno el nuevo individuo de la clase PiezaExposición.

Ejemplo ingreso de un valor. En la Figura 4.2 se muestra un ejemplo de ingreso de un nuevo valor. El primer campo del ejemplo es el nombre de una nueva Pieza asociada al individuo nuevo que se va a ingresar y que pertenecerá a la clase PiezaExposición. El segundo campo también es de ingreso de valores, es un botón para tomar una fotografía de la pieza a ingresar, la fotografía es un valor asociado a la pieza.

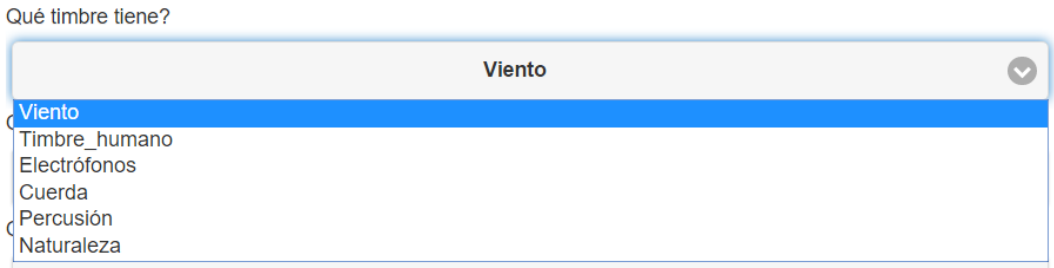

Figura 4.1: Impresión de pantalla del prototipo, ejemplo de selección de un conjunto de opciones. En el ejemplo se muestra una parte de un formulario múltiple opción creado para la inserción de un individuo de PiezaExposición en la ontología. El objetivo de la pregunta es (si corresponde) relacionar un individuo de la clase Timbre ya existente, con uno el nuevo individuo de la clase PiezaExposición.

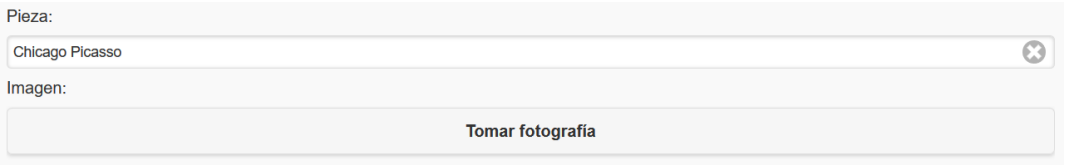

Figura 4.2: Impresión de pantalla del prototipo, ejemplo de ingreso de valores. El primer campo es el nombre de una nueva Pieza asociada al individuo nuevo que se va a ingresar y que pertenecerá a la clase PiezaExposición. El segundo campo también es de ingreso de valores, es un botón para tomar una fotografía de la pieza a ingresar, la fotografía es un valor asociado a la pieza.

Con la información obtenida de la ontología no se puede deducir que tipo de formulario utilizar debido a que depende de la intención del usuario que define el formulario, por lo tanto es necesario que un usuario administrador indique el tipo de formulario necesario en cada caso. Esta distinción se efectúa en la etapa de configuración. En dicha etapa, un administrador, en el momento de seleccionar las anclas, elige el tipo de formulario a generar para cada ancla. Por lo tanto, para el prototipo se definieron dos tipos de formulario diferentes. El primer tipo es un formulario en el que solo se agregan a la ontología nuevas instancias de relaciones de datos y objetos,

se lo llamó formulario múltiple opción. En el segundo tipo además de ingresar nuevas relaciones de datos y objetos, se ingresan nuevos individuos o datos, se lo llamó formulario de ingreso de nuevos datos. El formulario se puede componer de estos dos tipos de subformularios.

En el formulario de múltiple opción se realiza una pregunta construida mediante el uso de un algoritmo que utiliza el contexto establecido en la ontología para obtener los componentes de la frase, el contexto se compone de una clase de la ontología y una propiedad de objeto. Con estos elementos se creó un generador de preguntas que se describe en profundidad en la Sección 4.8. Cada pregunta se realiza acerca de un concepto que se encuentra relacionado mediante una propiedad de objetos con el concepto objetivo del formulario, las opciones presentadas son los individuos que pertenecen al concepto sobre el que se está preguntando.

En el formulario de ingreso de datos, el tipo de datos que se quiera ingresar define el tipo de campo del formulario que se utilizará. En el caso de ingreso de un nuevo individuo el campo que se utilizará será el de ingreso de texto. En este caso en lugar de realizar una pregunta se coloca el nombre del concepto al que se le agregará el nuevo valor.
El caso de las relaciones de datos es diferente debido a que el recorrido de una propiedad de datos nos puede dar mas información acerca del tipo de campo a insertar en el formulario. Los tipos de datos soportados actualmente por el prototipo son: cadena de texto, entero, booleano, fecha y URL. Es posible extender el soporte a todos los tipos de datos implementados en OWL 2.

4.3. Estructura para la construcción de formularios

Los formularios se generan en función de la estructura, los conceptos y las relaciones presentes en la ontología. Para poder generar estos formularios en forma dinámica se accede a la ontología para extraer los datos en los que se basa para generarlos. Se construyó una estructura auxiliar creada a partir de conceptos extraídos de la ontología para la generación de los formularios. Esta estructura permite que se deje de depender de la interfaz por medio de la que se gestiona la ontología (i.e. OWL API [2]), debido a que acceder a la ontología en forma intensiva tiene como consecuencia que la aplicación sea mas lenta al momento de generar los formularios. La estructura no cambia mientras que no se altere la configuración o cambie la ontología en la que se encuentra basada, esto permite que esté disponible cada vez que se crea un formulario y no se necesite acceder a la ontología. Esta representación interna también es la estructura en la que se basa el algoritmo por medio del que se guardan los datos ingresados en la ontología.

En la estructura se representan tres conceptos de la ontología: el de clase, el de propiedad de objeto y el de propiedad de datos. Para la representación de clases, se realizó una correspondencia de la jerarquía de clases a una representación de árbol, cuya raíz es un concepto ancla. Cada nodo del árbol representa una clase de la ontología. A su vez, cada uno de los nodos tiene asociado propiedades de datos, propiedades de objeto, subclases e individuos. Desde cada nodo, se hace referencia a un conjunto de propiedades de objeto y propiedades de dato, de las que la clase (representada por el nodo) es dominio. El conjunto de clases asociadas al nodo son las del primer nivel de la jerarquía de subclases. En caso de que la clase representada tenga individuos, se referencia al conjunto compuesto por estos.
Existe una referencia a la propiedad de objeto que relaciona esta clase con el concepto objetivo de forma directa o indirecta. Si dos clases se encuentran relacionadas por medio de una propiedad de objeto, esta relación se denomina *relación directa* entre estas clases. Además de las relaciones directas, se consideran las clases que se *relacionan indirectamente* con el concepto objetivo. En la Figura 4.3 se muestra un ejemplo de relación indirecta en la ontología de referencia: la clase Autor, la cual se relaciona indirectamente con el concepto objetivo (i.e. PiezaExposición) por intermedio de dos propiedades de objetos, la relación directa entre PiezaExposición con Pieza (i.e. propiedad de objeto *tienePieza*) y la relación entre Pieza con Autor (i.e. propiedad de objeto *tieneAutor*).

Figura 4.3: Ejemplo de relación indirecta. Relación tieneAutor relaciona indirectamente Autor con PiezaExposición.

Las representación de las propiedades de objeto tiene asociadas clases que representan el dominio y el recorrido de la propiedad de objeto en la ontología, también tiene un atributo que indica si la propiedad de objeto representa una relación funcional.

A la representación de las propiedades de datos se le asocia el nombre de la clase dominio y el tipo de dato recorrido de la propiedad. En el prototipo realizado se permite el ingreso de los siguientes tipos de datos: entero, fecha, texto y URL, este

último es utilizado para guardar referencias a imágenes o archivos. Una propiedad de datos puede tener una jerarquía de subpropiedades, se le asocia una correspondencia de dicha jerarquía a la representación de las propiedades de datos.

Combinar las representaciones de clases, propiedades de objeto y propiedades de dato, resulta en que la estructura sea un grafo en donde estas tres representaciones se corresponden con nodos. Es un grafo debido a que dependiendo de la estructura de la ontología, es posible que se generen ciclos. Una forma de generar un ciclo es cuando una clase tiene una relación que es inversa de otra con la clase padre, en este caso en el algoritmo se detecta que la relación es inversa y se crea el ciclo en el grafo. En caso que la relación inversa sea funcional, esta no se agrega al grafo debido a que introduce una inconsistencia en la ontología. Otra forma es que la relación no sea inversa de otra con la clase padre, en este caso se referencia a la clase padre formando el ciclo. En la Figura 4.4 se muestra un ejemplo de formación de un ciclo con relación inversa basado en la ontología de percepciones y medios de producción. En dicho ejemplo se muestra la propiedad de objeto tieneAutor que tiene como dominio a la clase Pieza y como recorrido a la clase Autor. La propiedad de objeto *esAutorDe* es inversa de *tieneAutor*, por lo tanto se forma un ciclo.

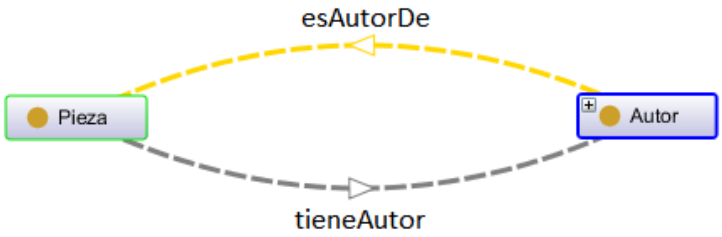

Figura 4.4: Ejemplo de ciclo en la ontología. Relación tieneAutor entre las clases Pieza y Autor es relación inversa de esAutorDe.

Esta estructura se utiliza para crear los formularios web para cada usuario de la aplicación. Crear este grafo permite separar el proceso de generación del formulario de la ontología, disminuyendo la cantidad de accesos a la ontología, ya que se mantiene una estructura en memoria.

Para poder representar el grafo, fue necesario crear tres estructuras (i.e. clases) que son los componentes básicos utilizados para construir el grafo, estas son: clases de la ontología, propiedades de objeto y propiedades de datos. Se agregan los conceptos en sus respectivas estructuras a medida que se encuentren en la ontología en el proceso de recorrida para el armado del grafo. El proceso de construcción del grafo se presenta en la sección 4.4.

4.4. Algoritmo utilizado para la construcción del grafo

Se utilizó un enfoque de jerarquía de clases como base para construir el grafo utilizado luego para la construcción de los formularios. Tomando como punto de partida una de las anclas definidas en la etapa de configuración, se construye el grafo que corresponde a la jerarquía de subclases de dicha ancla. Para la construcción de este grafo, en primer lugar se obtiene la clase ancla de la ontología por intermedio de una interfaz que provee operaciones para el manejo de la ontología. Luego se obtienen las propiedades de objeto de las que esta clase es dominio. También se obtiene la propiedad de objeto que la relaciona directamente con la clase objetivo. A su vez se obtienen las propiedades de datos que tienen como dominio a la clase ancla. Luego se recorren las subclases directas y se realiza el mismo proceso para cada una, al finalizar este proceso se asocia el conjunto de clases a la clase que representa el ancla, este conjunto contiene las subclases de la instancia representada. De esta forma se crea una instancia de representación de una clase de la ontología.

A cada clase de la jerarquía se le crea (si corresponde) el conjunto de propiedades de objeto de la que es dominio, así como también el conjunto de propiedades de datos que cumplen esta condición. Se mantiene dentro de cada clase una referencia a la propiedad de objeto que la relaciona con el concepto objetivo, en caso de no haber propiedad de objeto que la relacione con la propiedad objetivo, se referencia la propiedad de objeto que relaciona a la clase padre con la entidad objetivo. Además, se obtiene de la ontología el conjunto de nombres de individuos que pertenecen a dicha clase y se guarda una referencia a dicho conjunto de nombres.

Para cada clase agregada, se genera un subgrafo por cada propiedad de objeto que tenga a dicha subclase como dominio. En el caso de propiedades de datos, dada una clase en particular de la ontología (que es dominio de una propiedad de datos) se crea una estructura que se corresponde con la jerarquía de sus subpropiedades generando un árbol. También se registra el recorrido de la propiedad de datos, es decir el tipo de dato que se almacenará.

En algunos casos una clase de la jerarquía que parte del ancla no forma parte del recorrido de una propiedad de objeto de la que el concepto objetivo es dominio. Para estos casos se referencia a la propiedad de objeto de la clase padre de esta subclase, en caso de que la clase padre tampoco sea recorrido de una propiedad de objeto que lo relacione con el concepto objetivo se buscará sucesivamente a los padres hasta que se encuentre una propiedad de objeto a la que hacer referencia.

Se identificó otro enfoque para construir los grafos en los que se basan los formularios: basado en una estructura que representa la jerarquía de propiedades de objeto. Esta estructura es la base para la creación y posterior guardado de los datos de la ontología en lugar de la jerarquía de clases (que es lo que se propone en el proyecto). En este caso las clases están asociadas a las propiedades. En un enfoque basado en la jerarquía de propiedades de objeto se registran todas las propiedades de objeto que sean subclase de la propiedad de objeto que relaciona la clase objetivo con la clase ancla del formulario. Por otro lado, la especificación del recorrido de una propiedad de objetos puede no corresponder con con una clase de la jerarquía de subclases seleccionada, por lo tanto podría suceder que se agreguen al grafo clases que se convertirán en campos del formulario que podrían no ser de interés para el usuario final o el administrador.

4.5. Algoritmos para generación de formularios

En esta sección se explican los tipos de algoritmo utilizados para generar los formularios de la aplicación web. Se presentan dos tipos de algoritmos, uno para la representación de encuestas múltiple opción y otro para ingreso de datos. Para los dos tipos de algoritmo se utiliza el grafo descripto en la Sección 4.3 en la generación de los formularios.

Generación de formularios múltiple opción

Se diseñó un algoritmo para la generación de formularios de tipo múltiple opción. En este algoritmo, se recorre la jerarquía de clases en el grafo a partir del nodo ancla, dicha recorrida se hace en profundidad. Si bien toda la estructura representa un grafo, la jerarquía de clases corresponde a una estructura de árbol. Se recorre cada rama del árbol de la jerarquía de clases hasta las hojas, al llegar a las hojas se genera la pregunta. Las opciones de la pregunta que se muestran son los individuos que pertenecen a la hoja.

Ejemplo de formulario múltiple opción. En la Figura 4.5 se muestra un ejemplo de jerarquía de clases. Los puntos son clases de la ontología, mientras que los rombos son individuos. Se muestra la clase Timbre, que es subclase de Auditiva, que a su vez es subclase de Percepción. Los individuos de la figura pertenecen a la clase Timbre. Por otro lado, en la Figura 4.6 se puede observar una captura de pantalla de la encuesta múltiple opción generada a partir de esta jerarquía, con PiezaExposición como concepto objetivo.

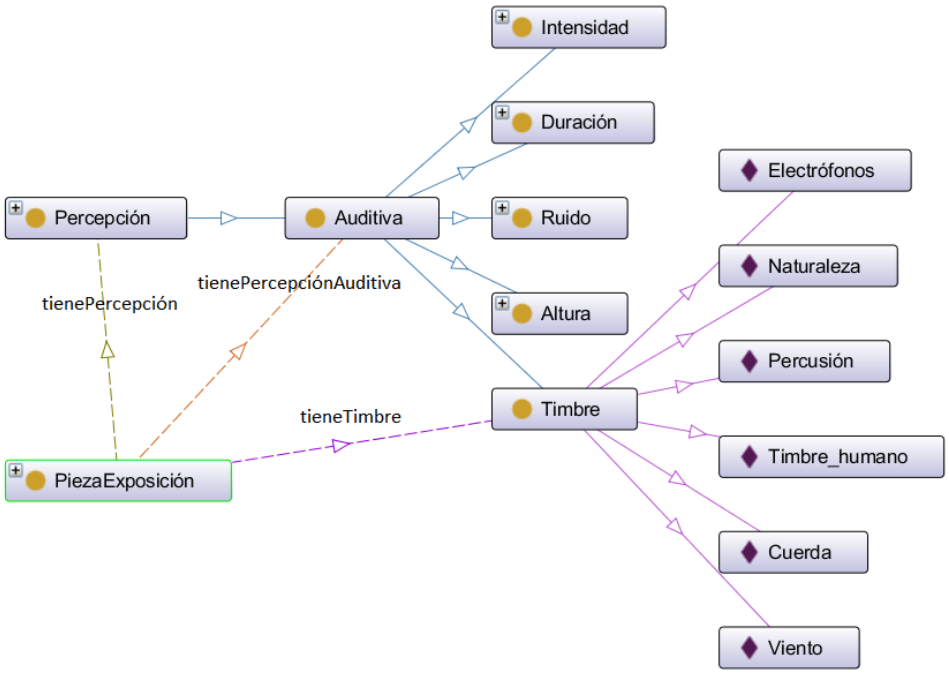

Figura 4.5: Ejemplo de jerarquía de clases y los individuos correspondientes.

Las hojas de la jerarquía pueden ser dominio de propiedades de objeto o de datos que las relacionen con otras clases o tipos de datos. En el formulario de múltiple opción solo se crean relaciones entre la entidad objetivo y los individuos que pertenecen a las hojas, es decir que no se insertan nuevos individuos en las hojas. Por lo tanto, las relaciones entre los individuos de las hojas con individuos de otras clases no se alteran, entonces no es necesario incluirlas en el formulario.

La recorrida en profundidad también se utiliza con la finalidad de establecer el contexto en el que se realizan las preguntas. En la Figura 4.6 se muestra un ejemplo de esta situación. Para representar la relación *es subclase de* se asocian todas las subclases directas a un panel, en el ejemplo de formulario web se puede observar que Timbre se encuentra dentro de un panel de nombre Auditiva, esto es debido a que, según se puede observar en la Figura 4.5, la clase Timbre es subclase de Auditiva. Las clases de mayor jerarquía son representadas como el título del formulario. En el ejemplo, la clase de mayor jerarquía se llama Percepción, se puede observar que es la

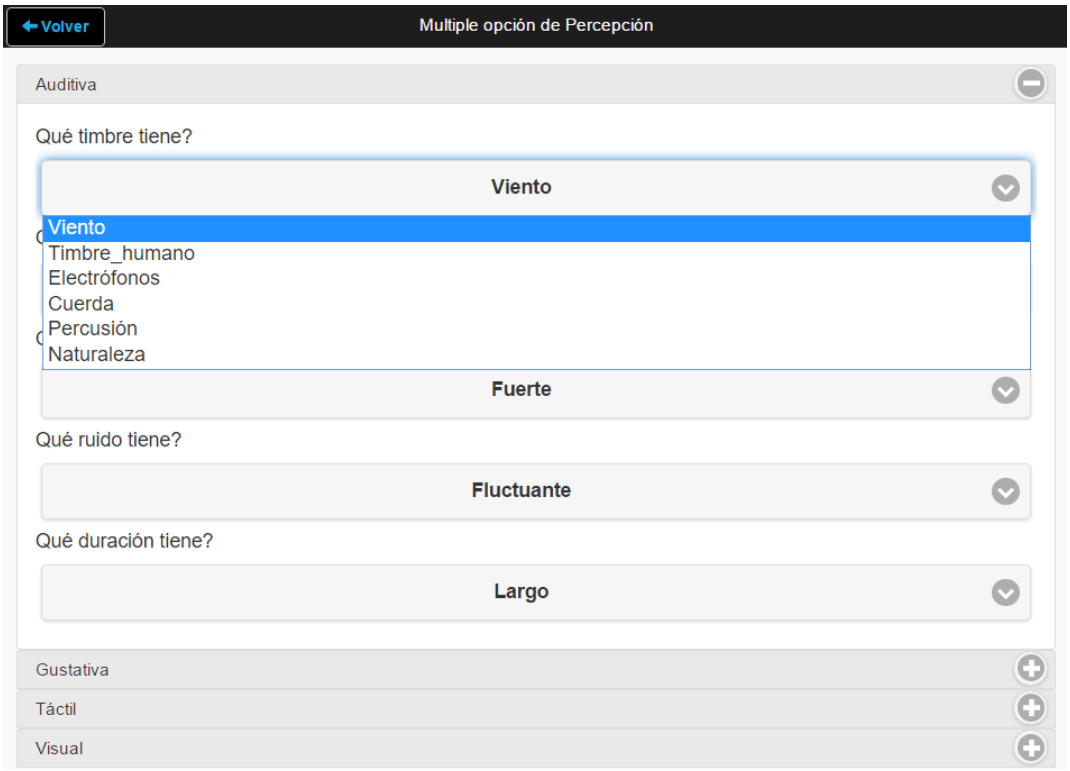

Figura 4.6: Captura de pantalla de encuesta múltiple opción.

misma que el título del formulario. En el ejemplo entonces se ayuda a establecer que la pregunta acerca del tipo de timbre se realiza en el contexto de la percepción auditiva.

Dependiendo de si la propiedad que relaciona la clase objetivo con la clase que tiene los individuos a seleccionar es funcional o no, se permite la selección de un individuo o de varios. En la Figura 4.5 se muestra un ejemplo de este caso. La clase objetivo es la clase PiezaExposición la cual tiene definida una propiedad de objeto llamada tieneTimbre, si dicha propiedad es funcional, en pantalla se mostrará un combo box de selección simple, en cambio si esta propiedad de objeto no es funcional se desplegará en pantalla un combo box de selección múltiple. En ambos casos estos tendrán cargados los individuos que se encuentran en la clase Timbre (i.e. Electrófonos, Naturaleza, Percusión, Timbre Humano, Cuerda y Viento).

Generación Formulario de ingreso de datos

En el formulario de ingreso de datos, se agregan nuevos individuos y se definen valores de las propiedades de datos que poseen los nuevos individuos. El algoritmo

diseñado consiste en realizar una recorrida en profundidad de la jerarquía de clases generada a partir del ancla. Cuando se llega a un nodo hoja del grafo (i.e. un nodo clase del grafo), se crea un campo de tipo ingreso de texto que luego se mostrará en pantalla. El valor ingresado en este campo corresponde al nombre de un nuevo individuo que pertenecerá a la clase representada por el nodo hoja.
Para el caso de ingreso de datos para las propiedades de datos que tienen como dominio la clase de la ontología, el recorrido de este tipo de propiedades indica el tipo de valor que se debe asociar a esta propiedad. Se utiliza el recorrido definido para saber de que tipo es el campo a generar. Para los tipos de datos texto y entero se utiliza un tipo de campo de entrada de texto, luego si la propiedad de dato indica que este campo es entero, se controla que el valor ingresado sea efectivamente un entero. Para el caso de ingreso de una fecha, se utiliza un componente de tipo calendario, para facilitarle al usuario la selección. Luego, para el tipo booleano se permitirá elegir entre los valores *SI* y *NO*. El último valor diferenciado es una URL (que en la ontología toma el nombre de AnyURI), para este caso en pantalla se despliega un botón que redirige a una página especial que conecta con la cámara del dispositivo utilizado para poder sacar una foto, la foto se almacena en formato PNG en una URL accesible desde fuera de la aplicación. Luego de tomar la fotografía se vuelve al formulario principal guardando la URL de la imagen como valor de la propiedad de datos.

Ejemplo de formulario de ingreso de datos. En la Figura 4.7 se muestra una captura de pantalla de un formulario de ingreso de datos. El ejemplo se generó por medio de la ontología OPPA. En la etapa de configuración (llevada a cabo previo a la generación del formulario) se definió como concepto objetivo a *PiezaExposición* y como una de las anclas de ingreso de datos al concepto *Pieza* que se encuentra relacionado directamente con el concepto objetivo.
En orden de arriba hacia abajo se puede observar que el primer campo que aparece es Pieza, en este campo se debe ingresar el nombre de la instancia de dicha clase, en este caso el nombre de la pieza de arte es *Chicago Picasso*. Se insertó un campo para la clase con la que Pieza se encuentra relacionada: Autor. En este caso el autor es Pablo Picasso. Los atributos de instancia son: Imagen, Dimensión y Fecha creación. Los valores de dimensión se encuentran ocultos, pero son el largo, ancho y alto de la pieza. La fecha de creación ingresada es 1 de enero de 1967. La clase Autor a su vez se encuentra relacionada con otra clase: Pieza. Por lo tanto, se presenta en este caso un ciclo. El ciclo, en este caso, es una relación inversa no funcional entre Autor y Pieza, se pregunta por otras piezas de arte de la autoría del artista ingresado como autor.

Figura 4.7: Ejemplo de formulario de ingreso de datos. Ingreso de datos de una Pieza. En los datos del autor se pueden elegir otras piezas de su autoría, es un ejemplo de ciclo.

4.6. Guardado de datos de los formularios en la ontología

En el proceso de guardado, las operaciones que se manejan son las de creación de nuevos individuos, inserción de nuevos individuos recientemente creados en las clases que corresponden, y el relacionamiento entre individuos según relaciones ya existentes en la ontología. No existen instancias en las que se cambia la estructura de la ontología, es decir que implican creación, modificación o eliminación de clases o de propiedades de objetos o de datos.
El primer paso del proceso de guardado de datos en la ontología es la creación de un nuevo individuo y asociar dicho individuo a la clase objetivo. Se guardan además otros individuos e instancias de relaciones de objetos y de datos.
El algoritmo de guardado es diferente dependiendo del tipo de formulario a guardar. Para los formularios de múltiple opción, no se insertarán nuevos individuos, únicamente se crearán relaciones entre individuos ya existentes dentro de la ontología. Las instancias de relaciones que se ingresan son únicamente de propiedades de objeto que tienen como dominio al concepto objetivo.
En el caso de los formularios de ingreso de datos, se diferencian tres tipos de situaciones: (i) ingreso de un nuevo individuo, (ii) ingreso de un valor de una propiedad

de datos, (iii) relaciones que forman ciclos. En (i), se comienza creando una nueva instancia del individuo a ingresar, pero en caso de que el individuo ya exista se obtiene dicha instancia. El individuo se crea con la finalidad de relacionarse con un individuo del concepto objetivo, por lo tanto se crea una instancia de la propiedad de objeto que los relaciona.
En (ii), se ingresa un valor de una propiedad de datos, se crea directamente la instancia de la propiedad de datos para el concepto objetivo. En las ontologías se diferencia entre datos e individuos, en el caso de los datos, no es necesario crear una instancia de estos antes de crear una instancia de una relación.
En (iii), los ciclos se crean con individuos ya existentes en la ontología. Se crea una instancia de la relación entre el individuo perteneciente el concepto objetivo y el individuo seleccionado de la clase en la que se genera el ciclo. No se crean nuevos individuos.
En las tres situaciones anteriores se explicó el guardado de las clases de la ontología que se relacionan en forma *directa* con el concepto objetivo. Para el caso de las relaciones indirectas, el proceso es el mismo pero tomando como concepto objetivo al dominio del último tramo de la relación directa (definición de relación directa e indirecta en Sección 4.3).

4.7. Restricciones

En la siguiente sección se describen las condiciones que tiene que cumplir una ontología para que, a partir de esta, se puedan generar formularios con el algoritmo planteado.

Las propiedades de objeto que relacionan a una clase principal de una jerarquía directamente o indirectamente con la clase objetivo, deben tener el dominio y el recorrido definido para ser elegibles como anclas. Se explicará esta restricción con un ejemplo. En la Figura 4.8 se muestra la parte de la jerarquía de Percepción que describe la percepción gustativa y su relación (mediante propiedades de objeto) con el concepto objetivo PiezaExposición. La relación de jerarquía se define con flechas celestes y las relaciones de objeto con flechas de línea punteada. En el caso de las propiedades de objeto se puede observar que PiezaExposición presenta relación con todas las clases de la porción de la jerarquía representada excepto con Amargo. Tal como se explicó en la Sección 4.4, no es necesario que esta relación se encuentre definida para que se realice la pregunta sobre Amargo. La razón es que existe una relación entre PiezaExposición y Gustativa. Dicha relación se utilizará para generar la pregunta, y se guardará en la ontología como una instancia de esta propiedad de objeto a la relación entre el individuo de PiezaExposición a ingresar y un individuo de Amargo.

Para el caso de la relación entre PiezaExposición y Percepción (i.e., tienePercepción), como Percepción no tiene padre en la jerarquía de clases, dicha propiedad de objeto debe tener definido el dominio (i.e., PiezaExposición) y su recorrido (i.e., Percepción), de forma que se tome en cuenta la relación con esta clase y con toda su jerarquía por el algoritmo de construcción del grafo.

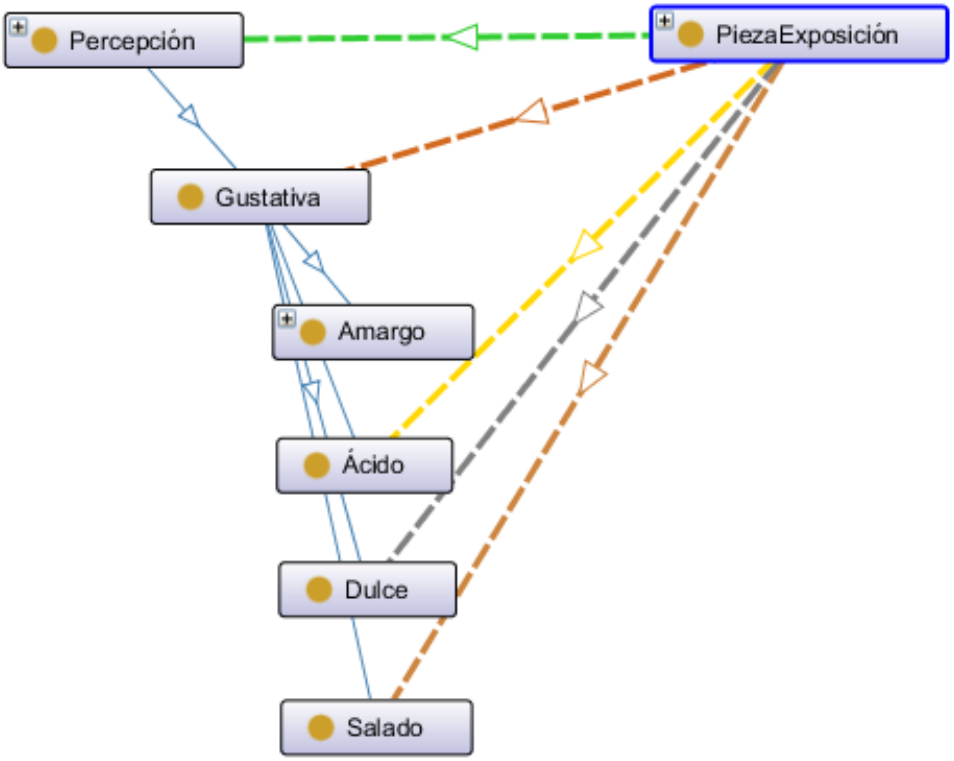

Figura 4.8: Diagrama de relaciónes de objeto entre PiezaExposición y la jerarquía de Percepción gustativa.

Como restricción a las propiedades de datos, estas son tenidas en cuenta si tienen un dominio definido y además alguna clase de ese dominio es parte del grafo (i.e.: estructura para la construcción de formularios). Si bien no es obligatorio que el recorrido esté definido, la definición del mismo en una propiedad de datos restringe los posibles tipos de datos a ingresar, ya que el recorrido es utilizado para saber de que tipo debe ser un campo para un a propiedad de datos, esto también permite implementar controles la momento de ingresar datos en estos campos. Por ejemplo si el recorrido de una propiedad de dato es entero y se ingresa un valor en el campo

que no es numérico se le indicará al usuario dicho error. Si no se define recorrido no se hará ningún control sobre el valor ingresado en el campo.

Los formularios pueden ser muy extensos, por tanto es importante que se formulen la menor cantidad de preguntas posible. Por esta razón se decidió formular las preguntas en las hojas de las jerarquías recorridas, por este motivo para el formulario de múltiple opción solo se mostrarán como opciones los individuos que se encuentren en dichas clases.

4.8. Generación de preguntas

Cada campo del formulario tiene una etiqueta asociada que indica qué dato se pretende que sea ingresado (en caso de que corresponda ingresarlo). En el caso de la parte del formulario que corresponde a los formularios de ingreso de datos, el valor de las etiquetas de los ingresos de nuevos individuos es el nombre de la clase del individuo a ingresar. El caso de ingreso de datos para propiedades de datos (i.e.: atributos de instancia), el valor de la etiqueta es el del nombre de la propiedad de datos.
En el caso de los campos de múltiple opción, la etiqueta asociada a dicho campo tiene como valor una pregunta. La forma de generación de preguntas se basa en la propuesta de Papasalouros (2008) [26]. La pregunta se genera en base al nombre de la propiedad de objeto que lo relaciona con la entidad objetivo y el nombre de la clase de las instancias que se presentan como opciones. Si el nombre de la clase se encuentra incluido dentro del nombre de la propiedad de objeto, este se ignora (e.g.: si la propiedad de objeto es *tieneAltura* y la clase es *Altura*, entonces la pregunta es *Qué altura tiene?*). La generación de preguntas para los ciclos con relaciones inversas es diferente, se utiliza el hecho de que se conoce para agregar la palabra *otra* (e.g.: con la propiedad *esAutorDe* y con *Pieza* como recorrido se genera la pregunta *De cuál otro/a pieza es autor?*).
Para el caso de la ontología de percepciones y medios de producción, la forma del nombre de las relaciones se adapta al caso general de pregunta propuesta. Pero no hay certeza de que el método para generar preguntas funcione correctamente en otras ontologías.

4.9. Modelo de dominio del grafo

En esta sección se explica el modelo del dominio del grafo, se puede observar el modelo en la Figura 4.9. Se realizan correspondencias entre componentes de la ontología y clases: un concepto de la ontología se corresponde con una instancia de

la clase de nombre *OntologyClass*, una propiedad de objetos de la ontología se corresponde con una instancia de la clase *OntologyObjectProperty* y una propiedad de datos se corresponde con una instancia de la clase *OntologyDataProperty*.

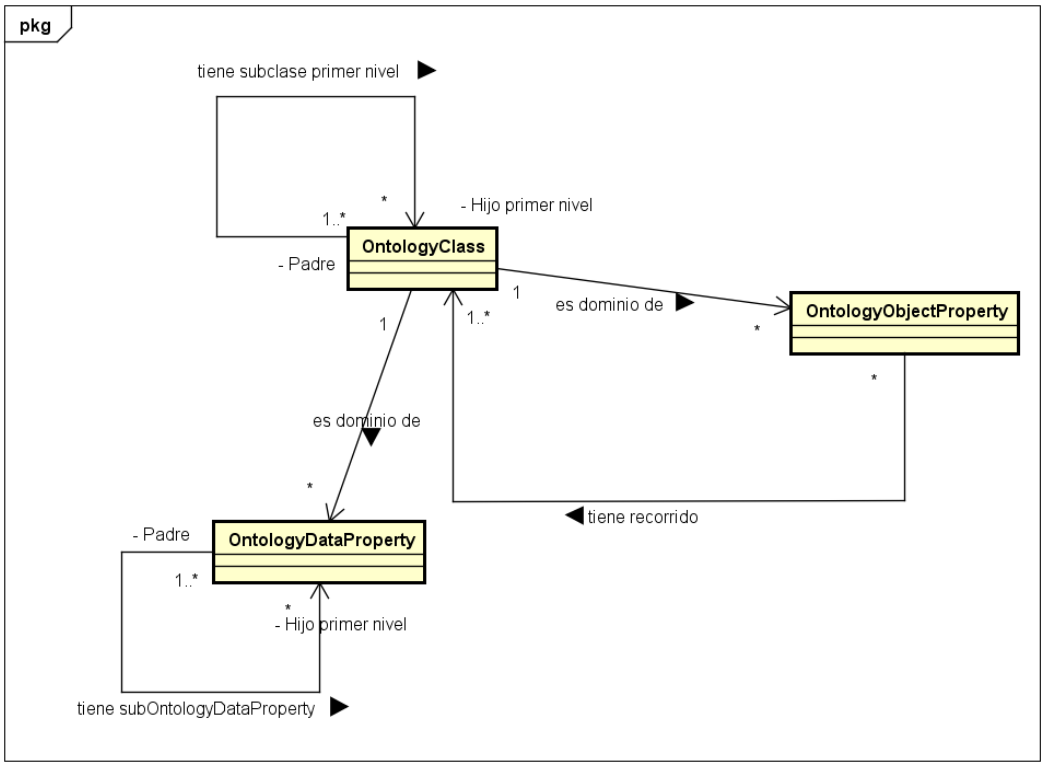

Figura 4.9: Modelo de dominio del grafo base para la creación de formularios.

Se realizó una correspondencia de un subconjunto de las relaciones que tienen los componentes de la ontología a relaciones entre las clases del modelo de dominio. El subconjunto de relaciones se encuentra formado por relaciones relevantes para el proceso de construcción de los formularios. A continuación se explican las relaciones entre las clases y su correspondencia con relaciones entre componentes de la ontología.
Un concepto en la ontología puede pertenecer al dominio de una o mas propiedades de objetos, en el modelo de dominio esto se representa por medio de la relación *es dominio de* entre una instancia de la clase OntologyClass y cero o mas instancias de la clase OntologyProperty. Este escenario es análogo en la ontología para el caso de la relación entre un concepto y una o varias propiedades de datos, por este motivo se definió una relación del mismo nombre entre OntologyClass y OntologyObjectProperty en el modelo de dominio.

En la ontología se representan jerarquías de conceptos mediante la relación *subclase de*, en el modelo esta relación se corresponde con la relación *tiene subclase de primer nivel* entre una instancia de OntologyClass y cero o mas instancias de OntologyClass. En forma análoga se representa la correspondencia de la jerarquía de propiedades de datos, en este caso por medio de la relación *tiene subOntologyDataProperty* entre una instancia de OntologyObjectProperty y cero o mas instancias de la misma clase. Una propiedad de objetos en una ontología puede tener definido como recorrido a uno o varios conceptos, en el modelo esta se corresponde con la relación *tiene recorrido* entre una instancia de OntologyObjectProperty y una o mas instancias de OntologyClass.

4.10. Conclusiones

Este capítulo presenta el diseño, la implementación y los resultados obtenidos del generador de formularios automáticos. Se comienza explicando la necesidad de utilizar un formulario como método de ingreso de datos a ontologías. Se explican los dos tipos de subformularios que constituyen el formulario, que son: formularios de múltiple opción y formulario de ingreso de datos. Asimismo, se explican las estructuras y algoritmos definidos para la construcción de los formularios y el proceso de guardado de los datos ingresados en una ontología. Se mencionan las restricciones que tiene que cumplir una ontología para lograr crear formularios a partir de las estructuras y algoritmos presentados. Se comenta acerca de la estrategia seguida para generar las preguntas correspondientes al formulario de múltiple opción y también al caso de ciclo con relación inversa. Finalmente, se muestra el modelo de dominio utilizado para representar el grafo que sirva como base para la generación de los formularios.

En la versión actual fue contemplada la posibilidad de configurar únicamente formularios de ingreso de datos y de múltiple opción. Es posible que el usuario administrador, al realizar una configuración, quiera indicar que al generar un formulario para una jerarquía de clases que parte de una ancla determinada, una parte del formulario sea de múltiple opción, mientras que otra parte sea de ingreso de datos. En el caso planteado anteriormente corresponde la generación de un formulario mixto.
A continuación se presenta una posible solución a esta situación, planteada desde el punto de vista de extender la aplicación. Se plantea que para generar este tipo de formularios, en la configuración se debería agregar un paso extra en el que se indique, para cada ancla de un conjunto de anclas, cuales clases de la jerarquía a partir del ancla serán un formulario de múltiple opción o de ingreso de datos. En la versión actual del prototipo, solo se indica el tipo de formulario a generar sobre la jerarquía completa de las anclas y no sobre las sub clases de su jerarquía.
Una forma de implementar este cambio sería que en la etapa de configuración, se

agregue un paso extra, en el que para cada ancla se muestre la estructura de la jerarquía de sus subclases, y permita que se seleccione para que sectores de la jerarquía se generará formulario de ingreso de datos y para que sectores se generará de múltiple opción.
Otra forma implementar la selección de los conceptos se basa en el hecho de que las clases que tendrán los valores tanto a ser seleccionados como para ingresar nuevos datos, son las hojas de la jerarquía a partir del ancla, por lo que se podría listar estas hojas de forma que el usuario decida si son de múltiple opción o ingreso de datos.

En el siguiente capítulo se presenta el prototipo de generador semiautomático de formularios en base a ontologías.

Capítulo 5

Prototipo: generador semiautomático de formularios en base a ontologías

En este capítulo se describe el prototipo de generador automático de formularios implementado. Se comienza con una visión general de la aplicación y luego se describen características específicas de su funcionalidad, la arquitectura y las pruebas realizadas.

5.1. Descripción general

Se desarrolló un prototipo para poner en práctica el funcionamiento del generador de formularios. Este se realizó como una aplicación web para atender el requerimiento de que la aplicación pudiera ser accedida tanto desde una pc de escritorio como desde un dispositivo móvil. Siendo una aplicación web alcanza con que el dispositivo cuente con una conexión a internet y tener instalado un cliente web (e.g., Firefox). La aplicación está pensada para ser usada por el experto y por un grupo reducido de usuarios designados.

El prototipo tiene dos funcionalidades básicas, la creación de la configuración y realizar el formulario. En la configuración se provee una interfaz para que en forma guiada se logre la definición de los parámetros necesarios para la generación de los formularios. Realizar el formulario comprende la generación de este en forma automática basado en una configuración definida y el posterior guardado de los datos ingresados en la ontología.
Además de las funcionalidades básicas, se implementaron otras funcionalidades que involucran el manejo de usuarios y distintas formas de visualización de datos de la ontología. Entre las formas de visualización de datos, se encuentra la representación

de un resumen de los datos ingresados en un tabla, y como requerimiento específico del cliente la generación de un grafo para ser visualizado en la plataforma Gephi [1].

Se definieron dos tipos de usuario de la aplicación, por un lado los que crean configuraciones y por otro lado usuarios (i.e. administradores), y por otro lado aquellos usuarios que solo utilizan la aplicación para completar los formularios (i.e. no administradores). Los tipos de usuario se diferencian mediante roles, estos se asignan a usuarios y definen las funcionalidades a las que tienen acceso. En principio se definieron dos roles: administrador y no administrador. Es posible crear mas roles en caso de ser necesario.

5.2. Descripción de la arquitectura

En esta sección se describe la arquitectura utilizada para la construcción del prototipo. Para ello se tomó como referencia el modelo *4+1* introducido por el profesor Philippe Kruchten [22]. Se presenta una simplificación de la vista de casos de uso, la vista lógica y la vista física (de despliegue).

Vista de casos de uso

En esta sección se presenta el conjunto de casos de uso más importantes, según (Krutchen 1995) [22] estos son aquellos que: son la razón de ser del sistema, o los que se utilizan mas frecuentemente, o los que implican un riesgo. Dentro de la categoría de casos importantes se distinguen dos tipos de casos, casos generales y casos exclusivos. Estos últimos fueron pedidos especialmente por el experto con el que se trabajó.

Casos de uso generales

El la figura 5.1 se muestra la relación que tienen los distintos tipos de usuarios con los casos importantes. El prototipo desarrollado soporta dos tipos de usuarios. Un tipo de usuario es el administrador, que tendrá la posibilidad de crear y activar configuraciones de formularios, administrar usuarios del sistema y completar los formularios. El otro tipo de usuario es el no administrador, que solo tendrá la posibilidad de realizar la encuesta, ver sus respuestas y además editar sus datos de usuario. Por petición del cliente se configuró un perfil de usuario no administrador de nombre *Investigador*, el nombre del perfil puede ser cambiado para alinearse a otra realidad. También es posible la configuración de otros perfiles para roles administrador y no administrador.

[1] Gephi - the open graph viz platform. https://gephi.org/. Accedida: 2016- 04-14.

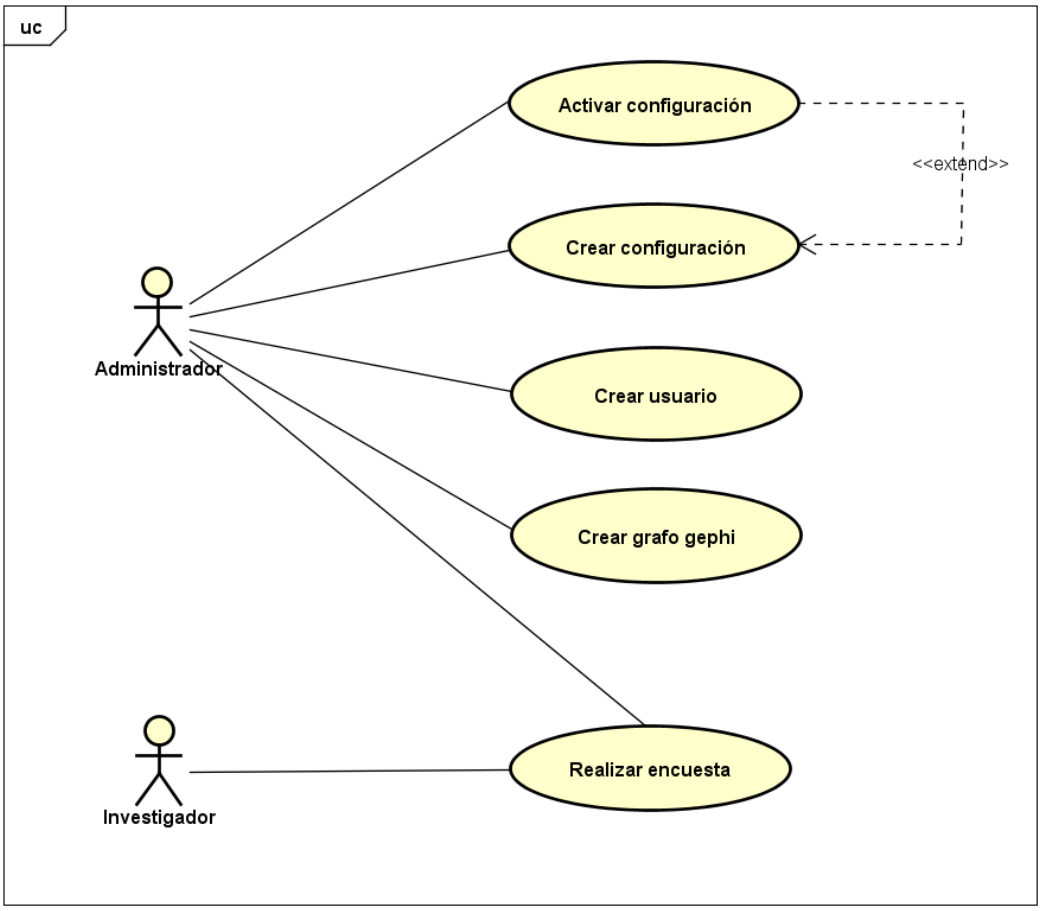

Figura 5.1: Casos de usos importantes.

Crear configuración. En este caso se define la información que se preguntará en los formularios a realizar por los usuarios no administradores. Crear configuración pertenece al conjunto de los casos que son la razón de ser de la aplicación. Cada configuración se aplica a un perfil de usuario (e.g. investigador). En esta funcionalidad se debe indicar cual es el concepto objetivo y las anclas (los términos concepto objetivo y ancla se encuentran definidos en la Sección 4.1 perteneciente al capítulo de generación de formularios). Un ancla es la clase base sobre la que se genera un formulario, se debe indicar si el formulario a generar es de tipo ingreso de datos o de múltiple opción.

Las configuraciones tienen un estado, este puede ser *activa* o *inactiva*. Se generan formularios a partir de la configuración activa de un perfil. Se permite crear varias configuraciones para cada perfil, pero solo una configuración puede estar activa para

un perfil determinado. Al confirmar la nueva configuración el sistema sugiere crearla en estado de configuración activa. En caso de crear la configuración en estado activa, si existe configuración activa para el perfil elegido, se cambia el estado de esta última a inactiva.

Activar configuración. Para un usuario con perfil perteneciente al rol administrador, se presentan todas las configuraciones que existen para todos los perfiles del sistema. Se puede filtrar el listado de las configuraciones por su nombre, por su identificador o por el rol para el cual fueron creadas. Dada una configuración de las presentes en el listado, el usuario le cambia el estado a activa. En caso de que ya existiese otra configuración activa asociada al mismo perfil de la configuración que se está activando, a esta se le cambiará el estado a inactiva.
Luego de activar la configuración se genera un nuevo grafo en el que se apoya el algoritmo generador de formularios. El grafo de formulario generado es el presentado en la Sección 4.3 perteneciente al capítulo en el que se describe el formulario automático (Capítulo 4).
En caso de que un usuario esté realizando una encuesta cuando el administrador decide activar otra configuración para su perfil, se mantendrá el formulario generado correspondiente a la configuración anterior hasta que cierre su sesión. Al volver a iniciar sesión formulario el usuario podrá completar el formulario generado a partir de la nueva configuración activa.

Crear un usuario. Solo usuarios con perfil perteneciente al rol administrador podrán gestionar los usuarios, en particular serán los encargados de crear nuevos usuarios. El usuario administrador se encuentra habilitado para crear usuarios de cualquier perfil perteneciente al rol administrador o no administrador. El caso consiste en ingresar los datos del nuevo usuario a crear, estos datos son: nombre, apellido, correo electrónico, fecha de nacimiento, sexo y perfil (e.g. investigador). Luego de ingresar esos datos se presiona el botón crear usuario. El correo electrónico es el identificador de usuario. Luego de creada la cuenta de usuario, se notificará al propietario enviando un correo electrónico a la casilla ingresada indicando cual es su contraseña para acceder a la aplicación web.

Completar un formulario. La funcionalidad de completar el formulario es la razón de existencia del prototipo. Por medio del completado de formularios se permite que los usuarios agreguen contenido a una ontología en forma controlada.
Este caso de uso puede ser ejecutado por usuarios con rol perteneciente al perfil administrador o no administrador. Tiene como precondición que debe existir una configuración activa asociada al rol del usuario que dispara el caso de uso. El usuario comienza el caso de uso presionando el botón ubicado en el panel lateral de nombre *Gephi*. Al momento de acceder se crea un nuevo formulario en base a la configuración activa de su perfil y se inicia con todos los campos del formulario vacío. Se accede a

una pantalla principal en la que se muestran los temas principales a tratar (i.e. las anclas) mediante una botonera. Al momento de comenzar a completar el formulario los botones tienen color azul, al completar una etapa, el botón correspondiente a dicha etapa cambia a color verde, lo cual significa que los datos ingresados para esa etapa están en condiciones de ser ingresados a la ontología. Si el usuario desea cambiar los datos luego de haber confirmado un subformulario, puede hacerlo. Sólo cuando se hayan confirmado todos los temas principales (cuando todos los botones se encuentran en color verde) se habilita al usuario a confirmar el formulario completo. Al confirmar el formulario, los datos ingresados se guardan en la ontología, es decir que se crea una instancia de individuo objetivo y sus relaciones. Luego de guardar, se borran todos los datos ingresados en el formulario, además los botones vuelven a color azul, el usuario queda habilitado a completar un formulario para ingresar otro individuo. Si el usuario se encuentra en un subformulario y presiona el botón de volver a la pantalla principal del formulario, al volver tendrá los datos previamente ingresados. Si el usuario presiona el botón *salir* que se encuentra en la pantalla principal del formulario, se borrarán todos los datos ingresados y al volver a entrar tendrá que completarlos nuevamente.

Casos de uso excepcionales

En esta sección se presentan los casos de uso pedidos como requerimientos especiales del cliente.

Crear un grafo de gephi. Si bien es una funcionalidad excepcional, el cliente enfatizó que el uso frecuente que le dará a dicha funcionalidad debido a la importancia de los grafos generados para el trabajo que el desarrolla. Por lo tanto es de especial interés que la funcionalidad exista.
La funcionalidad puede ser accedida solo por usuarios con perfil perteneciente al rol administrador. Se despliega una lista de exposiciones en una tabla. Se listan las exposiciones que tienen piezas ingresadas. En la tabla, cada exposición listada tiene asociada un botón, dicho botón dispara la generación del grafo de gephi para la encuesta a la que se encuentra asociado. La generación del grafo comprende que se muestren las relaciones de cada pieza perteneciente a la exposición seleccionada con el resto de los conceptos de la ontología, además se debe mostrar la familia a la que pertenece. Luego de presionar el botón de generación de encuesta, el botón desaparece y aparece un botón para descargar el archivo creado. Al presionar el botón se descarga un archivo de extensión *gexf* al dispositivo del usuario.

Vista lógica

Se presenta el estilo arquitectónico elegido para cumplir con los requerimientos funcionales, profundizando luego en cada capa de la arquitectura.

Estilo arquitectónico

Para el diseño arquitectónico del prototipo se utilizó un modelo de capas jerárquicas. Cada capa provee servicios a las capas superiores y requiere de servicios de las inferiores, a su vez utiliza un modelo estricto, por lo que sólo se consumen servicios de capas inmediatamente inferiores. En la figura 5.2 se muestra un diagrama UML de componentes, en el diagrama se describe el estilo arquitectónico utilizado y las capas que lo constituyen.

La capa de presentación es la encargada de proveer una vista adecuada dependiendo de si el usuario que se conecta a la aplicación tiene un perfil con rol administrador o no administrador. Por este motivo esta capa se constituye de dos componentes, en el componente Front-End se encapsula la presentación para acceder a funcionalidades de no administrador, y Back-Office donde se encapsula la presentación para acceder a funcionalidades de administrador. En la a figura 5.3 se muestra la conformación del subsistema que representa a la capa de presentación. En esta capa se consumen servicios provenientes de la capa lógica únicamente.

La siguiente es la capa lógica, en esta capa se implementa la lógica de negocio de la aplicación, proveyendo interfaces a la capa de presentación. En la capa lógica se implementa la gestión de usuarios y de las configuraciones. También se encuentra el componente encargado realizar operaciones sobre datos provenientes de la ontología, el cual es llamado *Manejador de Ontología*. Por último se encuentra el componente encargado de la generación de grafos para la aplicación gephi, llamado *Generador Gephi*. Este componente consume servicios que provee el *Manejador de Ontología* para obtener datos de la ontología y así poder generar el grafo para gephi, esto se visualiza en la figura 5.4, además en esta figura se muestra como se constituye la capa.

La última capa es la de datos, en donde se gestiona la comunicación con la base de datos relacional y con la ontología. En esta capa se encuentran los componentes llamados *Usuario Datos* y *Configuración Datos*, estos se comunican con la base de datos relacional para realizar las operaciones solicitadas por la capa lógica sobre las tablas correspondientes a usuarios y configuraciones. Otro componente de la capa de datos es el *Comunicador Ontología*, en donde se implementan las operaciones consumidas desde la capa lógica en lo referente a datos de la ontología. En la figura 5.5 se muestra la distribución del subsistema.

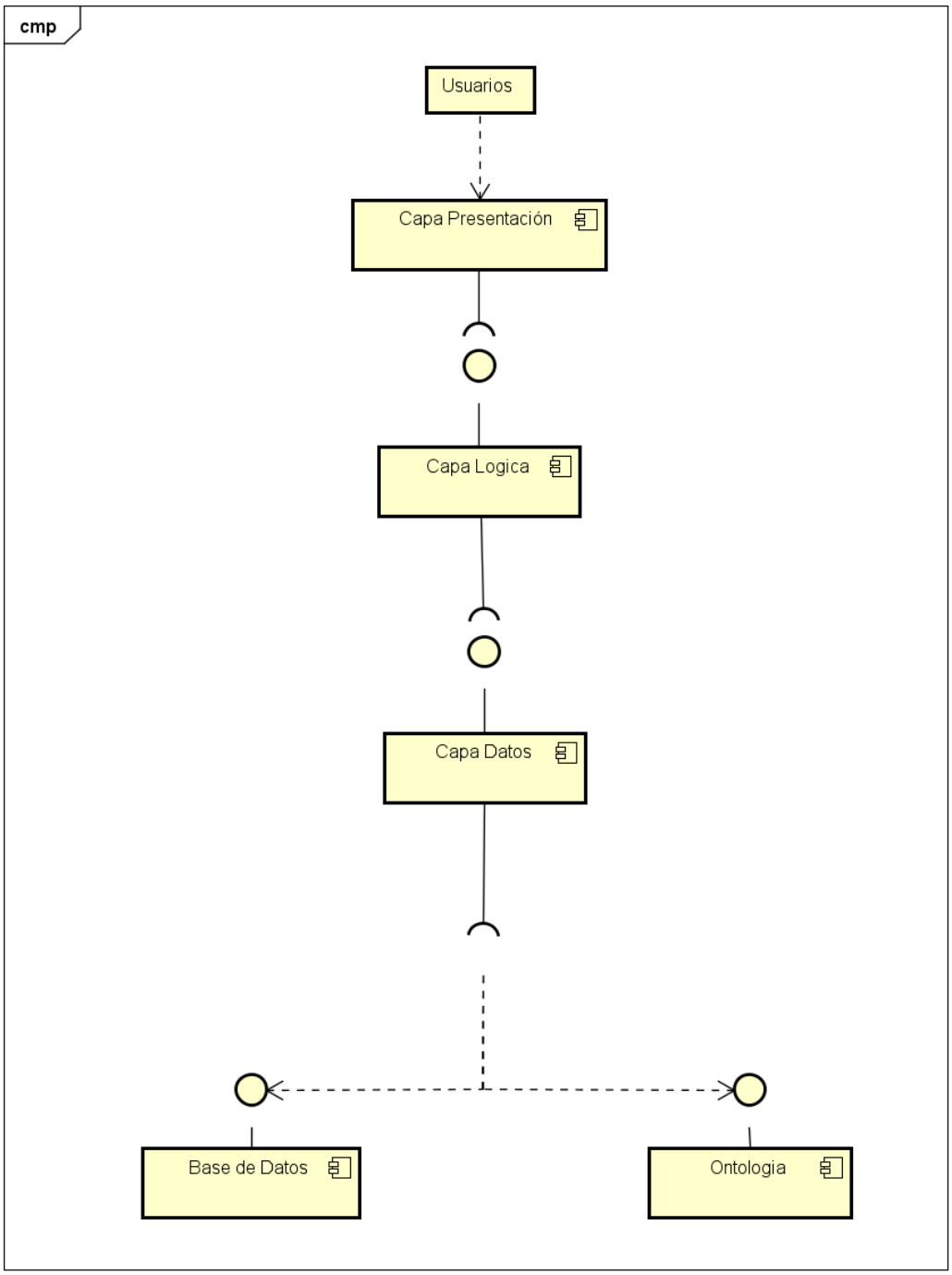

Figura 5.2: Arquitectura: visión lógica.

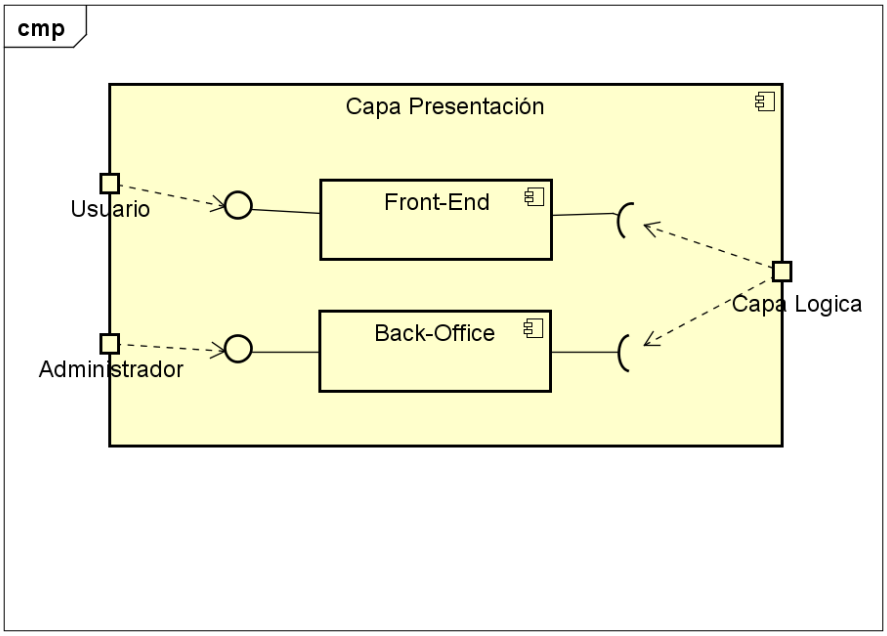

Figura 5.3: Subsistema capa de presentación.

Vista de Distribución

La distribución física del prototipo es la presentada en la figura 5.6, esta distribución se encuentra compuesta por nodos. Un nodo es el servidor web, en donde se aloja la capa de presentación, dicho nodo es el encargado de recibir las peticiones de usuarios, proveyendo funcionalidades para que usuarios interaccionen con la aplicación. En otro nodo se encuentra el servidor de aplicaciones, el cual contiene la capa lógica, la de datos y la ontología que utiliza el prototipo. En la capa de datos se utiliza la OWL API [2] para acceder a la ontología. En el prototipo se utiliza una base de datos relacional para gestionar los usuarios y la configuración, el servidor de base de datos en donde se encuentra la base de datos utilizada se encuentra en otro nodo. El manejador de base de datos utilizado es MySQL [2].

5.3. Pruebas realizadas sobre el prototipo

En esta sección se describen las pruebas realizadas sobre el prototipo desarrollado y se comentan los resultados obtenidos.

[2] Mysql. `https://www.mysql.com`. Accedida: 2016-04-16.

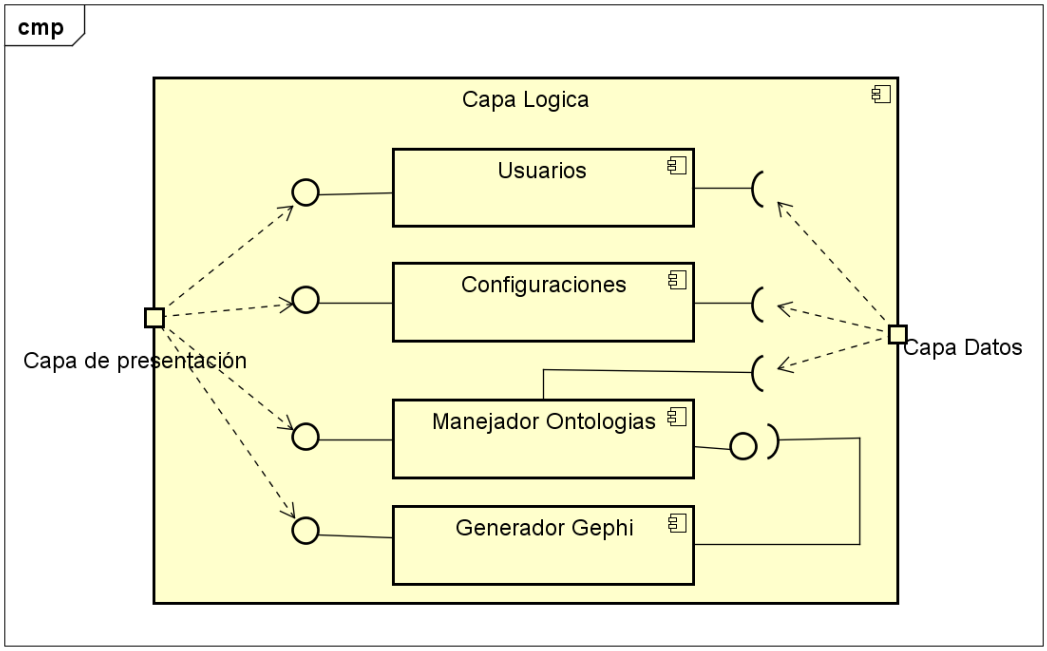

Figura 5.4: Subsistema capa lógica.

Pruebas con diferentes ontologías

El generador de formularios es capaz de adaptarse a ontologías de diferentes dominios. Para confirmar esta afirmación se probó con una selección de la lista de ontologías disponibles en el repositorio VIVO [3] y en el repositorio BioPortal [4]. Las pruebas realizadas consistieron en intercambiar la ontología y probar con un concepto que tuviera relaciones con otros de manera de crear subformularios.

A continuación se presentan ejemplos representativos de ontologías que resultaron no compatibles y se explica la razón.
MFOEM[5] es una ontología en la que se representa el dominio de las emociones y estados de ánimo, esta ontología no resultó compatible debido a que las relaciones no tienen el dominio y el recorrido definido.
IDODEN[6]**.** Es una ontología del virus del dengue. Sucedió lo mismo que en MFOEM, no resultó compatible debido a que el dominio y el recorrido de las relaciones no se encontraba definido.

[3]Directorio de ontologías VIVO - http://swl.slis.indiana.edu/repository/
[4]Sitio online de BioPortal - http://bioportal.bioontology.org/
[5]http://www.ontobee.org/ontology/MFOEM
[6]Descargada de - http://purl.bioontology.org/ontology/

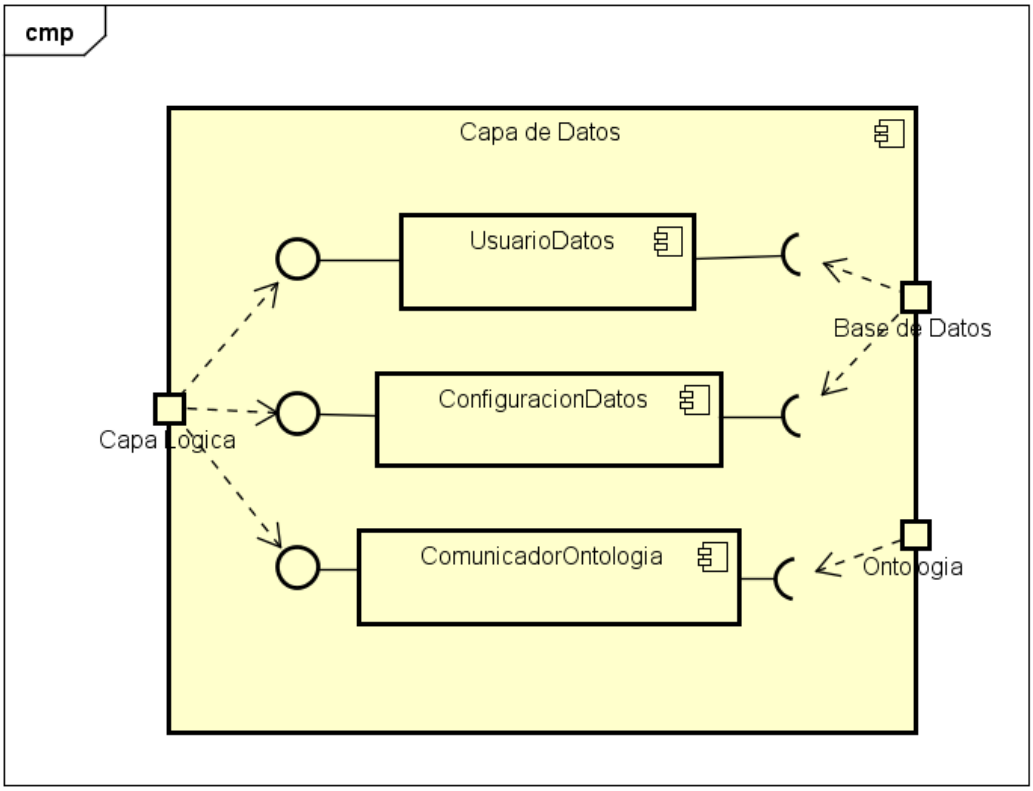

Figura 5.5: Subsistema capa de datos

Tesauro de términos de sicología[7] [8]**.** Es un tesauro con una lista de términos utilizados en el campo de la psicología. Esta ontología no tiene relaciones, por lo tanto ningún concepto tendría anclas que agregar.
Especificación de conceptos clínicos.[9] En esta ontología se especifican los términos clínicos y el contexto en el que se utilizan. En este caso se realizó una configuración, se generó el formulario y los resultados del mismo fueron guardados en la ontología. Por otro lado sucedió que los nombres de los conceptos no eran los mismos que su identificador como nombre del campo a completar, y en el generador se muestra el identificador, entonces el contexto no se logró comprender. Para lograr que el contexto se comprenda se debería haber usado como etiqueta del campo de los formularios, en este caso, el nombre del concepto en lugar de su identificador.

En las pruebas se concluye que el generador que se propone es útil para casos en

[7] Descargada de http://bioportal.bioontology.org/ontologies/APAONTO
[8] Thesaurus of Psychological Index Term® - http://taxonomies.labs.crossref.org/?p=114
[9] Clinical Building Blocks - WebProtege - http://webprotege.stanford.edu/

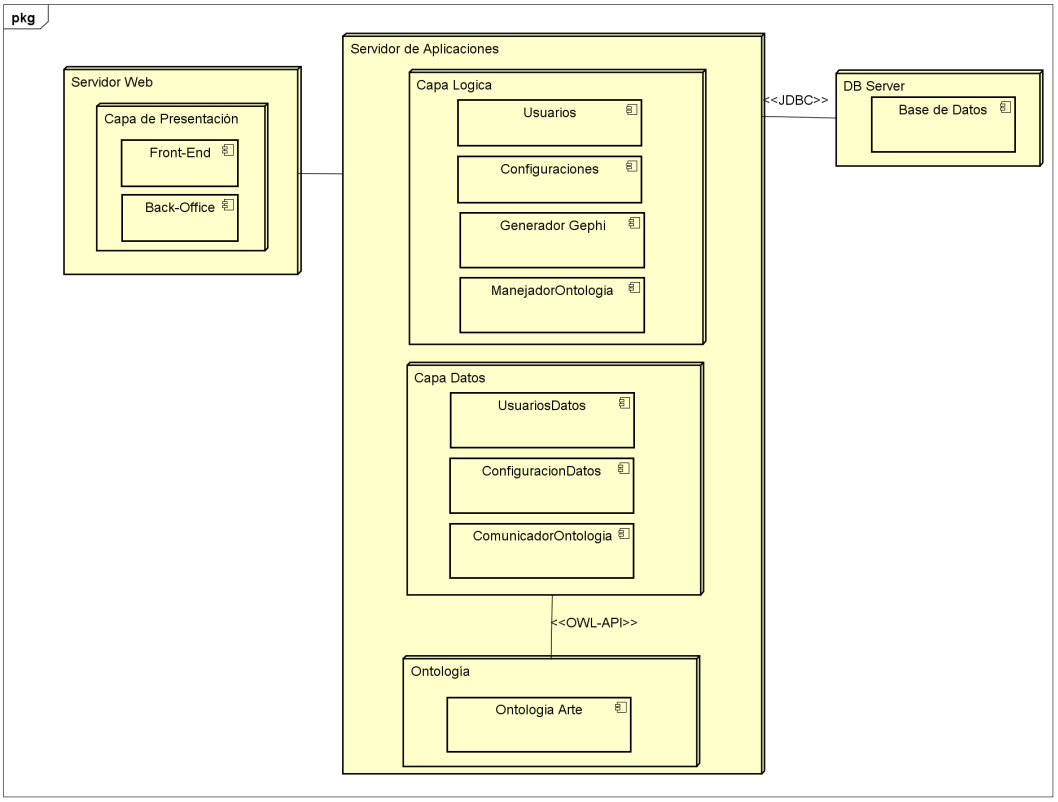

Figura 5.6: Arquitectura vista de distribución

los que se quiera ingresar datos sobre conceptos que tienen relación con jerarquías de conceptos (e.g. PiezaExposición se encuentra relacionado con Percepción, que tiene una jerarquía de conceptos). Si se tiene un concepto que tiene relaciones con otros conceptos de una ontología, pero estos no conforman una jerarquía, sucederá que la botonera principal será muy grande y se tendrá que navegar por muchas pantallas, transformando el proceso de completar el formulario en algo engorroso. De todas formas el funcionamiento del prototipo es correcto, es decir que es posible mediante el formulario generador ingresar datos a la ontología.
Se identificaron casos de ontologías que en realidad son taxonomías, y que no tiene propiedades de objeto, en este caso tampoco resulta útil el generador de formularios.

Las pruebas realizadas sobre conceptos en los que aplica la generación de formularios con el generador propuesto, fueron exitosos, esto es, se creó una configuración para este concepto, se completó el formulario y los datos fueron guardados en forma exitosa en la ontología correspondiente.

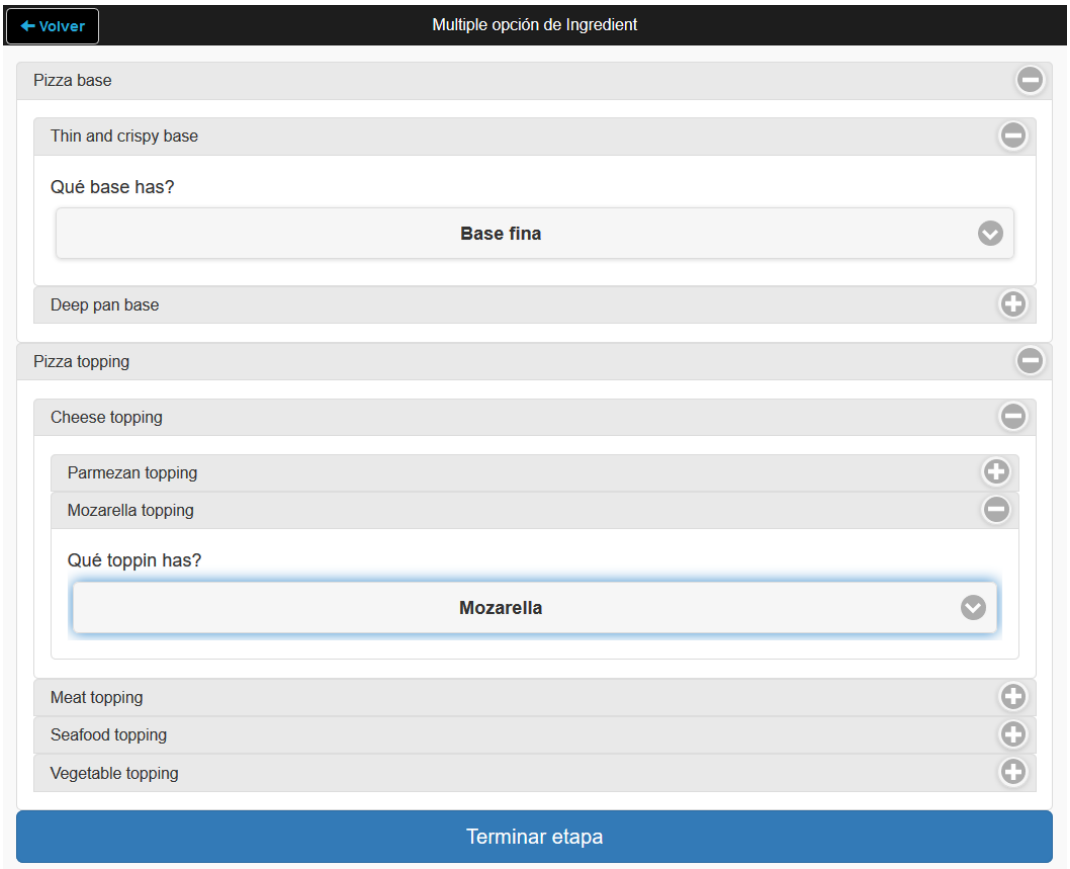

Figura 5.7: Ejemplo de formulario de tipo múltiple opción generado para la ontología de pizza.

Ejemplo. Un caso exitoso fue probar con la ontología *pizza.owl* que se genera en la Guía Práctica[16] de Protégé-OWL. En la ontología se representa el dominio de las pizzas. Un concepto importante es el concepto *Pizza* que tiene ingredientes de dos tipos, una base y una cubierta. Según la combinación de la base y la cubierta, un individuo de pizza es clasificado en diferentes categorías de pizza. La prueba con esta ontología consistió en realizar una configuración para ingresar una pizza y seleccionar la base y la cubierta de las existentes. El concepto objetivo en la configuración es *Pizza*, el ancla es *Ingredient* y es de tipo múltilpe opción. Se elije *Ingredient* como ancla debido a que es superclase de *PizzaBase* (i.e., base de la pizza) y también de *PizzaTopping* (i.e., la cubierta de la pizza). En la Figura se muestra la pantalla principal del formulario que contiene un solo botón para ingresar el nuevo ingrediente. En la Figura 5.7 se muestra el formulario de múltilpe opción generado con la selección de *Mozarella* como cubierta y de *Base fina* como base. Dado que el generador de

preguntas se pensó para ser utilizado con ontologías en español, las preguntas tienen componentes en español.

Pruebas Funcionales

Se realizaron pruebas funcionales para los casos de uso centrales del prototipo: *Crear Configuración* y *Completar el formulario*. Las pruebas consistieron en definir un flujo principal para cada caso y flujos alternativos. El flujo principal es el camino esperado del caso, pero a su vez este por distintas circunstancias puede tener eventos que cambien el curso del flujo, planteando así flujos alternativos que introducen nuevos escenarios de prueba.

Crear Configuración

Para realizar la prueba sobre el caso de uso *Crear configuración* se debe cumplir con la precondición de que el usuario haya iniciado sesión y tenga el rol administrador. Para este caso se definió el siguiente flujo principal:

1. El usuario ingresa al caso y el sistema lista todos los conceptos de la ontología.
2. El usuario selecciona uno de los conceptos como concepto objetivo y oprime siguiente paso.
3. El sistema lista todos los conceptos relacionados directamente con el concepto objetivo.
4. El usuario selecciona varios y oprime siguiente paso.
5. El sistema muestra los conceptos seleccionados para ser ordenados.
6. El usuario ordena los conceptos y se dirige al siguiente paso.
7. El sistema lista propiedades del concepto objetivo, para ser agregados al formulario.
8. El usuario selecciona uno y confirma la configuración.
9. El sistema consulta si quiere dejar activada la configuración.
10. El usuario selecciona que si y finaliza el flujo.

A partir de la definición del flujo principal, se identifican flujos alternativos, que son derivaciones en las distintas etapas. Los flujos alternativos son:

3A. El concepto objetivo seleccionado no tiene conceptos relacionados, el sistema no lista ningún concepto.

3A.1 El Usuario oprime siguiente paso

3A.a El Usuario oprime volver atrás

8A. El Usuario no selecciona propiedades del concepto objetivo y confirma

10A. El Usuario selecciona que no desea dejar activada la configuración.

Cada flujo presentado define un escenario para la realización de las pruebas. En la tabla 5.1 se muestra los distintos escenarios identificados, y la salida deseada para los mismos.

Tabla 5.1: Escenarios de la prueba funcional para el caso crear configuración

escenarios	*salida esperada*
Flujo principal completo (1).	Configuración activada
El concepto objetivo seleccionado no tiene conceptos relacionados y el usuario oprime siguiente paso (3A. y 3A.1).	El sistema despliega un mensaje de error.
El concepto objetivo seleccionado no tiene conceptos relacionados y el usuario oprime volver atrás (3A. y 3A.a).	Vuelve a la etapa 1 del flujo principal.
El Usuario no selecciona propiedades del concepto objetivo y confirma (8A).	Continua con la etapa 9 del flujo principal.
El Usuario selecciona que no desea dejar activada la configuración (10A).	Configuración no activada.

Los escenarios presentados fueron utilizados para crear los casos de prueba y verificar si el sistema tiene las salidas que se esperan. En las Tablas 5.2 y 5.3 se muestran los casos generados a partir de los escenarios. Para este caso el concepto objetivo define el escenario, por este motivo en la etapa de selección del concepto objetivo, se debe usar como entrada un concepto acorde para cada escenario. Para los escenarios *3A_3A.1* y *3A_3A.a* se debe seleccionar un concepto que no tenga otros conceptos relacionados, por este motivo se utilizó el concepto *Cromatismo*, que permite mantener las condiciones de estos 2 escenarios. A su vez para los otros escenarios se debía utilizar un concepto que sí contenga relaciones con otros, en este caso se utilizó *PiezaExposición*. Luego de elegir un concepto objetivo para poder

plantear cada escenario, se siguieron los pasos descritos por los flujos que definen los escenarios. En todos los escenarios el sistema respondió como se esperaba. Podemos afirmar que la funcionalidad *crear configuración* valida la prueba funcional propuesta.

Tabla 5.2: Casos de prueba a partir de escenarios, para crear configuración

	escenario 1	*escenario 3A_3A.1*	*escenario 3A_3A.a*
Entradas			
Selección concepto objetivo (2)	PiezaExposición	Cromatismo	Cromatismo
Selección propiedad de objeto objetivo (8)	tieneEnergía	—	—
Indicar si se quiere dejar la configuración activada	SI	—	—
Salida	Configuración nueva activada	ERROR! Debe seleccionar al menos un tema	Vuelve a selección concepto objetivo

Tabla 5.3: Casos de prueba a partir de escenarios, para crear configuración

	escenario 8A	*escenario 10A*
Entradas		
Selección concepto objetivo (2)	PiezaExposicion	PiezaExposicion
Selección propiedad de objeto objetivo (8)	no selecciona nada	tieneEnergía
Indicar si se quiere dejar la configuración activada	SI	NO
Salida	Configuración nueva activada	Configuración nueva no activada

Completar Formulario

Existen 2 tipos de formularios a completar, formularios de ingreso de datos y formularios de múltiple opción. Por este motivo, en la prueba funcional para el caso *Completar formulario* se probaron estos dos tipos de formularios.

Se define un formulario a completar a través de una configuración, este formulario tiene un subformulario de ingreso de datos para una pieza y un subformulario de múltiple opción para seleccionar el medio de producción de la pieza.
Para la realización de esta prueba se debe cumplir con la precondición de que el usuario haya iniciado sesión y que exista una configuración para el rol del mismo. A continuación se muestra el flujo principal de este caso:

1. El sistema muestra 2 botones de color azul, uno con el nombre pieza y otro con el nombre Medios de Producción.
2. El usuario oprime el botón pieza.
3. El sistema muestra un formulario de ingreso de datos para el concepto pieza.
4. El usuario ingresa todos los datos del formulario y oprime terminar etapa.
5. El sistema vuelve a mostrar los 2 botones, pero el botón pieza en verde.
6. El usuario oprime botón Medios de Producción.
7. El sistema muestra un formulario múltiple opción para el concepto MedioDe-Producción.
8. El usuario completa los datos de todos los campos del formulario y oprime terminar etapa.
9. El sistema vuelve a mostrar los 2 botones, pero ahora los 2 en verde.
10. El usuario confirma formulario.
11. El sistema solicita una nueva confirmación.
12. El usuario confirma.

A partir del flujo principal para el caso *completar formulario* se identificaron, los siguientes flujos alternativos:

4A. El usuario no ingresa ningún dato y oprime terminar etapa.

4B. El usuario oprime volver atrás.

8A. El Usuario no completa ningún campo y oprime terminar etapa.

8B. El usuario oprime volver atrás.

Tabla 5.4: Escenarios de la prueba funcional para el caso crear configuración

escenarios	*salida esperada*
Flujo principal completo (1).	El sistema indica que los datos fueron guardados y vuelve a la etapa 1
El usuario no ingresa ningún dato y oprime terminar etapa (4A.).	El sistema despliega un mensaje de error para campos obligatorios.
El usuario oprime volver atrás(4B.).	Vuelve a la etapa 1 del flujo principal.
El Usuario no completa ningún campo y oprime terminar etapa (8A.).	Continua con la etapa 9 del flujo principal.
El usuario oprime volver atrás (8B).	vuele al paso 5 del flujo principal.

En la Tabla 5.4 se muestran los escenarios identificados a partir de todos los flujos definidos para el caso.

Con estos escenarios identificados, se definieron los casos de prueba mostrados en las Tablas 5.5 y 5.6. Se puede afirmar que las salidas son correctas, ya que se planificó para que haya controles sobre campos vacíos en subformularios de ingreso de datos solo sí se confirmaba la etapa, por este motivo el caso de prueba del *escenario 4A*, tiene como salida mensajes de error de campos obligatorios que no se completaron. A su vez, para los subformularios de múltiple opción el control de campos obligatorios no se realizó, ya que en este subformulario, se pueden dejar muchos campos sin llenar. Por este motivo el *escenario 8A* prosigue su ejecución, como si se hubieran llenado todos los campos. En estas condiciones se puede afirmar que el caso paso la prueba con éxito.

Pruebas de validación con el cliente

Se realizaron reuniones con el cliente para validar las funcionalidades que definió como las más importantes para su aplicación. Se probaron funcionalidades generales, pero también excepcionales. Las funcionalidades generales que se probaron fueron, crear una nueva configuración y completar un formulario. Las funcionalidades excepcionales que se probaron fueron, verificar clasificación del individuo de PiezaExposición ingresado y crear un grafo de Gephi para una exposición.

La primera fase de las pruebas fue crear la configuración, se le explicó al cliente todas las etapas de crear configuración y se siguieron junto a él. Se creó una configuración para usuario administrador y se marcó como configuración activa para ese perfil. Por ejemplo, en la etapa de configuración, dado que la configuración lleva

Tabla 5.5: Casos de prueba a partir de escenarios, para completar formulario

	escenario 1	*escenario 4A*	*escenario 4B*
Entradas			
subformulario pieza	completa formulario y oprime termina etapa	no ingresa datos y oprime terminar etapa	no ingresa datos y oprime salir
subformulario Medio de Producción	Completa todos los datos y oprime terminar etapa	—	—
Salida	EXITO! Encuesta guardada correctamente.	El sistema muestra error de todos los campos obligatorios	vuelve al paso 1

Tabla 5.6: Casos de prueba a partir de escenarios, para crear configuración

	escenario 8A	*escenario 8B*
Entradas		
subformulario pieza	Completa formulario y oprime termina etapa	Completa formulario y oprime termina etapa
subformulario Medio de Producción	No completa ningun campo y oprime terminar etapa	oprime salir
Salida	Pasa a la etapa 9 del flujo principal	Vuelve a la etapa 5 del flujo principal

varios pasos (Ver manual configuración Anexo B), se identificó la necesidad de un botón para volver al paso anterior. Finalmente se corroboró que la configuración estuviera listada en la tabla de configuraciones y que estuviera activa.
Luego se realizó el caso de completar el formulario. Al comenzar el caso se verificó que el formulario generado correspondía con la configuración definida anteriormente. Luego se completó el formulario y se guardó. En esta etapa también se identificaron cuestiones relacionadas al nombre de botones y organización de los componentes en pantalla. Finalmente se guardaron los datos ingresados. Para verificar que la pieza en la exposición se clasificó correctamente, se consultó en la tabla de respuestas del usuario.
Se completó dos veces mas el formulario con piezas dentro de la misma exposición y luego se generó el grafo de Gephi para la exposición utilizada en el ejemplo.

La intención de las pruebas fue evaluar si el prototipo desarrollado tiene el funcio-

namiento esperado por el cliente (que también es usuario). Al realizar las pruebas se identificaron aspectos a mejorar relacionados a la interfaz gráfica que se impactaron en el prototipo (e.g. navegabilidad, ubicación de botones, etc).

Prueba de estrés

Las pruebas de estrés de la aplicación se realizaron con el objetivo de identificar riesgos que deberán ser mitigados si la aplicación es utilizada por un gran número de usuarios. Las pruebas fueron realizadas sobre el caso de uso completar encuesta que será el más utilizado en la aplicación.
Las pruebas se realizaron utilizando JMeter [10], una herramienta creada para realizar pruebas de carga y desempeño en aplicaciones, que se encuentra implementada en java y es de código abierto.

En la simulación se reprodujo el comportamiento de un usuario que consistió en que ingrese a la aplicación, complete el formulario y lo guarde. Antes de comenzar, se creó una configuración de formulario que se utilizó para todas las pruebas. El proceso de completado del formulario se detalla a continuación. Se ingresa al subformulario de exposición, se selecciona una exposición y se confirma. Se ingresa al subformulario de pieza, se ingresan los datos correspondientes y se confirma. Se ingresa al subformulario de percepción, se selecciona percepción visual de imagen fija con valor *superficie lisa*, y percepción de movimiento real con valor *movimiento frontal* y se confirma. Se ingresa a la etapa de medios de producción en donde se selecciona la técnica de grabado *litografía* y se confirma. Por ultimo se ingresa a la etapa otros datos y se confirma sin cambiar ningún dato. Luego de este proceso se confirma el formulario y los datos son guardados en la ontología.

La prueba consistió en correr el caso utilizando JMeter simulando distintas cantidades de usuarios concurrentes. Se simularon instancias con 10, 20, 40, 80 y 160 usuarios concurrentes. En cada instancia se ejecutó el caso 5 veces dado que el tiempo de ejecución es no determinista, por lo que los resultados reportados son el promedio de cada instancia.

En la Tabla 5.7 se muestra el tiempo de respuesta promedio para cada instancia. Los resultados mostrados son basados en el tiempo de respuesta de toda la prueba y el de las acciones mas importantes que tiene el caso de prueba, estas son: el acceso la formulario para completarlo, el acceso al subformulario de percepciones y por ultimo la confirmación y guardado de todos los datos del formulario. En todos los casos, salvo en acceder al subformulario de excepciones, al aumentar la cantidad de usuarios concurrentes simulados aumentó el tiempo de respuesta de la aplicación. El

[10] Pagina web de Jmeter. `http://jmeter.apache.org/`. Accedida: 2016-04-12.

Tabla 5.7: Tiempos promedio de respuesta de cada instancia medido en milisegundos

	# usuarios concurrentes simulados				
Acciones	10	20	40	80	160
Acceder al formulario	47 ms	54,4 ms	98,6 ms	170 ms	989,6 ms
Acceder al subformulario percepciones	82 ms	57,2 ms	66,8 ms	125,6 ms	248,2 ms
Confirmar formulario	362,6 ms	635,4 ms	1684,4 ms	7500,6 ms	20710,6 ms
Prueba completa	2629,2 ms	4104,4 ms	8529,4 ms	20351,4 ms	46517,4 ms

aumento en el tiempo de respuesta es un indicador de degradación del sistema al realizar dicha acción, en caso de que la aplicación sea utilizada por un gran número de usuarios se tendrá que poner foco en las operaciones que tuvieron el mayor tiempo de respuesta. La operación que tuvo un mayor crecimiento en el tiempo de respuesta fue el guardado en la ontología, también esta operación es la de mayor tiempo de respuesta en todas las instancias, esto es debido a que en al confirmar el formulario, se hacen accesos a la ontología para guardar los datos ingresados.

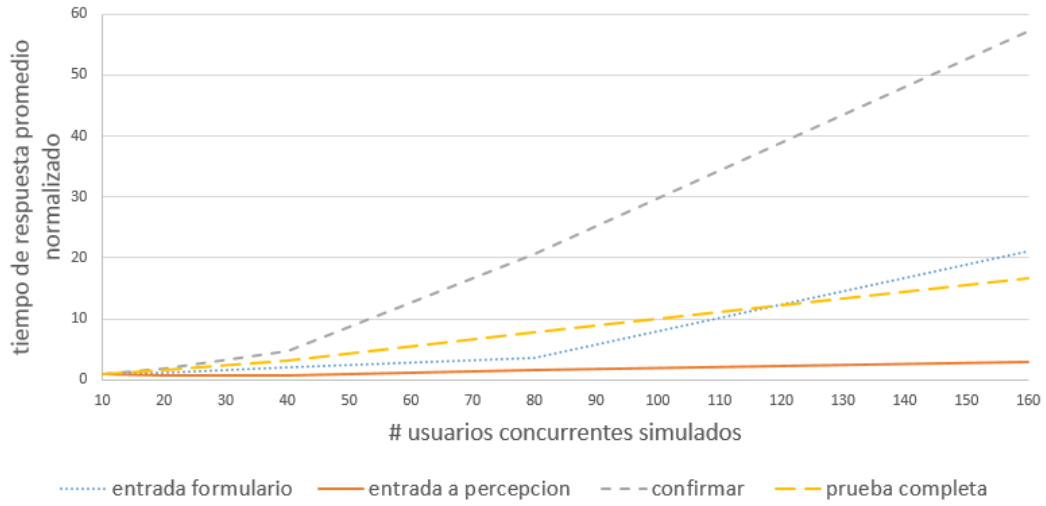

Figura 5.8: Tiempo de respuesta al confirmar el formulario general.

En la Gráfica 5.8, se muestra como evoluciona el tiempo de respuesta para las operaciones de acceso al formulario, acceso a percepciones, confirmación del formulario y para la prueba completa. La operación de confirmar el formulario es la que tiene el crecimiento mas acelerado del tiempo de respuesta. Esta situación podría

indicar un crecimiento en el tiempo de espera de las peticiones cuya causa podría ser un cuello de botella en el acceso a la ontología para guardar los datos.

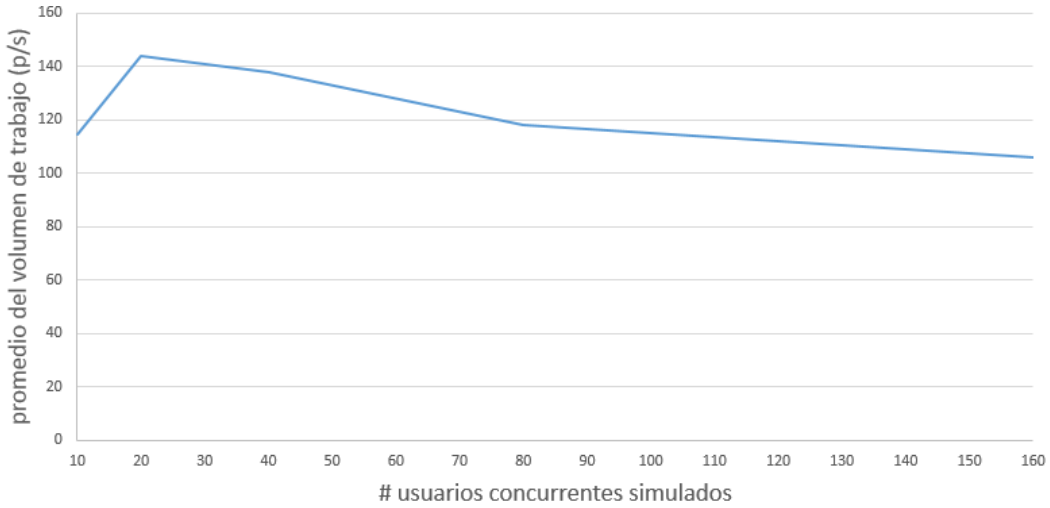

Figura 5.9: Promedio del volumen de trabajo total en cada instancia, medido en peticiones atendidas por segundo (p/s)

En la Gráfica 5.9 se muestra el volumen de trabajo promedio de cada instancia, medido en peticiones atendidas por segundo para el caso de prueba completo. La cantidad de peticiones procesadas es mayor en instancias en las que se simuló mas cantidad de usuarios concurrentes. Sin embargo, la cantidad de peticiones atendidas por segundo crece hasta los 20 usuarios concurrentes y luego comienza a decrecer. Esta situación podría justificarse porque a partir de 20 usuarios se llega a una cota superior de peticiones procesadas por segundo que tiene el prototipo, por este motivo, en las instancias con mayor cantidad de usuarios concurrentes, el número de peticiones por segundo crece sobrepasando la cota superior, formando así colas de atención de peticiones.

En caso de que la aplicación tenga un alto número de usuarios, se deberá evaluar de que forma se puede mejorar la ejecución del caso de uso completar formulario. En la Gráfica 5.9, se muestra que la operación de *confirmar el formulario* es la que tiene los tiempos de respuestas mas altos, por este motivo es la operación a mejorar para un mejor rendimiento del caso de uso. Una forma de mejorar el rendimiento en esta operación es evitar guardar en la ontología en el momento que se confirma el formulario, guardando las respuestas en una base de datos y programar un proceso automático, que inicie a una hora en particular, y pase todos las respuestas de la base de datos a la ontología. Esta solución tiene la desventaja de que no será posible saber en que familia clasificó una pieza de arte en el momento siguiente a confirmar

el formulario.

5.4. Tecnologías utilizadas

En esta sección se comentan las tecnologías utilizadas para el desarrollo del prototipo.

OWL API

La OWL (Web Ontology Language) API es una interfaz que provee soporte a la creación y manipulación de ontologías, está implementada en java y fue habilitada desde 2003 como código abierto. Fue evolucionando a la par de OWL en lo que se refiere a incorporar lo cambios en el estándar. Una ontología es vista por la API como un conjunto de axiomas y anotaciones, proveyendo a quien la utiliza de un nivel de abstracción mayor.

Hermit

Hermit es un razonador compatible con el estándar OWL 2. Un razonador se utiliza para realizar inferencias en base a las reglas definidas para clases y propiedades en una ontología. Por medio de las inferencias en el razonador se chequea la consistencia de la ontología, pero también provee otras funcionalidades como clasificación de clases y propiedades, y creación de grafos que describen la ontología. (Glimm et al 2014) [11].

Java EE 7

Según Juneau [19], Java EE (Enterprise Edition) 7 es una plataforma para desarrollo de aplicaciones de escritorio, web y mobile en lenguaje Java [18]. Desde su lanzamiento en 2001 ha evolucionado para ayudar a que el desarrollo de aplicaciones se efectúe en forma rápida y eficiente. Se provee soporte para Web Services, Java Server Faces (estándar de desarrollo de páginas web que provee un nivel de abstracción mayor a html), soporte para seguridad de autenticación de usuarios mediante JAAS, entre otros. Los productos asociados a Java EE 7 permiten que se puedan desarrollar aplicaciones de gran porte.

Primefaces

Según Çalışkan et al (2015) [6] Primefaces es un librería de componentes visuales para JSF de código abierto, facilita la creación de aplicaciones web para dispositivos móviles o de escritorio. Para correr una aplicación que utiliza Primefaces es necesario

tener la versión 5 o más de la maquina virtual de java y la implementación de JSF versión 2.

Wildfly

Según Kumar et al (2014) [23] wildfly es el nuevo nombre para el servidor de aplicaciones JBOSS construido por Red Hat. Está pensado para dar soporte a aplicaciones construidas en java empresarial (Java EE), fue construido en java y es de código abierto. La ventaja de ser construido en java es que Wildfly puede correr en cualquier plataforma que soporte java.

MySQL

MySQL [11] es un servidor de base de datos relacional ampliamante utilizado, está escrito en C y en C++ y es de código abierto. El servidor es provisto como un programa independiente, este a su vez provee librerías para que las aplicaciones utilicen el servidor de base de datos de forma independiente.

Gephi toolkit API

Gephi es una plataforma para la creación de grafos se encuentra desarrollada en Java y es de uso gratuito. Junto con Gephi se creó la Gephi toolkit API que permite la creación y edición de grafos desde otras aplicaciones. (Bastian et al 2009) [5].

JAAS

JAAS (Java Authentication and Authorization Service) [24] es un componente provisto por java, es utilizado en aplicaciones multiusuarios, JAAS provee el servicio de autenticación de los usuarios y la asignación de privilegios para los mismos.

5.5. Conclusiones

En este capitulo se describe el prototipo creado, como se constituye su arquitectura tanto del punto de vista lógico como del punto de vista físico, los casos de usos considerados críticos y la pruebas realizadas sobre el prototipo.
La aplicación fue pensada en principio para ser utilizada por un grupo de reducido de personas, designadas por el experto en arte. Eventualmente podría extenderse a ser utilizado por muchos usuarios, a su vez se podría utilizar el modulo de manejo de la ontología como proveedor de servicios para otras aplicaciones que necesiten de la ontología, ya sea para realizar otra implementación de los formularios o simplemente para extraer o insertar datos en el misma. En la Figura 5.10 se muestra una nueva

[11] Mysql. `https://www.mysql.com`. Accedida: 2016-04-16.

distribución para contemplar los nuevos escenarios planteados. Se propone mejorar la distribución física de la aplicación agregando un nuevo servidor de aplicaciones el cual contendrá una capa de servicios (para exponer los servicios que va a brindar el modulo), una capa lógica que contiene al componente Manejador de Ontologías, y una capa de datos con el Comunicador de Ontologías que mediante la OWL-API se conecta a la ontología. Esta nueva distribución independiza el modulo de gestión de datos y creación del grafo a partir de la ontología. Este modulo deja de compartir recursos con la aplicación web, teniendo a su disposición los mismos para brindar el servicio a otras aplicaciones.
Las distintas pruebas realizadas y planteadas en el capitulo sirvieron para detectar errores sobre funcionalidades importantes. A su vez las pruebas de estrés realizadas sobre el caso *completar formulario* brindan una visión de cual es el limite de usuarios que pueden estar al mismo tiempo completando el formulario, este limite cumple con la exigencia del experto, sin embargo se plantea un posible cambio sobre el prototipo para poder soportar aún más usuarios que lo exigido. Dichos cambios tienen la desventaja de que se pierde la posibilidad de clasificar la pieza en el instante posterior de ingreso de los datos de la misma mediante el formulario.

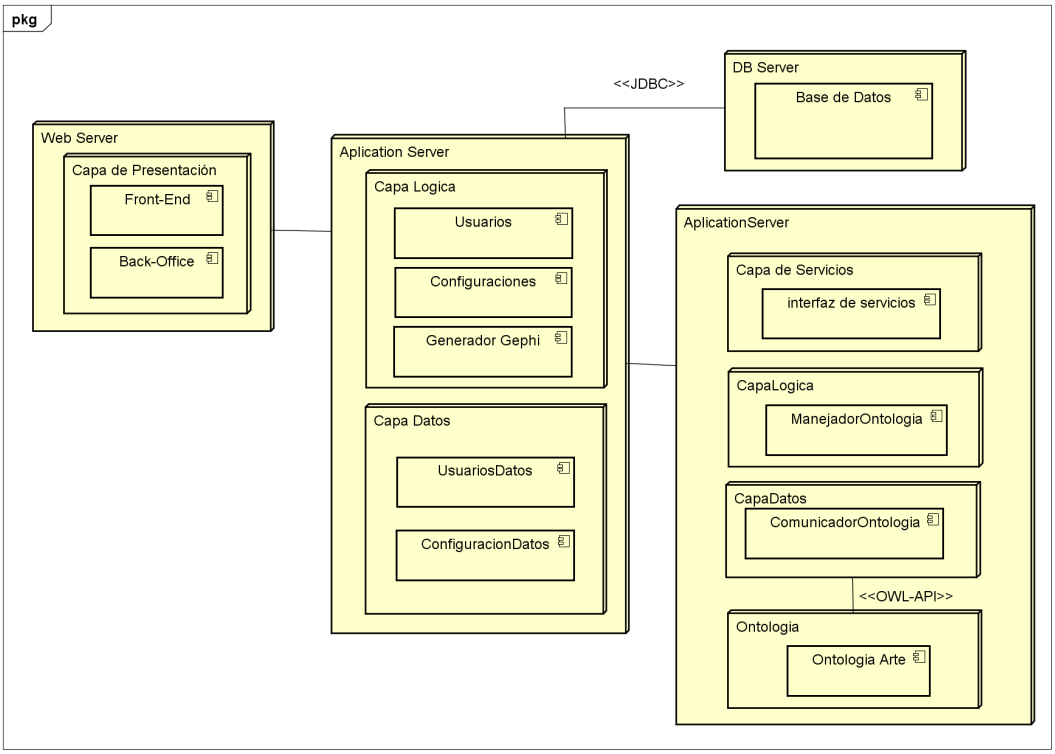

Figura 5.10: Arquitectura distribución

Capítulo 6

Conclusiones y Trabajos Futuros

En este trabajo se presenta el proceso de análisis, diseño, desarrollo y reporte de resultados del proyecto *Generación automática de formularios para el ingreso de datos en ontologías - Aplicación en la implementación de una ontología de percepciones de piezas de arte*. El resultado del proyecto es: (i) una ontología en la que se representa el dominio de percepciones, medios de producción, soportes y familias de producción de piezas de arte en exposiciones, (ii) un generador automático de formularios, y (iii) un prototipo en el que se utiliza el generador de formularios.

6.1. Objetivos Logrados y Conclusiones

Un primer resultado de este trabajo es contar con la Ontología OPPA sobre percepciones en piezas de arte. Esta ontología se desarrolló de forma conjunta con el experto en el campo del arte Vladimir Muhvich. La ontología tiene como objetivo conceptualizar la realidad del campo del arte, para investigadores en arte, al momento de atender a exposiciones y relevar datos de las piezas que allí se exponen. En la Ontología OPPA desarrollada en este proyecto se describen las características de las piezas desde el punto de vista de las percepciones sensoriales, los medios y soportes utilizados para su producción. Asimismo, se describen las familias de producción artística a las que pertenecen las piezas que se exponen.
En cuanto al proceso de diseño y desarrollo de la Ontología OPPA, se utilizó el método *METHONTOLOGY*, que consta de dividir el ciclo de vida de una ontología en etapas. En el Capitulo 3 de este informe se describen los pasos seguidos en la aplicación de *METHONTOLOGY* en el marco del proyecto. Durante el proceso se trabajó en conjunto con el experto en el campo del arte (i.e.: el dominio de la ontología) Vladimir Muhvich, quien proveyó material del proyecto *Engrama, investigación sobre modelos de visualización morfológica y evolutiva de Colecciones de Arte* en el cual utiliza una taxonomía en la que se basa la ontología, y ayudó a comprender el

dominio de la ontología construida. Muhvich, asumiendo el rol de cliente, validó los componentes de la ontología.
En la ontología desarrollada, se catalogan las *piezas de arte en exposiciones* en *Familias de producción* por medio de reglas definidas en conjunto con el uso de una herramienta llamada razonador, que es utilizado para inferir conocimiento a partir de dichas reglas. Con la información de las piezas en exposiciones que se ingresa por medio de los formularios, y por medio de la herramienta para realizar inferencias, se clasifican las piezas expuestas en la familia a la que correspondan en forma automática desde la ontología.

En cuanto al generador de formularios, se presentó la motivación de ingresar datos a ontologías por medio de formularios. Se presenta una estructura de grafo en la que se realiza una correspondencia de componentes de la ontología a instancias de clases, el grafo generado es en el que se basa el algoritmo generador de formularios. Se presentaron los dos tipos de subformularios implementados: múltilpe opción e ingreso de datos. Se presentaron ejemplos de utilización de ambos formularios.
Con la finalidad de mantener la generalidad se plantea que el generador tenga la capacidad de soportar cambios en la ontología utilizada, y eventualmente el cambio de una ontología por otra que incluso podría representar otro dominio. Para lograr la generalidad, los formularios se generan en forma semiautomática luego de realizarse un paso de configuración en el que se especifican algunas de sus características. En la etapa de configuración es en la que se decide el tipo de formulario a generar. Se comenta también acerca del algoritmo desarrollado para guardar los datos en la ontología.
En el caso del formulario de múltiple opción se presentó la utilización de una estrategia basada en el contexto del concepto que provee las opciones como apoyo para la generación de una pregunta. La estrategia de generación de preguntas se desarrolló para ontologías implementadas en español, no se provee soporte a otros lenguajes. Según Muhvich, las preguntas generadas con la estrategia planteada colaboran con un mejor entendimiento de los datos que se solicitan en los formularios generados. Esta estrategia fue una buena solución con respecto al costo que llevó su implementación y el beneficio que generó llevarlo acabo.

En lo referente al prototipo, este se construyó para comprobar el funcionamiento del generador de formularios automáticos. La etapa de configuración es llevada a cabo por un usuario de tipo administrador, este usuario debe conocer el *concepto objetivo* del formulario a generar perteneciente al dominio de la ontología (e.g., en el caso de la ontología de arte, un concepto objetivo es PiezaExposición). La configuración consiste en elegir un *concepto objetivo* del formulario, para cada objetivo elegir los conceptos *ancla* que serán los principales de los subformularios, y luego elegir el tipo de formulario a generar para cada ancla. La etapa de configuración fue planteada para mantener la independencia del generador con respecto a la ontología,

por lo tanto es posible cambiar la ontología por otra en la que se represente otro dominio, reutilizando todo el código del generador. Mediante la etapa de configuración, además se provee la libertad de realizar diferentes tipos de formularios, según el perfil de los usuarios que los completarán o el componente de la ontología al que se le quiera ingresar datos.
Cada configuración generada se asigna a un grupo de usuarios en particular, luego los formularios se generan en forma automática para todos los usuarios en base a la configuración establecida para el grupo al que pertenece. Se generan instancias de los formularios en forma automática en tiempo de ejecución para cada usuario que comienza a completar un nuevo formulario, por lo tanto si la estructura de la ontología cambia, sin realizar cambios en el código los cambios de la ontología impactan automáticamente en el formato del formulario. En ocasiones los cambios en la ontología provocan la obsolescencia de las configuraciones definidas, (e.g., un concepto ancla dejó de existir) en ese caso se deberá realizar una nueva configuración.
El objetivo del formulario es permitir que se ingrese a la ontología un nuevo individuo que pertenece al *concepto objetivo* del formulario y definir su relación con individuos pertenecientes a otros conceptos. Los individuos con los que se relaciona el concepto objetivo pertenecen a un concepto que es subclase de un ancla.
En el caso concreto del cliente, el objetivo del formulario es ingresar un nuevo individuo al concepto PiezaExposición (i.e. concepto objetivo), y relacionarlo con percepciones, medios de producción y soportes (i.e. anclas) existentes en la ontología. En este caso las anclas son el concepto de mayor jerarquía de una jerarquía de conceptos.

En el marco de una exposición de piezas de arte, se ingresan datos de varias piezas, luego de terminar de relevar los datos, el experto genera un grafo para ser visualizado en la herramienta Gephi. El grafo es generado mediante una funcionalidad que automatiza la recolección de datos ingresados para la exposición seleccionada en la ontología.

Como resultado del proyecto se obtuvo una aplicación que le permite al experto Vladimir Muhvich agilizar su actividad de investigación. El experto relevaba todos los datos de las obras en una exposición en papel, luego ingresaba los datos en un archivo de hoja de cálculo a mano. La hoja de cálculo luego se utilizaba como medio de entrada datos para generar grafos en la plataforma Gephi. Al utilizar el prototipo implementado, se ingresan todas las piezas de una exposición en la ontología por medio de los formularios, y luego mediante la funcionalidad de generación de grafos implementada, el experto puede exportar los datos de la exposición a un archivo en el formato a utilizar en la plataforma Gephi.

6.2. Trabajos futuros

Entre los lineamientos de los trabajos futuros se encuentra extender la clasificación en familias por medio de un estudio en mas profundidad de las características de cada tipo de obra. Además, interesa extender la ontología para incluir la realidad de las zonas del cerebro afectadas por los tipos de percepciones. Esto permitiría implementarlo como una clasificación de piezas en exposición paralela a la de Familias que se realizó en este trabajo.

En cuanto al prototipo implementado se planea realizar su implantación y su entrada a producción en un futuro cercano dado que el experto Vladimir Muhvich se mostró interesado en su utilización. En este contexto se realizará un seguimiento a la utilización del prototipo con la finalidad de proveer soporte ante posibles fallos en la aplicación y también para recolectar datos acerca de la experiencia de uso del prototipo para evaluar posibles mejoras a futuro.
Se observa interesante extender el prototipo incluyendo soporte a formularios mixtos, que se encuentren formados por componentes de múltiple opción y de ingreso de datos.
En caso de que se quiera que el prototipo pase a ser utilizado por un gran numero de personas, seria entonces necesario implementar los cambios en la arquitectura para pasar a un esquema distribuido tal como se presenta en la conclusión del Capítulo 5.

Dado el interés que despertó la existencia de la Ontología OPPA en contactos de V. Muhvich con otros investigadores del arte, se encuentra entre la planificación a futuro la publicación de un artículo acerca de la ontología de percepciones y medios de producción desarrollada, dicha publicación comprende el proceso de desarrollo utilizado, una descripción detallada de los resultados obtenidos.
Se planea otra publicación en la temática del generador de formularios automáticos implementado, en dónde se presentará la funcionalidad del generador implementado en el prototipo y la etapa de configuración previa para cada formulario.

Bibliografía

[1] Página en la red de protege. `http://protege.stanford.edu/`. Accedida: 2016-04-08.

[2] Página web de owl api. `http://owlapi.sourceforge.net/`. Accedida: 2016-04-15.

[3] Maha Al-Yahya. Ontoque: a question generation engine for educational assesment based on domain ontologies. In *Advanced Learning Technologies (ICALT), 2011 11th IEEE International Conference on*, pages 393–395. IEEE, 2011.

[4] Murtha Baca, Patricia Harpring, College Art Association, et al. *Categories for the Description of Works of Art*. 2006.

[5] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. 2009.

[6] Mert Çalışkan and Oleg Varaksin. *PrimeFaces Cookbook*. Packt Publishing Ltd, 2015.

[7] Oscar Corcho, Mariano Fernández-López, Asunción Gómez-Pérez, and Angel López-Cima. Building legal ontologies with methontology and webode. In *Law and the semantic web*, pages 142–157. Springer, 2005.

[8] Stephen Davies. Ontology of art. *The oxford handbook of aesthetics. Oxford University Press, Oxford*, pages 155–180, 2003.

[9] Andreas Eberhart. Automatic generation of java/sql based inference engines from rdf schema and ruleml. In *The Semantic Web—ISWC 2002*, pages 102–116. Springer, 2002.

[10] Mariano Fernández-López, Asunción Gómez-Pérez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. 1997.

[11] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: an owl 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[12] Rafael S Gonçalves, Samson W Tu, Csongor I Nyulas, Michael J Tierney, and Mark A Musen. Structured data acquisition with ontology-based web forms.

[13] Thomas R Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993.

[14] Georg Hohmanna and Mark Fichtnerb. Embedding an ontology in form fields on the web.

[15] Matthew Horridge, Scott Brandt, Bijan Parsia, and Alan L Rector. A domain specific ontology authoring environment for a clinical documentation system. In *Computer-Based Medical Systems (CBMS), 2014 IEEE 27th International Symposium on*, pages 329–334. IEEE, 2014.

[16] Matthew Horridge, Simon Jupp, Georgina Moulton, Alan Rector, Robert Stevens, Chris Wroe, and Sebastian Brandt. A practical guide to building owl ontologies using protégé 4 and co-ode tools edition1. 3. *The University of Manchester*, 2011.

[17] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible sroiq. *Kr*, 6:57–67, 2006.

[18] Bill Joy, Guy Steele, James Gosling, and Gilad Bracha. Java (tm) language specification. *Addisson-Wesley, June*, 2000.

[19] Josh Juneau. *Introducing Java EE 7: A Look at What's New.* Apress, 2013.

[20] Aditya Kalyanpur, Daniel Jiménez Pastor, Steve Battle, and Julian A Padget. Automatic mapping of owl ontologies into java. In *SEKE*, volume 4, pages 98–103. Citeseer, 2004.

[21] Holger Knublauch, Ray W Fergerson, Natalya F Noy, and Mark A Musen. The protégé owl plugin: An open development environment for semantic web applications. In *The Semantic Web–ISWC 2004*, pages 229–243. Springer, 2004.

[22] Philippe Kruchten. Architectural blueprints—the "4+ 1" view model of software architecture. *Tutorial Proceedings, Tri-Ada'95*, pages 540–555, 1995.

[23] C Raja Kumar and M Rajinikannan. A literature survey on various java application servers. 2014.

[24] Charlie Lai, Li Gong, Larry Koved, Anthony Nadalin, and Roland Schemers. User authentication and authorization in the java tm platform. In *Computer Security Applications Conference, 1999.(ACSAC'99) Proceedings. 15th Annual*, pages 285–290. IEEE, 1999.

[25] Vladimir Muhvich. Engrama, investigación sobre modelos de visualización morfológica. *Museo Nacional Centro de Arte Reina Sofía. Madrid*, pages 249–272, 2014.

[26] Andreas Papasalouros, Konstantinos Kanaris, and Konstantinos Kotis. Automatic generation of multiple choice questions from domain ontologies. In *e-Learning*, pages 427–434. Citeseer, 2008.

[27] Sebastian Rudolph. Foundations of description logics. In *Reasoning Web. Semantic Technologies for the Web of Data*, pages 76–136. Springer, 2011.

[28] Eliza Sachs. Getting started with protégé-frames. *Bmj Clinical Research Ed*, 307(6919):1619–1620, 2006.

[29] Graeme Stevenson and Simon Dobson. Sapphire: Generating java runtime artefacts from owl ontologies. In *Advanced Information Systems Engineering Workshops*, pages 425–436. Springer, 2011.

[30] Milorad Tosic and Marija Cubric. Semcq–protégé plugin for automatic ontology-driven multiple choice question tests generation. In *Procs of the 11th International Protege Conference*. Stanford Center for Biomedical Informatics Research, 2009.

[31] Max Völkel and York Sure. Rdfreactor-from ontologies to programmatic data access. In *Poster Proceedings of the Fourth International Semantic Web Conference, Galway, Ireland*, pages 55–60, 2005.

[32] Hai H Wang, Natasha Noy, Alan Rector, Mark Musen, Timothy Redmond, Daniel Rubin, Samson Tu, Tania Tudorache, Nick Drummond, Matthew Horridge, et al. Frames and owl side by side. In *Presentation Abstracts*, page 54. Citeseer, 2006.

[33] Moritz Weiten. *Ontostudio® as a ontology engineering environment*. Springer, 2009.

[34] Nicholas Wolterstorff. Toward an ontology of art works. *Nous*, pages 115–142, 1975.

Apéndice A

Manual de instalación y configuración

En este apéndice se presenta el proceso de instalación y configuración necesario para la implantación exitosa del prototipo en un ambiente de producción.

Programas a instalar

El lenguaje de programación utilizado para la implementación del prototipo fue java 8, por tanto es necesario tener instalada la maquina virtual de java versión 8 (JVM 8) para ejecutar la aplicación. El servidor de aplicaciones utilizado es Wildfly [1] versión 8.2.1 Final. Se utiliza como servidor de bases de datos a MySQL [2] versión 5.6.

Creación de la base de datos

Luego de instalar el software necesario, se debe crear la base de datos por intermedio del script de creación *TablasYRegistros.sql*.

Configuración de Wildfly

A continuación se explica como configurar el servidor wildfly. Se debe definir el datasource que la aplicación utiliza para conectarse a la base de datos a través del servidor de aplicaciones y se debe crear un módulo en Wildfly para que este se conecte a la base de datos MySQL. En la ruta `%WILDFLY_HOME%\modules\system\layers\base\com` se debe crear una carpeta llamada mysql, dentro de esta se crea

[1] Sitio web wildfly - http://wildfly.org/
[2] 3

otra llamada main, y finalmente dentro de main se guarda el jar de nombre *mysql-connector-java-5.0.8-bin.jar* (disponible en la carpeta de instalación de MySQL), este archivo es necesario para que el servidor se conecte al manejador de base de datos MySQL. Para finalizar con la creación del módulo, se deberá crear en la carpeta main mencionada anteriormente un archivo llamado module.xml con el contenido mostrado en el listado A.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.1" name="com.mysql">
  <resources>
    <resource-root path="mysql-connector-java-5.0.8-bin.jar"/>
  </resources>
  <dependencies>
    <module name="javax.api"/>
 </dependencies>
</module>
```

Listado A.1: contenido archivo module.xml.

Continuando con la configuración de wildfly, luego de crear el módulo de conexión al manejador de bases de datos, es posible definir el datasource. Se debe editar el fichero standalone.xml que se encuentra en `%WILDFLY_HOME%\standalone\configuration`, se debe agregar en el subsistema *urn:jboss:domain:datasources:2.0* la declaración del datasource mostrada en el listado A.2, donde *IPHOST* es la ip del host donde está instalado el manejador de base de datos MySQL y *PUERTO* es el puerto donde recibe las conexiones. Para finalizar con la configuración del datasource, se debe hacer referencia al modulo anteriormente creado, esto se realiza también en el archivo standalone.xml en el mismo subsistema, en la sección *drivers* se debe agregar el contenido mostrado en el listado A.3.

```
<datasource jndi-name="java:/MySQLOntologyDS" pool-name="MySQLOntologyDS" enabled="
    true" use-java-context="true">
                <connection-url>jdbc:mysql://IPHOST:PUERTO/ontologydb</connection-
                    url>
                <driver>mysql</driver>
                <security>
                    <user-name>MYSQL_USU</user-name>
                    <password>MYSQL_PASS</password>
                </security>
                <statement>
                    <prepared-statement-cache-size>100</prepared-statement-cache-
                        size>
                    <share-prepared-statements>true</share-prepared-statements>
                </statement>
            </datasource>
```

Listado A.2: Configuración de datasource para MySQL en standalone.xml.

```
<driver name="mysql" module="com.mysql">
  <driver-class>com.mysql.jdbc.Driver</driver-class>
</driver>
```

Listado A.3: referencia al modulo de MySQl

El prototipo utiliza JAAS (Java Authentication and Authorization Service) para el manejo de usuarios y sus privilegios. Para utilizarlo es necesario configurar el servidor de aplicaciones que se encuentra incluido en Wildfly, para esto se debe editar el fichero standalone.xml nuevamente, agregando en el subsitema *urn:jboss:domain:security:1.2* el contenido del listadoA.4.

```
 <security-domain name="secureDomain" cache-type="default">
  <authentication>
     <login-module code="Database" flag="required">
         <module-option name="dsJndiName" value="java:/MySQLOntologyDS"/>
         <module-option name="principalsQuery" value="select usupasswd from usuario
             where usunick=?"/>
         <module-option name="rolesQuery" value="select usurol, 'Roles' from
             usuariorol where usunick=?"/>
         <module-option name="hashAlgorithm" value="SHA-256"/>
         <module-option name="hashEncoding" value="base64"/>
     </login-module>
    </authentication>
 </security-domain>
```

Listado A.4: Configuración de JAAS en standalone.xml para reconocimiento de roles de usuario.

Configuración específica del prototipo

Desde el prototipo implementado se maneja un archivo de propiedades desde donde se indica el nombre de ontología a utilizar y la carpeta donde se encuentra dentro del servidor de aplicaciones, entre otras propiedades importantes. El archivo se llama *ontoforms.properties* y se debe ubicar en en la siguiente ruta `%WILDFLY_HOME%\standalone\configuration`. En este archivo se define la propiedad de nombre *ONTOLOGIA_FILE* que indica el nombre del archivo de la ontología utilizada, el valor es *percepciones_y_familias_de_arte.owl*. Por otro lado se debe indicar la ruta del directorio donde se aloja la ontología. La aplicación toma como directorio raíz, el directorio donde está instalado Wildfly, por este motivo se debe indicar la ruta de la carpeta donde se guarda la ontología dentro del servidor de aplicaciones. Se debe crear una carpeta en la ruta `%WILDFLY_HOME%` llamada *onto* donde se guarda la ontología, y se define en el archivo de propiedades *ontoforms.properties* la propiedad *ONTOLOGIA_PATH* con el valor *onto*.
Para realizar la clasificación, desde el prototipo se necesita saber qué clase de la ontología es la clase principal de la jerarquía sobre la que se clasificará, por este

motivo se definió la propiedad *CLASE_PRINCIPAL_CLASIFICACION* con el valor *Familia*, que corresponde a la clase principal de la jerarquía de familias de producción en la ontología desarrollada. La última propiedad es la definición del nombre de la aplicación, esta propiedad se llama *NOMBRE_APLICACION* y tiene el valor *Encuesta de Arte*.

Apéndice B

Manual de usuario para configuración

En este apéndice se presenta el manual de usuario para realizar una configuración, esta es la etapa previa que permite la generación de formularios automáticos para un grupo de usuarios. Se presentan los casos de uso involucrados con la configuración: la creación de una nueva configuración y el listado de las configuraciones.
La instancia de configuración es realizada por un usuario administrador. Es necesario tener configuraciones para poder completar los formularios.

B.1. Acceder a pantalla principal de configuraciones

Para poder ingresar a las funcionalidades de configuración el usuario tiene que haber ingresado con una cuenta con rol administrador. Se debe acceder a la pantalla de nombre *Configuraciones* por medio del botón en el panel lateral de nombre *Configuración*. En la Figura B.1 se muestra una impresión de la pantalla de *Configuraciones*, se encuentra una tabla en la que se muestran datos de las configuraciones creadas hasta el momento, en orden de izquierda a derecha son:

- **Identificador** de la configuración en la base de datos.
- **Concepto principal** a ingresar a la ontología.
- **Rol**: grupo de usuarios a los que se les asigna la configuración.
- **Fecha creación**: la fecha de creación de la configuración.
- **Activo:** Indica si la configuración se encuentra activa.
- **Identificación manual**: Indica si en la configuración se indicará el nombre de la instancia nueva o dicha acción se hará en forma automática.

- **Acciones**: Acciones disponibles sobre las configuraciones ya creadas. Las acciones disponibles son las de activar la configuración y la de eliminar la configuración.

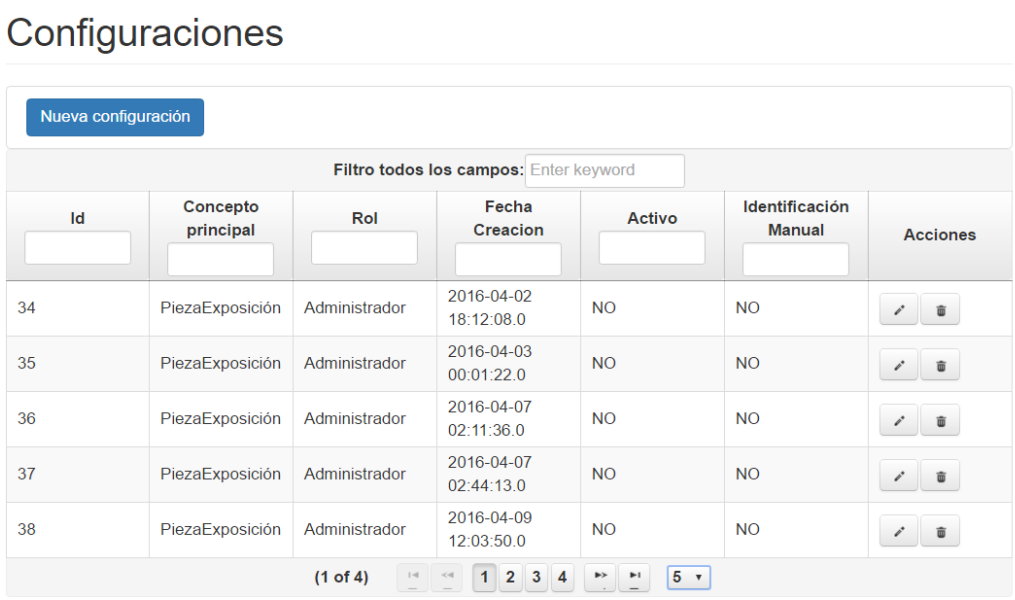

Figura B.1: Pantalla principal de configuraciones. Se muestra una tabla con las configuraciones existentes.

B.2. Acciones sobre las configuraciones ya creadas

En esta sección se presentan las posibles acciones que se pueden hacer con configuraciones ya creadas. Estas acciones son activar la configuración y eliminarla.

El sistema permite tener varias configuraciones asignadas a un solo rol, sin embargo solo es posible tener una única configuración activa para un determinado rol. Por este motivo al activar una configuración, si existe otra ya activada asociada al mismo rol, se desactiva esta ultima. Luego de activar la configuración se generarán formularios a partir de la nueva configuración activada para los usuarios que corresponda

La otra acción posible es eliminar una configuración ya creada, al oprimir el botón de eliminar y confirmar, el sistema elimina la configuración correspondiente.

B.3. Crear configuración

En esta sección se muestra el proceso de crear una configuración. Se accede a esta funcionalidad presionando el botón *Nueva Configuración* que se encuentra en la pantalla principal de configuraciones.
El proceso de crear una configuración consiste en 4 pasos. (i) selección del concepto principal del formulario a generar, (ii) elegir los temas a tratar en el formulario, (iii) seleccionar el orden de los temas en el formulario y (iv) seleccionar otros datos relevantes a ser incluidos en el formulario. Es altamente recomendable que esta funcionalidad sea realizada por usuarios que tengan conocimiento de la realidad modelada en la ontología y de su estructura.

B.3.1. Selección Concepto Objetivo

El concepto objetivo es el concepto al que se le creará el formulario, y por medio del formulario se ingresarán nuevos individuos pertenecientes a la clase del concepto objetivo y sus datos asociados en la ontología. En la Figura B.2 se muestra la pantalla correspondiente a este paso, se listan todos los conceptos de la ontología. Mediante el *Filtro por nombre* se puede reducir la cantidad de conceptos listados a medida que se escribe el nombre del concepto que se quiere seleccionar. En esta pantalla también se debe seleccionar el rol para la configuración y si el individuo resultado de la encuesta será nombrado por el usuario o se le asignará un nombre de forma automática. Para pasar al siguiente paso de la configuración se debe oprimir el botón *Siguiente Paso*.

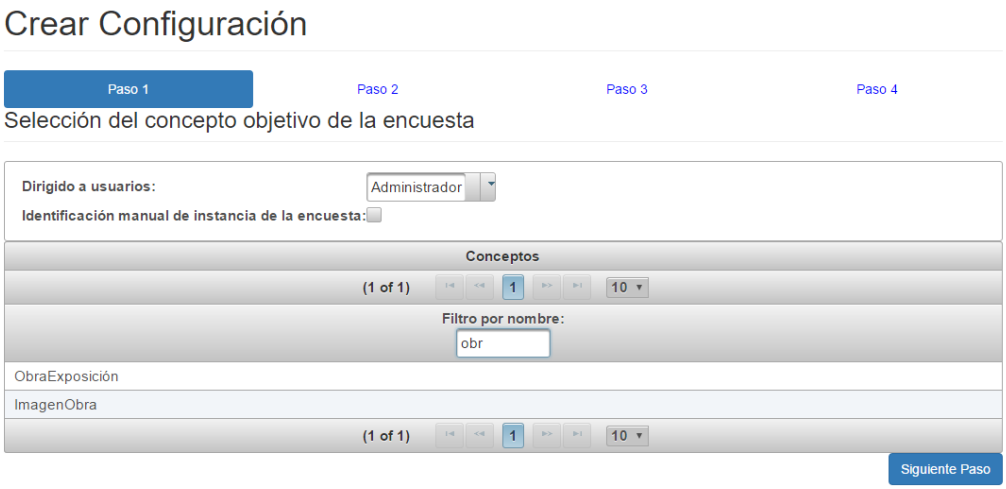

Figura B.2: Pantalla paso selección concepto objetivo

B.3.2. Selección de temas

En este paso el sistema lista todos los conceptos que pertenecen al recorrido de alguna propiedad de objeto que tiene como dominio al concepto objetivo seleccionado en el paso anterior (i.e., se listan las anclas).
El usuario deberá seleccionar los conceptos que luego serán temas del formulario. En la Figura B.3 se muestra la lista de los conceptos a seleccionar, a su vez se cuenta con un filtro llamado *Concepto nombre* para filtrar esta lista de conceptos a seleccionar. Luego de seleccionar las anclas, se debe presionar el botón *Siguiente paso* para continuar al paso 3.

Figura B.3: Pantalla paso selección temas

B.3.3. Orden de los temas

En este paso se muestran todos los conceptos seleccionados en el paso anterior, se podrá ordenar la lista de la tabla simplemente seleccionando el concepto y arrastrándolo a la posición que se quiera (en forma vertical). Los temas del formulario generado se mostrarán en el orden seleccionado en este paso. Además, se indica qué temas de la lista son para formularios de tipo ingreso de datos o de múltiple opción. Marcando con un *tick* en la tabla, se indica que para ese tema se debe generar un formulario de ingreso de datos. En la Figura B.4 se muestra el orden de los temas, y además que los temas de *Exposición* y *Pieza* serán tratados como formularios de ingreso de datos.

Figura B.4: Pantalla paso orden de temas

B.3.4. Selección de datos relevantes y confirmación

En este último paso se muestran las propiedades asociadas al concepto objetivo seleccionado. El usuario debe elegir las propiedades que serán incluidas al formulario. Estas propiedades seleccionadas serán agrupadas como un nuevo tema en el formulario, este tendrá el nombre de *Otros Datos* y será incluido en el final de los temas. En la Figura B.5 se muestran tres propiedades asociadas al concepto objetivo seleccionado en el paso 1. En este paso es posible confirmar la configuración oprimiendo el botón *Confirmar Configuración*. Al confirmar se despliega un cartel, consultando si se quiere dejar la configuración activada, en la Figura B.6 se muestra esta situación.

Figura B.5: Pantalla paso seleccion de otros datos relevantes y confirmación

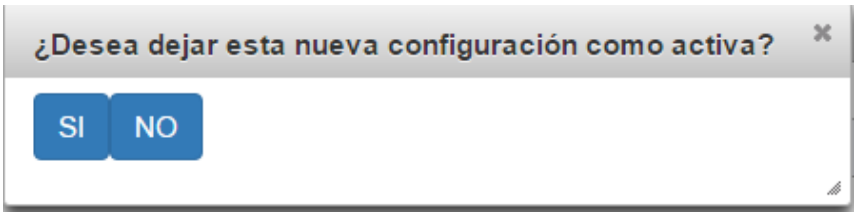

Figura B.6: Consulta de creación de configuración activada o desactivada

Apéndice C

Imágenes de la aplicación

Figura C.1: Consulta usuarios

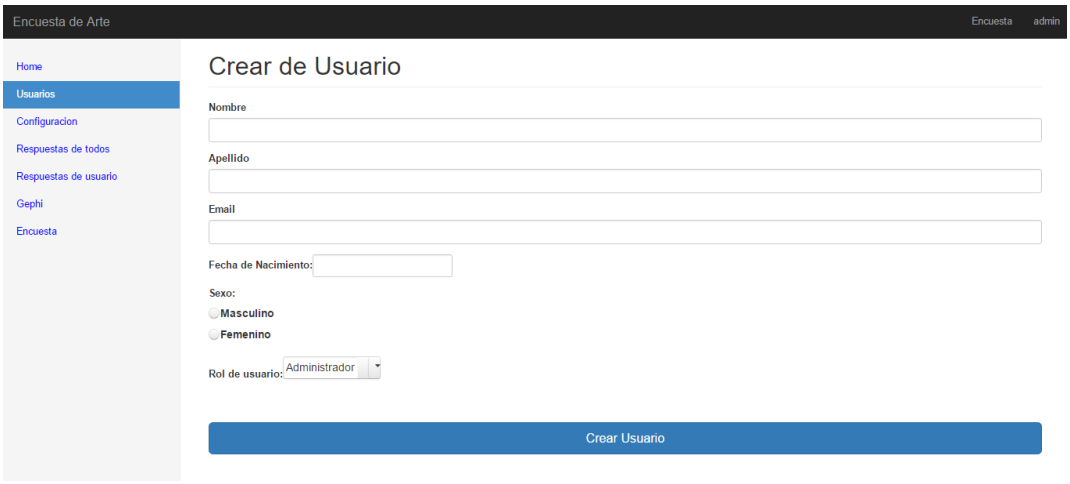

Figura C.2: Crear Usuario

Figura C.3: Respuestas Encuestas

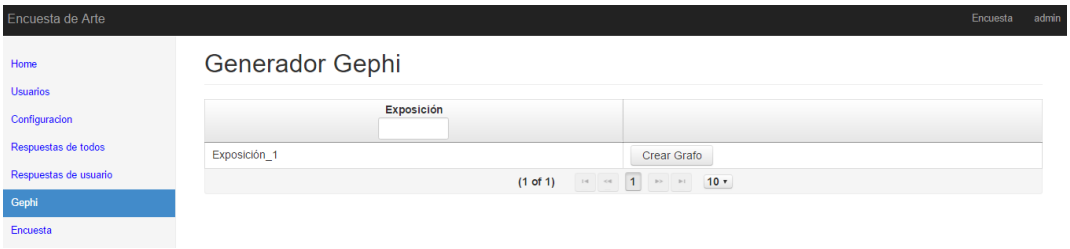

Figura C.4: Generador grafo Gephi

Figura C.5: Botonera del formulario, sin etapas por completar

Figura C.6: Botonera del formulario, con todas las completadas