UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA
ESPAÑA

TRABAJO DE FIN DE MÁSTER
Máster Universitario en IA Avanzada: fundamentos, métodos y aplicaciones. Especialidad de Sistemas inteligentes de diagnóstico, planificación y control

Desarrollo de una ontología para Smart Home

Alumno: Susana Mª Gutiérrez Caballero
Directores: José Luis Fernández Vindel (UNED)
Mari Carmen Suárez de Figueroa Baonza (UPM)

Valladolid, 5 de junio de 2012

Desarrollo de una ontología para Smart Home

Susana Mª Gutiérrez Caballero

5 de junio de 2012

Agradecimientos

Muchas veces no nos damos cuenta de la cantidad de gente que nos rodea y contribuye a nuestra felicidad, o a hacernos las cosas más fáciles en el día a día.

Pero cuando llega el momento de escribir páginas como ésta es cuando reflexionas y te percatas de que eres muy afortunada, porque tienes a tu lado personas que te ayudan, te apoyan, te quieren, que logran hacer del mundo un lugar mejor, sobre todo en los momentos en que a uno mismo no se lo parece tanto. Pues bien, para todos ellos escribo estas líneas.

En primer lugar quiero agradecer a José Luis, mi tutor en este trabajo, su confianza y ayuda, ya que desde el principio se ha mostrado afable e interesado en este proyecto.

Por otra parte, quiero dar las gracias a Óscar Corcho y Mari Carmen Suárez, del "Ontology Engineering Group" de la Universidad Politécnica de Madrid, por guiar mis pasos en los comienzos de este trabajo, y resolver las dudas que, antes o después, acaban surgiendo en este tipo de proyectos.

No puedo dejar de mencionar a mis compañeros de trabajo, sobre todo a Cris y a Álvaro, por su labor de "expertos" durante la captura de requisitos, una de las fases más importantes en un proyecto como éste.

Cómo olvidar a mis padres y mi hermana, siempre ahí, a mi lado, apoyándome y haciendo que todo sea más llevadero.

Y por último, y no por ello menos importante: gracias, gracias, gracias y gracias a tí, Arturo, mi vida, mi apoyo, mi sol, mi luna, mi todo, por estar siempre a mi lado, dándome ánimos para intentar cerrar esta etapa de mi vida antes de la llegada de quien será (bueno, ya lo es) nuestro tesoro más preciado dentro muy poquito tiempo. Mi agradecimiento para tí debería estar escrito en mayúsculas, negrita y subrayado, ya que sin tí ésto no hubiera sido posible.

Para Arturo, que me da la vida,
pero también para Gonzalo que,
aunque esté de camino,
seguro que me ha dado fuerzas
para terminar este trabajo
antes de su llegada.

Resumen

Actualmente, momento en el que cada vez se habla más del concepto de hogar inteligente (o "smart home"), se hace imprescindible el crear en dichos hogares una infraestructura de computación ubicua que permita desplegar sobre ella servicios de alto nivel que, teniendo siempre en cuenta el confort de los usuarios, pueden realizar, entre otras, ciertas labores relacionadas con la eficiencia energética. Dentro de esta infraestructura de computación ubicua, uno de los elementos más importantes es la ontología que facilite la interoperabilidad y la adquisición e inferencia de conocimiento.

El *objetivo* principal de este trabajo es desarrollar una ontología que incorpore también información relativa a la parte energética de un hogar digital, considerando también la posibilidad de que, utilizando la información contenida en la ontología, puedan realizarse ciertas labores de gestión de cargas eléctricas.

Aunque desde un primer momento se ha intentado que la ontología construida sea muy completa, siempre se ha tenido en mente el que sea fácilmente ampliable, ya que en un futuro puede que sea necesario realizar algún tipo de ampliación o modificación sobre ella. Este aspecto ha influido en la *metodología* a utilizar durante el desarrollo de la ontología, y por ello se han estudiado las posibles metodologías de aplicación en este trabajo, habiendo adoptado finalmente la metodología NeOn.

Respecto a la *estructura de este documento*, decir que comienza con un planteamiento del ámbito del problema, la situación actual y el contexto que los rodea, finalizando este primer capítulo con un enunciado claro de los objetivos planteados al comienzo del trabajo.

A continuación se presenta la revisión de la bibliografía realizada, tan necesaria para poder conocer lo que se ha hecho hasta el momento en este campo y poder definir claramente lo que se pretende aportar sobre lo que ya existe.

El capítulo siguiente aborda el estudio de metodologías realizado, exponiendo también cuáles han sido los motivos que han llevado a seleccionar la metodología NeOn como la más adecuada para el caso que nos ocupa.

Una vez descritos todos los capítulos anteriores, se presenta el capítulo dedicado a la construcción de la ontología, abordando en él todas las actividades que ha sido necesario realizar, siempre teniendo en cuenta la metodología a seguir.

Para terminar, se exponen las conclusiones extraídas de la realización de este trabajo, seguidas de una breve descripción de los posibles trabajos futuros a realizar.

Índice general

Índice de tablas

Índice de figuras

Capítulo 1

PLANTEAMIENTO Y OBJETIVOS

Este capítulo presenta los conceptos clave que es necesario conocer para poder entender perfectamente tanto el resto del documento como el alcance del trabajo realizado, terminando con un enunciado claro de los objetivos planteados al comienzo del trabajo.

1.1. Hogares Inteligentes: tendencias

En los tiempos que corren, donde cada vez se habla más de los hogares inteligentes, el concepto de '**smart home**' se está haciendo más y más popular, ya que hemos adoptado como cotidianos, e incluso como una necesidad, los nuevos avances tecnológicos que han surgido (y están surgiendo) en el ámbito del hogar.

Ha cambiado la forma en que nos comunicamos con los demás, la forma de buscar información, etc., pero también la forma en que interactuamos con nuestros hogares (y con los dispositivos que hay en ellos). Además, las necesidades de entretenimiento y confort en los hogares han evolucionado de forma sorprendente, buscando en todo momento una mejor calidad de vida en cada instante que pasamos en ellos. Es aquí donde aparece el concepto de 'smart home'/hogar digital.

El concepto de 'smart home' engloba múltiples aspectos, siendo uno de los principales el de la **integración** de todos y cada uno de los sistemas (e incluso redes) que están presentes en dicho hogar (iluminación, climatización,

red de datos, red domótica, red multimedia, etc.). Esta integración permitirá al usuario acceder a todos y cada uno de los sistemas de la misma forma, pudiendo así controlarlos de forma integral, automática y accesible para el usuario. En definitiva, lo que se pretende es integrar todos los equipos, redes, sistemas, etc. presentes en el hogar, para que así puedan controlarse de forma integral. Pero no sólo se trata de posibilitar esta integración, sino también de incrementar, y tener siempre en mente, el **confort** del usuario. Por otra parte, conviene tener muy en cuenta la **eficiencia energética**, concepto muy relacionado con el de '**Smart Grid**', redes diseñadas con el fin de optimizar la producción y distribución de electricidad, y de esta forma equilibrar mejor la oferta y la demanda entre productores y consumidores.

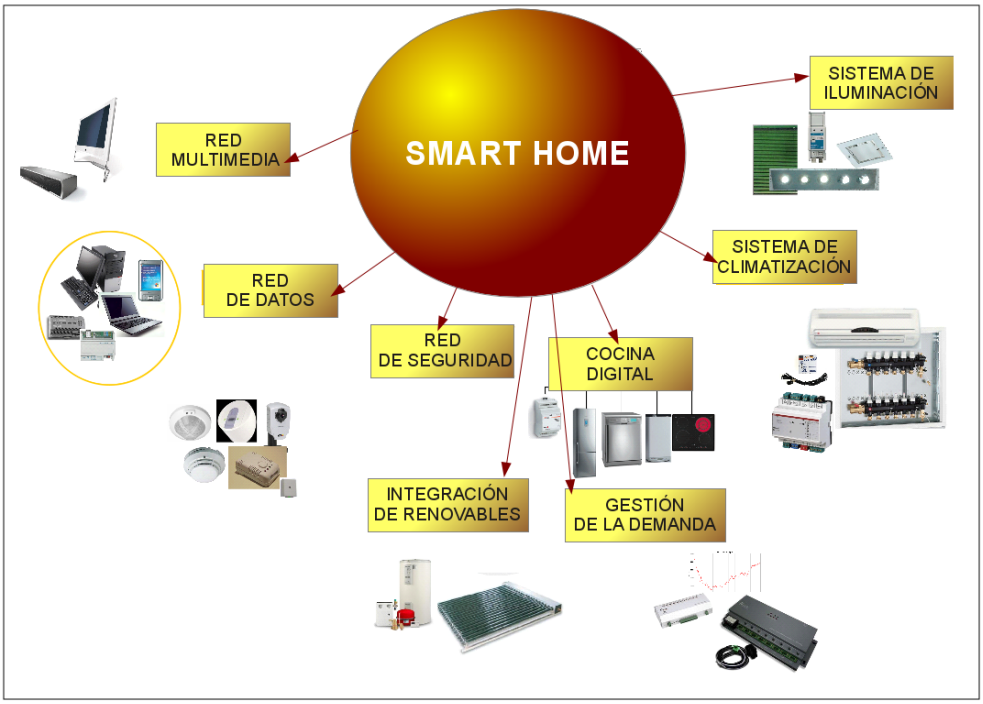

Figura 1.1: Principales sistemas presentes en un Smart Home

Como ha podido comprobarse, el concepto de 'smart home' no sólo está relacionado con el hogar y lo que pueda haber en él, sino que también es necesario poseer cierto conocimiento sobre los **usuarios** que habitan dicho hogar. Para lograr desplegar una tecnología centrada en el usuario en

hogares reales, es preciso crear antes una infraestructura de **computación ubicua** que permita a todos los servicios de alto nivel ser conscientes, en cada instante, del **contexto** que se está desarrollando en se momento en el hogar inteligente en cuestión.

La infraestructura de computación mencionada anteriormente engloba, habitualmente, tres componentes:

- *capa de sensorización*
- *middleware*
- *capa de aplicación* [1].

La capa de sensorización obtiene información procedente del entorno utilizando para ello diversos dispositivos, recogiendo también información relativa al usuario (estado actual, preferencias, actividades, etc.). El middleware almacena y analiza esta información "en bruto", obteniendo así información de contexto, de más alto nivel que la anterior. Finalmente, la capa de aplicación es la encargada de proporcionar al usuario servicios de alto valor añadido, siempre en función de la información relativa al contexto.

Uno de los elementos clave para el desarrollo de este tipo de sistemas (en adelante *sistemas context-aware*) son las ontologías, que permiten la comunicación entre los diversos servicios, así como el establecer una representación única para los conceptos que han de ser manejados por dichos servicios.

1.2. Ontologías: necesidad, aportaciones y conceptos básicos

La palabra ontología procede del campos de la filosofía, donde se utiliza para designar una explicación sistemática de la Existencia [2]. Una de las primeras definiciones de ontología proporcionadas en el ámbito de la Inteligencia Artificial fue la proporcionada por Neches et al. en el año 1991 [3], según la cual una ontología define los términos y relaciones básicos relacionados con el vocabulario de un determinado dominio, englobando también las relaciones y reglas de combinación de términos que pueden utilizarse para extender dicho vocabulario. Posteriormente, Gruber proporcionó otra definición, afirmando que una ontología es una especificación explícita de una conceptualización [4]. Más tarde, Borst modificó ligeramente la definición de

Gruber, añadiéndole el concepto de compartición, y afirmando así que una ontología es una especificación formal de una conceptualización compartida [5].

Como ha podido comprobarse, desde la primera definición de ontología múltiples han sido los autores que han proporcionado definiciones de este concepto en el ámbito de la Inteligencia Artificial.

El contexto puede definirse como una representación semántica de cierto conocimiento sobre el mundo real en un formato entendible por las computadoras. Una representación basada en ontologías para el modelado del contexto permitirá describir contextos semánticamente, y compartir dicho conocimiento entre usuarios, dispositivos y servicios. Schilit clasifica el contexto en tres categorías:

- *contexto computacional*
- *contexto del usuario*
- *contexto físico* [6].

El contexto computacional se refiere, por ejemplo, a las redes disponibles, costes asociados, ancho de banda, dispositivos cercanos (impresoras, pantallas, etc.). Por otra parte, el contexto del usuario está relacionado con el perfil del usuario, localización, personas que están cera, etc. Por último, el contexto físico hace referencia al nivel de luz, nivel de ruido, temperatura, etc. Este contexto está muy relacionado con la capa de sensorización mencionada anteriormente.

Las ontologías permiten establecer un marco de entendimiento común sobre un determinado dominio, y son muy útiles para compartir y reutilizar conocimiento en contextos de agentes inteligentes, sistemas multi-agente, context-awareness, web semántica, etc.

Las ontologías se componen de [7]:

- *Conceptos/Clases*: un concepto es cualquier entidad sobre la que puede decirse algo. Pueden ser abstractos o concretos, simples o compuestos, reales o ficticios, etc.
- *Relaciones*: representan las interacciones entre los elementos del dominio. Se dice que una relación es de tipo funcional (es una función) cuando una instancia sólo puede estar presente en una relación de este tipo. La mayor parte de los lenguajes ontológicos agrupan las relaciones dentro de las propiedades de los objetos.

- *Instancias y Objetos*: representan los objetos en la realidad (objetos concretos de la realidad).

- *Propiedades*: características que pueden tener las instancias, siempre dentro del dominio objetivo. También se conocen como slots o atributos.

Uno ontología se expresará en un lenguaje basado en sistemas lógicos (lenguaje ontológico), que permitirá almacenar el conocimiento sobre el dominio objetivo.

Gracias al carácter formal de una ontología, sobre ella pueden llevarse a cabo procesos de adquisición e inferencia de conocimiento. De hecho, es posible extraer información que no se muestra explícitamente, pero que está representada de forma implícita. Esta información podrá ser añadida a la ontología, enriqueciéndola enormemente (siempre que dicha información sea útil para el desarrollo que se está realizando).

No debe confundirse lo mencionado en el párrafo anterior con los motores de inferencia que pueden trabajar sobre una ontología, los cuales realizan cierto tratamiento sobre la información representada en ella para obtener conocimiento adicional (que no estaba representado en ella ni implícita ni explícitamente).

1.2.1. Principales aplicaciones de las ontologías

Las principales funciones que puede desempeñar una ontología son, entre otras, las siguientes:

- Pueden funcionar como repositorios para la organización de conocimientos e información

- Son susceptibles de ser utilizadas como herramienta para la adquisición de información en situaciones en las que un equipo de trabajo la utiliza como soporte para la organización del dominio.

- Pueden utilizarse como herramienta de referencia en la construcción de sistemas basados en conocimiento, incrementando y favoreciendo la interoperabilidad de y entre los mismos.

- Permiten la reutilización de conocimiento ya existente en las nuevas aplicaciones que se construyan dentro de un dominio concreto.

- Sirven como base para la construcción de lenguajes de representación de conocimiento.
- Permiten mejorar la búsqueda de información en la Web y en las intranets de las organizaciones.
- Pueden utilizarse para comprobar la validez de los datos, por ejemplo para validar los datos procedentes de bases de datos.
- Para favorecer la interoperabilidad semántica.
- Por otra parte, resultan muy útiles para facilitar el razonamiento automático. Partiendo de un conjunto de reglas de inferencia, un motor de razonamiento puede utilizar la información contenida en una ontología para inferir conclusiones sobre dicha información.

1.2.2. Lenguajes ontológicos

Tal es la cantidad de lenguajes ontológicos existentes, que para citarlos primero conviene clasificarlos, y así agruparlos en base a un determinado criterio. Una posible clasificación es la propuesta en [8, 7]:

- *Lenguajes para representaciones gráficas/Graphical notations*: no se basan en ningún tipo de lógica, sino únicamente en la sintaxis de las ontologías (clases, instancias y propiedades). Aunque no proporcionan la posibilidad de realizar inferencia, sí que son la base para muchos de los lenguajes ontológicos existentes, y también para representar ontologías gráficamente. Dentro de este grupo podemos encontrar:
  - *redes semánticas*: una red semántica es un conjunto de nodos conectados por flechas.
  - *mapas de conceptos*: son una ampliación de las redes semánticas que incluye el concepto de ocurrencias (referencias a fuentes externas al mapa de concepto).
  - *UML/Unified Modeling Language*: como lenguaje que permite representar objetos y relaciones entre ellos, puede utilizarse también para representar gráficamente una ontología.
  - *RDF/Resource Description Framework/Resource Description Framework*:

Aunque todos los lenguajes de descripción de ontologías son el resultado de un proceso de adaptación del lenguaje XML [9] para tal fin, cada uno tiene sus características particulares. A continuación se presentan los más conocidos:

- *RDF* (Resource Description Framework) [10]: basado también en XML, y orientado a la utilización de metadatos para describir recursos en la Web.
- *OWL* (Ontology Web Language) [11]: familia de lenguajes de representación de conocimiento para ontologías, todos ellos desarrollados por el W3C [1]. OWL incluye tres sub-lenguajes:
  - *OWL-Lite*: a grandes rasgos, es lo mismo que RDFS pero contemplando también la cardinalidad 0/1. Orientado a la representación de jerarquías de clasificación y restricciones simples.
  - *OWL DL*: contiene los constructores del lenguaje, pero con restricciones de jerarquía. Proporciona completitud computacional y decidibilidad.
  - *OWL Full*: está compuesto por el vocabulario completo interpretado más extensamente que en OWL DL. Este lenguaje incorpora el máximo poder expresivo, así como libertad sintáctica, pero no ofrece ningún tipo de garantía desde el punto de vista computacional.

Si bien es verdad que existen, como se ha podido comprobar, multitud de lenguajes con los que poder describir una ontología, son RDF y OWL los más conocidos y utilizados por la comunidad de desarrolladores. En este caso se ha decidido utilizar OWL para describir la ontología desarrollada.

1.3. Objetivos específicos del trabajo

Como ya se ha mencionado en apartados anteriores, cada vez está más extendido el concepto de "smart home"/"hogar digital", dentro del cual se despliegan tecnologías centradas en el usuario y sumamente relacionadas con la eficiencia energética, aunque sin olvidar nunca el confort de los habitantes de la casa.

[1] `http://www.w3c.es/`

Pero para desplegar este tipo de tecnologías y servicios es preciso contar con una infraestructura de computación ubicua que facilite a los servicios de alto nivel toda la información que necesiten para poder realizar sus funciones correctamente. Es en este ámbito es donde surge la necesidad de la creación de una ontología de hogar digital que contemple todos estos aspectos, y que permita además la interoperabilidad entre los diferentes servicios que puedan cooperar en un mismo hogar digital. Sin la existencia de esta posible ontología, la cooperación e interconexión de servicios de alto nivel (sobre todo los desarrollados en diferentes instantes de tiempo, o por diferentes grupos de trabajo) se hace muy complicada.

Por otra parte, la existencia de una ontología permite la realización de procesos de adquisición e inferencia de conocimiento, tan necesarios en este tipo de dominios.

Además de constituir un repositorio de información para la organización de conocimientos y la adquisición de nueva información, extraída de la operación del sistema, una ontología resulta muy útil en este ámbito para facilitar el razonamiento automático, lo cual aporta un valor añadido muy importante a los posibles servicios de alto nivel a desarrollar en un hogar digital.

El contemplar en la ontología conceptos relacionados con la energía permitirá vigilar los niveles de consumo energético del hogar, posibilitando una posible reducción de los mismos. Hoy en día, cuando entre un 60 % y un 80 % del consumo de energía se concentra en las ciudades, este tipo de acciones son muy necesarias para poder alcanzar dicha reducción del consumo energético en los hogares y de este modo mejorar la sostenibilidad del planeta.

Por ello, se plantean como objetivos principales de este trabajo:

- Como objetivo principal se propone realizar una contribución al diseño de Smart Homes desarrollando una ontología de hogar digital que, además de tener en cuenta los conceptos e información que se han venido contemplando hasta ahora en este tipo de ontologías, tenga en cuenta la mencionada parte energética de dicho hogar, contemplando también la tan necesaria integración con los diferentes modelos que, de un edificio, se van generando a lo largo del ciclo de vida del mismo. Para lograr desarrollar con éxito esta ontología, es muy importante la realización de un estudio de la bibliografía existente en lo concerniente a ontologías de hogar digital, pero también la obtención de información sobre la parte energética de un hogar digital, así co-

mo sobre la interoperabilidad entre los diferentes modelos de un edificio que se van generando a lo largo del ciclo de vida del mismo. El estado del arte en lo relativo a otras ontologías de hogar digital existentes permitirá plantear, en caso de ser posible, la reutilización de ciertos recursos ontológicos para construir la ontología final. No se trata de inventar la rueda, por lo que si ya existe una ontología que contempla una parte de lo que se pretende tener en cuenta en la ontología a construir en este caso, lo más conveniente es reutilizarla, añadiéndole después todo lo que se considere necesario.

- Aunque se pretende que la ontología desarrollada sea lo más completa posible, se propone como objetivo que ésta sea fácilmente ampliable, por lo cual se ha de prestar especial atención a la metodología a emplear para el desarrollo de la misma. Por ello, otro objetivo primordial es estudiar las metodologías disponibles para el desarrollo de ontologías y decidir cuál es la más idónea para ser utilizada en el ámbito de este trabajo. Se buscará también que la metodología en cuestión contemple ampliamente las tareas de captura de requisitos y reutilización de recursos ontológicos, muy importantes en este tipo de desarrollos.

Capítulo 2

REVISIÓN DE TRABAJOS PREVIOS

Actualmente, momento en el que el sistema energético se caracteriza por su ineficiencia y múltiples efectos contaminantes, se hace muy necesario vigilar exhaustivamente los niveles de consumo, especialmente en las ciudades, donde se concentra entre el 60 % y el 80 % del consumo de energía, y el 75 % de las emisiones de CO_2 [12].

Por una parte, cada individuo debería vigilar el consumo energético que se realiza en su hogar, e intentar reducirlo en la medida de lo posible. Pero, por otra parte, conviene tener en cuenta que las Smart Grids (redes de distribución eléctrica inteligentes) están a la orden del día. Estas Smart Grids se están diseñando para intentar optimizar la producción y distribución de electricidad, con el fin de equilibrar mejor la oferta y la demanda entre productores y consumidores. En un futuro puede que, siempre que el usuario lo permita, los contadores inteligentes sean capaces de recibir diversas consignas de actuación por parte de la suministradora de electricidad. Por ejemplo, podrían intentar gestionarse las cargas eléctricas de forma más eficiente, posponiendo las tareas que sea posible posponer si éstas se intentan ejecutar en horas de máxima demanda (así, los usuarios podrían tener, por ejemplo, un descuento en su factura de electricidad si deciden hacer caso de estas consignas, y poner la lavadora o el lavavajillas en momentos del día en que la curva de demanda no sea excesivamente elevada).

Pero para poder realizar este tipo de actuaciones dentro del hogar, es necesario añadir una parte energética a la ontología de 'Smart Home', para que los

futuros servicios energéticos que pudieran desplegarse tengan a su disposición la información energética necesaria para poder realizar sus funciones correctamente.

Como puede comprobarse en [13], este tipo de iniciativas cuentan actualmente con multitud de apoyos, tanto en el ámbito nacional como en el ámbito europeo.

Dado que actualmente se están realizando muchos esfuerzos para que la interoperabilidad sea total entre los diferentes modelos que se poseen de un edificio a lo largo del ciclo de vida del mismo, se propone además integrar de alguna forma la ontología a desarrollar con alguno de los **modelos de datos EEB** (Energy Efficiency in Buildings) estándar que están disponibles en la actualidad. A día de hoy existen dos modelos principales: **IFC** (Industry Foundation Classes) [14] y **gbXML** (Green Building XML) [15]. En este caso se ha decidido integrar la ontología con gbXML, al igual que se decidió en su momento en el ámbito de uno de los proyectos más importantes que trabajan con este tipo de modelos (el proyecto Fiemser[16]). Las principales razones que llevaron a los participantes en el proyecto **Fiemser** a seleccionar gbXML es que se trata de un modelo más simple y fácil de usar que IFC. El uso de XML proporciona flexibilidad y extensibilidad, y los datos pueden ser fácilmente procesados utilizando parseadores XML. Además, gbXML se integra con la mayor parte de las herramientas CAD, de diseño y simulación (REVIT, SketchUp, etc.). Comparado con IFC, la única limitación de gbXML es relativa a la geometría, pero en el caso que nos ocupa (recordemos que este trabajo está centrado en el concepto de hogar digital) ésto no es un problema, ya que las características geométricas de un hogar no adquieren nunca un alto nivel de complejidad.

GbXML ha sido desarrollado por Autodesk, por lo que el objetivo principal de los desarrolladores fue proporcionar interoperabilidad entre el paquete de programas que desarrolla el mencionado grupo empresarial. Ésto obliga a emplear varios de sus programas para convertir el modelo del edificio a un formato soportable en los posibles paquetes de simulación que deseen emplearse (si éstos no son propietarios de Autodesk, como pueden ser por ejemplo Energy Plus o TRNSYS) para convertirlo a un formato soportable en dichos paquetes de simulación. El proceso consiste en generar un archivo en formato IDF utilizando software de Autodesk (Ecotec o Green Building Studio). Este archivo ya podrá abrirse con los paquetes de simulación "alternativos" a Autodesk, por lo que queda solventados de este modo los

problemas de compatibilidad. Aunque a día de hoy puede ocurrir que este proceso traiga consigo algunos problemas en ciertos casos, cabe esperar que éstos sean solventados en un breve espacio de tiempo.

Uno de los proyectos europeos que más está trabajando en lo relativo a esta interoperabilidad entre los diferentes modelos generados durante el ciclo de vida de un edificio es el proyecto Fiemser [16], cuyos principales objetivos son disminuir el consumo energético y producir más energía, desarrollando también una plataforma que permita la gestión de los servicios asociados a todas las acciones que se planteen tomar al respecto durante el desarrollo del proyecto.

2.0.1. Trabajos anteriores

En esta sección se realizará una revisión de los trabajos más recientes que se han desarrollado en el campo de las ontologías para hogar digital que se han venido desarrollando hasta el momento.

Aunque varios sistemas desarrollados anteriormente, como pueden ser **Cooltown** [17] o **Context Toolkit** [18], sí que presentaron progresos en diferentes aspectos de computación ubicua, constituían únicamente un leve avance en lo relativo a compartición del conocimiento y razonamiento basado en el contexto. Una de las principales causas es que no estaban basados en ontologías, por lo que carecían de cualquier forma de representación semántica del conocimiento, no utilizando tampoco la información espacial para razonar sobre el contexto. Por otra parte, estos sistemas tampoco incluían información sobre la parte energética, que también se considera fundamental en el trabajo que se está presentando en esta memoria.

Se revisarán a continuación algunos trabajos que sí han desarrollado ontologías en el ámbito de Smart Home (unos añadiéndole también información contextual, y otros no).

Ya en el año 2004, **Wang et al.** [19] dieron a conocer un modelo de ontología basada en el contexto, al que añadieron también la parte de razonamiento. Se trataba de una ontología (**CONON**) desarrollada en OWL para modelar el contexto en entornos de computación ubicua. Los principales conceptos abordados en este trabajo son el concepto de entidad y varias subclases: localización, persona y actividad, pero no modela atributos, eventos ni situaciones en un contexto particular. Lo mismo ocurre con la ontología **SOUPA** [20], que es parte del

núcleo de CoBrA (Context Broker Architecture) y parecida a CONON, si bien es verdad que la representación espacio-temporal es bastante sofisticada.

Otro trabajo digno de mención es el realizado por **Gu et al.** [21] en el año 2004, tratándose también de un modelo del contexto basado en ontologías y de aplicación a todo tipo de ambientes inteligentes. Esta ontología no se centraba únicamente en el hogar, y por ello definía una serie de ontologías de dominio con la intención de contemplar también automóviles, oficinas, espacios abiertos, etc. El problema en este caso es que, al intentar abarcar tantos dominios, la ontología se vuelve más y más complicada, complicando también su utilización por parte de otros desarrolladores.

En 2008, Bonino y Corno [22] presentaron **DogOnt**, una ontología (bueno, en realidad un sistema completo) para el modelado del conocimiento en entornos domóticos inteligentes. Esta ontología soporta la descripción de entornos del hogar, realizando dicha descripción, en la mayoría de los casos, de forma totalmente independiente de dispositivos o redes, e incluyendo tanto elementos "controlables" como elementos arquitectónicos.

DogOnt fue diseñada teniendo en cuenta aspectos de interoperabilidad, integración e inteligencia en entornos domóticos, y contempla como elementos principales: Building Thing, Building Environment, State, Functionality y Domotic Network Component.

Esta ontología está modelada siguiendo tres ejes: tipología, funcionalidad y estado. Los aspectos dependientes de tecnologías específicas (protocolos domóticos específicos) se han modelado de forma independiente, separándolos del resto de entidades. Como se verá más adelante, esta ontología es una fuerte candidata a ser reutilizada para la construcción de la ontología planteada en este trabajo.

Por su parte, **Xu et al.** [23] también presentaron, en 2009, una ontología de smart home, pero ésta está más orientada a favorecer la composición de servicios, y no tanto a la parte relativa al contexto.

Conviene mencionar también la ontología de contexto presentada por **Nguyen** et al en 2010 [1]. Esta ontología, al igual que la mayor parte de las ontologías mencionadas, fue desarrollada utilizando RDF y OWL. Interesante el apunte que realizan sobre la cantidad y calidad de la información que se recoja, sobre todo de los sensores. El entorno

en un hogar digital cambia continuamente, por lo que la cantidad de información recogida puede incrementar exponencialmente en muy poco tiempo. Por lo tanto, es necesario utilizar ciertos mecanismos de "cribado" de los datos: desechar duplicados, establecer cuándo se considera que el entorno ha cambiado (establecer los niveles mínimos que tiene que cumplir el estado actual para considerarse diferente del estado anterior), etc.

Otra ontología muy interesante (aunque tampoco disponible para una posible reutilización) es la presentada por **Kofler et al.** en [24], y que está destinada a representar los diferentes parámetros energéticos disponibles en un hogar digital. En esta ontología se diferencian siete módulos principales (nombres tomados directamente de la bibliografía): ResourceInformation, BuildingInformation, ActorInformation, ProcessInformation, ExteriorInfluences, ComfortInformation y EnergyInformation. Cada una de estas partes corresponderían, por ejemplo, a: representación de los aspectos arquitectónicos del edificio; preferencias del usuario; posibles influencias del exterior (como por ejemplo la temperatura exterior), etc. La parte relativa a ResourceInformation hace describe las diferentes instalaciones y sus características (la mayor parte del hogar digital y su equipamiento ha de ser modelado en la base de conocimiento para que un posible sistema de control tenga disponible toda la información necesaria para poder realizar todas sus funciones). Con el término EnergyInformation pretende hacerse referencia a la demanda y el suministro de energía del hogar digital en cuestión. En este trabajo se realiza una descripción demasiado exhaustiva a tenor de quien escribe de algunos aspectos: cabe destacar, por ejemplo, la clasificación de los proveedores de energía, lo cual no tiene demasiada relevancia para un ámbito relativo al hogar digital.

Los entornos de hogar digital tienen sus propias características y, por ello, es necesario crear una ontología específica para dicho espacio específico: el 'Smart Home'. Muchos trabajos anteriores han realizado cierta implementación de razonamiento basado en el contexto, pero la mayor parte lo han hecho directamente a través del lenguaje de programación utilizado (por ejemplo, utilizando clases de Java), o lo han descrito ligeramente en la documentación del sistema. Pero para facilitar el intercambio de conocimiento lo que se necesita es definir una ontología que asiente las bases de dicho conocimiento, para que todos los servicios puedan comunicarse en base al mismo "idioma". Por otra parte, aunque se han mencionado sistemas que

sí han desarrollado una ontología incluyendo la información relativa al contexto y también una parte de razonamiento, muy pocos o ninguno son los sistemas que hasta ahora han incorporado a este tipo de ontologías la parte energética que se plantea añadir en el trabajo que se está presentando. En cualquier caso, el conocimiento adquirido de la observación de varias de estas ontologías han dotado al trabajo que nos ocupa de un valioso conocimiento, que se verá reflejado en la calidad de la ontología desarrollada.

Una característica común a la mayor parte de las ontologías presentadas es que hay un conjunto de entidades que siempre están presentes: actividad, localización y persona, añadiéndosele además en muchos casos la entidad dispositivo.

En vista de la revisión bibliográfica realizada, y como ya se ha comentado anteriormente, se plantea la ontología DogOnt como una posible candidata a ser reutilizada para la construcción de la ontología que se plantea en este trabajo. Si bien hay partes que habrá que eliminar, y que será necesario añadirle múltiples entidades, ésta es la ontología que, a priori, está más próxima a lo que se plantea en este trabajo.

Respecto al resto de ontologías estudiadas, el haber asimilado su estructura, conceptos y demás contribuirá a definir con mayor rigor la ontología que se pretende desarrollar en el ámbito de este trabajo. Como ya se ha mencionado anteriormente, prácticamente ninguna de las metodologías encontradas tienen en cuenta la parte energética (eficiencia energética) de un hogar digital, siendo éste uno de los aspectos que se considera más importante en el trabajo que nos ocupa.

Respecto a la revisión bibliográfica relacionada con la parte energética que se plantea en este trabajo, mencionar que **Han y Lim** [25] presentaron, en 2010, un sistema de gestión energética para hogar digital basado en los protocolos ZigBee e IEEE 802.15.4. Como puede comprobarse en el mencionado artículo, estos autores se centran en dos protocolos concretos, cosa que no se planteará en ningún momento en el trabajo que se presenta en esta memoria. En el caso de la la ontología en que se centra esta memoria, no se cerrará en ningún momento el desarrollo a ningún protocolo concreto, ya que se considera que la parte de protocolos concretos pertenece a un nivel más bajo y, por lo tanto, a nivel de ontología ya no se habla de protocolos, sino simplemente de información, proceda ésta de donde proceda.

Otro trabajo muy interesante (aunque su ontología tampoco está disponible) es el presentado por **Daouadji et al.** en [26], en el que se aborda el desarrollo de una ontología para redes de energía de bajas emisiones de carbono. Es de destacar la jerarquía de clases empleada en este caso para describir y categorizar los recursos energéticos TIC que consumen energía.

Capítulo 3

METODOLOGÍA

En este capítulo se presenta el estudio realizado sobre las diversas metodologías susceptibles de ser aplicadas para abordar el desarrollo de una ontología.

3.1. Metodologías para el desarrollo de ontologías

Son varias las metodologías que, a lo largo de los años, han ido surgiendo en relación al desarrollo de ontologías. Tras realizar una búsqueda a este respecto, cuatro fueron las metodologías identificadas para evaluar su posibilidad de aplicación en el trabajo que se presenta en esta memoria:

- *Methontology* [27]
- *On-To-Knowledge* [28]
- *DILIGENT* [29]
- *NeOn Methodology* [30]

Para la evaluación de estas metodologías se tendrá muy en cuenta (entre otros) el grado en el que éstas facilitan tanto el análisis de requisitos como a la reutilización de otras ontologías, pero también el modo en el que contribuyen a la consecución de ontologías fácilmente reutilizables y ampliables por parte de terceros. El planteamiento de estas necesidades obedece a la idea de que, a pesar de que la ontología a desarrollar en el ámbito de este

trabajo se desarrollará de la forma más completa posible, puede que en un futuro se detecte la necesidad de ampliar la misma. Por ello, y como ya se ha mencionado anteriormente, un objetivo primordial es el de proporcionar a la ontología de la claridad suficiente como para que alguien que en principio no está familiarizado con ella pueda asumirla fácilmente y realizar sobre ella todas las modificaciones o ampliaciones que considere oportunas.

A continuación se describen brevemente los aspectos más relevantes de cada una de estas cuatro metodologías, para luego realizar una comparativa crítica entre ellas y así poder decidir cuál es la elegida para su aplicación en el ámbito de este trabajo.

Methontology Esta metodología, desarrollada por el Ontology Engineering Group de la Universidad Politécnica de Madrid [31], está orientada a la construcción de ontologías a nivel de conocimiento, adoptando múltiples ideas y conceptos procedentes de la Ingeniería del software. De hecho, el proceso que define para el desarrollo de ontologías está basado en el estándar de desarrollo de software propuesto por la IEEE. Como muchas de las metodologías para el desarrollo de ontologías, Methontology diferencia entre el proceso de desarrollo de la ontología y el ciclo de vida a seguir durante dicho desarrollo.

El proceso de desarrollo de la ontología se refiere a las actividades que se llevarán a cabo durante la construcción de la ontología, mientras que el ciclo de vida (orientado en este caso hacia la construcción de prototipos) planifica cada una de las actividades definidas en el proceso de desarrollo de la ontología.

Como puede apreciarse en la tabla 3.1, Methontology no proporciona una clara definición sobre cómo abordar la parte de especificación de requisitos, y únicamente proporciona un simple listado de las actividades a llevar a cabo para abordar la reutilización de recursos ontológicos.

On-To-Knowledge El objetivo principal del proyecto On-To-Knowledge fue la aplicación de las ontologías a la información disponible en formato electrónico para mejorar la calidad de la gestión del conocimiento en grandes organizaciones. La ontología On-To-Knowledge fue desarrollada en el ámbito de este proyecto, acompañada de algunas herramientas orientadas a proporcionar un acceso más inteligente a grandes volúmenes de información.

La metodología On-To-Knowledge propone construir las ontologías teniendo en cuenta cuál para qué van a ser utilizadas, por lo que las ontologías

desarrolladas siguiendo esta metodología son extremadamente dependientes de la aplicación para la que, en un principio, van a utilizarse.

El modelo de ciclo de vida propuesto por esta metodología es incremental y cíclico, basado en el modelo de ciclo de vida de prototipado evolutivo.

Los procesos que propone esta metodología son los siguientes:

- Estudio de viabilidad: el estudio de viabilidad que se plantea en esta metodología está basado en el propuesto por la metodología CommonKADS.
- Lanzamiento: este proceso obtiene como salida el documento de especificación de requisitos. Se propone la utilización de preguntas de competencia, pero en ningún momento se proporcionan guías para la realización de esta actividad.
- Refinamiento: el objetivo en este caso es obtener una ontología que satisfaga los requisitos enunciados en el proceso anterior.
- Evaluación: el proceso de evaluación comprueba la usabilidad de las ontologías desarrolladas.
- Mantenimiento: es importante dejar claro desde el principio quién es el responsable del mantenimiento de la ontología, y cómo va a llevarse éste a cabo.

Como puede apreciarse en la tabla 3.1, On-To-Knowledge tampoco facilita una descripción clara de la parte de elicitación de requisitos, y únicamente proporciona recomendaciones para identificar las posibles ontologías a reutilizar. Por otra parte, no tiene en cuenta la posibilidad de desarrollar las ontologías de forma colaborativa. Además, la aplicación de esta metodología suele dar lugar a ontologías muy dependientes de la aplicación en la que va a ser utilizada en un primer momento, lo cual cierra las puertas a otros posibles usos que se le pudiese dar a las ontologías en cuestión.

DILIGENT

Esta metodología, desarrollada conjuntamente por el Instituto AIFB de la Universidad de Karlsruhe y el Instituto Superior Técnico de Lisboa, realiza especial hincapié en la parte colaborativa del proceso de desarrollo de la ontología, siendo una de sus principales preocupaciones el mantener un buen

registro de todos los posibles cambios que puedan surgir a lo largo de dicho proceso.

Las principales fases contempladas en el proceso de desarrollo de la ontología son, en este caso, las siguientes:

- Construcción: centrada en la creación (de forma rápida) de una versión inicial de la ontología. Conviene resaltar que no se propone en ningún momento la realización de una especificación de requisitos, o la búsqueda de posibles ontologías a reutilizar.

- Adaptación: durante esa fase se trata de adaptar la ontología a las necesidades específicas de los usuarios. Se plantea poner la ontología a disposición de los usuarios para que la utilicen, y que sean ellos mismos los que vayan adaptándola a sus necesidades (no la modifican directamente, sino que van anotando las sugerencias de mejora).

- Análisis: evaluación de los cambios sugeridos por los usuarios.

- Revisión: el grupo de desarrolladores revisa las sugerencias realizadas por los usuarios y decide cuáles pueden aplicarse.

- Actualización: en esta fase se hacen realidad las modificaciones aprobadas en la fase anterior.

Como puede apreciarse en la tabla 3.1, DILIGENT no propone ningún tipo de actividad relacionada con la especificación de requisitos, ni tampoco menciona nada sobre la reutilización de recursos ontológicos. Por otra parte, esta ontología está orientada a ser utilizada por los propios expertos del dominio objetivo, sin tener en cuenta a los desarrolladores de software o de ontologías.

NeOn Methodology La metodología NeOn, desarrollada por el Ontology Engineering Group de la Universidad Politécnica de Madrid [31], está destinada a la construcción de redes de ontologías. Se trata de una metodología basada en escenarios, y que soporta aspectos relacionados con el desarrollo de ontologías y su reutilización, además de la evolución dinámica de las redes de ontologías en entornos distribuidos. Conviene resaltar que la mayor parte de la información que aparece en esta memoria sobre la metodología NeOn se ha tomado de [30].

Los principales aspectos de NeOn son:

- Un conjunto de nueve escenarios para construir ontologías y redes de ontologías, todos ellos centrados en la reutilización de recursos (tanto ontológicos como no ontológicos) y en los procesos de reingeniería y fusión de ontologías.
- Un glosario [32] de procesos y actividades que identifica y define los diferentes procesos y actividades que puede haber que llevar a cabo cuando las ontologías son desarrolladas por grupos de trabajo.
- Guías metodológicas para los diferentes procesos y actividades del proceso de desarrollo de la ontología (reutilización, reingeniería de recursos ontológicos y no ontológicos, respecificación de requisitos, etc.). Todos los procesos y actividades están descritos y vienen acompañados de un diagrama de flujo y de ejemplos prácticos.

Como puede apreciarse en la tabla 3.1, NeOn Methodology contempla, y describe ampliamente, tareas como la especificación de requisitos, la planificación y la reutilización de recursos ontológicos. Además, está preparada para que pueda ser utilizada tanto por el público general, como por los usuarios habituales de recursos ontológicos, como por los desarrolladores de software.

En la tabla 3.1 se muestra una comparativa entre cuatro de las metodologías de desarrollo de ontologías más destacadas. Como puede apreciarse, la metodología NeOn contempla procesos que no son contemplados por el resto de metodologías, por lo que se ha considerado más completa, siendo ésta la razón por la que se ha seleccionado esta metodología para ser aplicada en el ámbito del desarrollo que nos ocupa. El hecho de que esta metodología contemple ampliamente las tareas de especificación de requisitos, planificación y reutilización de ontologías ya existentes la hace extremadamente adecuada para casos en los que, como en el trabajo que nos ocupa, se preste especial atención a la fase de análisis y a la facilidad de ampliación/modificación de la ontología desarrollada.

	Methontology	**On-To-Knowledge**	**DILIGENT**	**NeOn**
Collaboración	No mencionado	No mencionado	Tratado	Sin detalle

Especificación de requisitos	Sin detalle. Preguntas de competencia	Sin detalle. Preguntas de competencia	Actividad no considerada	Tratado en detalle
Planificación	No mencionado	No mencionado	No mencionado	Tratado en detalle
Reutilización de recursos ontológicos	No mencionado	No mencionado	No mencionado	Tratado en detalle
Orientada a desarrolladores de software y profesionales de las ontologías	Orientada a ingenieros de ontologías e investigadores	Orientada a ingenieros de ontologías e investigadores	Orientada a expertos en el dominio objetivo y usuarios	Orientada a profesionales de las ontologías, público general e ingenieros de software

Tabla 3.1: Comparativa de metodologías de desarrollo de ontologías. Fuente [30]

3.2. NeOn Methodology

En esta sección se describe ampliamente la metodología empleada para el desarrollo de la ontología: NeOn Methodology [30].

Como ya se ha indicado, la metodología NeOn contempla nueve escenarios, que se describen brevemente a continuación:

- *Escenario 1 - De la especificación a la implementación* Este escenario se utilizará en caso de que la ontología se desarrolle desde cero (sin reutilizar recursos ya existentes). Deberán especificarse los requisitos,

tras lo cual es aconsejable realizar una búsqueda de posibles recursos ya existentes que puedan ser reutilizados. Una vez hecho ésto, deberá planificarse la actividad a desarrollar [33].

- *Escenario 2 - Reutilización y reingeniería de recursos no ontológicos (NORs)* En este caso, el equipo de trabajo debería abordar el proceso de reutilización de NORs para así decidir, de acuerdo a los requisitos especificados para la ontología, qué NORs han de ser reutilizados para construir la ontología en cuestión. Los NORs seleccionados deberán transformarse en ontologías [34].

- *Escenario 3 - Reutilización de recursos ontológicos* En este escenario se contempla la posibilidad de reutilizar ontologías ya disponibles (ya sean ontologías completas ([35], [36]), módulos específicos de ontologías más grandes, ... [37]).

- *Escenario 4 - Reutilización y reingeniería de recursos ontológicos* Se contempla aquí la posibilidad de, además de reutilizar recursos ontológicos, abordar sobre ellos ciertas labores de reingeniería, para así adaptarlos a la ontología que se está desarrollando.

- *Escenario 5 - Reutilización y fusión de recursos ontológicos* Este escenario aborda la situación en la que existen múltiples recursos ontológicos del mismo dominio que van a ser reutilizados, de forma que se pretende crear una nueva ontología resultante de fusionar todos estos recursos ontológicos ya existentes.

- *Escenario 6 - Reutilización, fusión y reingeniería de recursos ontológicos* En este caso los desarrolladores van a reutilizar, fusionar y realizar reingeniería sobre recursos ontológicos ya existentes. Este escenario es similar al anterior, con la diferencia de que los desarrolladores deciden realizar reingeniería sobre el conjunto de recursos que se han fusionado.

- *Escenario 7 - Reutilización de patrones de diseño de ontologías (ODPs)* Los desarrolladores accederán, según este escenario, a repositorios de ODPs, para desarrollar su ontología en base a uno o varios de estos ODPs.

- *Escenario 8 - Reestructuración de recursos ontológicos* En este caso los desarrolladores pretenden reestructurar (podar, extender, especializar,

etc.) los recursos ontológicos que van a ser integrados en la ontología (o red de ontologías) a desarrollar.

- *Escenario 9 - Traducción de recursos ontológicos* Este escenario contempla la tarea de adaptar una ontología a otros lenguajes y/o culturas, obteniendo así una ontología multilenguaje.

Como se verá más adelante, existen múltiples dependencias entre los diferentes escenarios planteados:

- Si se selecciona el escenario 4, entonces el 3 ha de ser ejecutado también.
- Si se selecciona el escenario 5, entonces el 3 también ha de ser ejecutado.
- Si ha de ejecutarse el escenario 6, el escenario 5 es también obligado, y por ello el escenario 3.

Conviene mencionar que el escenario 1 ha de ser incluido siempre en cualquiera de los escenarios, ya que contiene la parte principal del desarrollo de la ontología. Por otra parte, también existe la posibilidad de combinar varios escenarios, intentando adaptar lo máximo posible los escenarios seleccionados a lo que en realidad se pretende hacer en el ámbito del proyecto. En la figura 3.1 se muestra un árbol de decisión que facilita la elección de los escenarios a considerar y el modelo de ciclo de vida a seguir para un desarrollo concreto. Si los requisitos de la ontología son conocidos desde el comienzo del desarrollo, conviene seguir un modelo de ciclo de vida en cascada, pero si no se conocen completamente, entonces el modelo de ciclo de vida a seguir sería el iterativo e incremental.

Por otra parte, y en lo que se refiere a los modelos de ciclo de vida, esta metodología plantea dos (para más información consultar [30]):

- *modelo de ciclo de vida en cascada*: en este tipo de modelos, debe completarse completamente una fase antes de poder pasar a la siguiente, permitiéndose la vuelta atrás desde la fase de mantenimiento hacia la fase anterior a la captura de requisitos. Existen diferentes tipos de ciclos de vida en cascada, en función de las fases que se deseen abordar durante el desarrollo del proyecto. Así, el más básico es aquel que contempla únicamente cuatro fases (iniciación, diseño, implementación y mantenimiento). A medida que se van añadiendo más fases (reutilización, fusión, reingeniería), el ciclo de vida irá pasando a ser de cinco,

seis o siete fases. En la figura 3.2 se muestra un esquema de lo que sería un ciclo de vida en cascada con seis fases.

- *modelo de ciclo de vida iterativo e incremental*:

En el ámbito de este proyecto se ha decidido cubrir el escenario 6 (ya que posiblemente habrá que reutilizar y fusionar diferentes recursos ontológicos, pero cabe la posibilidad de que haya que realizar algún tipo de reingeniería sobre dichos recursos ontológicos, para adaptarlos a las necesidades de la ontología que se pretende desarrollar) y seguir un modelo de ciclo de vida en cascada (ya que los requisitos de la ontología a desarrollar se conocerán desde el principio) en seis fases más la fase de fusión de recursos ontológicos (3.2).

Conviene indicar que para muchas de las tareas a realizar durante el desarrollo de una ontología utilizando NeOn Methodology conviene utilizar NeOn Toolkit [38]. Esta herramienta, basada en Eclipse (y por lo tanto multiplataforma), proporciona soporte a lo largo de todo el ciclo de vida de la ingeniería de ontologías. Existen más de 45 plugins disponibles para incorporar nuevas funcionalidades a la herramienta y que cubren multitud de actividades a realizar durante el proceso de ingeniería (anotación y documentación, desarrollo, adquisición de conocimiento, modularización, evaluación de ontologías, razonamiento e inferencia, etc.).

3.2.1. Escenarios para construir redes de ontologías

Como ya se ha mencionado anteriormente, la metodología NeOn contempla nueve escenarios para desarrollar ontologías colaborativamente, los cuales hacen especial hincapié en la reutilización y reingeniería de recursos (tanto ontológicos como no ontológicos).

En la figura 3.4 (fuente: [39]) se presentan los nueve escenarios ya mencionados, representados mediante círculos numerados. Cada escenario aparece descompuesto en actividades y procesos, representados éstos con círculos coloreados o rectángulos redondeados (la definición de todos ellos puede consultarse en el glosario de procesos y actividades de la metodología [32]).

Escenario 1: De la Especificación a la Implementación

Este escenario ha de utilizarse en los casos en que se desea construir la/s ontología/s partiendo de cero, desde el comienzo, sin reutilizar recursos ya existentes.

Para poder considerar este escenario, el conocimiento del dominio de la ontología ha de estar disponible.

La secuencia de actividades que plantea este escenario es la siguiente:

- Estudio de viabilidad, evaluando si para el dominio objetivo conviene desarrollar una única ontología, varias ontologías interconectadas, o una red de ontologías. Un conjunto de ontologías interconectadas incluye una serie de ontologías que poseen algún tipo de relación entre ellas. Sin embargo, una red de ontologías es una colección de ontologías interconectadas mediante varias meta-relaciones, como pueden ser:
  - hasPriorVersion: la ontología a desarrollar es una nueva versión de una ontología existente (evolución de ontologías).
  - useImports: la nueva ontología importará definiciones de una ontología ya existente (reutilización de ontologías).
  - isExtension: la nueva ontología extiende una ontología existente (enriquecimiento de ontologías, que pueden realizarse por especialización o por generalización).
  - etc.
- Una vez decidido el punto anterior, debe comenzarse con la adquisición de conocimiento, actividad que debe llevarse a cabo durante todo el proceso de desarrollo, si bien es verdad que la mayor parte del conocimiento debe ser adquirido al comienzo.
- En paralelo a la actividad anterior, deben especificarse los requisitos de la ontología, dando como salida un ORSD (Ontology Requirements Specification Document) que, como se verá más adelante, incluye, entre otros, el propósito, el ámbito y el lenguaje de implementación de la ontología. Para la especificación de requisitos deben utilizarse Cuestiones/Preguntas de Competencia.
- Una vez especificados los requisitos, deberá realizarse una búsqueda de posibles recursos ya existentes que podrían utilizarse como base para el desarrollo de la ontología.

- Llegado este momento, debe planificarse el resto de la actividad a realizar, proceso durante el cual se determinará el modelo de ciclo de vida a seguir.
- La siguiente tarea a llevar a cabo es la conceptualización de la ontología, en la cual se organiza y estructura el conocimiento adquirido en modelos a nivel de conocimiento.
- Una vez obtenido el modelo conceptual, debe formalizarse la ontología, lo cual significa transformar el modelo conceptual en un modelo semi-computacional.
- Ahora que ya está disponible el modelo semi-computacional, ha llegado el momento de implementar la ontología, generando así un modelo computacional (ya en un lenguaje de ontologías).

La salida principal que se obtiene en este escenario es una red de ontologías que, se supone, representan el dominio objetivo en un determinado lenguaje de ontologías (OWL, F-Logic, etc.). Además, se generan una serie de documentos, muy importantes durante todo el ciclo de vida: documento de especificación de requisitos, documento de descripción de la ontología documento de evaluación de la ontología.

Este escenario contempla las actividades "básicas" de especificación, conceptualización e implementación, por lo que estaría dentro de un ciclo de vida en cuatro fases (iniciación, diseño, implementación y mantenimiento).

Escenario 2: Reutilización y reingeniería de recursos no ontológicos (NORs)

A la hora de desarrollar ontologías no tiene ningún sentido hacerlo todo desde cero si ya existen recursos que podamos reutilizar. Por ello, si existen clasificaciones, tesauros, glosarios, ... que pueden servir, ¿por qué no utilizarlos? Ya que se trata de recursos no ontológicos, se hace necesario realizar sobre ellos una reingeniería que permita convertirlos en ontologías. Este escenario contempla precisamente esta situación: la necesidad de reutilizar recursos no ontológicos para el desarrollo de la/s ontología/s.

Al igual que en el escenario anterior, el conocimiento del dominio en el que ha de desarrollarse la ontología tiene que estar disponible desde el principio, así como los recursos no ontológicos a utilizar.

Para llevar a cabo el proceso de reutilización de recursos no ontológicos primero deben buscarse dichos recursos, decidiendo cuáles de entre todos los encontrados van a utilizarse. Posteriormente, dentro del proceso de reingeniería de recursos no ontológicos, se procederá a transformar dichos recursos no ontológicos en una (o varias) ontología.

La salida en este caso será una red de ontologías que representan el dominio objetivo y están implementadas en un lenguaje de ontologías. Además se generarán, al igual que en el caso anterior, varios documentos: especificación de requisitos, documentación de la ontología, evaluación de la ontología, etc.

El modelo de ciclo de vida adecuado para este escenario sería el ciclo de vida en cascada en seis fases.

Escenario 3: Reutilización de recursos ontológicos

Además de recursos no ontológicos, cabe la posibilidad de que ya existan ontologías que puedan servir para cubrir ciertos aspectos de la ontología a desarrollar, y para este caso se ha concebido este escenario.

El conocimiento del dominio y los recursos ontológicos a reutilizar deben estar disponibles desde el comienzo.

Dentro del proceso de reutilización de recursos ontológicos, las tareas a realizar consisten en buscar las posibles ontologías a utilizar, estudiarlas, compararlas y seleccionar las que mejor se adecúen a lo que se necesita.

Una vez seleccionadas las ontologías a reutilizar, debe evaluarse el modo en que se reutilizarán: se reutilizarán tal cual están, se pretende fusionar varias de las ontologías en una, etc.

Antes de reutilizar las ontologías seleccionadas conviene evaluarlas (actividad de evaluación de ontologías), siempre y cuando existan varias ontologías disponibles entre las cuales elegir, y sobre todo si ya existen parámetros evaluados para ellas.

Al igual que en los casos anteriores, la principal salida es una ontología (o red de ontologías) que representan el dominio objetivo en un determinado lenguaje, pero además se obtendrán otros documentos (especificación de requisitos, documentación de la ontología, evaluación de la ontología, etc.).

El modelo de ciclo de vida adecuado para este escenario sería el ciclo de vida en cascada en cinco fases.

Escenario 4: Reutilización y reingeniería de recursos ontológicos

Este caso es similar al caso anterior, sólo que es necesario realizar ciertas labores de reingeniería sobre los recursos ontológicos a reutilizar.

Los desarrolladores deberán abordar primero el proceso de reutilización de recursos ontológicos, para después pasar a realizar la correspondiente reingeniería sobre ellos. Finalmente, deberán utilizar los recursos ontológicos resultantes como entrada para algunas de las actividades mencionadas en el escenario 1.

Las labores de reingeniería sobre los recursos ontológicos a reutilizar pueden ser realizar ingeniería inversa, reestructuración de recursos ontológicos, etc.

La siguiente figura [30] representa el modelo de reingeniería de recursos ontológicos en la cual pueden apreciarse los diferentes caminos que pueden seguirse a la hora de realizar reingeniería sobre recursos ontológicos. La selección de uno u otro camino dependerá de las características de cada recurso ontológico a modificar. A modo de resumen, se puede decir que:

re-especificación : se da cuando el equipo de desarrollo decide modificar partes de la especificación de requisitos.

re-conceptualización : se da cuando los cambios son relativos a la estructura de la ontología, su granularidad, los axiomas, etc.

re-formalización : cambios en el nivel de formalización (por ejemplo, cambiar el paradigma de lógica descriptiva a marcos).

re-implementación : si los cambios son a nivel de implementación (por ejemplo, traducir la ontología de RDF a OWL).

Los desarrolladores deben decidir a qué nivel necesitan abordar el proceso de reingeniería y, una vez realizado éste, deberán integrar el resultado en la correspondiente actividad del escenario 1.

Este escenario se refiere al desarrollo de ontologías reutilizando y haciendo reingeniería sobre recursos ontológicos, por lo que estaríamos ante un ciclo de vida en cascada en seis fases.

Escenario 5: Reutilización y fusión de recursos ontológicos

Este escenario contempla la posibilidad de reutilizar varios recursos ontológicos, posiblemente con varias partes que se solapan, para crear con ellos

una nueva ontología. También contempla la situación en la que los desarrolladores necesitan establecer alineaciones entre los recursos ontológicos para crear una nueva red de ontologías.

En primer lugar debe abordarse el proceso de reutilización de recursos ontológicos [32], para así seleccionar los recursos ontológicos a reutilizar. En este caso van los recursos ontológicos van a reutilizarse tal cual están, pero no de forma completa como si fuesen utilizados de forma separada. Para ello, primero hay que alinear las ontologías a reutilizar, y después debe procederse a la fusión de dichas ontologías.

Para este escenario, el modelo de ciclo de vida a seguir sería uno en cascada con cinco fases más una fase de fusión de recursos ontológicos.

Escenario 6: Reutilización, fusión y reingeniería de recursos ontológicos

Este escenario debe contemplarse cuando se van a utilizar múltiples recursos ontológicos del mismo dominio para construir la red de ontologías. De esta forma, se creará una nueva ontología (o red de ontologías) fusionando dos o más recursos ontológicos (posiblemente con cierto solapamiento entre ellos). Si el resultado de fusionar varios de los recursos ontológicos a utilizar no puede utilizarse así, tal cual, y ha de ser modificado (han de llevarse a cabo tareas de reingeniería) éste es el escenario apropiado.

En este caso debe abordarse primero el proceso de reutilización de recursos ontológicos, para así seleccionar los recursos ontológicos que desean reutilizarse (al igual que se explicó para el escenario 3 3.2.1). Posteriormente debe decidirse cómo van a reutilizarse los recursos ontológicos seleccionados. En este escenario los desarrolladores deciden realizar actividades de alineación y fusión de ontologías, ya que los recursos ontológicos a reutilizar son válidos pero no de forma completa (situación ya comentada en el escenario 5 3.2.1). Tras fusionar los recursos seleccionados, deberá abordarse el proceso de reingeniería (3.2.1). Una vez realizadas todas estas tareas, ya podrá utilizarse el recurso ontológico resultante en las actividades incluidas en el escenario 1 3.2.1.

La principal salida que se obtiene en este escenario es una red de ontologías que representa el dominio objetivo en un determinado lenguaje (OWL, F-Logic, etc.), así como, en cualquiera de los casos anteriores, la serie de documentos que se va generando durante el proceso de desarrollo (especificación de requisitos, documentación de la ontología, evaluación de la ontología, etc.).

Por otra parte, también se genera un recurso ontológico fruto de la fusión de los recursos ontológicos reutilizados. Las alineaciones entre los recursos ontológicos de partida también pueden considerarse como una salida de este escenario.

Esta forma de construir ontologías estaría representada por un ciclo de vida en cascada en seis fases más la fase de fusión.

Escenario 7: Reutilización de patrones de diseño de ontologías (ODPs)

Este escenario contempla aquella situación en la que el equipo de desarrollo decide seguir cierto patrón de diseño de ontologías para la construcción de la ontología que les ocupa. Los patrones de diseño de ontologías se utilizan por múltiples motivos: para reducir las dificultades de modelado que surgen en algunas ocasiones, para acelerar el proceso de modelado, para comprobar que las decisiones de modelado que se han tomado son correctas, o incluso para evaluar el modelado obtenido.

En este caso, además de disponer desde el comienzo del conocimiento sobre el dominio objetivo (al igual que en el resto de escenarios), debe contarse con cierto conocimiento sobre patrones de diseño de ontologías, así como de los repositorios en que pueden encontrarse dichos patrones de diseño (por ejemplo, [40]).

Dado que los desarrolladores deben abordar el proceso de reutilización de patrones de diseño de ontologías (ver [32]), primero habrán de seleccionar el/los patrones a utilizar, después deberán adaptarlos a sus necesidades y, por último, integrarlos en su desarrollo.

La principal salida que se obtiene en este escenario es un patrón de diseño de ontologías integrado en la red de ontologías a desarrollar.

En este caso, estaríamos ante un modelo de ciclo de vida en cascada con cinco fases.

Escenario 8: Reestructuración de recursos ontológicos

Este escenario debe seleccionarse cuando el conocimiento contenido en el modelo conceptual desarrollado debe ser corregido y reorganizado para obtener una red de ontologías que cubra los requisitos especificados al comienzo del proyecto. De esta forma, los desarrolladores deberán abordar la actividad

de reestructuración de la ontología, siempre tras la actividad de conceptualización de la misma [32].

El modelo de ciclo de vida a adoptar en este aso sería muy similar al mencionado en el escenario 1, por lo que estaríamos ante un modelo de ciclo de vida en cuatro fases.

Escenario 9: Traducción de recursos ontológicos

Aunque existen multitud de ontologías disponibles para ser utilizadas, la mayor parte de ellas está en inglés, por lo que los desarrolladores que no vayan a trabajar en este idioma deberán traducir primero aquellos recursos ontológicos que vayan a reutilizar para realizar su desarrollo.

En este caso, el equipo de desarrollo deberá abordar la tarea de traducción de la ontología ("ontology localization activity" [32]), mediante la cual se traducirá una ontología de un lenguaje natural. Una vez finalizada esta actividad, el modelo conceptual obtenido podrá ser integrado en la actividad de conceptualización ya considerada en el ámbito del escenario 1 (3.2.1).

Este escenario correspondería también a un modelo de ciclo de vida muy similar al del escenario 1; es decir, un modelo de ciclo de vida en cascada con cuatro fases.

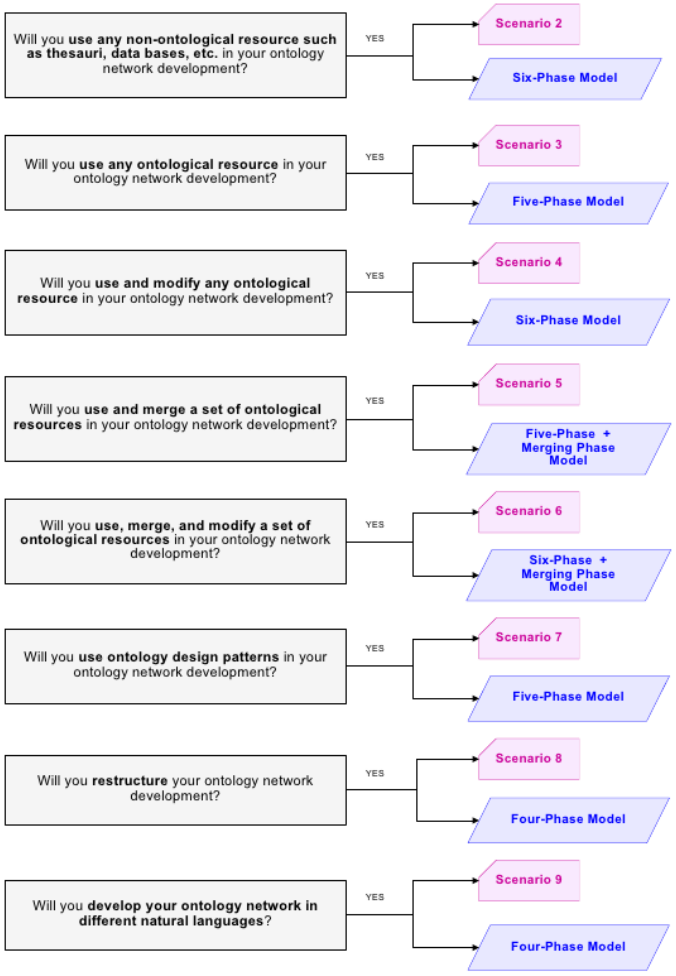

Figura 3.1: Árbol de decisión para la selección de los escenarios a cubrir

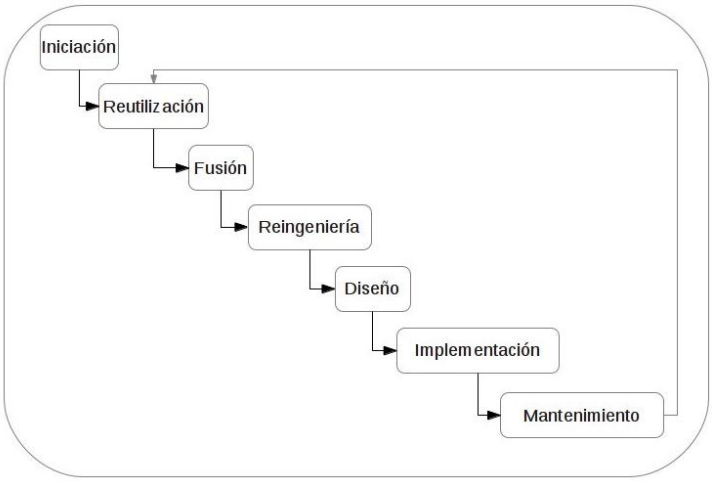

Figura 3.2: Modelo de ciclo de vida en cascada en seis fases

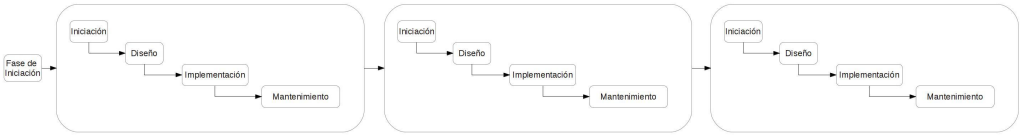

Figura 3.3: Modelo de ciclo de vida iterativo e incremental

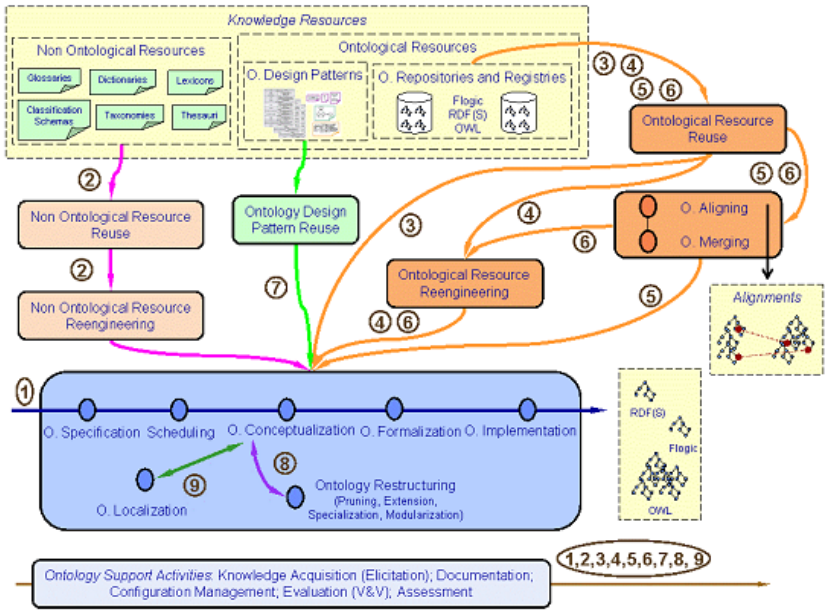

Figura 3.4: Escenarios definidos en la metodología NeOn

Capítulo 4

DESARROLLO DE LA ONTOLOGÍA

En este capítulo se detalla todo lo relacionado con el desarrollo del proyecto que nos ocupa: posibilidades de reutilización de recursos ontológicos, especificación de requisitos, planificación, etc.

4.1. Especificación de Requisitos

El documento de especificación de requisitos [41] juega un papel muy importante durante todo el proceso de desarrollo, ya que facilita la realización de diferentes actividades, como puede ser la utilización de patrones de diseño. Por otra parte, y al igual que en un proyecto de desarrollo de software, en un proyecto de desarrollo de ontologías no se puede cerrar la planificación (ni realizar una planificación detallada) hasta que no se conocen los requisitos de la ontología, e incluso puede también haberse realizado ya una primera búsqueda de recursos (ontológicos y no ontológicos) disponibles que pudieran reutilizarse para el desarrollo de la ontología en cuestión.

La especificación de requisitos toma como base una lista de preguntas de competencia [42], que identificarán perfectamente los requisitos que ha de cumplir la ontología en cuestión.

4.1.1. Preguntas de competencia

Las preguntas de competencia son una lista de preguntas a las que debe poder dar respuesta la ontología a desarrollar [42]. Constituyen una especie de requisitos, sólo que están en forma de preguntas. En cierta manera, se puede establecer un paralelismo entre las preguntas de competencia y los requisitos del sistema, ya que ambos identifican características de la ontología a desarrollar.

Lo más adecuado es que las preguntas de competencia estén formuladas de forma jerárquica, en la cual las preguntas de niveles superiores requieren que las preguntas de niveles inferiores asociadas tengan respuesta.

A continuación se muestra el listado de preguntas de competencia de aplicación a este desarrollo:

- *CQ1*: Características del hogar digital en cuestión
  - *CQ1.1*: ¿Cuántas estancias tiene?
  - *CQ1.2*: ¿De qué tipo es cada estancia?
  - *CQ1.3*: ¿Cuál es la orientación de cada estancia?
  - *CQ1.4*: Dada una determinada estancia, ¿cuántas personas hay dentro?
  - *CQ1.5*: Dada una determinada estancia, ¿es posible saber a qué estancia corresponde dentro del modelo CAD del edificio? (pregunta referida a la integración de la ontología con gbXML, idea que ya se ha comentado anteriormente en el capítulo 1.1).
- *CQ2*: Características técnicas del hogar digital
  - *CQ2.1*: ¿Cuáles son los dispositivos domóticos (sensores, actuadores, controladores) disponibles?
  - *CQ2.2*: ¿Cuáles son los dispositivos multimedia disponibles?
  - *CQ2.3*: ¿Cuáles son los computadores disponibles (incluidos periféricos (sobre todo impresoras y similares))?
  - *CQ2.4*: ¿Cuáles son los dispositivos móviles disponibles en cada instante?
  - *CQ2.5*: ¿Quién es el propietario de cada uno de los dispositivos móviles existentes?

- *CQ2.6*: ¿Cuáles son los electrodomésticos disponibles?
- *CQ2.7*: ¿Dónde están localizados los diversos dispositivos (al menos en qué sala de la casa)?
- *CQ2.8*: Dado un determinado dispositivo (sobre todo sensores), ¿cuál es su valor en un determinado instante?
- *CQ2.9*: ¿Qué consumo eléctrico medio tiene cada electrodoméstico? (únicamente los que se consideren interesantes, no despreciables)
- *CQ2.10*: ¿Cuál es la dirección IP de los diferentes dispositivos susceptibles de tener una dirección de este tipo asignada?
- *CQ2.11*: ¿qué alarmas hay definidas para las diferentes variables de los dispositivos? [1]
- *CQ2.12*: ¿qué alarmas se han generado para las distintas variables de los dispositivos?
- *CQ2.13*: ¿qué tareas programadas hay definidas para los diversos dispositivos [2]?

▪ *CQ3*: Características de los habitantes de la casa

- *CQ3.1*: ¿Cuántos habitantes tiene la casa?
- *CQ3.2*: ¿Cuál es la identificación y el rol de cada habitante?
- *CQ3.3*: ¿Cuál es el dormitorio de cada habitante?
- *CQ3.4*: ¿Cuál es la localización de cada habitante en el instante actual (al menos para los que estén dentro de la casa)?
- *CQ3.5*: ¿Cuál es la edad de cada habitante?
- *CQ3.6*: ¿Cuál es el sexo de cada habitante?

▪ *CQ4*: Características de la instalación eléctrica

- *CQ4.1*: ¿Cuál es la potencia eléctrica contratada?

[1] DogOnt utiliza el término "state" para lo que aquí se ha denominado "variable". Aunque en la ontología desarrollada se mantiene la nomenclatura utilizada por DogOnt, en las preguntas de competencia se mantiene la palabra "variable" por considerarla más explicativa en este contexto.

[2] Aunque aquí se mantiene la nomenclatura, en la ontología las tareas están asociadas a los Comandos (nomenclatura DogOnt) de los dispositivos.

- *CQ4.2*: ¿Cuántas líneas tiene la casa?
- *CQ4.3*: ¿A qué línea eléctrica está conectado cada electrodoméstico?
- *CQ4.4*: ¿Cuál es el plan de precios contratado?
- *CQ4.5*: ¿Cuáles son las consignas de carga para cada línea? (si hay alguna restricción en cuanto a la carga de tal o cual línea (o de la instalación en general))
- *CQ4.6*: ¿Cuál es la carga actual de cada una de las líneas eléctricas?
- *CQ4.7*: Dado un determinado electrodoméstico, ¿puede ser desconectado estando en marcha? (útil para implementar políticas de gestión de cargas) ¿Durante cuánto tiempo puede permanecer desconectado?

Conviene resaltar que, aunque el análisis de requisitos, incluidas las preguntas de competencia y demás, se ha realizado en español, la ontología se definirá en inglés por motivos de usabilidad y compatibilidad con otras ontologías disponibles. En cualquier caso, siempre existe la posibilidad de traducir un ontología, actividad que también está contemplada dentro del listado de procesos y actividades de la metodología NeOn [32].

A continuación se presentan posibles respuestas a las preguntas de competencia anteriormente enumeradas:

- *CQ1*: Características del hogar digital en cuestión
  - *CQ1.1*: 1, 2, 3, 4, ...
  - *CQ1.2*: Salón, cocina, dormitorio, baño, aseo, porche, pasillo, ...
  - *CQ1.3*: Para cada estancia, decir si tiene orientación norte, sur, este, oeste, noroeste, sureste, ...
  - *CQ1.4*: Para cada estancia decir: dentro hay 1,2,3, ... personas.
  - *CQ1.5*: La estancia salón tiene el id 34324 dentro del modelo gbXML 1.1).
- *CQ2*: Características técnicas del hogar digital

- *CQ2.1*: Sensor del luz, sensor de presencia, sensor de temperatura, sensor de inundación, sensor de fuga de gas, dimmer (actuador de luz, regulable y con on/off), cámara, micrófono, motor de persiana, sensor de persiana, sensor de apertura de puerta, sensor de apertura de ventana,...
- *CQ2.2*: Reproductor se sonido, reproductor de video, DVD, disco multimedia, televisión, ...
- *CQ2.3*: ordenador, portátil, tablet, netbook, impresora, escáner, router, ...
- *CQ2.4*: smartphone, móvil básico, ...
- *CQ2.5*: Juan es el propietario del smartphone de número 123456789, Ana es la propietaria del móvil convencional de número 321654987, ...
- *CQ2.6*: lavadora, secadora, lavadora-secadora, lavavajillas, frigorífico, arcón, microondas, horno, cocina, ...
- *CQ2.7*: En la cocina están los dispositivos: sensor de luz (con ID XXX), sensor de presencia (con ID YYY); en el salón están los dispositivos: sensor de luz (con ID ZZZ) y sensor de temperatura (con ID VVV), ...
- *CQ2.8*: Actualmente, el dispositivo sensor de luz de ID ZZZ tiene un valor de 300luxes, el sensor de temperatura VVV tiene un valor de 23ºC, ...
- *CQ2.9*: La potencia del frigorífico es de 150W, la potencia de la lavadora es 2000W, ...
- *CQ2.10*: El ordenador con MAC 14:da:e9:5e:86:b3 tiene asociada la IP 192.168.125.51, ...
- *CQ2.11*: Para la variable nivel de luminosidad del dispositivo sensor de luz con ID ZZZ existe configurada una variable que se disparará si este nivel baja de 50luxes
- *CQ2.12*: La variable nivel de luminosidad del dispositivo sensor de luz con ID ZZZ se disparó el 22 de Febrero de 2012 a las 22:15:23h (2012-02-22 T 22:15:23).
- *CQ2.13*: El dispositivo dimmer con ID DDD ha de modificar su nivel de luxes a 250luxes todos los días a las 22:00, y esta tarea

debe permanecer activa hasta el 13 de Agosto de 2012 a las 22:15 (2012-08-13 T 22:15)

- *CQ3*: Características de los habitantes de la casa
  - *CQ3.1*: La casa tiene 5 habitantes
  - *CQ3.2*: Gonzalo Ayuso (ID AAA) es el padre, María Pérez (ID BBB) es la madre, ...
  - *CQ3.3*: La habitación de los habitantes AAA y BBB es la HHH, ...
  - *CQ3.4*: El habitante AAA están en la estancia HHH, el habitante BBB está en la estancia H1H, ...
  - *CQ3.5*: El habitante AAA tiene 40 años, el habitante BBB tiene 20 años, ...
  - *CQ3.6*: El habitante AAA es un hombre, el habitante BBB es una mujer, ...
- *CQ4*: Características de la instalación eléctrica
  - *CQ4.1*: La potencia eléctrica contratada 3.7kW
  - *CQ4.2*: La casa tiene 9 líneas eléctricas (alumbrado - línea 1, exteriores - línea 2, usos varios - línea 3, ...)
  - *CQ4.3*: La lavadora está conectada a la línea 1, el frigorífico está conectado a la línea 5, ...
  - *CQ4.4*: El término de potencia es de 20.633129 €/kW AÑO, y el término de energía es de 0.142319 €/kWh Debe contemplarse la posibilidad de que exista tarificación horaria, lo cual está en la mente de muchas compañías eléctricas hoy en día: así, el término de potencia será único y el término de energía variará por tramos horarios a lo largo del día.
  - *CQ4.5*: La línea 1 no puede superar los 3000W, ...
  - *CQ4.6*: La línea 1 tiene actualmente 2000W, la línea 2 tiene una carga de 200W, ...
  - *CQ4.7*: La lavadora puede pararse estando en marcha, pero la vitrocerámica no.

4.1.2. Especificación de requisitos de la ontología

Se presenta en esta sección la especificación de requisitos que se ha realizado para la ontología desarrollada.

El objetivo de la especificación de requisitos de una ontología es justificar el porqué de la ontología, porqué está siendo desarrollada, cuáles son los usos que se plantean para ella, cuáles son los usuarios finales y, por supuesto, cuáles son los requisitos que ésta debe satisfacer.

Como ya se ha mencionado, para especificar los requisitos debe utilizarse la técnica de las "preguntas de competencia" para especificar los requisitos [42, 43], pero antes de identificar el conjunto de preguntas de competencia, conviene centrar claramente el propósito y alcance de la ontología, su nivel de formalidad, los usos para los que se está construyendo y los usuarios finales.

Para la actividad de especificación de requisitos ha de emplearse el formato que proporciona la metodología, y que incluye los siguientes campos: propósito, alcance, lenguaje de implementación, usuarios finales, usos para los que está prevista, requisitos y pre-glosario de términos.

1	**Propósito**
	Propósito general de la ontología.
2	**Alcance**
	Alcance general y grado de detalle que debe alcanzar la ontología.
3	**Lenguaje de Implementación**
	Lenguaje formal con que ha de especificarse la ontología.
4	**Usuarios finales previstos**
	Usuarios finales previstos.
5	**Usos previstos**
	Usos que se ha previsto que sea empleada la ontología.
6	**Requisitos de la Ontología**
	a.Requisitos no Funcionales
	Requisitos generales o aspectos que la ontología ha de cubrir, incluyendo las prioridades (opcionales) para cada uno de ellos.
	b.Requisitos Funcionales
	Requisitos específicos de contenido para la ontología, detallados en forma de preguntas de competencia y sus respectivas respuestas. Opcionalmente puede incluirse también la prioridad para cada grupo y/o para cada pregunta de competencia.

7	**Glosario inicial de términos**
	a.Términos relativos a las preguntas de competencia
	Lista de términos incluidos en las preguntas de competencia, y sus frecuencias.
	b.Términos relativos a las respuestas a las preguntas de competencia
	Lista de términos que aparecen en las respuestas a las preguntas de competencia, y sus frecuencias.
	c.Objetos
	Lista de objetos incluidos en las preguntas de competencia y sus respuestas

Tabla 4.1: Plantilla para la Especificación de Requisitos (ORSD - Ontology Requirements Specification Document Template)

En la tabla 4.1.2 se presenta la especificación de requisitos para la ontología que nos ocupa:

1	**Propósito**
	La presente ontología pretende representar el conocimiento relativo a un Hogar Digital, incluyendo tanto las características físicas y técnicas del hogar digital en cuestión como conocimiento relativo a los habitantes de la casa y la instalación eléctrica de la misma.
2	**Alcance**
	El alcance de la ontología deberá ser tal que permita desplegar sobre ella servicios de valor añadido orientados a gestionar dicho hogar digital, mejorar el confort de los usuarios y realizar operaciones de gestión de la energía sobre el mismo.
3	**Lenguaje de Implementación**
	El lenguaje formal a utilizar deberá ser OWL.
4	**Usuarios finales previstos**
	Los usuarios finales serán los desarrolladores de los mencionados servicios de valor añadido..
5	**Usos previstos**
	Como ya se ha mencionado, el objetivo principal de esta ontología es facilitar el entendimiento entre los diferentes servicios que pueden desplegarse en un hogar digital (y entre los desarrolladores de dichos servicios).
6	**Requisitos de la Ontología**
	a.Requisitos no Funcionales
	Requisitos generales o aspectos que la ontología ha de cubrir, incluyendo las prioridades (opcionales) para cada uno de ellos.
	b.Requisitos Funcionales
	Los requisitos funcionales se han detallado en forma de preguntas de competencia (ya detalladas anteriormente), a las cuales deberá poder dar respuesta la ontología desarrollada Una vez desarrollada la ontología, estas preguntas de competencia se convertirán en consultas SPARQL para poder verificar el cumplimiento de todas y cada una de ellas (y por tanto de todos los requisitos establecidos).
7	**Glosario inicial de términos**
	a.Términos relativos a las preguntas de competencia

	Las tablas que contienen esta información están disponibles en el anexo A.
	b.Términos relativos a las respuestas a las preguntas de competencia
	Las tablas que contienen esta información están disponibles en el anexo A.
	c.Objetos
	Aunque no todos los términos enumerados a continuación serán entidades en la ontología final, se listan aquí todos los términos que se han considerado de interés, sea lo que sea aquello en lo que se transformen en la ontología final: dispositivo, habitante, estancia, red eléctrica, línea eléctrica, móvil, electrodoméstico, variable, alarma, casa, sensor, carga, rol, tarea programada, consigna línea eléctrica, plan de precios contratado, consumo, potencia, localización, dormitorio, sexo, edad, actuador, controlador, multimedia, periférico, computador, orientación, impresora, propietario, lavadora, potencia, frigorífico, reproductor, cocina, smartphone, ordenador, persiana, dimmer, secadora, presencia, temperatura, luminosidad, porche, baño, vitrocerámica, madre, padre, aseo, hombre, mujer, televisión, portátil, video, tablet, netbook, escáner, gas, inundación, sureste, este, norte, sur, oeste, gbxml

Tabla 4.2: Especificación de Requisitos

4.2. Reutilización de Recursos Ontológicos

Una vez conocidos los requisitos, se ha realizado una búsqueda de posibles recursos ontológicos a utilizar, ya que la posibilidad de reutilizar ontologías (o partes de ontologías ya existentes) ayudan, en ocasiones, a reducir el tiempo de desarrollo.

La búsqueda se ha realizado tomando como base la lista de buscadores de ontologías disponibles en la web disponible en [44], sobre todo empleando Watson[3] y Swoogle[4].

Teniendo en cuenta las preguntas de competencia especificadas, se decidió que lo más factible era reutilizar recursos ontológicos relacionados con: tiempo, localización, electrodomésticos en el hogar, eventos, familia y amigos (para almacenar las relaciones entre los habitantes del hogar), ordenadores y periféricos, calendario, preferencias, etc.

De entre todas las búsquedas realizadas, éstas fueron las ontologías que, a priori, se seleccionaron como "posibles":

- *Tiempo*
  - `http://www.w3.org/TR/owl-time/`
  - Características de la ontología: 181 axiomas, 13 clases.
- *DogOnt (ontología ya mencionada en 2.0.1)*
  - `http://elite.polito.it/files/releases/dog/dogont/DogOnt-1.0.6/DOGOnt.owl`
  - Características de la ontología: 6185 axiomas, 626 clases.

Para que el conocimiento modelado por una ontología sea útil, es imprescindible que dicha ontología esté correctamente formada, sin presentar inconsistencias. Por ello, las ontologías anteriores fueron evaluadas utilizando Pellet [45], comprobando así que son consistentes.

Una vez seleccionadas las posibles ontologías a reutilizar, se procedió a su estudio y evaluación, revisando si de verdad eran apropiadas para su utilización en el desarrollo actual, evaluando su calidad y si, en caso de ser elegidas para su reutilización, se reutilizarían de forma completa o únicamente una parte de ellas.

Finalmente se decidió reutilizar tanto la ontología de tiempo como la ontología DogOnt, aunque modificando ambas, ya que en ambas hay partes que no interesan para nada en el ámbito de la ontología que nos ocupa. Las modificaciones realizadas están orientadas sobre todo a la eliminación de partes de la ontología (clases fundamentalmente, constituyendo a veces ramas enteras de la jerarquía) que no aportan nada en el dominio objetivo.

[3]http://watson.kmi.open.ac.uk/
[4]http://swoogle.umbc.edu/

Tras realizar las modificaciones oportunas, las ontologías anteriores poseen las siguientes características:

- *Tiempo*
  - `http://www.w3.org/TR/owl-time/`
  - Características de la ontología tras las labores de reingeniería: 153 axiomas, 8 clases.
- *DogOnt (ontología ya mencionada en 2.0.1)*
  - `http://elite.polito.it/files/releases/dog/dogont/DogOnt-1.0.6/DOGOnt.owl`
  - Características de la ontología tras las labores de reingeniería: 5316 axiomas, 406 clases.

Conviene destacar que ambas ontologías siguen siendo consistentes tras realizar las labores de reingeniería sobre ellas. Al igual que se ha comentado anteriormente, la comprobación de consistencia se realizó utilizando Pellet [45].

4.3. Planificación

Para la planificación de este proyecto se ha utilizado NeOn Toolkit que, a través de su plugin gOntt [46], permite planificar proyectos de desarrollo de ontologías y también puede ser utilizado a lo largo del desarrollo del proyecto para ir siguiendo dicha planificación y además seguir la metodología NeOn. Este plugin permite elaborar una planificación en función de cuál sea el escenario elegido (3.2.1). Además, gOntt informa sobre cómo llevar a cabo cada proceso o actividad a abordar, utilizando para ello las guías proporcionadas por la metodología NeOn y referencias concretas a los plugins de NeOn toolkit que se pueden utilizar en cada caso.

Aunque en [30] puede encontrarse una explicación detallada sobre los procesos y actividades a abordar en base a los escenarios y modelo de ciclo de vida seleccionados, gOntt facilita la tarea de la planificación, realizando una sugerencia de diagrama de Gantt (que luego habrá que personalizar para adaptar al proyecto en cuestión) en base a los escenarios que el equipo de desarrollo haya decidido cubrir.

Como ya se mencionó en 3.2, en el ámbito de este proyecto se ha decidido cubrir el escenario 6 3.2.1. y seguir un modelo de ciclo de vida en cascada en seis fases más la fase de fusión de recursos ontológicos (3.2).

En la figura 4.1 se muestra la planificación detallada que se ha seguido durante el desarrollo del proyecto que nos ocupa, realizada utilizando gOntt, y en la tabla 4.3 se detallan cada una de las tareas identificadas en dicha planificación.

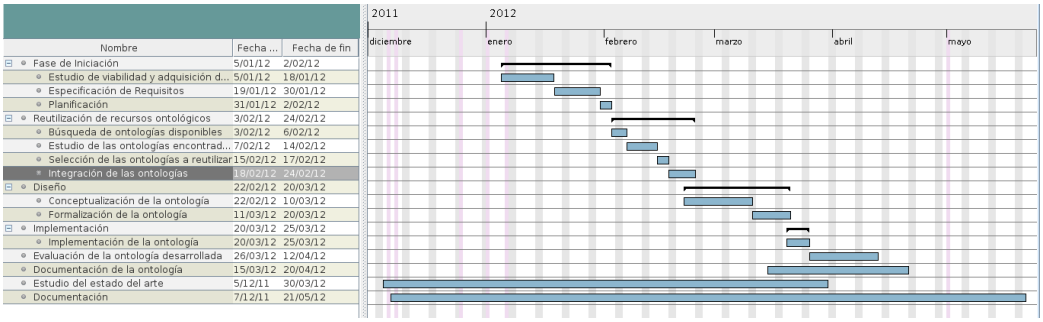

Figura 4.1: Diagrama de Gantt para la planificación

Tarea	**Fecha Inicio**	**Fecha Fin**
Revisión bibliográfica	05/12/2011	30/03/2012
Documentación	07/12/2011	21/05/2012
Fase de Iniciación	**05/01/2012**	**02/02/2012**
Estudio de viabilidad y adquisición de conocimiento	05/01/2012	18/01/2012
Especificación de requisitos	19/01/2012	30/01/2012
Planificación	31/01/2012	02/02/2012
Reutilización de recursos ontológicos	**03/02/2012**	**24/02/2012**
Búsqueda de ontologías disponibles	03/02/2012	06/02/2012
Estudio de las ontologías encontradas	07/02/2012	14/02/2012
Selección de las ontologías a reutilizar	15/02/2012	17/02/2012
Integración de las ontologías	18/02/2012	24/02/2012
Diseño	**22/02/2012**	**20/03/2012**
Conceptualización de la ontología	22/02/2012	10/03/2012
Formalización de la ontología	11/03/2012	20/03/2012
Implementación	**20/03/2012**	**25/03/2012**

Implementación de la ontología	20/03/2012	25/03/2012
Evaluación de la ontología desarrollada	26/03/2012	12/04/2012
Documentación de la ontología	15/03/2012	20/04/2012

Tabla 4.3: Detalle de las tareas a realizar

4.4. Ontología desarrollada

Como ya se ha mencionado anteriormente, dos han sido las ontologías reutilizadas, modificadas e incorporadas a la ontología a desarrollar. Ambas ontologías han sido sometidas a un proceso de reingeniería, para adaptarlas lo máximo posible al dominio objetivo (y, por tanto, a los requisitos establecidos). Por ello, en este apartado me centraré en las adiciones que ha sido necesario realizar para completar la ontología resultante.

La figura 4.2 presenta un extracto de la ontología desarrollada, en el cual se aprecian principalmente las entidades y relaciones que se han añadido (que no forman parte de ninguna de las dos ontologías reutilizadas). En el esquema no se han indicado ni las relaciones inversas (que se han creado en algunos casos para facilitar la búsqueda de información en la ontología) ni las propiedades de tipo de dato simple, ya que la inclusión de toda esta información haría que el esquema resultante fuese imposible de seguir. Como ya se ha dicho, además de añadir toda esta información a la ontología final, también ha habido que modificar las dos ontologías reutilizadas, para adecuarlas perfectamente al ámbito de aplicación definido.

Para obtener una información más detallada de los elementos principales presentes en la ontología final, consultar B.

Los datos de la nueva ontología obtenida son los siguientes:

- Número de clases: 425
- Número de propiedades de tipo de dato simple: 62
- Número de propiedades de objeto: 70
- Número de axiomas: 5729

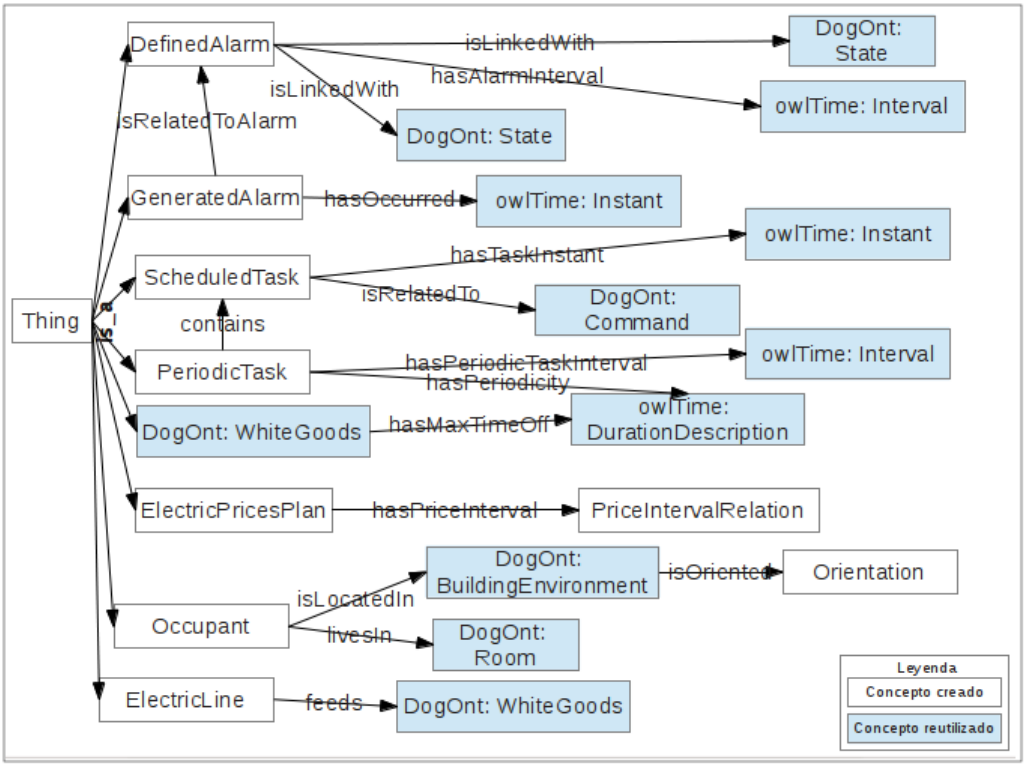

Figura 4.2: Extracto de la ontología desarrollada

4.5. Evaluación de la ontología

Como puede verse en el glosario de la metodología NeOn [32], la tarea de evaluación de una ontología está compuesta de varias subtareas (validación y verificación principalmente). La parte de verificación puede consultarse en el siguiente apartado, pero para el resto de las tareas relacionadas con la evaluación de la ontología desarrollada se ha utilizado el plugin RaDON [47] de Neon Toolkit, que permite evaluar la inconsistencias e incoherencias que pueden existir en las ontologías.

En concreto, RaDON consta de dos plugins:

- *Reparar una única ontología*. Este plugin proporciona las siguientes funcionalidades:

- Tratamiento de incoherencias: si hay al menos un concepto insatisfacible en la ontología, este plugin calificará la misma de incoherente.
- Tratamiento de inconsistencias: si la ontología es inconsistente, este plugin la identificará como tal.
- Reparar automáticamente: en caso de pulsar este botón en NeOn Toolkit, el algoritmo de RaDON procederá a seleccionar los axiomas a ser eliminados para hacer que la ontología sea consistente.
- Reparar manualmente: en este caso, es el usuario el que selecciona los axiomas a eliminar.

- *Reparar y diagnosticar una red de ontologías*: proporciona funcionalidades similares a las comentadas para el plugin anterior, sólo que este plugin contempla una red de ontologías en lugar de una única ontología por separado.

La evaluación de la ontología desarrollada utilizando el plugin RaDON ha permitido comprobar que dicha ontología es tanto coherente como consistente, y que no hay ninguna clase que no sea satisfacible. Por ello, no ha sido necesario utilizar ninguno de los plugins de reparación de ontologías mencionados anteriormente.

4.5.1. Verificación de la ontología

Como se ha comentado anteriormente, la mejor forma de verificar que la ontología desarrollada cumple todos y cada uno de los requisitos que se han especificado en forma de preguntas de competencia (4.1.1) es desarrollar una consulta SPARQL [48] para cada una de dichas preguntas de competencia, y comprobar que la ontología da respuesta a las consultas SPARQL. La tabla 4.5.1 muestra el resultado de la ejecución de las consultas SPARQL elaboradas para la validación de requisitos realizada.

Para la realización de consultas SPARQL sobre la ontología desarrollada se ha utilizado el plugin que NeOn Toolkit proporciona para ello [49]. Este plugin permite ejecutar consultas SPARQL sobre las ontologías que estén cargadas en NeOn Toolkit, y proporciona como salida una tabla con el resultado de la ejecución de la consulta en cuestión.

Los prefijos a utilizar para todas las consultas son los siguientes:

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns# >
PREFIX owl: <http://www.w3.org/2002/07/owl# >
PREFIX xsd: <http://www.w3.org/2001/XMLSchema# >
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema# >
PREFIX digitalhome: <http://localhost/DigitalHomeOntology.owl# >
PREFIX owltime: <http://localhost/owlTimeModificada.owl# >
PREFIX dogont: <http://localhost/dogontModificada.owl# >

Pr. de Competencia	**Consulta SPARQL**	**ok/no ok**
CQ1.1	SELECT ?individuo WHERE { {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Room } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:BuildingEnvironment } }	ok
CQ1.2	SELECT * WHERE { {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Room } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:BuildingEnvironment } }	ok

CQ1.3	SELECT *
	WHERE {
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Room . ?individuo digitalhome:isOriented ?orientacion }
	UNION
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:BuildingEnvironment . ?individuo digitalhome:isOriented ?orientacion }
	}
		ok

CQ1.4	SELECT *
	WHERE {
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Room . ?individuo digitalhome:isOccupied ?habitante }
	UNION
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:BuildingEnvironment . ?individuo digitalhome:isOccupied ?habitante }
	}
		ok

CQ1.5	SELECT *
	WHERE {
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Room . ?individuo digitalhome:idgbXML ?id }
	UNION
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:BuildingEnvironment . ?individuo digitalhome:idgbXML ?id }
	}
		ok

CQ2.1	consulta Sparql[5]	ok

CQ2.2	SELECT *
	WHERE {
	?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Entertainment
	}
		ok

CQ2.3	SELECT * WHERE { {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Computer } UNION {?individuo rdf:type dogont:Printer } UNION {?individuo rdf:type dogont:Scanner } }	ok

[5]Dado que esta consulta es extremadamente larga porque SPARQL no permite navegar recurrentemente por un árbol de profundidad no conocida, se ha decidido, por motivos estéticos, no plasmarla en esta tabla.

CQ2.4	SELECT * WHERE { ?individuo rdf:type digitalhome:MobilePhone }	ok
CQ2.5	SELECT * WHERE { ?individuo rdf:type digitalhome:MobilePhone . ?propietario digitalhome:owns ?individuo }	ok
CQ2.6	SELECT * WHERE { {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:WhiteGoods } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Cooker } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Oven } }	ok
CQ2.7	consulta Sparql[6]	ok
CQ2.8	consulta Sparql[7]	ok

[6]Dado que esta consulta es extremadamente larga porque SPARQL no permite navegar recurrentemente por un árbol de profundidad no conocida, se ha decidido, por motivos estéticos, no plasmarla en esta tabla.

[7]Dado que esta consulta es extremadamente larga porque SPARQL no permite navegar recurrentemente por un árbol de profundidad no conocida, se ha decidido, por motivos estéticos, no plasmarla en esta tabla.

CQ2.9	SELECT * WHERE { {?i rdf:type ?clase . ?clase rdfs:subClassOf dogont:WhiteGoods . ?i digitalhome:consumption ?consumo } UNION {?i rdf:type ?clase . ?clase rdfs:subClassOf dogont:Oven . ?i digitalhome:consumption ?consumo } UNION {?i rdf:type ?clase . ?clase rdfs:subClassOf dogont:Cooker . ?i digitalhome:consumption ?consumo } }	ok

CQ2.10	SELECT * WHERE { {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Appliances . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:BrownGoods . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Communication . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Phone . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Computer . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Entertainment . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:WhiteGoods . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Cooker . ?i dogont:IPAddress ?ip } UNION {?i rdf:type ?X . ?X rdfs:subClassOf dogont:Oven . ?i dogont:IPAddress ?ip } }	ok

CQ2.11	SELECT ?alarma ?disp WHERE {?alarma rdf:type digitalhome:DefinedAlarm . ?alarma digitalhome:isLinkedWith ?estado . ?disp dogont:hasState ?estado }	ok
CQ2.12	SELECT ?alarmagenerada ?alarma ?disp WHERE {?alarmagenerada rdf:type digitalhome:GeneratedAlarm . ?alarmagenerada digitalhome:isRelatedToAlarm ?alarma . ?alarma digitalhome:isLinkedWith ?estado . ?disp dogont:hasState ?estado }	ok
CQ2.13	select ?tarea ?comando ?disp where {?tarea rdf:type digitalhome:ScheduledTask . ?tarea digitalhome:isRelatedTo ?comando . ?funcionalidad dogont:hasCommand ?comando . ?disp dogont:hasFunctionality ?funcionalidad }	ok

CQ3.1	SELECT * WHERE { ?i rdf:type digitalhome:Occupant . ?i digitalhome:name ?name. ?i digitalhome:surname ?surname }	ok
CQ3.2	SELECT ?i ?rol ?dni WHERE { ?i rdf:type digitalhome:Occupant . ?i digitalhome:dni ?dni . ?i digitalhome:rol ?rol }	ok
CQ3.3	SELECT * WHERE { ?i rdf:type ?X . ?X rdfs:subClassOf dogont:Room . ?i digitalhome:isInhabited ?j }	ok
CQ3.4	SELECT ?i ?estancia WHERE {?i rdf:type digitalhome:Occupant . ?i digitalhome:name ?name. ?i digitalhome:surname ?surname . ?estancia digitalhome:isOccupied ?i}	ok

CQ3.5	SELECT ?i ?dateofbirth ?dni
	WHERE {
	?i rdf:type digital-home:Occupant . ?i digitalhome:dateofbirth ?dateofbirth
	}
		ok

CQ3.6	SELECT ?i ?gender ?dni
	WHERE {
	?i rdf:type digital-home:Occupant . ?i dig-italhome:gender ?gender
	}
		ok

CQ4.1	SELECT *
	WHERE {
	?i rdf:type digital-home:ElectricPricesPlan . ?i digital-home:contractedElectricPower ?potenciaElec
	}
		ok

CQ4.2	SELECT ?i
	WHERE {
	?i rdf:type digital-home:ElectricLine
	}
		ok

CQ4.3	SELECT * WHERE {
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:WhiteGoods . ?individuo digitalhome:isFed ?linea}
	UNION
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Cooker . ?individuo digitalhome:isFed ?linea}
	UNION
	{?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Oven . ?individuo digitalhome:isFed ?linea}
	}
		ok

CQ4.4	SELECT ?i ?intervalo ?precio
	WHERE {
	?i rdf:type digitalhome:ElectricPricesPlan . ?i digitalhome:hasPriceInterval ?intervaloPrecio . ?intervaloPrecio digitalhome:intervalValue ?intervalo . ?intervaloPrecio digitalhome:priceValue ?precio
	}
		ok

CQ4.5	SELECT ?i ?carga
	WHERE {
	?i rdf:type digital-home:ElectricLine . ?i digitalhome:maxLoad ?carga
	}
		ok

CQ4.6	SELECT ?i ?carga
	WHERE {
	?i rdf:type digital-home:ElectricLine . ?i digitalhome:currentLoad ?carga
	}
		ok

CQ4.7	SELECT ?individuo ?desc ?time WHERE { {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:WhiteGoods . ?individuo digitalhome:canBeDisconnected ?desc . ?individuo digitalhome:hasMaxTimeOff ?time } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Cooker . ?individuo digitalhome:canBeDisconnected ?desc . ?individuo digitalhome:hasMaxTimeOff ?time } UNION {?individuo rdf:type ?clase . ?clase rdfs:subClassOf dogont:Oven . ?individuo digitalhome:canBeDisconnected ?desc . ?individuo digitalhome:hasMaxTimeOff ?time } }	ok

Tabla 4.4: Consultas SPARQL para la verificación de requisitos

Capítulo 5

CONCLUSIONES Y TRABAJOS FUTUROS

5.1. Conclusiones

Como ya se mencionó en 1.3, varios fueron los objetivos planteados al comienzo de este trabajo:

- Realizar una contribución al diseño de Smart Homes desarrollando una ontología de hogar digital que, además de tener en cuenta los conceptos e información que se han venido contemplando hasta ahora en este tipo de ontologías, tenga en cuenta la mencionada parte energética de dicho hogar, contemplando también la tan necesaria integración con los diferentes modelos que, de un edificio, se van generando a lo largo del ciclo de vida del mismo.

- Hacer que la ontología desarrollada sea lo más completa posible, pero también, y anticipando futuras necesidades, que sea fácilmente ampliable/extensible. Por ello, se planteó prestar especial atención a la metodología a utilizar durante el desarrollo, realizando un estudio de las metodologías disponibles y seleccionando aquella que sea más adecuada para los objetivos planteados.

Al reflexionar sobre el trabajo realizado, puede afirmarse que el estudio de la bibliografía ha permitido conocer las publicaciones existentes sobre ontologías similares a la desarrollada, comprobando que prácticamente ninguna

de ellas cumple los requisitos establecidos en este trabajo. O bien son ontologías que no están disponibles para el público general, por lo cual no ha podido comprobarse su idoneidad, o bien no tienen en cuenta parte de las consideraciones que se pretendían tener en cuenta en la ontología a obtener con este trabajo. Únicamente una de las ontologías encontradas cumplía parcialmente con los requisitos planteados, y por ello se ha reutilizado para la construcción de la ontología final, aunque conviene destacar que ni siquiera esta ontología consideraba los tan mencionados aspectos relativos a la eficiencia energética y la integración con los modelos generados a lo largo del ciclo de vida de un edificio.

Por otra parte, y tras estudiar detenidamente las metodologías susceptibles de ser aplicadas en este desarrollo, se comprobó que la que mejor se adaptaba a las necesidades planteadas es la metodología NeOn, por lo que ha sido ésta la metodología seleccionada para ser aplicada durante el desarrollo de la ontología. Así, se ha logrado obtener una ontología que contempla todos y cada uno de los requisitos establecidos, lo cual ha sido comprobado mediante la realización de consultas SPARQL orientadas a verificar que se han contemplado todas las preguntas de competencia enumeradas en 4.1.1, comprobando además que dicha ontología es tanto consistente como coherente. Por otra parte, se han reutilizado dos ontologías ya desarrolladas y probadas, añadiendo posteriormente sobre ellas todo lo necesario para satisfacer todos y cada uno de los requisitos establecidos 4.1.

Si bien se han añadido a la ontología ciertos elementos que permiten una integración inicial de la misma con los modelos que se generan de un edificio, uno de los trabajos futuros descritos en 5.2 está también orientado a alcanzar un mayor grado de adaptación, que, dada su extensión, ya quedaba fuera del ámbito de este trabajo.

5.2. Trabajos Futuros

Uno de los aspectos que no se ha contemplado en la ontología desarrollada es todo lo relacionado con la accesibilidad del hogar digital en cuestión. Por ello, se plantea como trabajo futuro la ampliación de la ontología con todo lo relativo a accesibilidad. Una muy buena idea puede ser estudiar las ontologías desarrolladas en el ámbito del proyecto ASK-IT [1], pues pueden

[1] `http://www.ask-it.org/`, `http://askit.iti.gr/ontology/`

ser susceptibles de ser reutilizadas (de forma completa o únicamente ciertas partes) para este fin.

Un posible desarrollo a abordar en un futuro sería complementar la ontología con un parseador (o algo similar) que sea capaz de extraer toda la información posible contemplada en la ontología que pueda estar disponible en el/los ficheros IFC/gbXML que hayan podido generarse durante la construcción del edificio en el que se encuentra el hogar digital en cuestión.

Apéndice A

Análisis de las preguntas de competencia y sus respuestas

A continuación se presentan las frecuencias que presentan los términos que aparecen en las preguntas de competencia 4.1.1.

Término	**Ocurrencias**	**Frecuencia**	**Posición ranking**
dispositivos	7	3.4	2
dispositivos	7	3.4	2
son	6	2.9	3
habitante	4	1.9	4
tiene	3	1.4	5
estancia	3	1.4	5
hay	3	1.4	5
disponibles	3	1.4	5
eléctrica	2	1	6
sobre	2	1	6
instante	2	1	6
diversos	2	1	6
líneas	2	1	6
móviles	2	1	6
electrodoméstico	2	1	6
variables	2	1	6
dispositivos	2	1	6
alarmas	2	1	6

casa	2	1	6
disponibles	2	1	6
dirección	2	1	6
diferentes	2	1	6
determinada	2	1	6
sensores	2	1	6
carga	2	1	6
estancia	2	1	6
tipo	2	1	6
desconectado	1	0.5	7
estando	1	0.5	7
rol	1	0.5	7
marcha	1	0.5	7
ser	1	0.5	7
programadas	1	0.5	7
permanecer	1	0.5	7
desconectado	1	0.5	7
tareas	1	0.5	7
tiempo	1	0.5	7
cuántos	1	0.5	7
cuánto	1	0.5	7
habitantes	1	0.5	7
durante	1	0.5	7
eléctricas	1	0.5	7
está	1	0.5	7
línea	1	0.5	7
consignas	1	0.5	7
contratada	1	0.5	7
distintas	1	0.5	7
contratado	1	0.5	7
precios	1	0.5	7
electrodoméstico	1	0.5	7
conectado	1	0.5	7
línea	1	0.5	7
potencia	1	0.5	7
habitante	1	0.5	7

localización	1	0.5	7
dormitorio	1	0.5	7
actual	1	0.5	7
instante	1	0.5	7
sexo	1	0.5	7
edad	1	0.5	7
actual	1	0.5	7
plan	1	0.5	7
eléctrico	1	0.5	7
actuadores	1	0.5	7
domóticos	1	0.5	7
controladores	1	0.5	7
multimedia	1	0.5	7
periféricos	1	0.5	7
incluidos	1	0.5	7
computadores	1	0.5	7
modelo	1	0.5	7
personas	1	0.5	7
orientación	1	0.5	7
tiene	1	0.5	7
estancias	1	0.5	7
dentro	1	0.5	7
posible	1	0.5	7
dentro	1	0.5	7
corresponde	1	0.5	7
saber	1	0.5	7
impresoras	1	0.5	7
similares	1	0.5	7
medio	1	0.5	7
consumo	1	0.5	7
valor	1	0.5	7
dispositivo	1	0.5	7
susceptibles	1	0.5	7
tener	1	0.5	7
han	1	0.5	7
asignada	1	0.5	7

este	1	0.5	7
casa	1	0.5	7
sala	1	0.5	7
existentes	1	0.5	7
uno	1	0.5	7
propietario	1	0.5	7
quién	1	0.5	7
electrodomésticos	1	0.5	7
dónde	1	0.5	7
menos	1	0.5	7
localizados	1	0.5	7
están	1	0.5	7
generado	1	0.5	7

Tabla A.1: Frecuencia de los términos que aparecen en las preguntas de competencia 4.1.1

A continuación se presentan las frecuencias que presentan los términos que aparecen en las respuestas a las preguntas de competencia 4.1.1.

Término	**Ocurrencias**	**Frecuencia**	**Posición ranking**
sensor	16	5.7	1
tiene	9	3.2	2
luz	7	2.5	3
habitante	6	2.2	4
lavadora	5	1.8	5
línea	5	1.8	5
potencia	4	1.4	6
está	4	1.4	6
nivel	4	1.4	6
dispositivo	4	1.4	6
salón	3	1.1	7
variable	3	1.1	7
luxes	3	1.1	7
frigorífico	3	1.1	7
estancia	3	1.1	7
término	2	0.7	8
número	2	0.7	8
debe	2	0.7	8
reproductor	2	0.7	8
cocina	2	0.7	8
casa	2	0.7	8
smartphone	2	0.7	8
actualmente	2	0.7	8
años	2	0.7	8
ordenador	2	0.7	8
persiana	2	0.7	8
dimmer	2	0.7	8
secadora	2	0.7	8
están	2	0.7	8
presencia	2	0.7	8
temperatura	2	0.7	8
apertura	2	0.7	8

móvil	2	0.7	8
dispositivos	2	0.7	8
luminosidad	2	0.7	8
porche	1	0.4	9
habitantes	1	0.4	9
permanecer	1	0.4	9
febrero	1	0.4	9
disparó	1	0.4	9
activa	1	0.4	9
hasta	1	0.4	9
padre	1	0.4	9
tarea	1	0.4	9
días	1	0.4	9
agosto	1	0.4	9
baño	1	0.4	9
exista	1	0.4	9
tarificación	1	0.4	9
posibilidad	1	0.4	9
contemplarse	1	0.4	9
energía	1	0.4	9
kwh	1	0.4	9
horaria	1	0.4	9
superar	1	0.4	9
vitrocerámica	1	0.4	9
marcha	1	0.4	9
estando	1	0.4	9
carga	1	0.4	9
pararse	1	0.4	9
conectado	1	0.4	9
habitantes	1	0.4	9
habitación	1	0.4	9
madre	1	0.4	9
aseo	1	0.4	9
dormitorio	1	0.4	9
hombre	1	0.4	9
eléctricas	1	0.4	9

conectada	1	0.4	9
líneas	1	0.4	9
contratada	1	0.4	9
mujer	1	0.4	9
eléctrica	1	0.4	9
asociada	1	0.4	9
televisión	1	0.4	9
portátil	1	0.4	9
multimedia	1	0.4	9
disco	1	0.4	9
video	1	0.4	9
dvd	1	0.4	9
tablet	1	0.4	9
netbook	1	0.4	9
básico	1	0.4	9
router	1	0.4	9
escaner	1	0.4	9
impresora	1	0.4	9
sonido	1	0.4	9
ventana	1	0.4	9
fuga	1	0.4	9
gas	1	0.4	9
inundación	1	0.4	9
sureste	1	0.4	9
modelo	1	0.4	9
gbxml	1	0.4	9
actuador	1	0.4	9
regulable	1	0.4	9
motor	1	0.4	9
puerta	1	0.4	9
micrófono	1	0.4	9
cámara	1	0.4	9
noroeste	1	0.4	9
propietario	1	0.4	9
dentro	1	0.4	9
norte	1	0.4	9

valor	1	0.4	9
valor	1	0.4	9
pasillo	1	0.4	9
disparará	1	0.4	9
baja	1	0.4	9
temperatura	1	0.4	9
oeste	1	0.4	9
lavavajillas	1	0.4	9
convencional	1	0.4	9
propietaria	1	0.4	9
arcón	1	0.4	9
microondas	1	0.4	9
sur	1	0.4	9
cocina	1	0.4	9
horno	1	0.4	9
luxes	1	0.4	9

Tabla A.2: Frecuencia de los términos que aparecen en las respuestas a las preguntas de competencia 4.1.1

Apéndice B

Descripción de la ontología final

En este anexo se presenta una descripción más detallada de la ontología construida, aunque a un nivel de detalle tal que no se ha considerado oportuno incluirlo en ninguno de los capítulos que conforman la parte central de esta memoria.

Gracias a la ontología DogOnt y a las adiciones realizadas, se ha logrado establecer una taxonomía bastante detallada de todo lo que se ha considerado necesario dentro del dominio objetivo. En la figura B.1 pueden apreciarse los dos primeros niveles de la jerarquía (no se proporciona un diagrama porque se ha considerado que esta forma de mostrar esta información es más clara teniendo en cuenta el gran número de clases a mostrar).

A continuación se proporciona una breve explicación de lo que trata de modelar cada una de las clases que aparecen en B.1:

- *BuildingEnvironment*: esta entidad modela las diferentes estancias que pueden existir en un hogar digital.
- *BuildingThing*: dentro de esta entidad se diferencia entre entidades controlables (sobre las que se puede actuar, o con las que se puede interactuar) y entidades no controlables (muebles y elementos arquitectónicos principalmente, sobre los cuales no se puede interactuar de ninguna forma). Como parte de las entidades controlables se diferencia entre los electrodomésticos y similares (electrodomésticos, equipos destinados al entretenimiento, dispositivos de comunicación, computadores (y sus variantes), etc.) y lo que la ontología DogOnt denomina HousePlants, entidad que engloba todos los dispositivos que se podrían denominar domóticos, y que son los que compondrán los diferentes

sistemas presentes en el hogar digital 1.1 (sistema de climatización, sistema de seguridad, sistema de iluminación, etc.). Para todos estas clases conviene resaltar que se ha definido una propiedad denominada "isIn", y que pretende reflejar la información sobre la estancia en la que está ubicada cada entidad [1]. Otra propiedad que conviene ser mencionada es "hasState", propiedad mediante la cual se puede especificar el estado en el que se encuentra cada individuo 'controlable'.

- *Functionality y Command*: para cada "HousePlant" se especificarán las funcionalidades que se pueden 'ejecutar' sobre ella, y cada funcionalidad podrá traducirse en comandos concretos. Se distinguen varios tipos de funcionalidades: de control, de notificación y de pregunta (para preguntar el valor de una variable concreta de un dispositivo concreto).

- *DateTimeDescription*: clase relativa al tiempo, utilizada para crear individuos relativos a instantes en el tiempo.

- *DayOfWeek*: clase relativa al tiempo, utilizada para modelar los días de la semana.

- *DomoticNetworkComponent*: modelan las características particulares de cada sistema.

- *DurationDescription*: clase relativa al tiempo, utilizada para modelar la duración de un determinado suceso.

- *ElectricLine*: líneas eléctricas disponibles en el hogar digital en cuestión. Conviene resaltar la existencia de la propiedad "isFed", que sirve para identificar a qué línea eléctrica está conectado cada electrodoméstico, ya que esta información es muy interesante para realizar un posible servicio de gestión de cargas. Además, para cada línea existente se puede detallar la carga máxima que puede soportar.

- *DefinedAlarm*: alarmas definidas para controlar los momentos en que ciertos dispositivos toman valores fuera de rangos definidos como "permitidos".

[1] El hecho de que únicamente se citen aquí algunas propiedades y clases no significa que la ontología se reduzca sólo a dichas entidades descritas, sino que se considera que éstas son las más importantes, y que ir detallando todas y cada una de las entidades, propiedades de dato, propiedades de objeto, etc. sólo conseguiría hacer de este anexo algo totalmente inabordable.

- *GeneratedAlarm*: alarmas que se han generado de entre todas las alarmas definidas.
- *Notification*: notificaciones generadas por los equipos.
- *Occupant*: habitantes de la casa. En este caso, conviene resaltar las propiedades "isInhabited" e "isLocatedIn", que identifican, respectivamente, cuál es la habitación de la persona en cuestión y en qué estancia está localizada dicha persona en un determinado momento.
- *Orientation*: utilizada para definir, como un tipo enumerado, posibles orientaciones, que por el momento se han utilizado para especificar la orientación de cada estancia utilizando la propiedad ïsOriented".
- *ScheduledTask*: modelan las tareas que se pueden definir para que un determinado dispositivo. Son de una ocurrencia, se programa y sólo es para un determinado instante. Si se desea definir una tarea que se ejecutará periódicamente en un determinado intervalo de tiempo habrá que utilizar la siguiente entidad.
- *PeriodicTask*: modelan tareas periódicas a ejecutar por un determinado dispositivo en un intervalo de tiempo determinado.
- *State*: para modelar los estados de los equipos, dispositivos, etc. Se diferencia entre estados continuos y discretos.
- *StateValue*: valor de los estados modelados por la entidad anterior. Tanto para los estados como para sus valores se considera que pueden ser discretos o continuos.
- *TemporalEntity*: clase relativa al tiempo, utilizada para modelar entidades temporales que pueden ser instantes en el tiempo o intervalos de tiempo, muy útiles para la parte de tareas y alarmas.
- *TimeZone*: clase relativa al tiempo que puede ser interesante si quieren considerarse valores temporales relativos a diversas zonas del mundo.

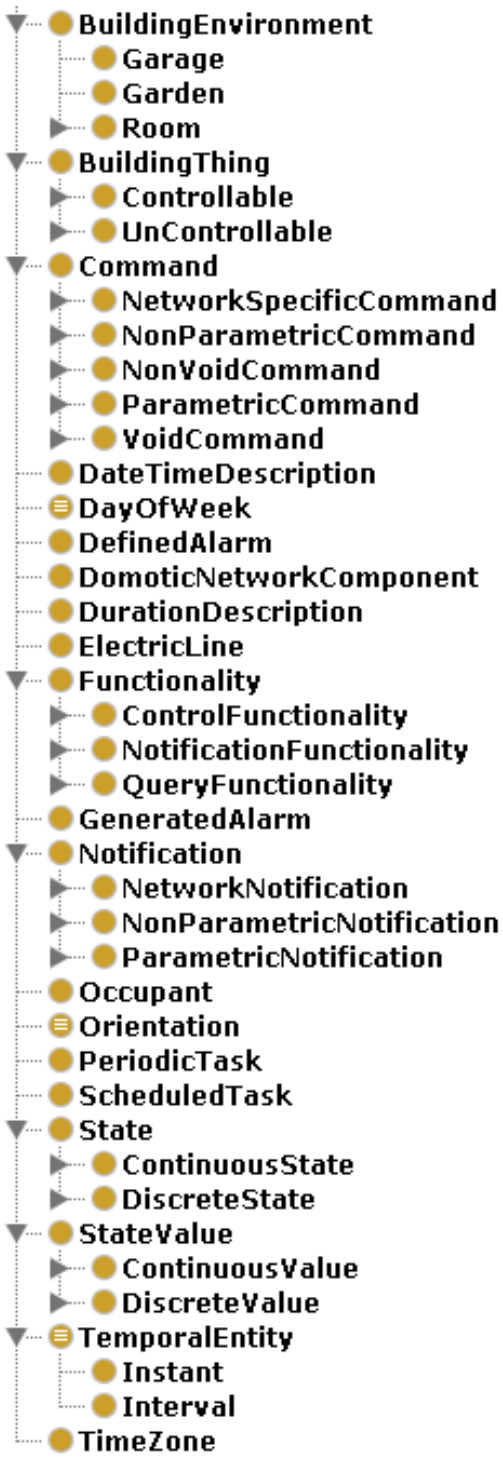

Figura B.1: Clases correspondientes a los dos primeros niveles de la ontología final

Bibliografía

[1] Tam Van Nguyen, Wontaek Lim, Huy Anh Nguyen, Deokjai Choi, and Chilwoo Lee. Context ontology implementation for smart home. *CoRR*, abs/1007.1273, 2010.

[2] Asunción Gómez Pérez. Ontological engineering: A state of the art. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.8567`. Última visita: Febrero de 2012.

[3] Robert Neches, Richard Fikes, Tim Finin, Thomas Gruber, Ramesh Patil, Ted Senator, and William Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):36–56, August 1991.

[4] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.

[5] Willem Nico Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, Enschede, September 1997.

[6] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *In Proceedings of the Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society, 1994.

[7] Roberto Romero Llop. *Especificación OWL de una Ontología para Teleeducación en la Web Semántica*. PhD thesis, Universidad Politécnica de Valencia. Departamento de Comunicaciones, 2007.

[8] S. Bechhofer, I. Horrocks, and P. Patel-Schneider. Tutorial on owl. `http://neon-toolkit.org/wiki/SPARQL`. Última visita: Mayo de 2012.

[9] Xml. `http://www.w3.org/XML/`. Última visita: Mayo de 2012.

[10] Rdf. `http://www.w3.org/RDF/`. Última visita: Mayo de 2012.

[11] Owl. http://www.w3.org/TR/owl-ref/. Última visita: Mayo de 2012.

[12] Rio+20: El futuro que queremos. http://www.un.org/es/sustainablefuture/cities.shtml. Última visita: Marzo de 2012.

[13] Observatorio Tecnológico de la Energía. Mapa tecnológico ciudades inteligentes. http://www.idae.es/uploads/documentos/documentos_Borrador_Smart_Cities_18_Abril_2012_b97f8b15.pdf. Última visita: Mayo de 2012.

[14] Ifc. http://buildingsmart.com/standards/ifc. Última visita: Abril de 2012.

[15] gbxml. http://www.gbxml.org/aboutgbxml.php. Última visita: Abril de 2012.

[16] Proyecto europeo fiemser. http://www.fiemser.eu/. Última visita: Abril de 2012.

[17] Harry Chen. *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. PhD thesis, University of Maryland, Baltimore County, 2004.

[18] Anind K. Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD thesis, College of Computing, Georgia Institute of Technology, 2000.

[19] Xiao Hang Wang, Da Qing Zhang, Tao Gu, and Hung Keng Pung. Ontology based context modeling and reasoning using owl. *Pervasive Computing and Communications Workshops, IEEE International Conference on*, 0:18, 2004.

[20] Harry Chen, Tim Finin, and Anupam Joshi. The soupa ontology for pervasive computing. In *Ontologies for Agents: Theory and Experiences*, pages 233–258. BirkHauser, 2005.

[21] T Gu, X H Wang, H K Pung, and D Q Zhang. An ontology-based context model in intelligent environments. *Work*, 2004(6 Suppl 1):270–275.

[22] Dario Bonino and Fulvio Corno. DogOnt - Ontology Modeling for Intelligent Domotic Environments. pages 790–803. 2008.

[23] Jingjing Xu, Yann-Hang Lee, Wei-Tek Tsai, Wu Li, Young-Sung Son, Jun-Hee Park, and Kyung-Duk Moon. Ontology-based smart home solution and service composition. In Tianzhou Chen, Dimitrios N. Serpanos, and Walid Taha, editors, *ICESS*, pages 297–304. IEEE, 2009.

[24] Mario J. Kofler, Christian Reinisch, and Wolfgang Kastner. An Intelligent Knowledge Representation of Smart Home Energy Parameters. In *Proceedings of the World Renewable Energy Congress (WREC '11)*, May 2011. pages 921–928.

[25] Dae-man Han and Jae-hyun Lim. Smart home energy management system using ieee. *Energy*, 56(3):1403–1410, 2010.

[26] A. Daouadji, K. K. Nguyen, M. Lemay, and M. Cheriet. Ontology-Based Resource Description and Discovery Framework for Low Carbon Grid Networks. pages 477–482, 2010.

[27] Mariano Fernandez-Lopez, Asuncion Gomez-Perez, and Natalia Juristo. Methontology: from ontological art towards ontological engineering. In *Proceedings of the AAAI97 Spring Symposium*, pages 33–40, Stanford, USA, March 1997.

[28] York Sure and Rudi Studer. On-to-knowledge methodology, 1999.

[29] H. Sofia Pinto, Steffen Staab, and Christoph Tempich. Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In *In Proceedings of the 16th European Conference on Artificial Intelligence (ECAI*, pages 393–397. IOS Press, 2004.

[30] María del Carmen Suárez de Figueroa. *NeOn Methodology for Building Ontology Networks: Specification, Scheduling and Reuse*. PhD thesis, Universidad Politécnica de Madrid. Facultad de Informática. Departamento de Inteligencia Artificial, 2010.

[31] Ontology engineering group. universidad politécnica de madrid. `http://www.oeg-upm.net/`. Última visita: Mayo de 2012.

[32] Glosario neon methodology. `http://mayor2.dia.fi.upm.es/oeg/files/pdf/NeOnGlossary.pdf`. Última visita: Mayo de 2012.

[33] Metodología neon: Planificación. http://www.neon-project.org/web-content/media/book-chapters/Chapter-07.pdf. Última visita: Mayo de 2012.

[34] Uso de recursos no ontológicos en la metodología neon. http://www.neon-project.org/web-content/media/book-chapters/Chapter-08-1.pdf. Última visita: Mayo de 2012.

[35] Metodología neon: Reutilización de recursos ontológicos. http://www.neon-project.org/web-content/media/book-chapters/Chapter-09.pdf. Última visita: Mayo de 2012.

[36] Metodología neon: Reutilizar ontologías enteras. http://www.neon-project.org/web-content/media/book-chapters/Chapter-10.pdf. Última visita: Mayo de 2012.

[37] Metodología neon: Reutilizar partes de una ontología. http://www.neon-project.org/web-content/media/book-chapters/Chapter-11.pdf. Última visita: Mayo de 2012.

[38] Neon toolkit. http://neon-toolkit.org/. Última visita: Mayo de 2012.

[39] Esquema escenarios metodología neon. http://mayor2.dia.fi.upm.es/oeg/index.php/en/methodologies/59-neon-methodology. Última visita: Mayo de 2012.

[40] Ontology design patterns.org. http://ontologydesignpatterns.org/wiki/Ontology_Design_Patterns_._org_%28ODP%29. Última visita: Mayo de 2012.

[41] Metodología neon. especificación de requisitos. http://www.neon-project.org/web-content/media/book-chapters/Chapter-05.pdf. Última visita: Mayo de 2012.

[42] M. Grüninger and M. Fox. Methodology for the Design and Evaluation of Ontologies. In *IJCAI'95, Workshop on Basic Ontological Issues in Knowledge Sharing, April 13, 1995*, 1995.

[43] Paulo C. Barbosa Fernandes, Renata S. S. Guizzardi, and Giancarlo Guizzardi. Using goal modeling to capture competency questions in ontology-based systems. *JIDM*, 2(3):527–540, 2011.

[44] Lista buscadores ontologías. http://www.w3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/SemanticWebSearchEngines. Última visita: Mayo de 2012.

[45] Pellet: Owl 2 reasoner. http://clarkparsia.com/pellet/. Última visita: Mayo de 2012.

[46] Gontt. http://neon-toolkit.org/wiki/Gontt. Última visita: Mayo de 2012.

[47] Radon. http://neon-toolkit.org/wiki/RaDON. Última visita: Mayo de 2012.

[48] Sparql. http://www.w3.org/TR/rdf-sparql-query/. Última visita: Mayo de 2012.

[49] Plugin sparql para neontoolkit. http://neon-toolkit.org/wiki/SPARQL. Última visita: Mayo de 2012.