SISTEMA INTELIGENTE DE CONTROL DOMOTICO, BASADO EN HABITOS DE COMPORTAMIENTO

ANEXOS

OSCAR JAVIER JIMÉNEZ CASTILLO
RICARDO ANTONIO GALLEGO FERNÁNDEZ

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
INGENIERÍA EN AUTOMÁTICA INDUSTRIAL
POPAYÁN
2008

SISTEMA INTELIGENTE DE CONTROL DOMOTICO, BASADO EN HABITOS DE COMPORTAMIENTO

ANEXOS

OSCAR JAVIER JIMÉNEZ CASTILLO
RICARDO ANTONIO GALLEGO FERNÁNDEZ

Trabajo de grado presentado para optar al título de
Ingeniero en Automática Industrial

Director

Ph.D. CESAR ALBERTO COLLAZOS ORDOÑEZ

UNIVERSIDAD DEL CAUCA
FACULTAD DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES
INGENIERÍA EN AUTOMÁTICA INDUSTRIAL
POPAYÁN
2008

TABLA DE CONTENIDO

ANEXO A: METODOLOGÍAS PARA LA CONSTRUCCIÓN DE ONTOLOGÍAS

La construcción de ontologías para representar el conocimiento acerca de un domino, es un proceso que no se encuentra aun bien documentado y en el cual no existe una metodología de facto para el desarrollo y mantenimiento de ontologías [1].

Cada grupo de investigación de esta disciplina plantea su propio método o metodología para la construcción de ontologías basada en su experiencia propia. Sin embargo existen varias metodologías que han sido adoptadas por muchos proyectos para el desarrollo de ontologías y que han ganado reconocimiento dentro de este proceso, entre las más populares usadas en la construcción de ontologías se tienen:

1. La metodología de Uschold y King

El método de Uschold y King [2] propone cuatro actividades:

- Identificar el propósito de la ontología: se debe tener claro porque se construirá la ontología y cuales serán sus posibles usos.

- Construir la ontología: esta etapa se puede dividir en tres pasos:

    - Capturar el conocimiento: que incluye (1) identificación de los conceptos claves y las relaciones en el dominio de interés. (2) realizar definiciones precisas para cada uno de los conceptos y relaciones identificados. (3) identificar términos para referirse a tales conceptos y relaciones. Para el desarrollo de esta etapa proponen tres estrategias para identificar los conceptos principales: una aproximación top-down, en la cual se identifican primero los conceptos mas abstractos, y luego, especializarlos en conceptos mas específicos; una aproximación bottom-up, en la cual los conceptos mas específicos se identifican primero y luego son

generalizados en los conceptos mas abstractos; y la aproximación middle-out, en la cual se identifican primero los conceptos más importantes y luego se especializa o se generaliza dentro de otros conceptos.

- Codificación: que se refiere a representar explícitamente el conocimiento adquirido en un lenguaje formal.

- Integración de ontologías existentes: puede surgir el interrogante de cómo y si se debe utilizar ontologías que ya existen.

- Evaluación: se realiza una valoración técnica de la ontología, de su ambiente software asociado y de la documentación con respecto al marco de referencia.

Esta metodología es independiente de la aplicación como el proceso de desarrollo de la ontología es independiente del propósito de la ontología.

2. La metodología de Gruninger y Fox

Esta metodología esta basada en el desarrollo de sistemas basados en conocimiento usando lógica de primer orden, la cual propone [3].

- Escenarios de motivación: se definen escenarios que presentan problemas que se encuentran en dominio particular. Estos escenarios tienen a menudo la forma de narración de problemas o mediante ejemplos.

- Preguntas informales de competencia: Estas permiten describir los requerimientos de la ontología basados en los escenarios propuestos y presentados en forma de preguntas que la ontología debe ser capaz de responder.

- Terminología: los objetos, propiedades y relaciones entre ellos son formalmente descritos. Esta debe suministrar la terminología suficiente para responder las preguntas informales de competencia.

- Preguntas formales de competencia: una ves las preguntas de competencia han sido definidas informalmente y la terminología de la ontología ha sido definida, las preguntas de competencia son definidas formalmente como un problema de vinculación o consistencia con respecto a los axiomas en al ontología.

- Especificación de axiomas: los axiomas que especifican la definición de términos y restricciones sobres sus interpretaciones son dadas en lógica de primer orden.

- Teoremas de amplitud: se definen las condiciones bajo las cuales las soluciones a las preguntas están completas.

3. La Metodología de Amaya Berneras Et al

Desarrollada dentro del proyecto Esprit KACTUS [4] propone los siguientes pasos para el desarrollo de ontologías:

- Especificación de la aplicación: en donde se describe el contexto de la aplicación y una visión de los componentes que la aplicación intenta modelar.

- Diseño preliminar: los términos y tareas propuestos en la etapa anterior son usados como entradas para obtener varios puntos de vista del modelo global. Este proceso involucra la búsqueda de ontologías desarrolladas para otras aplicaciones, las cuales pueden servir al desarrollo de la nueva aplicación.

- Estructuración y refinamiento de la ontología: se usan los principios de mínimo compromiso ontológico para asegurar que los módulos desarrollados no son muy dependientes uno del otro y son tan coherentes como sea posible, buscando obtener máxima homogeneidad dentro de cada modulo.

4. METHONTOLOGY

Esta metodología propone una serie de actividades a llevarse a cabo para el proceso de desarrollo de la ontología [5], las cuales se clasifican como:

a. Actividades de administración del proyecto:

- Planeación: identifica cuales han de ser desarrolladas, como se organizaran, cuánto tiempo y que recursos necesitan para su ejecución.

- Control: garantiza que las tareas planeadas se completen en la forma que fueron propuestas.

- Aseguramiento de la calidad: asegura que la calidad de la ontología, del software y de la documentación sea satisfactoria.

b. Actividades orientadas al desarrollo:

- Especificación: se determina por que se desarrolla la ontología, cuales son los posibles usos de esta y quien será el usuario final.

- Conceptualización: estructura el conocimiento acerca del dominio como modelos significativos a nivel de conocimiento.

- Formalización: lleva el modelo conceptual a un modelo formal o semi-computable (entendible por una maquina).

- Implementación: se construyen modelos computables en un lenguaje computacional.

c. Actividades de soporte: incluye una serie de actividades, que se desarrollan de forma paralela a las actividades orientadas al desarrollo, estas actividades comprenden

- Adquisición del conocimiento: aquí se adquiere conocimiento de un dominio dado.

- Evaluación: se realiza una evaluación técnica de las ontologías, su entorno de desarrollo y de la documentación desarrollada durante cada fase de su ciclo de vida.

- Integración: esta se lleva a cabo cuando se reutiliza otras ontologías existentes para el desarrollo de la nueva aplicación

- Documentación: documentación detallada y clara de cada uno de las fases completadas y de los productos generados.

- Administración de configuración: registra todas las versiones de la documentación, software y ontologías para controlar los cambios.

Metodología de desarrollo

Teniendo en cuenta el estudio realizado de las metodologías propuestas para el desarrollo de ontologías, que nos permitan representar el conocimiento acerca del dominio de interés de una forma explícita y formal, se plantea la siguiente metodología para la construcción de nuestra ontología:

1. **Identificación del propósito y del alcance:**

Se trata de identificar el propósito de la ontología, describiendo el dominio, los objetos de interés y las tareas que van a realizarse, y el alcance de la ontología incluyendo los ítems a ser representados y las características requeridas.

Se sugiere comenzar el desarrollo de una ontología definiendo su dominio y alcance. Es decir, respondiendo a varias preguntas básicas como por ejemplo:

- ¿Cuál es el dominio que la ontología cubrirá?
- ¿Para qué usaremos la ontología?
- ¿Quién usará y mantendrá la ontología?

Las respuestas a estas preguntas pueden cambiar durante el proceso de diseño de la ontología pero en cualquier momento ellas ayudaran a limitar el alcance del modelo.

El producto de esta primera etapa es el desarrollo de un documento de especificación de requerimientos, el cual incluye el dominio de la ontología, el fin para el cual fue desarrollada y quiénes serán los usuarios finales de esta.

2. Adquisición del conocimiento

En esta etapa se recopila la información necesaria sobre el dominio de interés, desde las diferentes fuentes disponibles para tal fin: internet, manuales, documentación bibliográfica; además de la asesoría de docentes e ingenieros expertos en el tema.

Cabe aclarar que la adquisición del conocimiento es un proceso que se lleva a cabo durante todo el ciclo de vida de la ontología y más aun, durante todo el ciclo de vida del proyecto [40]

3. Considerar la reutilización de ontologías existentes

Casi siempre vale la pena considerar lo que otra persona ha hecho y verificar si podemos refinar y extender recursos existentes para nuestro dominio y tarea particular. Reutilizar ontologías existentes puede ser un requerimiento si nuestro sistema necesita interactuar con otras aplicaciones que ya se han dedicado a ontologías particulares o vocabularios controlados.

Muchas ontologías se encuentran disponibles en forma electrónica y pueden ser importadas dentro del entorno de desarrollo de la ontología. El grado de formalidad en el cual esta expresada la ontología muchas veces no interesa, puesto que muchos sistemas de representación de conocimiento pueden importar y exportar ontologías.

4. Conceptualización

Se definen términos importantes tales como: conceptos, instancias, verbos, relaciones o propiedades más relevantes dentro del dominio de interés. Para realizar esta tarea se han definido los siguientes pasos:

4.1. Declaración de términos importantes en la ontología

Inicialmente, es importante obtener una lista general y completa de términos sin preocuparse por los conceptos que estos representan, relaciones entre los términos, o cualquier propiedad que los conceptos puedan tener, o si los conceptos son clases o slots. Los siguientes dos pasos (desarrollar la jerarquía de clases y definir las propiedades de los conceptos (slots)) están estrechamente relacionadas. Es difícil hacer primero uno de ellos y luego hacer el otro. Lo que generalmente se hace es, crear unas cuantas definiciones de los conceptos en la jerarquía y luego continuar describiendo las propiedades de esos conceptos y así sucesivamente.

4.1.1. Definición de clases y jerarquía de clases

Existen varios enfoques para desarrollar una jerarquía de clases propuestos por (Uschold96), pero dentro de este trabajo se aplicara el proceso de desarrollo **top-down,** el cual inicia con la definición de los conceptos más generales en el dominio y a continuación la especialización de esos conceptos

Hay que tener en cuenta que no hay una jerarquía de clases correcta para un dominio dado. La jerarquía depende de los posibles usos de la ontología, el nivel de detalle que es necesario para la aplicación, preferencias personales, y algunas veces requerimientos de compatibilidad con otros modelos. Por eso a la hora de definir clases es útil detenerse y verificar si la jerarquía desarrollada va de acuerdo a los requerimientos.

4.1.2. Definición de propiedades

Una vez se hayan definido algunas de las clases, se describe su estructura interna definiendo sus propiedades o slots, ya que las clases por si solas no proveerán información suficiente para responder las preguntas definidas en la etapa de identificación del propósito y del alcance de la ontología.

En el paso anterior se establecieron las clases de la ontología y es posible que la mayoría de los términos restantes definidos en el paso de declaración de términos importantes sean propiedades de esas clases. En general, hay varios tipos de propiedades que pueden llegar a ser slots en una ontología: [37]

- Propiedades intrínsecas: se refiere a esas propiedades que son propias de unas cuantas clases.
- Propiedades extrínsecas: son propiedades generales para todas o gran parte de las clases.
- Partes: si el objeto está compuesto por otros objetos; pueden ser “partes” físicas y abstractas.
- Relaciones con otras clases: son las relaciones entre miembros individuales de una clase y otros ítems.

4.1.3. Definición de las restricciones de las propiedades (facets)

Las propiedades pueden tener diferentes restricciones que describen por ejemplo el tipo de valor, valores admitidos, el número de los valores (cardinalidad), y otras características de los valores que los slots pueden tomar.

Describiremos tres de las restricciones más comunes; la cardinalidad, los valores permitidos y dominio y rango de un atributo.

- Cardinalidad: la cardinalidad de un slot define cuantos valores puede tomar un atributo. Algunos sistemas solamente distinguen entre cardinalidad simple

(admitiendo a lo sumo un valor) y cardinalidad múltiple (admitiendo cualquier cantidad de valores).

- Tipo de valores: son todos aquellos tipos de valores que puede tomar un atributo o slot. Algunos de los tipos de valores más comunes son:
    - *String* es el tipo de valor más simple, el valor es una simple cadena de caracteres.
    - *Number* describe slots con valores numéricos.
    - Los slots del tipo *Boolean* son simples banderas si/no
    - Los slots del tipo *Enumerados* especifican una lista de valores admitidos para el slot.
    - Los slots del tipo *Instance* admiten la definición de relaciones entre clases. Los atributos con tipo de valor Instance deben también definir una lista de clases admitidas de las cuales pueden venir las instancias [37].

- Dominio y rango de un atributo: Las clases definidas para los slots de tipo Instance son a menudo llamadas rango de un atributo. Se puede restringir el rango de un slot cuando el slot esta unido a una clase particular. Las clases a las cuales un slot esta unido o las clases cuyas propiedades son descritas por un slot son llamadas dominio del slot.

4.1.4. Definición de propiedades o slots especiales

Durante el desarrollo de ontologías es importante tener en cuenta propiedades especiales como las siguientes:

- Atributos inversos: hace referencia al valor de un slot que depende del valor de otro slot, es decir si tenemos un artículo que fue producido por una empresa, entonces esa empresa produce ese artículo, tener esta información en ambos, sentidos seria redundante, pero es necesario al inicio del modelado conocer toda la información posible.

- Valores por defecto: cuando se tiene que un valor particular de un slot es el mismo para la mayoría de las instancias de una clase, es posible definirlo como valor por defecto para ese slot, así cuando se crea una nueva instancia de una clase que contenga este slot, el sistema lo llenara con el valor por defecto automáticamente.

El resultado del trabajo desarrollado hasta ahora es un documento que describe el modelo del dominio de interés, expresado en lenguaje natural el cual podrá ser refinado durante todo el ciclo de vida de la ontología.

5. Implementación

En esta etapa de la construcción de la ontología surgen varias preguntas relacionadas con la herramienta a ser utilizada para la implementación, algunos de los criterios a tener en cuenta para la selección de la herramienta adecuada son [38]:

- ¿La herramienta soporta las actividades del proceso de desarrollo de la ontología?
- Las posibilidades de navegación visual dentro del modelo de conocimientos.
- ¿Vienen incorporados motores de inferencia y herramientas de consulta?
- ¿Cuáles formatos o lenguajes de ontologías genera la herramienta?
- Las facilidades de extracción de la información
- El soporte de ontologías superiores
- La capacidad de importar y exportar, lenguajes de representación de conocimientos ajenos, para la reutilización de ontologías.
- ¿En donde se almacenan las ontologías (en bases de datos o archivos)?

Después de estudiar y evaluar los diferentes lenguajes de representación y herramientas para la edición de ontologías (ver anexos) se determina cuál es el más conveniente para el desarrollo del proyecto y así realizar la implementación de la ontología.
El resultado que se obtiene al final de esta etapa, debe ser una ontología implementada en un lenguaje formal determinado, algunos de los lenguajes más comúnmente usados son citados y descritos en los anexos.

6. Evaluación

Después de desarrollar la ontología se realiza una evaluación y una validación de la ontología para verificar inconsistencias, redundancia y completitud de la misma. Esta evaluación se puede hacer a través de la colaboración de un experto que debe evaluar el modelo y emitir sus observaciones o también mediante la integración de la ontología en el sistema completo para observar su comportamiento y su desempeño. [38]

El resultado es un documento con las observaciones del experto y con una descripción de los resultados de la implantación de la ontología en el sistema global.

7. Documentación

Esta etapa debe desarrollarse paralela a la realización de las etapas anteriores. Debe incluirse los documentos que son el resultado de cada etapa, la evaluación realizada, el conocimiento adicional para usarla, etc. También ha de ser indexada y colocada con las ontologías existentes para su posible reutilización.

ANEXO B: HERRAMIENTAS PARA LA EDICIÓN DE ONTOLOGÍAS

El propósito principal de este anexo es dar un vistazo general a algunas de las herramientas disponibles en la actualidad para el desarrollo de ontologías, y describir brevemente aquellas herramientas que tienen mayor acogida en el campo de la representación del conocimiento a través de ontologías, buscando con esto seleccionar la herramienta adecuada para la construcción de la ontología.

1. Ontolingua

El servidor Ontolingua fue la primera herramienta ontológica creada por el Laboratorio de Sistemas de Conocimiento de la Universidad de Stanford, para facilitar el desarrollo de ontologías con una interfaz basada en aplicaciones Web. Este editor herramienta permite importar y exportar otros formatos como: DAML+OIL, KIF, OKBC, LOOM, Ontolingua, CLIPS, Prolog, y da la posibilidad de importar bases de conocimiento en Protégé, pero no exportarlas a este formato. Ontolingua usa el modelo OKBC con axiomas KIF, el lenguaje base es ontolingua, que consiste de la combinación de marcos y lógica de primer orden, permitiéndole representar clases. Estas clases se encuentran organizadas en taxonomías, relaciones, funciones, axiomas formales, e instancias. La principal diferencia con otros editores es que los usuarios de ontolingua deben tener alguna noción de KIF y del lenguaje Ontolingua para hacer uso de esta herramienta. La interfaz no es muy amigable y puede confundir al usuario en algunas ocasiones. El editor de la ontología le permite al usuario explorar, crear y editar ontologías usando un explorador Web [6].

2. Protégé

Protégé es una de las herramientas de desarrollo de ontologías más usada, desarrollada por la Universidad de Stanford. Protégé es una plataforma libre, de código abierto con un conjunto de herramientas para la construcción de modelos de dominio y aplicaciones basadas en conocimiento con ontologías. En su núcleo, Protégé implementa un conjunto

de estructuras de modelado del conocimiento y acciones para soportar la creación, visualización y manipulación de ontologías en diversos formatos de representación. Protégé puede personalizarse para proporcionar soporte a la creación de modelos de conocimientos e introducción de datos. Además, puede ser ampliado por medio de plugins e Interfaces de programación de aplicaciones basadas en Java (API) para la construcción de herramientas y aplicaciones basadas en conocimiento [7].

La plataforma Protégé soporta dos formas principales de modelar ontologías:

- El editor de marcos-Protégé permite a los usuarios crear y rellenar ontologías basadas en marcos, de acuerdo con el protocolo Open Knowledge Base Connectivity (OKBC). En este modelo, una ontología consta de un conjunto de clases organizadas en una jerarquía de componentes para representar los conceptos principales de un dominio, una serie de slots asociados a las clases para describir sus propiedades y relaciones, y un conjunto de instancias de esas clases, ejemplares individuales de los conceptos que mantienen valores específicos de sus propiedades. Las principales características incluyen:

    - Un extenso conjunto de elementos de interface de usuario que permite a los usuarios modelar conocimiento e ingresar datos de un dominio.
    - Una arquitectura plug-in que puede ser extendida con elementos personalizados tales como componentes gráficos (graficas y tablas), media (sonido, imagen, y video), varios formatos de almacenaje (RDF, XML, HTML, y bases de datos finales) y herramientas de soporte adicionales (administración de ontologías, visualización de ontologías, inferencias y razonamiento, etc.)
    - Interfaces de programación de aplicaciones basadas en Java (API) que hace posible acceder, usar y desplegar ontologías creadas con Marcos-Protégé

- El editor de OWL-Protégé - permite a los usuarios crear ontologías para la Web Semántica. "Una ontología OWL puede incluir descripciones de clases, propiedades y sus instancias. Dada una ontología, la semántica formal de OWL

especifica la forma de obtener sus consecuencias lógicas, es decir, hechos no literalmente presentes en la ontología, sino que se derivan de la semántica. Este permite a los usuarios:

- Cargar y grabar ontologías en OWL y RDF.
- Editar y visualizar clases y propiedades.
- Definir características de clases lógicas como expresiones OWL.
- Ejecutar razonadores tales como clasificadores de lógica descriptiva.

3. WebODE

WebODE es la contraparte Web de ODE (Ontology Design Environment) [8] desarrollado por el Laboratorio de Inteligencia Artificial de la Universidad Técnica de Madrid (UPM), esta herramienta esta basada en la metodología de desarrollo de ontologías Methontology ampliamente utilizada y probada, por lo que soporta la mayoría de actividades propuestas por esta metodología sin que le impida ser usado o seguir otras. WebODE no es usada como una aplicación standalone, pero si como un servidor Web con una interface Web.

Las ontologías de WebODE pueden ser integradas también con otros sistemas usando sus servicios automáticos de importación y exportación (WebODE's XML, RDF(S), OIL, DAML+OIL, OWL, X-CARIN, Java/Jess, UML, Prolog), edición de axiomas, documentación, evaluación e incorporación de ontologías, las ontologías en WebODE pueden ser almacenadas en una base de datos relacional la cual es acezada por medio de al estándar JDBC ((Java Database Connectivity)). WebODE contiene un editor de ontologías, un sistema de administración de conocimiento basado en ontologías (ODEKM), un portal generador automático de Web Semántica (ODESeW), una herramienta de anotación de recurso Web (ODEAnnotate), y una herramienta de servicio de Web Semántica (ODESWS) [9].

4. OntoEdit

OntoEdit [10] fue desarrollado por el Knowledge Management Group del Instituto AIFB de la Universidad de Karlsruhe. Esta herramienta permite crear, explorar, mantener y administrar ontologías. Actualmente el suceso de OntoEdit es OntoStudio el cual es el producto comercial basado en el ambiente de desarrollo Eclipse de IBM.

OntoEdit posee una poderosa interfaz de usuario que le permite representar jerarquías de conceptos, relaciones, dominios, rangos, instancias y axiomas, soporta F-Logic (Lógica Fuzzy), RDF Schema y OIL. El modelo de datos que maneja OntoEdit es OXML 2.0 el cual esta basado en marcos. OXML es definido en XML-Schema. Además de conceptos, relaciones e instancias, el modelo puede representar predicados, instancias de predicados y axiomas.

La representación interna del modelo de datos puede ser exportado a DAML+OIL, F-Logic, RDF(S), and OXML. Adicionalmente las ontologías pueden ser exportadas a bases de datos relacionales vía JDBC. El motor de inferencia que utiliza OntoEdit es OntoBrocker [23]. OntoBroker es el resultado de varios años de investigación y este es ahora un producto comercial. Como Protégé, OntoEdit esta basado en una arquitectura plug-in, lo que le permite al usuario extender OntoEdit con diferentes funcionalidades [11].

5. OilEd

OilEd es un editor de ontologías que permite al usuario construir ontologías basadas en DAML+OIL. El propósito inicial de OilEd fue facilitar un editor simple que despertara el interés en el lenguaje OIL [12].

La versión actual de OilEd no suministra un ambiente completo para el desarrollo de ontologías, no soporta activamente el desarrollo de ontologías a gran escala en términos de migración e integración de ontologías, argumentación y muchas otras actividades que están involucradas en la construcción de una ontología. El formato de datos interno de OILED es DAML+OIL, pero se puede importar ontologías de DAML+OIL y exportar como RDFS, DAML OWL, RDF/XML y otros formatos.

Pero los usuarios de OilEd pueden conectarse al motor de inferencia FaCT que proporciona características de chequeo de consistencia y clasificación de conceptos automática. OilEd también proporciona opciones de documentación (HTML, visualización grafica de ontologías, etc.) [13].

6. DOE

DOE (Differential Ontology Editor) es un editor simple desarrollado por el Instituto INA ((Institut National de l'Audiovisuel - France). DOE únicamente permite la parte de especificación del proceso de desarrollo de ontologías haciéndolo más bien un complemento de otros editores.

DOE es un prototipo desarrollado en Java que puede importar formatos RDFS y OWL, y exportar a CGXML, DAML+OIL, Texto plano OIL, OIL XML, RDFS, RDF/XML, y OWL. DOE usa XSLT para generar interoperabilidad. XSLT es u lenguaje para transformar documentos XML en otros documentos XML. La herramienta DOE no tiene soporte para usuarios, tampoco plug-ins ni manual de usuario.

En el sitio web de DOE existe un instalador del producto completo para Windows. El usuario llena un formulario para acceder a la descarga. Para ejecutar DOE en otra plataforma se requiere una plataforma Java2(TM), una Versión de Edición Estándar 1.3 o posterior [14].

Para los interesados en conocer un poco más sobre las herramientas para el desarrollo de ontologías se adjunta la siguiente lista de herramientas y su sitio web.

Apollo	http://apollo.open.ac.uk/index.html
DAMLImp (API)	http://codip.grci.com/Tools/Components.html
Differential Ontology Editor (DOE)	http://opales.ina.fr/public/

Disciple Learning Agent Shell	http://lalab.gmu.edu/
DUET	http://codip.grci.com/Tools/Tools.html
Integrated Ontology Development Environment	http://www.ontologyworks.com/
IsaViz	http://www.w3.org/2001/11/IsaViz/
JOE	http://www.cse.sc.edu/research/cit/demos/java/joe/
KAON (including OIModeller)	http://kaon.semanticweb.org/
LegendBurster Ontology Editor	http://www.georeferenceonline.com/
LinKFactory Workbench	http://www.landc.be/
Medius Visual Ontology Modeler	http://www.sandsoft.com/products.html
OilEd	http://oiled.man.ac.uk/
Onto-Builder	http://ontology.univ-savoie.fr/
Ontolingua with Chimaera	http://www.ksl.stanford.edu/software/ontolingua/
Ontopia Knowledge Suite	http://www.ontopia.net/solutions/products.html
Ontosaurus	http://www.isi.edu/isd/ontosaurus.html
OntoTerm	http://www.ontoterm.com/
PC Pack 4	http://www.epistemics.co.uk/

Protégé	http://protege.stanford.edu/index.html
RDFAuthor	http://rdfweb.org/people/damian/RDFAuthor/
RDFedt	http://www.jan-winkler.de/dev/e_rdfe.htm
SemTalk	http://www.semtalk.com/
Specware	http://www.specware.org/
TOPKAT	http://www.aiai.ed.ac.uk/~jkk/topkat.html
WebOnto	http://kmi.open.ac.uk/projects/webonto/

ANEXO C: LENGUAJES DE REPRESENTACIÓN DE ONTOLOGÍAS

El propósito principal de este anexo es dar un vistazo general a algunos de los lenguajes de representación de ontologías utilizados por muchas de las herramientas que se encuentran actualmente para el desarrollo de ontologías, y describir brevemente algunas de sus características principales [15].

1. OWL

- OWL Soporta un lenguaje de Ontologías Web.
- OWL esta construido en el nivel superior de RDF.
- OWL es empleado para el procesamiento de la información en la Web.
- OWL fue diseñado para ser interpretado por computadores.
- OWL no fue diseñado para ser leído por personas.
- OWL está escrito en XML.
- OWL tiene tres sub lenguajes.
- OWL es un estándar Web.

Por que emplear OWL?

OWL es un lenguaje de ontologías Web, diseñado para ser usado en aplicaciones que necesitan procesar el contenido de la información, en lugar de una simple presentación de información. Owl facilita una mayor interoperabilidad del contenido Web que el soportado por XML, RDF, y RDF Schema proporcionando vocabulario adicional junto con una semántica formal. OWL tiene tres sub lenguajes cada vez más expresivos: OWL Lite, OWL DL, Y OWL Full. Este facilita una mayor interoperabilidad del contenido Web que puede ser soportado por XML, RDF y RDF Schema que provee vocabulario adicional junto con una semántica formal. OWL esta basado en los lenguajes anteriores OIL y DAM+OIL y es ahora una recomendación de W3C. [16]

Una ontología OWL incluirá descripciones de clases propiedades y sus instancias. Dada una ontología, la semántica formal OWL especifica como derivar esta en consecuencias lógicas, por ejemplo características no presentadas literalmente en la ontología pero conllevadas con la semántica. Esta vinculación puede estar basada en un documento sencillo o en documentos múltiples distribuidos que han sido combinados usando mecanismos OWL definidos.

Una de las ventajas de las ontologías OWL puede ser la disponibilidad de herramientas que son la razón de ello. Las herramientas proveerán soporte genérico que no es específico para un sujeto de dominio particular.
El lenguaje OWL provee tres sub-lenguajes diseñados para el uso de comunidades de ejecutores y usuarios. [17]

- OWL Lite, esta diseñado para los usuarios que tan sólo piden posibilidades de clasificación jerárquica de conceptos (clases) de la ontología y simples restricciones. Por ejemplo, aunque OWL proporciona restricciones de cardinalidad, sólo permite valores 0 ó 1. Por tanto posee una complejidad formal inferior a la que OWL DL tiene.

- OWL DL (Descripción Lógica) este lenguaje esta indicado para los usuarios que requieren mayor expresividad pero exigiendo completitud computacional (se garantiza que todas las conclusiones son computables y que todos los cálculos acaban en un tiempo finito). Incluye todos los constructores de OWL, pero sólo se pueden usar con restricciones; por ejemplo: mientras una clase puede ser a la vez una subclase de muchas clases, no puede ser una instancia de otra clase.

- OWL Full esta dirigido para aquellos usuarios que necesitan la máxima expresividad y libertad sintáctica de RDF pero sin garantías computacionales. Permite, por ejemplo, aumentar el significado de vocabulario predefinido (en RDF o en OWL), por lo que es poco probable que ningún software de razonamiento sea capaz de soportar razonamiento completo para cualquier característica de OWL Full.

OWL Full se puede ver como una extensión de RDFS, mientras que OWL Lite y OWL DL se pueden ver como extensiones restringidas de RDF. Cualquier documento OWL (Lite, DL, Full) es un documento RDF, y cualquier documento RDF es un documento OWL Full pero sólo algunos documentos RDF serán documentos OWL Lite o OWL DL legales. Por esta razón, hay que tener cuidado cuando se desea migrar un documento RDF a OWL. [18]

2. RDF

El lenguaje RDF (Marco de Descripción de Recursos, del inglés Resource Description Framework) es una plataforma que permite procesar meta datos; brindando interoperabilidad entre aplicaciones que intercambian información legible por máquina en la Web. RDF facilita el procesamiento automatizado de los recursos Web, puede utilizarse en áreas de aplicación tales como la *recuperación de recursos* brindando mejores prestaciones a los motores de búsqueda, facilitando el intercambio y para compartir conocimiento. [19]

El lenguaje RDF provee un método flexible para descomponer cualquier conocimiento en partes pequeñas, con ciertas reglas semánticas. A estas se las llama "triples" y están compuestas por sujeto, predicado (o verbo) y objeto.

RDF provee una serie de elementos que pueden ser utilizados para que las máquinas procesen el conocimiento en sí mismo, en lugar de texto, usando procesos similares a los deductivos del hombre, obteniendo resultados de búsqueda más significativos varios millones de veces más rápido que en la actualidad.

Este modelo se basa en la idea de convertir las declaraciones de los recursos en expresiones con la forma sujeto-predicado-objeto. El elemento de construcción básica en RDF es el "triplete" o sentencia, que consiste en dos nodos (sujeto y objeto) unidos por un arco (predicado), donde los nodos representan recursos, y los arcos propiedades. El objeto es el valor de la propiedad o el otro recurso con el que se establece la relación.
El objetivo central que tiene RDF es el de definir un mecanismo para la descripción de recursos que no cree ninguna asunción sobre un dominio de aplicación particular, ni

defina la semántica de algún dominio de aplicación. Esta definición debe ser neutral con respecto al dominio, sin embargo el mecanismo deberá adecuarse para la descripción de información sobre cualquier dominio.

Modelo RDF Básico

RDF se basa en un modelo que representa propiedades designadas y valores de propiedades. El modelo RDF se fundamenta en principios establecidos por varias comunidades de representación de datos. Las propiedades RDF pueden recordar a atributos de recursos y en este sentido corresponden con los tradicionales pares de atributo-valor. Las propiedades RDF representan también la relación entre recursos y por lo tanto, un modelo RDF puede parecer un diagrama entidad-relación. En la terminología del diseño orientado a objetos, los recursos hacen referencia a los objetos y las propiedades a objetos específicos y variables de una categoría. [20]

El modelo de datos de RDF es una forma de sintaxis-neutral para representar expresiones RDF. La representación del modelo de datos es empleada para evaluar equivalencia en el significado. Dos expresiones RDF son equivalentes si y sólo si sus representaciones en el modelo de datos son las mismas. Esta definición de equivalencia permite algunas variaciones sintácticas en expresiones sin alterar el significado.

El modelo de datos básico consiste en tres tipos de objetos:

Recursos

Todas las cosas descritas por expresiones RDF se denominan recursos. Un recursos puede ser una página Web completa, parte de una página Web; p. ej. un elemento HTML o XML específico dentro del documento fuente. También puede ser una colección completa de páginas. Un recurso puede ser también un objeto que no sea directamente accesible vía Web, p. ej. un libro impreso. Los recursos se designan siempre por URIs (Uniform Resource Identifiers) más identificadores de anclas opcionales. Cualquier cosa puede tener un URI; la extensibilidad de URIs es introducir identificadores para cualquier entidad imaginable.

Propiedades

Una propiedad es un aspecto específico, característica, atributo, o relación utilizado para describir un recurso. Cada propiedad tiene un significado específico, define sus valores permitidos, los tipos de recursos que puede describir, y sus relaciones con otras propiedades.

Sentencias

Un recurso específico junto a una propiedad denominada, y con el valor de dicha propiedad para tal recurso es una sentencia RDF [RDF statement]. Estas tres partes se denominan, sujeto, predicado y objeto respectivamente. El objeto de una sentencia, es decir, el valor de la propiedad, puede ser otro recurso o pude ser un literal; es decir, un recurso o una string u otro tipo de datos definidos por XML.

Resource Description Framework (RDF)

El lenguaje de Esquema RDF *(RDF Schema)* se basó en investigaciones sobre metadatos que fueron realizadas por comunidades de Bibliotecas digitales.

Un Esquema RDF entrega información debida a la interpretación de las sentencias del modelo de datos RDF y especifica las restricciones que se deben seguir por éstos.

La especificación del Esquema RDF no tiene como objeto aspectos teóricos, sino que tiene como enfoque la solución de un pequeño número de problemas inmediatos. Sus creadores prevén que otros problemas compartirán características similares y que también pueden ser capaces de utilizar clases básicas que son descritas en esta especificación.

3. OIL

Ontology Inference Layer tiene sintaxis XML y está definido como una extensión de RDF Schema, fue el primer lenguaje para la representación de ontologías que tuvo como base los estándares W3C. OIL emplea la Lógica Descriptiva (declaración de axiomas o reglas) para la representación del conocimiento. [21]

OIL esta estructurado por varias capas de sublenguajes. La capa base de OIL coincide plenamente con RDF Schema, lo que teóricamente implica que los agentes diseñados para RDF Schema, pueden ser reutilizados.

4. DAML+OIL

Este lenguaje fue desarrollado por la cooperación entre OIL y DARPA y unifica los lenguajes DAML (DARPA's Agent Markup Language) y OIL (Ontology Inference Layer).

Este lenguaje esta basado en estándares del W3C. DAML fue desarrollado como una extensión del lenguaje XML y de RDF, y para incrementar el nivel de expresividad que RDF Schema tiene. DAML+OIL comparte muchas características con OIL, pero difiere del modelo basado en clases (frames) y potencia la lógica descriptiva.

DAML+OIL presenta un variado conjunto de elementos que permiten crear ontologías y marcar la información para que pueda ser interpretada por las maquinas.

Este lenguaje tiene algunas deficiencias debido a su complejidad conceptual y de uso, este problema se pretendió solventar con el desarrollo de OWL. No obstante, se desarrollaron muchas aplicaciones que utilizan DAML-OIL y también existen herramientas para convertir DAML a OWL:

5. KIF

Knowledge Interchange Format es un lenguaje para representar ontologías. KIF pretende ser un lenguaje capaz de representar la mayoría de los conceptos y distinciones actuales de los lenguajes más recientes de representación del conocimiento. Se trata de un lenguaje diseñado para intercambiar conocimiento entre sistemas de computación distintos, diferentes lenguas, y no para la interacción entre seres humanos. [22]

ANEXO D: MODELO DE LAS TAREAS DESARROLLADAS POR EL USUARIO

Localidad: Sala

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Socializando
Organización.	Ambiente domótico, Sala.
Objetivo y Valor.	Permitirle al usuario interactuar con otros usuarios, o invitados dentro del ambiente domótico. Por tal motivo el sistema de gestión deberá adecuar esta localidad en términos de iluminación y temperatura.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes Cargar perfiles de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, etc.).
Tiempo y Control.	Frecuencia y Duración: No se podrá definir una frecuencia, dado que esta tarea tiene características imprevistas, y podría realizarse en cualquier momento. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuarios del sistema.

Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 1 TM-1 Descripción refinada de la tarea *Socializando* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Aseo de localidad.
Organización.	Ambiente Domótico, Sala.
Objetivo y Valor.	Realizar limpieza a esta localidad.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta dependerá de las preferencias del usuario, podrían tener una realización periódica no específica. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del sistema.
Conocimiento y	Preferencias ambientales para el desarrollo de la actividad.

Competencias.	
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 2 TM-1 Descripción refinada de la tarea *Aseo de localidad* desarrollada por el usuario

Localidad: Dormitorio.

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Despertar.
Organización.	Ambiente domótico, Dormitorio.
Objetivo y Valor.	Ponerse en disposición para iniciar las actividades del día.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Aseo diario. Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos, hora del sistema. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad tendrá una realización periódica diaria, especialmente en horas de la mañana. La duración estará relacionada directamente con las preferencias del usuario. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.

Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 3 TM-1 Descripción refinada de la tarea *Despertar* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Descansando.
Organización.	Ambiente domótico, Dormitorio.
Objetivo y Valor.	Tomar un descanso a mitad del día con el fin de reponer energía.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos, hora del sistema. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad podrá tener una realización diaria no periódica alrededor del medio día, en épocas de vacaciones o fines de semana, directamente relacionada con las preferencias del usuario. Precondiciones: Dentro del sistema se debe encontrar al menos un usuario registrado. Debe existir presencia dentro de esta localidad específica.

Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 4 TM-1 Descripción refinada de la tarea *Descansando* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Durmiendo.
Organización.	Ambiente domótico, Dormitorio.
Objetivo y Valor.	Descansar después de un día de trabajo, estudio, o actividades diarias.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Despertar. Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos, hora del sistema. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad podrá tener una realización diaria no periódica alrededor del medio día, en épocas de vacaciones o fines de semana, directamente relacionada con las preferencias del usuario. Precondiciones:

	• Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 5 TM-1 Descripción refinada de la tarea *Durmiendo* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Ver Televisión.
Organización.	Ambiente domótico, Dormitorio.
Objetivo y Valor.	Hacer uso del televisor, como medio de entretenimiento o información.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos, hora del sistema. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, televisor, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta tendrá características eventuales, dependiendo de la hora

	como del día, dado que en fines de semana y en vacaciones puede tener una mayor duración. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 6 TM-1 Descripción refinada de la tarea *Ver televisión* desarrollada por el usuario

Localidad: *Cuarto de baño*.

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Aseo diario.
Organización.	Ambiente domótico. Cuarto de Baño.
Objetivo y Valor.	Realizar las actividades relacionadas con el aseo diario.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Despertar. Tareas siguientes: Despertar. Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos, hora del sistema.

	Objetos de salida: Elementos domóticos (calefacción, luces, válvulas de control de agua, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad tendrá una realización periódica diaria en horas de la mañana. En épocas de vacaciones o fines de semana, esta actividad no tendrá una hora específica. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción. • Válvulas de control de agua.
Eficiencia y Calidad.	

Formulario 7 TM-1 Descripción refinada de la tarea *Aseo diario* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Ducharse.
Organización.	Ambiente domótico. Cuarto de Baño.
Objetivo y Valor.	Tomar una ducha al final de una jornada laboral, escolar, etc.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada:

	Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, válvulas de control de agua, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad podrá realizarse de forma esporádica, o bien dependiendo de las preferencias del usuario del sistema. Esta actividad tendrá una mayor realización en horas de la tarde y noche. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción. • Válvulas de control de agua.
Eficiencia y Calidad.	

Formulario 8 TM-1 Descripción refinada de la tarea *Ducharse* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Usar.
Organización.	Ambiente domótico. Cuarto de Baño.
Objetivo y Valor.	Hacer uso de esta localidad.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes:

	Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, válvulas de control de agua, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad será calificada como imprevista, dado que esta localidad podrá ser usada en cualquier momento del día, sin tener un determinado patrón de uso en horas diferentes a las de la mañana. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción. • Válvulas de control de agua.
Eficiencia y Calidad.	

Formulario 9 TM-1 Descripción refinada de la tarea *Usar* desarrollada por el usuario

Localidad: *Cocina.*

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Desayunando.
Organización.	Ambiente domótico. Cocina.

Objetivo y Valor.	Preparar alimentos en horas de la mañana.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Despertar. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, cafetera, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta actividad tendrá una realización periódica; en días de semana se realizará en horas de la mañana alrededor de las 7 am, no obstante en épocas de vacaciones y fines de semana también tendrá una ejecución antes del medio día. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Cafetera.
Eficiencia y Calidad.	

Formulario 10 TM-1 Descripción refinada de la tarea *Desayunar* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Almorzando

Organización.	Ambiente domótico. Cocina.
Objetivo y Valor.	Preparar alimentos en a medio día.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (luces, etc.).
Tiempo y Control.	Frecuencia y Duración: Tendrá una realización posiblemente periódica, esto dependerá de las actividades propias del usuario del sistema, (por ejemplo si nuestro usuario es un empleado que sale en las mañanas y regresa en horas de la noche, esta tarea no tendría lugar en su perfil o agenda). Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Cafetera.
Eficiencia y Calidad.	

Formulario 11 TM-1 Descripción refinada de la tarea *Almorzando* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Cenando
Organización.	Ambiente domótico. Cocina.
Objetivo y Valor.	Preparar alimentos al final del día.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Despertar. Desayunar. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (luces, etc.).
Tiempo y Control.	Frecuencia y Duración: Tendrá una realización posiblemente periódica, esto dependerá de las actividades propias del usuario del sistema, (por ejemplo si nuestro usuario es un empleado que sale en las mañanas y regresa en horas de la noche, esta tarea no tendría lugar en su perfil o agenda). Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad especifica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Cafetera.
Eficiencia y Calidad.	

Formulario 12 TM-1 Descripción refinada de la tarea *Cenando* desarrollada por el usuario

Localidad: *Sala de Video.*

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Usando Dispositivo de Video.
Organización.	Ambiente domótico. Sala de video.
Objetivo y Valor.	Emplear los dispositivos de video como elementos de distracción.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta zona tendrá mayor uso en épocas vacacionales y fines de semana, dado que es cuando los usuarios del sistema tienen tiempo disponible para emplearlo en actividades de entretenimiento. Por otra parte la duración de la misma no se podrá definir con un valor fijo. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia física dentro de esta localidad específica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas.

	• Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 13 TM-1 Descripción refinada de la tarea *Usando dispositivo de video* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Reunión.
Organización.	Ambiente domótico. Sala de video.
Objetivo y Valor.	Hacer uso de esta localidad como centro de reuniones.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta es una tarea de características aleatorias, dado que las reuniones te este tipo son muy esporádicas y en ocasiones no planeadas. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación.

	• Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 14 TM-1 Descripción refinada de la tarea *Reunión* desarrollada por el usuario

Localidad: *Comedor.*

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Tomando alimentos.
Organización.	Ambiente domótico. Comedor.
Objetivo y Valor.	Hacer uso de esta localidad como espacio para tomar los alimentos, ya sea el desayuno, el almuerzo o la cena.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Desayunar. Almorzar. Cenar. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: El un día corriente podrá realizarse varias veces, pero esto dependerá, al igual que otras tareas, de la ocupación, o el perfil que cada usuario pueda tener. Precondiciones: • Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del Sistema.

Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 15 TM-1 Descripción refinada de la tarea *Tomando alimentos* desarrollada por el usuario

Localidad: *Estudio.*

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Estudiando
Organización.	Ambiente domótico. Estudio.
Objetivo y Valor.	Esta actividad esta enfocada al uso primordial de esta zona, para realizar actividades relacionadas con el desarrollo de trabajo mental, lectura, etc.
Dependencia y Flujo.	Tareas precedentes: Identificación de usuario dentro del ambiente. Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos domóticos (calefacción, luces, persianas, etc.).
Tiempo y Control.	Frecuencia y Duración: Esta dependerá de los perfiles de los usuarios registrados dentro del sistema, por ejemplo, si nuestro usuario es un estudiante, la realización de esta tarea será periódica. Precondiciones:

	• Dentro del sistema se debe encontrar al menos un usuario registrado. • Debe existir presencia dentro de esta localidad específica.
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	Preferencias ambientales para el desarrollo de la actividad.
Recursos.	Elementos domóticos que se encuentran dentro de la localidad. Sistema de gestión, encargado de controlar dichos elementos. • Elementos de iluminación. • Persianas. • Dispositivo de calefacción.
Eficiencia y Calidad.	

Formulario 16 TM-1 Descripción refinada de la tarea *Estudiando* desarrollada por el usuario

Localidad: *Puerta de Ingreso.*

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Identificarse.
Organización.	Ambiente domótico. Puerta de Ingreso.
Objetivo y Valor.	Identificarse antes de ingresar al ambiente domótico, con el fin de ejercer un control de acceso.
Dependencia y Flujo.	Tareas precedentes: * Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos de seguridad (cerraduras, alarmas, etc.).
Tiempo y Control.	Frecuencia y Duración:

	Se deberá realizar cada vez que un usuario pretenda ingresar al ambiente domótico, dado que esto se ha establecido como una directiva de seguridad, y para poder identificar a los usuarios registrados del sistema. Precondiciones: *
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	
Recursos.	Elementos de seguridad que se encuentran en esta localidad. Sistema de gestión, encargado de verificar la concordancia entre la información suministrada por el usuario del sistema y la que reposa en la base del conocimiento del mismo.
Eficiencia y Calidad.	

Formulario 17 TM-1 Descripción refinada de la tarea *Identificarse* desarrollada por el usuario

Modelo de Tarea.	**Análisis de Tarea.**
Tarea.	Activar / Desactivar sistemas de seguridad.
Organización.	Ambiente domótico. Puerta de Ingreso.
Objetivo y Valor.	Asegurar el ambiente domótico de personas no admitidas o posibles intrusos del sistema.
Dependencia y Flujo.	Tareas precedentes: * Tareas siguientes: Dependiendo al perfil de usuario.
Objetos Manipulados.	Objetos de entrada: Sensores, elementos domóticos. Objetos de salida: Elementos de seguridad (cerraduras, alarmas, etc.).
Tiempo y Control.	Frecuencia y Duración: Cada vez que en el ambiente domótico se encuentre deshabitado, se deben establecer elementos que permitan

	asegurar la integridad de los elementos que dentro del ambiente se encuentran. Precondiciones: *
Agentes.	Usuario del Sistema.
Conocimiento y Competencias.	
Recursos.	Elementos de seguridad que se encuentran en esta localidad. Sistema de gestión, encargado de verificar la concordancia entre la información suministrada por el usuario del sistema y la que reposa en la base del conocimiento del mismo.
Eficiencia y Calidad.	

Formulario 18 TM-1 Descripción refinada de la tarea *Activar/Desactivar seguridad* desarrollada por el usuario

ANEXO E. DESCRIPCIÓN DE LOS CONCEPTOS DEL DOMINIO

1. DOMINIO DE USUARIO

Concepto: Usuario

Propiedades:

Nombre	**Cardinalidad**	**Tipo**	**Otros Facets**
Edad	Simple	Integer	Min=1, Max=100
Genero	Simple	Symbol	Val={Femenino, Masculino}
Nombre	Simple	String	
Ocupación	Simple	Symbol	Val={Ama de casa, Estudiante, Trab_indep, Empleado, Otros}
Password	Simple	String	
Genera evento	Simple	Instancia evento	
Realiza actividad	Múltiple	Instancia actividad	
Tiene agenda	Simple	Instancia agenda	
Tiene datos personales	Simple	Instancia datos personales	
Tiene preferencias	Múltiple	Instancia preferencias	

Concepto: Agenda

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Involucra actividades	Múltiple	Instancia actividad	

Concepto: Actividad

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Se realiza en	Simple	Instancia intervalo	
Tiene lugar en	Simple	Instancia espacio	

Concepto: Evento

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Afecta un	Simple	Instancia dispositivo	
Genera una	Simple	Instancia respuesta	
Ocurre en	Simple	Instancia espacio	
Sucede en	Simple	Instancia tiempo	

Concepto: Respuesta sistema (acción)

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Controla un	Simple	Instancia dispositivo	

Concepto: Preferencias

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Se aplican en	Simple	Instancia espacio	

2. DOMINIO FISICO

Concepto: Espacio

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Hospeda usuario	Múltiple	Instancia usuario	
Tiene perfil ambiental	Simple	Instancia perfil ambiental	

Concepto: Edificio

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Tiene habitaciones	Múltiple	Instancia habitación	

Concepto: Habitación

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Localizada en	Simple	Instancia edificio	

Concepto: Pasaje

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
De habitación	Simple	Instancia habitación	
A habitación	Simple	Instancia habitación	

Concepto: Tiempo

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Año	Simple	Integer	Min=1920, Max=2100
Mes	Simple	Symbol	Val={Enero, Febrero, Marzo, Abril, Mayo, Junio, Julio, Agosto, Septiembre, Octubre, Noviembre, Diciembre}
Día	Simple requerida	Symbol	Val={lunes, martes, miércoles, jueves, viernes, sábado, domingo, fin semana, en semana, todos días}
Hora	Simple requerida	Integer	Min=0, Max=23
Minuto	Simple requerida	Integer	Min=0, Max=59

Concepto: Intervalo

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Tiempo inicio	Simple	Instancia tiempo	
Tiempo fin	Simple	Instancia tiempo	

Concepto: Época

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Nombre	Simple	Symbol	Val={laboral, vacacional}
Tiempo fin	Simple	Instancia tiempo	
Tiempo inicio	Simple	Instancia tiempo	

Concepto: Perfil ambiental

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Temperatura	Simple	Integer	
Iluminación	Simple	Integer	
Ruido	Simple	Integer	
Humedad	Simple	Integer	

3. DOMINIO DE DISPOSITIVOS

Concepto: Dispositivos domóticos

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Identificador	Simple requerida	String	
Estado	Simple requerida	Symbol	Val={On, Off, Alerta, Programado, Pausado, Falla}
Tipo	Simple	Symbol	Val={sensor, actuador, electrodoméstico, electrodoméstico programable}

4. DOMINO DE EVENTO

Concepto: Evento

Propiedades:

Nombre	Cardinalidad	Tipo	Otros Facets
Afecta un	Simple	Instancia dispositivo	
Genera una	Simple	Instancia respuesta	
Ocurre en	Simple	Instancia espacio	
Sucede en	Simple	Instancia tiempo	

BIBLIOGRAFÍA

[1] Noy N., McGuinness D. “Ontology Development 101: A Guide to Creating Your First Ontology”. Stanford: University. 2000.

[2] Uschold M., King M., “Towards a Methodology for Building Ontologies”, in: IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, 1995.

[3] Gruninger M., Fox M., Methodology for the design and evaluation of ontologies, in: Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, 1995

[4] KACTUS. “The Kactus Booklet”, version 1.0, Esprit Project. 1996. http://www.swi.psy.uva.nl/pjects/NewKACTUS/Reports.html

[5] Fernandez M., Juristo N. “METHONTOLOGY: From Ontological Art Towards Ontological Engineering”. Symposium on Ontological Engineering of AAI, Stanford California. 1997.

[6] http://www.ksl.stanford.edu/software/ontolingua/

[7] http://protege.stanford.edu/

[8] Blazquez M., Fernandez M., Garcia J., Gomez A., “Building ontologies at the knowledge level using the ontology design environment”. In: B.R. Gaines, M.A. Musen (Eds.), 11th International Workshop on Knowledge Acquisition, Modeling and Management (KAW_98) Banff, 1998.

[9] http://webode.dia.fi.upm.es/WebODEWeb/index.html

[10] Sure Y., Erdmann M., Angele J., Staab S., Studer R., Wenke D., “OntoEdit: collaborative ontology engineering for the semantic web”, in: First International Semantic Web Conference (ISWC_02), 2002.

[11] http://www.ontoknowledge.org/tools/ontoedit.shtml

[12] http://oiled.man.ac.uk/

[13] Bechhofer S., Horrocks I., Goble C., Stevens R., **“**OilEd: a reason-able ontology editor for the Semantic Web”, in: Joint German/Austrian conference on Artificial Intelligence (KIÕ01), Lecture Notes in Artificial Intelligence, vol. 2174, Springer, Berlin, 2001.

[14] http://opales.ina.fr/public/

[15] Ontology Language Standardization Efforts
http://www.ontoweb.org/About/Deliverables/d4.0.pdf

[16] http://www.w3.org/TR/owl-guide/

[17] http://www.w3.org/2001/sw/Europe/events/200406-esp/trabajo-final-extratesauros/node5.html

[18] http://en.wikipedia.org/wiki/Web_Ontology_Language

[19] Recuperación y organización de la información
http://serqlsparql.50webs.com/rdf.html

[20] http://www.hipertexto.info/documentos/rdf.htm

[21] http://www.ontoknowledge.org/oil/

[22] http://logic.stanford.edu/kif/dpans.html

[23] Decker S., Erdmann M., Fensel D., Studer R. “Ontobroker: Ontology-based access to distributed and semi-structured information”. In: R. Meersman et al. (eds.), Database Semantics: Semantic Issues in Multimedia Systems, Proceedings TC2/WG 2.6 8th Working Conference on Database Semantics (DS-8), Rotorua, New Zealand, Kluwer Academic Publishers, Boston, 1999. pp. 351–369.

[24] Lenat D., Guha R. “Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project”, Addison-Wesley, Boston, 1990.