# Universidad de las Ciencias Informáticas


**Título:** Método basado en ontología para representar decisiones de diseño de software en la actividad productiva de la UCI

Trabajo final presentado en opción al título de:

Máster en Calidad de Software

**Autor:** Ing. Yordani Cruz Segura

**Tutor:** Dr.C Nemury Silega Martínez

**Diciembre 2018**

**Declaración de autoría**

Declaro que soy el único autor de este trabajo y autorizo al Centro de Informatización de Entidades de la Universidad de las Ciencias Informáticas; así como a dicho centro para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de _____del año_____.

_____

**Ing. Yordani Cruz Segura**

Firma del Autor

_____

**Dr.C Nemury Silega Martínez**

Firma del Tutor

**Agradecimientos**

*En primer lugar a la UCI, a sus creadores y los que han luchado por mantener este maravilloso proyecto.*

*A mis padres y hermanos por sembrar los mejores valores y luchar siempre porque siga adelante.*

*A mi esposa por compartir cada uno de mis sueños y ser el motor impulsor de este resultado.*

*A mis amigos que han estado cerca en las buenas y las malas y me han dado fuerzas para seguir adelante.*

*A mis compañeros de trabajo del Centro CEIGE y de la Facultad 3 por todo su apoyo.*

*A mi tutor Nemury por toda su ayuda y por contribuir en mi formación científica.*

*A mis profesores de toda la vida por haber contribuido en mi educación.*

**Dedicatoria**

*Al mayor regalo que me ha hecho la vida, a esa personita que llegó en el momento indicado para eliminar cualquier estrés con una simple sonrisa, todo mi sacrificio es y será siempre por ti mi princesa Analía.*

**Resumen**

Para lograr el desarrollo de un sistema es necesario transitar por diversas etapas según la metodología de desarrollo que se adopte. Dentro de ellas se encuentra la etapa de diseño, en la cual se toman un conjunto de decisiones encaminadas a incrementar la productividad, reducir los costos de replanificación y retrabajo y obtener sistemas con la calidad suficiente para evitar catástrofes sobre todo en aquellas aplicaciones cuyo entorno de explotación se considere crítico. En la actividad productiva de la Universidad de las Ciencias Informáticas, las decisiones de diseño son descritas en los documentos de arquitectura lo que dificultan la realización de análisis sobre estas, se afecta además la consulta por parte de los arquitectos y diseñadores que cuentan con poca experiencia y la elaboración de resúmenes estadísticos relacionados con esta temática**.** En el presente trabajo se diseñó un método basado en ontología para representar decisiones de diseño con el objetivo de agilizar el análisis que se realice sobre las mismas. Como parte de la validación de los resultados se comprobó el cumplimiento de los requisitos, la calidad del diseño y el sistema lógico formal de la ontología. Se aplicó además la técnica de Iadov para constatar la satisfacción de los usuarios finales y se realizó un cuasi experimento para comprobar la hipótesis de la investigación.

**Palabras clave:** diseño de software, método, ontologías.

**Tabla de contenidos**

## Índice de tablas



## Índice de figuras

## Introducción

La construcción de sistemas de software es una de las disciplinas más prolíficas de los últimos tiempos, debido a los constantes avances tecnológicos y una masiva tendencia hacia la automatización e informatización de tareas. Para lograr el desarrollo de un sistema es necesario transitar por una serie de etapas, constituyendo una buena práctica dedicarle una de ellas al diseño del sistema que se desea desarrollar. Durante el diseño se toman un conjunto de decisiones encaminadas a incrementar la productividad, reducir los costos de replanificación y retrabajo y obtener sistemas con la calidad suficiente como para evitar catástrofes, sobre todo en aquellas aplicaciones cuyo entorno de explotación se considere crítico. Según Somerville [1], el diseño de software es una actividad creativa donde se identifican los componentes del software y sus relaciones, con base en los requerimientos del cliente.

Teniendo en cuenta la estadística brindada por *STANDISH-GROUP* [2] en el 2015 el promedio de proyectos de desarrollo de software exitosos fue de un 29%, retrasados un 52% y fallidos un 19%; dentro de las principales causas manifestadas por las personas encuestadas, que afectan la culminación de manera exitosa de los proyectos, están las relacionadas con los problemas en las especificaciones de requisitos, la falta de participación del usuario y la falta de recursos. A pesar de que las principales causas fueron las mencionadas anteriormente existe un 10.7 % de los encuestados que considera que los proyectos se retrasan debido a que las aplicaciones son desarrolladas con tecnologías que no son lo suficientemente robustas y la aparición de tecnologías más competitivas. Esta arista del problema es una de las decisiones de diseño más importantes que se toma cuando se decide emprender el desarrollo de un sistema [3].

El número de decisiones de diseño y su calidad durante el proceso de desarrollo de un sistema, está condicionado por diversos factores, tales como: alto impacto de la experticia del equipo encargado de desarrollar este proceso, necesidad de adoptar buenas prácticas de proyectos exitosos, las características propias del sistema a desarrollar así como la metodología adoptada para su desarrollo. Por otro lado, existen investigaciones [4-11] relacionadas con el tema donde se reflejan estudios de casos sobre las decisiones de diseño tomadas y el impacto de estas en cada caso. Como consecuencia de los factores mencionados anteriormente se ha evidenciado una dispersión en la bibliografía relacionada con las decisiones de diseño lo que dificulta su utilización en la toma de decisiones por parte de los diseñadores y arquitectos de software.

Gracias al avance de las Tecnologías de la Información y las Comunicaciones (TIC) en el mundo las empresas han apostado su crecimiento al desarrollo de sistemas informáticos que guíen cada uno de sus procesos. Cuba no se ha quedado atrás y en los últimos tiempos se ha enfocado en la informatización de la sociedad cubana, para ello se han creado un conjunto de empresas desarrolladoras de software que soporten la demanda de los diferentes sectores de la sociedad.

Una de las grandes acciones realizadas por el gobierno cubano en aras de impulsar el desarrollo de la informática en el país, fue la creación en el año 2002 de la Universidad de las Ciencias Informáticas (UCI). La UCI fue creada con la misión de formar profesionales comprometidos con su Patria, altamente calificados en la rama de la informática, y producir aplicaciones y servicios informáticos, a partir del vínculo docencia - investigación - producción como modelo de formación, sirviendo de soporte a la industria cubana del software [12]. Con el objetivo de garantizar la calidad en el proceso de desarrollo de software, la UCI basa su actividad productiva en el Modelo de Capacidad y Madurez Integrada (CMMI por sus siglas en inglés) y fue certificada con el nivel 2 en la variante de desarrollo de dicho modelo [13].

Como una de las buenas prácticas adoptadas por la universidad, durante el desarrollo de los proyectos se documentan y generan distintos artefactos que responden a las disciplinas definidas por la metodología utilizada y garantizan el cumplimiento de las áreas de proceso definidas por CMMI para el nivel 2. Toda la documentación y los artefactos generados están organizados en expedientes de proyectos. Dentro de la documentación generada se encuentran las relacionadas con la etapa del diseño y es en estas donde se describen las decisiones de diseño adoptadas por los arquitectos y diseñadores de software.

En una revisión realizada por el autor a los expedientes de proyecto se pudo constatar que en el expediente 4.0, relacionado con las decisiones de diseño se define un total de nueve vistas de arquitectura (vista de datos, vista de despliegue, vista de desarrollo de entorno tecnológico, vista de infraestructura, vista de integración, vista de presentación, vista de procesos, vista de seguridad y vista de sistema) y un documento de modelo de diseño. Las vistas asociadas a la arquitectura pueden ser descritas o no de acuerdo a las características del proyecto a desarrollar. En caso de que no aplique una vista determinada, esta debe de ser justificada debidamente en la guía base de la arquitectura que es otro documento que se genera en esta área. En el artefacto modelo de diseño se reflejan el diagrama de paquetes y el de clases del proyecto correspondiente. Teniendo en cuenta que la UCI se

encuentra enfrascada en la certificación del nivel 3 de CMMI, fue necesario evolucionar el expediente de proyecto a la versión 5.0 donde se adecuaron cada uno de los artefactos con el objetivo de cumplir con las áreas de procesos asociadas a este nivel. En la nueva versión del expediente de proyecto se reducen las vistas de arquitectura a cinco (vista de datos, vista de entorno de desarrollo tecnológico, vista de presentación, vista de proceso y vista de sistema).

En ambos expedientes de proyecto se describen un conjunto de decisiones de diseño como parte de cada una de las vistas. Es válido destacar que con la evolución del expediente de proyecto hay decisiones como las relacionadas con los patrones de diseño y arquitectura y los estilos arquitectónicos a emplear en el proyecto que no fueron incluidas en ninguna de las vistas. Durante la revisión realizada a ambos expedientes se identificaron las siguientes insuficiencias:

- Las decisiones de diseño son descritas por separado en cada una de las vistas correspondientes.
- Solo son descritas aquellas decisiones de diseño que se encuentran en los acápites, desestimando otras como es el caso en el expediente 5.0 de los patrones de diseño y las vistas de arquitectura empleadas en la solución.
- El acceso a los documentos relacionados con las decisiones de diseño se encuentra restringido a roles específicos dentro del proyecto.

Los elementos mencionados anteriormente dificultan la realización de análisis sobre las decisiones de diseño de los proyectos en la universidad, de manera que esta información pueda ser consultada de una forma más ágil y oportuna en la toma de decisiones por parte de los diseñadores y arquitectos, sobre todo en aquellos casos que no cuentan con la suficiente experiencia en el rol. Se dificulta además la elaboración de resúmenes estadísticos relacionados con estos temas a nivel de universidad y la reutilización de decisiones en proyectos similares.

Por lo antes expuesto se determinó el siguiente **problema científico de investigación**: ¿cómo ampliar el proceso de representación de las decisiones de diseño de software en la actividad productiva de la Universidad de las Ciencias Informáticas, de manera que se agilice el análisis sobre las mismas?

El **objeto de estudio** de la presente investigación lo constituyen los sistemas para representar las decisiones de diseño y el **campo de acción** los sistemas basados en ontología para representar decisiones de diseño.

El **objetivo general** que se persigue es desarrollar un método basado en ontología para representar las decisiones de diseño de software en la actividad productiva de la Universidad de las Ciencias Informáticas, de manera que se agilice el análisis sobre las mismas.

Para darle cumplimiento al objetivo general se han definido los siguientes **objetivos específicos:**

1. Elaborar la fundamentación teórica de la investigación relacionada con las principales tendencias en el desarrollo de soluciones relacionadas con la representación de decisiones de diseño de software.
2. Desarrollar una ontología para representar las decisiones de diseño de software.
3. Desarrollar un método que permita representar las decisiones de diseño de software basado en la ontología implementada.
4. Validar la solución propuesta mediante la utilización de métodos científicos de investigación.

Se plantea como **hipótesis** lo siguiente**:** si se aplica un método basado en ontología para representar las decisiones de diseño de software en la actividad productiva de la Universidad de las Ciencias Informáticas, se agilizará el análisis realizado sobre las mismas.

Como herramientas para guiar la investigación fue necesaria la utilización de diferentes **métodos científicos** que se relacionan a continuación:

**Métodos teóricos:**

Analítico – Sintético: posibilitó analizar por partes el objeto para facilitar su estudio mediante la determinación de sus componentes principales (los sistemas para la representación de decisiones de diseño, las técnicas de Inteligencia Artificial (IA), las ontologías y los marcos de trabajo para la manipulación de ontologías) y luego sintetizarlos en el método diseñado.

Hipotético – Deductivo: el presente método se utilizó con el objetivo de darle solución al problema planteado a partir de formular la hipótesis y después a través de reglas lógicas y deductivas, arribar a conclusiones particulares que posteriormente fueron sometidas a comprobaciones empíricas.

Histórico – Lógico: durante el desarrollo de la investigación fue necesario estudiar la evolución que han tenido los conceptos más importantes relacionados con la temática como es el caso del diseño y la arquitectura de software, las ontologías y las decisiones de diseño. Por otra parte, se estudió otras experiencias con puntos de coincidencia con la que aquí se propone.

**Métodos Empíricos:**

Análisis documental: se utilizó en la revisión de la literatura especializada para extraer la información necesaria y obtener los referentes teóricos y conceptuales de la investigación. Por otra parte, fue necesario realizar un estudio de las principales decisiones de diseño que se manejan por diferentes autores para luego seleccionar las más relevantes y representarlas en el método elaborado.

El presente documento se encuentra estructurado, en resumen, introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos. A continuación se reseña el contenido abordado en cada uno de los capítulos:
En el Capítulo 1 se realiza un estudio del estado del arte y se analizan algunos referentes teóricos relacionados con las decisiones de diseño y el uso de ontologías como herramienta para representar las mismas. Se realiza un análisis crítico de investigaciones similares con el objetivo de identificar fortalezas y debilidades de las mismas. Se describen además las herramientas utilizadas en el desarrollo de la solución.

En el Capítulo 2 se presenta el método desarrollado y la ontología como parte del resultado de la investigación. Se describen cada una de las actividades del método y se presentan los diferentes artefactos resultantes del trabajo realizado. Se describen detalladamente los elementos que componen la ontología desarrollada.

En el Capítulo 3 se presentan los resultados de la validación de la solución. Se diseña el esquema de validación de la investigación, donde se muestra la validación de la ontología y del método de manera general. Se exponen los

resultados de satisfacción de los usuarios finales de la investigación y muestra además el experimento realizado con el objetivo de validar el cumplimiento de la hipótesis propuesta.

# Capítulo 1 Fundamentación teórica

## 1.1. Introducción

En el presente capítulo se realiza un análisis de los principales conceptos relacionados con la problemática planteada. Se revisan investigaciones relacionadas con el objeto de estudio para identificar elementos que puedan ser aplicados a la solución que se desea desarrollar. Se analizan además lenguajes, herramientas y metodologías asociadas al desarrollo de ontología y el razonamiento realizado sobre un modelo ontológico.

## 1.2. Diseño de software

El diseño de software es una de las etapas comunes dentro de las metodologías de software existentes en la actualidad, sobre todo en aquellos sistemas que se consideren grandes. Existen diferentes conceptos relacionados con el diseño, Somerville [1] plantea que es una actividad creativa donde se identifican los componentes del software y sus relaciones, con base en los requerimientos de un cliente. En la presente investigación será adoptado el concepto planteado por Pressman [14] quien presenta al diseño de software como el lugar donde las reglas de creatividad, los requerimientos de los participantes, las necesidades de negocio y las consideraciones técnicas se unen para formular un producto o sistema.

Luego de revisar varias investigaciones relacionadas con el diseño de software [3, 5-9, 15], se define decisiones de diseño como todas aquellas sentencias que se adoptan durante la etapa del diseño, partiendo de los requisitos funcionales y no funcionales del software con el objetivo de garantizar el desarrollo de aplicaciones con calidad.

## 1.3. Soluciones para representar decisiones de diseño

Existen diferentes investigaciones relacionadas con la representación de decisiones de diseño, cada una de ellas con particularidades que se derivan del entorno en el que se aplican. A continuación se reflejan los trabajos revisados como parte de la presente investigación:

Una de las soluciones fue la planteada por Giraldo [7], en esta se presenta una ontología que permite representar el conjunto de diagramas estructurales y de comportamiento de un sistema, además de los patrones GRASP y GoF que

se emplean durante la fase del diseño de software. La ontología está enfocada a mostrar conceptos básicos relacionados con el diseño a estudiantes que cursen la carrera de ingeniería de sistemas. Esta solución no contempla otras decisiones importantes como por ejemplo, las tecnologías a emplear durante el desarrollo de la aplicación.

Otro de los estudios relacionados con la temática es el planteado por López [16], se basa en una solución denominada TREx, compuesta por dos ontologías (Toeska y NDR) encargadas de representar las decisiones de arquitectura. Basa su funcionamiento en la carga de manera automática de información descrita por arquitectos en documentos formales, de manera que se permita realizar razonamientos sobre esta información y estos datos puedan ser analizados por otras herramientas. Para la recuperación de información de los documentos se utilizan técnicas de minería de textos. Esta solución solo se centra en describir las soluciones arquitectónicas, dejando fuera aspectos elementales del diseño de manera general como los patrones, herramientas y lenguajes de desarrollo por solo citar algunos.

Resulta interesante la propuesta presentada por Ming [10], el trabajo consiste en la creación de una ontología que muestre las jerarquías presentes en las decisiones de diseño. La solución se aplica fundamentalmente al diseño de proyectos complejos de ingenierías que implican el diseño de sistemas principales y sistemas dependientes. Tiene como objetivo apoyar a los diseñadores que toman decisiones jerárquicas y en red, a capturar y representar el conocimiento asociado. Esta propuesta está dirigida a sistemas desarrollados con un enfoque jerárquico, lo que limita su aplicación a otro tipo de sistemas.

De manera general los enfoques analizados suelen depender de las características del sistema y consideran un bajo número de decisiones de diseño. Estos enfoques generalmente se crean para abordar un problema específico como por ejemplo la jerarquía entre las decisiones.

Resulta importante para la presente investigación, el empleo en estas soluciones de las ontologías como herramientas para representar la información que manejan. Este elemento constituye una fuente de motivación para emplear en la solución de la problemática planteada las ontologías como herramienta que soporte el método diseñado. A continuación se reflejan diversos elementos que caracterizan a las ontologías y sientan las bases para el empleo de las mismas.

### 1.4. Ontologías

Desde la década de los 80 los investigadores de la Inteligencia Artificial (IA), especialmente los del área de la representación del conocimiento, comprendieron la necesidad de emplear ontologías para describir el mundo de manera que pudiera ser comprendido por sistemas inteligentes [17]. Una de las definiciones de las ontologías plantea, que es una especificación formal y explícita de una conceptualización compartida [18]. Otra de las definiciones más acertadas manifiesta que las ontologías brindan una representación consensuada de un dominio específico y legible para ordenadores [19]. Por otra parte Noy [4] define ontología como: una descripción formal explícita de los conceptos (clases) en un dominio de discurso, las propiedades de cada concepto que describen sus rasgos y atributos y las restricciones. En la presente investigación se adoptará el concepto plantado por Lamarca [20] que visualiza a la ontología como un sistema de representación del conocimiento que resulta de seleccionar un dominio o ámbito del conocimiento y aplicar sobre él un método con el fin de obtener una representación formal de los conceptos que contiene y de las relaciones que existen entre dichos conceptos.

Existen diferentes maneras de clasificar las ontologías, una de ellas es la planteada por Guarino [11], quien atendiendo el nivel de generalidad plantea 4 tipos de ontologías:

- Ontolologías de alto nivel: describen conceptos generales, como espacio, tiempo, materia, objeto, etc., que normalmente son independiente de un dominio o problema particular.

- Ontologías de dominio: proporcionan un vocabulario genérico para describir conceptos, y relaciones entre ellos, de un dominio en específico. Los conceptos de una ontología de dominio normalmente son especializaciones de conceptos que aparecen en ontologías de alto nivel. Lo mismo ocurre con las relaciones entre ellos.

- Ontologías de tareas: prescriben el vocabulario relativo a una tarea genérica o actividad, como por ejemplo, diagnosticar, planificar y vender. Para ello al igual que las ontologías de dominio, se especializan los términos introducidos en ontologías de alto nivel.

- Ontologías de aplicación: describen conceptos que dependen de dominios y tareas particulares, y que son especializaciones de ambos tipos de ontologías relacionadas. Con frecuencia, estos conceptos corresponden a roles desempeñados por las entidades de un dominio al realizar ciertas tareas.

Para garantizar la representación del conocimiento, las ontologías están compuestas por los siguientes componentes [21]:

- Conceptos: ideas básicas que se intentan formalizar. Pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, entre otros.

- Relaciones: representan la interacción y enlace entre los conceptos de dominio. Suelen formar la taxonomía de dominio: “subclase de”, “parte de”, “parte exhaustiva de” y “conectado a”.

- Funciones: son un tipo concreto de relación donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, pueden aparecer funciones como categorizar clase, asignar fecha.

- Instancias: se utilizan para representar objetos que pertenecen a un concepto determinado.

- Axiomas: proposiciones que se declaran sobre las relaciones que deben cumplir los elementos de la ontología. Los axiomas, junto a la herencia de conceptos, permiten inferir el conocimiento oculto detrás de una taxonomía de conceptos.

Las ontologías permiten no solo representar el conocimiento, sino que facilitan el razonamiento automático sobre el mismo e intercambiar información con otras herramientas y sistemas inteligentes. Su amplia aplicación está sustentada en el propósito de alcanzar una comprensión común y compartida de algún dominio particular, donde puedan comunicarse las personas y las computadoras, lo cual considera la automatización de procesos [22].

Son diversas las ventajas que tiene el uso de ontologías en sistemas informáticos, dentro de las que se encuentran las que se mencionan a continuación [23]:

- Optimizan la interoperabilidad y extensibilidad de los sistemas informáticos.
- Permiten la preservación del conocimiento persistente de los expertos en cualquier campo de aplicación.
- Permiten independizar el conocimiento del dominio del código del sistema, lo cual favorece la reutilización.
- No requieren información histórica.
- Proveen un método estándar para intercambiar información que incluye semántica entendible por los agentes de software.
- Aumentan la calidad interna, facilitan el mantenimiento, la flexibilidad y la transparencia.

Las características y ventajas de las ontologías mencionadas anteriormente fundamentan la decisión de utilizarlas en la presente investigación como base del método propuesto para representar decisiones de diseño de software.

### 1.4.1. Metodologías para el desarrollo de ontologías

Uno de los elementos principales a tener en cuenta al iniciar el desarrollo de una ontología es la definición de la metodología a utilizar, una vez que se tenga decidido cuál metodología emplear se conocerá cada uno de los pasos a seguir para garantizar el éxito de la ontología desarrollada. Una metodología es un conjunto de métodos y técnicas que determinan que los resultados de un proceso tengan una calidad aceptable [24]. A continuación, se refleja el estudio realizado como parte de la presente investigación para definir la metodología a utilizar.

Una de las metodologías estudiadas fue la planteada por Noy [4] que visualiza el desarrollo de una ontología en siete pasos, no considerando en ninguno de ellos la validación del modelo ontológico a desarrollar:

1. Determinar el dominio y ámbito de la ontología.
2. Determinar la intención de uso de la ontología.
3. Reutilizar ontologías o vocabularios controlados existentes.
4. Enumerar los términos importantes del dominio.

5. Definir jerarquía de clases.

6. Crear las instancias.

Otra de las metodologías revisadas fue Methontology [25], la aceptación de la misma radica en reconocer el desarrollo de las ontología como un proyecto informático, teniendo en cuenta elementos tan importantes como la planificación del proyecto, la documentación y la calidad del mismo. Permite no solo crear ontologías nuevas, sino que propone y facilita la reutilización de trabajos realizados anteriormente. Esta metodología cuenta con los siguientes pasos:

1. Especificación

2. Conceptualización

3. Formalización

4. Implementación

5. Mantenimiento

Teniendo como base las metodologías planteadas anteriormente es creada por Alvarado [24] una nueva metodología que tiene como pilares fundamentales la construcción de un modelo conceptual robusto y la determinación clara y concisa de los requerimientos de la ontología que se desea desarrollar. La metodología está constituida por cinco pasos que se muestran a continuación:

1. Determinar los requerimientos de la ontología.

2. Reutilizar las ontologías o metadatos existentes.

3. Elaboración del modelo conceptual.

4. Implementación de la ontología.

5. Evaluación de la ontología.

Además de los pasos mencionados anteriormente esta metodología define un conjunto de pautas dentro de las que se encuentran las siguientes:

- La elaboración de la ontología es un proceso iterativo, cada versión obtenida se evalúa y se corrige.
- Para el modelo conceptual se elabora un glosario de términos a partir del cual se definen las clases y la jerarquía.

Teniendo en cuenta que esta última metodología se basa en lecciones aprendidas de las metodologías anteriores, propone una actividad para la evaluación del trabajo desarrollado, es sencilla y fácil de aplicar se decidió utilizar la misma durante el desarrollo de la presente investigación.

### 1.4.2. Lenguajes para la representación de ontologías

Desde la década de los años 90 se habla de lenguajes para la representación de ontologías y es aquí donde surgen los conocidos como "lenguajes tradicionales" que fueron diseñados antes de que el internet irrumpiera en los hogares y las empresas. Dentro de estos lenguajes se encuentran los siguientes [26]:

- CML (*Conceptual Modelling Language*): es un lenguaje estructurado para la especificación de modelos de conocimientos. Las ontologías en este lenguaje pueden ser vistas como metamodelos que describen la estructura del conocimiento acerca del dominio. Provee además mecanismos para establecer correspondencia entre distintos niveles de abstracción del conocimiento en una ontología, lo que facilita su construcción por capas, y donde las capas más altas contienen un tipo de conocimiento más abstracto [27].
- KIF (*Knowledge Intercbange Language*): es un lenguaje pensado para el intercambio de conocimiento entre agentes de computadoras, cada uno con el mismo o distinto sistema interno de representación de la información. Cuenta con una semántica declarativa, de tal forma que es posible entender el significado de las expresiones en este lenguaje, sin necesidad de utilizar un intérprete para manipular dichas expresiones, en KIF es posible expresar sentencias arbitrarias de cálculo de predicados, a diferencia también de lenguajes como Prolog SQL. Proporciona mecanismos para representar el conocimiento acerca del propio conocimiento, lo que permite hacer explícitas las decisiones sobre representación del conocimiento, así como introducir nuevos constructores para la representación del conocimiento sin cambiar el lenguaje [28].

- Ontolingua: fue publicado en 1992 por *Knowledge Systems Laboratory* de la Universidad de *Stanfort*. Se trata de un sistema de representación del conocimiento concebido para facilitar el diseño y especificación de ontologías mediante un lenguaje basado en KIF, pero con una semántica más lógica y clara. Ontolingua extiende de KIF con una sintaxis adicional para dar soporte a la construcción de módulos de ontologías que puedan ser ensamblados, extendidos o refinados en una nueva ontología [29].

Con el objetivo de aprovechar las potencialidades de internet a finales de la década de los 90 y principio de la década de 2000, surgen los lenguajes de marcas para representar ontologías. Estos lenguajes son auspiciados por la W3C (*World Wide Web Consortium*) y a continuación se muestran algunos de ellos [26]:

- XML Schema: es un lenguaje que permite definir estructuras de información en documentos XML para que sea conforme a un esquema determinado. También extiende de XML con la posibilidad de especificar tipos de datos. Sin embargo, la semántica acerca del contenido de la información que se puede expresar con este lenguaje es limitada, y queda restringida al ámbito del formato y tipos de datos en un documento. Por este motivo no se puede considerar como un lenguaje de ontología en sentido propio.

- RDF (*Resorce Description Framework*): es un modelo de datos para describir objetos y relaciones entre ellos, con una semántica simple. Fue concebido como un marco de trabajo para representar metadatos acerca de los recursos web. Los modelos RDF pueden representarse mediante lenguajes con una sintaxis XML denominado RDF/XML.

- RDF Schema (RDFS): es un vocabulario para describir propiedades y clases de recursos RDF. Cuenta con vocabulario y semántica para describir jerarquías de generalización de dichas propiedades y clases. Las clases definidas por RDFS pueden representar casi cualquier categoría de cosas, como páginas web, personas, tipos de documentos, conceptos abstractos, entre otras. Las clases son descritas por *recursos rdfs: Class* y *rdf: Resource* y por propiedades tales como *rdf: type* y *rdfs: subClassOf*. Los recursos que pertenecen a una clase son llamados instancias. Así que en RDF Schema, una clase es cualquier recurso que tenga un tipo *rdf: type* como propiedad cuyo valor sea un recurso *rdfs: Class*. Se entiende por la etiqueta *rdf: type* que todo recurso que la ocupe deberá tener las características que distinguen al recurso que se encuentre del lado derecho de la etiqueta [23].

- Lenguaje de ontologías para la WEB (OWL): es un lenguaje que fue diseñado para representar conocimiento complejo acerca de cosas, sus grupos y relaciones. Está basado en lógica computacional de modo que el conocimiento expresado en OWL puede ser razonado por programas de computadoras que además de verificar la consistencia del conocimiento, permiten que el conocimiento implícito se convierta en conocimiento explícito. Los documentos OWL, conocidos como ontologías, pueden ser publicados en la web e incluso hacer referencia a otras ontologías [30]. El lenguaje OWL cuenta con un conjunto de facilidades que lo hacen destacarse por encima del resto de los lenguajes como es el caso de su conjunto de operadores: intersección, unión y negación. Está basado en un modelo lógico que le permite definir los conceptos tal y como son descritos. Además, brinda la posibilidad de utilizar razonadores lo que permite chequear automáticamente la consistencia de los modelos representados [21, 31].

A continuación, se analiza con mayor profundidad el lenguaje OWL, haciendo énfasis en los fundamentos en los que se sustenta y que justificarán su elección como lenguaje de implementación de ontologías en el desarrollo de la presente investigación.

En OWL aparece la unión de clases, a diferencia de RDFS en OWL es posible representar la clase persona como la unión de las clases hombre y mujer. Asimismo, OWL proporciona mayor cantidad de sentencias en el vocabulario que RDFS para describir propiedades, tales como [23]:

- Relaciones entre clases
- Cardinalidad
- Diferentes tipos para las propiedades
- Características de las propiedades
- Clases enumeradas

El lenguaje OWL cuenta con tres sub-lenguajes como se muestra a continuación:

- OWL Lite: permite especificar jerarquías de clasificación y restricciones simples. Para definir restricciones de cardinalidad sobre las relaciones solo permite tomar como valores 0 ó 1. Puede ser un lenguaje útil para representar tesauros o taxonomías reducidas.

- OWL DL: proporciona la máxima potencia expresiva manteniendo la completitud computacional y la decidibilidad (todos los cálculos terminan en un tiempo finito) de los procesos de deducción automática que pueda llevar a cabo un razonador diseñado para este lenguaje.

- OWL Full: pensado para usuarios que deseen la máxima potencia expresiva y la libertad semántica de RDF, pero no brinda garantías computacionales con vista a procesos de razonamiento automático.

Para la presente investigación resulta oportuno el uso de OWL DL, teniendo en cuenta que se desea explotar las garantías computacionales que brinda el mismo y el empleo de los razonadores para garantizar la semántica de la ontología a desarrollar.

### 1.4.3. Herramientas para la creación de ontologías

Un elemento importante a definir antes de iniciar el desarrollo de una ontología es la herramienta a utilizar para la creación de la misma. Para ello, es importante tener en cuenta el lenguaje a emplear y los requerimientos de la ontología a desarrollar, de manera que se escoja una herramienta que cuente con soporte para ambos elementos.

Los editores de ontologías son herramientas especializadas que apoyan la construcción de estas. Las facilidades que proporcionan van desde la definición y modificación de conceptos, propiedades, relaciones, axiomas y restricciones, hasta la inspección y navegación [32].

Existen herramientas tales como Ontolingua Server, Ontosaurus y WebOnto, que son editores de ontologías construidos en los lenguajes Ontolingua, LOOM y OCML respectivamente. Como se mencionaba en el epígrafe anterior estos lenguajes soportados por las herramientas antes mencionadas son lenguajes que han evolucionado a otros con nuevas prestaciones.

Aparejado a la evolución de los lenguajes de programación, se desarrolla una nueva generación de herramientas que integran la tecnología de ontologías en los sistemas de información, tales como Protégé, WebODE, Swoop y

WebOnto, las cuales proveen un entorno de trabajo flexible, gráfico y con más funcionalidades que los mencionados anteriormente. Los modelos de conocimiento subyacentes son independientes del lenguaje y permiten traducir la ontología a distintos lenguajes y formatos. A continuación, se caracterizan cada una de estas herramientas:

- WebODE: es una herramienta para modelar el conocimiento usando ontologías. Fue desarrollada en la Escuela Técnica de Informática (FI) en Madrid y es el equivalente Web de ODE (*Ontology Desing Environment*). Su interfaz basada en formularios y editores gráficos permite guiar la conceptualización y editar taxonomías. Su arquitectura ofrece chequeo completo de la consistencia e importación avanzada de términos [33]. Esta herramienta no cuenta con soporte desde el año 2006, elemento que limita el uso de la misma en la presente investigación.

- Swoop: proyecto de código abierto desarrollado en el laboratorio de Mindswap de la Universidad de Meryland. Se trata de una herramienta para la creación, edición y depuración de ontologías OWL, basado en conceptos de diseño y navegación hipermedia. Su aspecto es similar a un navegador web, pudiendo recorrer los elementos de las diferentes ontologías cargadas a base de activar hipervínculos. Esta herramienta admite varias sintaxis de representación, desde OWL a la sintaxis habitual RDF/XML y permite la búsqueda semántica. Utiliza anotación colaborativa mediante el entorno de trabajo Annotea. Es fácilmente extensible por herramientas de anotación semántica tipo SMORE; además incluye el razonador Pellet integrado, con la posibilidad de depurar la ontología al tiempo que se está creando [34]. No permite el empleo de otros razonadores que no sea el Pellet, limitando la comprobación de las condiciones lógicas formales con diferentes razonadores.

- WebOnto: desarrollado por el *Knowledge Media Institute* (KMI) de la *Open University* (Reino Unido). Aunque es de libre acceso, para acceder y disfrutar de sus potencialidades de edición, los usuarios deben solicitar una cuenta a los administradores. Es un applet (pequeña aplicación que permite obtener una gran variedad de efectos en las páginas web) de Java, trabaja junto a un servidor web personalizado que permite a los usuarios navegar, crear y editar ontologías sin problemas de interfaz. WebOnto ofrece la gestión gráfica de ontologías e inspección de elementos, teniendo en cuenta la herencia de propiedades y el chequeo de consistencia; además, permite la generación automática de instancias a partir de definiciones de clases [33].

Solo cuenta con soporte para el lenguaje OCML, lo que limita su empleo como herramienta para el desarrollo de la ontología para representar las decisiones de diseño, que será desarrollada con OWL como se mencionó anteriormente.

- Protégé: es un software libre de código abierto implementado en Java, desarrollado en la Universidad de *Stanford*, que permite la construcción de ontologías de dominio. Es capaz de operar como una plataforma para acceder a otros sistemas basados en conocimiento o aplicaciones integradas, o como una librería que puede ser usada por otras aplicaciones para acceder y visualizar bases de conocimiento. La herramienta ofrece una interfaz gráfica que permite al desarrollador de ontologías enfocarse en la modelación conceptual sin que requiera de conocimientos de la sintaxis de los lenguajes de salida [32]. Es reconocida como una de las herramientas más utilizadas para la ingeniería ontológica [23, 35] y existen varios estudios que así lo avalan [31, 36-38]. Cuenta con soporte para OWL y permite integrar a su solución varios razonadores como el Pellet, HermiT y Fact++.

Teniendo en cuenta los elementos planteados anteriormente se decidió utilizar en la presente investigación como herramienta para el desarrollo de ontologías el Protégé, a continuación se caracteriza con mayor profundidad esta herramienta.

Protégé funciona como una aplicación local o a través de un cliente en una comunicación con un servidor remoto. El navegador web de Protégé permite a los usuarios compartir, navegar y editar sus ontologías utilizando un navegador web estándar, lo que proporciona un ambiente de colaboración que ayuda a las comunidades en el desarrollo de ontologías. La comunidad de usuarios de Protégé regularmente contribuye a mejorar la calidad del software y participa en grupos de discusión en línea dedicados a formular preguntas, realizar peticiones de nuevas características y cuestiones de soporte técnico. Protégé presenta una gran capacidad de extensión, debido al soporte de conectores (plug-ins), que son aditivos que se adquieren de manera individual y se acoplan al entorno de trabajo de Protégé para añadirle funcionalidad. Existen varios conectores disponibles para importar ontologías en diferentes formatos, incluyendo DAG-EDIT, XML, RDF y OWL. Las herramientas PROMPT, son conectores para Protégé que permiten a los desarrolladores integrar ontologías, trazar los cambios en las ontologías a través del

tiempo y crear vistas de las mismas. A continuación, se presentan algunos conectores de Protégé y las funcionalidades que ofrecen [32]:

- Protégé Web Browser: es una aplicación web basada en Java que permite a los usuarios compartir, navegar y editar sus ontologías utilizando un navegador web estándar; lo que proporciona un ambiente de colaboración que ayuda a las comunidades en el desarrollo de ontologías biomédicas.

- OWL: permite cargar, guardar y editar ontologías OWL en Protégé.

- DAML+OIL: permite crear y editar ontologías DAML+OIL en Protégé. RDF: permite crear, importar y guardar archivos RDF(S) en Protégé.

- Bean Generator: genera los archivos Java, correspondientes a una ontología desarrollada en Protégé, para su posterior uso desde JADE.

- Data Genie: permite importar los datos de una base de datos arbitraria en Protégé.

- Jambalaya: proporciona un ambiente de visualización extensible, flexible, y escalable para la exploración, navegación, y entendimiento de las ontologías. Las clases e instancias son representadas como nodos en un gráfico; los tipos diferentes se distinguen utilizando distintos colores. Las flechas dirigidas (arcos) son utilizadas para mostrar las relaciones entre los conceptos y las instancias.

- Media Slot Widget: permite la inclusión y despliegue de archivos de audio y video en Protégé.

- Prompt: permite manejar múltiples ontologías, comparar versiones de la misma ontología, integrar ontologías y extraer una parte de una ontología.

- PromptViz: crea representaciones visuales de las diferencias entre dos versiones de una ontología.

### 1.4.4. Razonadores

Para extraer más conocimiento que los conceptos almacenados en una ontología, son necesarias otras herramientas conocidas como razonadores. Los razonadores de manera general brindan los siguientes servicios de inferencia [23, 33]:

- Validación de la consistencia de una ontología: el razonador puede comprobar si una ontología no contiene hechos contradictorios.
- Validación del cumplimiento de los conceptos de la ontología: el razonador determina si es posible que una clase tenga instancias. En el caso de que un concepto no sea satisfecho (no pueda tener instancias) la ontología será inconsistente.
- Clasificación de la ontología: el razonador computa a partir de los axiomas declarados y las relaciones de subclases entre todos los conceptos declarados explícitamente a fin de construir la jerarquía de clases.
- Consultas de recuperación de información: a partir de la jerarquía de clases se pueden formular consultas tales como: conocer todas las subclases de un concepto, inferir nuevas subclases de un concepto, conocer las superclases directas de un concepto.

Existen distintos razonadores que permiten inferir conocimiento a partir de un contexto determinado, destacándose entre ellos: los siguientes [33]:

- Pellet: razonador Open Source para OWL-DL construido en JAVA, basado en los algoritmos Tableau desarrollados para Lógicas Expresivas potentes; soporta las nuevas características de la OWL 2.
- FaCT++: tiene licencia GPL y trabaja eficientemente con TBox de ontologías de tamaño grande y mediano; no tiene soporte para otros tipos de datos que no sean string o integer y tampoco para el razonamiento con la A-Box de la ontología.
- Racer: su nombre comercial es RacerPro. No tiene licencias libres. Razonador DL para la lógica descriptiva SHIQ.

- HermiT: es un razonador para ontologías escritas usando el lenguaje de ontología web (OWL). Dado un archivo OWL, HermiT puede determinar si la ontología es consistente o no. Es de código abierto y lanzado bajo licencia LGPL. Soporta las características de OWL y realiza inferencias en ontologías que por su tamaño resultaba difícil o imposible por otros [39].

Teniendo en cuenta los resultados alcanzados por diferentes investigaciones [23, 24, 31] se tomó la decisión de emplear en la presente investigación los razonadores Pellet y HermiT, la utilización de dos razonadores permitirá comprobar con mayor fiabilidad la consistencia de la ontología.

### 1.5. Conclusiones parciales

Una vez concluido el capítulo se arribaron a las siguientes conclusiones:

- El estudio realizado, sobre las soluciones que permiten representar decisiones de diseño contribuyó a demostrar que, cada una de ellas manejan un negocio particular y que no reflejan en su totalidad las decisiones de diseño. Resultó interesante el empleo de ontologías como herramienta fundamental, lo que influyó en la selección de la misma para soportar el método a desarrollar.

- Se identificó que existe consenso en la comunidad científica en cuanto al empleo de OWL como lenguaje para la representación y Protégé como herramienta para la creación; así como el uso de la metodología de Alvarado como metodología para guiar el desarrollo de la ontología de la presente investigación.

# Capítulo 2 Descripción de la propuesta

## 2.1.Introducción

En el presente capítulo se muestra un método basado en ontología para representar las decisiones de diseño en la actividad productiva de la UCI, lo que constituye la propuesta de solución de la presente investigación. Antes de la presentación del método se realiza un análisis bibliográfico sobre investigaciones relacionadas con la temática, con el objetivo de identificar las decisiones de diseño más comunes en la bibliografía, estas decisiones fueron sometidas posteriormente al juicio de experto.

## 2.2. Identificación de las decisiones de diseño

Como parte de la investigación se realizó una revisión sistémica a la bibliografía con el objetivo de identificar cómo se maneja el tema en cuestión por diferentes investigadores que desarrollan la temática. Mediante este método, se identifica, evalúa y combina la evidencia de los estudios de investigación; siendo este un medio para evaluar e interpretar la información relevante disponible asociada a una investigación [40].

Durante el análisis bibliográfico se constata que existen varios trabajos relacionados con el tema [1, 3, 5, 6, 8, 9, 14, 15, 41-46], destacándose la pertinencia de investigaciones y libros publicados hace más de 10 años. Se identificaron 73 publicaciones, de ellas se consideran relevantes para la presente investigación 12. De los trabajos seleccionados 7 son publicaciones en revistas referenciadas, 3 son libros pertenecientes a la literatura clásica relacionada con la materia y 2 son tesis de maestría discutidas en la Universidad de las Ciencias Informáticas.

Las investigaciones consultadas refieren autores de 13 países ubicados en varios continentes, destacándose Francia, España, Estados Unidos y Australia, esta dispersión demuestra el alcance internacional de la problemática. En la Figura 1 se muestra la distribución por países de todos los autores de las investigaciones consideradas como relevantes.

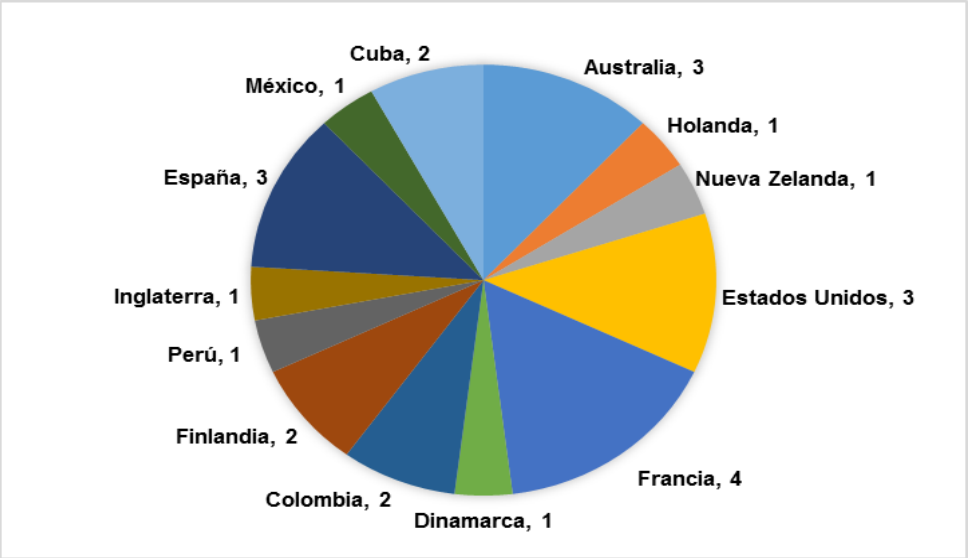



Figura 1. Distribución de autores por países

Como resultado de la revisión bibliográfica realizada se identificaron las 14 decisiones de diseño mostradas en la Tabla 1. Dentro de las decisiones de diseño con mayor porciento de representatividad se encuentran las relacionadas con la definición de los patrones de diseño, los patrones arquitectónicos, los componentes, subsistemas, clases del diseño y sus relaciones.

Tabla 1. Decisiones de diseño identificadas en la bibliografía

| No | Decisiones de Diseño | % | (Van Vliet, 2017) [8] | (Galster, 2017) [6] | (Silva, 2001) [15] | (Leinonen, 2014) [9] | (Jacobson, 2000) [41] | (Pressman, 2010) [14] | (Tibermacine, 2016) [3] | (Chauhan, 2017) [5] | (Somerville, 2011) [1] | (ISO/IEC 12207, 2006)[43] | (Codorniú, 2011)[45] | (León, 2012) [46] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Definir los patrones de diseño | 83.3 | X | X | | X | X | X | X | X | X | | X | X |

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | Definir los patrones de arquitectura | 66.6 | X | X | | | X | X | X | X | X | | X | |
| 3 | Definir la metodología de diseño | 16.6 | X | | X | | | | | | | | | |
| 4 | Definir la estructura del sistema | 75.0 | X | X | X | | | | X | X | X | X | X | X |
| 5 | Definir el modelo navegacional | 16.6 | | | X | X | | | | | | | | |
| 6 | Definir el diseño de la interfaz gráfica | 66.6 | X | | X | X | X | X | X | X | X | | | |
| 7 | Definir la información que se va a almacenar y la estructura de los datos | 58.3 | X | | X | | | X | | | X | X | X | X |
| 8 | Seleccionar las herramientas CASE | 8.3 | | | X | | | | | | | | | |
| 9 | Definir los estilos arquitectónicos | 58.3 | | | | | X | X | X | X | X | X | X | |
| 10 | Definir las vistas de la arquitectura | 50.0 | | | | | X | | X | X | X | | X | X |
| 11 | Definir las herramientas de desarrollo | 25.0 | | | | | X | | | X | | X | | |
| 12 | Definir las interfaces de comunicación interna y externa | 66.6 | X | | | | X | X | X | | X | X | X | X |
| 13 | Definir estándares de codificación y modelado | 25.0 | | | | | X | | | X | | | | X |
| 14 | Definir la forma de | 16.6 | | | | | X | | | | | X | | |

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | distribuir el sistema | | | | | | | | | | | | | |

Luego de contar con las 14 decisiones de diseño resultante de la revisión bibliográfica, se aplica el método de grupo focal con el objetivo de contar con la experiencia de personas capacitadas en la temática. Este método es un tipo de entrevista grupal para recolectar opiniones y conocimientos sobre un tema específico. Su aplicación estimula a los miembros a emitir ideas sobre el objeto en investigación y la interacción entre ellos permite considerar aspectos adicionales o identificar problemas comunes [47].

Para llevar a la práctica el método mencionado anteriormente se conformaron tres equipos de trabajo integrados por profesores, investigadores y especialistas que cuentan con experiencia en la temática y fueron convocados en dos momentos. En el primer encuentro se presentaron las 14 decisiones de diseño identificadas en la bibliografía y cada grupo emitió su criterio relacionado con las decisiones presentadas, proponiendo elementos novedosos que pudieran ser aplicados en la solución. En un segundo momento se convocó a los tres grupos unidos con el objetivo de llegar a un consenso relacionado con los elementos propuestos de manera independiente. La guía de las sesiones de los grupos se muestra en el anexo 1. Como resultado final de la aplicación de este método, se arribó a las siguientes conclusiones:

- Se decidió no tener en cuenta algunas decisiones de diseño que contaban con poca representatividad en la literatura, como es el caso de las relacionadas con la definición de las herramientas CASE, la metodología de diseño, los estándares de codificación y modelado y la forma de distribuir el sistema. Sobre esta última se manifestó que es una decisión que se toma incluso desde la etapa de requisitos.

- Los participantes en los encuentros consideran, que un elemento importante es la relación existente entre las decisiones de diseño. Una decisión puede traer asociada otras como es el caso de la definición del *framework* de negocio que está relacionada con el lenguaje de programación adoptado.

Una vez concluida la aplicación de los métodos mencionados anteriormente se procedió al diseño de un método basado en ontologías para representar decisiones de diseño en la actividad productiva de la UCI.

### 2.3. Descripción del método propuesto.

El método constituye, según las ciencias del diseño, una de las posibles salidas o resultados de las investigaciones relacionadas con los sistemas informáticos. En la bibliografía se reflejan varias definiciones para el método, una de ellas plantea que, lo definen procesos y proveen una guía de cómo solucionar problemas [44]. Vaishnavy [42] lo visualiza como un conjunto de pasos llevados a cabo para desempeñar una tarea. En la presente investigación se consideró adoptar la definición de método planteada por Offermann [48], quien lo define como un conjunto de actividades, posiblemente en un orden, que son realizadas por personas para apoyar el desarrollo de un sistema.

La Figura 2 muestra las seis actividades que incluye el método elaborado para representar decisiones de diseño en la actividad productiva de la UCI: revisar requisitos del software, revisar base de conocimiento, identificar conceptos aplicables, identificar elementos no descritos en la ontología, actualizar la ontología y describir decisiones de diseño.

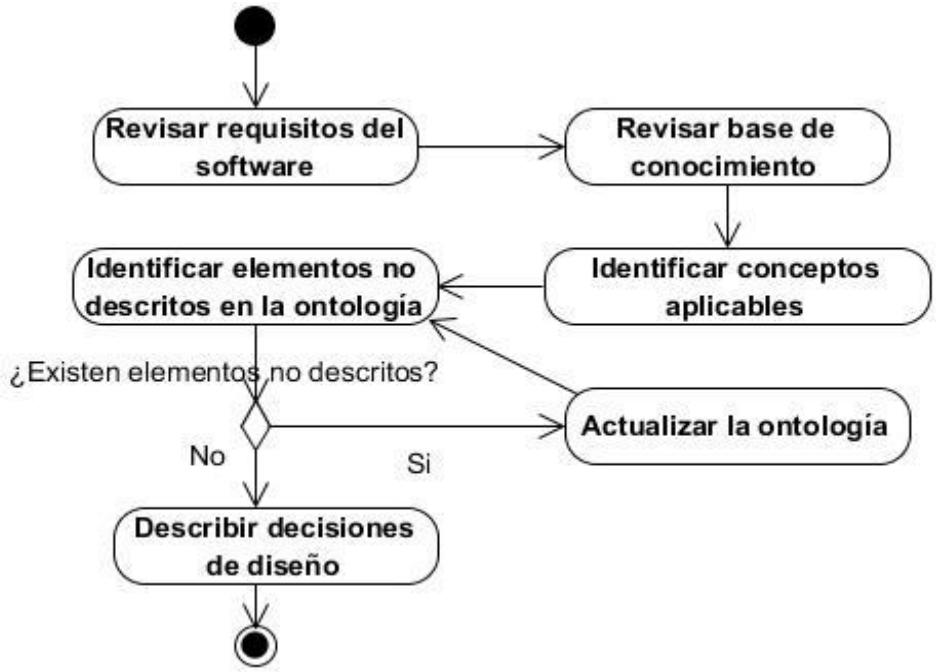



Figura 2. Diagrama de actividades del método propuesto

A continuación, se describen las seis actividades que componen el método diseñado. El método está concebido para aplicarse posteriormente a la etapa del levantamiento de requisitos de software [49, 50]:

- **Revisar requisitos del software**

La presente actividad tiene como artefacto de entrada la descripción de requisitos funcionales y no funcionales del software a diseñar. Durante la misma los diseñadores y arquitectos de software, realizan un estudio detallado de cada uno de los requisitos, de manera que puedan visualizar el diseño y la estructura arquitectónica con la que se va a garantizar el cumplimiento de cada uno de estos.

- **Revisar base de conocimiento**

Para la ejecución de esta actividad, es necesario contar con una base de conocimiento que contenga una representación de decisiones de diseño de diferentes proyectos. Para el caso de la presente solución la base de conocimiento estará constituida por una ontología, desarrollada a partir de los resultados de la revisión bibliográfica y de la aplicación del método del grupo focal. Al ser ejecutado el método por primera vez, en esta actividad solo se revisará la estructura de la ontología con el objetivo de familiarizarse con la misma. Una vez que esta cuente con datos registrados, favorecerá su comprensión y se podrán revisar otros aspectos como es el caso de la semántica de los mismos.

- **Identificar conceptos aplicables**

Luego de ser estudiada la base de conocimiento por los diseñadores y arquitectos, estos deben ser capaces de identificar, de todos los conceptos que se encuentran descritos en la misma, cuáles son los que se pueden aplicar a su solución, teniendo en cuenta los requisitos y las características del software a desarrollar. Como resultado de esta actividad, se cuenta con un listado de los conceptos fundamentales que se encuentran en la base de conocimiento y pueden ser aplicados durante el diseño de su solución.

- **Identificar elementos no descritos en la ontología**

Teniendo en cuenta los requisitos funcionales y no funcionales del sistema y el listado de los conceptos de la base de conocimiento aplicables a la solución, es importante identificar aquellos elementos necesarios para darle solución a los requisitos que no se encuentran modelados en la ontología. Esta actividad es la encargada de identificar nuevas

actualizaciones que posteriormente serán modeladas en la ontología, de manera que se puedan representar nuevas tendencias y se garantice la mejora continua de la base de conocimiento. Como resultado de esta actividad se contará, en caso de que se identifiquen, con un listado de conceptos que deben ser modelados en la ontología.

- **Actualizar la ontología**

La presente actividad se ejecuta solo si se identifican en la actividad anterior nuevos conceptos que deben ser representados en la ontología. Tiene como entrada el listado de conceptos identificados anteriormente y durante la misma se va a actualizar en la base de conocimiento cada uno de los nuevos conceptos identificados, de manera que se puedan describir satisfactoriamente las decisiones de diseño de la solución. Como salida de la actividad se contará con la ontología actualizada con los nuevos conceptos. Una vez que la ontología sea actualizada, se debe revisar nuevamente si existen elementos no descritos en la misma, con el objetivo de garantizar que la herramienta cumpla con todas las funcionalidades necesarias, para representar las decisiones de diseño del sistema que se desea desarrollar.

- **Describir decisiones de diseño**

Una vez concluidas las actividades anteriores, se podrán representar cada una de las decisiones de diseño que se deben de tomar como parte de la solución, de manera que se garantice sentar las bases para enfrentar la etapa de implementación. Esta actividad es la encargada de garantizar que sea poblada la base de conocimiento teniendo en cuenta que, entre más casos sean representados, será más útil a la hora de realizar razonamientos sobre la misma y de apoyar la toma de decisiones por parte del equipo de proyecto.

### 2.4. Ontología para representar decisiones de diseño.

Una vez definidas las decisiones de diseño se realizó un análisis para identificar cada uno de los componentes de la ontología a desarrollar de manera que fueran representadas semánticamente cada una de las decisiones. Para guiar el proceso de desarrollo de la ontología fue utilizada la metodología propuesta por Rubén Darío Alvarado, la cual propone cinco actividades fundamentales: determinación de los requerimientos, reutilización de ontologías,

elaboración del modelo conceptual, implementación y evaluación de las ontologías [24]. A continuación se explican cada una de las actividades propuestas.

### 2.4.1. Determinación de los requisitos

La metodología utilizada propone como primera actividad la definición de los requisitos de la ontología, determinando el alcance y el dominio de la misma, para ello se definen un conjunto de preguntas tales como:

- ¿Qué dominio cubrirá la ontología?
- ¿Para qué se va a emplear la ontología?
- ¿Qué preguntas debería contestar la ontología?
- ¿Quién utilizará y mantendrá la ontología?

Para darle cumplimiento a esta actividad se desarrolló el Documento de Especificación de Requisitos de la Ontología (DERO) que se muestra en el Anexo 2. Estos requisitos serán utilizados en el desarrollo de la ontología y posteriormente en la validación se verificará el cumplimiento de los mismos.

### 2.4.2. Reutilización de ontologías

Como elemento importante antes de iniciar el desarrollo de una ontología, la metodología utilizada propone la reutilización, con el objetivo de identificar elementos dentro de ontologías ya desarrolladas que puedan ser empleados y que agilicen el trabajo a realizar. Para facilitar la reutilización se han creado repositorios de ontologías relacionadas con diferentes dominios, como parte de la presente investigación fueron revisados los repositorios *DAML Ontology Library* (Disponible en: *http://www.daml.org/ontologies/*), *Suggested Upper Merged Ontology* (Disponible en: *http://www.adampease.org/OP/*) y la *DBpedia* (Disponible en: *http://mappings.dbpedia.org/server/ontology/classes/*). En la revisión realizada se identificó que cada una de las ontologías desarrolladas describe conceptos relacionados al dominio que representan y no fueron identificados elementos que pudieran emplearse como parte de la solución.

### 2.4.3. Elaboración del modelo conceptual

Una vez revisado los posibles elementos a reutilizar como tercer paso la metodología propone la elaboración del modelo conceptual. En la presente investigación los conceptos fueron identificados a partir de la revisión bibliográfica realizada que posteriormente fue sometida a la técnica del grupo focal. En la Tabla 2 se presenta el glosario de términos de los principales conceptos que serán empleados en el desarrollo de la ontología con una breve descripción del significado de cada uno en el contexto de la investigación.

Tabla 2. Glosario de términos

| **Nombre** | **Descripción** |
|---|---|
| Sistema | Sistema informático que permite almacenar y procesar información. |
| Estilos arquitectónicos | Conjunto de reglas de diseño que identifican las clases de componentes y conectores que se pueden utilizar para componer un sistema o subsistema, junto con las restricciones locales o globales [51]. |
| Estructura del sistema | Forma mediante la cual se decide organizar el sistema que se desea desarrollar, puede ser por subsistemas, componentes y clases. |
| Interfaz de comunicación interna | Mecanismo mediante el cual se garantiza la comunicación entre los diferentes componentes de la solución. |
| Interfaz de comunicación externa | Mecanismo mediante el cual se garantiza la comunicación del sistema desarrollado con otros sistemas. |
| Patrones de arquitectura | Son una descripción abstracta estilizada de buena práctica, que se ensayó y puso a prueba en diferentes sistemas y entornos. De este modo, un patrón arquitectónico debe describir una organización de sistema que ha tenido éxito en sistemas previos [1]. |
| Patrones de diseño | Son una descripción del problema y la esencia de su solución, de modo que la solución puede reutilizarse en diferentes configuraciones [1]. |
| *Framework* | Estructura base que permite el desarrollo rápido con soporte para un |

| | |
|---|---|
| | lenguaje determinado. En la presente investigación se manejan tres tipos de *framework*: de negocio, de interfaz y de acceso a datos. |
| Lenguaje de programación | Conjunto de órdenes o comandos que describen el proceso deseado. Cada lenguaje tiene sus instrucciones y enunciados verbales propios, que se combinan para formar los programas de cómputo. Son herramientas que permiten construir y adecuar aplicaciones. |
| Sistema gestor de base de datos | Conjunto de programas que permiten crear y mantener una base de datos. |
| Sistema operativo | El conjunto de programas informáticos que permite la administración eficaz de los recursos de los dispositivos electrónicos que utilizan microprocesadores [52]. |
| Vistas de arquitectura | Son las encargadas de representar cómo un sistema se descompone en módulos, cómo interactúan los procesos de tiempo de operación o las diferentes formas en que los componentes del sistema se distribuyen a través de una red. Para el diseño y la documentación, por lo general se necesita presentar múltiples vistas de la arquitectura de software [1]. |
| Decisión | Son todas aquellas sentencias que se adoptan durante la etapa del diseño, partiendo de los requisitos funcionales y no funcionales del software con el objetivo de garantizar el desarrollo de aplicaciones con calidad. |
| Autor | Persona encargada de tomar la decisión. |
| Estructura del sistema | Forma organizativa con la que va a contar el sistema a desarrollar. |

Teniendo como base el glosario de términos representado anteriormente, se definieron las clases, relaciones y axiomas que permitieron implementar la ontología. En la Tabla 3 se representa un listado de cada una de las clases de la ontología.

Tabla 3. Clases de la ontología

| | | |
|---|---|---|
| Autor | Patrones de arquitectura | Vistas de arquitectura |
| Estilos arquitectónicos | Patrones de diseño | Interfaz de comunicación externa |
| Estructura del sistema | *Framework* | Sistema |
| Interfaz de comunicación interna | Lenguaje de programación | Tipo de decisión |
| Sistema gestor de base de datos | Sistema operativo | Decisión |
| *Framework* de negocio | *Framework* de interfaz | *Framework* de acceso a datos |
| Interfaces de comunicación | Tecnologías | Patrones |
| Autor | | |

Una vez identificadas las clases de la ontología, se estableció la jerarquía entre las mismas como se muestra en la Figura 3.

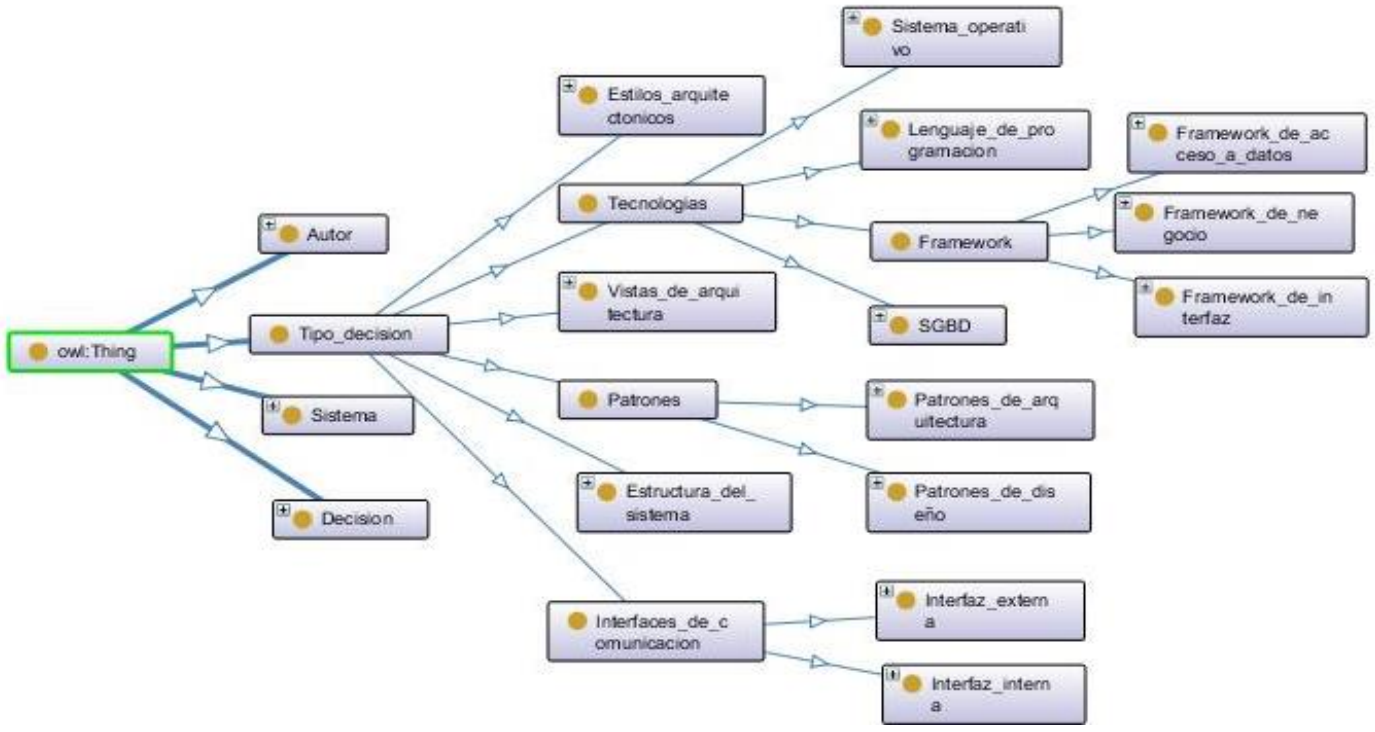



Figura 3. Jerarquía de clases

Luego de ser identificadas las clases y representada la jerarquía entre las mismas, se establecen las relaciones existentes entre ellas como se muestra en la Figura 4. La relación existente entre las clases *Decision y Tipo_decision* subsume un conjunto de relaciones que soportan la jerarquía de clases que se representa en la Figura 3, un ejemplo de ello son las relaciones *es_estilo_arquitectonico_de* y *tiene_estilo_arquitectonico* pertenecientes a las clases *Estilos_arquitectonicos y Decision*.

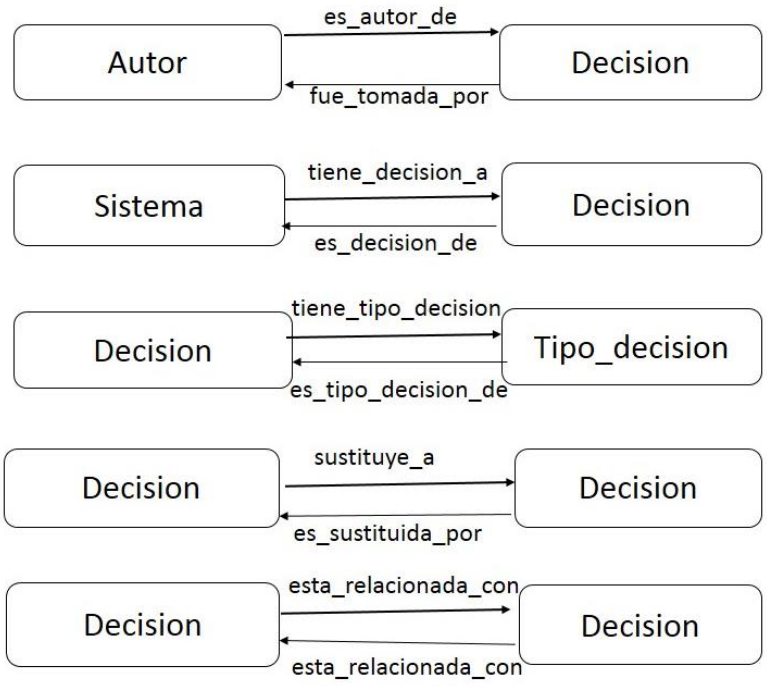



Figura 4. Diagrama de relaciones binarias de la ontología

Contando con una definición clara de las clases, la jerarquía entre estas, sus relaciones y atributos, se procede a definir el modelo conceptual que constituirá la base para la implementación de la ontología que se desea desarrollar. En la Figura 5 se muestra el modelo conceptual desarrollado, está constituido por 21 conceptos fundamentales y sus respectivas relaciones, es válido destacar que como se mencionó anteriormente la relación *tiene_tipo_desicion* subsume un conjunto de relaciones que serán explicadas con más detalles en la próxima etapa de la metodología.

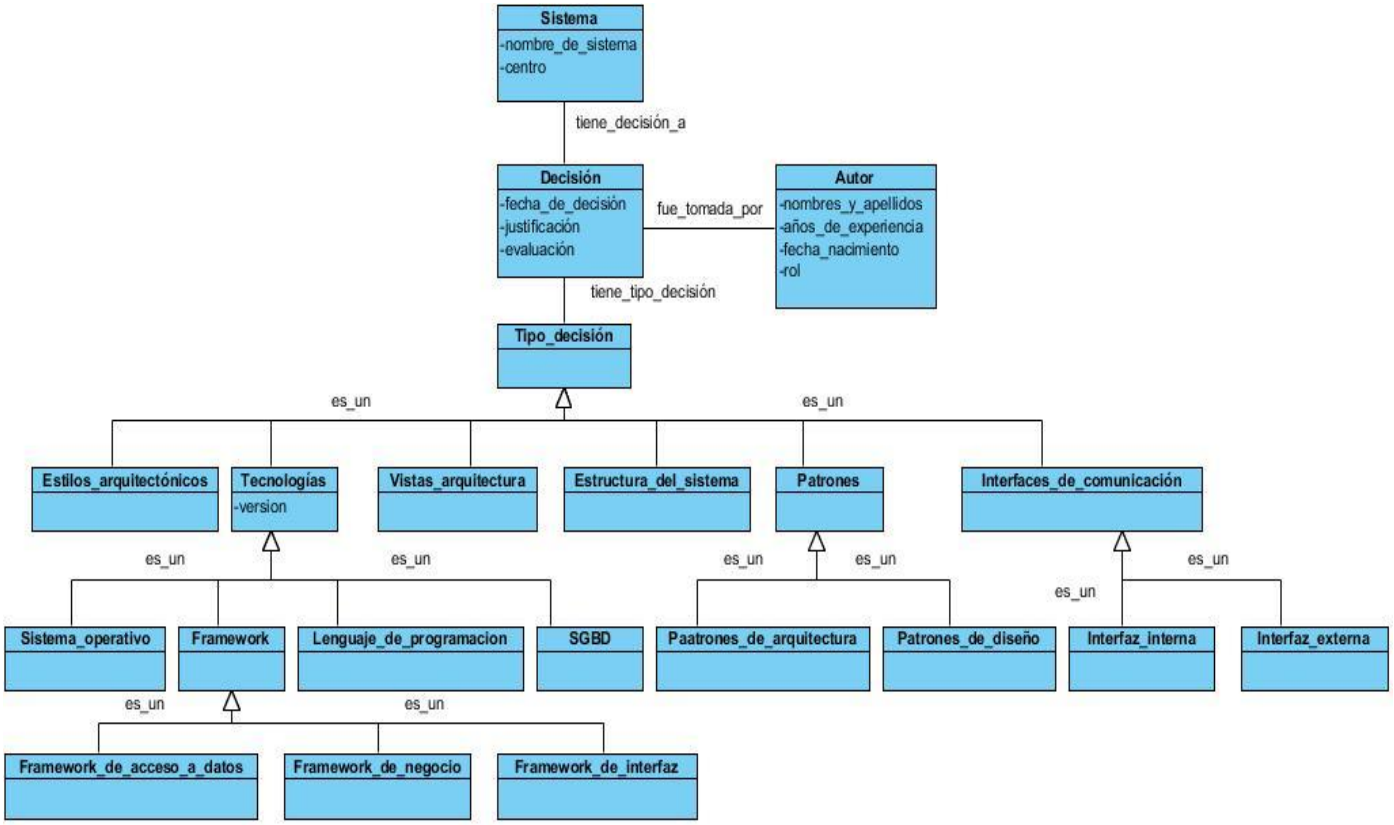



Figura 5. Modelo conceptual de la ontología

### 2.4.4 Implementación

Para la implementación de la ontología se eligió *Protégé* por su portabilidad entre diversas plataformas, su extenso uso y abundante documentación. Esta herramienta posee una interfaz gráfica que facilita el desarrollo de la ontología sin tener que preocuparse por la sintaxis del lenguaje de definición de ontologías escogido (OWL) [23, 24, 31]. A continuación se muestran cada uno de los pasos desarrollados para implementar el modelo conceptual representado en el epígrafe anterior:

- **Definición de los conceptos**

Mediante la utilización de la pestaña *Classes* del *Protégé* se representaron cada uno de los conceptos identificados en el modelo conceptual.

Para representar los conceptos resulta importante especificar el nombre de la clase, la clase padre y las clases disjuntas. En la Figura 6 se muestra la definición de las clases de la ontología desarrollada.

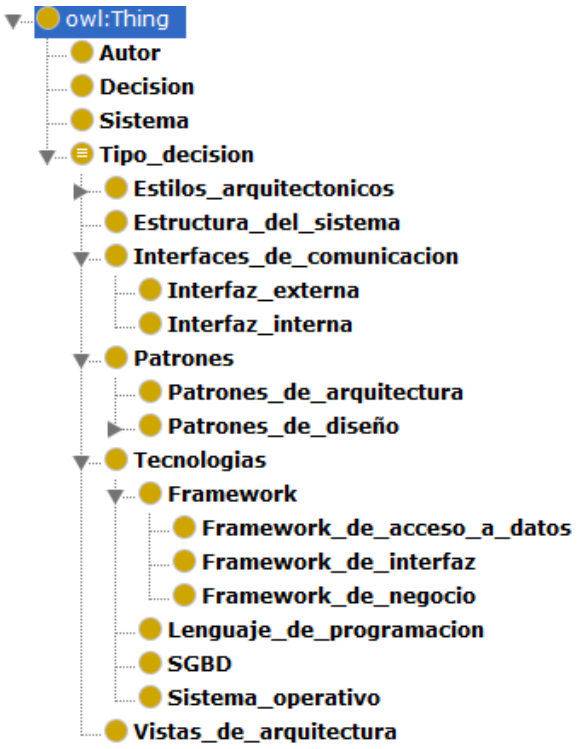



Figura 6. Clases de la ontología

Se identificaron 26 entidades como parte de la ontología desarrollada, es válido aclarar que la flexibilidad del método permite que la misma sea actualizada en caso de que aparezcan nuevas tendencias en el diseño de software.

- **Definición de las relaciones entre clases**

Luego de ser identificadas cada una de las entidades de la ontología y modeladas en la herramienta se establecieron las relaciones semánticas entre cada una de ellas mediante el uso de propiedades de objetos (*Object Properties*). Fueron representadas un total de 44 relaciones entre las clases. Es válido destacar que fueron representadas las jerarquías existentes entre cada una de estas como por ejemplo la relación entre *es_decision_de* y *es_estilo_arquitectonico_de*, siendo representada la primera como padre de la segunda. En cada caso se especifica

cuál es la relación inversa correspondiente. En la Figura 7 se representan las relaciones entre las clases de la ontología implementada.


Figura 7. Relaciones de la ontología

- **Definición de atributos**

Con el objetivo de describir cada uno de los conceptos fueron representados en la ontología los atributos presentes en el modelo conceptual. Para representar los mismos fue empleada la pestaña *Data Properties* de la herramienta *Protégé.* Es importante representar además del nombre, el dominio y el rango a la hora de agregar un nuevo atributo para garantizar la consistencia de los datos de la ontología. En la Figura 8, se muestra una representación de los atributos presentes en la ontología. Un ejemplo de ello es el atributo *nombre_y_apellidos* que tiene como dominio la clase *Autor* y como rango el tipo de dato *String*. Otra de las propiedades más importante es la *justificacion* perteneciente a la clase *Decision*, que permite reflejar el por qué la selección de una decisión determinada.

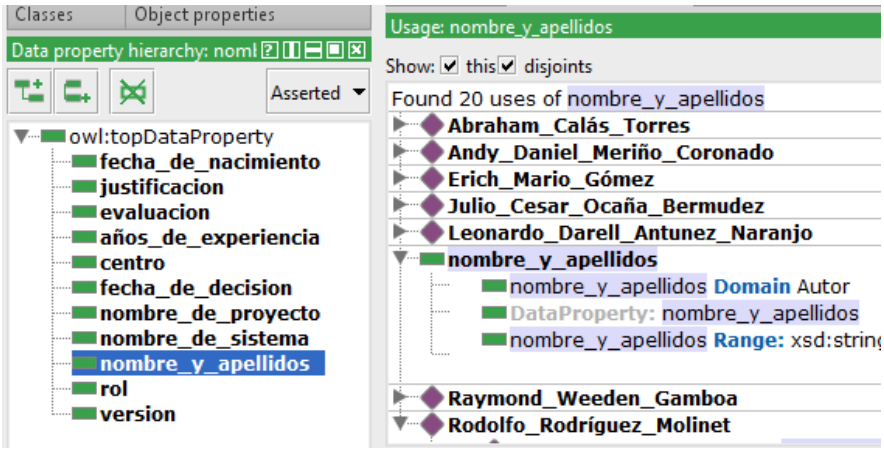



Figura 8. Atributos de la ontología

- **Definición de axiomas**

Asociado a los axiomas de la ontología se encuentran cada una de las operaciones que se realizan sobre la misma, desde crear una clase hasta representar un individuo. En la Figura 9, se representa la estadística relacionada con la ontología desarrollada, es válido destacar que la misma puede variar a medida que se utilice y actualice la ontología. En este caso se muestra por ejemplo que existen hasta el momento un total de 667 axiomas, entre ellos 25 clases, 44 relaciones, 12 propiedades y 134 individuos. El número total de axiomas se va incrementado en la medida que se utilice la ontología.

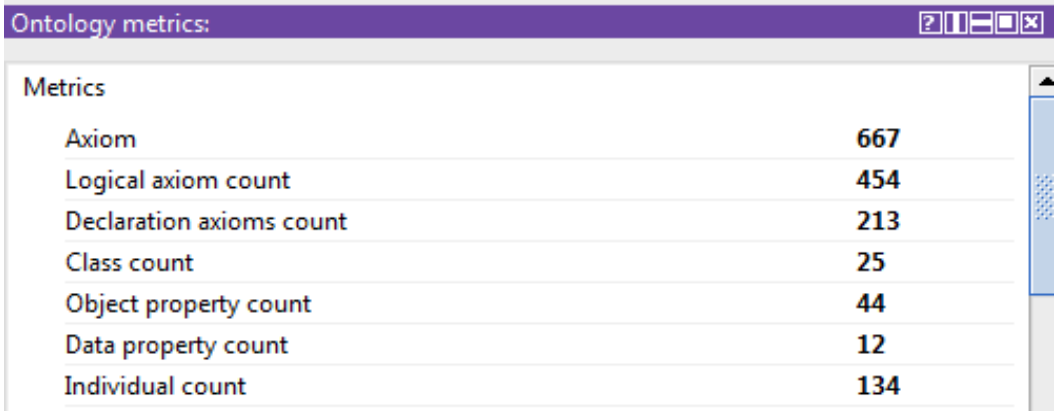



Figura 9. Estadística relacionada con los axiomas de la ontología

En la Figura 10, se muestran los axiomas correspondientes a la clase *Decision*, reflejando entre otros aspectos la definición de la clase, sus atributos y los individuos representados hasta el momento.

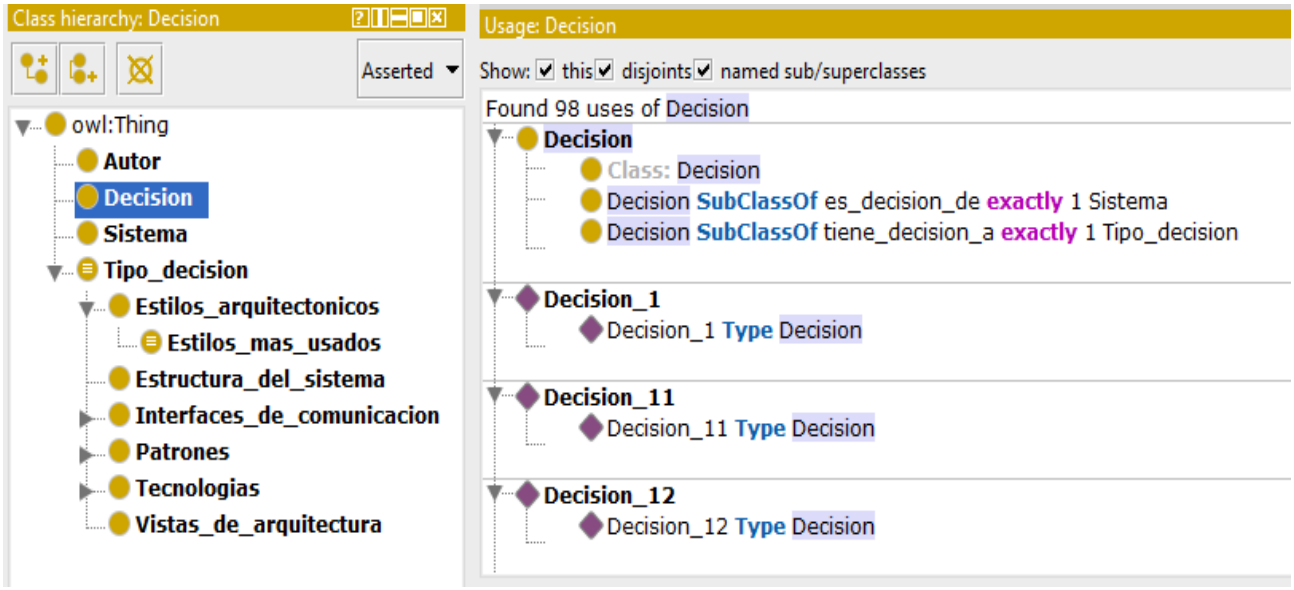



Figura 10. Axiomas de la clase *Decision*

- **Declaración de instancias**

Las instancias o individuos como también se les conoce, están asociadas a las clases definidas en la ontología. En la Figura 11 se muestra como ejemplo la creación de un individuo asociado a la clase *Autor*, al mismo se le definen las propiedades que describen a la clase como es el caso de*: nombre_y_apellidos, años_de_experiencia, fecha_de_nacimiento* y *rol*. Es importante describir correctamente cada uno de los individuos debido a que esta información puede ser útil a la hora de realizar análisis sobre el modelo ontológico. En este caso resulta interesante por ejemplo, la fecha de nacimiento de los autores pues a partir de ese dato se puede calcular la edad y conocer de esta manera, la edad promedio de las personas que se enfrentan a tomar las decisiones de diseño en la actividad productiva de la universidad. Otro dato de interés es el relacionado con la experiencia laboral de los decisores, ya que asociado a esto puede estar relacionada la calidad de las decisiones tomadas.

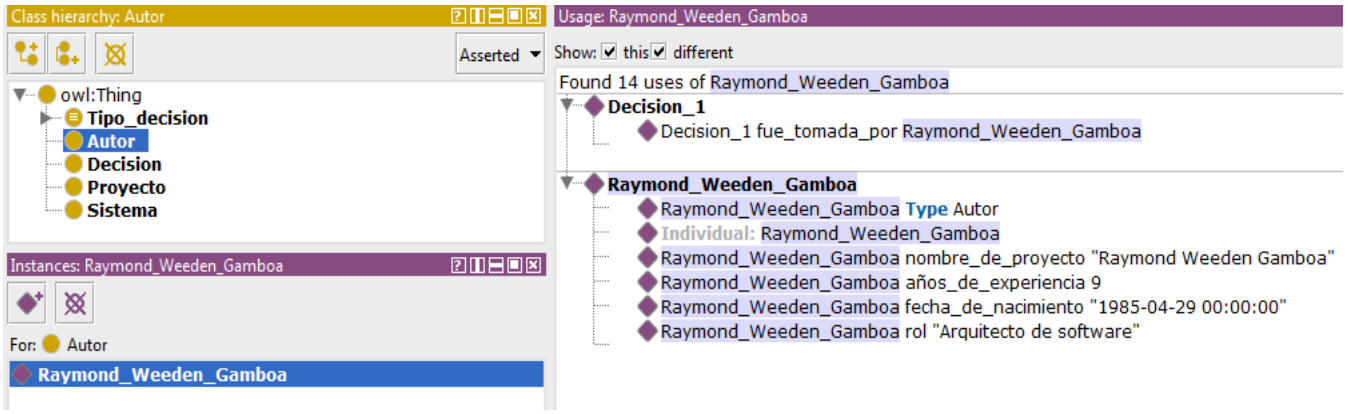



Figura 11. Creación de individuos

La actividad relacionada con la evaluación de la ontología, que constituye la quinta y última actividad propuesta por la metodología empleada, será desarrollada en el próximo capítulo como parte de la validación de la investigación desarrollada.

## 2.5. Conclusiones parciales

Luego de desarrollado el presente capítulo anterior se arribaron a las siguientes conclusiones:

- La aplicación de los métodos de revisión bibliográfica y el grupo focal contribuyeron a que se seleccionaran las decisiones de diseño fundamentales que garantizaron la implementación de la ontología.
- El método propuesto, permite representar las decisiones de diseño tomadas por los diseñadores y arquitectos de software en la actividad productiva de la UCI.
- La aplicación de la metodología seleccionada, garantizó que se implementara una ontología como herramienta para representar las decisiones de diseño y que sirva de soporte al método propuesto.

# Capítulo 3 Validación de la solución

### 3.1 Introducción

En el presente capítulo se describen los resultados de la validación de la investigación. En la Figura 12 se muestra un esquema donde se representan los métodos científicos a emplear durante la validación. Se aplicará la técnica Iadov para verificar la satisfacción de los usuarios finales, en el caso de la ontología se utilizará el método propuesto por Lopéz [53]. Se realizará además un experimento con el objetivo de verificar la disminución del tiempo como dimensión de la variable dependiente de la investigación.

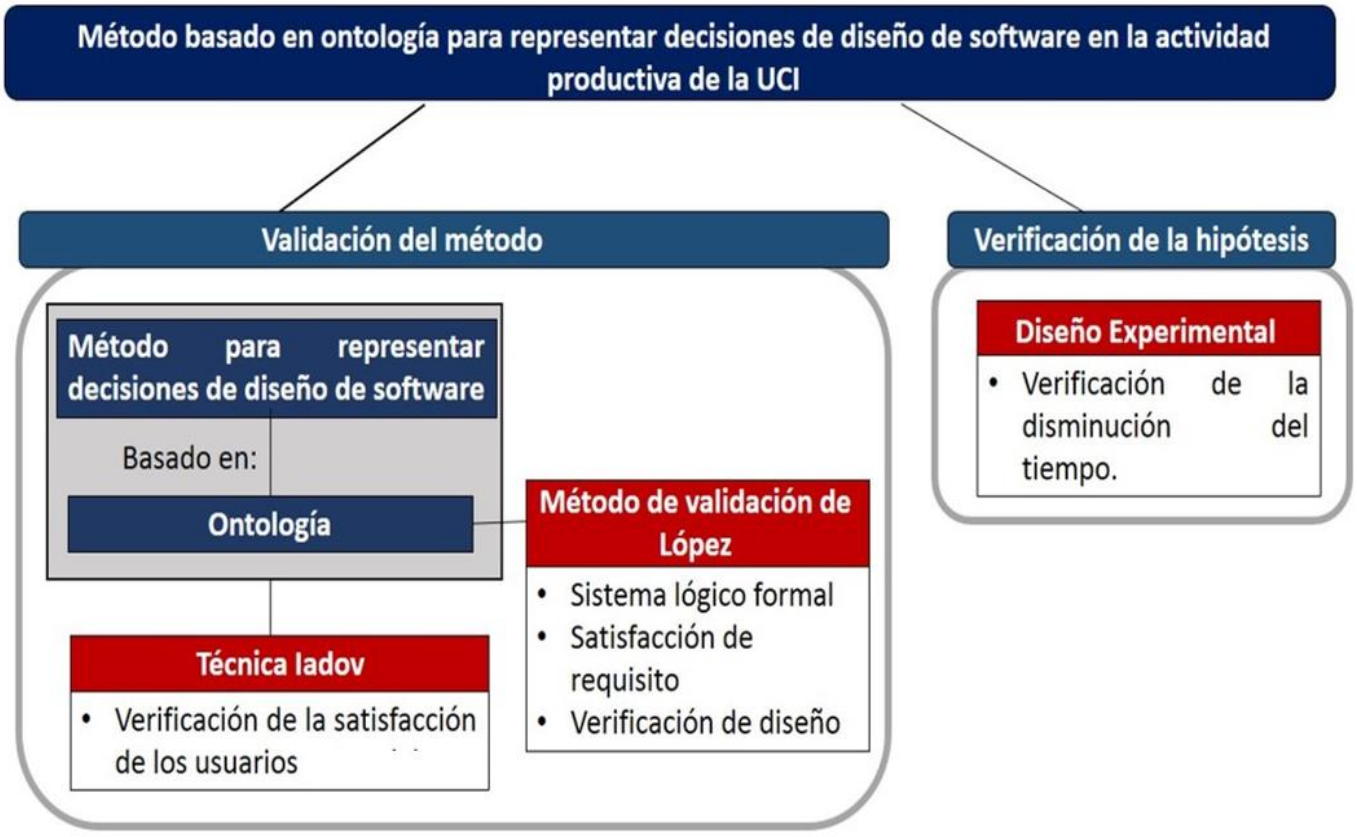



Figura 12. Esquema de validación de la investigación

### 3.2 Evaluación de la ontología

Siguiendo la secuencia de la metodología empleada para el desarrollo de la ontología, como quinta actividad se propone la evaluación de la misma, considerándose esta, como una de las actividades fundamentales durante el proceso de desarrollo del modelo ontológico.

Según Guerrero [39], el proceso de validación de las ontologías no solo se realiza luego de ser implementada la misma, sino que es un proceso transversal durante todo el desarrollo de la ontología. Para ello se realizan comprobaciones de las propiedades lógico formales por medio de razonadores que garantizan aislar los errores en contextos más reducidos. Investigaciones relacionadas con el desarrollo de las ontologías [23, 31, 39, 53] proponen tener en cuenta en la evaluación de las ontologías la comprobación de los siguientes elementos:

- Las condiciones y propiedades como sistema lógico formal.
- El diseño estructural.
- El cumplimiento de los requerimientos para los cuales fue creada.

A continuación se explica detalladamente la verificación de la ontología teniendo en cuenta los tres aspectos mencionados anteriormente.

#### 3.2.1 Verificación del sistema lógico formal

Las propiedades lógico formales de una ontología son críticas, en el sentido de que un fallo en ellas, más precisamente la ausencia de ellas, podría privarla de su carácter formal, lo que implica la ausencia de un grupo de posibilidades inferenciales, que en el caso de este trabajo son indispensables [39]:

- Chequear la consistencia de una ontología, es decir, debe ser capaz de asegurar que la ontología no contiene hechos contradictorios.
- Determinar si es posible que una clase tenga instancias.

- Clasificar la ontología a partir de las relaciones de subclase entre todos los conceptos declarados explícitamente para construir la jerarquía de clases.

- Inferir cuáles son las clases a las que directamente pertenece. Si además se utiliza la jerarquía inferida mediante la clasificación anterior, es posible obtener todas las clases a las que indirectamente pertenece una instancia dentro de la ontología.

Algunos autores coinciden en que la mejor manera de comprobar las propiedades lógicas formales de la ontología es mediante el empleo de razonadores [23, 26, 31, 39, 53]. Para comprobar estas propiedades en la presente investigación fueron utilizados como se especifica en el capítulo 1 los razonadores Pellet y HermiT.

La Figura 13 muestra el resultado que presenta el razonador al clasificar la ontología desarrollada, mostrando que no existen problemas de inconsistencia en la misma.

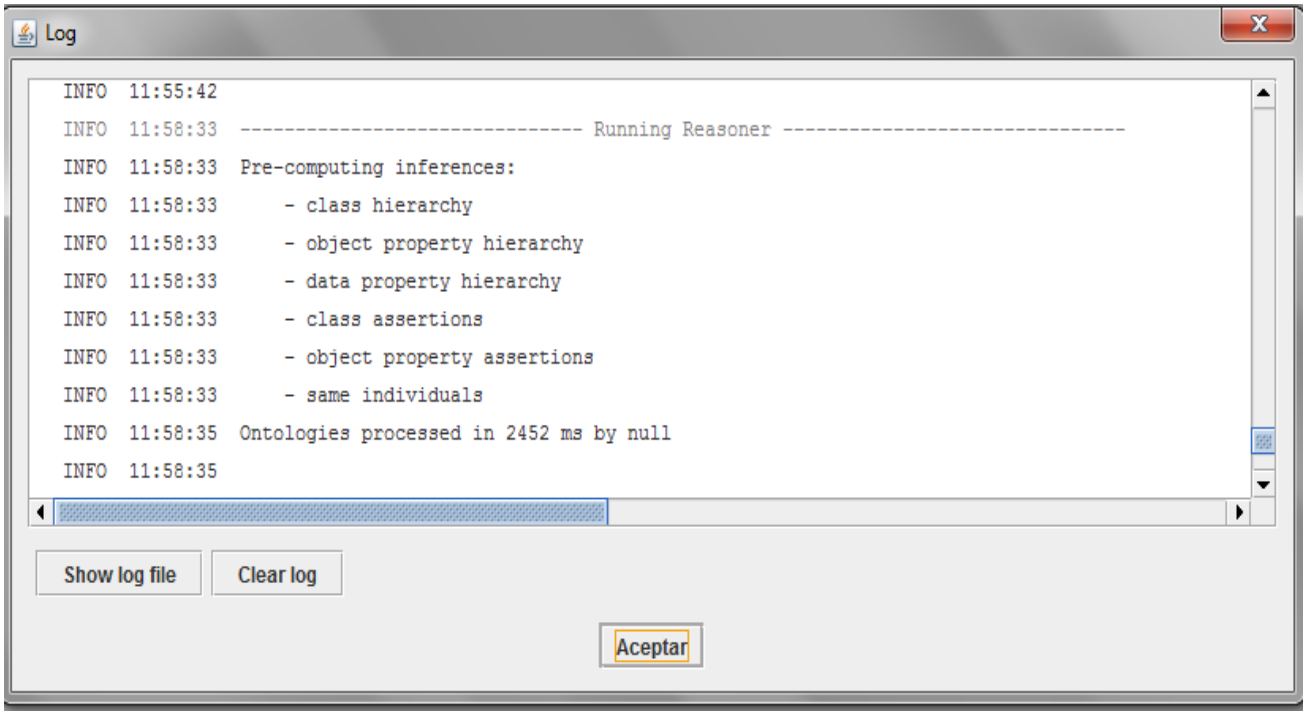



Figura 13. Resultado del razonador en la clasificación de la ontología

En la Figura 14 se muestra un ejemplo del empleo del razonador HermiT, en este caso una vez que inicia el razonador se detecta que no existen inconsistencias y se infiere conocimiento a partir de los datos que se encuentran registrados en la base de conocimiento. En este caso se muestran las decisiones de diseño relacionadas con el Sistema GINA.

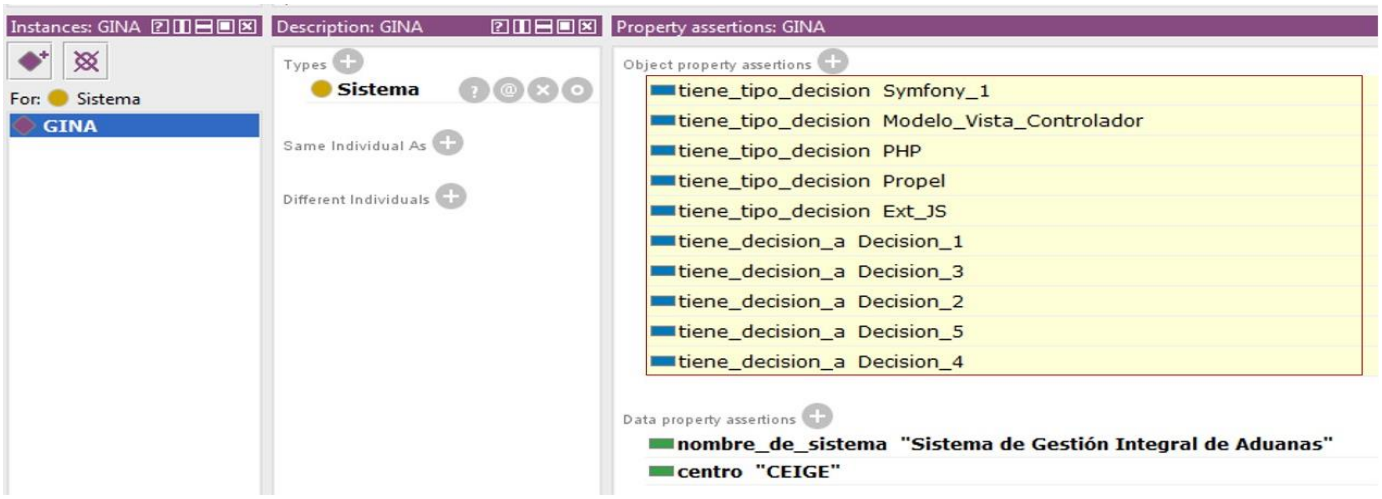



Figura 14. Empleo del razonador para el individuo GINA

En la Figura 15 se muestra un ejemplo de inconsistencia detectado a la hora de registrar la propiedad *fecha_de_decision* perteneciente a la clase *Decision*, en este caso el razonador detecta que el dato que se desea insertar no cumple con el rango definido para esta propiedad.

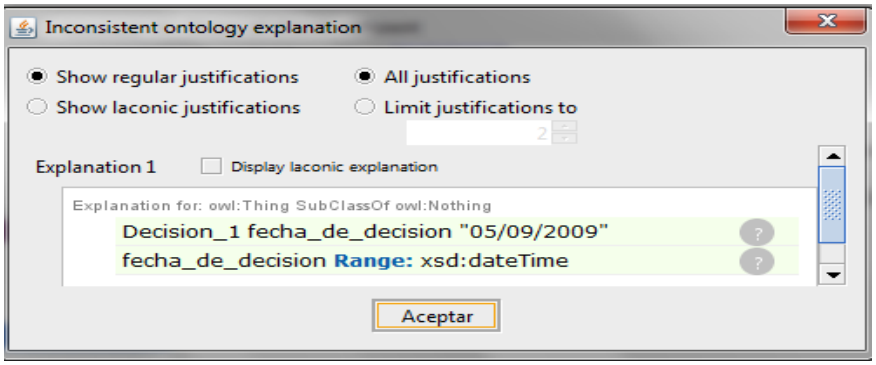



Figura 15. Ejemplo de inconsistencia detectada por el razonador

El empleo de los razonadores desde el inicio de la implementación de la ontología, garantizó además que se fueran resolviendo de manera operativa cada una de las inconsistencias que se presentaron.

### 3.2.2 Verificación del diseño

En la presente investigación se considera importante la validación desde el punto de vista del diseño de la ontología. Para ello fueron revisados los trabajos propuestos por López [23, 53] donde se propone como herramienta para validar este aspecto el uso de una lista de chequeo. La lista de chequeo empleada fue elaborada a partir de los errores más comunes detectados en el diseño de ontologías [53]. En la Tabla 4 se presenta la lista de chequeo aplicada a la ontología desarrollada para representar decisiones de diseño.

Tabla 4. Lista de chequeo de errores comunes en el diseño de ontologías

| **Tipo de error** | **Elemento a verificar** |
|---|---|
| I. | Una misma clase es definida como subclase y superclase al mismo tiempo en diferentes niveles de la taxonomía. |
| II. | Uso excesivo de la relación *es_un*. |
| III. | Existencia de más de un concepto principal. |
| IV. | Existencia de clases incompletas que provocan ambigüedad por no estar correctamente documentadas. |
| V. | Falta de conocimiento disjunto. No se declara la disyunción entre conceptos provocando una incorrecta formalización del conocimiento. |
| VI. | Falta de exhaustividad. Se declaran subclases sin tener en cuenta la división completa de los conceptos en partes. |
| VII. | Existencia de conceptos repetidos. |
| VIII. | Poca especificación o delimitación de las propiedades que provoca un pobre razonamiento. |
| IX. | No se corresponden los elementos del dominio con los conceptos declarados o no se corresponde el conocimiento del dominio con los conceptos, relaciones y axiomas declarados. |

| | |
|---|---|
| X. | Existencia de redundancia entre las extensiones disjuntas de un concepto. |
| XI. | No se tiene en cuenta la traducción de los conceptos de la taxonomía a otros idiomas. |
| XII. | Falta de estandarización. Los nombres de los términos no siguen un estándar. |

La lista de chequeo fue aplicada cuando se contaba con el diseño de la ontología, se detectaron errores de tipo II, III, VIII y XII. Dentro de los problemas detectados se encuentra el uso excesivo de la relación *es_un*, estas fueron disminuidas, aunque la existencia de varias relaciones de este tipo en la ontología se encuentra aparejada a la jerarquía definida entre las clases. Por otra parte, la ontología no contaba con un solo concepto principal por lo que luego de realizar un análisis de cada uno de los conceptos se decidió que el concepto principal fuera *Sistema* y asociados a él el resto de los conceptos. Existían además pocas propiedades asociadas a los conceptos, para darle solución a este error se analizó el modelo conceptual diseñado y se agregaron atributos al mismo que posteriormente se convirtieron en propiedades de las clases o conceptos. A pesar de contar desde el inicio con un estándar para nombrar los conceptos, relaciones y propiedades, fue detectado el nombre de un concepto y una relación que no estaban descrito de manera estándar. Es importante destacar que el error de tipo XI se consideró que no aplica para la presente investigación, pues la misma será implantada en un entorno donde se domina el idioma español, aunque se tendrá en cuenta como elemento a mejorar con el objetivo de extender la solución a otros entornos de trabajo.

### 3.2.3 Verificación de requisitos

Con el objetivo de comprobar el cumplimiento de los requisitos definidos al iniciar el desarrollo de la ontología, se elaboraron y aplicaron tres casos de prueba (ver Anexo 3). En la elaboración de los casos de prueba se tuvo en cuenta que se diseñaran los mismos de manera que cubrieran todas las preguntas contenidas en el DERO (ver Anexo 2). Los casos de prueba diseñados cuentan con la siguiente estructura: pregunta de competencia que aborda, escenario de prueba, resultado esperado y resultado obtenido. A continuación se muestra el caso de prueba 1:

**Caso de prueba 1**

**Pregunta de competencia:** ¿cuáles son las decisiones de diseño tomadas para un sistema determinado?

**Escenario:** en la herramienta Protégé se crearon las instancias de la ontología como se muestra en la Tabla 5.

Tabla 5. Instancias para caso de prueba 1

| **Clases** | **Instancias** | **Propiedades** | **Valor de las Propiedades** |
|---|---|---|---|
| Sistema | GINA | nombre_de_sistema | Sistema de Gestión Integral de Aduanas |
| | | centro | CEIGE |
| Autor | Raymond_Weeden_Gamboa | nombre_y_apellidos | Raymond Weeden Gamboa |
| | | años_de_experiencia | 9 |
| | | fecha_de nacimiento | 29/04/1985 |
| | | rol | Arquitecto |
| Decision | Decision_1 | fecha_de_decision | 05/09/2009 |
| | Decision_2 | fecha_de_decision | 05/09/2009 |
| | Decision_3 | fecha_de_decision | 05/09/2009 |
| | Decision_4 | fecha_de_decision | 05/09/2009 |
| | Decision_5 | fecha_de_decision | 05/09/2009 |
| | Decision_6 | fecha_de_decision | 05/09/2009 |
| Framework_de_acceso_a_datos | Propel | | |
| Framework_de_interfaz | Ext_JS | | |
| Framework_de_negocio | Symfony_1 | | |
| Patrones_de_arquitectura | MVC | | |
| Lenguaje_de_programacion | PHP | | |
| SGBD | Oracle | | |

**Resultado esperado:** al aplicar un razonador el sistema debe de mostrar las seis decisiones asociadas al Sistema GINA.

**Resultado obtenido:** satisfactorio. La Figura 16 muestra el resultado obtenido por el razonador en el recuadro rojo.

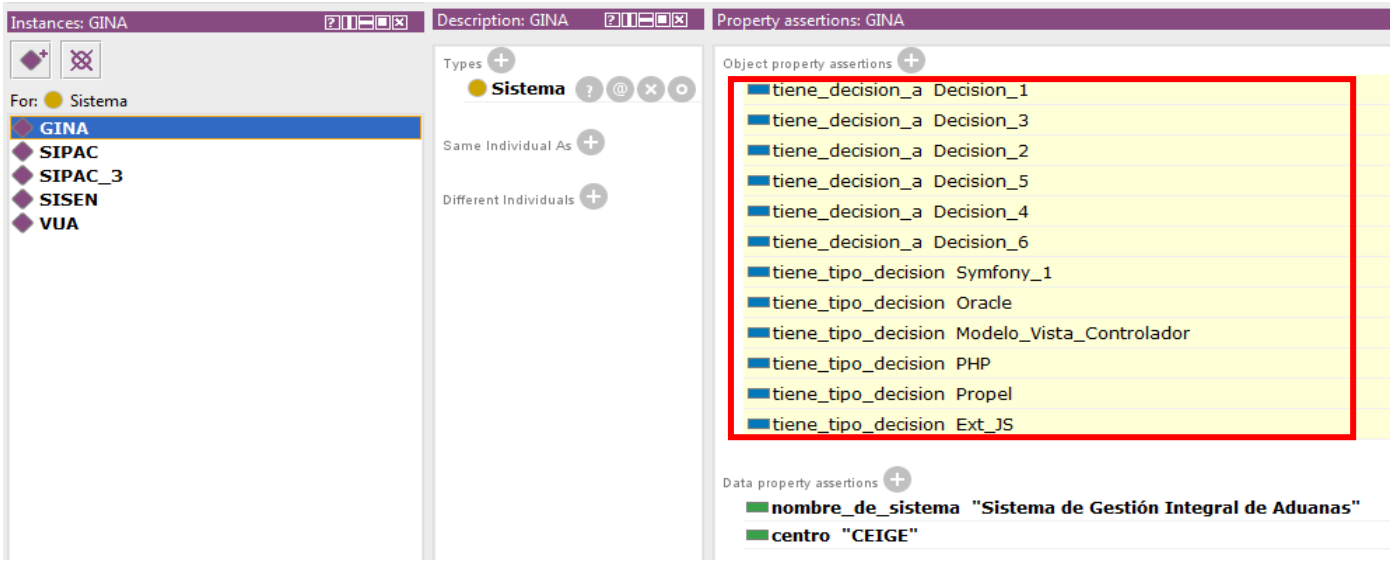



Figura 16. Resultado del caso de prueba 1

Existen casos de prueba que responden a más de una pregunta, porque la respuesta a esta se encuentra relacionada con el mismo concepto y los razonadores infieren varios resultados en una misma vista. Un ejemplo de esto es el caso de prueba 3 (ver Anexo 3) que responde a tres preguntas de las reflejadas en los requisitos. Luego de ser aplicados los casos de prueba, se pudo comprobar que todos arrojaron resultados satisfactorios, lo que demuestra que la ontología satisface los requisitos para los cuales fue creada.

### 3.3 Diseño experimental para verificar la hipótesis de la investigación

Para validar el cumplimiento de la hipótesis de la investigación se realizó un diseño experimental. La hipótesis planteada al iniciar la investigación es: “Si se desarrolla un método basado en ontología para representar las decisiones de diseño de software en la actividad productiva de la Universidad de las Ciencias Informáticas, se agilizará el análisis realizado sobre las mismas”.

Para la realización del diseño experimental se utilizó como referencia el libro “Metodología de la Investigación” de Roberto Hernández Sampieri en su segunda edición [54]. Según el autor citado anteriormente, los diseños experimentales se dividen en tres clases: pre experimentos, experimentos verdaderos y cuasi experimentos.

El tipo de diseño experimental que se realizó fue un cuasi experimento con pre y post prueba sobre un único grupo. El cuasi experimento se desarrolló en un ambiente natural. Se tomó como población inicial para desarrollar el experimento, los proyectos de desarrollo del Centro de Informatización de Entidades (CEIGE) desde el año 2014 hasta la fecha. El CEIGE es uno de los centros con mayor cantidad de proyectos desarrollados en el período con un total de 14 proyectos de este tipo. De los 14 proyectos identificados, se consideraron para la muestra solo 7 por la calidad de la documentación, encontrándose descrita correctamente las decisiones de diseño en los documentos relacionados con esta etapa. El tamaño de la muestra se consideró representativo de la población, siendo mayor al 10 % considerado el límite mínimo de confiabilidad según Rolando Alfredo [55].

El diseño del cuasi experimento quedó formalizado de la siguiente manera:

$$G\ O_1\ X\ O_2$$

Donde:

- G: es el grupo sobre el que se realizaron las observaciones. En el caso particular de este estudio G estuvo compuesto por los 7 proyectos de la muestra.
- $O_1$: se corresponde con la primera observación realizada sobre el grupo objeto de estudio. Se calculó el tiempo total aproximado empleado para la consulta de información, relacionada a las decisiones de diseño de los 7 proyectos sin la herramienta desarrollada.
- X: se corresponde con la aplicación sobre el grupo objeto de estudio de la variable independiente, en este caso la solución propuesta.
- $O_2$: se corresponde con la segunda observación sobre el grupo objeto de estudio. Se calculó el tiempo aproximado empleado para la consulta de información, relacionada a las decisiones de diseño de los siete proyectos con la herramienta desarrollada.

Para comprender los resultados del diseño experimental sobre la dimensión de la variable tiempo, se estableció como escala valorativa considerarlo bueno si se logra una disminución del tiempo por encima del 40 % con la

utilización del sistema. Para comprobar la disminución del tiempo fue utilizada una pregunta de las representadas en el DERO (ver Anexo 2) que plantea lo siguiente: ¿Cuáles son las decisiones de diseño tomadas para un sistema determinado? En el cuasi experimento participaron 9 especialistas del centro CEIGE y un representante de la dirección, en el caso de los especialistas todos ocupan un rol determinado en alguno de los proyectos del centro. En la primera observación ($O_1$), se midió el tiempo que demoró cada participante en identificar las decisiones de diseño de los proyectos que forman parte de la muestra, en este caso se utilizó la documentación de los expedientes de proyectos. Como resultado de $O_1$ se constató que los especialistas necesitaron como promedio 3 688 segundos para identificar las decisiones de diseño de los proyectos en la documentación, ver Tabla 6.

Tabla 6. Medición del tiempo en segundos sin el sistema

| No. de participante | BK-Import | VUA | BOSON | GINA | Orbita | SIPAC | SISEN | Total |
|---|---|---|---|---|---|---|---|---|
| 1 | 558 | 522 | 480 | 546 | 492 | 426 | 498 | 3522 |
| 2 | 612 | 570 | 492 | 480 | 462 | 450 | 492 | 3558 |
| 3 | 660 | 576 | 510 | 492 | 468 | 444 | 522 | 3672 |
| 4 | 582 | 546 | 522 | 462 | 426 | 390 | 450 | 3378 |
| 5 | 780 | 714 | 636 | 612 | 570 | 510 | 558 | 4380 |
| 6 | 504 | 480 | 456 | 474 | 396 | 390 | 444 | 3144 |
| 7 | 816 | 774 | 738 | 768 | 660 | 510 | 570 | 4836 |
| 8 | 546 | 534 | 492 | 480 | 426 | 414 | 480 | 3372 |
| 9 | 660 | 546 | 492 | 480 | 462 | 492 | 558 | 3690 |
| 10 | 570 | 498 | 450 | 492 | 426 | 414 | 480 | 3330 |
| | | | | | | | **Promedio:** | **3688,2** |
| | | | | | | | **Mediana:** | **3540** |

En la segunda observación ($O_2$), se midió el tiempo que demoraron los participantes para identificar las decisiones de diseño previamente registradas en la ontología. Se utilizó una computadora con las mismas prestaciones en todos los casos para garantizar se encontraran en igualdad de condiciones y que el sistema funcionara de manera similar. Antes de iniciar se le explicó cómo trabajar con la ontología y la estructura de la misma. Como resultado de

$O_2$ se constató que se necesita como promedio 370 segundos para consultar las decisiones de diseño de los siete proyectos que forman parte de la muestra (ver Tabla 7).

Tabla 7. Medición del tiempo en segundos con el sistema

| No. de participante | BK-Import | VUA | BOSON | GINA | Orbita | SIPAC | SISEN | Total |
|---|---|---|---|---|---|---|---|---|
| 1 | 45 | 59 | 51 | 62 | 47 | 43 | 60 | 367 |
| 2 | 48 | 50 | 49 | 69 | 49 | 42 | 64 | 371 |
| 3 | 47 | 52 | 56 | 65 | 52 | 46 | 59 | 377 |
| 4 | 45 | 51 | 48 | 56 | 44 | 40 | 55 | 339 |
| 5 | 50 | 57 | 53 | 59 | 58 | 44 | 60 | 381 |
| 6 | 43 | 45 | 47 | 51 | 55 | 40 | 57 | 338 |
| 7 | 57 | 65 | 55 | 73 | 63 | 49 | 71 | 433 |
| 8 | 46 | 55 | 44 | 55 | 50 | 45 | 56 | 351 |
| 9 | 52 | 54 | 48 | 57 | 53 | 49 | 68 | 381 |
| 10 | 55 | 57 | 46 | 56 | 41 | 46 | 61 | 362 |
| | | | | | | | **Promedio:** | **370** |
| | | | | | | | **Mediana:** | **369** |

Las observaciones realizadas permiten constatar que el sistema contribuye a disminuir en un 90% el tiempo empleado para consultar las decisiones de diseño. A partir de la escala valorativa establecida para la variable tiempo se considera bueno el resultado y por tanto el cuasi experimento aportó evidencias cuantitativas a favor de la hipótesis de la investigación.

### 3.4 Aplicación de la técnica de Iadov

Una vez concluida la propuesta de solución, se considera interesante conocer el grado de satisfacción y la opinión de los usuarios finales de la misma. Esta información es utilizada para identificar las fortalezas y las debilidades de la propuesta, con el objetivo de garantizar la mejora continua de la misma. La técnica de Iadov es un instrumento que ayuda a conocer este grado de satisfacción y ha sido aplicada en varias investigaciones realizadas en los últimos tiempos para conocer este indicador [23, 36, 56-58]. Para la aplicación de esta técnica se diseñó un cuestionario (ver

Anexo 4) compuesto por tres preguntas cerradas y tres abiertas. La relación entre las preguntas cerradas se establece en el cuadro lógico de Iadov. En la Tabla 8 se muestra el cuadro lógico modificado para la presente investigación.

Tabla 8. Cuadro lógico de Iadov modificado por el autor

| Luego de conocer el método desarrollado para la descripción de decisiones de diseño basado en ontología, refleje en qué medida le gusta la solución desarrollada marcando con una X donde considere. | ¿Considera usted que la descripción de decisiones de diseño sin herramienta de apoyo es eficiente? | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | No | | | No sé | | | Sí | | |
| | ¿Utilizaría el método que se presenta en esta investigación como una herramienta para describir las decisiones de diseño? | | | | | | | | |
| | Sí | No sé | No | Sí | No sé | No | Sí | No sé | No |
| Me gusta mucho | 1 | 2 | 6 | 2 | 2 | 6 | 6 | 6 | 6 |
| No me gusta tanto | 2 | 2 | 3 | 2 | 3 | 3 | 6 | 3 | 6 |
| Me da lo mismo | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Me disgusta más de lo que me gusta | 6 | 3 | 6 | 3 | 4 | 4 | 3 | 3 | 4 |
| No me gusta nada | 6 | 6 | 6 | 6 | 4 | 4 | 6 | 6 | 5 |
| No sé qué decir | 2 | 3 | 6 | 3 | 3 | 3 | 6 | 6 | 4 |

Los datos reflejados en el cuadro lógico forman parte del resultado de la aplicación del cuestionario diseñado a 15 profesionales con experiencia en el desarrollo de software en la UCI. Los participantes pertenecen a Centros de Desarrollo (9), Dirección de Calidad de Software (1), Dirección General de Producción (2) e investigadores en la temática (3), en todos los casos pertenecientes a la UCI y vinculados a la actividad productiva de la universidad. Cuentan con un promedio de 8 años de experiencia en el desarrollo de software y han ocupado los roles de arquitectos de datos y de sistemas, diseñadores, desarrolladores, analistas, líderes de proyecto y probadores, en todos los casos cuentan con experiencia en más de un rol. En el caso de los investigadores, se encuentran realizando investigaciones relacionadas con la temática y cuentan con resultados introducidos en la actividad productiva.

El número resultante de la interrelación de las tres preguntas indica la posición en la escala de satisfacción siguiente:

- Clara satisfacción (A)
- Más satisfecho que insatisfecho (B)
- No definida (C)
- Más insatisfecho que satisfecho (D)
- Clara insatisfacción (E)
- Contradictoria (C)

Tabla 9. Escala de satisfacción

| **Total de usuarios de la muestra (N)** | **15** | **Escala** |
|---|---|---|
| Clara satisfacción | 12 | A |
| Más satisfecho que insatisfecho | 2 | B |
| No definida | 0 | C |
| Más insatisfecho que satisfecho | 0 | D |
| Clara insatisfacción | 0 | E |
| Contradictoria | 1 | C |

Teniendo como punto de partida la cantidad de respuestas por cada una de las categorías presentadas anteriormente, se procede a calcular el índice de satisfacción grupal (ISG) utilizando la fórmula siguiente:

$$\frac{A(+1) + B(+0{,}5) + C(0) + D(-0{,}5) + E(-1)}{N}$$

Donde **N** es el número de la cantidad total de respuestas, en este caso **N** = 15

El valor del ISG permite identificar las siguientes categorías grupales:

- Insatisfacción: desde (-1) hasta (-0,5)
- Contradictorio: desde (-0,49) hasta (+0,49)
- Satisfacción: desde (+0,5) hasta (+1)

El gráfico de la Figura 17 muestra los resultados de la aplicación de la técnica de Iadov en la presente investigación. El valor obtenido del ISG fue de 0,87 lo que indica satisfacción de los usuarios con el sistema desarrollado.

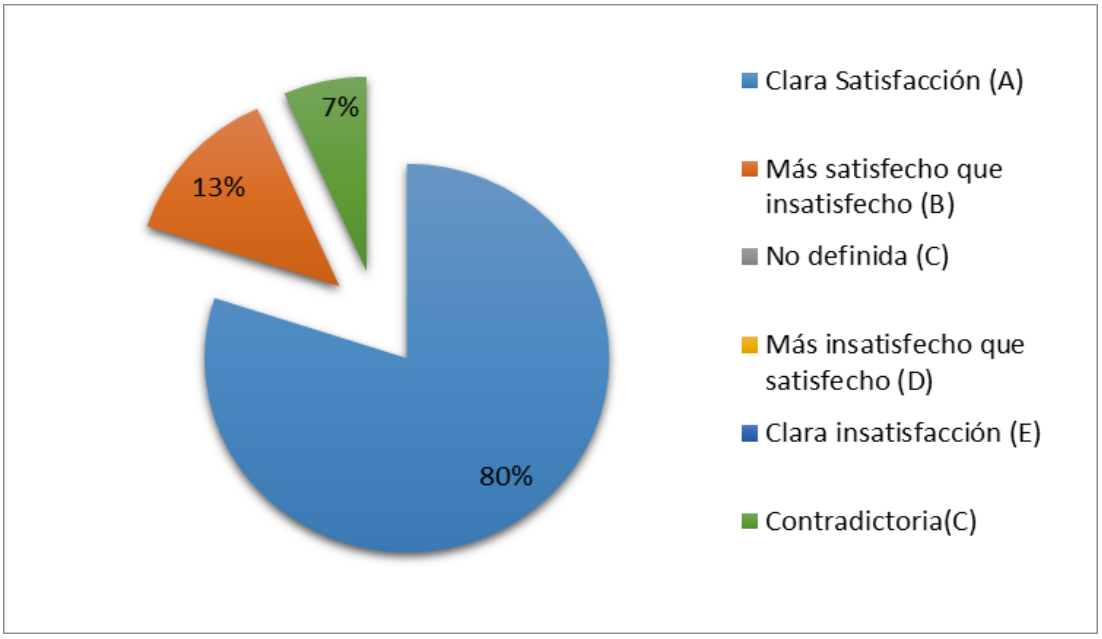



Figura 17. Niveles de satisfacción de usuarios finales

Como parte de la aplicación de la técnica de Iadov, fueron incluidas 3 preguntas abiertas. Estas ayudan a identificar algunas causas que condicionan el nivel de satisfacción de los usuarios. La respuestas obtenidas a las preguntas 5 y 6 (ver anexo 4) representan recomendaciones para mejorar la propuesta presentada:

- Desarrollar una aplicación web que facilite el empleo de la ontología.
- Extender la propuesta a todas las decisiones que se toman en el desarrollo de software.

En el caso de la pregunta 4 (ver anexo 4), fueron resaltados los elementos positivos de la propuesta:

- Agilidad a la hora de consultar información relacionada con las decisiones de diseño en la universidad.
- Contribuye al apoyo en la toma de decisiones para representar decisiones de diseño.
- Facilita la socialización del conocimiento.

### 3.5 Conclusiones parciales

Luego del desarrollo del capítulo se arribaron a las siguientes conclusiones:

- El método propuesto para validar las ontologías, permitió verificar las condiciones favorables de su diseño estructural y de sus condiciones y propiedades como sistema formal, así como el cumplimiento de los requisitos para los cuales fueron creadas.
- El cuasi experimento realizado, permitió verificar la disminución del tiempo en el análisis realizado sobre las decisiones de diseño por lo que aportó evidencias cuantitativas a favor de la hipótesis de la investigación.
- La aplicación de la técnica de Iadov, permitió identificar oportunidades de mejoras de la presente investigación y constatar la satisfacción de los usuarios finales.

## Conclusiones generales

Con el desarrollo del presente trabajo se arribaron a las siguientes conclusiones:

- El desarrollo del marco teórico de la investigación, permitió identificar soluciones relacionadas con la descripción de decisiones de diseño de software, aunque estas cuentan con limitaciones para ser aplicadas a la actividad productiva de la UCI al estar diseñadas para un negocio específico. Estas soluciones estaban basadas en ontologías lo que afirma a esta técnica de Inteligencia Artificial como factible para representar este tipo de conocimiento.

- La ontología desarrollada, permite describir y analizar las decisiones de diseño de software en la actividad productiva de la UCI.

- El método desarrollado está constituido por seis actividades que permiten representar las decisiones de diseño tomadas por los diseñadores y arquitectos de software en la actividad productiva de la UCI.

- El conjunto de métodos científicos utilizados para la validación de la propuesta (método de López, cuasi experimento y técnica de Iadov) permitió comprobar que:

    - El empleo de los razonadores, la lista de chequeo para el diseño y los casos de prueba, garantizaron verificar las propiedades lógicas formales, la calidad del diseño y el cumplimiento de los requisitos de la ontología respectivamente.

    - El cuasi experimento realizado aportó evidencias cuantitativas a favor de la hipótesis de la investigación, demostrando una disminución del tiempo de los análisis realizados sobre las decisiones de diseño con el empleo de la propuesta.

    - La aplicación de la técnica de Iadov permitió constatar la satisfacción de los usuarios finales.

**Recomendaciones**

1. Extender el método desarrollado a otras etapas del desarrollo de software de manera que cubra todo el proceso.

2. Desarrollar una aplicación web que implemente el método desarrollado para facilitar la interacción de los usuarios con el mismo.

3. Incorporar elementos que permitan evaluar las decisiones de diseño tomadas.

## Bibliografía


1. SOMERVILLE, I., *Ingeniería de Software*, ed. P. EDUCACIÓN. 2011, México.

2. STANDISH-GROUP. *Chaos Manifesto 2015* 2015 [cited 2018 29 de marzo de 2018]; Available from: https://www.infoq.com/articles/standish-chaos-2015.

3. TIBERMACINE, C., et al. , *Software architecture constraint reuse-by-composition.* Future Generation Computer Systems, 2016. **61**: p. 37-53.

4. NOY, N.F., et al. , *Ontology development 101: A guide to creating your first ontology.* . 2001.

5. CHAUHAN, M.A.B., Muhammad Ali; SHENG, Quan Z., *A Reference Architecture for provisioning of Tools as a Service: Meta-model, Ontologies and Design Elements.* Future Generation Computer Systems, 2017. **69**: p. 41-65.

6. GALSTER, M.M., Mehdi; MEDVIDOVIC, Nenad., *Bringing Architecture Thinking into Developers' Daily Activities.* ACM SIGSOFT Software Engineering Notes, 2017. **41**(6): p. 24-26.

7. GIRALDO, G., GLORIA L.; ACEVEDO, O, JUAN F.; MORENO, N, DAVID A., , *Una ontología para la representación de conceptos de diseño de software.* Avances en Sistemas e Informáticas, 2011. **8**(3): p. 103-110.

8. VAN VLIET, H.T., Antony., *Decision making in software architecture.* Journal of Systems and Software, 2017. **117**: p. 638-644.

9. LEINONEN, T.D., Eva., *Pensamiento de diseño y aprendizaje colaborativo.* . Comunicar, 2014. **21**(42): p. 107.

10. MING, Z.W., GUOXIN.; YAN, YAN.; PANCHAL, J.; , *Ontology-Based Representation of Design Decision Hierarchies.* Journal of Computing and Information Science in Engineering, 2018. **18**.

11. GUARINO, N., *Formal Ontology in information systems*, in *Proceeding of FOIS 98*. 1998, IOS Press: Italy. p. 3-15.

12. UCI. *Sitio Oficial de la Universidad de las Ciencias Informáticas*. 2018 [cited 2018 9 de Septiembre]; Available from: http://www.uci.cu/.

13. CMMI. *CMMI Institute*. 2015 [cited 2018 9 de Septiembre]; Available from: https://sas.cmmiinstitute.com/pars/pars_detail.aspx?a=25323.

14. PRESSMAN, R.S., *Ingeniería del software. Un enfoque práctico*. 2010, México: McGraw-Hill. 777.

15. SILVA, D.M., Bárbara. , *Construyendo aplicaciones web con una metodología de diseño orientada a objetos.* Revista Colombiana de Computación–RCC, 2001. **2**(12).

16. LÓPEZ, C.C., VÍCTOR.; ASTUDILLO, HERNÁN.; CYSNEROS, LUIZ M.;, *Bridging the gap between software architecture rationale formalisms and actual architecture documents: An ontology-driven approach.* Science of Computer Programming, 2010.

17. WELTY, C.G., Nicola. , *Supporting ontological analysis of taxonomic relationships.* Data & Knowledge Engineering, 2001. **39**(1): p. 51-74.

18. STUDER, R.B., V. R., *Knowledge engineering: Principles and methods.* . Science Direct. , 1998. **25**(1-2): p. 161-197.

19. TELLO, A.L., *Ontologías en la Web Semántica.* Jornadas de Ingeniería Web., 2001. **1**.

20. LAMARCA, M.J., *Hipertexto, el nuevo concepto de documento en la cultura de la imagen.* 2013, Universidad Complutense de Madrid: España.

21. KOTSIANTIS, S.B.K., D.; KARIOTI, V.; TAMPAKAS, V., *An ontology-based portal for credit risk analysis.*, in *Computer Science and Information Technology. ICCSIT 2009*. 2009, 2nd IEEE International Conference. p. 165-169.

22. LUNA, J.A.G.B., M. L. , *Metodologías y métodos para la construcción de ontologías.* UTP Universidad Técnica de Pereira., 2012. **20**(50).

23. LÓPEZ, R., Y., *Método para la integración de ontologías en sistemas relacionales para la evaluación de créditos*. Maestría en Informática Aplicada. Tutor N. SILEGA MARTÍNEZ, in *Centro de Informatización de Entidades*. 2017, Universidad de las Ciencias Informáticas: La Habana, Cuba.

24. ALVARADO, R.D. *Metodología para el desarrollo de ontologías*. 2010; Available from: http://es.slideshare.net/Iceman1976/metodologia-para-ontologias.

25. CORCHO, O.F.-L., M; GÓMEZ-PÉREZ, A. , *Methodologies, tools and languages for building ontologies. Where is their meeting point?* Data & Knowledge engineering, 2003. **46**(1): p. 41-64.

26. GARCÍA, M.N., *Modelado y análisis de sistemas CSCW siguiendo un enfoque de ingeniería dirigida por ontologías*. Tesis de Doctorado, Tutor L. GARRIDO BULLEJOS AND M. V. HURTADO TORRES, in *Departamento de lenguajes y sistemas informáticos*. 2009, Universidad de Granada: Granada, España.

27. SCHREIBER, G., WIELINGA,B., AKKERMANS, H., VAN DE VELDE, W., ANJEWIERDEN, A., *CML: The CommonKADS Conceptual Modelling Language.*, in *8th European Knowledge Acquisition Workshop*, Springer-Verlag, Editor. 1994. p. 1-25.

28. GENESERETH, M. *Knowledge Interchanged Format (KIF)*. 1998.

29. GRUBER, T.R., *A translation approach to portable ontology specifications.* Knowledge Acquisition 5, 1993: p. 199-220.

30. DELGADO, Y.H.P., R. , *La web semántica: una breve revisión. .* Revista Cubana de Ciencias Informáticas. , 2013. **7**(1): p. 76-85.

31. SILEGA, N., et al. , *Framework basado en ontología para la descripción y validación de procesos de negocio.* Ingeniería Industrial, 2017. **38**(3).

32. RAMOS, E.N., H., *ONTOLOGIAS: componentes, metodologías, lenguajes, herramientas y aplicaciones.* Lecturas en Ciencias de la Computación, 2007.

33. BARRERA, M.N., H; RAMOS, E., *INGENIERÍA ONTOLÓGICA.* Lecturas en Ciencias de la Computación, 2012.

34. SWOOP. *mindswap.org*. 2015 [cited 2018 14 de Septiembre]; Available from: http://www.mindswap.org/2004/SWOOP/.

35. HUIQUN, Z.S., ZHANG; JUMBAO, ZHAO. , *Research of Using Protégé to Build Ontology.*, in *IEEE/ACIS 11th International Conference*, C.a.I.S. (ICIS), Editor. 2012. p. 697-700.

36. SILEGA, M., N. , *Método para la transformación automatizada de modelos de procesos de negocio a modelos de componentes para sistemas de gestión empresarial*, Doctorado en Ciencias Técnicas, Tutor J. P. FEBLES RODRÍGUEZ AND M. NOGUERA GARCÍA, in *Centro de Informatización de Entidades*. 2014, Universidad de las Ciencias Informáticas: La Habana, Cuba.

37. GRIVOKOSTOPOULOU, F.P., I.; HATZILYGEROUDIS, I. , *Using Semantic Web Technologies in a Web Based System for Personalized Learning AI Course.*, in *IEEE Sixth International Conference*, T.f.E. (T4E), Editor. 2014. p. 257-260.

38. HORRIDGE, M.B., S.; PARSIA, B.; RECTOR, A.L., *A Domain Specific Ontology Authoring Environment for a Clinical Documentation System Computer-Based Medical Systems (CBMS)*, in *IEEE 27th International Symposium.* . 2014. p. 329-334.

39. GUERRERO, R.S., *Ontología para la representación de las preferencias del estudiante en la actividad de aprendizaje en entornos virtuales.* Doctorado en Ciencias de la Educación, Tutor A. GARCÍA MARTÍNEZ, in CEPES, 2012, Universidad de La Habana:La Habana, Cuba.

40. PETERSEN, K.A., Nauman Bin., *Identifying strategies for study selection in systematic reviews and maps.*, in *International Symposium on. IEEE*. 2011. p. 351-354.

41. JACOBSON, I.B., Grady; RUMBAUGH, James., *El proceso unificado de desarrollo de software.* 2000, Pearson Educación.

42. VAISHNAVI, V.K., W. *Design Research in Information Systems.* 2005 [cited 2018; Available from: http://desrist.org/design-research-in-information-systems/.

43. ISO/IEC12207, N.T.P.N., *Tecnología de la información. Procesos del ciclo de vida del software.*, C.d.R.T.y. Comerciales, Editor. 2006.

44. HEVNER, A.C., S., *Design research in information systems: theory and practice.* Springer Science & Business Media. , 2010. **22**.

45. CODORNIÚ, C.L., *Solución arquitectónica de la configuración general del sistema para la parametrización de negocio de Cedrux.* Maestría en Informática Aplicada, Tutor Y. PIÑERO PÉREZ, in Facultad 3, 2011, Universidad de las Ciencias Informáticas: La Habana, Cuba.

46. LEÓN, A.R.S.R., Martha Denia Hernández. , *Modelo de descripción de arquitectura de almacenes de datos para ensayos clínicos del Centro de Inmunología Molecular .* Revista Cubana de Ingeniería, 2012. **3**(1): p. 15-20.

47. TRUJILLO, Y., *Modelo para valorar las organizaciones desarrolladoras de software al iniciar la mejora de procesos.* Doctorado en Ciencias Técnica, Tutor A. FEBLES ESTRADA AND G. LEÓN RODRÍGUEZ, in Facultad 2, 2014, Universidad de las Ciencias Informáticas: La Habana, Cuba.

48. OFFERMANN, P.B., S.; Schönherr, M.; Bub, U. , *Artifact types in information systems design science–a literature review.*, in *In International Conference on Design Science Research in Information Systems.*, S.B. Heidelberg., Editor. 2010. p. 77-92.

49. CRUZ, S., Y.; SILEGA, MARTÍNEZ, N.; PARRA, FERNÁNDEZ, A., *Método basado en ontología para representar decisiones de diseño.* Revista Cubana de Ciencias Informáticas., 2018. **12**(2): p. 147-151.

50. CRUZ, S., Y.; et al, *Description and Analysis of Design Decisions: An Ontological Approach.*, in *International Conference on Technologies and Innovation.*, Springer-Cham, Editor. 2018: Guayaquil, Ecuador. p. 174-185.

51. SHAW, M., CLEMENTS, P. C., *A Field Guide to Boxology: Preliminary Classification of Architectural Styles for Software Systems.*, in *21st International Computer Software and Applications Conference (COMPSAC '97)*. 1997: Washington, DC, USA.

52. PÉREZ, P., JULIAN. *Definición.de*. 2008 [cited 2018 11 de octubre]; Available from: https://definicion.de/sistema-operativo/.

53. LÓPEZ, R., Y.; HIDALGO, DELGADO, Y.; SILEGA, MARTÍNEZ, N., *Un método práctico para la evaluación de ontologías*, in *International Workshop on Semantic Web IWSW 2018*. 2018: La Habana.

54. HERNÁNDEZ, S., R.; FERNÁNDEZ, COLLADO, C.; BATISTA, LUCIO, P. , *Metodología de la investigación.* 2 ed. 2003: Editorial Félix Varela.

55. LEÓN, R.A., COELLO, S. , *El proceso de investigación científica.* , ed. E. Universitaria. 2011, La Habana, Cuba. .

56. ARIAS, A.C., *Modelo de madurez de tres perspectivas para evaluar y planificar la adopción de arquitecturas orientadas a servicios en las organizaciones.* Doctorado en Ciencias Técnicas, Tutor V. ESTRADA SENTI AND A. FEBLES ESTRADA, 2012, Universidad de las Ciencias Informáticas (UCI): La Habana, Cuba.

57. CAÑIZARES, R., *Repositorio de recursos educativos para las instituciones de educación superior.* Doctorado en Ciencias Técnica, Tutor V. ESTRADA SENTI AND J. P. FEBLES RODRÍGUEZ, in Facultad 4, 2012, Universidad de las Ciencias Informáticas (UCI): La Habana, Cuba.

58. FEBLES, O., *MIDAC: Modelo para el desarrollo de aplicaciones compuestas basadas en arquitecturas orientadas a servicios.* Doctorado en Ciencias Técnica, Tutor V. ESTRADA SENTI AND J. P. FEBLES RODRÍGUEZ, in Facultad 5, 2012, Universidad de las Ciencias Informáticas (UCI): La Habana, Cuba.

## Anexos

### Anexo 1. Guía de grupo focal sobre decisiones de diseño

**No de participantes:** 11 expertos en el diseño de software

**Fecha:** 8 de marzo del 2018

**Lugar:** laboratorio 302 del Docente 2

**Hora:** 2:30 p.m.

**Nombre del moderador:** Yordani Cruz Segura

**Nombre del Observador:** Leidy Ramos González

**Objetivo de la investigación:** desarrollar un método basado en ontología para representar las decisiones de diseño de software en la actividad productiva de la Universidad de las Ciencias Informáticas, de manera que se agilice el análisis sobre las mismas.

**Descripción del foco de trabajo:** estudios recientes muestran el impacto que tienen las decisiones de diseño en la calidad del producto final. Relacionado con lo anterior se puede constatar que existe abundante bibliografía relacionada con la etapa del diseño, dejando claro que las buenas decisiones están aparejada a varios factores dentro de los que se encuentra la experticia del equipo encargado de desarrollar esta tarea, sobre todo es importante las buenas prácticas de proyectos exitosos o no, desarrollados con anterioridad. Otros factores importantes son las características propias del sistema a desarrollar y la metodología adoptada para su desarrollo. En la Universidad de las Ciencias Informáticas (UCI), se realiza la descripción de las decisiones de diseño en los documentos de arquitectura que se encuentran almacenados en los expedientes de proyectos. Lo anterior trae como dificultad, que se vea afectada la toma de decisiones por parte de aquellos arquitectos y diseñadores que cuentan con poca experticia y que resulte engorroso el proceso de análisis desarrollado sobre las mismas. Partiendo de la definición del cambio deseado hacia la valoración de la situación que presenta la UCI al realizar la descripción

de las decisiones de diseño, el **objetivo del grupo focal** es: conceptualizar las decisiones de diseño que se adoptan en la universidad e identificar recomendaciones para su descripción.

**Descripción de los participantes:** 11 personas, de ellas 3 profesores de la asignatura de Ingeniería de Software, 2 investigadores con estudios publicados en la temática y 6 especialistas de diferentes proyectos de la universidad.

**Guía de la actividad**

- Describir lo que constituye un grupo focal
- Explicar el objetivo de la reunión
- Explicar procedimiento, confidencialidad
- Presentación de cada participante, implica que se le pedirá a cada participante que se presente y que aborde en un minuto su experiencia en el diseño de software
- Ejecución del debate
- Conclusiones

**Guía de preguntas**

- ¿Cómo conceptualizar las decisiones de diseño?
- Identifique a partir de los elementos mostrados, cuáles usted considera que son decisiones de diseño.
- Mencione los elementos que usted considere que son decisiones que se deben tomar durante el diseño de software y no fueron tenidas en cuenta en la pregunta anterior.
- ¿Qué elementos se deben tener en cuenta a la hora de enfrentarnos al proceso de descripción de decisiones de diseño?
- ¿Considera usted que desarrollar un adecuado proceso de descripción de decisiones de diseño influye de manera positiva en la calidad del producto final?

**Anexo 2. Documentos de especificación de requisitos de las ontologías (DERO).**

1. Propósito

El propósito de esta ontología es describir las decisiones que son tomadas durante la etapa de diseño de software.

2. Alcance

La ontología permitirá describir las decisiones de diseño identificadas como parte de la investigación y que aplican a la actividad productiva de la Universidad de las Ciencias Informáticas [12]. Agilizará la realización de estudios estadísticos relacionados con la etapa de diseño de software en la UCI y contribuirá a la toma de decisiones relacionadas con estas.

3. Lenguaje de implementación

Debe estar implementada en el lenguaje de implementación de ontologías OWL usando la herramienta Protégé 5.0.

4. Usuarios finales previstos y administradores

La ontología está destinada en una primera instancia a los diseñadores y arquitectos de software que serán los encargados de representar las decisiones de diseño de cada uno de los proyectos. La herramienta será utilizada también por los directivos de la actividad productiva de la UCI para realizar análisis que apoyen la toma de decisiones, teniendo como base el conocimiento registrado. La ontología será mantenida por los administradores de la misma.

5. Requisitos

a. Requisitos no funcionales

   La ontología debe estar escrita en idioma español.

b. Requisitos funcionales: Grupo de preguntas de competencia

- ¿Cuáles son las decisiones de diseño tomadas para un sistema determinado?
- ¿Cuáles sistemas son desarrollados con una tecnología determinada?

- ¿Cuál es la justificación para tomar una decisión de diseño?
- ¿Cuál es la evaluación otorgada a una decisión de diseño?
- ¿Con cuáles decisiones de diseño está relacionada una decisión determinada?
- ¿Quién fue la persona que tomó una decisión determinada?
- ¿Cuáles decisiones fueron tomadas por un autor determinado?
- ¿A qué sistema pertenece una decisión determinada?
- ¿En qué fecha fue tomada una decisión determinada?

**Anexo 3. Casos de prueba de la ontología para representar decisiones de diseño.**

**Caso de prueba 2**

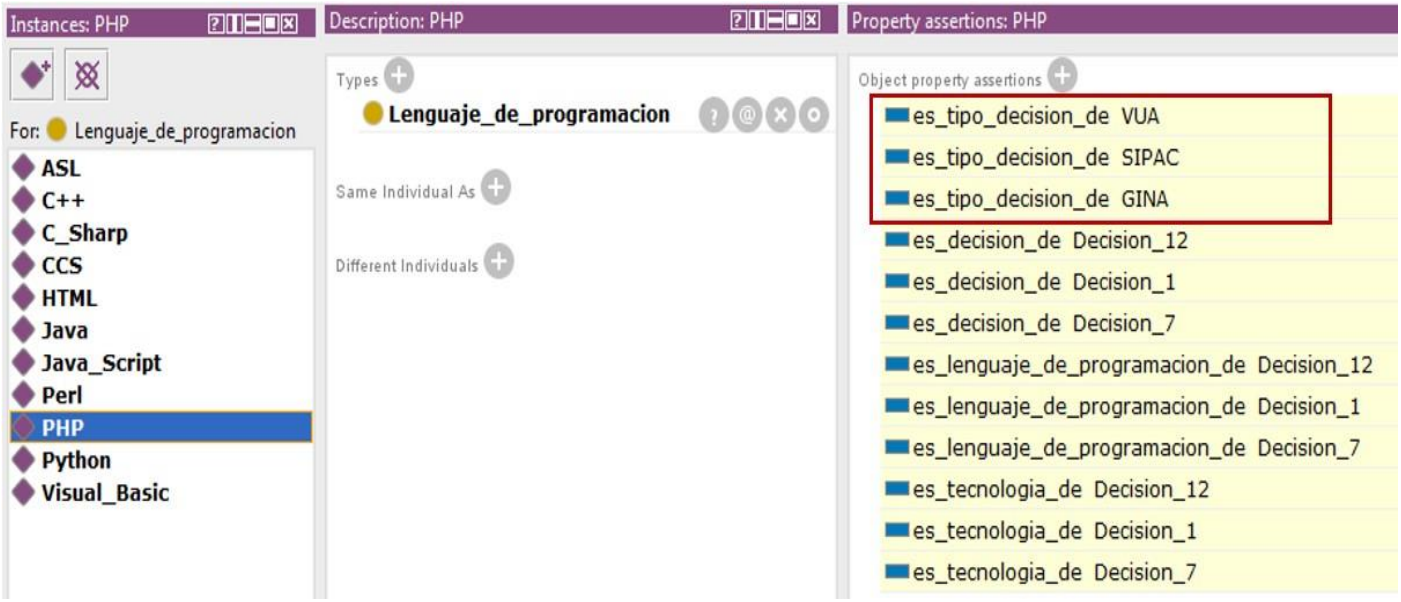
**Pregunta de competencia:** ¿cuáles sistemas son desarrollados con una tecnología determinada?

**Escenario:** en la herramienta Protégé se localiza una tecnología de las registradas, en este caso se selecciona el lenguaje de programación PHP y se inicia el razonador.

**Resultado esperado:** al aplicar un razonador el sistema debe de mostrar los sistemas desarrollados con el lenguaje de programación PHP que se encuentren registrados en la ontología, en este caso se deben de mostrar los sistemas GINA, VUA y SIPAC.

**Resultado obtenido:** satisfactorio. La figura siguiente muestra el resultado obtenido, dentro del recuadro rojo se infieren los sistemas desarrollados con el lenguaje de programación PHP que coinciden con los resultados esperados.

**Resultado del caso de prueba 2**




**Caso de prueba 3**

**Preguntas de competencias:**

- ¿Quién fue la persona que tomó una decisión determinada?
- ¿A qué sistema pertenece una decisión determinada?
- ¿En qué fecha fue tomada una decisión determinada?
- ¿Cuál es la justificación para tomar una decisión de diseño?
- ¿Cuál es la evaluación otorgada a una decisión de diseño?
- ¿Con cuáles decisiones de diseño está relacionada una decisión determinada?

**Escenario:** en la herramienta Protégé se localiza una decisión de las registradas, en este caso se selecciona la *Decisión_5* y se inicia el razonador.

**Resultado esperado:** al aplicar un razonador el sistema debe de mostrar los datos relacionados con la decisión seleccionada, dentro de esos datos se encuentra la persona que tomó la decisión que en este caso debe ser *Raymond_Weeden_Gamboa,* el sistema al que pertenece que en este caso es *GINA,* fue tomada el 05/09/2009, la evaluación es *True* que es satisfactoria, se encuentra relacionada con la *Decision_3*.

**Resultado obtenido:** satisfactorio. La figura siguiente muestra el resultado obtenido, dentro del recuadro rojo se encuentra el autor, el sistema al que pertenece y la decisión relacionada con la *Decision_5*. Dentro del recuadro azul se muestra la evaluación, fecha y justificación de la selección de esta decisión. En todos los casos coincide con el resultado esperado.

**Resultado del caso de prueba 3:**


**Caso de prueba 3**

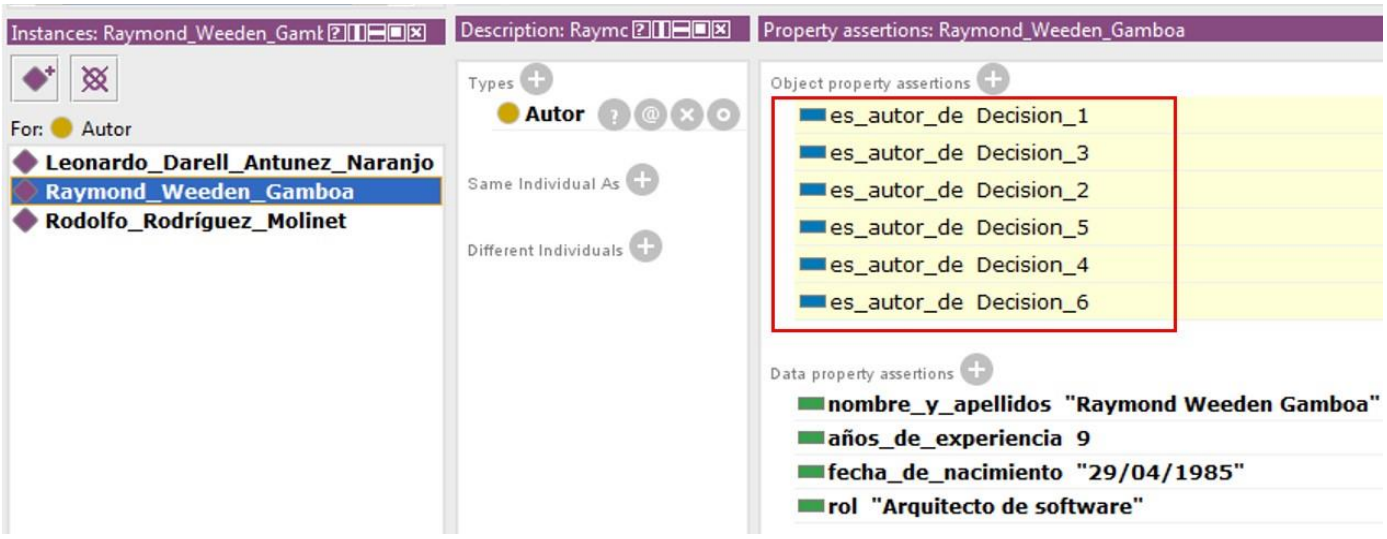
**Pregunta de competencia:** ¿cuáles decisiones fueron tomadas por un autor determinado?

**Escenario:** en la herramienta Protégé se localiza un autor de los registrados, en este caso se selecciona el autor *Raymond_Weeden_Gamboa* y se inicia el razonador.

**Resultado esperado:** al aplicar un razonador el sistema debe de mostrar los datos relacionados con el autor seleccionado, dentro de esos datos se encuentran las decisiones tomadas por él, que en este caso deben ser *Decision_1, Decision_2, Decision_3, Decision_4, Decision_5* y *Decision_6*.

**Resultado obtenido:** satisfactorio. La figura siguiente muestra el resultado obtenido, dentro del recuadro rojo se encuentran las decisiones tomadas por el autor seleccionado, que coincide con el resultado esperado.

**Resultado del caso de prueba 3:**




**Anexo 4. Cuestionario para evaluar la satisfacción de los usuarios finales con la solución**

La presente encuesta tiene como objetivo conocer la satisfacción de los usuarios finales con el método basado en ontologías para representar decisiones de diseño. Sus valoraciones servirán de ayuda en la mejora continua de la solución propuesta.

1. Luego de conocer el método desarrollado para la descripción de decisiones de diseño basado en ontología, refleje en qué medida le gusta la solución desarrollada marcando con una X donde considere.

____ Me gusta mucho ____ No me gusta tanto ____ Me da lo mismo

____ Me disgusta más de lo que me gusta ____ No me gusta nada ____ No sé qué decir

2. ¿Considera usted que la descripción de decisiones de diseño sin herramienta de apoyo es eficiente? Marque con una X donde considere.

_____Sí _____No _____No sé

3. ¿Utilizaría el método que se presenta en esta investigación como una herramienta para describir las decisiones de diseño?

_____Sí _____No _____No sé

4. ¿Qué elementos considera positivos de este método?
5. ¿Qué elementos considera negativos de este método?
6. ¿Qué sugerencias tiene para el desarrollo e implantación de este método?