Automatización del Modelo de Toulmin

Camilo Andrés Toledo Parra

Universidad Industrial de Santander
Facultad de Ingenierías Físico-Mecánicas
Escuela de Ingeniería de Sistemas e Informática
Bucaramanga
2012

Automatización del Modelo de Toulmin

Camilo Andrés Toledo Parra

Trabajo de Grado para optar al título de Ingeniero de Sistemas

Director
PhD. Sonia Cristina Gamboa Sarmiento

Codirector
Doctorando Juan Carlos Rincón Acuña

Universidad Industrial de Santander
Facultad de Ingenierías Físico-Mecánicas
Escuela de Ingeniería de Sistemas e Informática
Bucaramanga
2012

Dedicatoria

A mi familia por todo su apoyo y sacrificio;
a mi madre hermosa, por darme ese amor especial
a mis maestros por esas grandes enseñanzas y su dedicación;
a esta grandiosa universidad por el respaldo brindado;
al grupo de investigación por resaltarme el valor del trabajo en equipo y
a mis amigos y compañeros durante la inolvidable vida universitaria, los cuales aportaron mucho a mi construcción y fortalecimiento.
Hoy, con el fin de esta etapa, comienza un nuevo horizonte.
En memoria de mi Petronila.

Tabla de contenido

Pág

Lista de tablas

Pág.

Lista de figuras

Pág.

Lista de anexos

Pág

TITULO: AUTOMATIZACIÓN DEL MODELO DE TOULMIN*

AUTOR: CAMILO ANDRÉS TOLEDO PARRA**

PALABRAS CLAVES: MODELO DE TOULMIN, AUTOMATIZACIÓN, ARGUMENTACIÓN, DEMOCRACIA, TICS, BIG DATA.

Ésta investigación es enfocada a manejar la problemática de la enseñanza de la argumentación en las personas, específicamente para estudiantes –debido a su fácil acceso a éstas- por medio de las tecnologías de información y comunicación y el estudio filosófico realizado por Stephen Toulmin en su libro *Los usos de la argumentación*. El Modelo de Toulmin ofrece una estructura bien definida para la elaboración de un argumento desde el punto de vista de la nueva retorica. La complejidad del problema radica en las diversas formas posibles de solución relacionada estrechamente con la clase y tipo de argumentos que pueden ser analizados en el artefacto. A continuación se presentan dos formas de abordar el problema de categorización y reconocimiento del lenguaje natural con el cual se realizan los argumentos. Uno de los resultados más significativos es la propuesta de diseño arquitectónico de un sistema basado en lógica paraconsistente por medio de herramientas tecnológicas para el procesamiento del lenguaje natural y redes neurales con el fin de extraer información y reconocer nombres de entidades. Esto con el fin de que el artefacto pueda ser capaz de trabajar con argumentos extensos, denominado en el mundo digital como Big Data, como por ejemplo: en el campo judicial, campo ético, campo filosófico y demás campos; los cuales presentan un mayor grado de dificultad para reconocer sus estructuras, pero de igual forma hacen parte de la vida práctica.

---

* Trabajo de investigación

** Facultad de Ingeniería Físico Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director Sonia Cristina Gamboa Sarmiento. Codirector Juan Carlos Rincón Acuña.

TITLE: AUTOMATIZACIÓN DEL MODELO DE TOULMIN*

AUTHOR: CAMILO ANDRÉS TOLEDO PARRA**

KEYWORDS: MODELO DE TOULMIN, AUTOMATIZACIÓN, ARGUMENTACIÓN,

DEMOCRACIA, TICS, BIG DATA.

This research is focused on managing the problems of teaching of argumentation in people, specifically for students, because of its easy access to them, by means of information and communication technologies and the philosophical study conducted by Stephen Toulmin in his book the uses of argument. The Toulmin Model provides a well defined structure for the development of an argument from the point of view of the new rhetoric. The complexity of the problem lies in the various possible ways of solution closely related to the class and type of arguments that can be analyzed in the artifact. Here are two ways to address the problem of categorization and natural language recognition with which the arguments are made. One of the most significant is the proposed architectural design of a system based on paraconsistent logic through technological tools for natural language processing and neural networks to extract information and recognize entity names. This in order that the device may be able to work with extensive grounds, known in the digital world as Big Data, for example in the legal, ethical field, field of philosophy and other fields, which have a higher degree of difficulty recognizing their structures, but equally are part of practical life.

---

* Trabajo de investigación

** Facultad de Ingeniería Físico Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director Sonia Cristina Gamboa Sarmiento. Codirector Juan Carlos Rincón Acuña.

1. INTRODUCCION

Desde los tiempos de la Grecia Antigua se ha concebido la democracia como una forma de convivencia y de gobierno que garantiza el ejercicio de la ciudadanía y la participación de los sujetos en las decisiones colectivas, incluyendo las decisiones de estado; pero esta concepción requiere que todos los ciudadanos aprendan y practiquen la argumentación, como una habilidad, pero también como un conjunto de estructuras y procesos.

La democracia exige que todos los ciudadanos participen activamente en la construcción de proyectos colectivos mediante el debate de las ideas, que, en la medida que buscan persuadir a otros ciudadanos, tendrían que estar fundadas en la argumentación. Esto debido a que el uso de la argumentación implica que se renuncie a otros métodos para lograr la adhesión de los demás: como la violencia, la seducción o la amenaza.

Se requiere, entonces, crear entornos y estrategias para el aprendizaje de la argumentación como parte de los procesos de formación en los que están inmersos los jóvenes, con el fin de lograr que su interacción como sujetos sea más basada en lo racional, en el respeto y el reconocimiento del otro, y en la comprensión de sus argumentos.

Quienes están en la mejor posición de aprender y de contar con mejores entornos de aprendizaje son los estudiantes universitarios o estudiantes de educación media, es decir, personas que se encuentran en el espacio de las instituciones educativas. En estos espacios los estudiantes están dispuestos a aprender, y es allí donde es posible diseñar estrategias didácticas para lograr el aprendizaje de la argumentación.

De otra parte, teniendo en cuenta que las llamadas tecnologías de la información y la comunicación –TIC– se han insertado de tal manera en las actividades cotidianas de las personas, que su uso parece transparente, tales tecnologías son hoy en día potencialmente atractivas para los estudiantes, de manera que si se implementan en ellas tales estrategias didácticas, éstas pueden llegar a los estudiantes de manera natural.

Se propone, entonces, crear una aplicación como medio o interfaz para propiciar en estudiantes el aprendizaje de la argumentación, especialmente, en el enfoque de Stephen Toulmin, que propone un modelo para construir argumentos.

Titulo	Automatización del Modelo de Toulmin para construir argumentos
Objetivo general	Construir una aplicación computacional en la cual se implementen las estructuras propuestas por Stephen Toulmin en el Modelo de Toulmin.

Objetivos específicos	Resultados
Formalizar los procesos argumentativos, según el Modelo de Toulmin, en diagramas de flujo y algoritmos en cláusulas lógicas.	En primer lugar se realiza un análisis exhaustivo y riguroso del libro Los usos de la argumentación, de Stephen Toulmin, para comprender la estructura del Modelo de Toulmin y poder describirla y representarla en un artefacto tecnológico.
Diseñar un dispositivo computacional en el que se implementen tales cláusulas.	Para describir los procesos y métodos usados que se deben realizar para que cumplan los requerimientos, resultado de los análisis, se especifican mediante modelos UML: modelo funcional; modelo de objetos; modelo dinámico. Con esto se plantea la arquitectura: Netbeans; Java Server Faces 2.0; PrimeFaces; Maven; Maxent; OpenNLP; Protégé; JENA; SPARQL.
Validar la estructura argumentativa de Stephen Toulmin en un dispositivo computacional	Bajo el concepto de interoperabilidad [1] se llegaron a dos posibles soluciones planteadas. Una, abarcando totalmente el problema con máquinas de aprendizaje: redes neurales, con mayor robustez, flexibilidad y escalabilidad, pero con mayor costo de tiempo de implementación. Otra, por medio de la combinación de redes neurales y ontologías, más práctica y ligera con el fin de automatizar el modelo de Toulmin pero siendo ésta rígida y menos dinámica. La última solución fue la escogida para dar cumplimiento a la funcionalidad, validar la estructura del modelo de Toulmin presentada en un argumento de forma automática.

Tabla 1. ObjetivosMarco teórico

[1]La habilidad de organizaciones y sistemas dispares y diversos para interaccionar con objetivos consensuados y comunes y con la finalidad de obtener beneficios mutuos. La interacción implica que las organizaciones involucradas compartan información y conocimiento a través de sus procesos de negocio, mediante el intercambio de datos entre sus respectivos sistemas de tecnología de la información y las comunicaciones.

1.1 Argumentación

La argumentación tiene como objetivo que determinado auditorio, conformado por una o varias personas, se adhiera a las tesis presentadas en discursos o construcciones coherentes o internamente consistentes, es decir, basadas en la lógica de primer orden; pero, a diferencia de las posibilidades que ofrece la lógica de primer orden, mediante la cual es posible establecer el valor de verdad (verdadero o falso) de una conclusión a partir de operaciones entre sus premisas, en la argumentación se presenta la situación de que no es posible establecer un valor de verdad para las conclusiones, pues, precisamente, se requiere argumentar aquello para lo que no es posible establecer un valor de verdad (como sucede, por ejemplo, con los hechos), y su aceptabilidad depende de qué tan creíble o plausible resulta para quien hace las veces de auditorio.

Según esto, una mejor argumentación, una argumentación eficaz es "(…)la que consigue aumentar esta intensidad de adhesión de manera que desencadene en los oyentes la acción prevista (acción positiva o abstención), o, al menos, que cree en ellos una predisposición que se manifestará en el momento oportuno." (Perelman&Olbrechts-Tyteca, 2006, p. 91). Aristóteles propone tres géneros argumentativos según si las acciones que se busca desencadenar sean con respecto al pasado, al presente o al futuro, respectivamente: judicial, que consiste en la defensa o en la acusación con respecto a un hecho ya sucedido; epidíctico, que se caracteriza por el elogio o la censura; y deliberativo, que ofrece un consejo o busca la disuasión para acciones futuras (Cf. *Ret.*, 1358b7-20).

El Modelo de Toulmin privilegia el análisis de la estructura de los discursos deliberativos o justificatorios: "este tipo particular de argumentos recibirá la mayor parte de nuestra atención, (…) nuestro interés se centrara en los argumentos justificatorios utilizados para apoyar afirmaciones, en las estructuras que pueden tener, en el valor que pueden reivindicar para sí y en el modo en que nos enfrentamos a ellos al clasificarlos, nos formamos un juicio sobre ellos y los criticamos" (Toulmin, 2007, p.30).

Sonia Gamboa (2011) realiza un estudio sobre la argumentación según el enfoque de Aristóteles, el cual "(…) abarca el estudio de la retórica como un conjunto de técnicas para hablar en público de forma persuasiva. Este estudio delimita el ejercicio de la retórica a discursos acerca de temas determinados, pronunciados ante multitudes que requieren ser caracterizadas previamente para la construcción del aquél y de las acciones que se buscan" (Gamboa, 2011, p.50); es decir, delimita la argumentación y un tipo de discurso retórico que se dirige a un auditorio

previamente determinado, pero solamente en sus características general y no en sus particularidades.

En la argumentación, según Gamboa, "(…) los autores han dejado de lado el interés por el estudio de la elocución para ubicarse solamente en el tratamiento de los medios discursivos para persuadir o convencer. Nueva retórica se refiere entonces a una forma de retórica que no considera ni modificar el discurso ni realizar acciones sobre la marcha con el fin de persuadir a la audiencia, pero no porque le reste importancia a su aceptación, sino porque tal aceptación tiene que lograrse desde el discurso mismo" (Ibíd., p.52).

1.2 Modelo de Toulmin

Siendo el discurso lo que soporta una afirmación, y éste permanece en el tiempo y espacio de manera virtual, se hace indispensable el análisis sobre la estructura del discurso en que está soportada dicha afirmación para decidir sobre la validez que tenga para aquellos a quienes va dirigida la argumentación, es decir: el *auditorio universal*[2].

El modelo de Toulmin es un camino para evaluar los procesos argumentativos, éste describe un procedimiento que logra representar patrones y variables en el proceso de la construcción de un argumento, las cuales están relacionadas con la estructura y ésta determina la dependencia entre el valor de una afirmación y los "fundamentos o razones en los que se apoya, datos, hechos, pruebas, consideraciones, componentes" (Toulmin, 2003, p. 30); así, para Toulmin, la argumentación consiste en el conjunto de "los argumentos justificatorios utilizados para apoyar afirmaciones, en las estructuras que pueden tener, en el valor que puede reivindicar para sí y en el modo en que nos enfrentamos a ellos al clasificarlos" (Íd.); con base en estas consideraciones "nos formamos un juicio sobre ellos y los criticamos" ((Toulmin, 2003, p. 30)).

Toulmin plantea como proceso de análisis de un argumento su representación o enunciación como una estructura de invariantes, y el proceso de construcción de la misma; la estructura está determinada por la forma en la que se enlazan los

[2] *Auditorio universal* no se refiere al conjunto de todos estos posibles oyentes, sino a cada uno de los que encarnan, de una manera intemporal y absoluta, las características de lo que se ha definido como *auditorio*, de manera que la argumentación se experimenta por cada sujeto, como encarnación del *auditorio universal*, en primera persona, y su adhesión o rechazo a las tesis argumentadas se produce como resultado del análisis individual de cada sujeto miembro de tal *auditorio*. (Gamboa, 2011, p.52)

enunciados que soportan la afirmación final o conclusión; el proceso de construcción consiste en las *fases de un argumento* que a continuación se describen:

Formular clara y explícitamente un problema cuya solución se presume que puede ser soportada a través de un conjunto de enunciados, argumentos. Toulmin señala que la mejor forma es “planteando una pregunta clara; incluso si, como sucede frecuentemente, nos limitamos a apuntar sólo la naturaleza de nuestra confusa búsqueda de modo interrogativo” (Toulmin, 2003, p. 35).

Presentar los enunciados posibles como solución al problema, pregunta, planteado al inicio; evaluar la consistencia de cada uno y encontrar cuáles enunciados se descartan y con cuales contamos debido a su peso y grado de credibilidad de acuerdo a la información disponible.

La última fase, “se otorga un veredicto y se pronuncia la sentencia u otro acto judicial derivado del veredicto” (Toulmin, 2003, p. 35).

Toulmin asume como *forma estándar* de un argumento la estructura lógica propuesta por Aristóteles “«premisa menor, premisa mayor; *por tanto*, conclusión»” (*Ibíd*., p. 131), pero parecería que esta estructura no es suficientemente elaborada para argumentos más complejos, como los que son propios del Derecho, por ejemplo. A continuación se sintetiza la estructura propuesta por Toulmin:

Primero se formula una conclusión (C) que no se evidencia por sí misma, para ello debe contar con un conjunto de *datos* (D) que justifican tal afirmación; es posible que se constituya aceptable la afirmación formulada (C) como la conclusión del conjunto de datos D, pero también existe la posibilidad que se requieran establecer un conjunto de proposiciones, llamadas garantías (G) las cuales establecen una conexión legitima y fuerte entre D y C. Toulmin realiza una distinción entre *datos* y *garantías*: “a los datos se apela explícitamente, a las garantías implícitamente. (…) las garantías son generales, certificando la validez de todos los argumentos del tipo correspondiente, por lo que tienen que establecerse de manera muy diferente que los elementos justificatorios que ofrecemos como datos” (Ibíd., p. 136). Además Toulmin también es consciente que existen excepciones a la regla y que es necesario saber con qué fuerza se ejercen las garantías a las conclusiones, estos dos nuevos elementos los propone: “Los calificativos o matizadores modales (M) y las condiciones de excepción o de refutación (E)”(Toulmin, 2003); M indica la fuerza que le da la garantía entre D y C; E son circunstancias que representan excepciones o refutaciones. La estructura del modelo de Toulmin es:

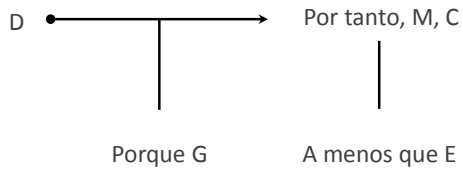


Figura 1. Esquema de análisis de un argumento (Tomada de Toulmin, ob. cit., p. 138)

Esta perspectiva de estudio de la argumentación orienta a un discurso estructurado de tal manera que una conclusión se hace creíble, probable, plausible en la medida que haya evidencias, motivaciones y garantías que la soporten.

1.3 Virtualidad

La construcción de un país con un estado democrático en el contexto actual es posible mediante los entornos virtuales, entendidos como espacios en los que se propicia la experiencia de elementos y contextos reales mediante desarrollos computacionales. Pierre Levy propone: "(…) virtual procede del latín medieval *virtualis*, que a su vez deriva de *virtus*: fuerza, potencia. En la filosofía escolástica, lo virtual es aquello que existe en potencia pero no en acto." (Levy, p.10).

La virtualización se puede comprender, de manera general, como la creación de un proyecto, sobre un ente, objeto, persona natural o jurídica, el cual engendra problemas, partiendo de lo actual, mas no como buscando una solución a un problema; más bien una visión de una transformación por medio de una misión: "La virtualización puede definirse como el movimiento inverso a la actualización. Consiste en el paso de lo actual a lo virtual, en una «elevación a la potencia» de la entidad considerada (...)una mutación de identidad, un desplazamiento del centro de gravedad ontológico del objeto considerado: en lugar de definirse principalmente por su actualidad(Una «solución»), la entidad encuentra así su consistencia esencial en un campo problemático" (Levy, 2004, p.12).

En contraste "La actualización aparece entonces como la solución a un problema, una solución que no se contenía en el enunciado" (Levy, 2004, p.11). Desde esta perspectiva se puede observar el problema existente actualmente sobre la falta de proyección y la necesidad de generar y obtener resultados inmediatos que impera en la sociedad colombiana; el dinero fácil, la baja calidad en escuelas, colegios y universidades debido a la intensión de crecer económicamente pero en realidad

colocando al país en una posición desfavorable respecto al futuro. Debido a que se ha olvidado la potencia que existe en visionar los objetivos y planear acciones y estrategias para lograrlos a diferencia de la solución continua de problemas que no tienen relación con la búsqueda de objetivos; es decir la virtualización hace que uno mismo ubique qué clase de problemas quiere solucionar para conseguir un estado deseado, en cambio la actualización está en un estado deseado resolviendo problemas para continuar en ese estado deseado.

Entonces la virtualización llega a ser de mayor importancia al reconocer el carácter y la actitud crítica que deben tener las personas quienes creamos la democracia o por defecto no la creamos. Crítica debido al efecto que tiene la virtualidad en su proyección y para esto se vale de artefactos como libros o el hipertexto en internet.

La argumentación, una vez más, se abre un espacio interesante y trascendental en el mundo debido a que pertenece a lo actual, y es sobre la actualidad misma la que queremos llevar a una virtualización y encontrar lo real: “Si la virtualización no fuera más que el paso de una realidad a un conjunto de posibles, sería desrealizante. Sin embargo, implica tanta irreversibilidad en sus efectos, indeterminación en sus proceso se indeterminación en su esfuerzo como la actualización. La virtualización es uno de los principales vectores de la creación de realidad.”(Levy, 2004, p.13).

La herramienta puede ser un autómata que virtualmente ponga al estudiante en la posición de construir un argumento teniendo en cuenta la estructura fundamental del Modelo de Toulmin, es decir, ser usada como herramienta didáctica para motivar la lectura crítica de textos históricos y literarios, y la composición de nuevos textos con sólidas estructuras argumentativas. La herramienta será un servicio web, aprovechando una de las principales características de las TIC, ya que éstas permiten crear discusiones estructuradas alrededor de temas específicos, entre personas en distintos lugares geográficos; con ellas es posible *hacer clase virtual,* a distancia, con la exigencia de que los estudiantes deben contar con computadoras, pero también para hacer valer su presencia posiblemente tendrán que acudir a formulaciones escritas. Las TIC más que herramientas de uso necesario son un espacio potencial de encuentro. La interacción de sujetos en el ciberespacio ha dado lugar a la conformación de la *cibercultura*, esta forma de cultura “podría convertirse en un medio de exploración de los problemas, de discusión pluralista, (…) [de] hacer visibles procesos complejos, de toma de decisión colectiva y de evaluación de los resultados cercanos a las comunidades en cuestión (…) El uso socialmente más útil de la informática de comunicación es sin dudas proporcionar a los grupos humanos los

medios para mancomunar sus fuerzas mentales para constituir colectivos inteligentes y hacer vivir una democracia en tiempo real" (Lévy, 2004, p. 41).

La aproximación al lenguaje natural es un subconjunto controlado de palabras y sintaxis de acuerdo al dominio particular de un argumento basada en la estructura del Modelo de Toulmin.

Para establecer un escenario que permita evaluar la efectividad de la solución propuesta, se escogió un caso de estudio, el cual es un ejemplo planteado en el libro *Los usos de la argumentación* de Stephen Toulmin, para el cual se desarrolla una ontología respecto a su sentido.

La creación del prototipo que permita acceder a la ontología por parte de usuarios y así poder consultar la información que contiene la ontología es uno de los objetivos de este proyecto de investigación. Para esto es necesario proponer una arquitectura software que se ajuste al Modelo de Toulmin.

- La aplicación debe Interpretar una solicitud realizada por un actor, estudiante, a través de una aproximación al lenguaje natural.
- Dicha aproximación al lenguaje natural tiene en cuenta el vocabulario del argumento respecto a la estructura del Modelo de Toulmin.
- La ontología que incorpora el conocimiento relacionado con el modelo de Toulmin deberá ser consultada por medio de esta arquitectura.

ALTERNATIVAS PARA LA SOLUCION

Durante la investigación sobre la Automatización del Modelo de Toulmin se presentaron dos caminos para recorrer el problema y cumplir los objetivos propuestos:

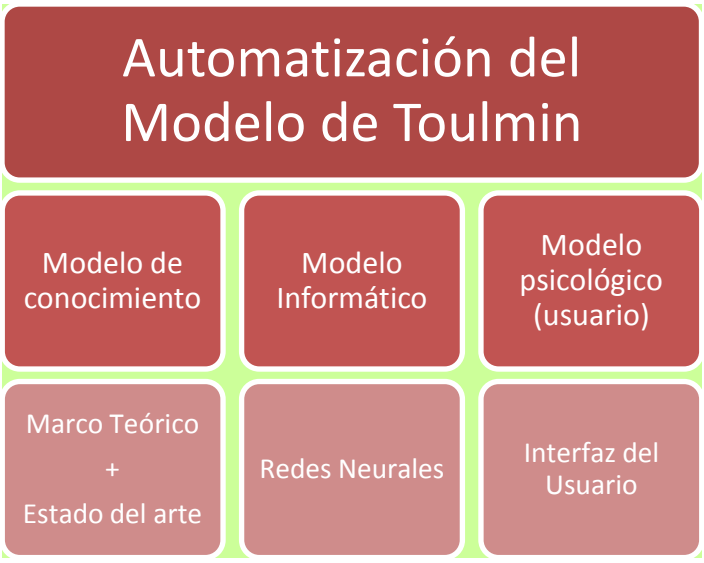


Figura 3. Solución propuesta dominando problemas de Big-Data con máxima entropía y máxima interoperabilidad. (Lógica paraconsistente)

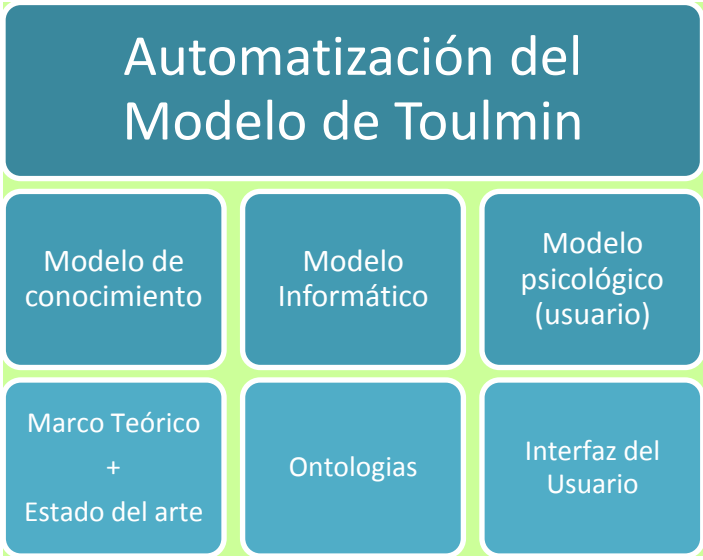


Figura 2. Solución desarrollada (Lógica de primer Orden)

El contraste entre la figura 2 y figura 3 se debe al tipo de lógica utilizada para manejar el problema. Veremos en la sección 2 (Estado del Arte) las diferencias de utilizar lógica paraconsistente y lógica de primer orden. El propósito de la investigación actual también es poder dejar un modelo para desarrollar una herramienta tecnológica con las herramientas de punta para estos tipos de problemas.

Primero se describe la solución desde el punto de vista de la figura 2. Es decir, donde el modelo informático es basado en ontologías.

1.4 Ontologías

Las ontologías son ideales y se ajustan para la automatización del modelo de Toulmin. A continuación se describe sus características y las herramientas que

existen para realizar consultas acerca de la información en ellas representada, y su codificación a un lenguaje estándar de la Web semántica como OWL.

Existen varias definiciones de ontología, Gruber (1995) realiza un buen acercamiento de ontología para el propósito de esta investigación: “Una ontología es una especificación formal de una conceptualización compartida”. Una ontología puede representar el conocimiento de un argumento de tal forma que la información almacenada en repositorios de datos tenga significado para el artefacto, es decir que esta es la base de conocimiento de SYSTOUL, haciendo posible la realización de inferencias a partir de axiomas sobre un dominio particular.

Las ontologías están clasificadas según el problema que se quiere modelar o representar, un acercamiento a esta clasificación la brinda Breis (2008) (ver Tabla 2):

Tipo	Descripción
Ontologías de Nivel Superior	Son aquellas que permiten modelar niveles abstractos de una realidad, generando conceptos que pueden ser usados de forma genérica para clasificar términos.
Ontologías Generales	Éstas permiten representar elementos como el tiempo, el espacio, eventos, entre otros. La importancia de este tipo de ontologías radica en que dichos elementos pueden ser usados en cualquier dominio.
Ontologías de Dominio	Estas ontologías sirven para representar elementos particulares de determinado ámbito, teniendo en cuenta los objetivos y las restricciones que impone el área de aplicación.

Tabla 2. Clasificación de Ontologías.

El tipo de ontología que se usa para automatizar el modelo de Toulmin es de dominio. Este dominio es el establecido por el argumento o los argumentos que dispone el artefacto con el fin de ser el validador de los textos ingresados por los estudiantes.

Las ontologías se desciben en **RDF** (Resource Definition Framework). RDF es un marco para la descripción de recursos que establece un modelo de datos para objetos (recursos) y relaciones entre ellos. “Este tipo de modelo de datos puede ser representado con una sintaxis XML y su estructura proviene de la lógica con componentes de lingüística en donde se parte de tres entidades lógicas que corresponden a los componentes” (Codina & Rovira, 2006, p. 22)

1.4.1 Lenguajes para definir ontologías

"Los lenguajes de descripción son lenguajes cuya función principal no es hacer programas, sino describir cómo se representan los datos, como por ejemplo, páginas web (HTML), ecuaciones (MathML, LaTeX, CML), representaciones geográficas (GML), puntos geográficos (waypoints), subtítulos (VobSub, OGM), animaciones (SMIL), imágenes vectoriales (SVG), representaciones 3D (X3D), etc."(RDF, 2012)

Para inferir conocimiento de las ontologías es necesario que estén expresadas en un lenguaje compatible con las características del entorno Web y que estén disponibles en artefactos conectados a dicho entorno. Para ello existen lenguajes y herramientas para abordar este problema:

Entidades lógicas	Elementos lingüísticos
Recursos	Sujeto
Propiedades	Predicado
Valores	Objeto

Tabla 3. Comparación de lenguajes formales para consulta de ontologías.

El sujeto es el recurso, es decir aquello que se está describiendo. El predicado es la propiedad o relación que se desea establecer acerca del recurso. Por último, el objeto es el valor de la propiedad.

Con los tres elementos anteriores se pueden formar declaraciones sobre recursos en la forma de: *el recurso X tiene la propiedad Y con el valor P.* Ejemplo de una tripleta Recurso-propiedad-valor:

Harry	Nació en	Bermuda
Recurso	Propiedad	Valor

Tabla 4. Equivalencias lógico-lingüísticas en una declaración RDF.

Aunque RDF permite dar valores a las propiedades de los recursos, no dispone de mecanismos para describir esas propiedades ni las relaciones existentes entre ellas y otros recursos. Es por esto que se requiere de un lenguaje que permita definir vocabularios RDF. Dicho lenguaje, construido mediante RDF, es RDF Schema o RDFS. Según Brickley&Guha (n.d.), en él se definen clases y propiedades que permite, describir nuevas clases, propiedades y recursos. Sin embargo, RDF y RDFS no son capaces por sí solos de modelar ontologías, por tal razón se han desarrollado nuevos lenguajes tales como OWL para este fin, con la diferencia de que estos se construyen sobre el estándar RDFS.

“El lenguaje OWL se convirtió en una recomendación del W3C para la construcción de ontologías en febrero de 2004. La última versión es OWL 2, que proporciona más primitivas de modelado, mayor cardinalidad y tipo de datos ampliados y compatibilidad con las anotaciones de la especificación del lenguaje original.”(Abrahams, Condliffe, & Zeleznikow, 2011).

Web OntologyLanguage (OWL) McGuinness& Van Harmelen (2004) definen OWL como el lenguaje estándar de la web semántica para expresar y codificar ontologías. Tiene como objetivo facilitar un modelo de marcado construido sobre RDF y codificado en XML, permite formalizar las relaciones entre las clases de una forma más amplia que RDFS, indicando aspectos básicos para el razonamiento como la existencia de clases disjuntas. El lenguaje OWL tiene tres sub-lenguajes en donde se aumenta el nivel de expresividad en cada uno:

OWL Lite. Está diseñado para aquellos usuarios que necesitan principalmente una clasificación jerárquica y restricciones simples.

OWL DL. Está diseñado para aquellos usuarios que quieren la máxima expresividad conservando completitud computacional[3] y resolubilidad[4].

OWL Full. Está dirigido a aquellos usuarios que necesitan la máxima expresividad y la libertad sintáctica de RDF pero sin garantías computacionales. Permite, por ejemplo, aumentar el significado de vocabulario predefinido (en RDF o en OWL), por lo que es muy improbable que ningún software de razonamiento sea capaz de soportar razonamiento completo para cualquier característica de OWL Full. El lenguaje escogido para implementar la ontología de la automatización del Modelo de Toulmin es OWL-DL dado que se requiere que siempre se dé una respuesta al usuario, resolubilidad, y debe garantizar que las afirmaciones escritas por el usuario siempre se computen con el fin de validar si éstas pertenecen al dominio de la ontología.

1.4.2 Construcción de ontologías

A continuación se estudian unas herramientas para construir, reutilizar o modificar ontologías.

Nombre	Tipo	Entorno Colaborativo	Servicios	Otras Características

[3] Se garantiza que todas las conclusiones sean computables.
[4] Todos los cálculos se resolverán en un tiempo finito.

Protégé	Open Source	Si	- Creación. - Visualización. - Manipulación.	Modelado a través de Marcos y OWL. Soporte para múltiples formatos.
Ontolingua[5]	Open Source	Si	- Análisis. - Creación. - Edición. - Intercambio.	Proporciona una biblioteca de ontologías modulares y reutilizables.
WebOnto[6]	Open Source	Si	- Visualizar. - Crear. - Editar.	Facilita a los usuarios la ampliación de ontologías de gran tamaño.
Neon[7]	Open Source	Si	- Creación. - Mantenimiento.	Ofrece un plugin para cada actividad especificada en la metodología de desarrollo Neón.
Kaon[8]	Open Source	No	- Creación. - Almacenamiento. - Recuperación. - Mantenimiento. - Aplicación.	Ofrece otro tipo de herramientas para construir aplicaciones basadas en ontologías.
OntoStudio[9]	Comercial	Si	- Modelado. - Creación. - Mantenimiento. - Herramientas de mapeo. - Editor gráfico de reglas. - Entorno integrado de pruebas.	Soporte para varios formatos.

Tabla 5. Herramientas para construcción de ontologías. (Ruiz &Belalcázar, 2012, p. 19)

Protégé es la herramienta que mejor se ajusta a esta investigación dado a su amplia documentación, soporte, interfaz amigable para el usuario y es open source.

1.4.3 Gestión de ontologías

Estas herramientas se usan para consultar la información contenida en las ontologías.

Nombre herramienta	Descripción	Características
Jena	Es un marco Java para construir aplicaciones de la web semántica, provee	Jena considera que la abstracción Java del recurso es sólo una vista del

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

[6] http://projects.kmi.open.ac.uk/webonto/

[7] http://neon-toolkit.org/wiki/Main_Page

[8] http://kaon.semanticweb.org/main_kaonOverview.pdf/view

[9] http://www.ontoprise.de/en/home/products/ontostudio/

	un entorno de desarrollo para RDF, RDFS, OWL, SPARQL (éste último utilizado para realizar consultas a ontologías) permite gestionar todo tipo de ontologías (añadir hechos, borrarlos editarlos)almacenarlas y realizar consultas contra ellas, además cuenta con un motor de inferencia basado en reglas, es por tal motivo que Jena cuenta con APIs especiales para cada uno de los lenguajes anteriormente mencionados además da soporte de persistencia.	mismo. Incluye los siguientes componentes: - ARP: un parser de RDF - API RDF - API de Ontologías para OWL, DAML y RDF Schema. - Subsistema de razonamiento - Soporte para persistencia - RDQL y SPARQL: Lenguajes de consultas de RDF
pOWL	Es un marco de trabajo en PHP[10]para aplicaciones de la Web Semántica, es de código abierto y permite el análisis sintáctico, almacenamiento, consulta, manipulación, control de versiones y serialización de RDF-S y bases de conocimiento en OWL en un entorno web colaborativo	Presenta características tales como: - Navegación y edición de ontologías RDF-S/OWL. - Edición de datos visual. - Extensible según la filosofía plug-in. - Sistema de consulta RDQL. - Autenticación en el modelo. - Banco de pruebas para implementaciones rápidas

Tabla 6. Herramientas para consultar ontologías. (Ruiz &Belalcázar, 2012, p.20)

Jena ofrece características que se ajustan perfectamente para abordar la solución al problema de esta investigación. Además es open-source.

1.4.4 Lenguajes formales para consultar ontologías

Lenguaje formal	Descripción
RQL	Se basa en un modelo gráfico formal que captura las primitivas de modelado RDF y permite la interpretación de las descripciones de los recursos. Adapta la funcionalidad de los lenguajes de consulta semi-estructurados o XML a las peculiaridades de RDF. Permite el uso de variables tanto para denotar clases y propiedades, tiene funcionalidades para consultar esquemas RDF, RDF-S, y descripciones RDF en una misma consulta. RQL está definido por medio de un conjunto de consultas básicas, e iteradores que permiten construir otras consultas a través de una composición funcional.
	Sesame RDF QueryLanguage es un lenguaje de consulta en RDF o RDF Schema

[10]Hypertext Pre-processor (PHP) es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Se usa principalmente para la interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica.

Lenguaje formal	Descripción
SeRQL	que es desarrollado como parte del software Sesame. Combina los mejores aspectos de otros lenguajes de consulta (p.ej. RQL, RDQL, N3, etc.) con algunos añadidos propios
RDQL	RDF Data QueryLanguage es un lenguaje de consultas para RDF desde un enfoque totalmente declarativo. Considera un modelo RDF como un conjunto de tripletas: (Objeto, Propiedad, Valor).Permite especificar patrones que son mapeados contra las tripletas del modelo para retornar un resultado.
SPARQL	Protocol and RDF QueryLanguage, es un lenguaje estandarizado de recuperación para RDFs, permite centrarse en la información que se desea tener recopilada, sin tener en cuenta la tecnología de la base de datos o el formato utilizado para almacenar esos datos, existen tres implementaciones de SPARQL las cuales se ocupan de aspectos como el lenguaje de consulta, el formato usado para la devolución de las consultas y el motor para almacenamiento y recuperación de datos.

Tabla 7. Lenguajes formales para consultar ontologías. (Ruiz &Belalcázar, 2012, p. 21)

Los estudios hechos por Giraldo, Mateus, & Ruiz (2009) demuestran que desde el punto de vista conceptual SPARQL presenta mejores prestaciones que otros lenguajes de consulta, además es el lenguaje recomendado por la World Wide Web Consortium - W3C[11]. A continuación se muestran los resultados del estudios en donde "+" significa que cumple la característica, "-" que no la cumple y "+-" que la cumple débilmente, además se tiene que EC = expresiones condicionales, CE= cuantificadores existenciales, OM = operadores matemáticos, CO = clases y objetos, RD = rango y dominio, RA = recursos adyacentes, PR = predicados sobre los recursos, DR = distancia entre recursos RF = reificación.

	EC	CE	OM	CO	RD	RA	PR	DR	RF
XQUERY	+	+	+	-	-	-	-	-	+-
XQL	+	+	+	-	-	-	-	-	+-
SPARQL	+	+-	+	+-	+-	+	+	+	+
RQL	+	+-	+	+-	+-	+	+	+	+
SeRQL	+	+-	+	+-	+-	+	+	+	+
OWL-QL	-	+-	-	+	+	+	+	+	+

Tabla 8. Comparación de lenguajes formales para consulta de ontologías.(Giraldo, Mateus, & Ruiz, 2009)

Dado estas razones, el lenguaje a utilizar para consultar ontologías será SPARQL.

[11]World Wide Web Consortium.http://www.w3.org/standards/techs/sparql#w3c_all (10/04/2011)

1.5 Procesamiento de Lenguaje Natural

El procesamiento del lenguaje natural – PLN-, “es un intento de simular el comportamiento lingüístico humano, de manera que el sistema de signos que constituye la lengua, sea adquirido y procesado por el computador, siendo éste capaz de reconocer, comprender, interpretar y generar lenguaje humano, ya sea escrito o hablado” (Gómez & Chamizo, 2008, p. 315) ,por lo tanto el PLN es una disciplina que surge para que la comunicación entre el hombre y la maquina resulte más fluida, para que sea la maquina la que se adapte al lenguaje del hombre y no al contrario, según Moreno (1999), de forma que los usuarios puedan llegar a comunicarse con el ordenador de la misma forma que lo harían con otro humano, esto se logra por medio de la formulación e investigación de mecanismos eficaces computacionalmente para dicho propósito.

En el mismo orden de ideas, Tomás (2010) describe el objetivo del PLN así: “su objetivo es el estudio de los problemas derivados de la generación y comprensión automática del lenguaje natural. Lo que busca esta disciplina es construir sistemas y mecanismos que permitan la comunicación entre personas y maquinas mediante lenguajes naturales” facilitando la búsqueda de información.

Es así como estos sistemas de PLN necesitan estudiar el lenguaje natural en profundidad y para esto se ayudan de cinco niveles de análisis lingüístico: “fonológico, morfológico, sintáctico, semántico y pragmático” (Sosa, n.d.)

1.5.1. Problemas del procesamiento del lenguaje natural

Los problemas básicos que debe afrontar el PLN debido a la libertad que permite el lenguaje humano son dos:

1. La variación lingüística: se refiere a la posibilidad de usar diferentes términos o expresiones para comunicar una misma idea
2. La ambigüedad lingüística, que tiene lugar cuando un término o expresión admite diferentes interpretaciones

A continuación se presentan ejemplos de estos problemas basado en la investigación de Tomás (2010):

- Palabras homófonas, palabras que se pronuncian igual, pero se escriben diferente, ejemplo: “has” – “as”, “hora” – “ora”. Afecta solo a programas que utilizan reconocimiento de voz.

- Palabras homógrafas, palabras que se escriben igual pero que tiene un significado diferente, ejemplo “casa” -> verbo “casar”, “casa”-> “edificio”.
- Palabras polisémicas, son palabras que pueden tener uno o varios significados, ejemplo gato (animal o instrumento mecánico).
- Sinonimia, varias palabras se refieren a un mismo concepto, ejemplo PC, computador, ordenador.
- Palabras que se refieren a un mismo concepto pero que escritas en plural o en singular no podrían ser identificadas como un mismo concepto.
- Palabras mal escritas o con errores de ortografía.
- Multipalabras, palabras que se refieren a una misma entidad, pero que se encuentran separadas, ejemplo. Grupo de investigación, Universidad del Cauca.
- Variación lingüística; las ilimitadas posibles combinaciones de palabras y estructuras para expresar lo mismo, ejemplo preguntas como: ¿Quién escribió la Ilíada? y ¿la Ilíada es una obra de qué autor?, que son sintácticamente muy distintas pero semánticamente iguales. Lo que conllevaría la construcción de complejos árboles sintácticos y de extensas reglas gramaticales para determinar si una consulta está bien formulada.

“Para hacer frente a todas estas dificultades el PLN afronta tareas que pese a su aparente simplicidad para los humanos, esconden una elevada complejidad para su resolución de forma automática desde una perspectiva computacional”(Tomás, 2010, p. 156)

1.5.2. Sistemas PLN

“El objetivo de los sistemas de PLN es permitir analizar lenguaje natural mediante la utilización de módulos o herramientas que en general se usan encadenadas. Dado un conjunto de módulos, los sistemas permiten que estos se ordenen para formar un flujo de componentes. Este conjunto de módulos encadenados, realiza el análisis de un texto.”(Techera, 2007).

1.5.3. OpenNLP[12]

La librería de Apache OpenNLP es una máquina de aprendizaje basada en un kit de herramientas para el procesamiento de texto del lenguaje natural. Es compatible con las tareas de NLP más comunes, como la tokenización, segmentación, la extracción de entidades nombradas, fragmentación, etc. Estas

[12]http://opennlp.apache.org/

tareas suelen ser necesarias para crear servicios más avanzados de procesamiento de texto. OpenNLP incluye también la máxima entropía y el aprendizaje basado en la máquina del perceptron.

> OpenNLP tiene por objeto proporcionar un paraguas para varios proyectos de código abierto de PNL para trabajar con mayor conciencia y (potencialmente) mayor interoperabilidad. Estos proyectos podrían ser muy diferentes y tienen poco que ver entre sí, pero la idea es tener una infraestructura básica para los investigadores y desarrolladores para colaborar en proyectos de código abierto PNL. De esta manera, OpenNLP puede ser pensado como una organización similar a GNU.(OpenNLP, 2004)
>
> El objetivo de OpenNLP es crear un kit de herramientas para las tareas anteriormente mencionadas. Otro objetivo es pre-construir modelos para una variedad de lenguajes.

1.5.4. Recuperación de la información

La recuperación de la información es “una disciplina que involucra la localización de una determinada información dentro de un almacén de información o base de datos” (Meadow et al., 2007). Un sistema de recuperación de la información es aquel que “dada una colección de documentos y una consulta formulada por un usuario en un cierto momento, proporciona el subconjunto de documentos que es más relevante para la consulta del usuario”(Baeza-Yates, 1999). Identificando el siguiente proceso:

- El usuario introduce una consulta en el sistema. Esta consulta representa sus necesidades de información.
- El sistema procesa dicha consulta. Se buscan documentos que, de alguna forma, sean coincidentes con los términos que aparecen en dicha consulta.
- El sistema muestra los documentos que son coincidentes con la consulta, ordenándolos de mayor a menor relevancia.

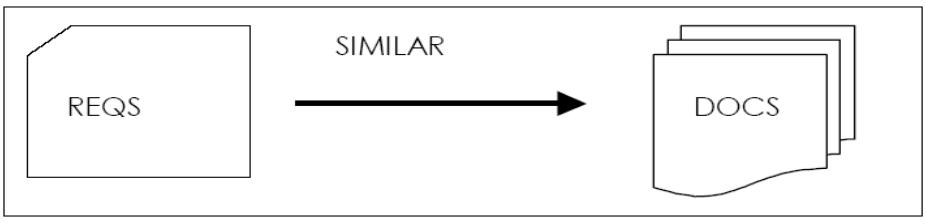


Figura 4. Esquema simple de un SRI.(Salton, 1983)

1.5.5. Búsqueda de respuestas

"La búsqueda de respuestas es una tarea automática que tiene como finalidad encontrar respuestas concretas a necesidades de información precisas formuladas por un usuario" (Vicedo, 2004).

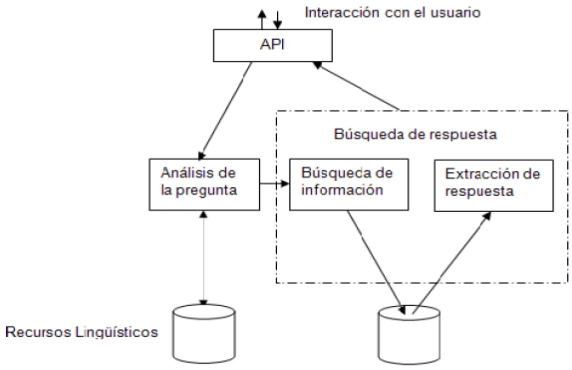
Un sistema de búsqueda de respuestas es "un tipo especial de sistemas de recuperación de la información en los que el sistema no devuelve un grupo de documentos sino una respuesta concreta, evitando al usuario analizar una serie de documentos para encontrar la respuesta."(Ruiz &Belalcázar, 2012).

1.5.6. Arquitectura de un sistema de búsqueda de respuesta

Estudios como el de Tomás (2010) demuestran que existen una arquitectura común dada en tres fases y se abordan secuencialmente:

- Análisis de la pregunta
- Búsqueda de información
- Extracción de la respuesta

Figura 5. Arquitectura de un sistema de búsqueda de respuestas.(Celaá, 2010)



i. Análisis de la pregunta: Recibe una consulta en lenguaje natural y por medio de herramientas LN la consulta se procesa y se transforma en consulta que la máquina comprende.

ii. Los sistemas de BR "deben disponer de un sistema de Recuperación de Información, que devuelva una colección de documentos, enlaces,

propiedades (…) que contengan la información solicitada. Esta información puede estar almacenada de distintas maneras, pero el sistema de Recuperación de Información debe seleccionar la importante y relevante que esté relacionada con la información solicitada, y la salida del análisis de la pregunta".(Celaá, 2010)

iii. Extracción de la información: En esta fase se encarga de encontrar la respuesta más adecuada a partir de un grupo de documentos relevantes seleccionados en la recuperación de la información y mostrar una respuesta al usuario.

1.5.7. Interfaces en lenguaje natural sobre ontologías

Éstas permiten la comunicación entre humanos y máquinas. La web semántica tiene como objetivo crear una intuitiva interfaz general de lenguaje natural

"Las interfaces en lenguaje Natural sobre ontologías surgen de la necesidad de hacer que los contenidos de la Web Semántica sean más accesibles a los usuarios finales, debido a que la cantidad de información almacenada en ontologías aumenta constantemente"(Ruiz &Belalcázar, 2012).

Estas interfaces proporcionan una herramienta de consulta de acceso a los usuarios finales sin tener que aprender lenguajes como RDF, OWL, SPARQL, etc, ya que las interfaces de lenguaje natural "permiten ocultar la formalidad de las ontologías y lenguajes de consulta a los usuarios finales, ofreciéndoles una manera familiar e intuitiva para la formulación de consultas" (Schwitter&Tilbrook, 2006).

La investigación de Ruiz &Belalcázar (2012) muestran los principales aportes de las interfaces de lenguaje natural encontradas, tanto las que son dependientes del dominio en un entorno particular y las que son independientes de un dominio específico. La siguiente tabla es extraída de su estudio:

Propuesta	**Principal aporte a esta arquitectura**	**Soporte consultas en idioma español**	**Desventaja para el propósito de esta arquitectura.**
Iuriservice: Un FAQ Inteligente para los Jueces en su Primer Destino	Conceptualización	SI	No especifica procedimientos.
En el proyecto	Estudio de	SI	No especifica

Propuesta	Principal aporte a esta arquitectura	Soporte consultas en idioma español	Desventaja para el propósito de esta arquitectura.
Análisis y diseño de un agente semántico basado en ontologías para el dominio de la salud	herramientas para gestión de ontologías.		procedimientos
Natural language querying for video databases	Conceptualización.	NO	Realiza consultas sobre bases de datos y no sobre ontologías.
Addressing ontology-based question answering with collections of user queries	Enfoque de utilización de herramientas de PLN ya existentes para el análisis de la pregunta en LN.	SI	Enfoque estático, solo funciona para la ontología a la que se le hace el estudio, no soporta cambios en el dominio
AquaLog: Anontology-driven question answering system fororganizational semantic intranets	Demuestra la eficiencia de la utilización de herramientas de PLN existentes, utiliza el enfoque de mapeo a términos ontológicos y paso a tripletas RDF.	NO	No utiliza SPARQL como lenguaje formal, necesita demasiados diálogos de clarificación con el usuario para resolver problemas de ambigüedad.
Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Look up through the User Interaction	Utiliza ontología para procesamiento semántico.	NO	Utilización de herramientas y heurísticas que solo sirven para el idioma inglés, método de compleja adaptación al idioma español. Utiliza arboles sintácticos para el paso de la consulta de usuario a SPARQL demasiado compleja.
Semantic Processing of Natural Language Queries in the Onto NL Framework	Utiliza ontología para procesamiento semántico de la consulta	NO	Propone reutilización de tecnologías, paso a SPARQL dependiente de la ontología
Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs	Permite consultas en LN completo, Independiente del dominio	NO	No muestra como podría hacer dinámicamente el paso a SPARQL. No muestra detalles de los procedimientos.
PANTO: A Portable Natural Language Interface toOntologies	Se ayuda de un lexicón builder para generar sinónimos propios del dominio al extraer las entidades de la ontología que se	NO	Complejidad de la implementación del árbol sintáctico necesario para la obtención de tripletas RDF y posterior paso SPARQL. La implementación del árbol sintáctico necesita

Propuesta	Principal aporte a esta arquitectura	Soporte consultas en idioma español	Desventaja para el propósito de esta arquitectura.
	carga, con lo cual lo hace una interfaz dinámica independiente del dominio.		del Stanford Parcer , el cual solo sirve para consultas en el idioma inglés y su adaptación al idioma español sería demasiado compleja.
Queryng Ontology using Keywords and Quantitative Restriction Phrases	Enfoque que permite el paso dinámico de términos compatibles con la ontología a lenguaje SPARQL.	NO	No específica herramientas de PLN para categorización morfológico y solo permite consultas en idioma inglés.
NLP-Reduce	Independencia del dominio, permite consultar al usuario por medio de palabras clave, fragmentos de sentencia o sentencia completas en inglés – se ayuda de la reducción de la consulta a palabras clave por medio de la eliminación de stop words.	NO	No especifica procedimientos, solo describe de forma general su funcionamiento.

Tabla 9. Principales aportes ILN encontradas.(Ruiz &Belalcázar, 2012, p. 37).

A continuación se muestran las herramientas tecnológicas para abordar el problema enmarcado en el estado del arte(Ver sección 2. Estado del arte). Es decir que esta propuesta maneja los problemas de escalabilidad de datos, computacional y de usuarios. (Ver figura 2).

1.6 Segundo Modelo Informático propuesto: Redes neurales – Máquinas de aprendizaje-

1.6.1.1 OpenNLP

La librería de Apache OpenNLP es una máquina de aprendizaje basada en un kit de herramientas para el procesamiento de texto del lenguaje natural. Es compatible con las tareas de NLP más comunes, como la tokenización, segmentación, la extracción de entidades nombradas, fragmentación, etc. Estas tareas suelen ser necesarias para crear servicios más avanzados de

procesamiento de texto. OpenNLP incluye también la máxima entropía y el aprendizaje basado en la maquina del perceptron.

1.6.1.1.1 Maxent

> Los modelos de máxima entropía son un marco para la integración de información de muchas fuentes heterogéneas con el fin de clasificarla. Los datos para un problema de clasificación son descritos como un número (potencialmente grande) de características. Estas características pueden ser bastante complejas y permiten que el experimentador haga uso del conocimiento previo acerca de qué tipo de información se espera que sea importante para la clasificación. Cada característica corresponde a una restricción en el modelo. (…) Si optamos por un modelo con menor entropía, añadiríamos,'información',restricciones al modelo que no están justificadas por la evidencia empírica disponible para nosotros. La elección del modelo de máxima entropía está motivada por el deseo de conservar la incertidumbre tanto como sea posible.
>
> El objetivo de OpenNLP es crear un kit de herramientas para las tareas anteriormente mencionadas. Otro objetivo es preconstruir modelos para una variedad de lenguajes.
>
> (Maxent, 2008)[13]

Maxent es utilizado para crear programas que realizan tareas difíciles de clasificación. "Las cifras de precision y recall de programas que utilizan los modelos Maxent han alcanzado, o son, el estado del arte en tareas como speechtagging (etiquetado del habla), sentencedetection (detector de oraciones) y namedentityrecognition (reconocimiento de entidades nombradas) "(Maxent, 2008).

Las características en maxent son funciones booleanas de resultados (outcomes)–clases- y contextos (context).

Ejemplo de una característica útil:

```
feature (outcome, context)  = { 1   if  outcome=DETERMINER
                              {         &&  currentword(context) = "that"
                                              { 0   otherwise
```

El trabajo para la persona que crea el modelo de una tarea de clasificación, es seleccionar las características que serán de utilidad en la toma de decisiones. La

[13] http://maxent.sourceforge.net/about.html

representación teórica de las características es una y la representación en la implementación es otra. “no se necesita saber el aspecto teórico para empezar a seleccionar las funciones con openNLP.maxent”. (Maxent, 2008).[14]

Básicamente las funciones se reducen, para sus propósitos, en predicado contextual de la función, es decir, para el ejemplo anterior el predicado contextual es: *currentword(contexto)= “that”*, denominado *current=that.* Con el fin de aclarar,cada vez que se utilice la palabra característica en realidad se está hablando de un predicado contextual que se extenderá a varias características.
Por ejemplo, en la siguiente oración se debe reconocer que Terrance es una entidad pero no se sabe si es o no un nombre.

He succeeds Terrance D. Daniels, formely a W.R.[15]

Las características creadas son:

Previous=succeeds; current=Terrance; next=D.

Esta información se traduce en la implementación.

Para la implementación suponemosque:

- Existe un modelo pre-entrenado para encontrar nombres (es-ner-per.bin).
- La creación de una instancia de interfaz MaxentModel con el modelo.
- El resultado apunta a una entidad reconocida (Terrance).

Para considerar si la entidad Terrance es o no un nombre, se le envía un String [] con todas las características, como las del ejemplo, por medio del método:

```
public double[] eval(String[] context);
```

El double[] que devuelve contiene las probabilidades de los distintos resultados que ha asignado el modelo basado en las características que enviamos. Por ejemplo los resultados asociados con las probabilidades pueden ser True para el índice 0 y False para el índice 1.

Para encontrar el nombre del String de un resultado particular de un índice, es llamado por el método:

[14] http://maxent.sourceforge.net/howto.html
[15] Ejemplotomado de MaxentHowTo.

```
publicStringgetOutcome(inti);
```

Método para la obtención del nombre del String del resultado mas probable:

```
public String getBestOutcome(double[] outcomes);
```

Con este último método obtenemos el nombre del String del resultado más probable.

1.6.1.1.2 Entrenando un modelo

Con el fin de entrenar un modelo es necesario producir de alguna manera un conjunto de eventos que sirven como ejemplos para nuestro modelo. Esto normalmente se realiza mediante el uso de los datos que han sido anotados por alguien con los resultados que el modelo está tratando de predecir. Esto se hace con un objeto EventStream. Un flujo de eventos –EventStream- es solo un iterador sobre un conjunto de eventos. El evento consta de un resultado y un contexto. Siguiendo el ejemplo, un evento luce:

> (resultado) outcome: T
> (contexto) context: previous=succeeds current=Terrence next=D. currentWordIsCapitalize

Una vez tenga un flujo de eventos –EventStream- y los datos de entrenamiento a la mano, es posible entrenar un modelo. OpenNLP.maxent tiene una implementación iterativa de escala generalizada (OpenNLP.maxent.GIS) utilizada para este propósito. Para llamar el métodoGIS.trainModel escriba un código en algún lugar.

```
public static MaxentModeltrainModel(DataIndexer di, int iterations) {  ...  }
```

Intiterations: Numero de veces que el proceso de entrenamiento debe iterar al encontrar los parámetros del modelo.

DataIndexer di: Objeto abstracto. Este extrae todos los eventos que ha recogido el EventStream. Luego se manipula en un formato más eficiente para trabajar el procedimiento de entrenamiento.
Para crear una instancia de DataIndexer es:

```
publicOnePassDataIndexer(EventStreames, intcutoff){ ... }
```

Una vez el modelo retorna se puede escribir en el disco usando el código siguiente:

```
File outputFile = new File(modelFileName+".bin.gz");
GISModelWriter writer = new SuffixSensiiveGISModelWriter(model, outputFile);
writer.persist();
```

Esto guarda nuestro modelo en un formato binario comprimido, usando la clase *BinaryGISModelWriter* basada en la extensión del archivo.

Así mismo es posible cargar el modelo en el disco:

```
GISModel m = new SuffixSensitiveGISModelReader(new
File(modelFileName)).getModel();
```

La tabla estructura general, muestra los paquetes (interfaces), clases abstractas y clases usadas para crear un modelo pre-entrenado de máxima entropía –maxent-.

Packages	ClassAbstract	Class
OpenNLP.model		*Context*
		DataIndexer
		MaxentModel
		OnePassDataIndexer
		EventStream
OpenNLP.maxent		*GIS*
		GISModel
		GISTrainer
OpenNLP.maxent.io	GISModelWriter	*SuffixSensitiveGISModelWriter*
		BinaryGISModelWriter
	GISModelReader	*SuffixSensitiveGISModelReader*
		BinaryGISModelReader

Tabla 10. Maxent. Estructura general.

Las clases se describen en el anexo 3.

1.6.1.2 Funciones comunes de PLN

Detección de oraciones

Este detector de OpenNLP puede detectar un carácter de puntuación marcado o no al final de la sentencia. Usualmente la detección de sentencias se hace antes de la tokenización de texto. Este detector no identifica límites basados en los contenidos de las sentencias. Por ejemplo un titulo de un artículo.(SentenceDetectorMEsentenceDetector = new SentenceDetectorME(model).

El detector de sentencias puede ser fácil de integrar dentro de una aplicación a través de su API. Para instanciar el modelo de detector de sentencias primero debe cargarse. Después de cargarse se instancia a través de SentenceDetectorME.(bin/opennlpSentenceDetectorTrainer)
Componente: Sentence Detector

Tokenization

El tokenizador de OpenNLP segmenta una entrada de caracteres en tokens. Los tokens son usualmente palabras, puntuación, números, etc.
OpenNLP ofrece tres implementaciones para tokenizacion:

- Tokenizacion de espacios en blanco: Un espacio en blanco setokeniza, una secuencia de espacios en blanco no se identifican como tokens.
- Tokenizacion simple. Caracteres
- Tokenizacion de aprendizaje: Tokenizador con máxima entropía, detecta limites de tokens basados en la probabilidad del modelo.

Es importante asegurarse de que su tokenizacion produce tokens del tipo esperado por los componentes posteriores al procesamiento de texto.

Herramientas de tokenizacion: Muchas personas utilizan el código Java mediante la creación de objetos SentenceDetector y Tokenizer e invocan sus métodos como corresponde.
La tokenizacion se puede integrar en una aplicación mediante el API definido. La instancia compartida por el TokenizadorWhiteSpace puede ser recuperada desde un campo estaticoTokenizer.INSTANCE. Para crear una instancia del TokenizerME primero se debe crear y cargar un modelo de token.
El tokenizador ofrece dos métodos de tokenizacion, ambos esperan un objeto string de entrada el cual contiene el texto sin tokenizar. El primer método retorna un arreglo de strings, donde cada string es un token. El segundo método es

TokenizePos el cual devuelve un arreglo de tramos, cada tramo contiene las compensaciones del carácter inicial y final del token en el string de entrada.
Componente: Tokenizer.

NamedEntityRecognition

El buscador de nombres (NameFinder) puede detectar entidades con nombre y números en el texto. Para que pueda detectar los nombres de las entidades (NamedEntityRecognition) es necesario un modelo. El modelo es dependiente del tipo de idioma y tipo de entidad para que fue entrenado. OpenNLP ofrece modelos pre-entrenados de buscador de nombres –NameFinder-. Para encontrar nombres en el texto, este debe dividirse en tokens y oraciones. Es importante que la tokenizacion de los datos de entrenamiento y la entrada de texto sea idéntica.

Los datos deben ser convertidos al formato OpenNLP de namefinder. El cual es una oracion por línea. La sentencia debe ser tokenizada y contenida

Componente: NameFinder.

Chunker

La fragmentación del texto consiste en dividir un texto en partes sintácticamente correlacionadas de palabras, como grupos de nombres, grupos de verbos, pero no especifica su estructura interna.

2. Estado del arte

Desde los años 70 y comienzos del nuevo milenio, el modelo dominante de bases de datos era el modelo relacional. “A principios del nuevo milenio los desarrolladores comienzan a darse cuenta que sus datos no se ajustan a el modelo relacional y algunos de ellos comienzan a desarrollar otras arquitecturas para el almacenamiento en bases de datos”(Björklund, 2011, p. 3). En la actualidad el problema no es saber que proveedor de bases de datos relacional se va a elegir, sino, “decidir cual arquitectura de almacenamiento de datos es la mas adecuada para los datos” (Björklund, 2011, p. 1).

No es práctico manejar este problema dejando al margen las inconsistencias en la base de datos, entonces, es necesario que el sistema tolere la falta de consistencia. El “sistema debe ser capaz de identificar fuentes de inconsistencias y tratar con declaraciones contradictorias de tal manera que se pueda seguir produciendo derivaciones confiables”(Seth&Lytras, 2007, p. 8).

El problema de aceptar o no contradicciones pertenece al tipo de lógica que usemos. La lógica clásica decide entre las contradicciones, evalúa dos declaraciones y escoge la que parezca correcta; “el desarrollo de la lógica paraconsistente se inició con el fin de desafiar el principio lógico de explosión, el cual dice que de una contradicción se sigue cualquier cosa, ex contradictioneQuodlibet (ECQ). La lógica clásica, la lógica intuicionista, y la mayoría de las lógicas estándar son explosivas. Una lógica se dice que es paraconsistente si y sólo si su relación de consecuencia lógica no es explosiva” (Priest, Graham and Tanaka, Koji, 2001, párr. 1). La lógica clásica es explosiva, y esto es posible en dominios axiomáticos donde se han explorado completamente o todas sus declaraciones son verdaderas por definición, es decir es una lógica trivial, que es tan precisa que no es posible derivar conclusiones matizadas, sino conclusiones totalmente falsas o totalmente verdaderas.

Einstein: Dios no juega a los dados
Bohr: Einstein, deje de decirle a Dios lo que tiene que hacer(Wikiquote, n.d.)

El razonamiento monótono es aquel que desde el momento que se asume un conocimiento como cierto siempre permanece así. Debido a esto se dice que la lógica de primer orden puede ser completa y consistente. Si de una base de conocimiento, KDB, se deduce una expresión *X* y se tiene otra base de conocimiento (KDB’), siendo (KDB)⊂(KDB^'), entonces también se puede deducir *X* de KDB’. Debido a esto, la contención de datos o el nuevo conocimiento en cualquier base de datos incrementa el volumen de datos y se requiere de más

espacio. “La revolución informática está sobre nosotros con dimensiones grandes y con presentaciones de altos volumen de datos, y ante el incremento de este aspecto debemos estar preparados(…)Nos debemos concentrar no en el número de datos sino en los datos”(Graham, 2011, p. 1) y para ello ya debe estar superado el problema de almacenamiento de grandes volúmenes de datos. Cuando existe conocimiento incompleto se llega a modelos no monótonos, por ejemplo, cuando sucede un nuevo hecho que produce un cambio respecto a nuestras creencias personales hasta ese momento. Este reto, procesar y gestionar grandes volúmenes de datos, surge en la ciencia como x-informática (donde x es una ciencia escogida, como la bioinformática, geo-informática, astro-informática, etc) con el fin de sustentar los descubrimientos de este siglo.

La lógica paraconsistente es una lógica no trivial, “si en la Lógica Clásica se acepta el que se puede derivar una contradicción, en ella se puede "demostrar cualquier cosa" y en este sentido se dice que es trivial. Lo que interesaría es construir Lógicas no triviales, es decir lógicas en que se aceptase la contradicción pero que no se pudiese demostrar cualquier cosa. A este tipo de Lógicas se les denomina Lógicas paraconsistentes.” (Ramírez, n.d., p.1), la cual trata las contradicciones en forma atenuada. Una razón contundente para demostrar que existen teorías incompatibles y se debe usar la lógica paraconsistente se muestra en la ciencia. Cuando admitimos la existencia de una teoría es evidente que admitimos tales teorías y su lógica subyacente debe ser paraconsistente. “Considere la teoría de Bohr sobre el átomo. De acuerdo con esto, un electrón gira alrededor del núcleo del átomo sin energía radiante. Sin embargo, de acuerdo con las ecuaciones de Maxwell, que forman parte de la teoría, un electrón que se está acelerando en órbita debe irradiar energía. Por lo tanto la afirmación de Bohr sobre el comportamiento del átomo era inconsistente. Sin embargo, evidentemente, no todo lo concerniente al comportamiento de los electrones se deduce de ella. Por lo tanto, cualquiera que sea el mecanismo de inferencia lo que subyace es que esto debe haber sido paraconsistente” (Priest&Tanaka, 2009, párr.11).

La lógica de primer orden asume que el conocimiento es exacto, completo y consistente. “El límite de la representación del conocimiento es el de tener la habilidad que tiene el razonamiento humano para hacer imprecisiones, construir verdades parciales, aproximaciones, procesar sin certezas o con inconsistencias”(Seth&Lytras, 2007, p.8). La lógica difusa o borrosa (Fuzzy) es un enfoque para el razonamiento impreciso el cual permite difuminar las fronteras impuestas artificialmente por diferentes conjuntos permitiendo un grado de incertidumbre en las formas en que vemos el mundo y de las entidades que interactúan entre sí. “Tal vez la forma más sencilla de generar una lógica paraconsistente (…) es el uso de una lógica multivaluada. En la lógica clásica hay exactamente dos valores de verdad. El enfoque multiforme es dejar de lado esta premisa clásica y permitir más de dos valores de verdad. La estrategia más simple

es el uso de tres valores de verdad: (V) verdadero (solamente), (F) falso (solamente) y (B) ambos (verdadero y falso) para las evaluaciones de las fórmulas."(Priest&Tanaka, 2009, párr.9.)

Los sistemas de lógica multivaluada MVL siguen 2 patrones estándar: la lógica proposicional, compuesta de variables proposicionales y conectivos, esta lógica es capaz de formar otras proposiciones de mayor complejidad; la lógica de predicados (Lógica de Primer Orden) la cual tiene el poder expresivo suficiente para definir prácticamente todas las matemáticas. En la lógica de primer orden existen variables de objeto, junto con símbolos de predicados y "posiblemente constantes de objetos y símbolos de función, cuantificadores y conectores. Estos lenguajes son la base desde el punto de vista semántico y sintáctico para los sistemas fundados en la lógica" (Gottwald&Siegfried, 2000, párr.3).

2.1 Redes neuronales

La inspiración de la computación en los sistemas biológicos cubre áreas importantes como la inteligencia artificial en el campo de las redes neuronales. En la actualidad se necesita enfrentar conjuntos cada vez más grandes de datos en cualquier área, desde la biología hasta el manejo de cualquier tipo de información. "El conocimiento sobre los sistemas biológicos en un alto nivel desarrollaron los modelos clásicos de computación sobre redes neuronales; ahora el conocimiento sobre los sistemas biológicos en bajo nivel como el funcionamiento de sistemas de formas moleculares son las que tienen un gran impacto en redes neuronales"(Kroeker, 2011, p. 12).

La investigación del profesor Ziv Bar-Joseph[16] se acerca a resolver una de las diferencias entre el cerebro y la computación. El cerebro tolera errores. ¿Cómo se debe diseñar el almacenamiento de datos para permitir los errores? La mayoría de los actuales sistemas de bases de datos esperan que todo sea perfecto, por el contrario, nuestro cerebro abarca el error. El cerebro es una batalla de patrones de competencia, del que surge la comprensión. Se trata de un "sistema construido sobre el ruido, que espera el error y la recuperación del error. El cerebro abandona la precisión y se nutre de la aproximación"(Hong & Linden, 2012, p.11). De las lecciones que se deben tomar de los sistemas biológicos es que el cerebro es lleno de charcos. Y eso no esta mal. La precisión no es necesaria en todo o la mayoría de casos. Los fracasos se manejan mejor esperándolos y no tratándolos como excepciones. Debemos esperar errores y manejarlos de forma rutinaria.

[16]Profesor en el departamento de Maquinas de aprendizaje y Centro de Biología computacional de la Escuela de ComputerScience de Carnegie MellonUniversity.

“Debemos dejar de preocuparnos y aprender a amar el ruido”(Hong & Linden, 2012, p.11).

Bar-Joseph reconoce el problema de los Big Data estudiando la secuencia del genoma humano en el MIT. “La explosión de datos provenientes de la biología creaba la necesidad de desarrollar nuevos métodos computacionales para dar sentido a esta cantidad de Big Data” (Kroeker, 2011, p. 11).

Algoritmo FlyFruit para la gestión de redes distribuidas:

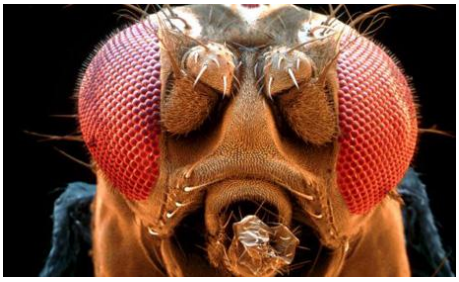


Figura 6. FlyFruit.
Fuente:http://static.guim.co.uk/sys-images/Environment/Pix/pictures/2008/07/16/FruitFly460.jpg

El problema de la gestión de redes distribuidas por medio del algoritmo de la mosca de fruta (FlyFruit) nace de la observación, de Bar-Joseph, sobre una investigación de un estudiante de biología acerca de la determinación del destino celular. Este método es simple y más robusto en comparación de los existentes debido a que asignan células líderes para hacer conexiones directas con otras células, llamado autoselección. La relación la establece debido a que las células de mosca de fruta utiliza diminutos pelos de detección para comunicarse llamados SOP (sensoryorgan precursor). Tales SOP se pueden tomar como MIS (maximalindependent set) que se utiliza para comunicarse con otros nodos de la red. Los MIS son un conjunto de nodos lideres usados por los métodos de redes neuronales para la creación de un marco distribuido o marco de red. Siguiendo el patrón de desarrollo de la mosca de fruta, el algoritmo varía la probabilidad con el tiempo, comenzando con un nivel bajo y aumentando la probabilidad de autoselección después. Muchas investigaciones son enfocadas a elegir el mejor MIS en las redes distribuidas y el comportamiento de los SOP resulta efectivo debido a que tienen poca información y envían señales químicas que inhiben las células vecinas para que no se conviertan en SOP (MIS). El proceso termina cuando las células son SOP o vecinos de SOP, es decir son MIS o vecinos de MIS, pero no puede existir una relación de un vecino de MIS con otro vecino de MIS. La investigación del algoritmo de la mosca de fruta de Bar-Joseph acepta que el proceso de selección de MIS es mas lento que los otros métodos convencionales pero se recompensa debido a la eficiencia de operar con pocos supuestos acerca de la ubicación de un nodo. Bar-Joseph es un científico que señala que las ideas basadas en la biología ayudaran a mejorar los marcos de redes para que operen con mayor eficacia donde uno de los más grandes retos es la tolerancia a fallos.

Continúa creciendo la responsabilidad y la complejidad en el manejo de los Big Data. Las bases de datos clásicas emanan respuestas nítidas y precisas dado a que la cantidad de datos a procesar es relativamente baja. La dinámica y caótica estructura que adquieren las bases de datos actuales de Big Data esta dejando cada vez más sin oportunidad a que se procesen los datos de manera relacional y estructurada por medio de SQL. Los sistemas tienen altas tasas de cambio entre miles de computadoras para retener y procesar información.

Procesos y patrones. Clasificación y reconocimiento

Cuando la corteza parietal es afectada por una cirugía en el cerebro, puede influir en la fuerza de las convicciones religiosas de los pacientes después del procedimiento quirúrgico, menciona la investigación de Fish(2012). Esto hace "suponer la existencia de neuronas especializadas para tareas de procesamiento y por lo tanto el desarrollo de RNA especializadas"(Fish, 2012, p.1).

Las redes neurales artificiales se aproximan al comportamiento de los sistemas biológicos reales, pero su tamaño y complejidad hacen aumentar significativamente su simulación en tiempo real, siendo necesario los recursos de supercomputación. Si vemos que hay redes neuronales con fines altamente especializados, la pregunta que surge es ¿cómo estas redes neurales discretas pueden producir algo más que la suma de sus partes?

Admitir inconsistencias en los sistemas de bases de datos actuales, como una alternativa de eficiente de manejar los problemas de grandes cantidades de datos, es la mejor alternativa. Desde el área de la neurociencia cognitiva, la investigación realizada por Nosofsky, Little, & James (2012), acerca de los principales procesos cognitivos, muestra las áreas en el cerebro, por medio de la imagen por resonancia magnética funcional (fMRI), que se activan cuando este realiza procesos de decisión de categorización y reconocimiento.

Los procesos cognitivos fundamentales dice Nosofsky et al. (2012) son la categorización y el reconocimiento de objetos. Estos tienen diferentes objetivos. La categorización consiste en que los observadores toman decisiones acerca de si diferentes objetos pertenecen a la misma clase. En el reconocimiento se pide a los observadores determinar si cada objeto de prueba es una coincidencia exacta del objeto de estudio. Los estudios en la neurociencia cognitiva parecen ser divergentes debido a que investigaciones muestran que el cerebro realiza estos dos tipos de procesos por medio de un mismo sistema cerebral el cual sirve tanto para categorizar como para reconocer. "La gente almacena ejemplares

individuales en la memoria, y toman decisiones de categorización y reconocimiento sobre la base de la similitud de los objetos de prueba respecto a los ejemplares." (Nosofsky et al., 2012, p.333).Aunque el modelo de un solo sistema cerebral haya proporcionado excelentes resultados cuantitativos sobre categorización y reconocimiento, existen estudios los cuales difieren. Por ejemplo, se demuestra en una investigación que personas amnésicas con baja capacidad de reconocimiento en su memoria muestran un rendimiento normal de categorización, lo que sugiere que diferentes sistemas de memoria neurales están en capacidad de realizar la tarea. Entonces los sistemas cerebrales están por separado y completamente aislados uno del otro? O existen varios sistemas pero comparten algunos sectores neuronales?

Basado en estudios de imagen por resonancia magnética funcional (fMRI), la investigación de Nosofsky et al. (2012), afronta este problema seducido por la hipótesis que compartimos sistemas de memoria para realizar la categorización y el reconocimiento de objetos. Pero estos sistemas no son completamente independientes y excluyentes, es decir que algunos de los sectores que se activan cuando se esta categorizando también se pueden activar cuando se esta en el proceso de reconocimiento, y viceversa. Esta divergencia de resultados lleva a plantear si las diferencias presentadas en las imágenes cerebrales son reflejo de las diferencias de los estímulos y no una diferencia entre categorización y reconocimiento. Los resultados de Nosofsky et al. (2012), arrojaron que la categorización perceptiva y el reconocimiento de elementos antiguos y nuevos son en gran parte mediada por la superposición de los sistemas neurales de la memoria. De esta forma demuestra que la existencia de sistemas diferentes de neuronas no significa que no se compartan las neuronas para categorizar y para reconocer, es decir, "la evidencia pasada sobre categorización y reconocimiento que están mediadas por sistemas separados pudo haber sido un reflejo de las condiciones de cambios de estímulo, o de cambios adaptativos en los ajustes de los parámetros o de ambas cosas". (Nosofsky et al, 2012, p. 337).

Los resultados de la investigación de Nosofsky et al. (2012). *De los comportamientos de los datos*: una versión básica de un modelo ejemplar proporciona una buena descripción cuantitativa de las probabilidades de elección de todas las condiciones de categorización y reconocimiento, simplemente permitiendo cambios sistemáticos en ajustes de parámetros; *del análisis del modelo formal:* el análisis de los datos promediados indican que las diferencias en los patrones de actividad cerebral a través de las tareas de categorización y reconocimiento fueron interpretables en términos de cambios en la acumulación de pruebas derivadas de los cambios en la configuración de criterios; análisis del

cerebro por imágenes*: del análisis de modelos realizados a nivel de sujeto individual por fMRI*: revelaron correlaciones significativas entre la configuración de criterios y estos patrones de actividad cerebral. Esto indica que las propuestas teóricas de la categorización perceptiva y el reconocimiento se rigen por representaciones y procesos similares, con cambios sistemáticos en la configuración de los parámetros que se producen a través de las tareas.

Podemos decir que no son sistemas de memoria separados los que actúan para cada tipo de proceso (categorización y reconocimiento) o un mismo sistema, sino es por la superposición de sistemas neurales.

Esta investigación marca un comportamiento interesante no solamente para el área de la neurociencia cognitiva. Observando los resultados de fMRI durante la investigación, se notaron patrones en las imágenes en el momento de hacer categorización de objetos sin tener al alcance mucha información para poder tomar la decisión, es decir que decide bajo condiciones que no conoce o son inciertas en el momento. Esto se reflejó en el momento de contrastar dos clases de categorización, una prueba que presentaba facilidad en el momento de identificar los elementos que hacen meritorio la clasificación. El segundo presentaba de manera borrosa los elementos para realizar la clasificación, y aun así realizaba correctamente la clasificación. Las imágenes que resultaron en el caso de presentar objetos con elementos difíciles de identificar para hacer su categorización mostraron que en el cerebro se activaba otro sector que no se activa realizando una categorización en condiciones normales. Esto fue interpretado por los investigadores como la capacidad de tomar decisiones y realizar predicciones bajo inconsistencias con la información disponible. En comparación con el proceso de reconocimiento es mas fácil determinar si existe o no una coincidencia exacta para tomar o no la decisión; la categorización sin el uso de retro alimentación correctiva hace que la asignación del error sea arbitraria. Sin embargo los resultados demostraron conclusiones positivas de categorización sin poca información del objeto, reconociendo que el cerebro parece tener una capacidad para predecir el error y tomar decisiones correctas sin tener el 100% de seguridad, por lo tanto se manifiesta nuevamente la necesidad de admitir inconsistencias en los sistemas de bases de datos.

Las diferentes formas de abordar los problemas complejos para el procesamiento de Big Data en el área de la inteligencia artificial se deben a los distintos objetivos que tienen que cumplir los sistemas expertos y máquinas de aprendizaje. A

continuación veremos como se sigue difuminando la ilusión de trabajar en un universo finito pequeño de datos sin tolerar fallos e inconsistencias con el fin de sacar conclusiones y predicciones, con grandes cantidades de datos.

Las investigaciones de Fish(2012) y de Angadi&Venkatesulu (2012), nos muestran dos formas actuales de enfrentar estos problemas. Las dos investigaciones enseñan como hacer clasificaciones y reconocimiento de patrones sobre conjuntos grandes de datos provenientes de cantidades de fuentes diferentes, es decir que el problema continúa siendo la escalabilidad de datos, la escalabilidad computacional y la escalabilidad de usuarios.

El sistema HiveMind ofrece un marco para la selección apropiada de redes neurales especializadas para estructurar las funciones de una o varias RNA en su propio contexto y determinar un curso de acción razonable en ese contexto. HiveMind está diseñado para "ser un marco para la negociación semántica y las relaciones lógicas entre los diversos conceptos y una herramienta de gestión para detectar y analizar conceptos"(Fish, 2012, p.2). Además una característica importante de este sistema es la capacidad para procesar grandes cantidades de datos, Big Data, para ello combina y prueba enfoques nuevos y existentes de la IA para dar sentido a los datos sin estructura, recibidos de su entorno.
La escalabilidad de clientes de HiveMind es soportada por la arquitectura SOA[17], usando XML para definir el lenguaje de la interfaz con la cual se va a comunicar e intercambia los datos XML entre dos objetos con SOAP [18] y llama a los procedimientos remotamente a través de RPC[19].

Mediante ontologías HiveMind representa conceptos los cuales permiten conceptualizaciones de más alto nivel y procesos de toma de decisiones basados en puntos de datos relevantes apoyados por el framework. Diferentes servicios ofrece HiveMind; por medio de la estadística justifica sus decisiones; la lógica de asignación y recuperación de componentes y almacenamiento de relaciones se lleva a cabo por el servicio web HiveMind.WS; a través de HiveMind.DB se almacenan los componentes estándar de procedimientos.

-HiveMind.WS | HiveMind.DB-

Como objetivo, HiveMind.WS recupera de manera eficiente los datos y los organiza según lo especifique el esquema HiveMind.DB y así dar un framework a

[17] ServiceOrientedArchitecture. Permite la creación de sistemas de información altamente escalables que reflejan el negocio de la organización

[18] Simple Object Access Protocol. Protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

[19] RemoteProcedureCall. Protocolo que permite a un programa de ordenador ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos.

los usuarios para manipular los recursos de las redes neurales relevantes y sus formalismos lógicos.

"Podemos enseñar a una máquina que aprenda áreas expertas de conocimiento mediante máquinas de aprendizaje a las cuales se le debe implementar código explicito y ser sometidos a horas de ensayo, pero sus limitaciones se verán en la comprensión del ambiente y en que no van a realizar nuevas tareas o enfrentar obstáculos nuevos en ambientes dinámicos"(Fish, 2012. p.7).

Figura 7. Neural net. Fuente: http://www.shencorpore.es/images/Servicios/Neural_1.jpg

Una RNA tiene la capacidad de tomar decisiones nuevas en ambientes dinámicos, es decir que tiene la capacidad de derivar nueva información y descubrir nuevos patrones a medida que reconoce nuevos estados o problemas a través de sus mecanismos de percepción.

HiveMind resuelve este problema entre el sistema experto y las RNA. "HiverMind.WS permite no solo poner en práctica los comportamientos de la máquina al cual esta enfocada según el conocimiento y poner a prueba sus mecanismos de detección para una mayor eficacia y precisión, también para experimentar con múltiples metodologías, tecnologías y enfoques hacia la navegación de un mundo real de forma semiautónoma con robots desplegados que actúan como un súper-organismo esforzándose hacia los mismos objetivos y controlado por un único cerebro artificial"(Fish, 2012. p.8).La complejidad del medio ambiente se aborda mediante el servicio HiveMind.WS con su concepto de mapeo de la maquina-RNA.

La maquina esta capacitada para sacar conclusiones matizadas, es decir que no llega a la comprensión total de lo que detecta, sino que la detección de un componente o de una acción sería suficiente para producir una lista de posibilidades y evaluando la probabilidad de estas identifica cual es la correcta.

La capacidad de ser autónomas le permite tener la habilidad de tomar decisiones sobre el ambiente que los rodea o también permite la manipulación del usuario para cambiar objetivos o metas a realizar por la maquina. Un ejemplo son las arañas de motores de búsqueda que rastrean la web para indexar el contenido de miles de millones de páginas de contenido, principalmente no estructurado, contando con variedad de factores para evaluar la exactitud de sus resultados. Para ello se utiliza el conocimiento formal y particular de conceptos los cuales

ayudan a orientarse, sin una estructura de texto definida, pero asegurando que las palabras aparecerán, en el contexto adecuado y con una frecuencia adecuada con el fin de ser una pieza de contenido relevante para la búsqueda.

El otro campo importante en estos últimos tiempos y en la actualidad es el estudio de las secuencias biológicas como la del ADN y proteínas, campo difícil pero no imposible. Para ello realizan un trabajo interdisciplinar los científicos de la computación y los biólogos, con el fin de transformar las secuencias biológicas en información útil para comprender la función biológica y la estructura de las secuencias. La necesidad de este tipo de investigaciones es la predicción de comportamientos dado el conocimiento de las funciones y la estructura de una secuencia desconocida. Es decir que el problema central en el área de la biología computacional trata sobre la clasificación estructural y funcional de secuencias.

Con el fin de obtener información sobre la función, la estructura y la evolución de la proteína y las estructuras de proteínas, los análisis se realizan a grupos de proteínas y no individualmente, clasificándolas en superfamilias y familias.

¿Cómo se puede manejar y procesar una gran cantidad de datos para obtener información trascendental? El problema es debido a la cantidad de datos disponibles sin capacidad eficiente de procesamiento de estos, obteniendo como resultado la existencia de solamente 38.221 proteínas clasificadas según la SCOP[20] 1,75. “Podemos decir que estos resultados han sido provechosos pero poco profundos si comparamos el número total de estructuras disponibles que existían hasta junio de 2011: 68.467”.(Angadi& Venkatesulu, 2012, p.601).

La automatización para clasificación de estructuras ayuda al procesamiento y la rapidez en que nuevas estructuras se están generando para que los científicos tengan disponible la información de manera más veloz. Mediante una biblioteca de modelos ocultos de Markov HMM, se representan las estructuras de proteínas conocidas. El algoritmo utilizado para comparar secuencias biológicas es BLAST[21] , este tiene una precisión del 82% en sus comparaciones. AutoSCOP[22] calcula patrones de secuencia única para luego asignarlos a superfamilias SCOP con una predicción de superfamilias del 94% y de especificidad del 98%. Entonces, se

[20] SCOP: Structural Classification of Protein. Es la clasificación realizada manualmente por expertos y es aceptada por muchos investigadores como el estándar más alto para el análisis de la relación estructural, funcional y evolutiva de las proteínas.

[21] Basic Local AlignmentSearchTool.Es un algoritmo para comparar secuencias biológicas primarias, como por ejemplo las secuencias de diferentes tipos de proteínas o secuencias del ADN.

[22] Automated prediction of SCOP classifications using unique pattern-class mappings(Gewehr, Hintermair, & Zimmer, 2007)

redefine el problema de clasificar estructuras de proteínas por el problema del reconocimiento de patrones agrupando un conjunto de secuencia de proteínas.

Resolver problemas no lineales y complejos a través del procesamiento de datos imprecisos y ruidosos, es la función y la capacidad que tienen las redes neuronales artificiales.

Por medio del algoritmo de aprendizaje sin supervisión ART2(Carpenter, Grossberg, & Rosen, 1991, p.493) el cual esta diseñado para funcionar por continuos valores de entrada –valores de BLAST -, se entrena la red neuronal. Los resultados del experimento han sido validados analizándolos desde el punto de vista del método de control manual de SCOP. "El enfoque de ART2[23] lleva a buenas y razonables selecciones" (Angadi & Venkatesulu, 2012, p.607).

Ontología de la representación del problema de procesamiento y almacenamiento de los Big Data:

Ver archivo Onto2.cys

[23] AdaptiveResonanceTheory. Son redes neuronales que llevan a cabo una estable auto organización de patrones de entrada de secuencias arbitrarias por medio de los códigos de reconocimiento de secuencias.

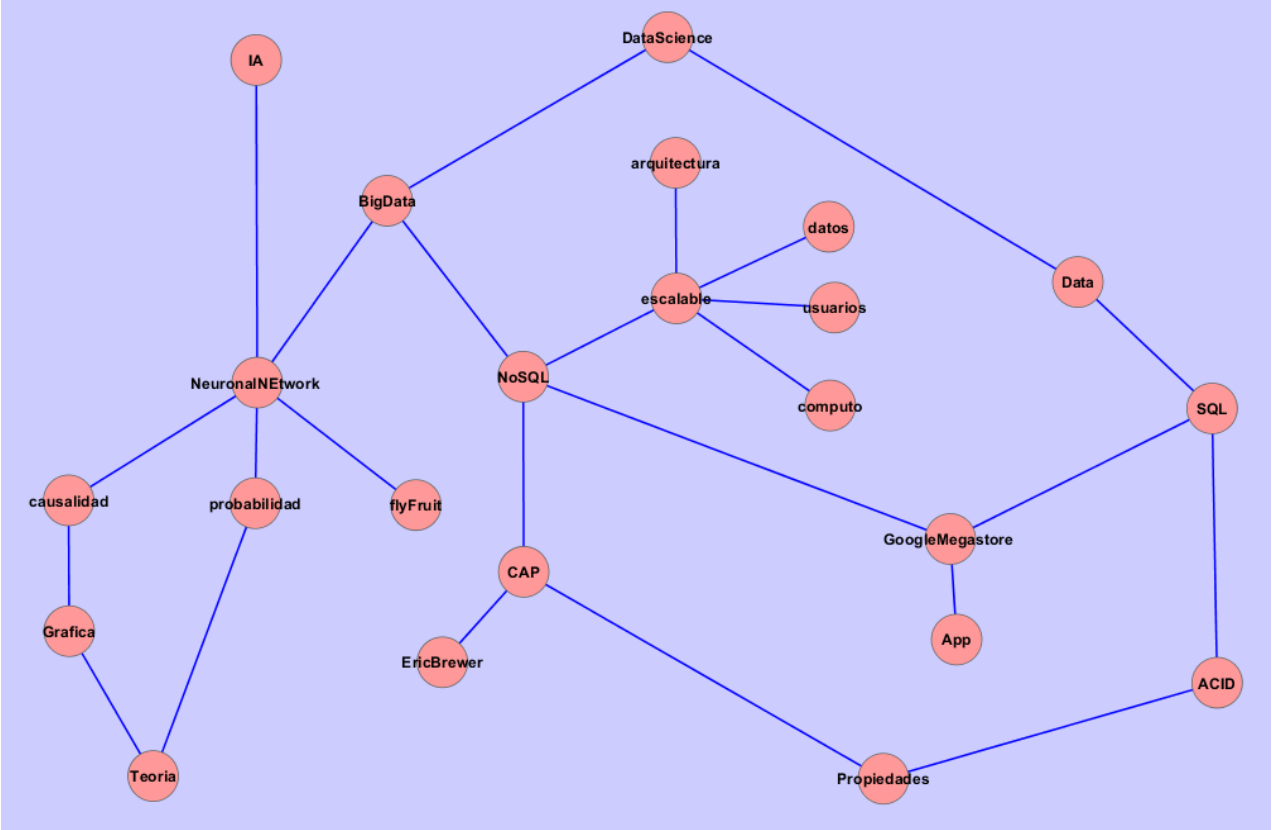


Figura 8. Meta-modelo de conocimiento 1.

Si es un ave, entonces, vuela, ¿y los pingüinos? "La capacidad de considerar el conocimiento inconsistente tan relevante como cualquier otro tipo y producir de esto un conocimiento nuevo e interesante es lo que se define como razonar bajo conocimiento inconsistente" (Ramírez, n.d., p.1).

Una ontología define los términos básicos y las relaciones que componen el vocabulario de un campo temático, así como las reglas para la combinación de términos y las relaciones que definen la extensión del vocabulario. "La Web semántica es un esfuerzo colectivo liderado por el W3C en la que se desarrollo una web que describe los datos en un formato común y

formal para ser útil a personas y maquinas por igual, permitiendo que los datos sean compartidos y reutilizados a través de aplicaciones, empresas…"(Abrahams et al., 2011). Una aplicación relevante actualmente, Victoria BodyCorporateServices(Abrahams et al., 2011), son las ontologías de dominios legales, con el fin de recuperar, extraer e integrar la información, para obtener razonamientos basados en casos judiciales, con el propósito de conocer la mejor alternativa para llegar a un acuerdo negociado. El sistema para la gestión de conflictos, creado por Abrahams, Condliffe, &Zeleznikow, le brinda al gobierno australiano la capacidad y la posibilidad de agilizar sus procesos judiciales a través de una aplicación construida con ontologías y redes bayesianas. "Un enfoque en la ontología basado en casos de razonamiento funciona bien cuando los hechos de una consulta coincidan exactamente con los hechos de los resultados almacenados en la base de casos. Es difícil deducir resultados judiciales, sin embargo, cuando algunos datos acerca de un caso se saben, pero también hay información incompleta, o, alternativamente, en donde algunos datos son los mismos que en los casos anteriores, pero otros hechos difieren. Este problema se conoce como el problema monotonicidad."(Abrahams et al., 2011, p.53).

La incertidumbre que se genera al tomar decisiones bajo inconsistencias está "restringida a un subconjunto de modelos que sean compatibles con el conocimiento cualitativo. Los modelos que cumplan estas restricciones se consideran como candidatos para la red subyacente. Estos modelos compatibles se usan para realizar la inferencia cuantitativa (Chang, Shoemaker, & Wang, 2011, p.1170). Mediante la representación numérica se puede relacionar grados de valor, vaguedad, en vez de una completa incertidumbre. La importancia de estos métodos estadísticos está en la capacidad de realizar predicciones de cualquier tipo y tomar decisiones con conocimiento incompleto o una cantidad de datos insuficiente para que podamos tomar conclusiones deductivas, las cuales son debido a: la ignorancia, el tiempo con que se debe tomar la decisión no es suficiente para disponer de todos los datos, problemas del azar, juicios morales, juicios éticos, etc.
Representar el conocimiento por medio de bases de datos no estructuradas es la propuesta de las ontologías y de la lógica paraconsistente, las cuales, permitiendo inconsistencias en la base de conocimiento se puede generar nuevas relaciones y nuevo conocimiento sin la necesidad de almacenar constantemente nueva información, sino se crean nuevas relaciones a través de los conceptos representados en la ontología y producen nueva información a partir de inferencias lógicas sobre la representación.

Figura 9. Spouting–Oiloncanvas (Kupka, 1920)

A continuación se muestra el meta-modelo cognitivo hecho con la herramienta Cytoscape
Ver archivo Onto1.cys

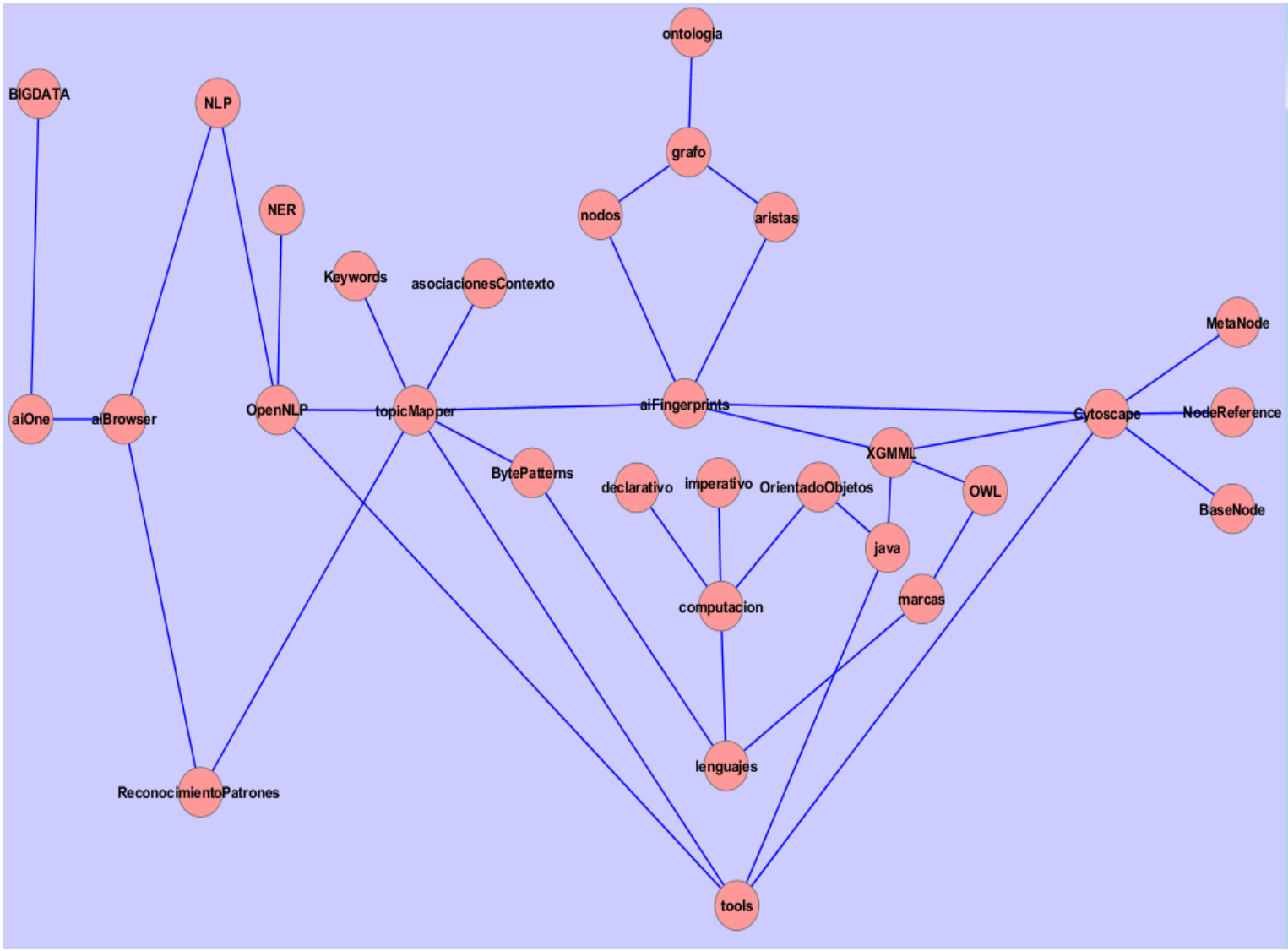


Figura 10. Meta-modelo de conocimiento 2

El texto desestructurado en la red no puede procesarse como una estructura de datos estática. Se requiere un aprovechamiento que permita la representación del conocimiento de forma que pueda ser procesada por maquinas.

DESCRIPCIÓN NARRATIVA DEL MODELO COGNITIVO

A continuación se expone la ontología de la figura Meta-modelo de conocimiento1.

La computadora tiene un poder de extraer y procesar datos de manera instantánea y perfecta, en comparación con el cerebro humano que recuerda y procesa datos sin estas velocidades y precisiones en el mismo tiempo. “La inteligencia de la máquina probablemente nunca remplazará la inteligencia humana, simplemente porque nosotros mismos no somos ‘máquinas pensantes’. Los seres humanos tenemos una inteligencia intuitiva que las máquinas de razonamiento simplemente no pueden igualar.”(Dreyfus&Dreyfus, 1986, p.44). La experiencia es la que hace que tengamos conocimientos sobre las cosas, nosotros no adquirimos conocimiento en forma de hechos y reglas. Para resaltar el hecho que los sistemas expertos no son iguales y no pueden llegar a un ser como un humano experto, Dreyfus&Dreyfus (1986) comparan la habilidad del experto para solucionar un problema respecto al de un sistema experto. “Por ejemplo, un boxeador parece reconocer que el momento de empezar no es un ataque siguiendo las normas y la combinación de varios hechos sobre la posición de su cuerpo y el de su oponente. Por el contrario, la escena visual desencadena toda la memoria de situaciones similares anteriores en las que un ataque fue exitoso. El boxeador está usando su intuición, o ‘know-how’ (...) ‘know-how’ es la clase de habilidad que usamos todo el tiempo a medida que avanzamos en nuestra tarea diaria. (…) Los expertos generalmente saben qué hacer porque tienen una comprensión madura y practica. Cuando están profundamente involucrados en hacer frente a su entorno, ellos no ven los problemas de manera individual e inconscientemente trabajan en su solución.”(Dreyfus&Dreyfus, 1986, p.45).
La inteligencia no está basada en hechos y reglas, las máquinas que tienen reglas y hechos son capaces de resolver problemas específicos, mas no son capaces de resolver cualquier tipo de problemas. Los enfoques de Newell y Simon (1996)[24] eran basados sobre los informes de como la gente resolvió algún tipo de problema para sistematizar estos procesos. En los años sesenta se cambia la perspectiva de Newell y Simon y se reconoce que “para resolver los problemas del mundo real, la computadora tiene que simular de alguna forma la intuición y la comprensión del mundo real.”(Dreyfus&Dreyfus, 1986). Sin embargo se encuentra el mismo problema el cual un programa solo servía para resolver un caso específico. El primer programa de computador que capta el lenguaje humano fue

[24] Las ciencias de lo artificial.

SHRDLU[25], estaba basado en reglas gramaticales, la semántica y hechos. A partir de este momento nace una nueva forma de abordar el problema sobre el poder de las máquinas para tomar decisiones y puedan aprender: los micromundos. Los micromundos son espacios de dominios restringidos, los cuales resuelven cada uno problemas simples. “Minsky y Papert creen que mediante la combinación de un gran número de estos micromundos, los programadores de computadoras con el tiempo podrían dar la comprensión de un mundo real.”(Dreyfus&Dreyfus, 1986). En la actualidad podemos representar estos micromundos como redes neurales artificiales, RNA, especializadas en resolver algún tipo de problema y observar la interacción y relaciones entre estas para producir una mejor y más amplia comprensión del problema. Un buen ejemplo de estudios y trabajos actuales en los cuales utilizan IA para descubrir patrones entre miles de datos realizados en el área de la bioinformática para interpretar y conocer la cadena del genoma humano y el análisis de estructuras de proteínas.

Figura 11. Dubuffet, Jean. Theprotestor

La importancia derivada de la investigación de Dreyfus (1986) recae en las capacidades que deben tener los sistemas los cuales van a resolver esos problemas. Primero se observa que la cantidad de datos para tomar decisiones es ilimitada y se necesita de una alta capacidad de escalabilidad tanto de datos, como de usuarios y de computación. Segundo, debemos ser conscientes del hecho que resulta imposible incluir todas las excepciones posibles de las reglas, porque no se puede tener un sistema de reglas completo, con el fin de poder cubrir todas las situaciones posibles. Los sistemas deben aceptar las inconsistencias. “En el mundo real, un sistema de reglas tiene que ser incompleto. La ley, por ejemplo, siempre se esfuerza por completo, pero nunca lo consigue. La ‘Ley común’ se basa más en los precedentes que en un código específico. Pero el gran número de abogados en la empresa nos dice que es imposible desarrollar un código de leyes tan completo que todas las situaciones están claramente cubiertas.” Con esto es posible suponer que hace 25 años se empezaba a marcar la necesidad de trabajar con sistemas de bases de datos que permitiesen escalabilidad de manera rápida y el manejo de inconsistencias.

En la actualidad existen herramientas que permiten construir aplicaciones para el aprendizaje de máquinas como ai-one. Ai –one proporciona kits de desarrollo para programadores. Esta tecnología permite encontrar respuestas rápidamente en una mina de datos sin estructuras por medio de ai-Browser. “Esta tecnología transforma la información en un conjunto de reglas generalizadas (…) puede

[25]Diseñado en el MIT por Terry Winograd en 1968.

aprender conceptos tomando múltiples capas de información contextual, tales como estructuras de información visual de las imágenes o toda la gama de contexto léxico, sintáctico, semántico y pragmático en el texto. Estas estructuras de información se transforman en conjuntos de reglas generalizadas, que luego pueden ser aplicados a los hechos de entrada adicionales."(Gartner, 2012, párr.2.)

Ai –Browser es un prototipo de interacción hombre-maquina, HCI. "La interacción humano-computadora es una disciplina que estudia el diseño, la evaluación e implementación de sistemas informáticos interactivos para uso humano con el estudio de los fenómenos más importantes que los rodean."(ACM, 2009, párr.2). Ai –Browser combina el aprendizaje de texto natural por medio de OpenNLP y con la API Topic_Mapper detecta patrones en el texto después de ser procesado el aprendizaje. La subtarea de OpenNLP para extraer información que busca localizar y clasificar los elementos atómicos en el texto en categorías predefinidas es name-entityrecognition. Topic-Mapper crea una ontología ligera[26] la cual se representa en un grafo de coordenadas $G(V, A)$. Son de la forma $(x, y_{0\to t})$; x representa los nodos (vértices) que son las palabras claves detectadas y $y_{0\to t}$ representa la serie de palabras de asociación para la palabra clave específica x. Esta topología se crea para el archivo dinámico, documento con un grafo $G(V, A)$, que describe cada relación entre cada elemento de dato. Este grafo se llama *ai-Fingerprints*, el cual esta descrito en XGMML[27], para acomodar los datos dinámicamente teniendo en cuenta la variación dinámica a través del tiempo.

ai-Fingerprints es un "modelo de representación del conocimiento sin perdidas"(Gartner, 2012, párr.40.) debido a que capta todas las relaciones del grafo y revela el significado de una palabra que solo puede aparecer una sola vez y su ventaja es que trabaja con lenguajes estándar de computación existentes. Topic-Mapper es "preciso porque busca solo patrones de Bytes (Byte Patterns) pero debe existir consistencia en las reglas gramaticales" (Ibíd.). Los resultados de ai-Browser se dan en Cityoscape, se muestran las asociaciones, la relevancia, los patrones, anomalías[28]

Nota: Revisar el anexo 1 si desea obtener más precisiones sobre las herramientas usadas y la metodología para conseguir la información conceptual y software para realizar las ontologías.

[26] Lightweight ontologies: http://www.ai-one.com/tag/lightweight-ontology/

[27]Lenguaje de Marcado Extensible para el Modelado Grafico, es una aplicación XML basado en Lenguaje de Modelado Grafico o Lenguaje Meta-Grafico, utilizado para la descripción grafica.

[28]http://www.youtube.com/user/semsys

2.2 Arquitectura a gran escala

A continuación se propone la forma de abordar este problema con mayor eficiencia, eficacia y precisión, para realizar el proceso de validación de muchos argumentos. Por medio de la inteligencia artificial y herramientas opensource con gran capacidad de interoperabilidad es posible potencializar el producto software que nace de esta investigación, Automatización del Modelo de Toulmin.

La forma propuesta no se desarrolla en la actual solución dado al alto nivel de implementación tecnológica de última punta y por lo tanto mayor cantidad de tiempo requerida. Se deja esta propuesta con el objetivo de continuar desarrollando una automatización más poderosa y con mayor inteligencia artificial en estudios de posgrado.

El estado del arte que se debería manejar para los problemas de aprendizaje son: Máquinas de aprendizaje, utilizando el perceptron, el cual hace uso del algoritmo Logistic Reggresion para trabajar en problemas de clasificación de patrones y se optimiza con el algoritmo Stochastic Gradient Descent, el cual permite comprimir con bajos niveles de error las dimensiones de los Big Data pertenecientes al los conjuntos de entrenamiento y conjuntos nuevos de datos. Este estado del arte es presentado por el profesor Andrew Ng de Stanford en el curso virtual de Machine Learning de la plataforma gratuita Coursera (Ng, 2012).

En las recomendaciones (sección 5) se presenta con mayor detalle y profundidad cómo abordar esta solución

2.3 Teoría de la argumentación y sus automatizaciones

Los sistemas estudiados a continuación tienen como fin realizar aportes al desarrollo en el campo de la teoría de la argumentación con el objetivo de mejorar las habilidades del pensamiento crítico en la sociedad, desde las escuelas hasta universidades y profesiones como en el derecho. Estos estudios sobre los argumentos se basan principalmente en la investigación realizada *en Los Usos de la Argumentación* de Stephen Toulmin en 1953.

Programas diversos, existentes en la actualidad, tienen también como objetivo el análisis de los argumentos en la comunidad académica para mejorar el desempeño del pensamiento critico en sus comunidades. Estos se basan en distintas soluciones para resolver el problema de validar si el estudiante, o usuario,

esta argumentando de manera correcta. Revisando, se evidencia que ninguno de los sistemas disponibles en la actualidad para el análisis de los argumentos aborda la solución desde el punto de vista del procesamiento del lenguaje natural.

En el caso de Araucaria de Reed Y Rowe (2001), el problema se trata "mediante la identificación de la estructura de un argumento en términos de sus componentes y las relaciones entre ellos".(Reed & Rowe, 2001) Este software ofrece una interfaz que soporta el proceso de diagramación diseñado en XML para la descripción de la estructura argumental.

El software ArgumentMapping (Argument, 2010) realizado en 2006, es otro de esta clase, interesado en la teoría de la argumentación para mejorar las habilidades de los estudiantes aprovechando las tecnologías de la información. ArgumentMapping intenta tratar este problema con procesamiento del lenguaje natural, sin embargo evalúa las expresiones del argumento por medio de identificaciones visuales sobre la estructura. Estas identificaciones se realizan por medio de señalamientos de flechas de una categoría hacia otra categoría, la cual el estudiante cree que debe ser relacionada para construir el argumento. El sistema es capaz de interpretar estas relaciones y proporcionar una validez de que tanto comprende las estructuras principales de un argumento. ArgumentMapping tiene un enfoque social el cual su uso da"un gran potencial para evolucionar nuestra manera de entender y ejecutar la democracia, en referencia a la evolución en curso de e-democracia" (ArgumentMap, 2012)

Otros software realizan estos trabajos pero utilizando como base las tecnologías descritas anteriormente. Este es el caso de Rationale[29], el cual implementa paquetes de ArgumentMapping para desarrollar este tipo de tareas (validar las estructuras de la argumentación en estudiantes).

Otros sistemas son ARGUE! y ARGUMED. El software ARGUE! es un sistema desarrollado para la realización de una teoría de la argumentación pero "esta no es lo suficientemente natural para aplicar a la argumentación cotidiana. Su interfaz permite al usuario dibujar y organizar datos argumentativos en la pantalla"(Verheij, 2003,p. 320)

Una mejora de ARGUE! fue ARGUMED basado en DEFLOG(Verheij, 2003, p. 320) El aporte realizado es la sensibilidad para construir las partes del argumento mostrados en la interfaz con el mouse. No es necesario dibujar flechas las cuales

[29] http://lpr.oxfordjournals.org/content/6/1-4/23.full.pdf?keytype=ref&ijkey=GFHrAQMNkJ09woP

relacionan las partes del argumento, simplemente se colocan al lado de las que corresponde y el sistema tiene la capacidad de interpretar si es valido o no la estructura del argumento realizada.

“Los sistemas ARGUE! y ARGUMED, se construyen con el objetivo de guiar al usuario a la producción de argumentos, mas no para remplazar el razonamiento.”(Verheij, 2003, p. 391) Es decir que no se construyeron para realizar inferencias sobre el sentido del lenguaje natural.

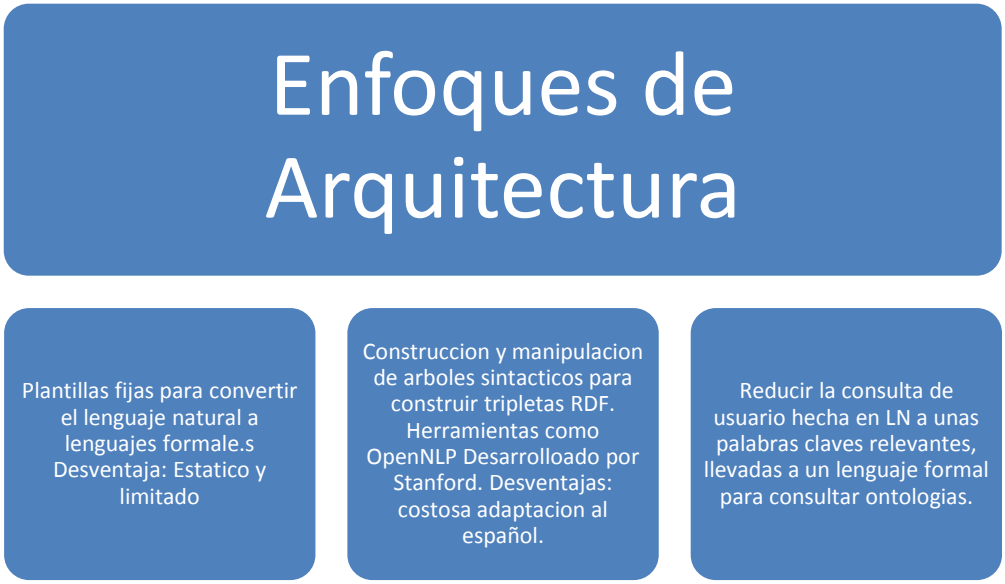


Figura 12. **Enfoques de arquitecturas**

Un proceso común observado es que es necesario pasar una consulta realizada en lenguaje natural a una consulta SPARQL, es decir que se necesita llevar el LN a un modelo de tripletas RDF (sujeto, predicado, objeto) el cual, sea compatible con los conceptos ontológicos que se convierten en sentencias SPARQL.A continuación se muestran las arquitecturas básicas para convertir consultas del lenguaje natural a un lenguaje formal de consultas de ontologías tomado de la investigación de (Ruiz &Belalcázar, 2012).

3. Especificación de la arquitectura

Para el desarrollo del prototipo esta investigación se apoya del proceso unificado de desarrollo de software según Jacobson et al. (2000)

El proceso de desarrollo es un “conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software”. Está basado en componentes interconectados a través de interfaces. Dirigido por casos de uso, centrado en la arquitectura, y es iterativo e incremental.

La arquitectura es un “Conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema, las interfaces entre ellos, su comportamiento, sus colaboraciones, y su composición.”(Jacobson et al, 2000).

Uno de los objetivos es la formalización de los procesos argumentativos según el Modelo de Toulmin. Para ello es necesario realizar una buena selección de los elementos estructurales del sistema SYSTOUL.

3.1 Planteamiento del problema

El gobierno Colombiano, en su pretensión de ser un estado democrático, debería estar orientado en la búsqueda de proyectos que beneficien ampliamente a toda su sociedad. Uno de los entornos mediante los cuales el estado puede desarrollar proyectos de nación son las instituciones educativas, en las cuales, según el Ministerio de Educación Nacional, se busca "Lograr una educación de calidad, que forme mejores seres humanos, ciudadanos con valores éticos, competentes, respetuosos de lo público, que ejercen los derechos humanos, cumplen con sus deberes y conviven en paz. Una educación que genere oportunidades legítimas de progreso y prosperidad para ellos y para el país." (MEN, 2011)

La Universidad Industrial de Santander muestra en su misión la importancia de formar ciudadanos caracterizados por la concepción y materialización de la democracia; la importancia de formar estudiantes que generen procesos de cambio por el progreso y mejor calidad de vida de la comunidad, con reflexión crítica y que construyan una cultura de vida[30]. Todos estos valores podrían

[30] http://www.uis.edu.co/webUIS/es/acercaUis/index.html

construirse por vía de la argumentación como una de las estrategias posibles, para, en consecuencia, construir una nación democrática.

La argumentación es, según lo expuesto, la base para una sociedad democrática, para una sociedad en la que los ciudadanos participan en la formulación de proyectos, y persuaden a otros de su conveniencia, de manera que logren el apoyo necesario para ser tenidos en cuenta por las mayorías; pero también para una sociedad en la que los sujetos convivan con base en el respeto y el reconocimiento del punto de vista de los otros. Esto implica que los procesos educativos basados en la argumentación tendrían que poner al sujeto en posición de considerar que las ideas que son producto de su razonamiento no se constituyen por sí mismas en verdades, y por ellos no pueden ser impuestas a los demás: "La pretensión implícita en una aseveración es como la pretensión o reivindicación de un derecho o un título. Como ocurre con la reivindicación de un derecho, aunque puede darse el caso de que se conceda sin discusión alguna, su valor depende de los méritos de los argumentos que puedan aducirse en su apoyo" (Toulmin, 2003, p.29).

La técnica propuesta por Stephen Toulmin (2003), el Modelo de Toulmin, se constituye en una herramienta que contribuye a los fines mencionados de cara a la construcción de un estado democrático. La manera sistemática como está propuesto el Modelo hace que sea posible su automatización quedando ésta como instrumento y estrategia para aprender a argumentar.

La tecnología, por su parte, permite disponer de un puente entre la teoría y la práctica. Una disciplina como la biogenética, por ejemplo, no sólo propone soluciones a problemas existentes, sino que también propone retos en términos de la creación de modelos teóricos y conocimientos que "revolucionan y convulsionan la estructura de las ciencias". (Vargas Guillen, 2006, p.13). Mientras que la técnica se trata de modos de hacer, la tecnología transforma la realidad en la medida que ofrece dispositivos que materializan conocimientos y permiten a los sujetos experimentarlos. “La tecnología es la forma contemporánea de dar curso a los saberes en medio de la realidad. Por eso, para ella la técnica es sólo un antecedente en la medida en que ésta no implica ni supone los niveles de racionalidad y de formalización que le son propios. Queda así descartada la posibilidad de una reducción de la una a la otra. (…) La técnica habla de la reproducción de un modo de hacer [arte] mientras (…) la tecnología tiene intereses de generar una comprensión sobre las cosas o al menos intenta ser fuente de hipótesis para ello”. (Vargas, 2006, p. 14).

A partir de este planteamiento es posible la transformación del Modelo de Toulmin en una herramienta computacional que ubique a los aprendices en la posición de experimentar la construcción de argumentos. Para ello se busca con este proyecto una representación válida del Modelo y su digitalización o automatización que permita llegar a una comprensión de las estructuras de la argumentación propuestas en tal modelo, es decir generar el conocimiento de la argumentación a la sociedad por medio de la tecnología, "(…) Más aún, toda teoría de lo social o de la naturaleza que no implique un mejor modo de actuar, está en un verdadero déficit con la historia, es decir, todo conocimiento parece ser incompleto si no ofrece una tecnología." (Vargas, 2006, p.14).

3.2 Alcance

Por medio de un procesador de lenguaje simple se pretende vincular las palabras de consulta del vocabulario o jerga utilizado en el argumento almacenado en la ontología de dominio. Esto permitirá al usuario realizar contrastaciones por medio de términos claves, fragmentos de sentencias o sentencias completas en español, retornando una respuesta sobre qué tanto corresponde su texto, en probabilidad, respecto al argumento del dominio de la ontología.

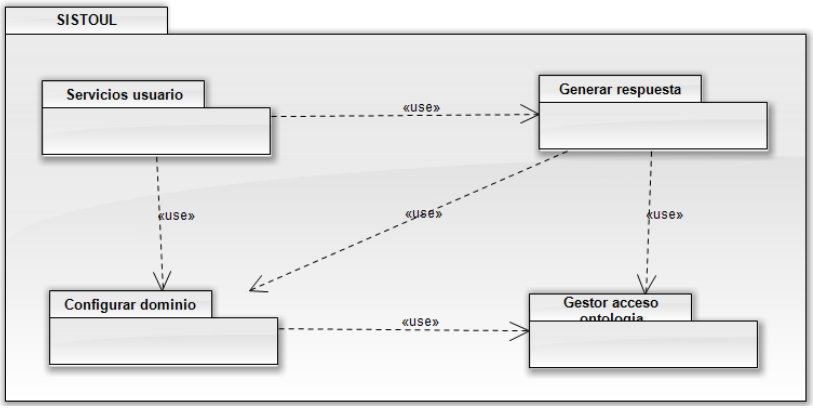


Figura 13. Subsistemas del sistema SYSTOUL

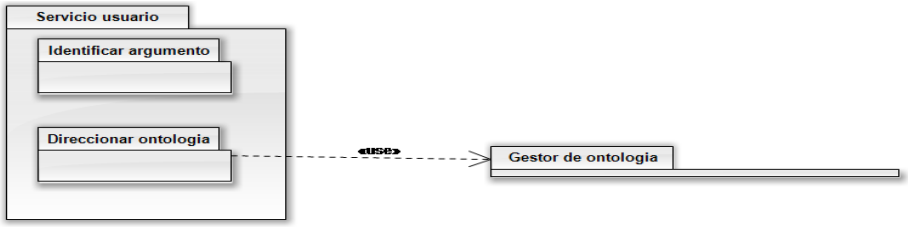
Módulo	Descripción	Casos de uso
Configurar Dominio	Dispuesto al administrador para configurar el sistema a un domino a partir del direccionamiento de una ontología especifica, del argumento especifico.	CU1, CU2, CU3
Servicios de Usuario	Dispuesto para que los usuarios interactúen con el sistema	CU4, CU5

Generar Respuesta	Soporta los procesos para retornar una respuesta en un formato que puede ser entendido por el usuario.	CU5

Tabla 11. Descripción de los subsistemas

Cada módulo contiene componentes, subsistemas, que lo apoyan en sus procesos. El modulo Servicio a usuario:

Figura 14. Subsistema servicio usuario.



- **Identificar argumento:** Este componente permite al usuario realizar una consulta sobre sus identificaciones por medio del lenguaje natural presentado en pantalla brindándole opciones al usuario de subrayar con colores diferentes las estructuras de una argumentación dada por el modelo de Toulmin, forma tal que permita al sistema hacer un mejor análisis de ésta.
- **Componente Direccionar Ontología:** Este componente permite al administrador dar la ubicación de una ontología al sistema ayudándose del Subsistema Configurar Dominio.
- **Componente Gestor Acceso a Ontología:** Este componente permite cargar la ontología especificada.

Módulo configuración de dominio:

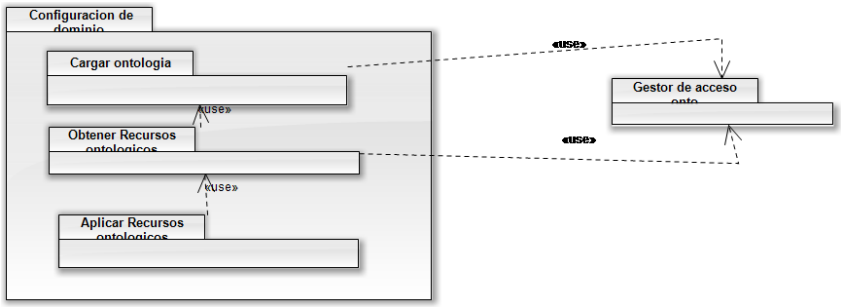


Figura 15.Subsistema configuración de dominio

- **Componente cargar ontología:** Este componente permite aplicar una ontología al sistema, proporcionándole al Gestor de Acceso a la Ontología la ruta o ubicación de la ontología.
- **Componente Obtener Recursos Ontológicos:** Este componente permite a partir de la ontología de dominio, generar una serie de recursos ontológicos necesarios para la adaptación del sistema al dominio especificado en la ontología.
- **Componente Aplicar Recursos Ontológicos al Sistema:** Este componente utiliza los recursos ontológicos obtenidos por el componente obtener recursos ontológicos para adaptar el sistema al dominio especificado.

Módulo Generar respuesta:

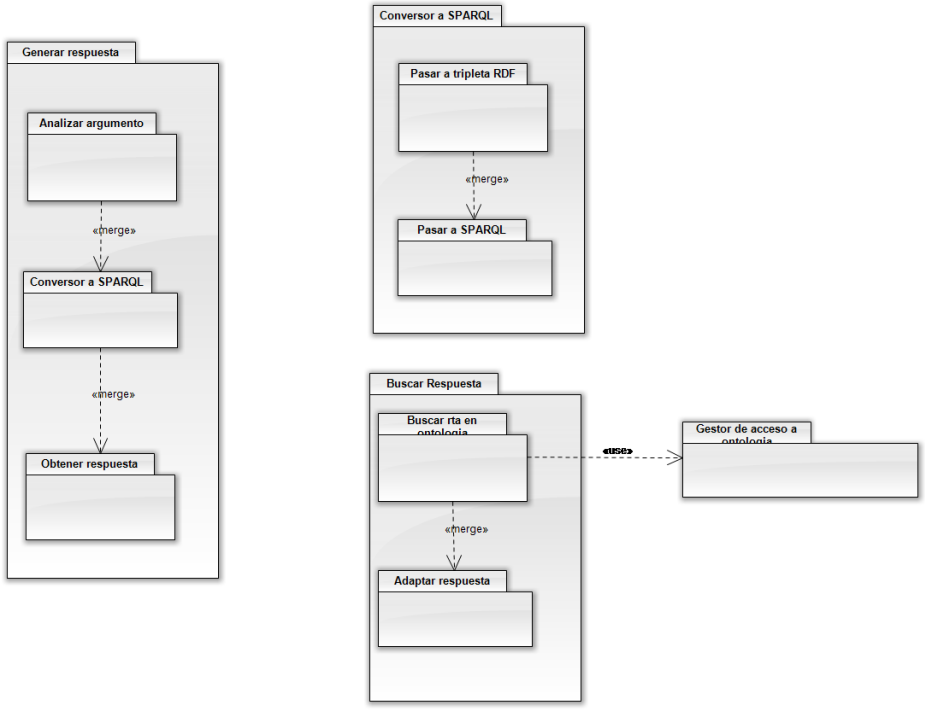


Figura 16. SubsistemaGenerar respuesta

- **Componente Analizar identificación del argumento**: Este componente utiliza la salida de componente identificar argumento y soporta los procesos

de análisis en lenguaje natural. La salida de este componente es la el argumento analizado y restructurado.

- **Componente Conversor a Lenguaje Formal:** Este componente utiliza la salida del componente analizar argumento LN y soporta los procesos de paso de la pregunta, argumento, analizada a un lenguaje formal de ontologías, la salida de este componente es una consulta en leguaje formal.
- **Componente Buscar y Adaptar Respuesta:** Este componente utiliza la salida del componente conversor a lenguaje formal y soporta los procesos de búsqueda de respuesta en la ontología y el envío de ésta al usuario. La salida de este componente es una respuesta en un formato comprensible para el usuario.

Cada componente de los subsistemas tiene unos paquetes los cuales representan las funcionalidades específicas de cada subsistema.

Subsistema Generar respuesta

Conformado por: Componente Analizar y Restructurar Pregunta LN, Componente Conversor a Lenguaje Formal, Componente Buscar y Adaptar Respuesta.

Componente Analizar y Restructurar Argumento en LN

- **Analizador morfológico:** Divide la oración en sentencias, se dividen cada punto. Su salida es un vector donde se encuentran todas las frases con una cantidad igual al número puntos existentes. (OpenNLP: SentenceDetector)
- **Tokenizador:** Divide las sentencias y las convierte en un vector con una cantidad igual al numero de palabras y caracteres que existen en todo el argumento.
- **Mapear a Términos Ontológicos:** Este componente utiliza la salida del componente eliminador tokenizador y es el encargado de mapear las palabras relevantes con su respectivo término ontológico, utilizando los recursos ontológicos generados por el componente obtener recursos ontológicos.

Componente convertir a lenguaje formal –SPARQL-

- **Pasar a tripleta RDF:** Usa la salida de analizar y restructurar pregunta LN y toma los términos mapeados ontológicos y se generan tripletas RDF.

- **Pasar a SPARQL:** Usa la salida de pasar a tripletas RDF y genera una consulta sintácticamente correcta en lenguajes SPARQL. La salida es una consulta SPARQL.

Componente Obtener respuesta

- **Buscar respuesta en ontología:** Utiliza la salida de conversor a lenguaje formal y extrae información de la ontología ayudándose del gestor de acceso a la ontología.
- **Adaptar respuestas a usuarios:** Usa la salida del componente anterior y la adapta a un formato comprensible para el usuario, el cual la pueda leer.

Diagrama de despliegue

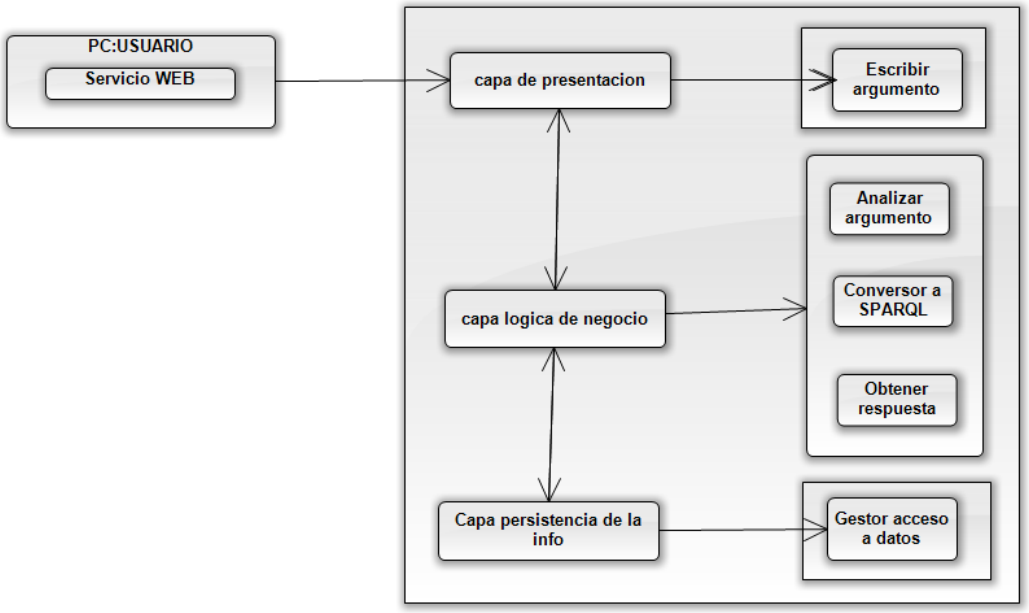


Figura 17. Diagrama de despliegue

3.3 Implementación del prototipo SYSTOUL

El Proceso Unificado de Desarrollo de Software –PU– (Jacobson, Booch&Rumbaugh, 2000) puede entenderse como un conjunto de actividades

para transformar los requisitos manifestados por los usuarios en un sistema software, con base en componentes interconectados a través de interfaces.

El PU, que orienta el desarrollo tecnológico, es dirigido por casos de uso, centrado en la arquitectura y es iterativo e incremental.

Dirigido por casos de uso

Los casos de uso son las secuencias de eventos generales que describen todas las acciones posibles entre un actor y el sistema para un fragmento de funcionalidad dada.

Estos son vitales para la adquisición de los requisitos funcionales: “Los casos de uso son herramientas que nos sirven para determinar y especificar las funcionalidades del sistema que orientan las fases de diseño, implementación y prueba.” (Jacobson et al, 2000, p. 4).

Centrado en la arquitectura

El sistema debe estar organizado de alguna manera lógica para darle funcionalidad, dando importancia al diseño de la estructura para brindar el servicio o cumplir su propósito, según Booch et al (2000) es una vista del diseño que muestra la relación entre los componentes del sistema software. Dado que la modificación de los casos de uso puede alterar la arquitectura y viceversa, el diseño de estos dos aspectos debería hacerse en paralelo.

Iterativo e incremental

Compuesta por 4 fases: inicio, elaboración, construcción y transición y cada una de estas realiza iteraciones para añadir retroalimentación al producto a medida que se avance.

La fase de inicio y elaboración corresponden al análisis de requerimientos e investigación realizada al libro Los Usos de la Argumentación de Stephen Toulmin (2010) y a la revisión del estado del arte (capítulo II)

Se utiliza el ambiente de desarrollo integrado Netbeans para proyectos JAVA. Con el fin de darle gran interoperabilidad al artefacto, se construye el proyecto con un administrador de proyectos software, llamado Maven. Maven es una herramienta de compresión. No depende del shell del sistema operativo, este se basa en archivos de configuración xml y clases Java para realizar las tareas. “Es similar en

funcionalidad a Apache Ant" (Maven, 2012). Para ver más detalle de como instalar Maven y como funciona Maven ver anexo 1.
Maven utiliza un POM (Project ObjectModel), –Proyecto de Modelo de Objetos-. POM "se trata de una representación XML de un proyecto Maven en un fichero llamado pom.xml." (Maven, 2012). Esta es la unidad fundamental para trabajar en Maven. En vez de utilizar librerías de las distintas herramientas Java y adaptarlas, Maven utiliza de manera uniforme dependencias y repositorios de las funcionalidades de las API para JAVA, este factor es lo que hace de este software una herramienta con gran capacidad de interoperabilidad.

A continuación se expone la fase de construcción. El objetivo es especificar la arquitectura anteriormente diseñada en un nivel técnico.

MODULOS

Modulo de configuración del dominio, componentes:

- **Cargar ontología**
- **Obtener recursos ontológicos**
- **Aplicar recursos ontológicos**

Componente Cargar ontología

Permite aplicar una ontología en OWL al sistema, proporcionándole al Gestor de Acceso a la Ontología la ruta o ubicación de la ontología. El Gestor de Acceso a la Ontología, es el que permite la interacción con la ontología, este gestor se ayuda de la herramienta JENA que provee todas las funcionalidades necesarias para la gestión de ontologías.

Para interactuar con JENA desde Maven-Netbeans simplemente llamamos a sus repositorios y dependencias dentro del POM.xml de nuestro proyecto (así para cada API). Veamos el ejemplo:

Dependencia JENA	Repositorio JENA
<dependency> <groupId>org.apache.jena</groupId> <artifactId>jena-arq</artifactId> <version>2.9.3</version> </dependency>	<repository> <id>apache-repo-releases</id> <url>https://repository.apache.org/content/repositories/releases/</url> <releases> <enabled>true</enabled> </releases> </repository>

Tabla 12. Maven: dependencia y repositorio para JENA

Componente Obtener recursos ontológicos

Este componente permite a partir de la ontología de dominio aplicada, generar una serie de recursos ontológicos necesarios para la adaptación del sistema al dominio especificado en la ontología.

El funcionamiento se enmarca desde el punto de vista de Modelo Ontológico propuesto por Hasany, Jantan, & Selemat (2010). Este se describe a continuación:

MO :< Cs, Is, DPs, OPs, GPOCC, GDPC >

Un modelo ontológico MO – que está compuesto por Cs, Is, DPs, OPs, GPOCC, GDPC

En donde:

Cs = Conjunto de clases.
Is = Conjunto de Individuals o instancias.
OPs = Conjunto de objectproperties
DPs = Conjunto de datatypeproperties
Ts = Conjunto de literal datatypes

RO: Elementos que definen el vocabulario formal de la ontología. Se denota RO.

GPOCC = Conjunto de objectproperties que mapean Ci a Cj, en donde Ci, Cj pertenecen Cs
GDPC = Conjunto de datatypeproperties(atributos) que mapean Ci a DPi, en donde Ci pertenecen Cs, DPi pertenecen DPs
Donde,

GPOCC = Conjunto de sentencias, en donde cada sentencian es una tripleta del tipo:
< Cs, Is > x POs x < Cs, Is >

DPs= Es un conjunto de sentencias, en donde cada sentencia es una tripleta del tipo:

< CS, IS > x DPs x <Ts>

El problema es convertir una consulta de “lenguaje formal Nq a todas las posibles consultas formales Fq1,….., Fqn”(Hasany, Jantan, & Selemat, 2010, p. 67). Esto implica dos pasos:

1. f1: Nq→ Pq1……..Pqn
2. f2: Pqi→Fqi | null

Donde f1 proporciona un mapeo a las palabras del usuario con el vocabulario de la ontología RO. La colección de todos los términos mapeados con su categoría ontológica es un patrón Pq.

f2 aplica reglas para interpretar la combinación en el Pqi y obtener una consulta formal Fq que sea representativa de Nq.

A cada RO se le asigna su categoría ontológica. Se extraen sus respectivos sinónimos, encontrados en la misma ontología, y se le asigna su respectivo lema por medio de la herramienta OpenNLP.

Los sinónimos se extraen directamente de la sinonimia proporcionada en la ontología de cada argumento particular.

Componente Aplicar Recursos Ontologicos a SYSTOUL

Con los recursos ontológicos anteriormente obtenidos este componente adapta el sistema al dominio especificado.

Con el vocabulario formal de la ontología en las listas de RO y en los grafos de asociación, este componente se encarga de la serializacion de estos productos y su posterior almacenamiento en archivos de texto plano para que estén disponibles para los componentes Identificar argumento y Generar respuesta. Así el sistema queda configurado al dominio específico de la ontología cargada.

Generar lexicón de RO y adaptación a OpenNLP

Componente encargado de crear un lexicón de reconocimiento de RO que contiene las listas de los RO, la cual se adapta a OpenNLP. Esto se hace con el fin de tomar las multipalabras que conforman un RO como una sola entidad perteneciente a los RO de la ontología particular, evitando problemas en el análisis, generados por lematizar por separado palabras que se refieren a una misma entidad, pero que se encuentran separadas.

Ejemplo de adaptación:
Categorización de las palabras “Probablemente es súbdito” con OpenNLP sin adaptar:

palabra	categoría	lema
Probablemente	NC	probablemente
es	PREP	es
súbdito	NC	súbdito

Tabla 13. Ejemplo lematización de palabras con OpenNLP sin adaptar.

Categorización de las palabras “Probablemente es súbidto” con OpenNLPadaptado:

palabra	categoría	lema
Probablemente es súbdito	RO	Probablemente es súbdito

Tabla 14. Ejemplolematización de palabras con OpenNLP adaptado.

Modulo analizar y restructurar argumento en LN

- **Analizador Morfológico**
- **Eliminar palabras y frase**
- **Mapear a términos ontológicos**

Componente analizador morfológico

Implementamos las herramientas de OpenNLPSenteceDetectorME y TokenizerME con el fin de obtener en vectores todas las palabras y signos de puntuación del argumento. Ejemplo

Palabra	Categoría gramatical	lema
El	Frase ruidosa	El
Probablemente es súbdito	RO	Probablemente es súbdito
Británico	RO	Británico

Tabla 15. Salida componente analizadormorfológicoOpenNLP adaptado

Componente Mapear a términos ontológicos

La salida del componente analizador morfológico es utilizada por este componente, el cual es encargado de mapear las palabras relevantes con su respectivo termino ontológico, utilizando los recursos ontológicos generados por el componente Obtener recursos ontológicos.

Cuando tenemos todas las palabras señaladas por el estudiante, el sistema procede a compararlas con las palabras del vocabulario ontológico RO.

Se toma cada palabra y se compara con la lista de recursos ontológicos, buscando que esta palabra este escrita exactamente igual, si la encuentra le asigna su respectivo tipo de recurso ontológico (clase, individual, obejectproperty, datatypeproperty). Si no encuentra la palabra relevante exactamente escrita el sistema toma el lema de la palabra relevante y la compara con los lemas de los RO, si coinciden la emparejan con el RO correspondiente, asignándole su categoría ontológica.

Si no encuentra la palabra relevante por medio de la comparación con los lemas de los RO, toma el lema de la palabra relevante y lo compara con el lema de los sinónimos y si coinciden empareja la palabra relevante con el nombre y la categoría ontológica del RO al que pertenece ese sinónimo.

A esta serie de emparejamiento se les llama patrones – P-, cuyo resultado final es una lista de RO y su respectiva categoría ontológica, como se muestra a continuación:

Lista de patrones	
RO	**Categoría**
Harry	Clase
Nació	Objectpropoerty
Bermuda	Datatypeproperty
Probablemente es súbdito	Clase

Tabla 16. Ejemplo salida componente mapear a términos ontológicos

El componente retorna la lista de patrones.

Modulo Convertir a lenguaje formal, componentes

- **Pasar a tripletas RDF**
- **Pasar a SPARQL**

Componente Pasar a tripletas RDF

La salida del componente Mapear a términos ontológicos es utilizada por este componente. Se encarga de tomar los términos ontológicos mapeados (patrones P) y según estos términos generar las respectivas tripletas RDF, la salida de este componente son todas las tripletas RDF generadas.

La investigación de Hasany, Jantan, & Selemat (2010) propone el método Queriontо, el cual no se requiere un estudio previo del dominio para generar tripletas RDF. Este método consiste en identificar unas primitivas necesarias para generar una consulta SPARQL.

Relation triple

> Esta tipo de tripleta se utiliza para determinar la conexión entre clases, por ejemplo la tripleta {?i1 ?p1 ?i2} indica que las instancias representadas por ?i1 son conectadas a instancias representadas por ?i2 con ayuda de la objectproperty representado por ?p1. Para la exploración del grafo se requiere GOPCC para determinar estas tripletas de relación.

Classassociation triple

> Este tipo de tripleta asocia a una variable los recursos de la clase ontológica para acceder a las instancias de la clase, por ejemplo, en las tripletas {?i1 rdf:type :C1} o {?i1 a :C1}, la variable ?i1 es usada para representar las instancias de la clase C1.

Propertyspecification triple

> Este tipo de tripleta es usada cuando un recurso property es determinado en el mapeo. Por ejemplo en la tripleta {?i1 :supervisa ?i2} ?i1 e ?i2 actúan como instancias de dominio y rango para los que el objectproperty “supervisa” está definido.

Individual specificationfilter

> Este filtro es usado para restringir las instancias de una clase a una instancia o instancias en particular. Este filtro es usado cuando una instancia es encontrada en el mapeo. Por ejemplo: filter (?i1= :I1), en donde la instancia ?i1 es restringida a I1.

(Hasany, Jantan, & Selemat, 2010, p.70)

Construcción de tripletas

Este componente lo que hace es tomar los patrones que le llegan del Componente Mapear a Términos Ontológicos y a partir de estos patrones P, se

generan las tripletas nombradas necesarias para su posterior paso a SPARQL ordenándolas en una estrucutura E. Para lograr esto es necesario adaptar e implementar los algoritmos thepattern_processing, find_class_relation, check_class_for_instance propuestos en Querionto (Hasany, Jantan, &Selemat, 2010, p.71), además del algoritmo de Dijkstra[31]

Partiendo de los patrones identificados en el componente mapear a términos ontológicos, se explicara la obtención de las tripletas por medio del siguiente ejemplo:

Lista de patrones	
RO	categoría
C1	Clase
I1,I2,I3	Instancia
Op1	Objectproperty

Tabla 17. Ejemplo de Lista de patrones.

Pasos:

1.6.2 Aplicar algoritmo Find_class_for_individuals a los patrones encontrados. Este alogoritmo se encarga de encontrar la clase a la que pertenece cada instancia. Por ejemplo las instancias I1, I2 son mapeadas a la clase C2 y la intancia I3 es mapeada a una clase C3.

1.6.3 A continuación aplicamos el algoritmo Find_class_relation, el cual toma las clases encontradas y retorna las aristas que las relacionan entre ellas. Por ejemplo, al tomar las clases C1, C2 y C3, retorna las aristas (C1, C2), (C2, C3).

1.6.4 Con las relaciones entre clases, se crean dos filas de ObjectProperty con las variables ?p1 y ?p2 almacenadas en la estructura E, como se muestra a continuación:

?p1 ?i1 ?i2
?p2 ?i1 ?i3

1.6.5 En la lista de patrones P se encuentra un ObjectProperty (relación) Op1, el algoritmo thepattern_processing retorna todas las tripletas paraObjectProperty Op1 encontradas en el grafo GPOCC. Para este caso encuentra la tripleta C1 Op1 C3 que se encuentra también en la estructura E, ahora a las clases C1 y C3 se le asignan las variables ?i1 e ?i3 y la objectproperty ?p2 es reemplazada por Op1.

[31] El algoritmo de Dijkstraes un algoritmo para la determinación del camino más corto dado un vértice origen al resto de vértices en un grafo con pesos en cada arista.

Ahora la estructura E contiene:

Op1 ?i1 ?i3

?p1 ?i1 ?i2

1.6.6 A las clases C1, C2 y C3 se les aplica el algoritmo check_class_for_instance. Este es el encargado de determinar si estas clases tienen instancias, en caso que alguna clase no tenga instancias, esta clase es remplazada con las clases hijas que tengan instancias. Por ejemplo C1h1 tiene instancias, pero C1h2 no tienen instancias, por lo cual se exploran las clases hijas C1h2h1 y C1h2h2 si están instanciadas, por lo tanto la clase C1 es reemplazada por las clases C1h1, C1h2h1 y C1h2h2.

A las clases C2 y C3 no se les hace el chequeo de clases debido a que provienen del mapeo de instancias.

A partir de los anteriores procesos se obtienen las tripletas RDF, que cumplen con la sintaxis de SPARQL como se muestra a continuación:

Formclasstriples: Como la estructura E contiene tres clases principales, tres tripletas de asociación de clases son formadas:

```
{ {?i1 a :C1h1} union {?i1 a : C1h2h1} union {?i1 a : C1h2h2 } }
{?i2 a C2}
{?i3 a C3}
```

Formindividualfilters: Como I1 e I2 provienen de la misma clase C2, se combinan usando el operador OR en un filtro simple, como se muestra a continuación:

```
Filter(?i2= :I1 || ?i2= :I2)
Filter(?i3 = :I3)
```

FormObjectPropertytriples: Las tripletasformadasparaObjectproperty son:

```
?i1 :Op1 ?i3
?i1 ?p1 ?i2
```

Componente Pasar a SPARQL

Este componente utiliza la salida del componente paso a tripletas RDF para generar una consulta sintácticamente correcta en lenguaje SPARQL, la salida de este componente es una consulta en SPARQL.

Este componente define una plantilla basada en la plantilla dinámica propuesta por Hasany, Jantan, &Selemat (2010) la cual contiene los términos necesarios para que una consulta en SPARQL sea correcta, veamos:

Plantilladinámicaconsulta SPARQL = Prefix + “SELECT” + variables select + “where {” + Relation triple + “ ” + Class association triple + “ ” + Property specification triple + “ “ + Individual specification filter + “ “ + “}”

Figura 18. Plantilla dinámica adaptada.

El prefix define los prefijos para los espacios de nombres, permite asociar una URI[32] a una etiqueta la cual puede ser usada más adelante para acceder a los elementos de la ontología, dicha URI es obtenida automáticamente de la misma ontología.

La sentencia SELECTsirve para definir los datos que deben ser devueltos en la respuesta, es aquí donde se pueden incluir las variables a usar indicando un signo de interrogación “?” al inicio de la variable y separando las diferentes variables con comas “,”.

La sentencia WHERE indica el patrón sobre el que se filtrarán las tripletas del RDF. Es así como este componente recibe la salida del componente paso a tripletas RDF (Relation triple, Classassociation triple, Propertyspecification triple e Individual specificationfilter) y los va adicionando en el orden que se muestra en la Plantilla dinámica de consulta SPARQL generando una consulta en SPARQL sintácticamente correcta.

A continuación mostramos un ejemplo, de la consulta generada:

```
PREFIX p1: <http://ontologia.owl>
SELECT ?i1 ?i2 ?i3 WHERE
{{ {?i1 a p1:C1h1} union {?i1 a p1: C1h2h1} union {?i1 a p1: C1h2h2 } }
{?i2 a p1: C2} {?i3 a p1: C3}. filter (?i2 = p1: I1 || ?i2 = p1: I2).
?i1 p1:Op1 ?i3. ?i1 ?p1 ?i2. }
```

Figura19. Consulta SPARQL generada.

Modulo Buscar Respuesta, componentes

- **Buscar respuesta en ontología**
- **Adaptar respuestas a usuarios**

[32] URI: es una cadena de caracteres corta que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etc.). Normalmente estos recursos son accesibles en una red o sistema. Los URI pueden ser localizadores uniformes de recursos (URL), UniformResourceName (URN), o ambos. (Wikipedia, 2012)

Componente Buscar Respuesta en la ontología

Este componente utiliza la salida del componente Convertir a lenguaje formal y se encarga de extraer información de la ontología ayudándose del gestor de acceso a la ontología, la salida de este componente es una respuesta en formato ontológico a la pregunta del usuario. El componente recibe la consulta SPARQL y se la proporciona a la capa de acceso de JENA que se encarga de consultar la ontología con esta sentencia en SPARQL y traer la respectiva respuesta que se encuentro en la ontología.

Componente adaptar respuesta al usuario

Este componente utiliza la salida del componente buscar respuesta en la ontología para adaptarla a un formato comprensible por el usuario, la salida de este componente es una respuesta en un formato que el usuario pueda entender.

3.4 Modelo de análisis

Desarrollo del prototipo Toulmin.

Con el fin de caracterizar, reconocer y describir la funcionalidad del sistema software. La representación del sistema software desde su funcionalidad (modelo funcional), como objetos (modelo de objetos) y de su comportamiento en el tiempo (modelo dinámico) se lleva a cabo usando el Lenguaje Unificado de Modelado, UML, desde el enfoque de Bruegee&Dutoit (2002) en el análisis y diseño orientado a objetos.

Para representar el Modelo Toulmin como una aplicación de software, es necesario comprenderlo, se requiere una descripción del mismo que sea correcta, completa, consistente y verificable, es decir: el resultado del análisis, llamado modelo de análisis el cual tiene estas características. (Bruegee&Dutoit, 2002, p. 132). En el modelo de análisis se formaliza la especificación del sistema producida por los requerimientos. “La formalización ayuda a identificar áreas de ambigüedad, así como inconsistencias y omisiones en una especificación de sistema.” (Bruegee&Dutoit, 2002, p.132). Esta actividad es iterativa e incremental, como lo es la obtención de requerimientos.

Para conocer los aspectos del problema del usuario debemos conocer el dominio de aplicación. Este incluye terminología especializada del dominio o

referencias a conceptos del dominio. Los requerimientos del dominio son importantes porque entre ellos se encuentran los procesos de trabajo y "los asuntos relevantes para el usuario final, como funcionalidad, facilidad de aprendizaje y uso" (Bruegee&Dutoit, 2002, p. 45).

Con el objetivo de comprender el dominio de aplicación para realizar adecuadamente la tarea que se pretende, se debe tener en cuenta que los procesos de trabajo y las personas cambian a lo largo del tiempo y es crítico que esté disponible a los analistas y a los desarrolladores información sobre el dominio de solución.

El enfoque y la formalización de los requerimientos de los usuarios se encuentran en el modelo de análisis. El modelo de análisis está compuesto, a su vez, por tres modelos:

- Modelo funcional
- Modelo de objetos de análisis
- Modelo dinámico

Modelo funcional: Casos de uso y actores

Los actores representan las entidades externas que interactúan con la aplicación, representación, del Modelo de Toulmin.

El comportamiento del sistema Toulmin desde el punto de vista de un actor se describe por medio de los casos de uso.

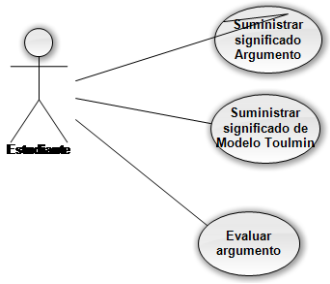


Figura 20. Diagrama de caso de uso

Actores:

- Estudiante

Identificación de casos de uso:

ID	CU1
Nombre	Acceso a servicio de adaptación al dominio :: Iniciar sesión
Actores participantes	Administrador

Condiciones iniciales	Se inicia cuando el administrador desea iniciar sesión para conectarse a SYSTOUL.
Flujo de eventos	1. El administrador Introduce su login y su clave. 2. El administrador envía los datos al sistema. 3. El sistema confirma la validez de los datos introducidos. 4. El sistema inicia una sesión. 5. El sistema muestra las funcionalidades que puede realizar el administrador.
Requerimientos especiales	Los datos introducidos por el administrador en el paso 1 son inválidos: El sistema no inicia sesión. El sistema envía mensaje a usuario notificando que el login y/o la clave son incorrectos.

Tabla 18. CU1

ID	**CU2**
Nombre	Acceso a servicio de adaptación al dominio :: Direccionar Ontología
Actores participantes	Administrador
Condiciones iniciales	Se inicia cuando el administrador desea direccionar una ontología al sistema - SYSTOUL- cargando una ontología de dominio.
Flujo de eventos	1. El administrador ingresa al sistema. 2. El sistema muestra las opciones disponibles para el administrador. 3. El administrador elige la opción direccionar ontología. 4. El sistema solicita la localización de la ontología. 5. El administrador suministra la ruta de localización de la ontología a cargar. 6. El sistema utiliza la ontología para hacer un pre-procesamiento y adapta el sistema al dominio específico proporcionado por la ontología.
Requerimientos especiales	Si la ruta es invalida, el sistema muestra el respectivo mensaje de error.

Tabla 19. CU2.

ID	**CU3**
Nombre	Acceso a servicio de adaptación al dominio :: Finalizar sesión
Actores participantes	Administrador
Condiciones iniciales	Se inicia cuando el administrador desea finalizar su sesión para desconectarse del sistema.
Flujo de eventos	1. El Administrador selecciona la opción “Cerrar Sesión”. 2. El sistema termina la sesión del administrador.

Tabla 20. CU3

ID	**CU4**
Nombre	Suministrar significado
Actores participantes	Estudiante
Condiciones iniciales	El estudiante da clic en la “Ayuda” del menú. El sistema software Toulmin -SYSTOUL- responde presentando una ventana emergente donde muestra las definiciones, de qué es un argumento y qué es el

	Modelo de Toulmin, al estudiante.
Flujo de eventos	1. El estudiante lee las definiciones con el propósito de comprender qué es un argumento y qué es el Modelo de Toulmin y por qué es necesario aprender argumentar y saber si tiene una estructura.
Condiciones de salida	2. Una vez que ha leído y comprendido la definición, el Estudiante cierra la ventana y retorna a la página principal.
Requerimientos especiales	El estudiante debe concientizarse sobre el significado del texto presentado con el fin de comprender el ejercicio antes de cerrar la ventana.

Tabla 21. CU4

ID	**CU5**
Nombre	Evaluar
Actores participantes	Estudiante
Condiciones iniciales	El Estudiante debe conocer primero la definición de qué es un argumento y qué es el Modelo de Toulmin presentado en el caso de uso Suministrar significados. El estudiante activa la función "Argumentar" del menú principal. SYSTOUL responde presentando: • Texto de un enlace de página web el cual representa un argumento propuesto por el sistema Toulmin.
Flujo de eventos	1. El Estudiante lee el argumento presentado por Toulmin, lo comprende y deriva conclusiones, teniendo en cuenta las definiciones presentadas en los casos de uso "Argumento" y "Modelo de Toulmin". 2. El Estudiante debe establecer sus ideas y señala subraya con el mouse las afirmaciones que el considera que representan cada categoría de la estructura del Modelo de Toulmin. Estas corresponden a conjuntos de datos, garantías, términos modales y excepciones y conclusiones organizado y cada uno tiene su color para subrayar. El sistema software Toulmin se encarga de procesar esa información y contrastarla con la ontología de dominio en el momento que da clic en Valorar. Las cajas de texto representan los módulos que componen la estructura del modelo de Toulmin con el fin de lograr una valoración a sus juicios dispuestos en ellas. Esta valoración se realiza por medio de redes neurales, ML –perceptron-, y ontologías.
Condiciones de salida	Al estudiante retorna un mensaje el cual designa el grado de probabilidad que tuvo sus ideas presentadas en las afirmaciones subrayadas respecto el modelo original del argumento representado por el Modelo de Toulmin. La interpretación al valor de probabilidad es qué tanto representa un argumento en la estructura del Modelo de Toulmin.
Requerimientos especiales	Antes de dar clic en Valorar, revisar que exista información en todos lo módulos y haber realizado un análisis sobre los componentes del modelo para su debida representación.

Tabla 22. CU5

Modelo de objetos

Los diagramas de clase describen la estructura del sistema. “Las clases son abstracciones que especifican los atributos y comportamientos. Cada objeto tiene una identidad: se puede hacer referencia a él de manera individual y es distinguible con respecto a otros objetos.” (Bruegee&Dutoit, 2002, p. 45).

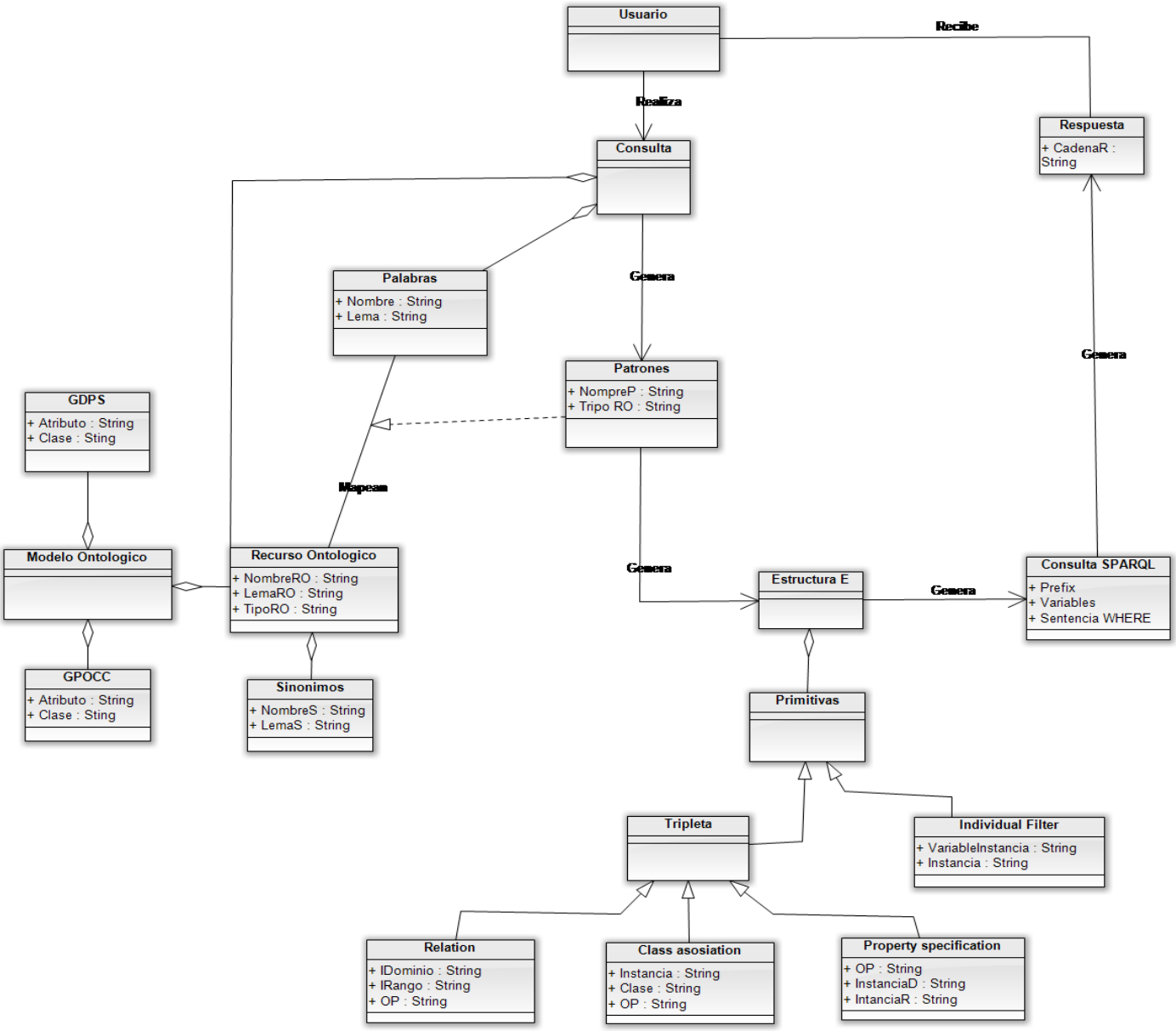


Figura 21. Modelo de objetos

Modelo dinámico

Diagrama de secuencias para el caso de uso EvaluarArgumento.

El comportamiento de un caso de uso entre sus objetos participantes se describe en un diagrama de secuencia, de esta forma se formaliza el comportamiento del sistema. A continuación vemos el diagrama de secuencia para el caso de uso EvaluarArgumento.

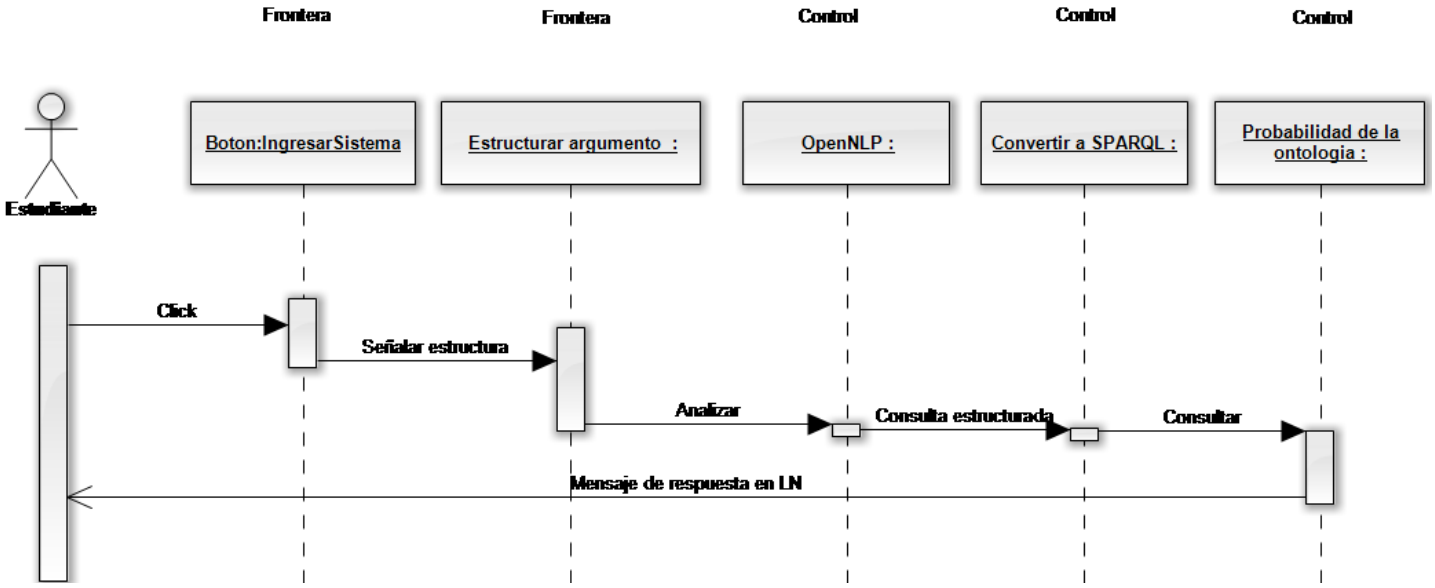


Figura 22. Modelo dinámico

Objeto Identificad	Objeto frontera	Objeto control
Información persistente rastreada por el sistema. • Entidad:Estudiante	Interacción entre actores y sistema. • Ingresar sistema • Estructurar argumento	Tareas realizadas por el usuario y soportadas por el sistema Toulmin. • Control:OpenNLP • Control: Convertir a SPARQL • Probabilidad de ontología

Tabla 23. Objetos

Las columnas de un diagrama de secuencia representan a los objetos que participan en el caso de uso. Las flechas horizontales representan mensajes o

estímulos que son enviados de un objeto a otro. El tiempo avanza verticalmente de arriba hacia abajo.

4. Resultados

El estudiante ingresa a la página web. Para poder utilizar SYSTOUL es necesario leer qué es argumentar y en qué consiste el modelo de Toulmin, para por último leer cómo usar el sistema.

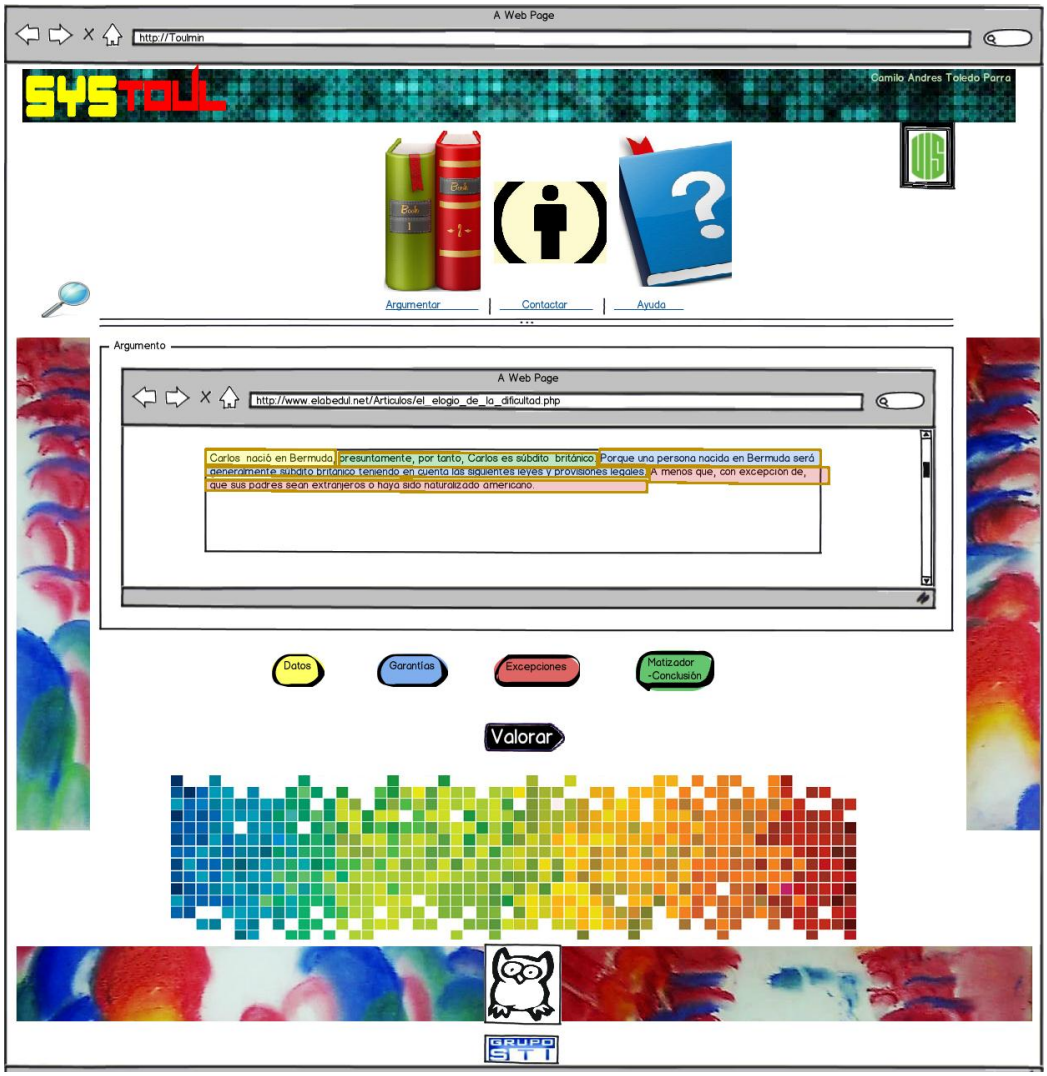
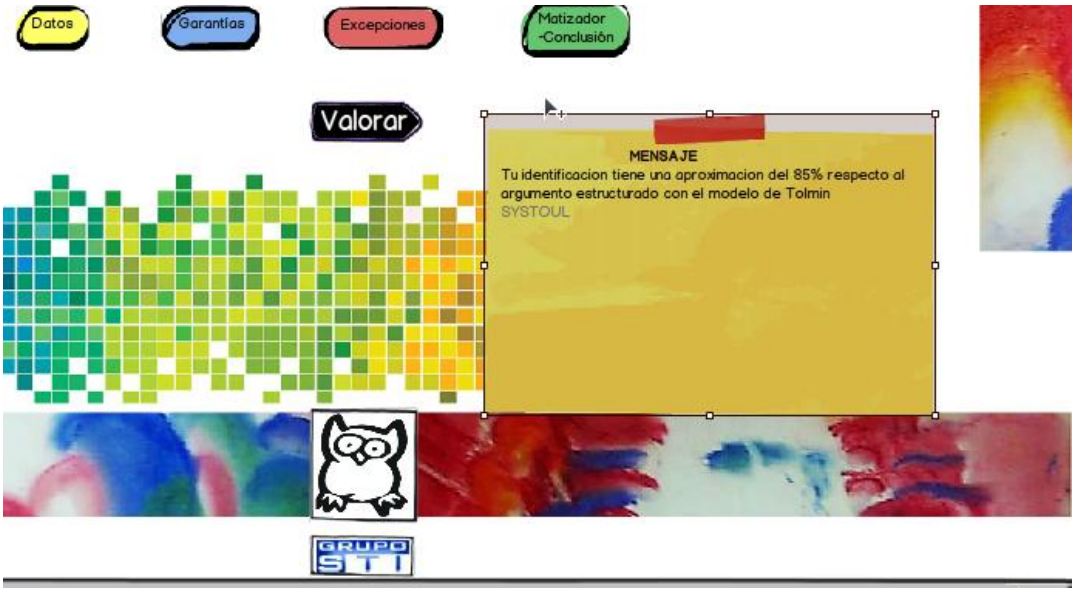


Figura 23. Interfaz del usuario

Después de leer da clic en argumentar y se dispone a identificar (subrayar) con el mouse (ratón) las partes correspondientes al modelo de Toulmin con sus colores respectivos indicados en la interfaz del usuario. Cuando haya realizado toda la identificación da clic en el botón valorar para que SYSTOUL le devuelva qué tan acertado fue la identificación realizada respecto al dominio de la ontología (base



de conocimiento).

Fig ulmin.

5. Recomendaciones

El problema

¿Podemos almacenar datos sin estructuras definidas en una base de datos SQL? si no es posible, entonces ¿Que alternativas existen de almacenar este tipo de datos no estructurados?

¿Como manejar la escalabilidad y el crecimiento de los datos que maneja la ontología a medida que la base de conocimiento aumente de tamaño en el tiempo?

5.1 -BigData-

“Nos debemos concentrar, no en el número de datos, sino, en los datos. Necesitamos entender qué reglas obedecen –los datos- cómo se simbolizan y se comunican y cuál es su relación con el espacio físico y el tiempo.”(Graham, 2011, p.2)

Problema clásico de Big Data: encontrar soluciones en dominios amplios, dinámicos y variables de datos. “Hace miles de años, los datos eran un bien preciado, que residía en localidades selectas y que se debían preservar a toda costa. La transmisión se trataba de una tarea laboriosa que requiere muchas horas de esfuerzo para codificar en un medio adecuado. Pequeños volúmenes de datos en la era de la información nos han llevado a aceptar una posición contraria a la norma, ahora el péndulo esta de vuelta.”(Graham, 2011, p.6)

“La semántica proporciona un aprovechamiento complementario, formula el conocimiento a cerca de los datos en términos de estructuras programables en lugar de funciones verosímiles. Las construcciones semánticas como las ontologías permiten el conocimiento de dominios con grandes cantidades de datos (BigData), que se expresan en un formato procesable por maquinas en términos de clases, propiedades (relaciones) y sus operaciones. La inferencia lógica a través de las clases y las instancias permiten las inconsistencias en los datos y la descripción determinada. Los datos de diferentes descripciones e interpretaciones se pueden combinar eficientemente.” (Graham, 2011, p.2)

Un problema de los BigData es que aprendan las maquinas por si mismas.
Solucionar este problema es fundamental para la gestión del conocimiento

¿Cómo encontrar la información con esta dificultad?

¿Qué pasa si la información textual cambia constantemente?

“Las RDBMS no funcionan mas halla de los 100 TeraBytes de tamaño, esto crea la necesidad de sistemas equivalentes para soportar escalas de PetaBytes.”(Graham, 2011, p.8).

Para febrero de 2012 se estimó que existirían mas de 1.8 ZB[33] (zettabytes) de información. Esto significa que existe un amplio dominio de texto no estructurado en la web. La investigación realizada por Björklund(2011) examina esta expansión y resalta la diferencia de las interfaces para acceder a estos virtualmente; desde los años 70s el modelo arquitectónico de bases de datos era el relacional y el acceso a estos únicamente era desde terminales con interfaz de texto, hoy en día los dispositivos son móviles con interfaces graficas, por lo tanto tienen una mayor disponibilidad para acceder a las bases de datos para modificarla, alterarla o simplemente consultarla. “Es decir que tanto el software como el hardware han cambiado y por lo tanto cambia la forma de almacenar los datos”(Björklund, 2011, p.3).

Muchas bases de datos relacionales pueden proporcionar al mismo tiempo consistencia y disponibilidad y esto resulta bueno para pequeños sistemas. Si este es el objetivo principal, los datos se ajustan al modelo relacional y no hay requisitos sobre el tiempo de actividad, entonces el modelo relacional es una buena opción. “Si hay requisitos sobre el tiempo de actividad o si la información es masiva, podría ser necesario dividir los datos entre varios nodos y crear un compromiso de uno sobre otro (…) Una nota importante es que la inconsistencia no siempre es falta de coherencia, sólo significa que la base de datos no puede garantizar que todos los nodos tienen la misma imagen exacta de los datos en todo momento.” (Björklund, 2011, p. 4).

Es necesario establecer que el termino NoSQL no hace referencia a una base de datos. “NoSQL no denota una base de datos especifica pero puede dividirse en varias y diferentes tipos de bases de datos no relacionales, las cuales tienen características diferentes y son adecuadas para distintos tipos de datos y diferentes situaciones.” (Björklund, 2011, p. 5).

-NoSQL| BIG DATA-

Debido a que la arquitectura tradicional de almacenar, procesar y consultar datos, modelo relacional, tiene la desventaja de la baja escalabilidad, entonces, se debe enfrentar esta necesidad sin bases de datos SQL. “El principal mercado de las bases de datos relacionales RDBMS era el negocio de procesamiento de datos, hoy existen muchos mercados diferentes con requerimientos distintos.”(Björklund, 2011, p.3).

[33] http://www.ai-one.com/ Consultado el 30 de mayo de 2012.

NoSQL presenta una alternativa interesante, eficiente y con mayores ventajas que SQL para almacenar y procesar Big Data. NoSQL tiene la capacidad de brindar escalabilidad en cualquier momento teniendo en cuenta el teorema CAPestablecido por Brewer el cual afirma que es imposible para un sistema de cómputo distribuido garantizar simultáneamente: Consistencia (c), Disponibilidad (A) y Tolerancia a fallos (P)(Pritchett, 2008, p.50). El teorema declara que se puede tener simultáneamente hasta dos de las tres características. “No debe observarse el hecho que tolerar fallos es un sacrificio de la base de datos sino que se transforma información del pasado debido a la dinámica continua de los significados dado a la calidad de relación entre los conceptos y el transcurso del tiempo.”(Helland, 2011, p.40). Estudios recientes sobre el tratamiento de Big Data hechos por Helland (2011) muestran las diferencias de manejar SQL y NoSQL. Según Helland los sistemas de NoSQL surgen debido a que el mundo de los datos está cambiando. El tamaño y la heterogeneidad de los datos significan que las viejas garantías no pueden ser satisfechas.

“El termino NoSQL se uso por primera vez en 1998 para nombrar una base de datos relacional ligera que no trabajaba con una interfaz SQL. A principios de 2009 fue reutilizada por los organizadores de un evento para discutir las bases de datos de código abierto distribuido y era una referencia a la convención de nomenclatura de las tradicionales bases de datos relacionales, tales como MySQL y PostgreSQL (…) La idea no es que las bases de datos relacionales sean malas o erróneas, solo que en algunos casos los modelos relacionales no son suficientes(…) Las tendencias actuales de la computación en nube y el crecimiento de las redes sociales sólo estimularía aun mas la necesidad de grandes almacenes de datos.” (Björklund, 2011, p.3)

En la actualidad hay muchas aplicaciones donde se implementa NoSQL. “BigTable y sus variantes pertenecen a una superclase de los sistemas conocidos como NoSQL que proporcionan un almacenamiento distribuido de datos estructurados y se puede utilizar como equivalentes bases de datos escalables para muchos tipos de datos.”(Graham, 2011, p.8). Escenarios de código abierto como Hadoop son más adecuados para procesar BigData sin SQL. “Todas estas arquitecturas físicas no tienen un conocimiento de la estructura de los datos que están tratando.”(Graham, 2011, p.8).

El ganador del premio ACM A.M. TuringAwardJude Pearl 2012, en Savage(2012) justifica la necesidad de escalabilidad de los sistemas actuales. Perl en 1980 describe que los sistemas expertos de ese entonces son basados en un conjunto de reglas con una capacidad de razonamiento frágil, debido a que

cuando el sistema daba una respuesta incorrecta no era fácil averiguar cual de las reglas era la incorrecta. Argumenta sus investigaciones en la probabilidad que utilizamos los humanos para tomar decisiones sin estar completamente seguros. La probabilidad no es demasiado compleja para los humanos. “Si nosotros lo hacemos, simplemente las computadoras deben ser capaces de hacerlo, simplemente significa que debe haber una aproximación práctica” (Savage, 2012, p.23) dice Pearl. Pearl Trabaja con la teoría de causalidad y de probabilidad, se pregunta ¿Qué pasa si? Si haces A, entonces el resultado debería ser B y puedes considerar que el resultado era probable. “Tenemos un motor causal en nuestras mentes, y lo decora una probabilidad. Así es como los científicos almacenan conocimiento científico. ¿Qué lleva a qué?”[34]. Los estudios de Pearl contribuyen a muchas áreas, en la computación se destacan el aprendizaje automático y el procesamiento del lenguaje natural. Este abismo existente entre la capacidad de manejar pequeñas cantidades de datos y Big Data a través de los sistemas y lenguajes clásicos de bases de datos se hace difícil de salvar en el esplendor de la era de los medios de comunicación e información a nivel mundial. Para aceptar la escalabilidad de usuarios, datos y computación debe aceptarse la tolerancia a fallos y las inconsistencias en los sistemas computacionales, es decir que la naturaleza nos sirve de espejo para reflejar los sistemas no perfectos.

Sin embargo existen sitios y aplicaciones trabajando de manera hibrida según Rys(2011), SQL + NoSQL. Empresas como Facebook, Twitter, Myspace, sitios de e-Commerce y otras grandes que mueven cantidad de información aun continúan escalando en SQL con unas técnicas complejas, rigurosas y con mayores costos comparados con NoSQL.

Para potencializar la herramienta automática y que esta sea capaz de validar cualquier tipo de argumento según la estructura del Modelo de Toulmin y evitar dificultades de actualizaciones de manera manual, es decir que no tenga que ser actualizado cada dominio particular de cada argumento, por ejemplo no tener que realizar una ontología por cada nuevo argumento que se desea incluir al sistema, sino que el sistema sea capaz de reconocer y categorizar las categorías que representan el modelo de Toulmin disponiéndole el argumento via texto, es necesario abordar herramientas como las expuestas para el tratamiento de Big Data. Herramientas como OpenNLP, modelos MAXENT, base de datos NoSQL: *OrientDB,* disposición de grandes cantidades de entidades en una ontología por

[34] Palabras de Pearl.(Savage, 2012)

medio de Freebase y un gran trabajo de entrenamiento de las entidades de las categorías, es posible generar una herramienta poderosa capaz de soportar grandes cantidades de argumentos y dar respuesta a la validación sin tener que estar realizando un trabajo manual, esto es, usando maquinas de aprendizaje sin supervisión.

5.2 Redes neurales

Entre los tipos de aprendizaje existentes se encuentran el aprendizaje supervisado y aprendizaje no supervisado. “Aprendizaje supervisado refiere al algoritmo que se aplica a un conjunto de datos,*x*, pertenecientes a respuestas dadas correctamente, *y,* y su objetivo es realizar predicciones teniendo en cuenta el conjunto de entrenamiento (x,y)”. (Stanford, 2011). En los algoritmos supervisados, para cada uno de los datos *x* existe una respuesta correcta *y*. Un algoritmo de aprendizaje supervisado es el algoritmo deLinealRegression. El trabajo de este algoritmo consiste en aprender de los datos del conjunto de entrenamiento la forma del comportamiento de estos, a esto se le llama la función de hipótesis, *h.* La hipótesis refiere a un comportamiento asociado al conjunto de datos de entrenamiento con el fin que dada una entrada nueva, un dato *x,* sea capaz de predecir una salida *y,* con el mínimo costo. La función de costo está en función de los parámetros que acompañan a la hipótesis *h(x).*

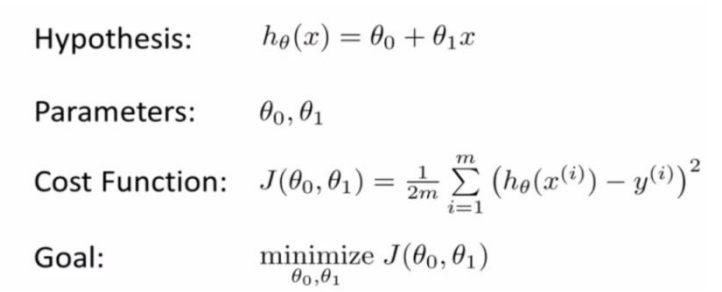


Figura 26. Partes principales de un algoritmo de aprendizaje. Fuente: Coursera. Machine Learning

Pueden existir hipótesis, *h,* que no se ajusten al conjunto de entrenamiento (x,y), esto significa que la función de costo es alta. Para solucionar esta dificultad, se dispone de métodos eficientes para encontrar los valores de los parámetros

necesarios para minimizar la función de costo *j.* GradientDescent es uno de estos métodos, es un algoritmo que encuentra automáticamente los valores de los parámetros, los cuales minimizan la función de costo *j*. Gradientdescent no es exclusivo para maquinas de aprendizaje supervisadas con algoritmos de regresión lineal, también sirve para minimizar otras funciones. Los algoritmos de aprendizaje supervisado representan variables continuas.

Los algoritmos de aprendizaje no supervisados se nos dan datos que no tienen ninguna etiqueta asociada a ellos. Todos se parecen. El algoritmo tiene como objetivo encontrar una estructura en los datos. Existen muchos algoritmos que encuentran diferentes clases de patrones estructurales entre los datos para su clasificación.

Para el problema de clusterización (clustering) dado un conjunto de datos sin etiquetas, tenemos un algoritmo automático que agrupa los datos coherentemente dentro de subconjuntos o clusters. “El algoritmo K-means es el más popular para hacer clusterización”(Stanford, 2011). Básicamente K-means hace de manera iterativa dos cosas:

i. Asignar cluster
ii. Paso del centroide

Para asignar datos a un cluster se basa en un método que utiliza centroides. Estos se comparan respecto a la distancia del conjunto de datos y dependiendo de la distancia del centroide a cada dato se asigna al cluster más cercano los datos.

El segundo paso consiste en ubicar los centroides en espacios donde los clusters tengan el mismo promedio, esto significa que cada cluster debe tener aproximadamente la misma cantidad de datos. Esto se logra computando las diferentes posiciones que pueden tomar los centroides y computar de cada una de estas posiciones, las distancias para asignar al cluster más cercano. Finalizada esta tarea podemos encontrar las mejores posiciones para los centroides, los cuales agrupan de manera proporcional los datos en clusters dado su comportamiento.

Otro algoritmo eficiente es DimensionalityReduction –Reduccion Dimensional-. Las razones para trabajar con DimensionalityReduction son:

- Compresión de datos. Esta característica no solamente comprime los datos, también permite aumentar la velocidad del algoritmo de aprendizaje, ocupa menos memoria computacional, menor espacio en el disco.

- Visualización de datos.

(Stanford, 2011)

La compresión de datos trata problemas como reducir la dimensión, por ejemplo si los datos se encuentran en 3-D este algoritmo tiene la capacidad de volverlo a 2-D o hasta 1-D.
La visualización de los datos también ayuda a desarrollar efectivos algoritmos de aprendizaje. Después de efectuar la tarea de reducir la dimensión es posible graficar los nuevos datos y comprender su comportamiento.
Este tipo de algoritmo resulta muy efectivo cuando tenemos muchas características de grandes dimensiones.

El problema de los BigData se hace evidente también en las maquinas de aprendizaje. Esto se denomina “aprendizaje de maquina a gran escala”(Stanford, 2011). Las investigaciones en el área de machine learning demuestran que “no gana quien tenga el mejor algoritmo, sino quien tenga la mayor cantidad de datos.”(Stanford, 2011). El algoritmo GradientDescent para optimizar la función de costo, minimizar *j,* en algoritmos como Lineal Regression, LogisticRegression y red neural se modifica con el objetivo de que nos sirva para conjuntos grandes de datos, BigData. Esta modificación se llama algoritmo StochasticGradientDescent.
Existen diferentes algoritmos para trabajar con problemas para maquinas de aprendizaje con un numero grande de datos, pero es de resaltar que sea cual sea el algoritmo lo que va a determinar su buen desempeño es la cantidad de datos con la cual esté entrenada la máquina de aprendizaje.

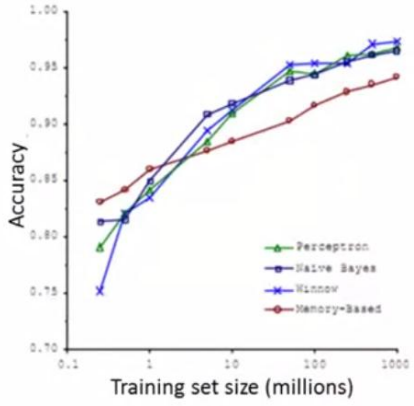


Figura 27. Comparacion de algortimos vs. Precision.
Fuente: Coursera. Machine Learning

EL PERCEPTRON[35]

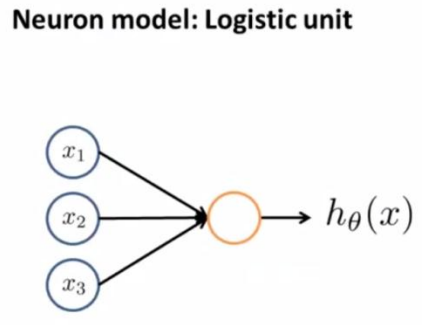


Figura28. Red neural.Fuente: Courserea Machine Learning

OpenNLP es basado en la máquina de aprendizaje llamada el perceptron.

En las maquinas de aprendizaje con supervisión se presenta un problema en el momento que el numero de características, *n,* o categorías que tienen que reconocer es muy grande. Abordar este problema desde la óptica del algoritmo de LogisticRegression va a resultar costoso dado a que este enfoque es para sistemas lineales[36]. Es necesario manejar hipótesis no lineales para abordar el problema. Las redes neurales plantean que con un solo algoritmo se deben aprender todas las tareas por ejemplo en el cerebro "se demuestra que cuando se corta el lazo de la corteza auditiva, entonces se activa, por medio del ojo, osea la vista, un nervio hacia esta corteza auditiva:Esto significa que la corteza auditiva puede aprender viendo, por los ojos." (Stanford, 2011).

La representacion de las redes neurales son:

- Las dendritas: son los cables de entrada a la neurona
- El axon: es el cable de salida de la neurona. Usado para enviar mensajes a otras neuronas

Observemos un modelo simple de una neurona en la imagen Red neural.

Este diagrama representa una simple neurona. La primera capa (capa de entrada) son las dendritas y la segunda capa (capa de salida) es el axon.

[35]Es una red neuronal artificial (RNA) formada por múltiples capas, esto le permite resolver problemas que no son linealmente separables, lo cual es la principal limitación del perceptrón (también llamado perceptrón simple). (Wikipedia, 2012)

[36] La linealidad de un sistema permite a los investigadores hacer ciertas suposiciones matemáticas y aproximaciones, permitiendo un cálculo más sencillo de los resultados. Ya que los sistemas no lineales no son iguales a la suma de sus partes, usualmente son difíciles (o imposibles) de modelar, y sus comportamientos con respecto a una variable dada (por ejemplo, el tiempo) es extremadamente difícil de predecir. (Wikipedia, 2012)

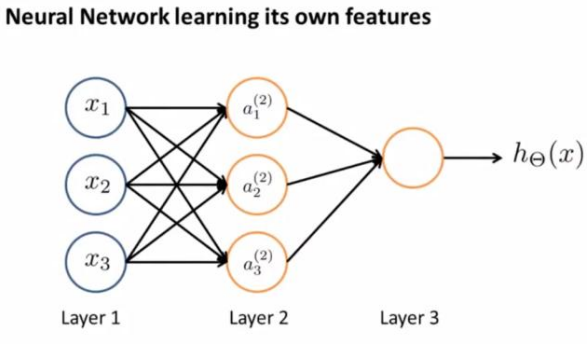


Figura 29. Red neural con capa oculta. Fuente: Courserea Stanford.

Existen redes neurales que entre sus dendritas y axones tienen capas escondidas:

A la activación de la capa escondida desde las entradas (dendritas) se le denomina propagación hacia adelante, se le denomina así debido a que "comienza activando las unidades de la capa de entrada las cuales hacen activar la capa oculta, computa, y luego se propaga y computa la salida de la capa de salida." (Stanford, 2011).

Para clasificación multi-clase, el número de salidas (axones) es mayor a 2, las categorías que debe reconocer representan el número de axones.

"Las redes neurales son los algoritmos de aprendizaje mas poderosos en la actualidad." (Stanford, 2011).La función de costo de las redes neurales es una generalización de la función de costo de LogisticRegression. Para optimizar la función de costo del algoritmo de red neural FowardPropagation, se utiliza el algoritmo de Backpropagation. Este término se debe a que el algoritmo computa los errores de las capas en el sentido de axón hacia las dendritas. Los errores son los deltas que representan las derivadas parciales de la función de costo *j.* Backpropagation realiza la misma función que Forward Propagation pero en sentido contrario. Matemáticamente se minimiza el error cuadrático por medio de Gradientdescent.

¿Cómo saber si la máquina de aprendizaje esta clasificando correctamente las clases?

Cuando la proporción de los ejemplos positivos (1) a negativos (0) está cercana a cualquiera de los dos extremos, es decir si la proporción de ejemplos positivos es de 0.5% respecto a 99.5% de ejemplos negativos o viceversa. Por ejemplo en el caso de la predicción de si un tumor es o no cancerígeno dado su tamaño físico, es posible que exista un margen de error de 0.5%, significando que

la probabilidad que tenga un tumor maligno es muy insignificante. Esto es debido a que el conjunto de datos positivos es muy pequeño en comparación con el conjunto de datos negativos. A este problema se le llama *skewedclasses*–clases sesgadas-. En pocas palabras existe este problema cuando tengo muchos ejemplos de una clase más que de la otra clase.

Se maneja la convención de utilizar y=1 para detectar la presencia de clases raras, para el ejemplo de cáncer utilizamos y=1 si tiene cáncer, y=0 si no tiene cáncer.

Para este tipo de problemas utilizamos las métricas de evaluación:

- Precision
- Recall
- F-Score.

Precision y recall son métricas de evaluación para problemas de clasificación con clases sesgadas.

Estas métricas consisten en realizar una operación dada las observaciones del comportamiento de la máquina de aprendizaje. Continuando con el ejemplo de la predicción de cáncer o no se define precision y recall, dado un conjunto de entrenamiento –train set-:

- Precision: De todos los pacientes donde se predijo *y*=1, es decir que tienen cáncer, que fracción tiene cáncer realmente?
- Recall: De todos los pacientes que actualmente tienen cáncer, que fracción detecto correctamente la presencia de cáncer?

A continuación se presenta la forma grafica de representar los datos con el fin de obtener las formulas generales para precison y recall:

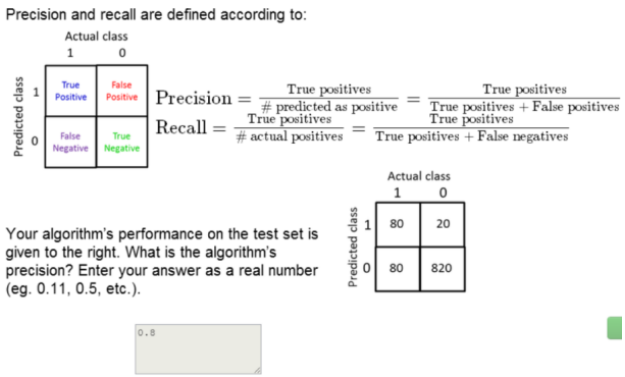


Figura 30. Precision & Recall. . Fuente: Courserea Stanford.

Si deseamos aumentar cualquiera de las dos características vamos a observar que son inversamente proporcionales. Si deseamos aumentar precision, por ejemplo, aumentar el umbral de un 50% a un 70%, recall se verá afectado dado a que existe mayor confianza para realizar la predicción, sin embargo este hecho hace que disminuya la brecha de las predicciones. En el caso contrario, aumentamos recall, vamos a obtener una disminución de precision dado a que si disminuimos el umbral de un 50% a un 70% existirá una brecha amplia para predecir, es decir buen recall, pero disminuye precision.

Para solucionar este problema es posible seleccionar un buen umbral de manera automática. Esta solución es adecuada dado a que es recomendable "tener un solo número real como métrica de evaluación para que de esta forma me pueda decir que B es mejor que A." (Stanford, 2011). Es decir, determinar que umbral se comporta mejor para el algoritmo, A o B.

Para esto utilizamos la métrica F-Score. La dificultad de escoger entre dos algoritmos es que no se tiene una métrica, precision y recall. F-Score es una métrica que combina las dos en una misma medida y evalúa los umbrales en el conjunto de datos llamado Cross Validation Set[37]. El umbral escogido debe ser el que haya registrado el más alto nivel de F-Score.

La precisión del algoritmo aumenta simplemente a medida que aumenta el conjunto de entrenamiento.

LogisticRegression para clasificación de múltiples clases: maxent

Actualmente, logisticregression o clasificador de máxima entropía, es el algoritmo más popular y el más usado para los problemas de clasificación, donde la variable que se quiere predecir es discreta. "El enfoque de este clasificador de máxima entropía es reducir al mínimo el consumo de memoria en conjuntos de datos muy grandes, especialmente en documentos matriciales de términos dispersos" (Jurka, 2012, p. 56)

"La regresión logística multinomial o de máxima entropía, históricamente ha sido un fuerte contendiente para la clasificación de texto a través de aprendizaje supervisado en comparación con el algoritmo de Bayes."(Jurka, 2012, p. 56)

El paquete maxent, máxima entropía, utiliza la optimización de SGD –StochasticGradientDescent- con el fin de encontrar los mejores parámetros que se ajustan a la hipótesis *h,* para que la función de costo sea mínima. "El algoritmo de estimación de parámetros SGD proporciona particularmente eficacia cuando se

[37]Machine Learning. Stanford. 2011.

trata con problemas de aprendizaje a gran escala”(Jurka, 2012, p. 57). El paquete maxent sirve para diversas tareas de clasificación, entre las cuales están “la clasificación de texto y de procesamiento de lenguaje natural.”(Jurka, 2012, p. 57)

Este paquete, maxent, se utiliza con el fin de entrenar un conjunto de datos, modelo de entrenamiento, para clasificar los datos de entrada, texto ingresado por el estudiante, en las categorías que tiene la estructura del modelo de Toulmin. Para esto es necesario entrenar el clasificador logisticregression con un número suficiente de argumentos, estos argumentos se dividen en 4 categorías: Datos, Garantías, Matizador + Conclusión y Excepciones. Para entrenar el clasificador se debe dividir el conjunto de entrenamiento en dos: 70% para los datos de entrenamiento y 30% corresponden a los nuevos datos que deben ser clasificados, este último conjunto se llama cross-validation. Esta división se realiza para dar solución a los problemas de las maquinas de aprendizaje. Los problemas que se presentan son:

Overfitting (sobre ajuste) = highvariance (alta varianza)

Underfitting (bajo ajuste)= highbias (sesgo alto)

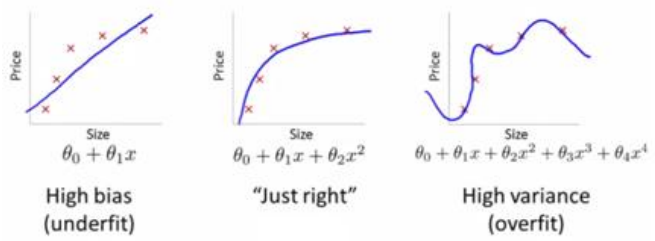


Figura 31.Bias&Variance. Fuente: Courserea Stanford.

El problema de sobre ajuste se presenta cuando la función de hipótesis se ajusta bien a los puntos del conjunto de entrenamiento, sin embargo no representa el comportamiento de estos. Cuando esto sucede se dice que el modelo de entrenamiento tiene una varianza alta. El problema de bajo ajuste, denominado sesgo alto, ocurre cuando la función de hipótesis representa el comportamiento que tienen los datos pero con un costo muy alto debido a que la función de hipótesis no se ajusta a los puntos del conjunto de entrenamiento. Veamos la siguiente grafica, suministrada por el curso de Machine Learning de Stanford, para interpretar estos problemas:

Si corremos el algoritmo de aprendizaje y no hace lo que esperamos, por lo menos todo el tiempo, es porque tenemos un alto sesgo ó alta varianza, en otras palabras: problema de overfitting ó underfitting. Es importante distinguir cual de los dos problemas presenta, dado que esto es crítico para la implementación del algoritmo de aprendizaje.

Estos problemas se deben a una mala selección del modelo. El problema de la selección del modelo significa saber qué grado de polinomio arregla los datos con un mínimo error para los dos conjuntos (conjunto de entrenamiento, 70%; cross-validation, 30%).

Matemáticamente el problema de sesgo alto corresponde a que el conjunto de datos de entrenamiento tanto el conjunto de cross-validation tienen alto porcentaje de error. Cuando ocurre una alta varianza es porque existe un error bajo en el conjunto de entrenamiento pero un error alto en el conjunto cross-validation. Estos errores se deben observar en el mismo grado de polinomio en que se está evaluando.

Para dar solución a los problemas de sesgo y varianza se utiliza una función regularizadora a la función de costo.

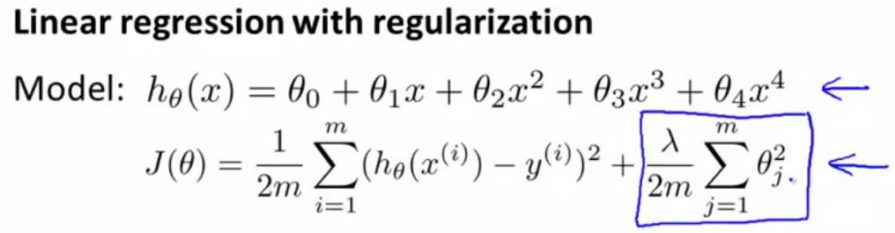


Figura 32. Regresion logistica. Fuente: Courserea Stanford.

5.3 Base de datos

La base de datos Orient DB sirve para guardar cualquier cantidad de datos sin estructura:

ORIENTDB

OrientDB es un motor de base de datos NoSQL de código abierto. Está escrito en Java y puede almacenar más de 150.000 registros por segundo en un hardware compartido. Esta base de datos pertenece al movimiento de NoSQL aunque admite un subconjunto de SQL como lenguaje de consulta.

Existen tecnologías que disponen de entidades almacenados como grafos, es decir sin estructuras fijas, flexibles y con capacidad de manejar Big-Data.

Freebase

La base de conocimiento colaborativa Freebase esta compuesta con Metadatos los cuales describen las entidades. El objetivo de Freebase es crear un recurso global que permita a personas (y máquinas) acceder a la información de forma más eficaz

Freebase es una base de datos para grafos. Esto significa que en vez de usar tablas y llaves en las bases de datos tradicionales, Freebase define datos estructurados como un conjunto de nodos y un conjunto de enlaces que establecen relaciones entre los nodos. Dado a que la estructura de datos no es jerárquica, Freebase puede modelar relaciones más complejas entre elementos individuales en comparación con una convencional base de datos. Es de fuente abierta con licencia de datos estructurados para casi 23 millones de entidades[38]. Una entidad es una persona, un lugar o cosa. Freebase conecta estas entidades por medio de un grafo.

El back-end de la base de datos de Freebase es ***graphD***. Esta escrito en C y corre en maquinas Unix. Esta "procesa consultas de lenguaje para grafos GQL (traducida de las consultas MQL presentadas a través de la API de Freebase)"[39](Freebase, 2012)

A diferencia de las bases de datos relacionales, las cuales usan SQL (structuredquerylanguage) que aceptan consultas y retornan resultados usando protocolos especializados de red, Freebase usa MQL el cual se comunica por peticiones y respuestas estándar HTTP.

MQL es una Interfaz de programación de aplicaciones –API- para hacer consultas programáticas en bases de datos libres. Esto permite incorporar conocimiento desde la base de datos Freebaseen las aplicaciones y sitios web propios.

Actualmente Freebase contiene escritos más de 10 millones de tópicos, mas de 3000 tipos y mas de 30000 propiedades. Freebase esta diseñada para almacenar tipos de datos amorfos que encontramos en nuestra vida diaria. Los tópicos en

[38]http://wiki.freebase.com/wiki/What_is_Freebase%3F

[39] MQL: MetawebQueryLanguage

diferentes dominios como política, educación, negocios, etc, están conectados entre si extendiéndose a través de cualquier combinación de tablas prácticamente.

“Freebase no es sólo un sitio web donde las personas pueden utilizar directamente con sus navegadores, es también un conjunto de servicios web donde sus propias aplicaciones Web se pueden utilizar para lograr cosas que no serían posibles sin datos adicionales o una plataforma de alojamiento donde se puede desarrollar y ejecutar de forma segura sus aplicaciones web, directamente en la infraestructura de servidor propio de Freebase.” (Freebase, 2012)

6. Conclusiones

La dinámica actual en la red y la gran adaptabilidad y expansión de todos sus recursos, hacen necesario que los sistemas software tengan la propiedad de interoperabilidad. Los ambientes actuales en las organizaciones necesitan compartir información y conocimiento de sus actividades y procesos a través del intercambio de datos entre sus sistemas TICs sin la necesidad de solucionar grandes problemas de escalabilidad de datos, usuarios y computación. Esta investigación demuestra la capacidad de interoperabilidad de las herramientas escogidas debido a que a la solución que se planteó en primera instancia no se terminó de llevar a cabo dada la dimensión del problema. Sin embargo, para la solución y automatización del Modelo de Toulmin, se trabajó con las herramientas básicas y se incorporaron otras con gran facilidad permitiendo así un intercambio y flujo de datos entre sus diferentes tareas.

La solución por medio de ontologías nos brinda mayor rapidez para obtener una funcionalidad del software SYSTOUL. Sin embargo este costo es reflejado en su falta de dinamismo y adaptabilidad, es estático, dado que abordando una solución desde el punto de vista de redes neurales, máquinas de aprendizaje sin supervisión, es posible tener una buena maquina de aprendizaje la cual clasifica sus categorías con altos niveles de precisión y sin tener que realizar una ontología para cada dominio, es decir, para cada argumento.

La utilización, creación y reutilización de los modelos de máxima entropía permiten darle potencia a problemas de categorización, reconocimiento, extracción y recuperación de la información de manera óptima tanto para la memoria como para el procesador y en tiempo de respuesta, aplicando los algoritmos de herramientas sobre grandes cantidades de datos. Es decir que el factor de la escalabilidad es irrelevante tanto para la computación, usuarios y la escalabilidad de datos.

SYSTOUL brinda una herramienta para ser utilizada por estudiantes, posiblemente pedagógica, dispuesta a validar cualquier argumento depositado en la base de conocimiento de su sistema como ontologías.

Este artefacto, importante para la teoría de la argumentación, tiene la característica de validar de manera automática la descripción de la estructura del modelo de Toulmin por medio del procesamiento del lenguaje natural. Esto es una característica que lo hace único respecto a otros artefactos tecnológicos del mismo tipo, es decir artefactos que buscan enseñar y validar la capacidad de argumentar.

BIBLIOGRAFIA

ACM. (2009). SIGCHI Curricula for Human-Computer Interaction. Retrieved May 16, 2012 from http://old.sigchi.org/cdg/cdg2.html

Abrahams, B., Condliffe, P., & Zeleznikow, J. (2011). Using an OWL ontology to support legal negotiation about owners corporation disputes. Proceedings of the 13th International Conference on Artificial Intelligence and Law - ICAIL '11, 194-198. New York, New York, USA: ACM Press.

Angadi, U. B., & Venkatesulu, M. (2012). Structural SCOP Superfamily Level Classification Using Unsupervised Machine Learning. IEEE/ACM transactions on computational biology and bioinformatics, 9(2), 601-608.

Argument (2010).In Argument Mapping. Retrieved October 8, 2012, fromhttp://argumentative.sourceforge.net/WhatisArgumentMapping..html

ArgumentMap.(2012). In Wikipedia. Retrieved Octuber 8, 2012, from http://en.wikipedia.org/wiki/Argument_Maps

Baeza-Yates, R. &Ribeiro-Neto, B.(1999) Modern information retrieval vol. 82: Addison-Wesley New York, 1999.

Björklund, A. (2011). NoSQL Database for Software Project Data.

Breis, J. (2008)"Un entorno de integración de ontologías para el desarrollo de sistemas de gestión del conocimiento," .

Brickley, D.Guha., G.(n.d.)"RDF vocabulary description language 1.0: RDF Schema. W3C Recommendation".Retrieved December 15, 2011, from http://www.w3.org/TR/rdf-schema/

Bruegge, B., & Dutoit, A. H. (1999). Object-Oriented Software Engineering. Pittsburgh: Prentice Hall.

Carpenter, G. a., Grossberg, S., & Rosen, D. B. (1991). ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition. Neural Networks, 4(4), 493-504.

Celaá, D. (2010). *Sistema de Respuesta Automática Basado en Recursos Semánticos*.

Codina, L. & Rovira, C. (2006) "La Web semántica," in Tendencias en documentación digital, J. Tramullas, Ed., ed: Trea. Paginas 9-54.

Coursera (2012)Machine Learning. (Virtual Class). Retrieved July 27, 2012, from https://class.coursera.org/ml/lecture/preview/index#close

Fish, G. (2012). Managing contextual artificial neural networks with a service-based mediator, 1.

Freebase, (2012). In Wiki.Freebase. Retrieved April 25, 2012, from http://wiki.freebase.com/wiki/Main_Page

Gamboa Sarmiento, S. C. (2011). *Formación y argumentación: automatización de procesos argumentativos*. Bogota: Universidad Pedagógica Nacional

Gartner. (2012). Ai-one. Retrieved from http://www.ai-one.com/sdk-products/

Gewehr, J. E., Hintermair, V., & Zimmer, R. (2007). AutoSCOP: automated prediction of SCOP classifications using unique pattern-class mappings. Bioinformatics (Oxford, England), 23(10), 1203-10. doi:10.1093/bioinformatics/btm089

Giraldo, J., Mateus, S., & Ruiz, M. (2009). LENGUAJES DE RECUPERACIÓN DE INFORMACIÓN SOBRE LA WEB SEMÁNTICA. *Revista politecnica*, 39-46.

Gómez, J. &Chamizo, J. (2008)"GODO: Generación inteligente de Objetivos para el Descubrimiento de servicios web semánticos," Procesamiento del lenguaje natural, pp. 315-316.

Gottwald, Siegfried, "Many-Valued Logic", The Stanford Encyclopedia of Philosophy (Spring 2010 Edition), Edward N. Zalta (ed.), URL = <http://plato.stanford.edu/archives/spr2010/entries/logic-manyvalued/>. Consultado el 28 de julio de 2012.

Graham, M. J. (2011). The Art of Data Science, 1-12.

Gruber. T,R. (1995). "Toward principles for the design of ontologies used for knowledge sharing," International Journal of Human Computer Studies, vol. 43, pp. 907-928,

Helland, P. (2011). If You Have Too Much Data, then “Good Enough” Is Good Enough.
Communications of the ACM, 40 - 47.

Hong, J., & Linden, G. (2012). Protecting Against Data Breaches;Living with Mistakes. Communications of the ACM, 55(6), 10 - 11.

Interoperabilidad.(2012). In Wikipedia.Retrieved June 22, 2012, from http://es.wikipedia.org/wiki/Interoperabilidad

Jacobson, I., Booch, G., & Rumbaugh, J. (2000) *El proceso unificado de desarrollo de software. Primeraedición*. Addison Wesley: Madrid.

Jurka, T. P. (2012). maxent : An R Package for Low-memory Multinomial Logistic Regression with Support for Semi-automated Text Classification, *4*(June), 56-59.

Kroeker, K. (2011). Biology-Inspired Networking. Communications of the ACM, 54(6), 11- 13.

Kupka(1920). Arte abstracto. Retrieved June 22, 2012, from http://alleragarcia.files.wordpress.com/2011/02/01-kupka-c3a1lvaro-llera-garcc3ada.jpg

Maven (2012).About Maven.Retrieved July 20, 2012, from http://maven.apache.org/download.html

McGuinness, D. & Van Harmelen, F. (2004) "OWL web ontology language overview," W3C recommendation, vol. 10, pp. 2004-03.

Lexicón.(2012) Wikipedia. Retrieved Setember 11, 2012, from http://es.wikipedia.org/wiki/Lexic%C3%B3n

MEN. (2011) Misión y visión del Ministerio de Educación Nacional. Recuperado de: http://www.mineducacion.gov.co/1621/article-89266.html, consultado el 22 de octubre de 2011.

Meadow.,M,C., et al. Text information retrieval systems: Academic Press, 2007.

Moreno, L. &Boronat (1999) "Introducción al procesamiento del lenguaje natural," *Introducción al procesamiento del lenguaje natural.*

Nosofsky, R. M., Little, D. R., & James, T. W. (2012). Activation in the neural network responsible for categorization and recognition reflects parameter changes. Proceedings of the National Academy of Sciences of the United States of America.

Priest, Graham; Tanaka, Koji, "Paraconsistent Logic", The Stanford Encyclopedia of Philosophy (Winter 2004 Edition), Edward N. Zalta (ed.), URL =< http://plato.stanford.edu/archives/win2004/entries/logic-paraconsistent/>
Consultado el 28 de julio de 2012.

Pritchett, D. A. N. (2008). AN ACID ALTERNATIVE. ACM queue, (June 2008).

Ramirez, J., (n.d) “Una versión de lógica paraconsistente proposicional: enfoque semantico”,URL=<http://www.google.com.co/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCwQFjAA&url=http%3A%2F%2Fwww.revistaalternativa.org%2Fnumeros%2Fno8%2Fdoc8%2Fjoseluis8.doc&ei=QfYXUPygJY6m8gSzm4G4Ag&usg=AFQjCNFM6U148J9JoTSpi1Q28DlL2nX1Ag&sig2=5B4SNS_ZiUoob4Z5uQ-j6Q > . Consultadoe el 21 de junio de 2012.

Reed, C., & Rowe, G. (2001). Araucaria : Software for Puzzles in Argument Diagramming and XML, 1-21.

RDF. (2012). In Wikipedia. RetrievedOctober 14, 2012, from http://es.wikipedia.org/wiki/Categor%C3%ADa:Lenguajes_de_descripci%C3%B3n

Rys, M. (2011). Scalable SQL: How do large-scale sites and applications remain SQL-based? Communications of the ACM, 54(6), 48 - 53.
Salton, G. (1983). "Introduction to modern information retrieval," McGraw-Hill.

Savage, N. (2012). Game Changer. Communications of the ACM, 55(6), 22 - 23.

Schwitter, R.&Tilbrook, M.(2006) "Let’s talk in description logic via controlled natural language," Logic and Eng. of Natural Language Semantics, Tokyo, Japan.

Simon, H. A. (1996). The sciences of the artificial (III ed.). Cambridge, Massachusetts: The MIT Press.

Sosa, E. (1997)“Procesamiento del lenguaje natural: revisión del estado actual, bases teóricas y aplicaciones” (parte i). Retrieved april 25, 2012, fromhttp://www.elprofesionaldelainformacion.com/contenidos/1997/enero/procesamiento_del_lenguaje_natural

Techera, C. (2007). *Lavina: Ambiente Web para PLN*.

Telombardi (2011). Introduction to Cytoscape (part 1 of 2) [Video file]. Recuperado desde http://www.youtube.com/watch?v=GGbLcgBAzlc. Consultado el 2 de junio de 2012.

Tomás. (2010). Sistemas de clasificación de preguntas basados en corpus para la Sistemas de clasificaci ´ b ´ usqueda de respuestas Corpus-basedquestionclassification in questionansweringsystems, 155-156.

Toulmin, S. (2007). Los usos de la argumentación. (M. Morrás, & V. Pineda, Trads.) Barcelona: Península.

Verheij, B. (2003). Artificial argument assistants for defeasible argumentation. Artificial Intelligence, 150(1-2), 291-324. doi:10.1016/S0004-3702(03)00107-3

Vicedo González, J.(2004). "La búsqueda de respuestas: Estado actual y perspectivas de futuro," Inteligencia Artificial: revista iberoamericana de inteligencia artificial, vol. 8, pp. 37-56.

ANEXOS

ANEXO A: Instalación

Este tutorial esta escrito usando el siguiente entorno:

- Hardware: Desktop Dell optiblex 380 (Intel G41 Express con ICH7, 4GB de RAM, 300GB HD)
- Sistema operative: Windows 7 de 32 bits.

Para implementar el artefacto se dispuso de las siguientes tecnologías:

Java Development Kit JDK 7.[40]
Netbeans 7.2.[41]
OpenNLP apache 1.5.2 version Incubating.[42]
Maven apache.[43]
PrimeFaces 3.4.[44]

Los repositorios de OpenNLP los implementa Maven. El sistema de construcción de OpenNLP es basado en Maven apache. Maven es un administrador de proyectos software y es una herramienta de compresión. No depende del shell del sistema operativo, este se basa en archivos de configuración xml y clases Java para realizar las tareas. “Es similar en funcionalidad a Apache Ant” (Wikipedia, 2012). Maven se usa para la realización de tareas mecánicas y repetitivas, es un software de automatización de compilación para proyectos Java. La mayor diferencia de Ant respecto a Maven es que el primero es una librería de utilidades y el segundo un framework configurable y altamente extensible. Para construir el proyecto Maven se utiliza POM –Proyecto de Modelo de Objetos-. POM “se trata de una representación XML de un proyecto Maven en un fichero llamado pom.xml.” (Maven, 2012). Esta es la unidad fundamental para trabajar en Maven.
Para instalar Maven es necesario agregar una variable de entorno: M2_HOME. La variable de entorno se agrega desde la terminal de comandos o desde las propiedades de Mi PC, opciones avanzadas, clic en variables de entorno. Se agregan las siguientes variables del sistema:

[40] http://www.oracle.com/technetwork/java/javase/downloads/index.html
[41] http://netbeans.org/
[42] http://opennlp.apache.org/cgi-bin/download.cgi
[43] http://maven.apache.org/download.html
[44] http://primefaces.org/downloads.html

1. Nombre: JAVA_HOME; valor de variable: el path donde se encuentra instalado el JDK apuntando a la carpeta jdr6. (omita este paso si ya existe esta variable).
2. Agregue al valor de la variable Path la ruta de la carpeta bin de la carpeta JDK1.6.0.
3. Agregue una nueva variable de entorno. Nombre: M2_HOME; valor de variable: path donde esta instalado apache Maven.
4. Agregue al valor de la variable Path la ruta de la carpeta bin de la carpeta de apache Maven.

A continuación verificamos el estado de la instalación de maven. Desde el cmd de Windows escribimos: *mvn –version.* Como resultado positivo de la instalación:

Después de verificar la instalación correcta de Maven . Para usar OpenNLP se debe agregar un repositorio Maven al archivo pom.xml. [45] Desde Netbeans, buscamos la carpeta ProjectFiles y actualizamos el archivo pom.xml. Para poder usar los repositorios debemos también incluir en el pom la dependencia respectiva de las herramientas de OpenNLP y agregarla al pom.xml. Las dependencias cumplen una función similar a la de las librerías.

OpenNLP ofrece modelos pre construidos en, estos modelos son dependientes del idioma y funcionan bien si coincide el texto de entrada con el idioma. Estos modelos se pueden descargar[46] o se pueden crear modelos propios a partir de nuestros propios datos anotados.

Para colocar en funcionamiento la ML debemos tener un modelo y luego implementamos la herramienta SentenceDetector. Los modelos son archivos .bin, por ejemplo el modelo *en-sent.bin.* Este archivo (modelo) debe guardarse en una carpeta preferiblemente que este dentro del proyecto Maven. Este modelo se guarda en la dirección: C:\Users\Qamilo\Documents\NetBeansProjects**ToulminBrain\src\main\webapp\bin**.

[45] El repositorio se encuentra en: http://opennlp.sourceforge.net/README.html
[46] http://opennlp.sourceforge.net/models-1.5/

Para instanciar una herramienta, Sentence Detector, debe estar cargado primero el modelo de sentencia. Para SYSTOUL, este se carga desde Netbeans desde el Package controlador. Después de cargar el modelo se crean instancias de la herramienta, SentenceDetectorME. La herramienta da como salida una matriz de datos, donde para este ejemplo seria una matriz de cadenas, donde cada cadena representa una frase.

Netbeans, el requisito que este pide es estar instalado el Java Development Kit, JDK. El componente para JSF PimeFaces es usado para facilitar la creación de aplicaciones web. PrimeFaces 3.4 trabaja con JSF2.

ANEXO B: Método de búsqueda y registro de información

1. Software usado: CytoscapeVersion 2.8.3. Se descargó de la página http://www.cytoscape.org/download.html.

Para instalar Cytoscape es necesario ingresar información personal (Nombre, organización y correo electrónico) y aceptar los términos.
El software es GNU (licencia libre) y depende de la arquitectura que tenga descarga el software de 64 bits o 32 bits. El asistente de instalación le guía los pasos para que funcione correctamente en su equipo.

2. Archivos de las representaciones ontologicas:
   Metamodelo de conocimiento 1, ver archivo Onto1.cys
   Metamodelo de conocimiento 2, ver archivo file2.cys

3. Método científico

   i. Descargué Cytoscape desde la pagina oficial: http://www.cytoscape.org/download.html
   ii. Ejecute el siguientearchivo Cytoscape_2_8_3_windows_32Bit
   iii. Para saber como funcionan las herramientas de Cytoscape, a traves de youtube estudie los siguientes videos:

http://www.youtube.com/watch?v=GGbLcgBAzlc
http://www.youtube.com/watch?v=9zZOzyO9YhM&feature=relmfu

   iv. Profundización teórica:
      a. sobre ontologías informáticas. Cytoscape trabaja con grafos para aprovechar las ontologías que se quieren representar. Librousado: Semantic Web-Based Information Systems: A state of the art applications. Capitulos I, II y V.
      b. Sobre lógica paraconsistente.
      c. Sobre redes neurales.
      d. Sobre escalabilidad de sistemas de bases de datos[47]
   v. Establecimiento del primer modelo de ontología general en Cytoscape.

[47] Para obtener información de punta recurrí a material bibliográfico electrónico de revistas y artículos de los dos últimos años referenciados de otros artículos o también investigando y comunicándome con desarrolladores, investigadores y temas relacionados al Big Data por medio de Twitter y Facebook.

vi. Revisar conceptos, relaciones y necesidades. Si se necesitan profundizar conceptos pasar a iv.
vii. Modificar o ampliar el modelo de ontología general en Cytoscape.
viii. Guardar el archivo .cys en el disco.
ix. Estructurar la investigacion por medio de informe final.

ANEXO C: Maxent

Las siguientes tablas muestran detalles individuales de cada paquete, cada clase abstracta y cada clase.

	Interface MaxentModel
Constructor	public double[] eval(String[] context);
Método	double[] eval(java.lang.String[] context)
Explicación	Evalua un contexto
Parámetro	Context
Return	Vector outcomes, este es un vector de probabilidades de los predicados contextuales que deben ser evaluados.

Tabla 24MaxentModel

	Interface EventStream
Constructor	public Event(String outcome, String[] context) ;
Método	Event nextEvent()
Explicación	Objeto que entrega un stream de eventos entrenados para el procedimiento de GIS.
Parámetro	Outcome, context
Return	Objeto evento del siguiente EventStream

Tabla 25EventStream

	Class GIS
Constructor	public GIS();
Método	public static GISModeltrainModel(EventStreameventStream)
Explicación	Entrena un modelo usando el algoritmo GIS, asumiendo 100 iteraciones.
Parámetro	eventStream
Return	Modelo recién entrenado. Se puede utilizar inmediatamente o guardar en el disco usando un objeto: opennlp.maxent.io.GISModelWriter

Tabla 26. Clase GIS

	Class GISModel
Constructor	public GISModel(Context[] params, String[] predLabels, String[] outcomeNames, intcorrectionConstant, double correctionParam) ;

Método	double[] eval(java.lang.String[] context)
Explicación	Modelo de máxima entropía que ha sido entrenado usando el procedimiento Iterativo Generalizado de eScala –GIS-
Parámetro	Params; predLabels; outcomeNames; correctionConstant; correctionParam.
Return	

Tabla 27.ClaseGISModel

	Class Context
Constructor	public Context(int[] outcomePattern, double[] parameters);
Método	public int[] getOutcomes() public double[] getParameters()
Explicación	Clase que asocial un parámetro de valor real o valor esperado con un predicado contextual particular o caracteristica
Parámetro	outcomePattern – Vector de outcomes, resultados, para los cuales existen parámetros de este contexto. parameters – Parametros para los resultados especifacados.
Return	• getOutcomes: vector de outcome para los cuales existen parámetros para ese contexto. • getParameters: vector de parámetros de los outcomes de ese contexto.

Tabla 28. Clase Context

	Interface DataIndexer
Constructor	Class OnePassDataIndexer
Método	int[][] getContexts()
Explicación	Objeto que comprime los eventos en memoria y realiza la selección de características.
Parámetro	parameters – Parametros para los resultados especifacados.
Return	Devuelve el conjunto de predicados que se ven en cada evento de este contexto.

Tabla 29. Dataindexer

	Class OnePassDataIndexer
Constructor	public OnePassDataIndexer(EventStreameventStream, int cutoff)
Método	protected java.util.List index(gnu.trove.TLinkedList events, gnu.trove.TObjectIntHashMappredicateIndex)

Explicación	Indizador de datos del modelo MaxEnt. Proporciona un índice de numero entero único para cada uno de los predicados.
Parámetro	• eventStream: Un Event[] que contiene la lista de todos los eventos observados en los datos de entrenamiento. • Cutoff: El número mínimo de veces que debe estar un predicado observado con el fin de ser incluidos en el modelo.
Return	

Tabla 30. Clase OnePassDataIndexer