UNIVERSIDAD DEL BÍO-BÍO

FACULTAD DE CIENCIAS EMPRESARIALES

DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN

"Modelo experimental para la adquisición de conocimiento y toma de decisiones a partir de información extraída desde la web"

TESIS PARA OPTAR AL GRADO DE

MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN.

AUTOR:

San Martín Villarroel, Eduardo

PROFESOR GUÍA:

Rubio Manzano, Clemente

Chillán, Octubre de 2013

Resumen

En la web podemos encontrar una enorme cantidad de datos de diversa índole. Muchos de estos datos se desaprovechan, cuando pudiesen ser usados por empresas u otros organismos para obtener conocimiento de algún área y apoyar sus procesos de toma de decisiones.

En este trabajo planteamos que es posible hacer uso de esos datos si contamos con los mecanismos apropiados de extracción, procesamiento y consulta. Para tal efecto, proponemos una integración de cuatro tecnologías (Programación lógica-difusa para realizar consultas flexibles, XML para almacenamiento y transporte de datos, harvesting como mecanismo de extracción, y Modelo Relacional para complementar la Base de Conocimiento). Esta integración tecnológica deriva en una arquitectura multicapa que permite hacer una implementación práctica para efectuar consultas clásicas y flexibles sobre bases de conocimiento construidas con información extraída desde la web.

Abstract

We can find on the Web a huge amount of data of various kinds. Much of this data is wasted, when they could be used by companies or other entities to gain knowledge of some area and support their decision-making processes.

In this research we argue that it is possible to use these data if we have the appropriate mechanisms of extraction, processing and consulting. For this purpose, we propose an integration of four technologies (fuzzy-logic programming for flexible queries, XML for storage and data transport, jarvesting as extraction mechanism and relational model to complement the Knowledge Base). This technological integration results in a multilayer architecture that allows a practical implementation for perform conventional and flexible querying on knowledge bases built with information extracted from web.

Agradecimientos

A Dios primeramente por permitirme llegar hasta aquí, a mi familia, pues el tiempo dedicado a este emprendimiento, por derecho propio les pertenece, a Clemente Rubio Manzano por sus consejos y orientación y al equipo académico quienes con sus cátedras han inspirado este trabajo y otros proyectos personales. Muchas gracias a todos ellos.

Tabla de contenido

Índice de ilustraciones

Índice de tablas

Organización de la tesis

La tesis se organiza en 6 capítulos. En general estos conducen el tema desde una introducción y el establecimiento de objetivos, pasando por una revisión de la literatura en busca de antecedentes, técnicas y tecnologías candidatas para ser incorporadas en la solución. Continuamos con el planteamiento de la solución y un caso de estudio donde se aplica el modelo para concluir finalmente con conclusiones y reflexiones.

1. *Capítulo I:* En este capítulo se realiza una reflexión de la importancia de la información en nuestras vidas y como esta resulta fundamental en todas las decisiones que tomamos, ya sea en el ámbito personal o profesional. Presentamos la información como un elemento que evoluciona desde los simples datos hasta el conocimiento y presentamos en este sentido, a los datos como "materia prima", enfatizando el hecho de que la web es un repositorio gigantesco de datos y por ende los encontramos en abundancia. En esta reflexión puntualizamos en que muchos de estos datos no se aprovechan, dejando abierto el tema sobre qué hacer para utilizarlos y con esto, apoyar la toma de decisiones.
2. *Capítulo II:* En este capítulo se establece la hipótesis, objetivos y alcance de la investigación explicando además la metodología de trabajo que se utilizará.
3. *Capítulo III*: En este capítulo se presenta el estado del arte considerando dos grandes tareas. Primero, una planificación de una Revisión Sistemática de la Literatura (RSL) que nos proporcionará orden y claridad en los criterios de búsqueda y selección del material bibliográfico y segundo, una síntesis de la RSL que en concreto es el resultado de la RSL, donde es posible encontrar una serie de elementos (herramientas, tecnologías, técnicas, etc.) revisados en el contexto de nuestra investigación con el propósito de contextualizar y determinar con base en el estudio de estos elementos, cuáles de ellos pueden ser incluidos o considerados en nuestra solución.
4. *Capítulo IV:* En este capítulo se presenta evidencia de trabajos similares y resultados preliminares de acuerdo a las interrogantes de investigación planteadas en la RSL. Se responden las interrogantes señaladas con base en los elementos estudiados en el estado del arte. Las respuestas a estas interrogantes permiten finalmente obtener claridad del contexto en el que se encuadra una posible solución y los elementos que debieran componer dicha solución.
5. *Capítulo V*: En este capítulo se presenta una arquitectura multicapa como una aproximación a la solución frente a la necesidad de utilizar la información existente en la web para apoyar los procesos de toma de decisiones. Junto con la explicación de la arquitectura, se presenta una implementación de esta con un caso de estudio relacionado con la venta de vehículos usados.

6. *Capítulo VI*: Finalmente en este capítulo se presentan las conclusiones y reflexiones de la investigación realizada considerando el grado en que se han cumplido los objetivos propuestos como así también los principales aportes que de aquí se desprendan. Además, identificamos las posibilidades de extensión del modelo mediante trabajos futuros.

CAPÍTULO I: Preliminares

1.1. Introducción

Frases como "El conocimiento es poder" y "La información: divisa del futuro" de alguna u otra manera nos vaticinan lo cada vez más indispensable que será contar con información oportuna y de calidad en todas las áreas en que nos desenvolvemos, al punto en que prácticamente las personas, hoy por hoy, somos "un producto de información", "elementos cuantificables", "registros" o "perfiles". No es que perdamos humanidad, sino que en un mundo donde la globalización es altísima y va en aumento, nosotros las personas, somos parte de la información. Comenzamos siendo únicamente quienes producimos y manejamos la información pero hoy además, somos parte de ella.

Esta investigación se fundamenta sobre la base empírica de que la información cumple un rol fundamental en la vida de las personas. Esta premisa se hace evidente al observar que en las muchas actividades que una persona puede realizar, se deja entrever un elemento en común, la toma de decisiones. Ya sea que se trate de actividades domésticas, laborales, de mucha o poca importancia, lo cierto es que a diario las personas tomamos decisiones de diversa índole, como la marca del vehículo que se desea comprar, el lugar donde salir el próximo fin de semana, escoger entre un perfume o un chocolate para regalar, etc. Cada vez que las personas nos enfrentamos a una decisión que tomar, esta, en la mayoría de las situaciones resulta simple si se cuenta con información en cantidad y calidad suficiente, y muy por el contrario, cuando no se cuenta con la información necesaria, hasta una decisión simple puede tornarse en una tarea difícil. En efecto, existe una relación directa entre información y certidumbre, en consecuencia, a mayor información mayor certeza o certidumbre, o lo que es igual, a mayor información, menor incertidumbre.

La Real Academia Española define certidumbre como "Conocimiento seguro y claro de algo" y también como "Firme adhesión de la mente a algo conocible, sin temor de errar". En este sentido, si la decisión en cuestión se toma con absolutamente toda la información necesaria, sería razonable pensar en un cien por ciento de probabilidad de éxito para esa decisión, y además podríamos decir que la decisión fue tomada "sin temor a errar" o con plena certidumbre, pero en realidad se trata de una

situación ideal y utópica puesto que muchas decisiones se toman considerando hechos futuros de los cuales no es posible tener certeza sino pronósticos e información aproximada que en alguna medida reduce la incertidumbre pero no la elimina.

La simple decisión de elegir un regalo resulta simple si se conoce con certeza el presupuesto, los gustos o preferencias de la persona para quien se comprará el regalo, las tendencias o modas del momento, lugares donde comprar, ofertas, etc. Cada uno de estos elementos aporta significativamente con certidumbre y por ende, suma puntos a la probabilidad de éxito para con la decisión tomada. No obstante, si se reduce la información hasta ser eliminada, la incertidumbre aumentará considerablemente y la decisión, más que un problema estratégico será cuestión de azar.

Precisamente los sistemas de información (SI) (1) (2) tienen origen en la necesidad de contar con información para tomar buenas decisiones, de hecho, un sistema de información se puede definir como "conjunto de elementos interrelacionados con el propósito de prestar atención a las demandas de información de una organización, para elevar el nivel de conocimientos que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones". Algunas preguntas interesantes entonces serían: ¿de dónde obtener la información necesaria para apoyar el proceso de toma de decisiones?, ¿cómo obtener dicha información? ¿Son los SI las únicas fuentes de información para los procesos de toma de decisión empresariales?. Pues bien, es sobre esta última interrogante que se fundamenta este trabajo, en donde se propone un modelo y los mecanismos que permitan la obtención de información significativa para la toma de decisiones desde una fuente de información prácticamente inagotable.

1.2. Planteamiento

En el apartado anterior se asienta el principio de que la información es vital en cualquier proceso de toma de decisiones y por ende resulta necesario comprender qué es la información y cómo llegamos a obtenerla. La Figura 1 muestra el proceso evolutivo de la información, desde un estado básico y atómico hasta otros más complejos y elaborados.

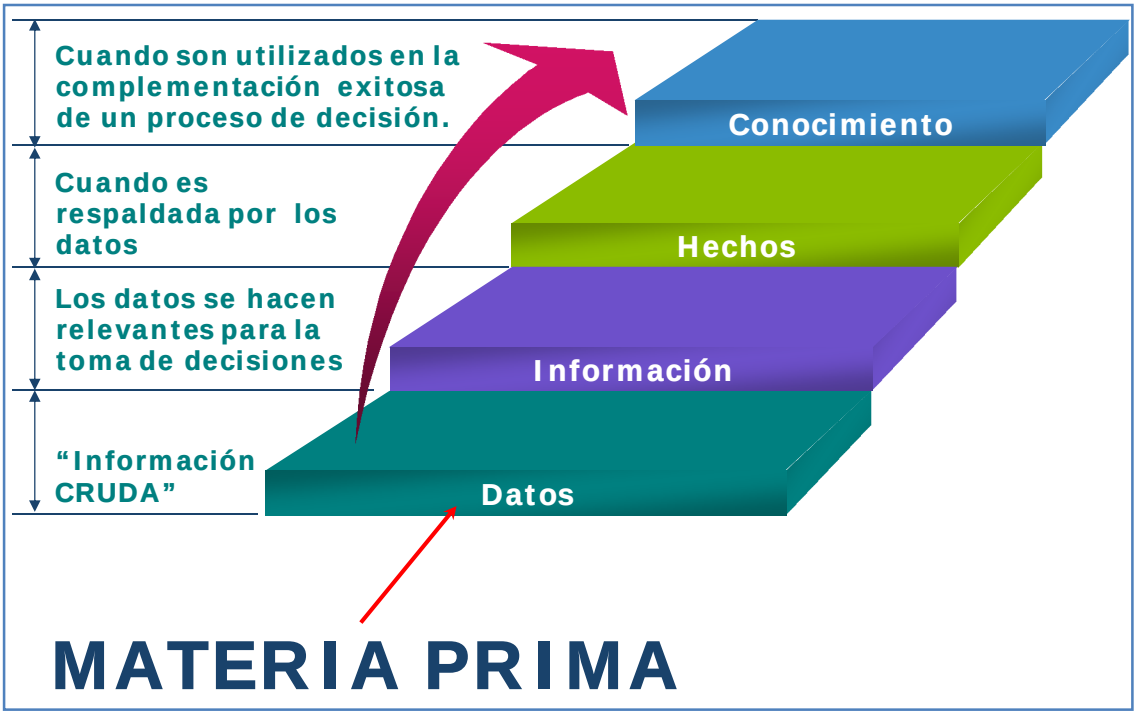


Figura 1. Evolución de la información

La figura 1 señala de manera simple como la información tiene sus comienzos en los datos para luego terminar siendo conocimiento. No obstante, lo relevante aquí es comprender que todo parte con los simples datos. Sin datos "No hay Información", sin información no hay certidumbre y como se señaló anteriormente, el proceso de toma de decisiones se traduce en un asunto de azar.

Los datos tradicionalmente se obtienen de sistemas de información operacionales encargados de automatizar tareas altamente repetitivas los que a su vez registran sus operaciones en bases de datos. Estas bases de datos actúan como repositorios para diversos sistemas de información de nivel superior (SIs estratégicos) (3) diseñados para apoyar la toma de decisiones al interior de una empresa. Estos sistemas de Inteligencia de Negocios tienen sentido únicamente si cuentan con datos sobre los cuales puedan operar. Las bases de datos de los sistemas de inteligencias de negocios y dataminig (4) (5) no son para nada despreciables en cuanto a su tamaño, sin embargo, la cantidad de datos que almacenan puede resultar marginal si se compara con la cantidad de datos que se alojan y transmiten en internet.

Nuestra forma de ver y hacer nuestras actividades ha cambiado con la globalización y sus nuevos paradigmas, y como consecuencia, internet cambió desde su concepción inicial. El comercio, por ejemplo, se ha tornado un tanto más complejo sin importar la perspectiva bajo la cual se mire. El desarrollo de Internet y la integración de este con el comercio tradicional, dio origen al ya bien conocido comercio electrónico, el cual en un momento parecía ser un lujo optativo, pero hoy es posible observar como los nuevos paradigmas llegados de la mano de la globalización, demanda que sea

simplemente una necesidad si lo que se desea es permanecer vigente y competitivo. Esto último, simplemente da origen a un concepto de negocios y economía diferente, la Economía de Internet (6). Una característica importante de esta economía (también conocidas como la economía digital y sociedad de la información) es el cambio a lo intangible. La creación y manipulación de contenidos en soporte electrónico se ha convertido en una importante fuente de valor económico (6) y este desarrollo se acelera aun más en la Web 2.0 y Web 3.0 (7). Este movimiento hacia lo intangible afecta a todos los sectores y actividades transformando profundamente las relaciones económicas, las interacciones, la estructura de las empresas, la forma en que los mercados se organizan y la manera en que las transacciones se llevan a cabo.

Estos nuevos escenarios inundan cada día internet con más y más información. La información se encuentra ahí, pero ¿cómo consultarla?. Solo esta pregunta constituye toda un área de investigación y constituye otro aspecto fundamental de este trabajo.

La teoría económica clásica no suele abordar el tema de información, esta es considerada usualmente como un derivado exclusivo de la reducción de incertidumbre. En la economía de Internet, sin embargo, la información es al mismo tiempo una producción de activos y un bien. La información es transable, es una divisa por la cual muchos están dispuestos a pagar y existe en abundancia en internet en forma de datos no homogéneos y semiestructurados, pero nuevamente, ¿cómo conseguirla entonces?, ¿Cómo sintetizarla en información útil?. Estas interrogantes sustentan la necesidad de construir mecanismos que permitan usar apropiadamente los datos existentes en internet para elaborar información útil.

Bajo la perspectiva de que la información también constituye un bien, no solo las empresas que comercian intangibles pueden lucrar con el "producto información". El desarrollo de Internet también permite que las empresas que comercian productos tangibles puedan integrar el comercio electrónico dando origen a nuevos modelos de negocios que solamente existen gracias al crecimiento del comercio electrónico (Herramienta de e-commerce, hosting, servicios de pagos en línea, servicios de entrega, etc.). En consecuencia, estos servicios proporcionan más información para la red, donde cada empresa, cada nuevo servicio, cada uno de nosotros proporciona a diario mucha información que se registra "inerte" en la web. Si a esto se le suma otro gran desarrollo tecnológico como es el crecimiento e impacto de las redes móviles (8) (9) en conjunto con las redes sociales (10) (11), internet, como repositorio de información crece aun más.

Para dimensionar el tamaño actual de internet, se presenta en la Tabla 1, cifras que resumen el escenario.

Cifras de internet	
Servidores	760 millones
Sitios Web	162,662 millones
Redes sociales	> 6
Usuarios de Facebook	900 millones
Gigabytes trasferidos entre 2009 – 2020	35 trillones

Tabla 1. Cifras de internet (http://royal.pingdom.com/, http://www.ceis-itesm.com/)

Existen datos suficientes en internet como para sintetizar información que apoye un sin número de decisiones en diversas áreas. Los datos se encuentran ahí, "Solo hay que torturarlos hasta que confiesen" (12), pero el principal inconveniente para usar esta información en un contexto de toma de decisiones es que la información que se encuentra y circula en internet no puede ser interpretada por las máquinas ya que internet fue diseñada y concebida para que sean las personas las que interpreten la información. De inmediato aparece otra dificultad, esta radica en que la cantidad de información en internet es tan grande que cualquier intento de implementar algún mecanismo manual para la extracción e interpretación sería infructuoso. En este sentido, disponer de la información existente en la web para apoyar algún proceso de toma de decisiones, queda supeditado a las capacidades o mecanismos existentes para automatizar en alguna medida los procesos de extracción y procesamiento.

El apartado anterior tiene como propósito contextualizar sobre como la globalización ha cambiado la forma en que operan muchas áreas como el comercio, para dar lugar al comercio electrónico y como este en conjunto con otras áreas relacionadas y las personas terminan siendo productoras de información en la web, dando origen a un enorme repositorio de información con características particulares que potencialmente puede ser usado para dar respuestas a muchas interrogantes, lo cual derivaría en beneficios concretos si sostenemos que la información es la divisa del futuro.

Con este escenario como punto de partida, se pretende elaborar un modelo para la extracción y procesamiento de datos desde la web que incorpore el uso de Agentes Inteligentes u otros mecanismos, con objeto de construir información significativa con la cual se pueda apoyar el proceso de toma de decisiones en algún área específica, siendo el Marketing y el comercio electrónico las áreas seleccionadas para ilustrar su implementación.

La idea de efectuar tal investigación se origina al concebir internet como un enorme y complejo sistema de información (13) (14) o también repositorio de datos para muchos (15), con lo que potencialmente se podría responder un sinfín de interrogantes de manera similar a como lo hace un sistema de información tradicional o consultando una base de datos tradicional, aun cuando se requiera

previamente un proceso de extracción y otro de preprocesamiento. Un sistema de stock y ventas por ejemplo, permite responder de manera muy simple interrogantes del tipo ¿cuáles son los productos más vendidos o los de mayor valor?, y mediante una o más consultas a la base de datos, las respuestas a las interrogantes serán conseguidas.

Así por ejemplo, un usuario de internet puede formularse la pregunta: ¿Qué modelo de vehículos usados se vende más? o ¿qué concesionario de vehículos tiene el mejor bono de descuento para un mismo vehículo?. La respuesta se puede conseguir mediante el intuitivo procedimiento de revisar sistemáticamente los sitios web de uno o varios sitios de venta de vehículos con objeto de examinar y comparar cada publicación encontrada ahí. En el caso del mejor bono, revisar los sitios web de cada concesionario y registrar los bonos ofrecidos para el vehículo en particular, para luego compararlos. Situaciones como estas se estudiarán para buscar mecanismos de extracción, procesamiento y consulta que hagan posible concebir la idea de un sistema prototipo que permita responder interrogantes como las señaladas anteriormente, sin perjuicio de que las consultas requieran de procesamientos previos, lo que evidentemente impedirá realizar consultas directamente sobre internet, debiendo hacerlas sobre un nuevo repositorio que será el resultado de los procesos de extracción y preprocesamiento.

Un repositorio de datos tan grande, particular, complejo y no homogéneo como internet, demanda la existencia de mecanismos que permitan de manera autónoma, la extracción y procesamiento de datos para la construcción de información y conocimiento. Es en este contexto que se estudiarán los robots o harvesters como organismos capaces de realizar una serie de tareas referente a la extracción de datos en la web.

Como mecanismos de consulta, aparte de lo que ofrece un repositorio como una base de datos relacional, se estudiaran mecanismos que permitan además el manejo de la imprecisión, de modo tal que sea posible responder a preguntas como ¿Cuáles son los vehículos oscuros a la venta? o ¿Qué vehículos son baratos?. El estudio de los mecanismos de extracción, procesamiento y consulta proporcionarán los elementos necesarios que integrarán el modelo propuesto: "*Modelo experimental para la adquisición de conocimiento y toma de decisiones a partir de información extraída desde la web*".

Para contextualizar sobre los temas planteados y acotar la dimensión y proyección de la investigación, se realizará un estudio reducido del estado del arte considerando temáticas como: uso de información de la web para la toma de decisiones: uso directo o extracción, uso de robots y agentes inteligentes y web semántica.

1.3. Justificación

Internet ha evolucionado desde sus inicios complementándose con el desarrollo de otras tecnologías y tendencias. Facebook por ejemplo, a julio del 2011 alcanza la ostentosa cifra de 700 millones de usuarios inscritos y a la fecha ya son más de 900 millones. La telefonía móvil por otro lado, ha experimentado tal desarrollo que los dispositivos móviles actuales son verdaderos computadores personales que permiten entre otras cosas el procesamiento de datos, navegación en internet y localización (GPS). Este fuerte crecimiento le permite a los usuarios de las redes sociales y de internet en términos generales no estar sujetos al computador con conexión a internet alámbrica, sino que le otorga amplia movilidad y al mismo tiempo conectividad. Como si eso fuera poco, se suma la posibilidad de localización mediante la masiva difusión de teléfonos inteligentes (smartphones) con GPS incorporado, lo que posibilita interactuar en redes sociales basadas en la localización (16) de una manera distinta a la conocida hasta hace algún tiempo.

La convergencia del conjunto de tecnologías apunta a una cosa: conectividad móvil. En un escenario como este, la información que se necesita recolectar ya no proviene solo de una "página web", hay al menos 4 situaciones que considerar (Ver Figura 2):

1. Sitios web
2. Internet móvil
3. Redes Sociales
4. Redes Sociales mediante Internet móvil

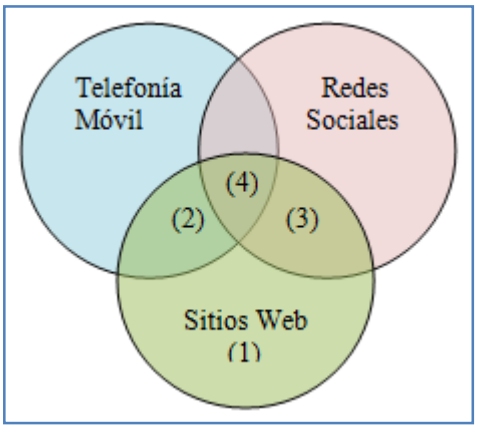


Figura 2. Integración de fuentes de información

Lo anterior potencia el concepto de supersistema, refiriéndonos al gran repositorio de datos donde existe mucha información potencialmente útil, el asunto es, que se trata de un escenario heterogéneo , donde la información se encuentra "ahí", pero no siempre resulta simple su extracción y síntesis.

Mucha de esta información podría obtenerse en tiempo real mientras el usuario visita y navega por un sitio web. También es posible de obtener otro tipo de información examinando los rastros que deja un internauta tras navegar y dejar información personal o de otro tipo.

En este escenario, los mecanismos manuales de extracción de datos no ofrecen buenos resultados, estos carecen de precisión y profundidad. Se necesita de mecanismos que puedan rastrear, monitorear y descubrir información valiosa de entre los datos. También resulta deseable examinar y monitorear el comportamiento de los usuarios en Internet, como si se tratara de un objeto valioso vigilado por una serie de cámaras. En este sentido los agentes y robots pueden ser las cámaras necesarias para la captura de información en tiempo real. No obstante, los pequeños robot de tipo software también resultan tremendamente útiles para el rastreo, recolección y procesamiento de información, que es donde recaerá nuestro interés en esta investigación.

A pesar de que, como señalamos, existe una vasta cantidad de datos en este megarepositorio llamado Internet, lamentablemente y en términos muy generales, estos se desaprovechan, pues los datos están, pero de momento no contamos con mecanismos difundidos ampliamente que faciliten el

uso eficiente de estos en los procesos de toma decisiones. Existe una suerte de caos en el mar de información que dificulta esta tarea, no obstante, son muchas las propuestas que intentan dar solución parcial o total al problema, algunas más inmediatas que otras, otras más o menos definitivas, pero la mayoría dependen de cambios profundos de la arquitectura de internet. El concepto de súper sistema de información, por ejemplo, se basa en la web 3.0 o también llamada web semántica (17) (18), la cual propone una serie de cambios radicales que sin duda favorecen la consulta y extracción automática de datos, pero esta implementación no es ni simple ni rápida y únicamente es posible si se redefine internet. Esto sitúa el concepto de "internet como sistema de información" en un lugar poco accesible, al menos como solución inmediata, y por otra parte, justifica la propuesta de mecanismos que permitan hacer uso de la información existente en internet sin una completa reingeniería de la misma, aun cuando pueda resultar un poco menos eficiente que la web semántica.

Mientras la web semántica no sea una realidad concreta y tangible, será necesario proponer mecanismos que permitan extraer datos desde la web para dotarlos de cierto significado y forma, y de esta manera utilizarlos. En consecuencia, el planteamiento final se traduce en la búsqueda de mecanismos que permitan la extracción, procesamiento y consulta directa, con lo cual será posible apoyar los procesos de toma de decisiones dependiendo del contexto y cantidad de datos.

CAPÍTULO II: Hipótesis, objetivos, alcance y metodología de trabajo

2.1. Hipótesis y objetivos

Internet hoy en día es más que un canal de información como se concibió durante el boom de las "punto com", hoy converge junto con otras tecnologías en un sistema aun mayor que recibe a diario una cantidad abismante de información de parte de empresas, internautas y redes sociales, formando un enorme y heterogéneo repositorio de datos al que llamaremos en el contexto de esta investigación Web Data System (WDS).

Se reconoce la dificultad de tratar con la información proveniente de WDS debido a su representación variada y no homogénea que escapa a cualquier estructura y modelos de representación formal de la información. Sin embargo, la información sigue estando ahí de forma no estructurada o semiestructurada en algunos casos, esperando ser extraída, procesada y representada apropiadamente

para su utilización. Suponemos entonces que si se dispone de mecanismos apropiados de extracción, representación y consulta sobre la información proveniente de WDS, será posible interactuar con ella con la posibilidad de construir nueva información y con ella apoyar la toma de decisiones sobre un dominio específico.

2.1.1. Objetivo principal

Proponer un modelo para la extracción, procesamiento y consulta de información semiestructurada proveniente de la web para apoyar la toma de decisiones en un área de conocimiento específico.

2.1.2. Objetivos específicos

a) *Objetivo específico 1*: Proponer mecanismos que permitan la extracción y procesamiento de información semiestructurada existente en la web.
b) *Objetivo específico 2*: Proponer el o los mecanismos necesarios para interactuar con los datos extraídos desde la web con el propósito de descubrir o construir nueva información útil para el proceso de toma de decisiones sobre un área de conocimiento específico.

2.2. Alcance de la investigación

La investigación considera una revisión bibliográfica reducida que permita establecer el punto de partida de las acciones, metodologías y técnicas propuestas para dar cumplimiento a los objetivos planteados. La revisión en cuestión contemplará investigaciones primarias que involucren de manera conjunta e independiente los siguientes temas: uso de información de la web (uso directo o extracción), robots y agentes inteligentes en la web, web semántica y ontologías en la web.

La investigación se considerará satisfactoria si como resultado final es posible proponer un modelo que incorpore entre sus elementos robots que permitan extraer y procesar datos desde la web para luego interactuar con ellos intentando crear o descubrir nueva información. El modelo estará pensado inicialmente para producir información que pueda servir de apoyo en la toma de decisiones vinculadas a un dominio específico.

Se espera probar el modelo con un conjunto de datos reducidos y alguna aproximación inicial de lo que serán los robots o harvester, para de esta forma, determinar su aplicabilidad. No obstante, la generalización del modelo para extracción y procesamiento de datos de la web correspondientes a cualquier área será parte de una investigación posterior en la que se espera además construir en su totalidad los agentes de software encargados de la extracción y procesamiento de datos en conjunto con otros artefactos de software que pudiesen ser necesarios.

2.3. Metodología de trabajo

Una revisión parcial de la literatura (realizada mediante una Revisión Sistemática de la Literatura (RSL)) es fundamental en la investigación, pues por una parte establece el punto de partida de la investigación respecto de los avances e investigaciones realizadas sobre el tema en cuestión u otros relacionados, y por otra, proporcionará elementos de investigación que puedan ser empleados como piezas del modelo que se desea proponer. En este sentido, se entenderá como pieza de investigación, sistemas experimentales, mecanismos, desarrollos, tecnologías, técnicas de extracción, trabajos similares, etc., es decir, elementos que puedan ser utilizados parcial o totalmente como parte del modelo.

En la selección y valoración de los elementos de interés se aplicará en primera instancia los criterios de selección e inclusión establecidos en la propia RSL.

La investigación tendrá un enfoque autoreflexivo que estará sustentado por la realización de un estudio de caso (en el contexto de la extracción y procesamiento de datos para la toma de decisiones en materias de marketing, particularmente relacionado con venta de vehículos usados) e investigación-acción. Por consiguiente, el segundo aspecto que incidirá en la inclusión de las piezas de investigación corresponde a las posibilidades que tenga cada elemento para integrar el modelo y contribuir con las acciones probatorias de la teoría, es decir, de la hipótesis o el cumplimiento de los objetivos de forma práctica.

CAPÍTULO III: Estado del arte

El modelo para la extracción, procesamiento y consulta que se busca desarrollar, depende de varios elementos tecnológicos como agentes de software, web semántica, ontologías e internet por mencionar algunos, por lo que es posible referirse al modelo como una integración tecnológica. En consecuencia, es necesario describir y comprender el funcionamiento de estos elementos para proveer de un contexto a la futura investigación.

Para dar inicio a esta investigación, en el apartado siguiente se presenta la planificación de la Revisión Sistemática de la Literatura (Dado que no es el fin de este la RSL, se presenta una RSL reducida) que será parte fundamental de este trabajo.

3.1. Planificación de la revisión sistemática de la literatura

En esta primera etapa se consideran aspectos fundamentales para realizar una buena revisión. Los cuales constituyen los cimientos de la investigación que se pretende realizar. Estos "cimientos" evitarán que se caiga en ambigüedades y subjetividades que pueden derivar en una revisión con resultados difusos o poco certeros, es decir, inclusión de literatura que tal vez "no aporta", o por el contrario, la exclusión de material que pudo contribuir, pero que por un mal o inexistente protocolo de revisión no fue considerado.

3.1.1. Identificación de la necesidad de revisión

Como se planteó en la hipótesis, el objeto de esta investigación es suponer que internet puede ser usada como una gran fuente de datos de la cual, mediante los mecanismos apropiados, es posible extraer información para luego ser utilizada como apoyo a la toma de decisiones o para descubrir nueva información.

Una de las principales dificultades para utilizar tal caudal de información que deriva finalmente en un desaprovechamiento de la misma, es la escases de "mecanismos apropiados" para usar inteligentemente los datos que confluyen en internet.

La naturaleza heterogénea y sin semántica de los datos dificulta la interpretación automatizada y la consulta directa de estos. En este contexto se cree que la incorporación de agentes de software en la elaboración de un modelo para la extracción y procesamiento de datos desde la web puede aumentar el valor intrínseco de los datos en internet y por ende, reducir el desaprovechamiento de los mismos. Además, se advierte un especial interés en la creación de nuevos negocios basados en el valor de la información elaborada con datos desde la web.

Bajo este contexto se establece lo siguiente:

3.1.1.1. Objetivo

Explorar y resumir la información existente con relación a la implementación de agentes de software y otros mecanismos vinculados a la extracción de datos desde la Web actual, como también información pertinente a la evolución y proyección de internet.

3.1.1.2. Interrogantes de investigación

1. ¿Cómo es la arquitectura y funcionamiento de un agente de software, particularmente la de un agente inteligente?
2. ¿Cómo es la arquitectura de internet actualmente y cómo se vislumbra en el futuro?

3. ¿De qué manera los agentes de software o agentes inteligentes pueden ser utilizados para la extracción de datos desde la web?
4. ¿Hay otros mecanismos aparte de los AI que puedan ser utilizados eficientemente en la extracción y procesamiento de datos desde la web?

3.1.1.3. Recursos disponibles

1. Bibliotecas digitales: IEEE, ACM, SCIELO y similares
2. Libros digitales
3. Buscadores científicos (SCIRUS, Google Scholar, CiteSeer.IST, GoldRush)
4. Otros:
    a. http://gredos.usal.es/jspui/
    b. http://ilpubs.stanford.edu:8090/

3.1.2. Definición del protocolo de búsqueda

3.1.2.1. Términos

Id	Término
1	Bot
2	WebBot, Web Bot
3	Semantic Web, Web 3
4	Intelligent Agent, IA
5	Web Ontologies
6	Web Architecture
7	Web Agent
8	Web mining
9	Mining
10	Web harvesting
11	Linguistic Query
12	Fuzzy logic

Tabla 2. Términos de búsqueda

La tabla 2 muestra una lista de términos que serán utilizados en la búsqueda.

3.1.2.2. Combinaciones

Id	Combinación
1	(Bot | WebBot | Web Bot | Intelligent Agent | IA) ^ (Mining | Web Mining)
2	(Bot | WebBot | Web Bot | Intelligent Agent | IA) ^ (Semantic Web | Web 3)
3	(Bot | WebBot | Web Bot | Intelligent Agent | IA) ^ (Web Ontologies)
4	(Bot | WebBot | Web Bot | Intelligent Agent | IA) ^ (Web harvesting)
5	Linguistic Query ^ Fuzzy logic

Tabla 3. Combinaciones de términos

La tabla 3, a diferencia de la tabla 2, muestra una lista de de las combinaciones entre términos que serán utilizadas en la búsqueda.

3.1.2.3. Estrategia de búsqueda

1. *Publicaciones digitales:* La búsqueda se realizará mediante buscadores científicos como SCIRUS, GoogleScholar, CiteSeer.IST y GoldRush. Alguno de los artículos (muchos de ellos) no son públicos. Para acceder a la información, se realizar una búsqueda por autor intentando encontrar sitios personales donde hayan publicaciones abiertas, de lo contrario tratar de contactar al autor para que comparta su artículo y así poder referenciarlo.

2. *Bibliotecas electrónicas:* Estas bibliotecas generalmente permiten buscar palabras o frases con relación a las temáticas del libro, materia, título o autor. Aun cuando la biblioteca no sea completamente abierta para la lectura de libros, siempre ofrece la posibilidad de buscar y conocer al menos los títulos disponibles. De esa forma si se encuentra un texto que aporte a la investigación que se está realizando se puede intentar buscar el texto completo o parte de él en google libros o sitios alternativos como www.flaxz.com.

3. Con relación a los artículos que se hayan seleccionado, resulta conveniente considerar sus referencias, ya que estas pueden conducir a otros artículos que pueden aportar a la investigación.

3.1.3. Definición de un protocolo de revisión

3.1.3.1. Normas de revisión

Lo primero será ponderar adecuadamente las combinaciones para establecer en función de ellas una categorización de documentos como así también las reglas de inclusión y exclusión. La Tabla 4 permite determinar si un documento se debe incluir o excluir conforme a distintos parámetros establecidos. La primera columna tiene como única función indexar el documento bajo análisis, en la siguiente columna bajo el nombre "Términos buscados" se registran simplemente los términos buscados. La columna 3 indica de qué combinación provienen los términos. Las últimas 5 columnas bajo el título "Cantidad térm. encontrados en (Lugares):" registra la cantidad de ocurrencias (para cada término, es decir, T1, T2, .., Tn) y el lugar donde se encontraron los términos, es decir, Título (T), Conclusión (C), Abstratc (A), Cuerpo (B) y Resumen (R). Cada uno de estos lugares tiene una ponderación o peso como sigue:

- Título: 4
- Conclusión: 3
- Abstract: 2
- Cuerpo: 0,5
- Referencias: 1

El peso otorgado a las ocurrencias encontradas en el cuerpo del documento se establece como significativamente menor pues, en esta parte del documento, la más extensa por lo demás, pueden encontrarse varias ocurrencias de un término sin que esto necesariamente signifique que el documento trata el tema a investigar con la profundidad que se requiere, ya que estas ocurrencias pueden ser únicamente referencias o mención al tema de interés, y el tema del documento no corresponda en su mayoría al tema de interés. Si las ocurrencias en cambio tienen lugar en el resumen y título del documento, la situación resulta muy distinta, pues sugiere un puntaje mayor para el documento ya que al encontrarse los términos en el título o resumen, esto indica que el documento en un porcentaje mayor se encuentra relacionado con las temáticas buscadas. De esta forma los pesos permiten valorar un documento mediante la siguiente fórmula de valoración (VALDOC).

$$VALDOC = (T_1t+T_1c+T_1a+T_1b+T_1r) + (T_2t+T_2c+T_2a+T_2b+T_2r) +\ldots+ (T_nt+T_nc+T_na+T_nb+T_nr)$$

Donde:

- T_it es la cantidad de términos encontrados en el Título
- T_ic es la cantidad de términos encontrados en la Conclusión
- T_ia es la cantidad de términos encontrados en el Resumen
- T_ib es la cantidad de términos encontrados en el Cuerpo
- T_ir es la cantidad de términos encontrados en las Referencias
- Pt es la Ponderación para el título
- Pc es la Ponderación para la Conclusión
- Pa es la Ponderación para el Resumen
- Pb es la Ponderación para el Cuerpo
- Pr es la Ponderación para las Referencias

Id	Términos buscados	Comb. utilizada	Cantidad térm. encontrados en (Lugar):				
			T	C	A	B	R
	T1						
	T2						
	Tn						

Tabla 4. Registro de búsqueda y ponderación

A modo de ejemplo, mostramos en las Tablas 5 y 6 la búsqueda de la combinación “Linguistic Query ^ Fuzzy logic” en dos documentos, A y B respectivamente obteniéndose los siguientes resultados:

Id	ID término buscado	Comb. utilizada	Cantidad térm. encontrados en (Lugar):				
			T	C	A	B	R
	11	5	1	1	2	4	1
	12		0	0	0	5	0

Tabla 5. Ponderación para documento A

- Tt =4; Tc =3; Ta =2; Tb =0,5; Tr =1 (Común para todos los términos)

  Para término 1:

- T_1t =1; T_1c =1; T_1a =2; T_1b =4; T_1r =1

  Para término 2:

- T_2t =0; T_2c =0; T_2a =0; T_2b =5; T_2r =0

VALDOC(Documento A) = (1*4+1*3+2*2+4*0,5+1*1) + (0*4+0*3+0*2+5*0,5+0*1)

VALDOC(Documento A) = 14 + 2,5 = 16,5

Id	Términos buscados	Comb. utilizada	Cantidad térm. encontrados en (Lugar):				
			T	C	A	B	R
	11	5	0	0	1	4	0
	12		1	1	1	8	1

Tabla 6. Ponderación para documento B

Con:

- Tt =4; Tc =3; Ta =2; Tb =0,5; Tr =1 (Común para todos los términos)

Para término 1:

- T_1t =0; T_1c =0; T_1a =1; T_1b =4; T_1r =0

Para término 2:

- T_2t =1; T_2c =1; T_2a =1; T_2b =8; T_2r =1

VALDOC(Documento B) = (0*4+0*3+1*2+4*0,5+0*1) + (1*4+1*3+1*2+8*0,5+1*1)

VALDOC(Documento B) = 4 + 14 = 18

De acuerdo a los criterios de inclusión y exclusión establecidos en el punto siguiente, ambos documentos se deben incluir.

Para contar las ocurrencias de términos durante la valoración de documentos se usarán las herramientas básicas de búsqueda intra-documento, las que dependen del formato del documento (html, pdf, word o ps) y la aplicación que se use para leer el documento. De esta manera, luego de seleccionar los documentos según criterio de selección establecido, en una etapa posterior se efectuará la lectura de estos según ponderación calculada. Este mecanismo propuesto supone una demanda de trabajo adicional para el investigador, pero aleja la subjetividad de él. No obstante no es un método que haya sido probado, razón por la cual pueda requerir ajuste.

3.1.3.2. Criterios de inclusión

VALDOC >= 8

3.1.3.3. Criterios de exclusión

VALDOC < 8

3.1.4. Estrategia de síntesis

La información será sintetizada de acuerdo a los siguientes temas:

1. Funcionamiento y arquitectura de agentes de software, en especial de los agentes inteligentes
2. Arquitectura actual y futura de internet
3. Potencialidad de los agentes inteligentes o agentes de software en la extracción y procesamiento de información desde la WEB.
4. Mecanismos y tecnologías que puedan ser utilizadas eficientemente en la extracción y procesamiento de información desde la WEB
5. Mecanismos para manejar la imprecisión en las consultas a base de datos (Linguistic Query)

3.2. Síntesis de la RSL

3.2.1. Internet

En general cuando las personas hablan de "Internet", en realidad están hablando de la World Wide Web. La Web es la parte más interesante, más innovadora, más visible y de mayor crecimiento de la Internet. En gran medida, el crecimiento explosivo de la Web ha sido lo que ha impulsado el enorme interés en el Internet. Muchas de las personas usan la expresión "navegar por Internet", cuando en realidad están hablando sobre el uso de la World Wide Web o simplemente el uso de la web.

A internet se le conoce también como la red de redes, la más grande de todas por lo demás y con algunas características muy especiales. A diferencia del resto de las redes, Internet no está dirigida ni administrada por nadie en particular, puesto que es una agrupación de miles de redes autónomas de computadores (19), es decir, independiente en cuanto a su administración interna se refiere, las que cooperan entre sí para integrar internet. No obstante, debe existir alguna especie de gestión y coordinación que controle la diversidad (20). Esta gestión y coordinación tiene relación, entre otras tareas, con la dirección de dominios y tráfico para que la información pueda pasar entre las redes. Dado que se trata de una enorme red donde es posible advertir variedad y diversidad en todo el sentido de la palabra (contenido, conexión, protocolos, etc.) resulta indispensable la presencia de organismos que permitan llevar a acuerdo la forma en que se deben realizar ciertas cosas estableciendo procedimientos y normas para cada una de ellas. Así por ejemplo si hablamos de Internet estamos obligados a referirnos a W3C (World Wide Web Consortium), dado que se trata del organismo de

regulación más importante del mundo en relación a Internet (20) que entre otras cosas a normado la arquitectura de la misma (21).

Con relación a internet, algunos de los aspectos importantes a destacar en el contexto de este trabajo son los siguientes:

a) *El corazón de Internet:* A pesar de que Internet está formada también por una diversidad de organizaciones y redes que se preocupan de monitorear, prestar servicios varios y controlar distintos aspectos de su funcionamiento, son las redes locales las que constituyen el corazón de Internet. Estas redes se pueden encontrar en las empresas privadas, universidades, agencias gubernamentales, servicios en línea, etc.

b) *Interconexión de las redes:* Las redes están conectadas en una variedad de maneras. Por razones de eficacia, las redes locales se unen a redes regionales y mediante varios tipos de líneas arrendadas se conectan las redes locales y regionales. Las líneas arrendadas que conectan las redes pueden ser tan simple como una sola línea telefónica o tan compleja como un cable de fibra óptica con enlaces de microondas y transmisiones por satélite.

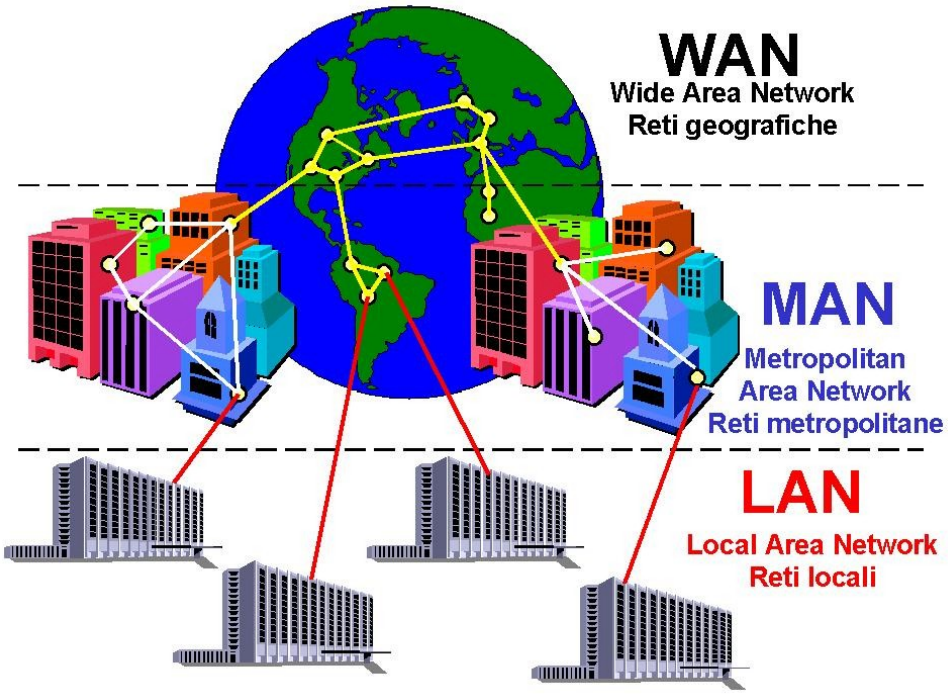


Figura 3. Organización de internet como red (22)

Se dice que las redes locales son el corazón de Internet pues son ellas las que proporcionan acceso a los computadores, servidores de archivos y servicios ubicados en lugares distantes. Las Wan por otra parte, utilizan enlaces de datos suministrados por el proveedor de internet (ISP) para acceder a internet y conectar los sitios de la empresa entre sí, con las otras entidades, con servicios internos e incluso con usuarios remotos.

La arquitectura que da soporte a Internet es la denominada cliente/servidor, esto es, unos computadores almacenan la información (los computadores servidores) y otros acceden a ella (los computadores clientes). Esta comunicación e intercambio de información entre los computadores conectados a la red es posible gracias a protocolos de comunicación, normas y estándares comunes.

Existen diversos protocolos que sirven para ofrecer una gran variedad de servicios en Internet, donde los más utilizados son la transferencia de archivos, el correo electrónico y el protocolo de la Web entre otros.

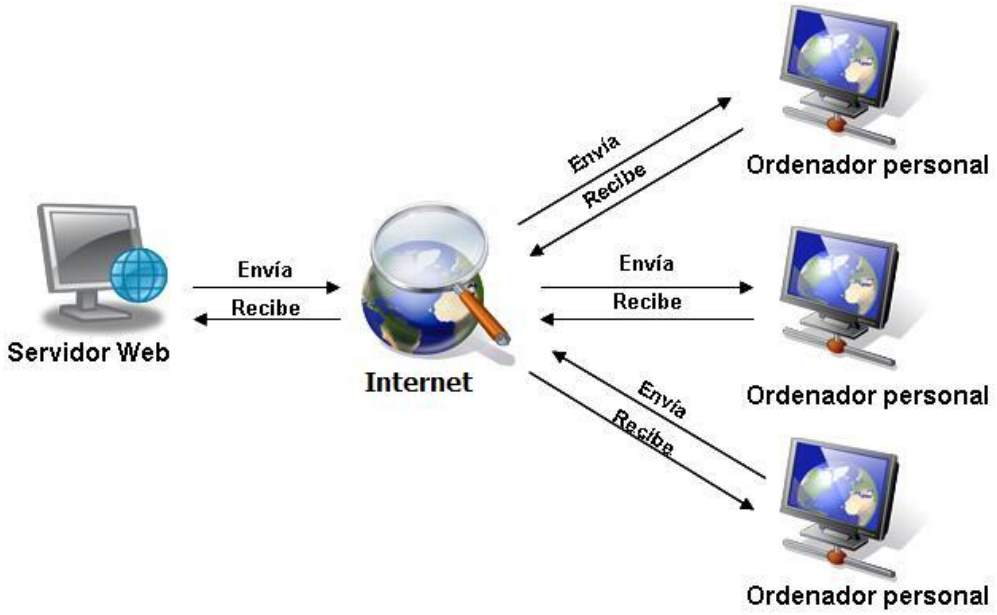


Figura 4. Comunicación Cliente / Servidor (http://servhost.wordpress.com/category/servidor-web/)

El llamado protocolo de la web es de particular interés en este trabajo, pues establece las bases y mecanismos que deben ser utilizados para que la comunicación entre el cliente y el servidor ocurra de manera exitosa. En el caso de la Internet, el cliente es en realidad el navegador de su PC y el servidor es un ordenador central situado en algún lugar en el Internet. Típicamente, el navegador envía al servidor una petición de algo llamado "una página web". El servidor procesa dicha petición y envía una respuesta de vuelta al navegador muy habitualmente en la forma de una página web. Lo que

llamamos página web no es otra cosa que un documento electrónico hipertextual diseñado especialmente para Internet. Este documento contiene generalmente información en forma de texto, imagen, video, animación u otros. Una de las principales características de estos documentos son los hipervínculos, también conocidos como links o enlaces cuya principal función es la de vincular una parte del documento con otra o una página con otra página, permitiendo que la lectura no necesariamente deba ser secuencial.

Eso es posible gracias al protocolo HTTP (HyperText Transfer Protocol) o Protocolo de Transferencia de Hipertextos, que es utilizado por los servidores de internet desde el nacimiento de la Web en 1990. El protocolo HTTP es el que permite el intercambio de información hipertextual (enlaces) de las páginas web. Se trata de un protocolo genérico orientado a objetos, que puede usarse para muchas tareas como servidor de nombres y sistemas distribuidos orientados a objetos, por extensión de los comandos o los métodos usados. Una de sus características principales es la independencia en la visualización y presentación de los datos, lo cual permite que los sistemas sean construidos independientemente del desarrollo de nuevos avances en la representación de los datos.

Las páginas web o documentos hipertextuales revisten un particular interés en esta investigación pues, en estricto rigor, son estos los que de alguna u otra forma concentran el 100 % de la información que existe en la web. Y dado que uno de los objetivos de la investigación es extraer información de la web, resulta imprescindible conocer la forma en que esta información se organiza y se transmite en internet.

Hay mucho que decir con relación a internet respecto de su arquitectura y funcionamiento (23) (Protocolos, enrutamiento, direccionamiento, topologías de redes y networking por mencionar algunos temas), pero muchos de estos escapan al alcance de esta investigación. El propósito aquí no es comprender exactamente cómo trabaja internet, sino contextualizar y mencionar aspectos fundamentales como entender la forma en que se organiza la web y la información existente en ella, puesto que así, será posible proponer mecanismos apropiados para la extracción de datos.

3.2.1.1. Organización de la información en la web

La información que se acumula en la web se agrupa en conjuntos en documentos de hipertexto (páginas web) denominados sitios web alojados en servidores. Cada una de estas páginas se construye utilizando un lenguaje llamado Hypertext Markup Language (HTML) (23). Este lenguaje contiene comandos que le indican al navegador cómo mostrar el texto, los gráficos y archivos multimedia. También contiene comandos para vincular la página a otras páginas y recursos de Internet. A pesar de que existen muchos lenguajes para la construcción de páginas

dinámicas como PHP, JSP, ASP, ASP.net, Pyton, Perl y otros más, finalmente la ejecución del script o programa generará código HTML que será enviado al cliente (navegador) para que el contenido generado sea desplegado como si se tratare de un documento creado con HTML. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>). El lenguaje HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo, JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML. HTML desde su primera aparición formal mediante un documento con su especificación publicado por primera vez en Internet por Tim Berners-Lee en 1991 (24), consta de varios componentes vitales, entre ellos los elementos y sus atributos, tipos de data y la declaración de tipo de documento.

a) *Elementos:* Los elementos son la estructura básica de HTML que tienen dos propiedades básicas: atributos y contenido. Cada atributo y contenido poseen restricciones para que el documento HTML sea considerado válido. Un elemento generalmente tiene una etiqueta de inicio, como por ejemplo <nombre-de-elemento> y una etiqueta de cierre como </nombre-de-elemento>. Los atributos del elemento están contenidos en la etiqueta de inicio y el contenido está ubicado entre las dos etiquetas. Ejemplo:

```
<nombre-de-elemento atributo="valor">Contenido</nombre-de-elemento>
```

Otros elementos como
, no tienen contenido ni llevan una etiqueta de cierre.
Para ver una completa lista de los elementos con su descripción ver (25).

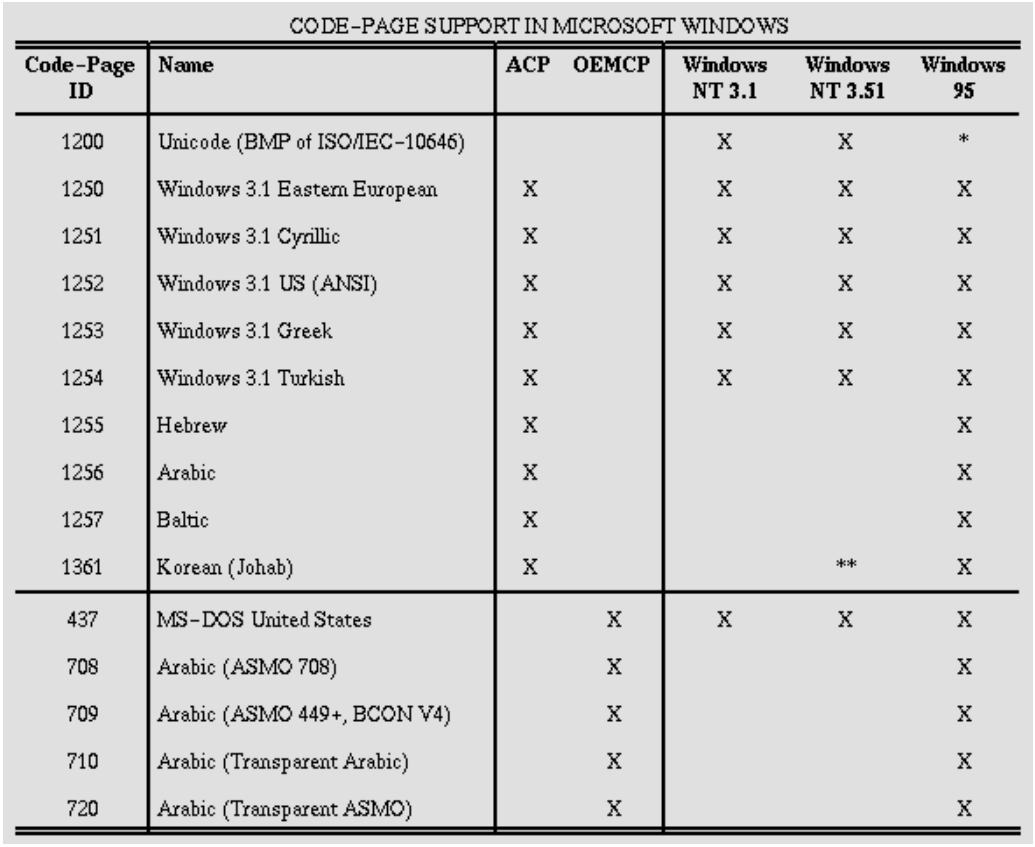
b) *Atributos:* Los atributos de un elemento en su gran mayoría son pares nombre-valor separados por un signo de igual "=" y escritos en la etiqueta de comienzo de un elemento, después del nombre de éste. El valor puede estar rodeado por comillas dobles o simples, aunque ciertos tipos de valores pueden estar sin comillas en HTML, pero no en XHTML (26). De todas maneras, dejar los valores sin comillas es considerado poco seguro. Ejemplo:

```
<A href="chapter2.html">chapter two</A>
```

Los documentos html deben tener una cierta estructura para su correcta presentación en los navegadores. Esta estructura es la que se muestra a continuación.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
        <HEAD>
                <TITLE>A study of population dynamics</TITLE>
                ... Otros elementos del encabezado
        </HEAD>
        <BODY>
                ... cuerpo del documento
        </BODY>
</HTML>
```

La mayor parte del contenido se encuentra entre las etiquetas <body> y </body> el que, como ya hemos mencionado, puede ser texto, imágenes, hipervínculos e incluso audio. En el anexo A se muestra el código html que corresponde a la figura 5.



CODE-PAGE SUPPORT IN MICROSOFT WINDOWS

Code-Page ID	Name	ACP	OEMCP	Windows NT 3.1	Windows NT 3.51	Windows 95
1200	Unicode (BMP of ISO/IEC-10646)			X	X	*
1250	Windows 3.1 Eastern European	X		X	X	X
1251	Windows 3.1 Cyrillic	X		X	X	X
1252	Windows 3.1 US (ANSI)	X		X	X	X
1253	Windows 3.1 Greek	X		X	X	X
1254	Windows 3.1 Turkish	X		X	X	X
1255	Hebrew	X				X
1256	Arabic	X				X
1257	Baltic	X				X
1361	Korean (Johab)	X			**	X
437	MS-DOS United States		X	X	X	X
708	Arabic (ASMO 708)		X			X
709	Arabic (ASMO 449+, BCON V4)		X			X
710	Arabic (Transparent Arabic)		X			X
720	Arabic (Transparent ASMO)		X			X

Figura 5. Vista en un navegador del código html de anexo 1 (23)

El contenido que se muestra en la figura 5 corresponde únicamente a texto y además es semi-estructurado.

Los contenidos estructurados proveen información semántica precisa, la que en muchos casos corresponde a una estructura predefinida con etiquetas definidas cuya función es mostrar información relevante del documento más allá de la información que se presenta como contenido. Esta información relevante es semántica.

El contenido semi-estructurado, por el contrario, no posee una estructura semántica predefinida, sino un conjunto de etiquetas de marcado que permiten representar la información textual (27). El resultado es un documento que en su mayoría de contexto contiene elementos de un documento estructurado, es decir, contiene la estructura que proporciona html para representar la información, pero deja algunas partes del documento sin estructura, no determinando de esta forma el contenido semántico. Lo que resta es identificar el contenido no estructurado, esto resulta bastante simple, pues un documento no estructurado carece completamente de estructuras predefinidas que permitan escribir la semántica del contenido textual. Ejemplo de ello, un archivo de texto. No obstante, a pesar de que todas las páginas web contienen implícitamente la estructura que proporciona html para representar la información, esto no determina en si mismo que se trate de contenido estructurado. Si observamos el código del siguiente cuadro, advertiremos que a pesar de las etiquetas de marcado, el contenido textual que es lo que nos interesa, es un párrafo en el cual no es posible ver estructura alguna ni menos aun semántica.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
        <HEAD>
                <TITLE> Documento no estructurado </TITLE>
        </HEAD>
        <BODY>
                El llamado protocolo de la web es de particular
        interés en este trabajo, pues establece las bases y
        mecanismos que deben ser utilizados para que la
        comunicación entre el cliente y el servidor ocurra de
        manera exitosa. En el caso de la Internet, el cliente es
        en realidad el navegador de su PC y el servidor es un
        ordenador central situado en algún lugar en el Internet.
        Típicamente, el navegador envía al servidor una petición
        de algo  llamado "una página web". El servidor procesa
        dicha petición y envía una respuesta de vuelta al
        navegador muy habitualmente  en la forma de una página
        web. Lo que llamamos página web no es otra cosa que un
        documento electrónico hipertextual diseñado especialmente
        para Internet.
        </BODY>
</HTML>
```

Desde la primera versión de html hasta la última (HTML 5) y XHTML (28), las condiciones para representar la información, en esencia, son prácticamente las mismas, a pesar de las extensiones y evolución del lenguaje de marcas y las nuevas tecnologías como CSS (28) para definir el formato con que se tiene que presentar la información. No obstante el uso de internet y el tipo de información que circula en internet ha cambiado bastante. Así por ejemplo, la web de los 90's giraba en torno a hipertexto, las imágenes, el audio y video. La web en este período se concentraba en presentar los sitios y directorios como canales de difusión donde los internautas en su mayoría se comportaban únicamente como consumidores de información. Desde entonces hasta nuestros días hemos podido presenciar cambios graduales, tanto en el uso que se le da a internet como en el tipo de información que alberga la web. Estos cambios han propiciado la inclusión de internautas y con ello la creación de comunidades virtuales y redes sociales. Como consecuencia de ellos hemos presenciado fenómenos como Facebook, Youtube y la computación en la nube por mencionar algunos. El resultado final, una espiral creciente de datos e información en la web, puesto que al involucrar a los internautas de forma directa, ahora son ellos los que producen la mayor cantidad de tráfico e información. A todo este fenómeno se le conoce como Web 2.0 (29), una web donde abundan las redes sociales y un conjunto de servicios que nos permiten elaborar, modificar, almacenar, introducir y compartir información en la red. La Web 2.0 es la representación de la evolución de las aplicaciones tradicionales hacia aplicaciones web enfocadas al usuario final, no es una actitud y no precisamente una tecnología, es la transición que se ha dado, de aplicaciones tradicionales hacia aplicaciones que funcionan a través de la web enfocadas al usuario final. Se trata de aplicaciones que generen colaboración y de servicios que reemplacen las aplicaciones de escritorio. No vamos a profundizar en la arquitectura de la Web 2.0 (30), pero lo que sí vamos a precisar es que en los navegadores de hoy (Google Crome 2013, Internet Explorer 10 y Firefox 20.01) el contenido que se muestra sigue siendo HTML o XHTML (31) sin perjuicio que se haya generado dinámicamente como resultado de la ejecución de algún script (PHP, JSP, ASP, ASP.net, Pyton, Perl). Esto, para nuestros intereses dice mucho, el contenido que necesitamos extraer se encuentra disperso en un sinfín de documentos HTML. Por lo tanto, nuestra tarea ahora será buscar las herramientas apropiadas para extraer la información de los documentos HTML.

Esto no supone en ninguna manera una falencia del lenguaje, pues este permite hacer exactamente lo que se planificó hiciera, es decir, representar apropiadamente información textual en los navegadores clientes para que los usuarios interactúen con ella.

3.2.1.2. Proyección de la web

Como se mencionó en el apartado anterior, la web 2.0 es la transición de aplicaciones tradicionales hacia aplicaciones que funcionan a través de la web enfocadas al usuario final. Esta transición no aplica únicamente a las aplicaciones, si no a la web entera y además no termina aquí. La web sigue evolucionando y los usuarios continúan demandando nuevas funcionalidades. La próxima parada de este autobús llamado transición será una web que incorpore lenguaje natural para realizar consultas sobre su contenido, lenguajes interpretables y entendibles por organismos de software que busquen, integren y construyan información. En esta nueva web se espera que las máquinas puedan leer comprensiblemente las páginas web de manera autónoma, donde la web entera se comporte como una especie de gran base de datos con mecanismos de búsqueda donde sea posible advertir la fusión de web e inteligencia artificial. En esta web se espera que las páginas, es decir, el contenido textual, posean significado en sí mismo. A la web de la que hablamos se le conoce como Web 3.0 (31) o Web Semántica concepto acuñado por Tim Berners-Lee. La web semántica supone la existencia de una serie de prestaciones inteligentes para con los usuarios de la web. La Web semántica, conceptualmente hablando, proporciona un marco común que permite que los datos sean compartidos y reutilizados a través de aplicaciones, empresas y fronteras comunitarias. Hablamos entonces de mecanismos subyacentes en la web que permitan la búsqueda, transporte e intercambio de información de manera autónoma. El despliegue completo de la web semántica, de acuerdo a las estimaciones de la W3C, puede prolongarse varios años después del 2010 (“Web Semántica: Principios y Estándares”; Samanta P. Cueva, 2008). No obstante, la web semántica ya se encuentra entre nosotros en varias formas. Partiendo por el hecho de movilizar a una gran cantidad de investigadores quienes proponen cambios profundos, cuya aplicación lamentablemente tienen lugar en el mediano y largo plazo. Con el propósito de propiciar estos cambios, se han desarrollado nuevos estándares, entre ellos el XML (32), que ha resultado de uso cotidiano en un sin fin de aplicaciones.

En la visón de web semántica se pueden observar dos grandes definiciones que confluyen y se complementan para dar solución a todo el grupo de necesidades planteadas en el apartado anterior. Primero, la Web semántica debe promover una web futura cuyas páginas estén organizadas, estructuradas y codificadas de tal manera que los computadores sean capaces de efectuar inferencias y razonar a partir de sus contenidos. Y por otra parte, debe permitir que la Web pueda ser concebida como una gran base de datos capaz de soportar un procesamiento sistemático y consistente de la información. La arquitectura (33) propuesta para esta nueva web sería como se muestra en la figura 6.

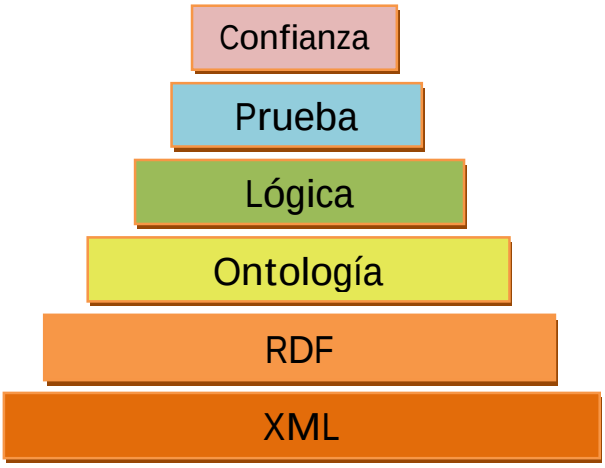


Figura 6. Arquitectura Web Semántica (Basada en (33))

Esta arquitectura implica cambios radicales y profundos en la forma en que se almacenan y organizan los datos en la web, cambios que como se señaló recientemente, pueden tomar bastante tiempo. No obstante, el estándar XML como elemento base de la arquitectura se ha difundido enormemente para ser usado en muchas aplicaciones y con diversos propósitos (Almacenamiento, transporte e intercambio de datos entre algunos).

En la capa XML se estructuran los datos y documentos por lo que bajo esta capa base subyacen los datos "duros" de la web. La capa RDF proporcionan los mecanismos para describir bajo este estándar, ontologías y metadatos. En la capa ONTOLOGIA encontramos la descripción de los recursos (vocabularios y metadatos) propiamente tal. Una ontología se puede definir como una descripción formal y estructurada de un cuerpo de conocimiento con el propósito de que este conocimiento pueda ser compartido sin perder semántica o significado. En la capa LOGICA se dota de flexibilidad a la arquitectura para realizar consultas e inferir conocimiento a partir de las ontologías, por ejemplo, mediante lenguajes formales o programación lógica. La capa PRUEBA sugiere el máximo nivel de fiabilidad que puede tener un computador respecto de las decisiones tomadas, razonamientos o procesos deductivos realizados. Finalmente, la capa de confianza no difiere mucho conceptualmente de lo que significa la capa de confianza en la web tradicional, esto es, grado de seguridad y confianza a los servicios web mediante diversos mecanismos y/o tecnologías (autentificación, firmas digitales, protocolos de seguridad, etc.).

Existe bastante literatura sobre todos los niveles, pero actualmente solo los tres primeros disponen de desarrollo. Lamentablemente para utilizar completamente los servicios ofrecidos por la web 3.0 se requiere de una implementación completa y global de la arquitectura. Esta situación junto con la gran cantidad de datos en la web constituye la motivación principal de buscar alternativas que

permitan hacer uso inmediato de la abundante información existente en la web aunque se requieran procesos de extracción y transformación (adecuación) antes de poder utilizarla.

Siendo la web semántica una realidad o no, los agentes y agentes inteligentes cobran relevancia, pues, en presencia de una nueva arquitectura como la señalada anteriormente, estos pueden interpretar e inferir a partir de las ontologías. En caso contrario, donde se requiere extracción previa o adecuación de los datos, los harvesters (34), una aplicación específica de agentes de software, pueden realizar fácilmente las labores de extracción.

3.2.2. Agentes y Agentes inteligentes

De acuerdo con la Real Academia Española agente quiere decir: "Que obra o tiene virtud de obrar", "Que designa a la persona, animal o cosa que realiza la acción del verbo", "Persona o cosa que produce un efecto". En todas las definiciones anteriores, el concepto agente tiene asociado un comportamiento dinámico que involucra una o más acciones que debe realizar, lo cual es consecuente con el contexto informático, ya que cuando nos referimos a Agente, generalmente se hace referencia a un organismo artificial que puede ser de tipo software, hardware o un sistema híbrido formado por los dos anteriores, que también tienen asociada una o varias acciones establecidas para este.

La palabra agente, también proviene del latín *agere* que quiere decir "obrar por otro". Esta es otra de las cualidades que permite explicar el concepto de agente, puesto que el agente es un encomendado del usuario, el cual fue concebido con el propósito de realizar tareas específicas. En este sentido es el usuario quién delega en el agente diferentes labores, generalmente con objeto de alivianar la carga realizando tareas que son altamente repetitivas o también para asistir en tareas complejas.

Una cualidad importante de los agentes es su continuidad temporal, puesto que se trata de un proceso temporalmente continuo, esto quiere decir que a diferencia de un software corriente del cual se conoce su inicio y fin, un agente debe ejecutarse hasta que se haya alcanzado el conjunto de objetivos solicitados, o bien, mientras su usuario no desee detenerlo. En este sentido, el concepto de continuidad temporal es la propiedad que da "vida" al agente, posibilitando que se mantenga alerta a una solicitud o a algún cambio en el medio. (35)

Según [Maes 1995, página 108] "Los agentes autónomos son sistemas computacionales que habitan en algún complejo entorno dinámico actuando de forma autónoma en este entorno, y de esta manera realizan un conjunto de tareas para las que fueron diseñados.". No se trata únicamente de un software que automatiza tareas repetitivas, sino que en las tareas que realiza existe autonomía en la toma de decisiones. Por esa razón no son solo agentes de software, sino Agentes Inteligentes, los que por definición deben continuamente realizar tres funciones básicas (36): 1) Percibir las condiciones dinámicas del medio ambiente, 2) Reaccionar frente a las condiciones en el medio ambiente y 3)

Razonar para interpretar percepciones, resolver problemas, hacer inferencias, y determinar las acciones.

La conceptualización general de un agente como el que se describe en el apartado anterior, se concibe como un organismo inteligente (Agente Inteligente) y autónomo. Se dice que es inteligente por sus capacidades proactivas y reactivas de cara a los estímulos del entorno en el cual opera, es decir, debe poseer la capacidad de tomar las mejores decisiones que le permitan la consecución de sus objetivos de diseño, aun cuando su entorno esté en constante cambio o sea impredecible. Según (37), los Agentes Inteligentes cuentan con una serie de características. Algunos, además de reaccionar ante los cambios del medio, tienen la capacidad de poder adaptarse progresivamente a cambios en entornos dinámicos mediante técnicas de aprendizaje, lo cual aumenta su grado de inteligencia y por ende las posibilidades en cuanto a la eficacia y eficiencia en el desarrollo de diversas tareas. En resumen, un Agente Inteligente es un Agente de software que puede funcionar fiablemente en un entorno rápidamente cambiante o impredecible. El Agente Inteligente no debe ser programado nuevamente si cambia su entorno. No obstante, los agentes inteligentes son programas que han sido definidos por diversos autores y en muchos casos de manera imprecisa o ambigua, lo cual dificulta establecer con claridad y certeza, si un determinado sistema es o no un agente (39). Las clasificaciones que podemos hacer de los agentes son variadas y dependen de las diferentes características que se tengan en cuenta. Así por ejemplo, un sistema o un software en la medida que se le dote gradualmente de ciertas características podrá ser concebido como un agente, inteligente o medio inteligente, también dependerá del grado de inteligencia. Un organismo de software parcialmente autónomo programado para asistir o automatizar una tarea altamente repetitiva, pero sin la capacidad de reaccionar ante un entorno dinámico ni adaptarse a él también puede ser considerado un agente.

Lo importante aquí es establecer como parte del enfoque de nuestro planteamiento, que una clasificación taxativa, con objeto de determinar si un software es o no un agente inteligente, resulta menos clara frente a un mecanismo que permita realizar una clasificación gradual del agente. Para tener más claridad sobre este punto se muestra una gráfica espacial donde es posible ubicar un tipo de agente según tres características.

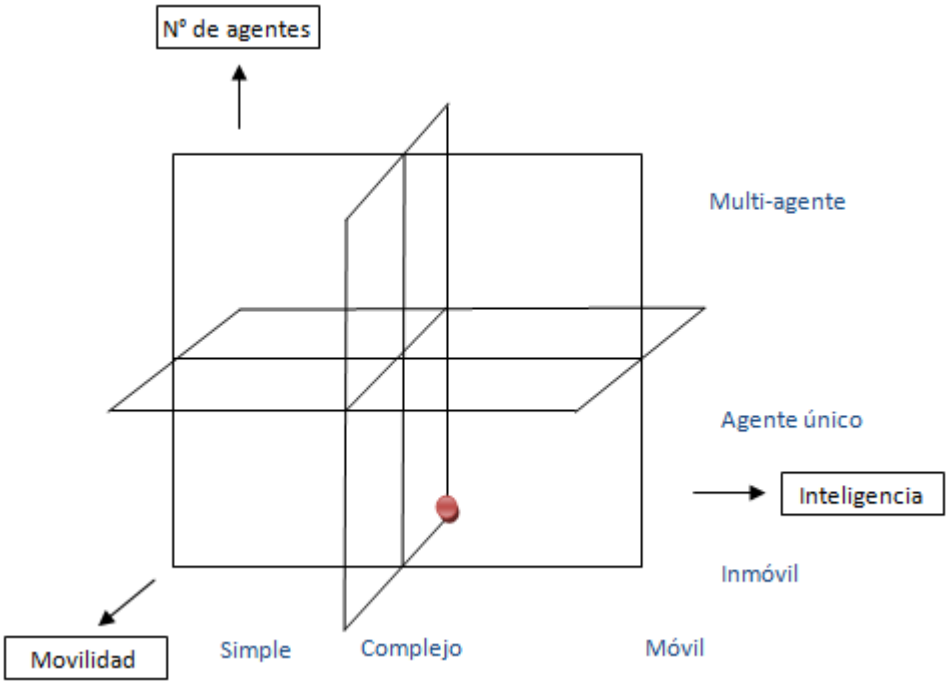


Figura 7. Clasificación de agentes

En la figura 7 se puede observar gráficamente como un agente de software, de manera no taxativa sino gradual, puede ser clasificado de acuerdo a tres parámetros (N° de agentes, grado de inteligencia y movilidad). El grado de inteligencia cubre aspectos como la capacidad de reacción y adaptación frente al entorno (36) (37). El parámetro N° de agente simplemente determina la cantidad de agentes que opera en la solución (Sistema multi-agente) (40) (41). Por último, el parámetro movilidad mide la capacidad de migrar de un nodo a otro en una red preservando su estado en los saltos entre nodos. (42) (43).

El punto en el plano muestra el tipo de agente en el cual estamos interesados, un agente no móvil, único y de inteligencia media.

3.2.2.1. Arquitectura

En general la arquitectura de un agente inteligente, tal como se muestra en la figura 8, incorpora cuatro elementos básicos:

a) Sensores: Corresponden a los mecanismos que permiten percibir el entorno sin razonamiento alguno. Solo se encargan de efectuar lecturas sobre el ambiente o entorno.

b) Efectores: Estos elementos son los que realizan las acciones sobre el medio.

c) Mecanismo de observación: En estos mecanismos se encapsula el componente inteligencia. Es aquí donde, de acuerdo a los antecedentes proporcionados por los sensores, se realizan interpretaciones que indiquen como está el entorno y posteriormente las acciones que se deben realizar. Si el agente, en función de lo que percibe, es capaz de incorporar nuevas reglas al mecanismo de observación que permitan continuar interpretando lo que percibe del entorno aun cuando este cambie constantemente, se le llama agente adaptativo (44) (45).

d) Conjunto de acciones predefinidas: Este componente concentra las acciones que predefinidamente se establecen en el agente para reaccionar ante el entorno de acuerdo a las observaciones e interpretaciones realizadas. Si el agente, de acuerdo a las observaciones e interpretaciones, es capaz de modificar el conjunto de acciones establecidas inicialmente, será capaz de modificar su comportamiento. A estos agentes se les llama evolutivos (46) (47).

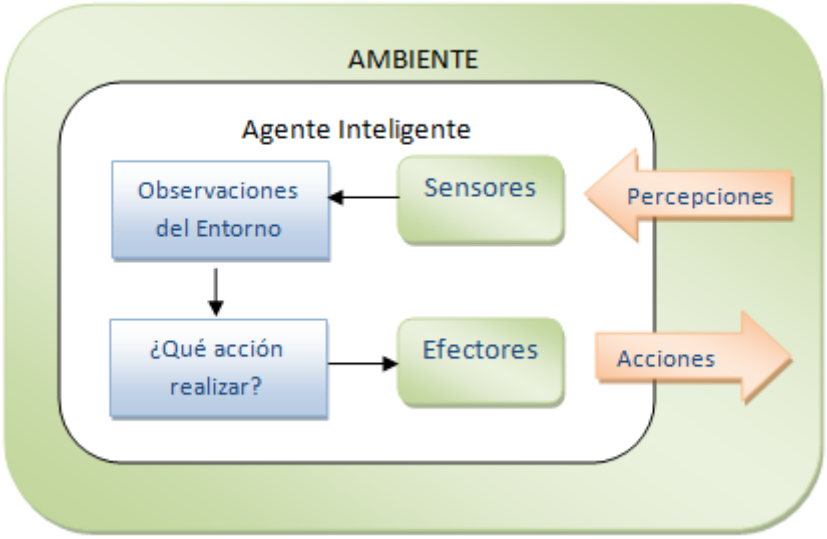


Figura 8. Clasificación de agentes

Esta arquitectura es coherente con la clasificación de agentes realizada en la figura 7 puesto que el agente de nuestro interés (no móvil, único y de inteligencia media) posee estos cuatro elementos.

3.2.2.2. Agentes de software en la web

Debido al enorme caudal de información que circula en este escenario (ambiente web) los agentes de software ofrecen muchas posibilidades. Con relación a las posibilidades, una clasificación tentativa puede ser la siguiente:

a) Extracción de información: En este tipo de aplicaciones se encuadran todas las formas posibles de recuperar información de la web, como por ejemplo, seguir el rastro que dejan los

internautas cuando navega en internet, monitorear y registrar todas las acciones que realiza un cliente cuando visita una sitio web, extraer información de datos semiestructurados.

b) Procesamiento de información: En esta categoría se encuentran los Agentes destinados a labores de Minería de datos (47) (48), mediante la cual es posible efectuar actividades asociadas a dos temas diferentes pero perfectamente complementables: 1) Descubrir conocimiento y 2) Predicción.

c) Interfaces Hombre-Máquina: Estos agentes permiten simular directamente el comportamiento de un usuario en alguna determinada situación. Así por ejemplo, los *Chat Bots* o Robots de chat son una especie de agentes conversacionales (50) que permiten simular una conversación inteligente con un usuario real. Todos ellos simulan la presencia de un usuario real tras un computador, pero en realidad están destinados a simular una conversación bajo algún determinado requerimiento. Existen algunos programados para dar la bienvenida al sitio web de una empresa con la posibilidad de responder una serie de preguntas frecuentes. Otros en cambio cumplen un rol únicamente social y de entretención.

Para la concreción de los objetivos planteados en esta investigación se abordarán los agentes destinados a la extracción de información desde la web.

3.2.2.3. Proyección

Los agentes de software están abarcando cada vez más escenarios en donde sea posible ayudar a los usuarios a encontrar y gestionar la información, en particular en entornos abiertos como la Internet. En su mayor parte, se trata de Agentes Inteligentes que “hacen lo que se les pide”, realizan las tareas para las cuales fueron diseñados interactuando con el medio establecido y de la forma en que se concibió, de manera absolutamente autónoma. Por lo tanto, dada su autonomía, no toma ventaja de las capacidades o resultados de otros agentes que pudiesen encontrarse en ejecución en el mismo escenario, provocando una duplicación de trabajo. Esto puede derivar en una práctica ineficiente. Con el propósito de ser más eficientes en este sentido, existe una tendencia a que los Agentes Inteligentes deben ser conscientes de sí en el sentido de comunicarse sus reglas de operación. Una forma de hacerlo es mediante el intercambio de mensajes. ("Hola, soy agente “A” consulto precios de CDs, las reglas que rigen mi comportamiento son las siguientes…), de esta forma sin un agente en operación, mientras recorre la red (Agentes móviles) (42) (43) se encuentra con otro, podrán interactuar y usarse entre sí en el sentido de solicitar servicios uno del otro par ser más eficiente (51). A los agentes que describen un comportamiento como el de esta tendencia se les llama Agentes Colaborativos. Otra tendencia que toma fuerza sugiere que los Agentes Inteligentes no solo deciden de manera autónoma con base en las directrices establecidas durante su programación y configuración y no solo resuelven los problemas

para los cuales fueron construidos, sino que, en el marco de la Inteligencia Artificial, algunos ya incorporan mecanismos de aprendizaje autónomo (52) que les permite incorporar nuevo conocimiento, el cual se sintetiza como nuevas reglas en la forma de interactuar con el medio que se encuentra en constante cambio. Estos mecanismos convierten a los Agentes Inteligentes en Agentes Inteligentes "más Inteligentes" en el sentido de no tener limitaciones con relación a su programación y reglas iniciales, pues estas, de acuerdo al aprendizaje obtenido pueden cambiar y perfeccionarse en el tiempo para ejecutar de mejor manera las tareas encomendadas o tomar mejores decisiones. A estos tipos de agentes se les llama Agentes Evolutivos.

En concreto, estas tendencias dan cuenta del aumento en la investigación en el área de los agentes inteligentes, específicamente en torno a los atributos *grado de inteligencia* y *colaboración* (N° de agentes) presentados en la figura 7. Además, esta tendencia se alinea con la visión de web semántica, pues esta arquitectura incorpora prácticamente de forma nativa el uso de agentes inteligentes.

3.2.3. Uso de la información que está en la web

La web como la conocemos hoy alberga y genera una enorme y cada vez más creciente cantidad de datos. La cantidad de información ha crecido en conjunto con el interés que las personas tienen sobre esta. Según (53) esto se debe a que los datos tienen una importancia crucial para el mejoramiento de la web desde un punto de vista social y también comercial. Por esta razón, cada vez más son las personas que prestan atención a los datos existentes en la web, pues estos pueden ser utilizados para apoyar algún proceso de toma de decisiones y/o posibilitar la elaboración de nueva información a partir de los datos duros.

3.2.3.1. Uso directo

Un primer enfoque para utilizar los datos que existen en la web concibe la idea de consultar directamente los datos como si se tratara de un gran repositorio de datos. Esta idea de momento solo es posible en el planteamiento que hace la web semántica. La web semántica es un área prolífera, situada en la confluencia de la inteligencia artificial y las tecnologías web, que propone nuevas técnicas y paradigmas para la representación de la información y el conocimiento para facilitar tanto localizar como el compartir, integrar y recuperar recursos. La web semántica propone enriquecer la estructura de la información y agregar componentes semánticos que puedan procesarse de forma automática, pero considerar este primer enfoque, requiere de una completa implementación de la arquitectura presentada en la figura 6. A pesar de que la web semántica representa un área de investigación cada vez más

creciente, no en todos los niveles de su arquitectura existen desarrollos, por lo tanto nuestro primer enfoque tendrá que esperar algún tiempo.

3.2.3.2. Extracción

Proceso de extracción de datos de páginas web también se conoce como raspado Web (Web scraping). La Web contiene muchos datos que nos gustaría consumir para nuestras necesidades. El problema es que esta información se encuentra en la mayoría de los casos mezclada entre sí, con código HTML, esto es necesario para que la información se despliegue apropiadamente en los navegadores clientes, pero dificulta la extracción automática. Un proceso manual de separar, copiar y pegar el contenido útil estaría propenso a errores, sería tedioso y casi imposible de hacer. Los diseñadores de software Web suelen discutir cómo hacer una separación limpia entre el contenido y el estilo, lo que en muchos casos se consigue con diversos frameworks y patrones de diseño. No obstante, este diseño generalmente se realiza a nivel de servidor, de manera que el manejo de HTML se entrega al cliente web.

La extracción de datos desde la web como segundo enfoque, demanda la existencia de mecanismos que permitan obtener los datos antes de poder trabajar con ellos. En este contexto no podemos pasar por alto el concepto de Web Minig (53) (54). Esta área de las Ciencias de la Computación involucra el uso de técnicas para la extracción automática de información de documentos y servicios de la web. Frente a la inminente tarea de extracción tenemos tres posibilidades:

a) *Minería de contenido:* Corresponde a la exploración de documentos HTML (texto, imágenes, etiquetas y metadatos entre otros) mediante el uso de robots o agentes específicos llamados también *crawler* o *rastreador* (55) (56). Si la función del robot en lugar de rastrear será únicamente extraer datos de acuerdo a ciertos patrones, la tipificación más apropiada para este robot sería *harvester* (*cosechador*) (56) (34)
b) *Minería de estructura:* Exploración y seguimiento de enlaces de las páginas web y las relaciones existentes con otros enlaces de otros sitios, también para encontrar enlaces no existentes o páginas desconectadas. Para extraer este tipo de datos se emplean robots de tipo *crawlers.*
c) *Minería de uso:* Se encarga de extraer la información que se produce cuando las personas interactúan con la web. Esto se consigue analizando los *registros logs* de servidores web o aplicaciones especificas. Analizar los logs resulta de mucho interés y valor comercial. Si una página nunca es visitada tal vez no tiene razón de ser o, por el contrario, si las muy visitadas no están en los primeros niveles de jerarquía del sitio web, esto sugiere mejorar su organización y

navegación. Por lo tanto, es importante detectar patrones, tendencias y relación entre estos datos. Esta detección puede ser realizada examinando los registros o logs y los resultados obtenidos, pueden ser usados para recomendar servicios o productos, posicionar en la página en un buscador mediante el cálculo de un pageRank (58).

En términos generales, crawler es un término genérico para cualquier programa *robot* o *spider* dedicado a escanear páginas web. Un programa o script automático que navega por internet de forma metódica y automatizada. Estos programas son casi tan antiguos como la propia web. El primer crawler fue escrito en 1993, pero debido a la explosión de la web, los crawlers son hoy en día un componente esencial de todos los motores de búsqueda y son cada vez más importante en la minería web y otras aplicaciones de indexación. Un crawler visita sitios web, lee sus páginas y otra información, en algunos casos con el fin de crear entradas de índices en motores de búsqueda. Los principales motores de búsqueda en la Web tienen un programa de este tipo, que también se conoce como "bot". Los crawlers aparentemente ganaron el nombre porque se arrastran a través de un sitio de una página a la vez, siguiendo los enlaces a otras páginas en el sitio hasta que se hayan leído todas las páginas. Existen algunos crawlers famosos como el del motor de búsqueda de Google llamado *Googlebot* o el de AltaVista llamado *Scooter*.

En otros casos los crawlers son empleados para recorrer las estructuras de los documentos HTML con el propósito de extraer datos que se ajusten a patrones establecidos en su programación. A esta extracción o cosecha de datos se le llama harvesting, de ahí que a los crawlers que realizan exclusivamente esta labor se les llamen *harvesters* o *robot cosechadores*. En este contexto, un harvester es un agente que funciona como un cliente en un protocolo de red para acceder a la World Wide Web y realizar sus operaciones de extracción. En general, podemos decir que entre las tareas más comunes de los crawlers tenemos:

1. Indexar páginas para los motores de búsqueda.
2. Analizar los enlaces de un sitio web para buscar links rotos.
3. Recolectar información de un cierto tipo, como por ejemplo, precios de productos para un análisis posterior o generar un catálogo.

Cualquiera sea la tarea de un crawler, este comienza con una lista de URLs para visitar llamada semilla. A medida que el rastreador visita estas direcciones, identifica todos los hipervínculos de la página y los agrega a una lista de URLs para visitar, llamada frontera de rastreo (Crawl Frontier). Las URLs de la frontera son recurrentemente visitadas de acuerdo a un conjunto de políticas programadas o configuradas en el Crawler. El gran volumen implica que el rastreador sólo puede descargar un número

limitado de páginas Web dentro de un tiempo determinado, por lo que es necesario dar prioridad a sus descargas. La alta tasa de cambio en las prioridades implica que las páginas se actualizan o eliminan constantemente. Si bien es bastante fácil construir un crawler lento que descarga unas cuantas páginas por segundo durante un corto período de tiempo, la construcción de un sistema de alto rendimiento que puede descargar cientos de millones de páginas durante varias semanas presenta una serie de retos en el diseño del sistema, E/S, la eficiencia de la red, y la robustez y capacidad de gestión. En virtud de lo señalado, la arquitectura de un crawler puede ser como sigue (58):

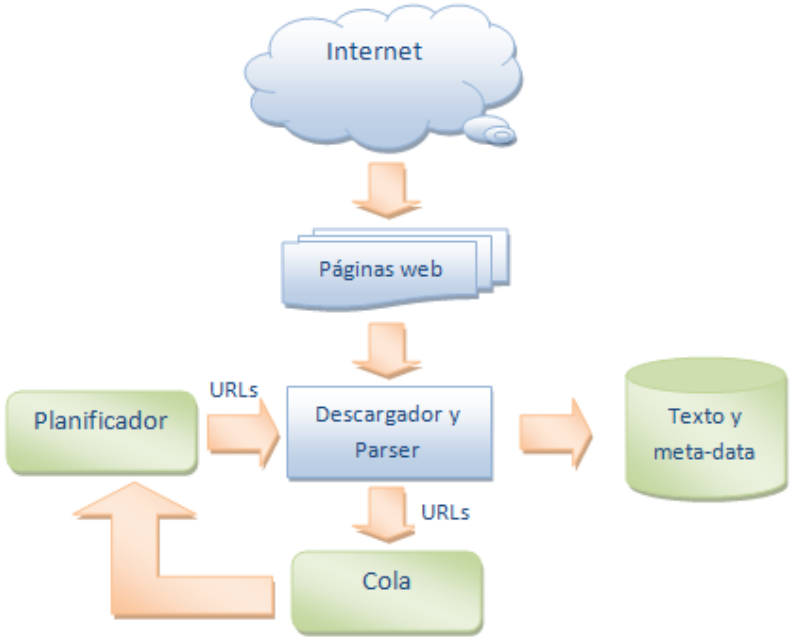


Figura 9. Arquitectura general de un crawler (http://www.milkaddict.com/web-crawlers-googlebot/)

Como se aprecia en la figura 9, la arquitectura de un crawler estándar incorpora cuatro elementos básicos, los que se describen brevemente:

a) *Descargador y Parser:* El trabajo del crawler comienza con la descarga de páginas web de acuerdo a una URL semilla proporcionada por el usuario. Este módulo además de realizar las descarga del documento HTML lo debe limpiar antes de ser escaneado y analizado. La limpieza implica quitar etiquetas poco significativas para el análisis, quitar formato de texto y en general, quitar el estilo del documento html (colores, fondos, etc.), solo se deja la estructura del documento con su contenido. Por parsing entenderemos lo que sucede en un proceso donde se separa lo útil de lo que no lo es; detectando, extrayendo y almacenando elementos como imágenes, palabras, estructuras y cualquier otro elemento que sea de interés desde el documento HTML. En este sentido, luego del parsing, el contenido del documento descargado

se reduce a un tamaño cercano al 30% del original (antes de ser limpiado). Como parte del parsing y dependiendo del propósito del crawler, se deben identificar las URLs presentes en el documento y restaurar aquellos enlaces correspondientes a URLs relativas, estas deben ser trasformadas a URLs absolutas antes de ser agregadas a la lista de URLs por visitar. Como las URLs de la lista son enviadas nuevamente al módulo de descarga y parsing, se debe detectar la presencia de enlaces y contenido duplicado, pues algunas URLs del documento pueden apuntar a contenido del mismo documento.

b) Planificador: Los mecanismos de descarga y parsing generalmente son aplicaciones multi-hilo, por lo que resulta muy necesario un planificador que controle y asigne URLs a los distintos hilos de ejecución para efectuar el proceso de descarga y parsing correspondiente como si se tratara de un sistema operativo que asigna procesamiento a distintos programas en ejecución. Como hablamos de múltiples hilos de descarga, el diseño del planificador es crucial para el rendimiento del sistema. En este sentido el planificador debe ser capaz de determinar qué hacer cuando todos los hilos se encuentran ocupados o las acciones a realizar cuando el contenido descargado se encuentra liberado para su análisis.

c) Almacenamiento de texto y metadata: En un sistema de alto rendimiento, también resulta de suma importancia contar con mecanismos apropiados para el almacenamiento y recuperación del contenido descargado. Los mecanismos de almacenamiento deben además permitir la indexación del contenido para una eficiente recuperación.

d) Cola: La función de la cola o frontera de rastreo es simplemente almacenar temporalmente todas las URLs que son extraídas en el parsing y que esperan ser enviadas una a una a un nuevo proceso de descarga y parsing.

Finalmente, la mecánica del crawler para realizar su cometido, puede ser vista como el conjunto de las siguientes tareas:

1) Selección de una URL a partir de la o las semillas
2) Agregar URL a la frontera de rastreo
3) Ahora elegir URL de la frontera de rastreo
4) Obtención la página web correspondiente a la URL
5) Analizar esa página web para encontrar nuevas URLs
6) Agregar todas las nuevas URLs a la frontera de rastreo
7) Ir al paso 2 e iterar hasta que la frontera se encuentre vacía

De acuerdo a los siete pasos señalados, el proceso de rastreo comienza con un conjunto inicial de URLs semillas. Luego se descargan las páginas web correspondientes a las direcciones URL de

semillas y se extraen los enlaces descubiertos en las páginas descargadas. Las páginas web recuperadas se almacenan e indexan en el área de almacenamiento, de modo que con la ayuda de estos índices más tarde puedan ser recuperadas. Las URLs extraídas de las páginas descargadas se confirman en el sentido de saber si sus documentos relacionados ya han sido descargados o no. Si no han sido descargadas, las URLs son enviadas nuevamente al módulo de descarga para su posterior descarga. Este proceso se repite hasta que no haya más URLs en la lista.

3.2.3.3. Herramientas para la extracción

Existen numerosas herramientas para la extracción de datos desde la web dependiendo del objeto, necesidad y tipo de datos que se desee extraer. No obstante hemos preseleccionado cuatro herramientas como las más representativas en su área de aplicación. Algunos factores considerados como criterio de selección fueron: vigencia, comunidad de desarrolladores, implementaciones realizadas, flexibilidad y curva de aprendizaje.

Respecto de los tipos de datos en materia de extracción, hay tres tipos principales. El tipo de dato más importante y difícil de procesar es el contenido, que es multimedial, en el cual el texto juega un rol dominante. El segundo (estructura) proviene de la estructura no lineal de la web, es decir, los enlaces de las páginas. Finalmente, el último (uso) corresponde a los logs o bitácoras de los servidores Web que resultan del uso e interacción con la web.

Los crawlers son herramientas bastante versátiles que pueden ser empleadas prácticamente en la obtención de los tres tipos de datos. Existen APIs, Aplicaciones con interfaz de usuario y otras híbridas. Las aplicaciones presentadas a continuación corresponden a herramientas de este tipo.

a) *Scrapy:* Scrapy (60) (61) es una Framework open sourse de alto nivel para el web crawling y web scraping (61)(en esta investigación tanto scraping como harvesting significan lo mismo). Puede ser utilizado en el rastreo de sitios web y la extracción de información semi-estructurada de sus páginas para una amplia gama de propósitos como minería de datos, seguimiento y testing. La API está escrita en Pyton y corre en las principales plataformas (Linux, Windows, Mac y BSD). Los crawlers producidos requieren poca codificación y su comportamiento se determina mediante reglas que se establecen en su programación. En la figura 10 se muestra la arquitectura de Scrapy.

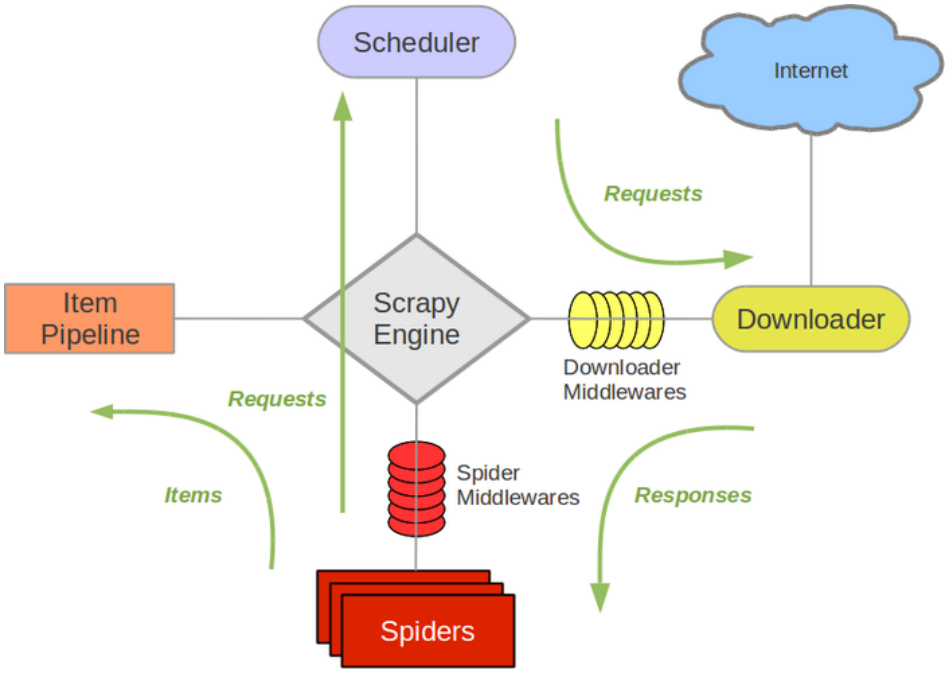


Figura 10. Arquitectura de Scrapy **(60) (61)**

Es una API muy completa y versátil que además ofrece la posibilidad de ser extendida modularmente mediante varios mecanismos sin necesidad de modificar el núcleo. Posee varias dependencias de software para que pueda ejecutar correctamente. Los crawlers requieren de poca programación, no obstante, incluso los muy pequeños requieren de una configuración y estructura de directorios precisa. Se trata de una API diseñada para aplicaciones escalables y en producción con buen rendimiento. Para detalles sobre la instalación y construcción de crawlers con Scrapy ver (60) (61).

b) *Wire:* El crawler WIRE (62) (63) (65) (del acrónimo en inglés Web Information Retrieval Environment que significa "Entorno de Recuperación de Información Web"), es una crawler de propósito general, escalable, altamente configurable, de alto rendimiento y open source. Fue desarrollado en el Center for Web Research (CWR) del departamento de Ciencias de la Computación de la Universidad de Chile para ser utilizado en investigaciones sobre las características y estructura de la Web. Está diseñado para realizar rastreo sobre grandes volúmenes de documentos, del orden de millones o decenas de millones de documentos. Sus principales características son:

1. Buena escalabilidad: Diseñado para trabajar sobre una gran cantidad de documentos.
2. Altamente configurable: La mayoría de los parámetros de rastreo se pueden configurar, incluso las políticas del planificador.

3. Alto rendimiento: Al estar escrito en C/C++ tiene un excelente desempeño.
4. Análisis: En esta materia ofrece herramientas para realizar estadísticas sobre los datos extraídos con la posibilidad de generar reportes.
5. Es Open-Source

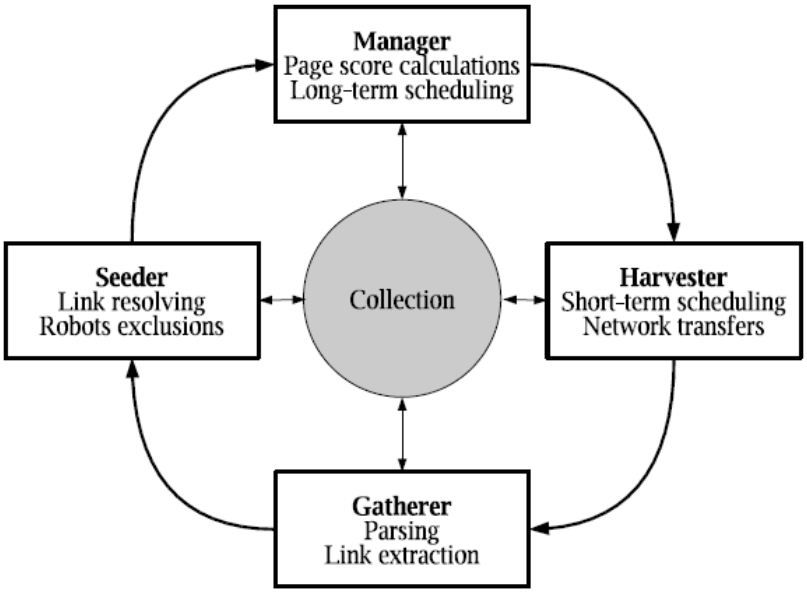


Figura 11. Arquitectura de WIRE (62)

Para detalles de instalación configuración y uso ver (62) (63) (65).

c) jARVEST: jARVEST (Java web harvesting library) es un framework muy simple que facilita el web harvesting. Esta construido mediante un lenguaje de propósito específico DSL (del inglés Domain Specific Language) basado en JRuby. Este framework, permite la construcción de harvesters sin la necesidad de utilizar una gran cantidad de código. Posee un intérprete de un lenguaje muy reducido destinado a la construcción de los harvesters. El código del harvester puede ser embebido en lenguaje JAVA o guardado en un archivo de texto para ser llamado desde una aplicación o directamente desde consola. Está escrito completamente en JAVA, lo cual lo convierte en un framework multiplataforma, pudiendo ser empleado tanto en aplicaciones cliente como servidor. Sus principales características son:

1. Basado en Stream
2. Ensamblaje de robots mediante conexión serie o paralelo
3. Selección de contenido basado en Xphat
4. Manejo de variables dentro del harvest

5. Selector de nodos CSS
6. Línea de comando y API
7. Formulario de envió y seguimiento de cookies (Permite autentificarse y mantener la sesión durante la ejecución del harvest)
8. Construido 100% en JAVA y open-source

La construcción de los harvester (también llamados robots en este documento) se basa en la composición de elementos llamados transformadores, los cuales según el modelo propuesto por jARVEST, pueden ser conectados en serie (también llamado cascada) o paralelo. Un transformador es un componente que recibe un flujo de Strings y entrega como salida otro flujo de Strings. jARVEST proporciona varios transformadores con diversas utilidades que permiten elaborar robots bastante complejos. Cada transformador puede tener hijos los cuales también son transformadores. El flujo de strings enviado a los hijos depende de la política definida en el padre, la que puede ser de dos tipos: Cascada (serial) o Ramificado (paralelo). En el modo cascada si un transformador padre (T1) tiene un hijo (T2) y este hijo tiene otro (T3), la salida producida por T1 es pasada T2 y la salida de este es pasada finalmente a T3. La figura 12 (a) muestra una conexión en cascada de los transformadores T1, T2 y T3.

(a) (b)

Figura 12. Conexión de transformadores cascada (a) y ramificada (b)

Cuando la política de conexión de un transformador que contiene hijos es de tipo Ramificada, la salida del transformador padre (T1) es enviada duplicada a cada uno de los transformadores hijos (T2 y T3) como se observa en la figura 12 (b). Un transformador puede ser concebido como un harvester primitivo que recibe una entrada y produce una salida. La conexión de varios transformadores (en serie o paralelo) componen un harvester que finalmente también recibe una entrada y produce una salida, por lo tanto, un harvester formado por varios transformadores puede ser concebido como un nuevo transformador no primitivo que puede volver a ser conectado con otros transformadores (primitivos o no primitivos) para integrar un harvester mucho más complejo como se muestra en la figura 13.

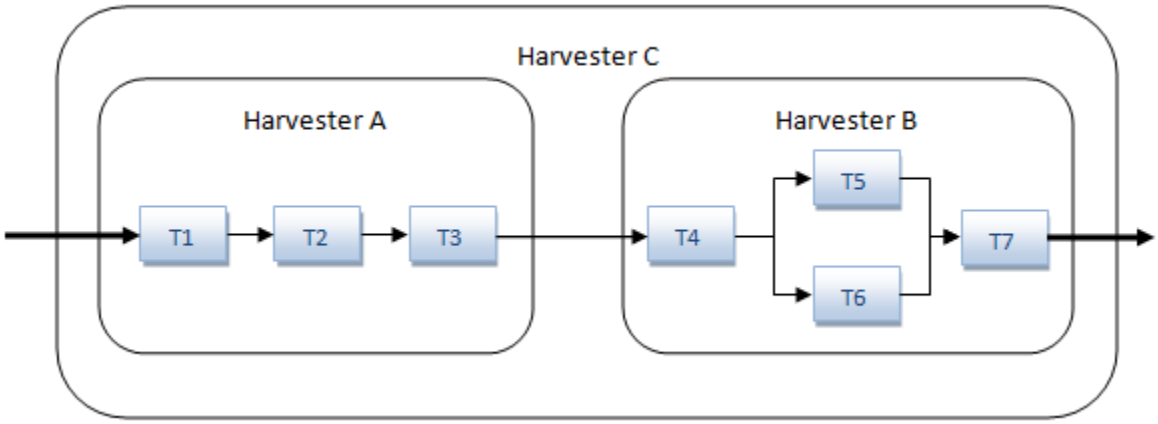


Figura 13. Integración de harvesters

Aquí es posible observar como el harvester A compuesto por tres transformadores primitivos es conectado en serie con un harvester B compuesto por cuatro transformadores primitivos y como los harvesters A y B (trasformadores no primitivos) integran un trasformador C más complejo. Esta integración se puede realizar tantas veces como las capacidades de procesamiento de la máquina cliente lo permita. La sintaxis general para los transformadores es la siguiente:

```
nombre_transformador(:parametro1=>'value', :parametttro2=>'valor'){
          #hijos...
}
```

En el siguiente ejemplo se observa como un transformador primitivo inicia sesión con un sitio web.

```
post(

    :URL=>'http://host.com/login.php',
    :queryString=>'user=foo&pass=bar'

)
```

Para conectar dos transformadores se puede utilizar el carácter pipe "|" como se aprecia en el código de más abajo, donde "wget" y "xpath" son los transformadores que se conectan en serie.

```
wget | xpath('//a/@href')
```

Lo mismo se puede realizar utilizando las siguientes dos sintaxis:

```
wget
xpath('//a/@href')
```

El solo hecho de poner un transformador después de otro en líneas distintas, hace que el intérprete del harvester entienda que se trata de dos transformadores conectados en serie. Cuando los transformadores no son primitivos, y por ende están formados por varias líneas, la sintaxis para conectar transformadores en serie es la siguiente:

```
pipe{
  wget
  xpath('//a/@href')
}
```

En los tres ejemplos de códigos anteriores se obtienen los mismos resultados. El transformador "wget" extrae, limpia y normaliza el código HTML correspondiente a una url establecida con anterioridad en otro transformador. Dado que wget y xpath están conectados en serie, xpath recibe el contenido extraído por wget y realiza sobre él una búsqueda xpath. Notar que el último transformador requiere un parámetro (expresión xpath).

Si el código anterior correspondiera a un harvester, entonces utilizando la sintaxis señalada, podemos conectar dos harvester como sigue:

```
#Harvester 1
pipe{
  wget
  xpath('//a/@href')
}
#Harvester 2
pipe{
  wget
  xpath('//a/@href')
}
```

Si lo que se desea es conectar dos transformadores, pero esta vez en paralelo (ramificado), se debe hacer como sigue:

```
Wget                                        #Transformador A
branch(:BRANCH_DUPLICATED, :SCATTERED){  #Transformador B
  xpath('//a/@href') #link's url         #Transf. Hijo 1 de B
  xpath('//a') #link's text              #Transf. Hijo 2 de B
}
```

En este último código se puede apreciar como los transformadores A y B se conectan en serie y también como este último es padre de dos hijos (transformadores xpath) los cuales reciben la misma entrada producida por wget pero producen diferentes salidas. Así, el hijo 1 produce la url de un enlace mientras que el hijo 2 proporciona como salida el texto del enlace (objeto <a> del código HTML). Este mismo harvester puede también ser escrito como:

```
Wget(:BRANCH_DUPLICATED, :SCATTERED){     #Transformador B
  xpath('//a/@href')                      #Transf. Hijo 1 de B
  xpath('//a')                            #Transf. Hijo 2 de B
}
```

Para conocer más sobre un jHarvester, la API y una descripción completa de cada transformador ver (34) (56).

d) *Web Harvest:* Se trata de una herramienta para la extracción de datos desde la web escrita completamente en JAVA y de código abierto que también permite hacer Crawling. Para la extracción de datos útiles ofrece varios mecanismos, técnicas y tecnologías de manipulación de texto y XML (XSLT, XQuery y expresiones regulares). Las posibilidades de esta herramienta se pueden extender gracias a su arquitectura, la cual permite, la construcción y utilización de librerías personalizadas. En WebHarvest a los mecanismos de extracción se les llama procesadores. Cada procedimiento de extracción es definida por el usuario a través de archivos de configuración basados en XML. Cada archivo de configuración describe la secuencia de las tareas que ejecutan los procesadores con el fin de lograr el objetivo final. Los procesadores ejecutan en forma de tubería (serie) como se muestra en la figura 14. Por lo tanto, la salida de un procesador es la entrada a otro.

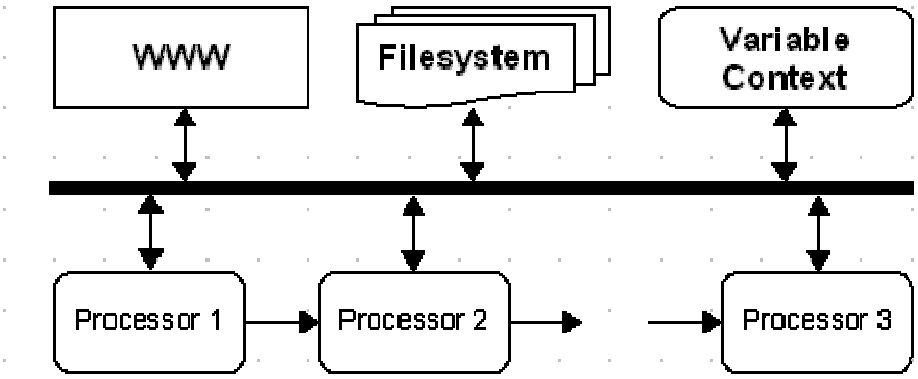


Figura 14. Conexión de procesadores en serie (65)

Un ejemplo de lo mencionado se aprecia en el siguiente archivo de configuración.

```
<xpath expression="//a[@shape='rect']/@href">
    <html-to-xml>
        <http url="http://www.somesite.com/"/>
    </html-to-xml>
</xpath>
```

Cuando Web-Harvest ejecuta este código suceden 3 cosas:

1. El procesador *html* descarga el contenido de la url especificada.
2. El procesador *html-to-xml* limpia el contenido html y produce xhtml.
3. El procesador *xpath* busca links específicos (según expresión xpath) en contenido xhtml.

La conexión es del procesador más interno hacia el externo. Web-Harvest proporciona una vasta colección de procesadores que pueden ser consultados en (65).

Puede ser usado mediante su IDE (figura 15), línea de comando (figura 16) y dentro de código JAVA (figura 17).

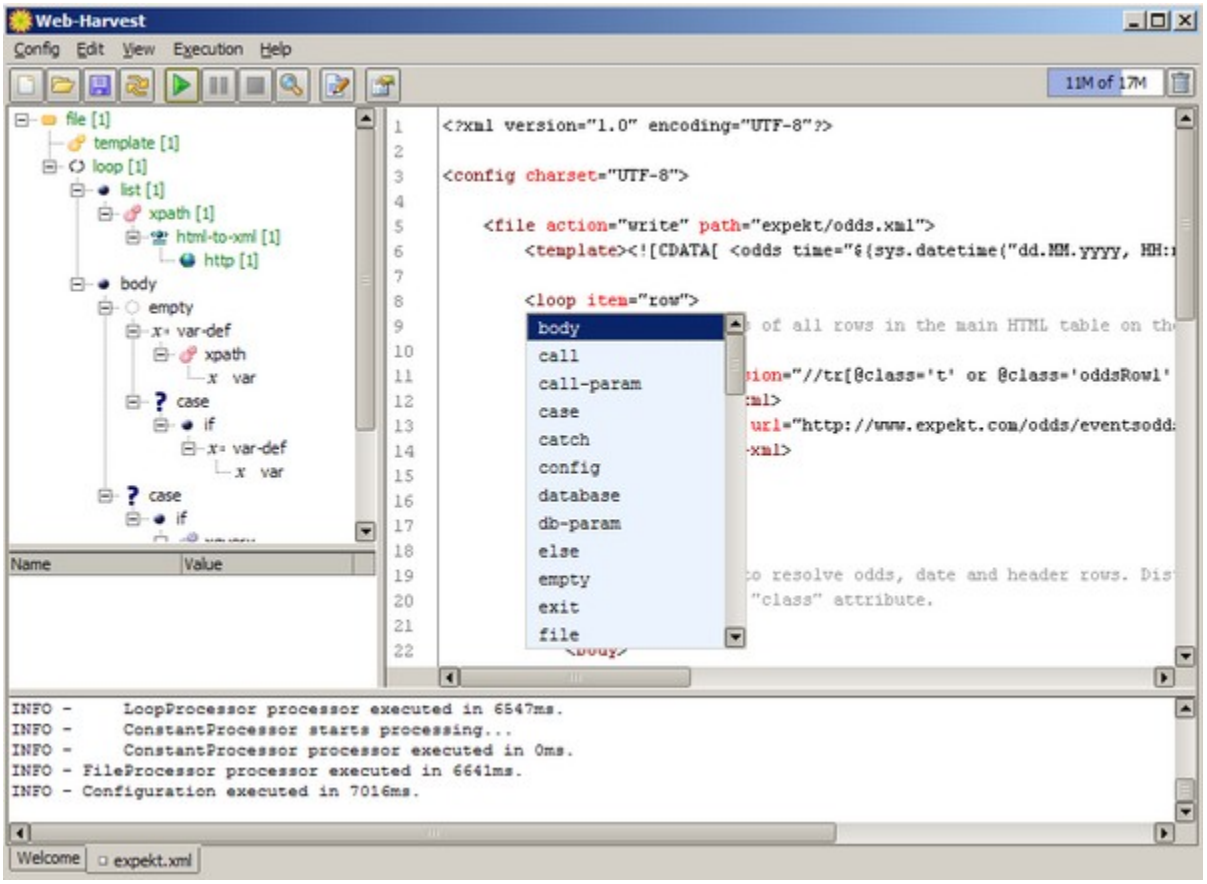


Figura 15. Web-Harvest: IDE (65)

```
java -jar webharvest_all_XX.jar [-h] config=<path> [workdir=<path>] [debug=yes|no]
          [proxyhost=<proxy server> [proxyport=<proxy server port>]]
          [proxyuser=<proxy username> [proxypassword=<proxy password>]]
          [proxynthost=<NT host name>]
          [proxyntdomain=<NT domain name>]
          [loglevel=<level>]
          [logpropsfile=<path>]
          [plugins=<list of plugin classes>]
          [#var1=<value1> [#var2=<value2>...]]
```

Figura 16. Web-Harvest: Línea de comando (65)

```java
import org.webharvest.definition.ScraperConfiguration;
import org.webharvest.runtime.Scraper;
import org.webharvest.runtime.variables.Variable;
import mypackage.MyXmlLibrary;

public class WebHarvestTest {

    public static void main(String[] args) {
        // register external plugins if there are any
        DefinitionResolver.registerPlugin("com.my.MyPlugin1");
        DefinitionResolver.registerPlugin("com.my.MyPlugin2");
        DefinitionResolver.registerPlugin("com.my.MyPlugin3");

        ScraperConfiguration config =
            new ScraperConfiguration("c:/wh/configs/news.xml");
        Scraper scraper = new Scraper(config, "c:/wh/work/");

        scraper.addVariableToContext("username", "web-harvest");
        scraper.addVariableToContext("password", "web-harvest");
        scraper.addVariableToContext("myXmlLib", new MyXmlLibrary());

        scraper.setDebug(true);

        scraper.execute();

        // takes variable created during execution
        Variable articles = (Variable) scraper.getContext().get("articles");

        // do something with articles...
    }

}
```

Figura 17. Web-Harvest: Código JAVA (65)

Las principales características de esta herramienta son las siguientes:

1. Herramienta Multi plataforma y Open Source para Harvesting y Crawling
2. Uso de variables en el contexto de ejecución
3. Uso de templates
4. API
5. IDE y Depurador para la construcción de harvester
6. Línea de comando
7. Extensa colección de transformadores
8. Posibilidad de construir e integrar a Web-Harvest procesadores a la medida
9. Conexión con base de datos vía ODBC
10. Herramienta robusta con una comunidad desarrollando constantemente

Para conocer la lista completa con las características de la última versión ver (http://web-harvest.sourceforge.net/release.php)

A modo de resumen se presenta la siguiente tabla con algunos criterios aplicados en la comparación de las cuatro herramientas. Para la evaluación de estos criterios se emplea una escala simple de 3 valores basada en una escala de Likert:

Puntos	Concepto
0	No Aplica
1	No implementado
2	Medianamente implementado
3	Completamente implementado

Tabla 7. Escala de 3 puntos

Id	Característica	Scrapy	Wire	jARVEST	Web Harvest
1	Posee API	3	1	3	3
2	Posee entorno gráfico	1	1	2	3
3	Línea de comandos	3	3	3	3
4	Selección y extracción de datos	3	1	3	3
5	Crawling	3	3	1	3
6	Exportar contenido a múltiples formatos	3	1	1	2
7	Extensible	2	2	1	3
8	Manejo de cookies	3	3	1	2
9	Compresión HTTP	3	1	1	1
10	Autentificación HTTP	3	3	3	3
11	Cache HTTP	3	1	1	1
12	Estándar Robots.txt	3	3	1	3
13	Codificación robusta	3	0	2	3
14	Uso de templates	2	0	2	3
15	Múltiples crawlers en producción	3	3	2	1

16	Estadísticas	1	3	1	1
17	Indexación y búsqueda de texto	1	3	1	1
18	Buen Soporte	3	1	2	3
19	Buena Comunidad de desarrollo	3	1	1	2
20	Baja Curva de aprendizaje	2	2	3	2
21	Documentación	3	2	2	3
22	Usado en sistemas en producción	3	1	1	3
23	Rendimiento	3	3	2	2
24	Escalable	2	3	1	1
	TOTAL	**62**	**45**	**41**	**55**

Tabla 8. Comparación de herramientas de extracción

La tabla 8 tiene el propósito de mostrar comparativamente las principales características de cuatro herramientas que fueron seleccionadas para su evaluación. Hay que mencionar que wire adquiere una puntuación no muy alta a pesar de tener cualidades muy especiales que lo hacen una alternativa seria si lo que se desea es realizar crawling de alto rendimiento en producción. Esto se debe a que es una herramienta un poco diferente a las otras tres, las que prácticamente están enfocadas al harvesting, mientras que wire está dedicado a exclusivamente al crawling de alto rendimiento.

Una de las mejores herramientas evaluadas fue Scrapy, pues permite crear tanto Crawlers como Harvester y posee cualidades que lo hacen muy versátil y potente. La tercera alternativa (jHARVEST) a pesar de no ser la mejor evaluada, dependiendo de los objetivos del harvester que se quiera construir, resulta una alternativa muy atractiva, pues es muy versátil y con una muy buena curva de aprendizaje.

La tabla 9 muestra la herramienta recomendada según tipo de uso.

		Harvesting	Crawling
Complejidad	Bajo	- jHARVEST - Web-Harvest	- Web-Harvest - Scrapy
	Medio	- jHARVEST - Web-Harvest - Scrapy	- Web-Harvest - Scrapy - Wire
	Alto	- Web-Harvest - Scrapy	- Scrapy - Wire
Rendimiento	Bajo	- jHARVEST	- Web-Harvest - Scrapy
	Medio	- Web-Harvest - Scrapy	- Web-Harvest - Scrapy - Wire
	Alto	- Web-Harvest - Scrapy	- Scrapy - Wire
Escalabilidad	Bajo	- Web-Harvest - Scrapy	- Scrapy - Wire
	Medio	- Web-Harvest - Scrapy	- Wire
	Alto	- Web-Harvest - Scrapy	- Wire

Tabla 9. Herramientas recomendadas según uso

3.2.3.4. XPath y XQuery

Un aspecto fundamental de Scrapy, Web-Harvest y jARVEST con relación a sus posibilidades para la extracción de datos semi-estructurados de páginas HTML (XHTML luego del proceso de parsing) es el uso XPath (XML Path Language) (66) o XQuery (67) (68) en el proceso de extracción. El proceso de limpieza que se realiza como parte del parsing convierte el código HTML en XHTML. Lo relevante de esto es que el conjunto etiquetas que forman el código XHTML generado se estructuran en forma de árbol n-ario y por otra parte, el nuevo código es estructurado y bien formado, lo que quiere decir que cada etiqueta de apertura posee una de cierre. Esto se puede apreciar en el siguiente fragmento de código (vehiculos.xml).

```
<lista-vehiculos>
  <vehiculo>
    <marca>Toyota</marca>
    <anio>2013</anio>
    <precio>$12,600,000</precio>
    <motor>
      <cc>1600</cc>
      <tipo>En línea</tipo>
    </motor>
    <color>Rojo</color >
  </vehiculo>

  <vehiculo>
    <marca>Peugeot</marca>
    <anio>2010</anio>
    <precio>$4,200,000</precio>
    <motor>
      <cc>1800</cc>
      <tipo>En línea</tipo>
    </motor>
    <color>Verde</color >
  </vehiculo>
</lista-vehiculos>
```

Este fragmento contiene información de una lista de vehículos (con un solo vehículo en este fragmento) que muestra la marca, año, precio, color y datos del motor; cilindrada y tipo (en línea o en "V"). La estructuran en forma de árbol n-ario se puede ver en la figura 18.

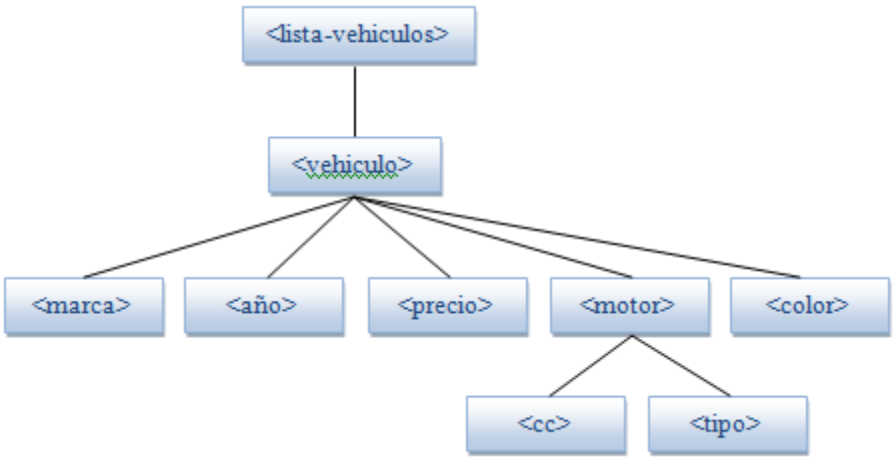


Figura 18. Estructura n-aria del fragmento XML

La relevancia de esta estructura n-aria es que cada nodo puede ser seleccionado con consultas XQuery o Xpath.

a) *XQuery:* Para efectos de entender que es XQuery, podemos usar un símil en donde XQuery es a XML lo mismo que SQL es a las Bases de Datos Relacionales. Se trata de un lenguaje funcional

(Como lo es SQL) que permite buscar y extraer elementos y atributos de documentos XML. Al ser un lenguaje funcional cada consulta es una expresión que se evalúa para luego devolver un resultado. Su principal función es extraer información de un conjunto de datos organizados como un árbol n-ario de etiquetas XML. Las expresiones y consultas realizadas con XQuery pueden ser combinadas para formar otras expresiones más complejas y potentes. XQuery ha sido construido sobre la base de Xpath, que es un lenguaje declarativo para la localización de nodos y fragmentos de información en árboles XML. XQuery se basa en este lenguaje para realizar la selección de información y la iteración a través del conjunto de datos. Xquery usa funciones en todas sus operaciones. Por ejemplo, para abrir el documento XML vehículos.xml se usa la función *doc()* como se muestra en el código:

```
doc("vehiculos.xml")
```

Si le agregamos una expresión XPath como la que se muestra en el siguiente recuadro se obtendrán todos los precios de vehículos.

```
doc("vehiculos.xml")/lista-vehiculos/vehiculo/precio
```

Esta consulta mostrará el siguiente resultado:

```
<precio>12,600,000</precio>
<precio>4,200,000</precio>
```

Al igual que SQL Xquery usa predicados para restringir los datos extraídos del documento XML. El siguiente predicado " /[anio <2013]" se usa para mostrar los vehículos de año menor a 2013 como se muestra a continuación:

```
doc("vehiculos.xml")/lista-vehiculos/vehiculo/[anio <2013]
```

Mostrando el siguiente resultado:

```
  <vehiculo>
    <marca>Peugeot</marca>
    <anio>2010</anio>
    <precio>$4,200,000</precio>
    <motor>
      <cc>1800</cc>
      <tipo>En línea</tipo>
    </motor>
    <color>Verde</color >
  </vehiculo>
```

En cambio la misma instrucción con el predicado "`[anio = 2013]`"

```
doc("vehiculos.xml")/lista-vehiculos/vehiculo/[anio <2014]
```

Producirá el resultado:

```
  <vehiculo>
    <marca>Toyota</marca>
    <anio>2013</anio>
    <precio>$12,600,000</precio>
    <motor>
      <cc>1600</cc>
      <tipo>En línea</tipo>
    </motor>
    <color>Rojo</color >
  </vehiculo>
```

Esto es claro, pues el vehículo de año 2014 no cumple con la condición establecida en el predicado.

La siguiente expresión muestra solo los elementos `<marca>` bajo el elemento `<vehiculo>` que cumple con la condición establecida en el predicado.

```
doc("vehiculos.xml")/lista-vehiculos/vehiculo/[anio <2014]/marca
```

Esto produce el siguiente resultado:

```
<marca>Toyota</marca>
<marca>Peugeot</marca>
```

La siguiente expresión FLWOR selecciona los mismos elementos que la expresión XPath anterior.

```
for $x in doc("vehiculos.xml")/ lista-vehiculos/vehiculo
where $x/anio<2014
order by $x/anio
return $x/marca
```

Notar como mediante la instrucción "**order by $x/anio**" los resultados, en caso de haber más de uno, se mostrarán ordenados por año.

FLWOR es un acrónimo para FOR, LET, WHERE, ORDER BY y RETURN.

La cláusula **FOR** selecciona todos los elementos vehículo bajo el elemento vehículos en una variable de nombre $x.

La cláusula **WHERE** selecciona solo los vehículos cuyo año sea menor a 2014

La cláusula **ORDER BY** define el orden que será aplicado a los resultados obtenidos.

La cláusula **RETURN** lo que debe ser devuelto. En el ejemplo devuelve el elemento marca de cada vehículo.

Por supuesto, es posible realizar consultas mucho más complejas que las expuestas en este pequeño ejemplo, pero no es el propósito de esta investigación presentar con detalle el funcionamiento de XQuery, sino mostrar en parte las posibilidades que justifiquen su uso en el contexto de este trabajo. Para obtener un conocimiento más profundo de este tema se sugiere ver la referencia oficial de la W3C (http://www.w3.org/TR/xquery/) en (67) (68).

b) *XPath:* Se trata de un lenguaje de expresión que se utiliza para referenciar nodos dentro de una jerarquía XML. Estas expresiones son similares a las expresiones regulares utilizadas para seleccionar pequeñas porciones de texto que coincidan con el patrón definido por la expresión regular. De manera similar, las expresiones xpath permiten recorrer y procesar un documento XML para seleccionar el o los nodos que coincidan con la expresión. El procesamiento del documento XML, o en nuestro caso XHTML, se basa en la posibilidad de direccionar cada una de las partes que lo componen, de modo que sea posible tratar cada uno de los nodos de forma diferenciada. Existen diferentes tipos de nodos. Solo por mencionar algunos están: nodo raíz, nodo elemento, nodos texto, nodos tributos y nodos comentarios.

Ejemplo:

Si consideramos el fragmento XML con información de un vehículo, algunas posibles consultas pueden ser las siguientes:

```
/lista-vehiculos/vehiculo/motor
```

Esta expresión mostrará como resultado los nodos <motor> como se muestra en el siguiente fragmento:

```
<motor>
  <cc>1600</cc>
  <tipo>En línea</tipo>
</motor>
<motor>
  <cc>1800</cc>
  <tipo>En línea</tipo>
</motor>
```

En cambio si la expresión es:

```
/lista-vehiculos/vehiculo/motor/cc
```

Se mostrará solo la cilindrada de los motores.

```
<cc>1600</cc>
<cc>1800</cc>
```

Ahora, si sobre el código HTML que corresponde a la página principal de Google realizamos la consulta "//a/@href", veremos que el resultado será una lista de 40 enlaces (figura 19), dado que

son 40 las etiquetas <a> (nodos) encontrados con la expresión. Al poner "/@href" después del nodo "a" estamos diciendo que muestre el atributo "href" del nodo, por lo tanto se entiende que @ se usa para mostrar nodos que son atributos. En consecuencia si usamos la expresión Xpath "//a" obtendremos como resultado nuevamente 40 nodos, pero esta vez correspondientes a la descripción del enlace (figura 20).

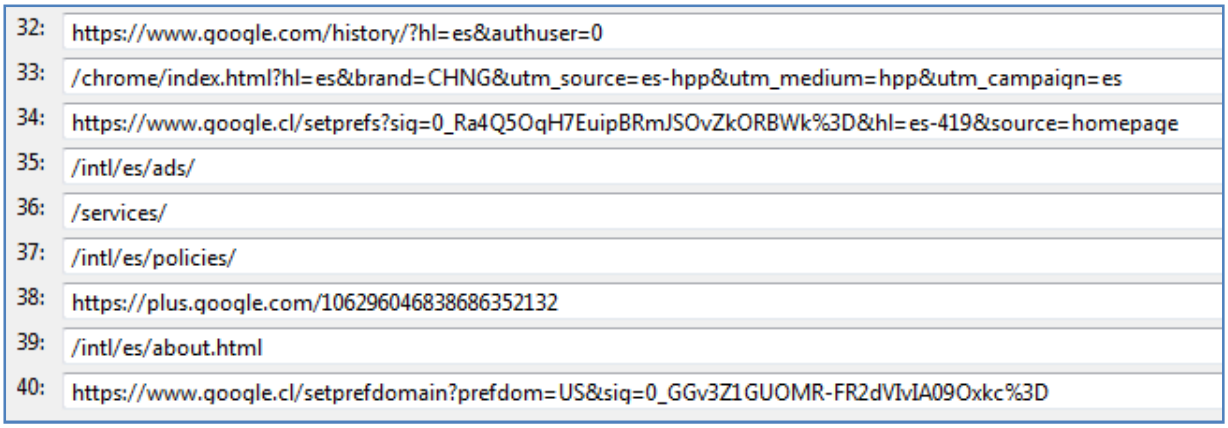


Figura 19. Resultado de la consulta XPath "//a/@href" en la página principal de Google

Figura 20. Resultado de la consulta XPath "//a" en la página principal de Google

No es el propósito de este trabajo proveer una explicación detallada y profunda sobre el funcionamiento, funciones y en general uso de XPath. Para tal efecto se recomienda ver la referencia oficial de la W3C (66) (69).

CAPÍTULO IV: Trabajo previo y resultados preliminares

4.1. Trabajo previo

Lo que se desea presentar en este trabajo es una integración de las tecnologías y técnicas disponibles actualmente para interactuar con bases de conocimiento construidas con información extraída desde la web y en virtud de ello apoyar el proceso de toma de decisiones sobre algún área de conocimiento específico. Sobre algún modelo o arquitectura que permita lo señalado recientemente, no hay trabajos conocidos. No obstante, existe en abundancia mucha investigación y desarrollo en temas específicos como Web Mining, Web harvesting o Web Scryping, Agentes de Software, Representación del Conocimiento, consultas flexibles sobre bases de conocimiento y Frameworks para trabajar con estas tecnologías. Cada uno de estos elementos por sí solo no resuelve el problema planteado, pero constituyen piezas elementales de una posible aproximación al modelo propuesto. La Web Semántica como integración tecnológica o enfoque queda excluida por las razones que se mencionan en el punto 2.1.2 (Proyección de la web). A pesar de no existir una integración similar, existe un trabajo interesante (71) que consiste en la construcción de una aplicación capaz de obtener y analizar datos de repositorios web semiestructurados. En él se aplican mecanismos de extracción, filtrado y análisis. Se realiza además una interesante aplicación de una API para el procesamiento del lenguaje natural como parte del análisis que se debe realizar sobre los datos extraídos. No obstante se trata de la construcción de una aplicación a la medida sobre un conjunto de sitios con contenido semiestructurado que difícilmente se puede extender o generalizar.

4.2. Resultados preliminares

Con los antecedentes obtenidos en los apartados anteriores, lo siguiente es responder las interrogantes de investigación planteadas en el punto 1.1.2:

1. ¿Cómo es la arquitectura y funcionamiento de un agente de software, particularmente la de un agente inteligente?
2. ¿Cómo es la arquitectura de internet actualmente y cómo se vislumbra en el futuro?
3. ¿De qué manera los agentes de software o agentes inteligentes pueden ser utilizados para la extracción de datos desde la web?
4. ¿Hay otros mecanismos aparte de los AI que puedan ser utilizados eficientemente en la extracción y procesamiento de datos desde la web?

Respondemos a las cuatro interrogantes de manera conjunta como sigue: Internet es una enorme red donde se aplica arquitectura cliente/servidor, está constituida por una gran cantidad de nodos (servidores) los cuales se encuentran conectados alrededor del mundo. La información que nos interesa "aprovechar" se almacena como parte de decenas de millones de páginas web alojadas en dichos nodos. Estas páginas se guardan en forma de documentos HTML que dan formato al contenido para que pueda ser presentado apropiadamente en los navegadores (clientes), pero su estructura y organización interna no es capaz de contener, junto con el contenido, la semántica de este. Por otra parte, la información contenida en los documentos HTML se mezcla con el código HTML de cada página web, lo cual dificulta su obtención automática. Además, por la enorme cantidad de páginas y datos existente, una exploración manual a gran escala resulta inviable. En esencia, la arquitectura de la web mucho no ha cambiado, sí su uso y las tecnologías que circundan en torna a esta. Así por ejemplo, la web en sus inicios se presenta como una web estática que muestra únicamente contenido y se utiliza como una vitrina electrónica donde los usuarios consumen la información presentada en el sitio web. Un canal de información entre la entidad o persona que desea mostrar un contenido mediante un conjunto de documentos HTML y el internauta que los observa en su navegador. En la web 2.0 esto cambió básicamente en el sentido de hacer al internauta parte de la dinámica responsable de la generación de contenido, es decir, ahora el usuario de internet produce y genera contenido como resultado del fenómeno de las redes sociales y otras muchas operaciones que realiza. Por otra parte, muchas aplicaciones de cómputo y sistemas de diversos tipos han migrado de sus plataformas tradicionales a la web, lo que produce aun más información. La abundancia de información en conjunto con las tecnologías emergentes y las favorables capacidades de procesamiento de computadores y dispositivos móviles dan forma a un conjunto de nuevas necesidades con relación al

uso de la web como por ejemplo: Uso de lenguaje natural para realizar búsquedas cada vez más precisas, codificación de los contenidos de tal manera que sean encontrados e interpretados por organismos de software y que estos a su vez puedan inferir y producir nueva información. En definitiva, estas nuevas necesidades se esbozan dentro de un marco que propone la siguiente generación de la web, la web semántica. Esta, resume el conjunto de nuevas necesidades en una web futura cuyas páginas estén organizadas, estructuradas y codificadas de tal manera que los computadores sean capaces de efectuar inferencias y razonar a partir de sus contenidos (Web + Inteligencia Artificial) y por otra parte, también como una web que pueda ser concebida como una gran base de datos. A pesar de existir mucha investigación en esta área, su implementación completa por ahora se encuentra un poco distante, pues solo existen desarrollos concretos en los tres primeros niveles de la arquitectura propuesta para la web semántica y por otra parte, requiere cambios profundos en toda la web. Además, suponiendo que contáramos hoy con desarrollo completo en cada una de los niveles de la arquitectura que propone la web semántica, sería indispensable reescribir completamente todas las páginas web si lo que se desea es satisfacer hoy el conjunto de nuevas necesidades. En este escenario, donde la web se empalma con componentes inteligentes, los agentes inteligentes se perfilan como elementos protagónicos. Pero aunque la web semántica no sea una realidad inmediata, la información existente puede ser aprovechada, solo requiere de un proceso de extracción previo en lugar de consultarla directamente. En este sentido los agentes inteligentes, y en general agentes de software, pueden ayudar bastante en la tarea de extraer información desde internet. La arquitectura de un agente inteligente posibilita la automatización de ciertas tareas aun cuando el entorno en el que operan sea dinámico, incluso, dependiendo del nivel de inteligencia pueden aprender, modificar su comportamiento y moverse en la red. Específicamente con relación a la extracción de datos, aparecen los harvester, arañas o web-bots, que son un tipo de agente de software que responde a las técnicas existentes de webharvest o webscraping. Para la implementación de estos mecanismos de extracción existen numerosas herramientas desde APIs y Frameworks hasta implementaciones altamente configurables que pueden utilizarse invirtiendo un poco de tiempo en configuración y puesta a punto.

El apartado anterior tiene el propósito de responder en conjunto las interrogantes de investigación planteadas en el punto 1.1.2 con base en la revisión parcial de la literatura realizada como parte fundamental de esta investigación. De esta forma, los antecedentes recolectados en la investigación permitirán proponer idóneamente los elementos que conformarán nuestro “Modelo experimental para la adquisición de conocimiento y toma de decisiones a partir de información extraída desde la web”.

CAPÍTULO V: Solución propuesta

5.1. Solución propuesta

Para poder apoyar algún proceso de toma de decisiones con información existente en la web, como solución inicial proponemos el modelo mostrado en la figura 21, formado por 5 capas, las cuales hacen posible construir bases de conocimiento con información extraída desde la web para apoyar el proceso de toma de decisiones sobre algún área de conocimiento específica.

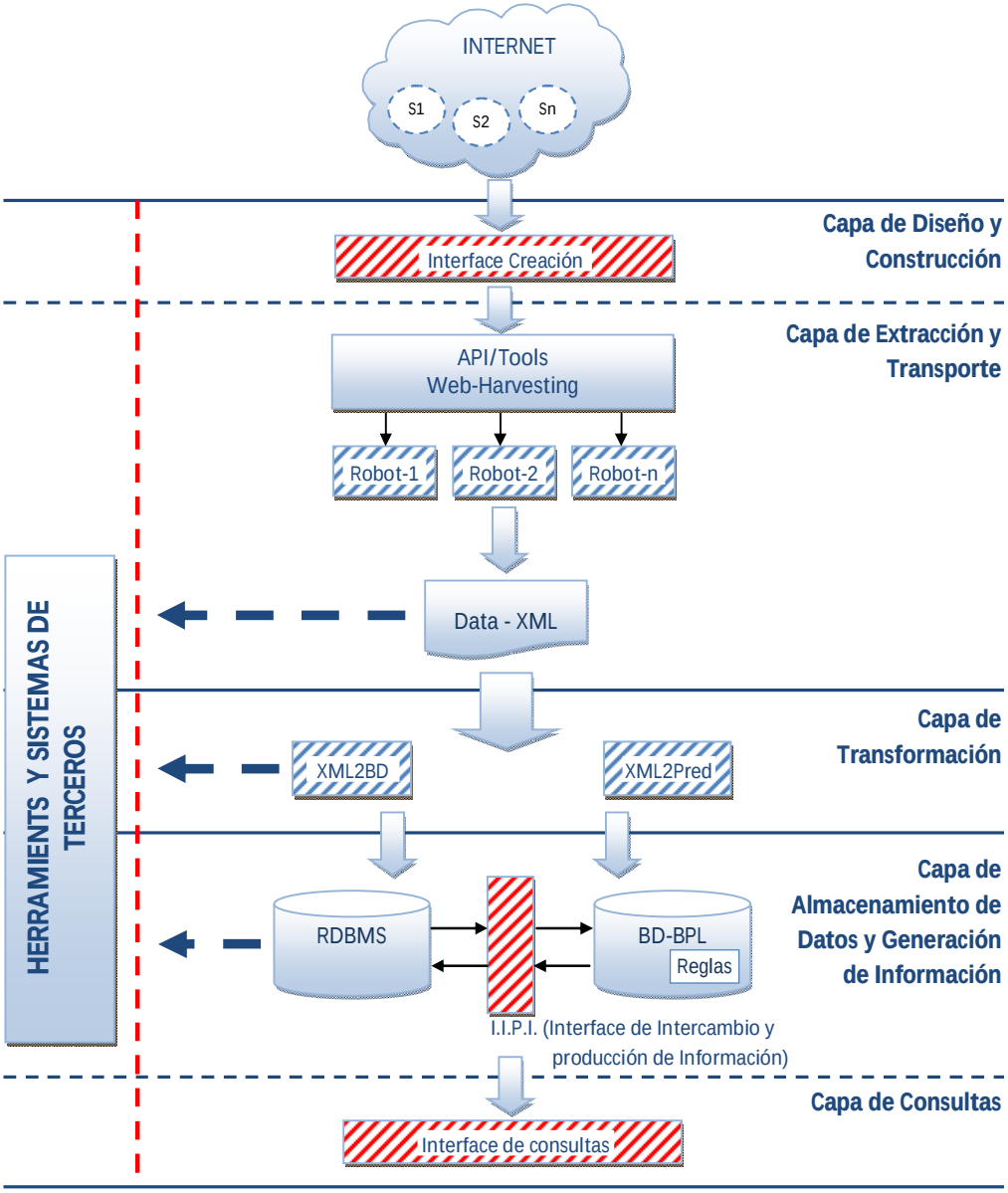


Figura 21. Arquitectura de la solución propuesta

El modelo de nuestra solución se construye utilizando una arquitectura n-capas (basada en la arquitectura cliente/servidor) con el propósito de lograr una clara separación en los distintos tipos de desarrollo que componen el sistema. De esta manera las piezas de una determinada capa del modelo podrán desarrollarse y mejorarse en el tiempo sin que las demás capas se vean afectadas. Además, permite que el modelo sea escalable y pueda ser ampliado con facilidad en caso de que las necesidades aumenten. Este aspecto es crucial, pues no todos los artefactos que componen la solución se encuentran completamente desarrollados y los que sí lo están, con seguridad pueden ser mejorados o reemplazados.

La solución propuesta integra tecnologías y técnicas para la extracción de datos desde la web, XML como trasporte de datos, una Base de Datos Relacional, piezas de software que realizan la transformación de datos y un componente lógico-deductivo que posibilita la deducción y la realización de consultas utilizando términos lingüísticos sobre la información extraída. Cada uno de estos elementos se encuentra distribuido en las siguientes 5 capas.

5.1.1. Capa 1 (*Diseño y construcción*)

En esta capta se proporcionan los artefactos de software necesarios que actúan como interface gráfica entre el usuario y el API utilizada para la construcción de los harvester. En esta capa existe un solo componente:

a) *Interface de Creación:* La interface debe permitir diseñar el robot y asistir en su construcción. Dado que los mecanismos de extracción utilizados se basan en el uso de expresiones XPath, una de las tareas que se deben realizar como resultado de esta interacción es la construcción de las expresiones XPath necesarias. No obstante, el conjunto de tareas que debe efectuar o asistir son las siguientes:

1. Cargar el documento HTML para explorar sus estructuras
2. Generar las expresiones XPtath que se requieran
3. Representar gráficamente los harvester indicando claramente su entrada y salida
4. Ensamblar harvester más complejos a partir de harvester existentes.

Finalmente la interacción del usuario con la interface proporciona el código del robot que será interpretado en la siguiente capa por el API correspondiente.

Notar que en la figura 21 aparece una línea horizontal segmentada separando la primera capa de la siguiente. Esto quiere decir que la capa Diseño y Construcción no se encuentra implementada pero es necesaria, y como tal forma parte del trabajo futuro.

5.1.2. Capa 2 (*Extracción y Transporte*)

En esta capta se lleva a cabo el proceso de extracción como resultado de poner los harvester creados en un ambiente de producción. En esta misma capa, de manera predefinida se utiliza XML como estándar para el transporte de datos, pues estos al momento de ser extraídos deben guardarse utilizando alguna estructura. Los elementos que componen esta capa son los siguientes:

a) API/Toosl Web-Harvesting: Los robots pueden ser construidos utilizando diversas APIs como las presentadas en el apartado 2.3.3, pero seleccionamos jARVEST como herramienta para construir los robots o harvesters. La elección se realizó utilizando dos criterios, curva de aprendizaje y versatilidad. En definitiva este componente provee una capa de abstracción para construir los robots sin programarlos directamente en algún lenguaje de propósito general, sino utilizando la API en cuestión. Esto supone una construcción, modificación y reutilización más limpia y la posibilidad de ensamblar robots más complejos.

b) Robots o harvesters: El nombre harvester deriva de la acción de usar una técnica de extracción llamada harvesting (del inglés que traducido quiere decir cosechar), la cual se usa principalmente con contenido semi-estructurado (27), es decir, listados, directorios, clasificados y toda aquella información en donde sea posible identificar algún tipo de estructura y/o patrones de cadenas de texto. Esta técnica basada en el análisis del código HTML requiere la programación de robots o harvester que se encarguen de dicho análisis (construidos utilizando la API/Tools).

El robot o conjunto de robots encargado de la extracción se compone de una serie de piezas atómicas proporcionadas por la API, en esta investigación llamadas transformadores, las cuales realizan una tarea específica. Cada trasformador recibe una entrada con la cual produce una salida. Tanto la entrada como la salida son cadenas de texto. Las operaciones de los transformadores son variadas, pueden ir desde la trasformación de una URL a su contenido HTML, hasta búsquedas de patrones en el código usando XPATH o expresiones regulares. Para componer el robot final los transformadores se conectan en serie o en paralelo como se muestra en el punto 2.3.3. El robot final es una serie de transformaciones sobre una entrada de texto.

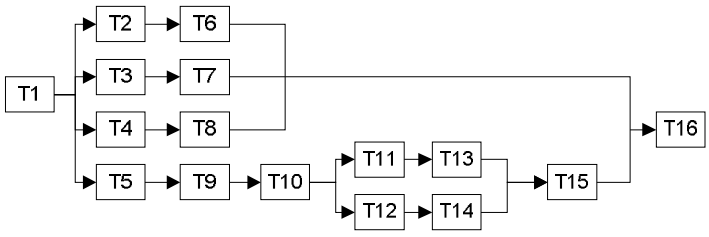


Figura 22. Harvester productor del fragmento XML asociado a la figura 18

La figura 18 representa la estructura un documento XML construido con los datos extraídos por el robot de la figura 22. El trasformador T1 (Wget) recibe como entrada una URL y entrega el contenido del documento HTML correspondiente, el cual se pasa como entrada a los trasformadores T2, T3, T4 y T5, los que mediante una consulta XPATH obtienen elementos de texto específicos como marca, año y precio del vehículo. La consulta XPATH que realiza T5 obtiene una nueva URL (formateada en T9) que es la entrada de otro Wget (T10) el cual entrega como salida un nuevo contenido HTML del cual se extraen dos datos más (color (T11) y cilindrada (T12)), los cuales se formatean como elemento XML. Finalmente, la salidas de los transformadores conectados en paralelo se mezclan primeramente en T15 y luego en T16. La salida final es un documento XML con muchos registros <vehiculo></vehiculo> como los del código que representa la estructura n-aria de la figura 18.

Una vez construidos el o los robots, estos deben ser llevados a un ambiente de producción con las condiciones idóneas para un buen rendimiento, a pesar de que en términos muy generales, los robots construidos con APIs o Frameworks basados en JAVA, no tienen alto rendimiento al requerir de doble interpretación. El código de los robots es interpretado por la API y el código de esta es interpretado por la máquina virtual.

En el anexo 2 se puede ver el código completo del harvester construido para nuestro caso de estudio.

c) *Data - XML:* Este componente es un documento XML que contiene los datos extraídos por los robots. El contenido de este documento se organiza como una colección de registros de la forma:

```
<Tabla>
  <tupla-1>
    <atributo-1>[valor]</atributo-1>
       [<atributo-1.1>
          <atributo-1.n>[valor]</atributo-1.n>
       </atributo-1.1>]
  <atributo-n>[valor]</atributo-n>
  </tupla-1>

  <tupla-2>
   ...
  </tupla-2>

  <tupla-n>
   ...
  </tupla-n>
</Tabla>
```

Ejemplo de esto es el fragmento XML cuya estructura es representada por la estructura n-aria de la figura 18. El único propósito de este elemento es contener y transportar los datos extraídos utilizando el estándar XML.

5.1.3. Capa 3 (*Transformación*)

En esta capa los datos extraídos son transformados en ANSI-SQL y Predicados BPL, mediante los cuales es posible construir una BD relacional y una Base Deductiva (Base de Datos Bousi~Prolog (72) (72)) respectivamente, lo que en conjunto forman una Base de Conocimiento. En esta etapa intervienen los siguientes elementos:

a) Conversor XML a BD: Al igual que el *Conversor XML a Predicados BPL*, este componente utiliza como entrada el mismo documento XML y genera un script SQL ANSI-92 compuesto por dos partes. La primera contiene sentencias DDL para generar la estructura de la Base de Datos Relacional y la segunda, instrucciones DML para insertar los datos en la estructura generada. Esto puede ser visto como un modelo relacional que organiza las tuplas del documento XML. Entonces, este componente actúa como un "Normalizador de datos asistido", pues los datos almacenados en el documento XML están organizados como registros de una sola estructura (tabla) donde es posible encontrar atributos que claramente pertenecen a diversas entidades. Así por ejemplo, el fragmento vehiculos.xml puede ser visto como una base de datos sin normalizar con una única tabla de nombre vehículo como se muestra en la tabla 10.

Vehiculo					
Marca	**Anio**	**Precio**	**cc**	**Tipo_motor**	**Color**
Toyota	2013	12,600,000	1600	En línea	Rojo
Peugeot	2010	4,200,000	1800	En línea	Verde

Tabla 10. Datos de vehículos.xml sin normalizar

La misma información puede ser representada mediante las tablas 11(a), 11(b) y 11(c):

Marca	
idMarca	**nombre**
1	Toyota
2	Peugeot

Tabla 11. (a) Datos de vehículos.xml normalizado

Tipomotor	
idTipomotor	**nombre**
1	En línea
2	En V

Tabla 11. (b) Datos de vehículos.xml normalizado

Vehiculo						
idVehiculo	**idMarca**	**Anio**	**Precio**	**Motor_cc**	**idTipomotor**	**Color**
1	1	2013	12,600,000	1600	1	Rojo
2	2	2010	4,200,000	1800	1	Verde

Tabla 11. (c) Datos de vehículos.xml normalizado

Aquí se puede apreciar como las tablas se relacionan mediante el uso de sus claves primarias, relaciones que se observan con mayor claridad en la figura 23.

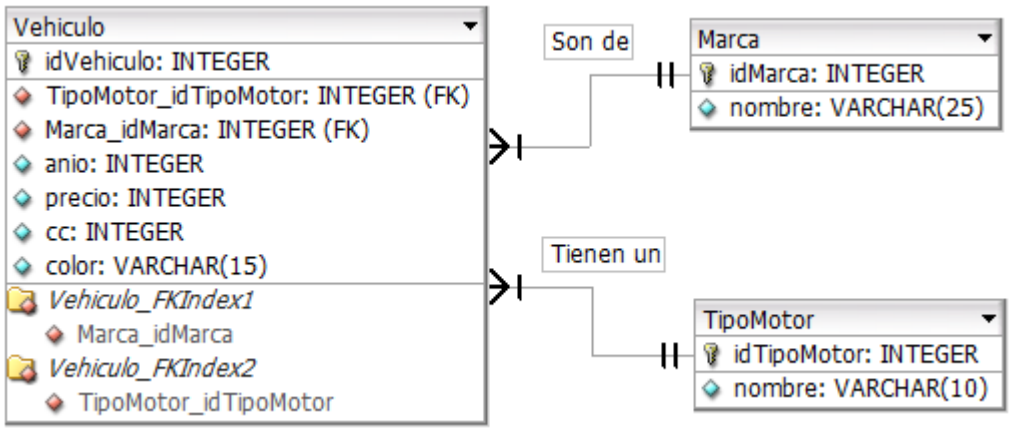


Figura 23. Representación relacional del fragmento vehiculos.xml

Para que la transformación se lleve a cabo y se genere un modelo como el de la figura 23, lo primero es seleccionar el documento XML, “vehiculos.xml” para efecto de este ejemplo, el cual contiene dos registros.

Luego de seleccionar y cargar el documento XML los nombres de nodos (campos candidatos) se cargan en una lista de campos disponibles, de donde se seleccionan todos aquellos que serán utilizados en el modelo de datos. En este caso todos los campos son utilizados, pues el modelo que queremos construir los requiere (ver figura 24).

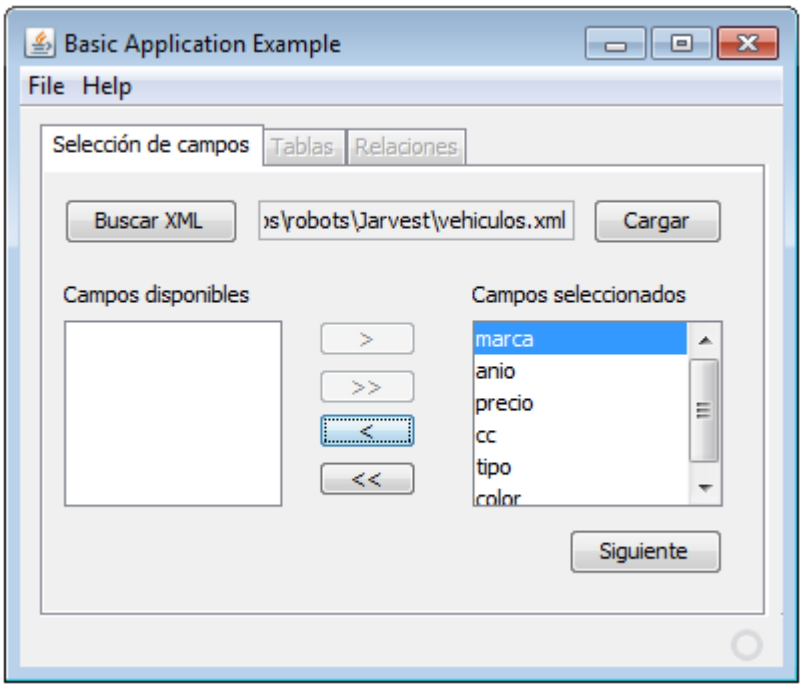


Figura 24. XML2BD – Carga de vehiculos.xml y selección de campos

Lo siguiente es crear las tablas utilizadas en el modelo. Esto lo realizamos en la pestaña "Tablas" de la aplicación a la cual accedemos presionando el botón siguiente como se muestra en la figura 25.

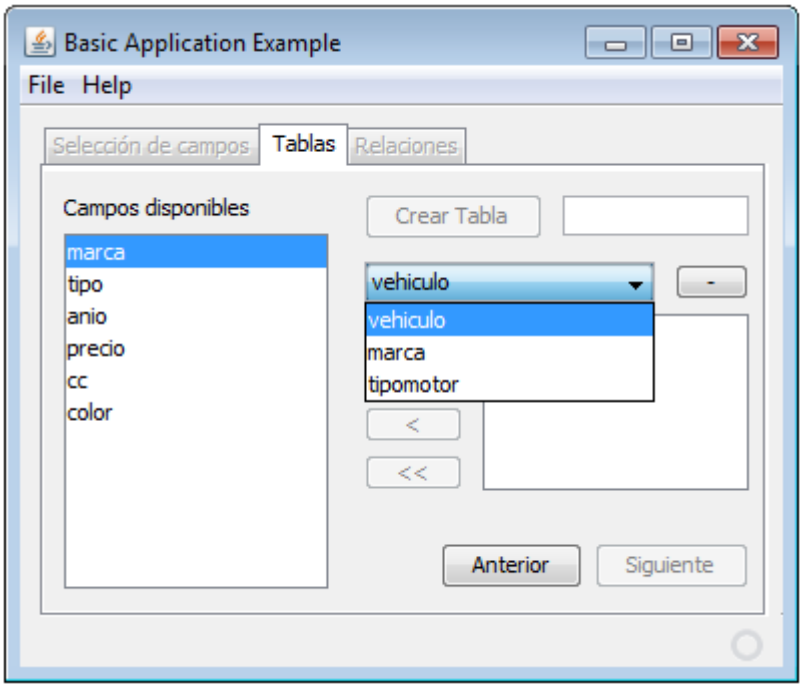


Figura 25. XML2BD – Creación de tablas

Luego de crear las tres tablas que serán utilizadas (*vehiculo, marca y tipomotor*), a estas se les debe asociar al menos un campo de la lista de campos disponibles. Así, los campos *anio*, *precio*, *cc* y *color* son asociados a la tabla *vehiculo* (figura 26), el campo *marca* se asocia a la tabla *Marca* y finalmente el campo *tipo* a la tabla *Tipomotor*.

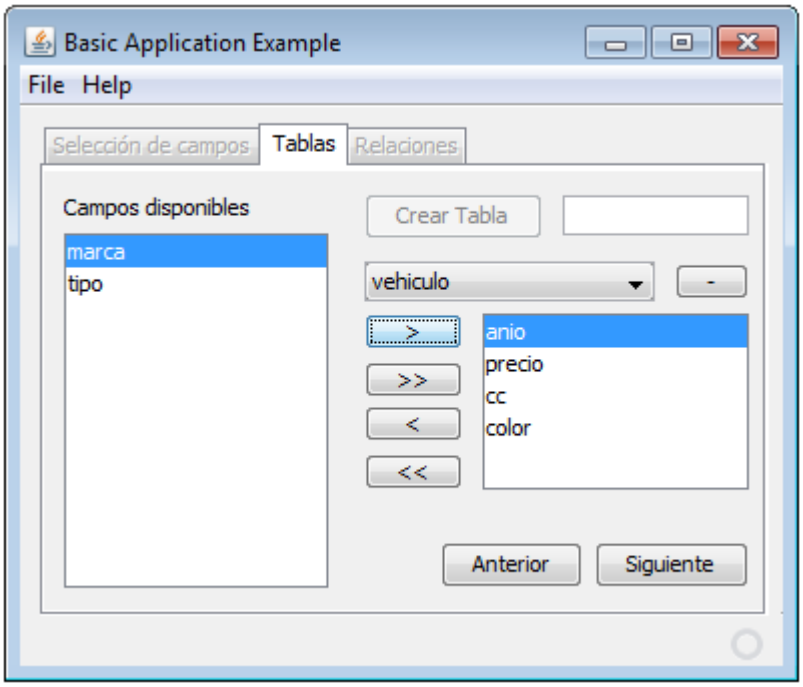


Figura 26. XML2BD – Asociación de campos a tablas

Luego de asociar los campos con sus respectivas tablas, lo siguiente es establecer las relaciones que existen entre las tablas. Para ello, la aplicación en la pestaña "Relaciones", a la cual se accede presionando en el botón "Siguiente" de la pestaña Tablas, proporciona tres listas: La primera y la tercera con tablas disponibles y la segunda con los tipos de relaciones para asociar las tablas seleccionadas (figura 27). Cada relación creada forma parte de una cuarta lista que contiene todas las relaciones del modelo, las cuales en caso de ser necesario se pueden eliminar y volver a crear utilizando los controles "+" y "-".

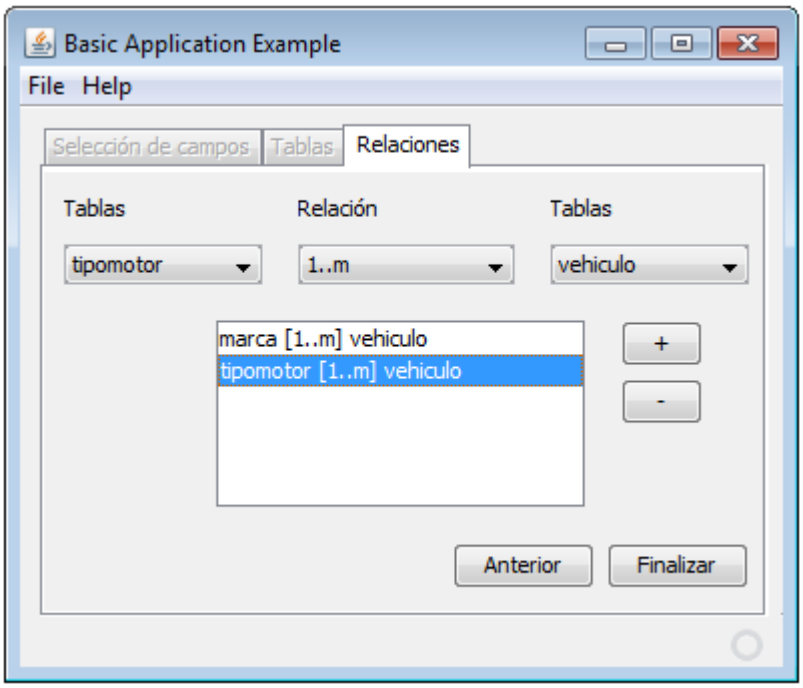


Figura 27. XML2BD – Creación de relaciones entre tablas

Finalmente al presionar el botón "Finalizar" la aplicación generará un archivo SQL con la estructura (Modelo) y los datos contenidos en el documento vehículos.xml como se muestra en los recuadros siguientes:

```
# Estructura del documento vehiculos.xml

CREATE TABLE vehiculo (
idvehiculo VARCHAR(36),
anio NUMERIC,
precio NUMERIC,
cc NUMERIC,
color VARCHAR(200),
idmarca VARCHAR(36),
idtipomotor VARCHAR(36),
FOREIGN KEY (idmarca) REFERENCES marca,
FOREIGN KEY (idtipomotor) REFERENCES tipomotor,
PRIMARY KEY (idvehiculo)
);

CREATE TABLE marca (
idmarca VARCHAR(36),
marca VARCHAR(200),
PRIMARY KEY (idmarca)
);

CREATE TABLE tipomotor (
idtipomotor VARCHAR(36),
tipo VARCHAR(200),
PRIMARY KEY (idtipomotor)
);
```

```
# Tabla: vehiculos
INSERT INTO vehiculos (idvehiculo, anio, precio, cc, color, idmarca,
idtipomotor) VALUES ('1a369e36-038d-48c0-a3bf-efe02af4b0b5', 2013,
12600000, 1600, 'Rojo','7779256e-a5d0-44fc-9797-f5710b13c934',
'3b50f444-3b02-4903-85e8-3f8360a11ee2');

INSERT INTO vehiculos (idvehiculo, anio, precio, cc, color, idmarca,
idtipomotor) VALUES ('6a81483b-5e46-439a-b705-b3d25dcae3da', 2010,
4200000, 1800, 'Verde','c2258b19-994a-4b32-a972-b8b851407e0e',
'3b50f444-3b02-4903-85e8-3f8360a11ee2');

# Tabla: marca
INSERT INTO marca (idmarca, marca) VALUES ('7779256e-a5d0-44fc-9797-
f5710b13c934', 'Toyota');

INSERT INTO marca (idmarca, marca) VALUES ('c2258b19-994a-4b32-a972-
b8b851407e0e', 'Peugeot');

# Tabla: tipomotor
INSERT INTO tipomotor (idtipomotor, tipo) VALUES ('3b50f444-3b02-
4903-85e8-3f8360a11ee2', 'En linea');

INSERT INTO tipomotor (idtipomotor, tipo) VALUES ('03e4c1da-1aeb-
4e9e-8312-13ab0fc9c734', 'En V');
```

Por último, el SQL generado debe ser cargado en el RDBMS para que la Base de Datos Relacional sea construida.

b) *Conversor XML a Predicados BPL:* Este componente trabaja utilizando como entrada el documento XML producido por los transformadores para generar como salida un conjunto de predicados BPL. Una vez cargado el documento XML, el paso siguiente es seleccionar las variables lingüísticas que serán objeto de consultas y transformar cada registro (tupla) en un predicado BPL equivalente. La primera tarea se realiza de manera asistida la interface gráfica de la aplicación como se indica en la figura 23. Una vez que se ha seleccionando el o los atributos, tras presionar el botón “Transformar”, la aplicación continúa sin intervención con la trasformación de cada tupla almacenada en el documento en un predicados BPL.

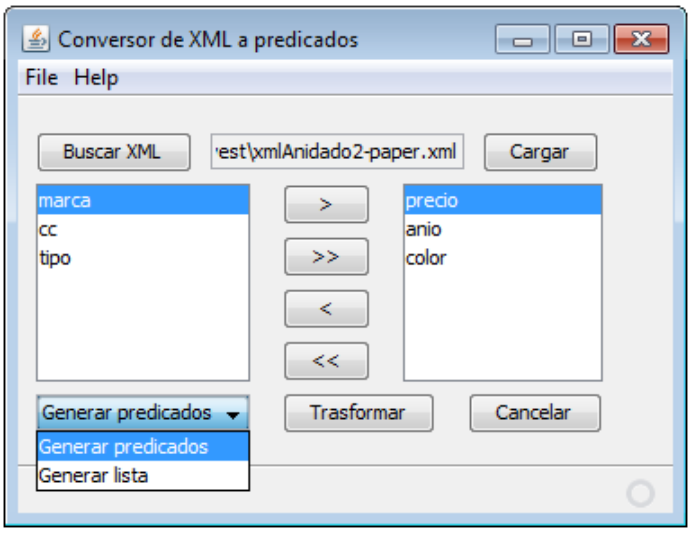


Figura 28. Conversor XML a predicados Bousi~Prolog

Se puede observar cómo luego de buscar y cargar un documento XML se muestra una lista con los atributos disponibles (Lista izquierda), de la cual se seleccionan los atributos precio, anio y color para generar los predicados (o lista) BPL que constituirán el conocimiento preciso de la Base de Datos Bousi~Prolog. Al presionar el botón Transformar los predicados serán generados y guardados en un archivo de nombre *nombreArchivo.xml.bpl*. Luego de un mensaje confirmando el éxito de la operación, para el ejemplo de la figura 23 se generará el archivo vehiculos.xml.bpl. Los predicados generados son los siguientes:

```
vehiculo(anio#2013,precio#12600000,color('Rojo')).
vehiculo(anio#2010,precio#4200000,color('Verde')).
```

De haber seleccionado todos los atributos, el predicado generado sería:

```
vehiculo(marca('Toyota'),anio#2013,precio#12600000,
motor(cc#1600,tipo(en_linea)),color('Rojo')).

vehiculo(marca('Peugeot'),anio#2010,precio#4200000,
motor(cc#1800,tipo(en_linea)),color('Verde')).
```

Una vez generados los predicados estos deben ser cargados en Bousi~Prolog (figura 29), a partir de donde será necesario modelar la imprecisión y establecer el conjunto de reglas necesarias (otra parte del conocimiento preciso) para realizar las consultas lingüísticas. Ver en anexo 4 un fragmento de los predicados generados para nuestro caso de estudio.

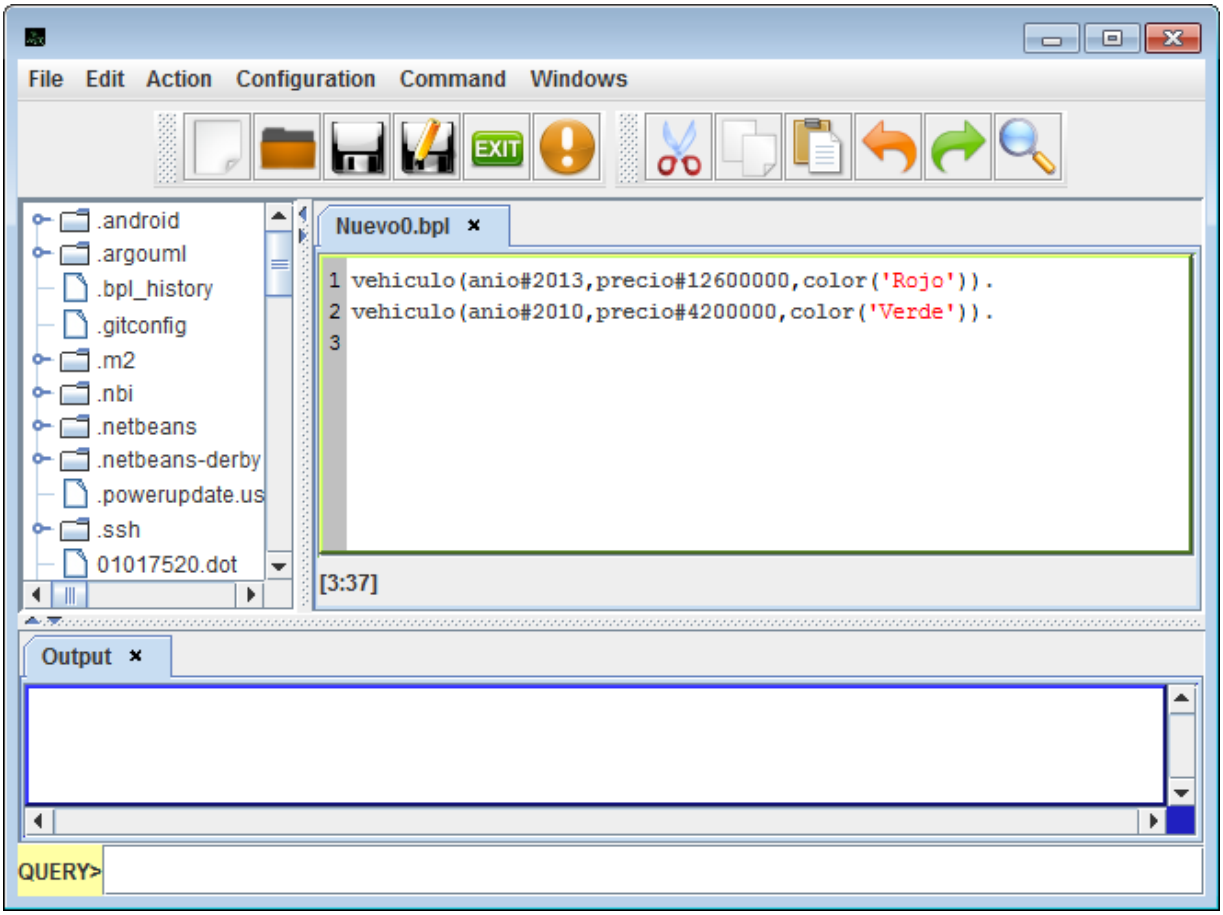


Figura 29. Conversor XML a predicados Bousi~Prolog

5.1.4. Capa 4 (*Almacenamiento de Datos y Generación de Información*)

En esta capa, a partir del proceso de transformación de la capa anterior, se construyen las Bases de Datos, en donde mediante consultas y reglas es posible generar conocimiento. Además, posibilita la formulación de consultas directas sobre los datos incorporando variables lingüísticas. Aquí operan los siguientes elementos:

a) Sistema administrador de Base de Datos Relacionales (RDBMS): A pesar de que la base de conocimientos en estricto rigor se construye con los hechos, reglas y relaciones de proximidad (73) en la Base de Datos Bousi~Prolog, el RDBMS, alternativamente también complementa la base de conocimiento con más datos precisos (datos resúmenes), ofreciendo además la posibilidad de realizar consultas directas tradicionales sobre los datos.

Dependiendo de cuales sean las necesidades de implementación, el RDBMS puede ser prácticamente de cualquier fabricante, de propietario u open-source, pues para crear la Base de Datos basta con cargar y ejecutar el SQL ESTANDAR generado en la capa anterior.

b) *Base de Datos Bousi~Prolog (BD-BPL) o componente Lógico-Deductivo:* Nuestro componente lógico-deductivo, Bousi~Prolog (BPL, por sus siglas) (73) (72), es una extensión del lenguaje Prolog estándar consistente en un marco de programación lógica basada en proximidad. Su semántica operacional es una adaptación del principio de resolución SLD donde la unificación clásica ha sido sustituida por un algoritmo de unificación borrosa basado en relaciones de proximidad. Informalmente, el algoritmo de unificación débil establece que dos términos $f(t_1,\ldots,t_n)$ y $g(s_1,\ldots,s_n)$ unifican débilmente si los símbolos raíz f y g se aproximan y sus argumentos t_i y s_i unifican débilmente. Por lo tanto, el algoritmo de unificación débil no falla si hay un encuentro de dos símbolos sintácticamente distintos siempre que sean aproximados, es decir, obtiene éxito con un cierto grado de aproximación. Este componente debe dar soporte a necesidades que implican tareas relacionadas con la utilización del sistema como Sistema Basado en Reglas, y por otro lado, a otro grupo de tareas que en conjunto permitan realizar consultas flexibles sobre la información extraída. El conjunto de tareas que dan sustento a ambas necesidades son las siguientes:

1. *Modelar la imprecisión:* En nuestro contexto la imprecisión se debe modelar con el fin posibilitar la realización de consultas flexibles utilizando términos lingüísticos establecidos en la capa 3 mediante el componente "Conversor XML-BD". Intuitivamente, la imprecisión y la vaguedad aparecen cuando no sabemos qué valor exacto elegir para un atributo. Un conjunto difuso (74) se puede utilizar para representar información vaga y/o imprecisa. Existen dos líneas de trabajo en la utilización de conjuntos difusos aplicados a las Bases de Datos, esto es de particular interés, ya que nuestra fuente de datos es un documento XML (Capa 2 del modelo) que puede ser visto como una Base de Datos. En la primera línea de trabajo el modelo de datos se deja intacto y el procesamiento de consulta se extiende para permitir conceptos vagos que están representados por conjuntos difusos y/o variables lingüísticas (75) (76) (77). En la segunda línea de trabajo se aplica la teoría de conjuntos difusos para modificar el modelo de datos relacional (78) (79) (80). El primer planteamiento parece ser más práctico y prometedor ya que el modelo relacional sigue siendo la alternativa más utilizada. Otros enfoques adoptan técnicas de agrupamiento como una herramienta para generar un mapeo entre los términos difusos, definidos como un nivel de abstracción mayor, y el registro de la base de datos (81) (82). Mientras que la primera opción requiere de una especificación exacta del significado de cada término difuso conocido por el sistema, el segundo hace que la tarea de definición sea más fácil ya que en lugar de definir explícitamente el rango de valores conceptuales correspondientes a cada término, el usuario solo define el orden relativo de los términos lingüísticos, y el sistema, mediante el

algoritmo de clustering hará coincidir cada término lingüístico con el adecuado grupo de registros (82). El componente lógico-deductivo (Bousi~Prolog) emplea ambos mecanismos. En definitiva, el modelamiento de la imprecisión se consigue realizando dos tipos de construcciones: primero, relaciones entre los conceptos lingüísticos y los datos precisos del documento XML y segundo: la definición de relaciones de proximidad. Para ambas es necesario especificar cuáles son las variables lingüísticas, pues estas se utilizan en las construcciones anteriores. Una variable lingüística (83) es una quíntupla $\langle X, T(X), U, G\rangle$, donde: X es el nombre de la variable, $T(X)$ es el conjunto de términos lingüísticos de X (es decir, el conjunto de nombres de valores lingüísticos o etiquetas lingüísticas de X), U es el dominio o universo de discurso, G es una gramática que permite generar $T(X)$ y M es una regla semántica que asigna a cada término lingüístico x en $T(X)$ su significado (es decir, un subconjunto difuso de U caracterizado por su función de pertenencia μ_x). Lo habitual es hacer distinción entre términos atómicos (términos primarios) y términos compuestos que se componen de términos primarios. Los subconjuntos borrosos (74) que M aplica a los términos compuestos se calculan, mientras que los que se aplican a los términos primarios son definidos (en un manera subjetiva y dependiente del contexto). En Bousi~Prolog, para una variable X, sólo el dominio U y los subconjuntos borrosos asociados a los términos lingüísticos primarios en $T(X)$ se consideran por su definición, el resto de términos compuestos se calculan automáticamente. Por otra parte, no hace una distinción léxica entre los componentes sintácticos y semánticos de X. Por lo tanto, hace uso de dos directivas para definir y declarar la estructura de una variable lingüística X. La directiva de dominio permite declarar y definir el dominio asociado a una variable lingüística. La sintaxis de esta directiva es:

```
:-domain(Nombre_Dominio(n,m,Magnitud)).
```

Donde, *Nombre_Dominio* es el nombre del dominio, n y m (con $n < m$) son los límites inferior y superior del subintervalo real $[n, m]$, y la *magnitud* es el nombre de la unidad en la que se miden los elementos del dominio. La directiva fuzzy_set permite declarar y definir una lista de subconjuntos difusos (Los que se asocian a los términos principales de una variable lingüística) en un dominio predefinido. La sintaxis de esta Directiva es:

```
:-fuzzy_set(
Nombre_Dominio,[SubS_1(a1,b1,c1[,d1]),…,SubS_n(an,bn,cn[,dn]])).
```

Los Subconjuntos borrosos son definidos indicando su nombre, subs_i, y el tipo de función de pertenencia (funciones trapezoidales, si se dan cuatro argumentos, o funciones triangulares, si se utilizan tres argumentos). También es importante determinar el tipo de términos difusos.

Consideramos que un término difuso se puede clasificar como: un descriptor numérico cualitativo, un descriptor no numérico cualitativo o una descripción de cuantificación. Un descriptor numérico cualitativo es una palabra que describe algún valor numérico o un rango de valores numéricos. Un descriptor cualitativo no numérico es una palabra que describe un concepto no numérico. Una descripción de cuantificación es una palabra que describe la cantidad de respuestas deseadas en una consulta en lenguaje natural que se refiere únicamente a los descriptores numéricos cualitativos.

Bousi~Prolog permite efectuar dos tipos de consultas lingüísticas: consultas lingüísticas difusas y consultas flexibles basadas en proximidad.

i. *Consultas lingüísticas difusas*: Se realizan sobre datos numéricos como edad, peso, estatura, cilindrada de un vehículo, etc. Para realizar la consulta es necesario construir conjuntos borrosos, los que se asocian a términos primarios de una variable lingüística. Los conjuntos deben construirse sobre un domino definido previamente. Por ejemplo, para la variable lingüística *Edad* es posible definir los términos primarios *joven*, *adulto* y *viejo* con los conjuntos borrosos *(0,0,20,30), (20,40,55,70) y (60,80,100,100)* respectivamente definidos sobre el dominio *edad* cuyos límites inferior y superior se definen arbitrariamente como 0 y 100 respectivamente y *años* como la unidad de medida (85).
   Para el ejemplo de la variable lingüística *Edad* las instrucciones BPL serían las siguientes:

```
:-domain(edad(0,100,años)).

:-fuzzy_set(edad,[joven(0,0,20,30),adulto(20,40,55,70),
viejo(60,80,100,100)]).
```

   Después de esto será posible realizar consultas flexibles como ¿Quienes son jóvenes? mediante la sentencia:

```
?-person(X,joven).
```

ii. *Consultas flexibles basadas en proximidad*: Este tipo de consultas en cambio, se realizan sobre datos no numéricos como *rojo*, *sedan*, *station*, *Santiago*, etc. Una relación de proximidad se denota por la ecuación $a \sim b = \alpha$, la cual representa una entrada de una relación binaria difusa, donde *a* y *b* son símbolos de predicados y α es el grado de proximidad. Se dice entonces que *a* y *b* se relacionan en un grado α (80) (72). De esta manera, si disponemos de registros de vehículos almacenados conteniendo entre sus

atributos *color*, podemos establecer las siguientes ecuaciones de proximidad: (rojo ~ negro = 0.6.), (rojo ~ amarillo= 0.4.), (amarillo ~ celeste = 0.8.) y (blanco ~ negro = 0.1.). Cualquiera sea el tipo de consulta, finalmente todo se traduce en términos de compilación a relaciones de proximidad, pues la semántica operacional de Bousi~Prolog es una adaptación del principio de resolución donde la unificación clásica se reemplaza por un algoritmo de unificación borrosa basado en relaciones de proximidad.

2. *Modelar y almacenar conocimiento:* Bousi~Prolog calcula respuestas como grados de aproximación haciendo una clara distinción entre el conocimiento preciso y el vago. El conocimiento preciso en BPL es especificado por un conjunto de hechos y reglas Prolog, mientras que el conocimiento vago es principalmente especificado por un conjunto de lo que llamamos ecuación de proximidad, las cuales definen relaciones difuso-binarias que expresan cuán cerca están dos conceptos. Como ejemplo, un fragmento de una base de datos que almacena información sobre las personas. Supongamos algunos subconjuntos difusos sobre el dominio edad, de la que se han obtenido los grados de proximidad entre las etiquetas lingüísticas joven, adulto y viejo. Este conocimiento puede ser codificado por un conjunto de ecuaciones de proximidad, como lo muestra el siguiente fragmento de una base de datos deductiva.

```
% HECHOS Y REGLAS (Datos, CONOCIMIENTO IMPRECISO)
edad(mary, adulto).
edad(sam, joven).
edad(john, viejo).

% REGLAS (Inferencia, CONOCIMIENTO IMPRECISO)
amigo(X,Y):-edad(X,Z), edad(Y,Z), X \= Y.

% ECUACIONES DE PROXIMIDAD (CONOCIMIENTO VAGO O IMPRECISO)
joven ~ adulto = 0.75.
viejo ~ joven = 0.25.
adulto ~ viejo = 0.75.
```

En este sentido, lo que conseguimos como salida de la capa 3 es conocimiento preciso en forma de predicados BPL como los que se ven en el fragmento de código anterior.

3. *Inferir:* El mecanismo de inferencia que utiliza Bousi~Prolog está basado en proximidad, concepto que se modela mediante ecuaciones de proximidad. El concepto de relación difusa fue introducido por Zadeh en (74). Una relación difusa binaria en un conjunto U es un subconjunto difuso en U x U (es decir, un mapeo de U x $U \rightarrow$ [0, 1]). Se dice que una relación difusa binaria R es una relación de proximidad si satisface la propiedad reflexiva (es decir, $R(x, x) = 1$ para cualquier $x \in U$) y la propiedad simétrica (es decir, $R(x, y) = R(y, x)$ para cualquier x, y

$\in U$). Si además se tiene la relación transitiva (es decir, $R(x, z) \geq R(x, y)\ \Delta\ R(y, z)$ para cualquier $x, y, z \in U$; donde el operador 'Δ' es una t-norma arbitraria), se le llama relación de similitud. Un ejemplo de uso del mecanismo de inferencia es la consulta que hace uso de los datos, reglas y ecuaciones de proximidad. Observe el código siguiente:

```
?-amigo (mary, sam).
```

La pregunta "Si mary es amiga de sam" nos permite obtener la respuesta "Sí con 0.75". Pero la respuesta no es posible de obtener únicamente con los datos, es necesario modelar la imprecisión y el resto del conocimiento preciso (reglas).

c) *1nterface de Intercambio y Generación de Información (I.I.P.I.):* La función de esta interface, en general, es proporcionar datos y datos resúmenes con los cuales se construyen reglas (85) para formar la base de conocimiento (86) Bousi~Prolog. Ej.
   - vehículo puesto en venta más de una vez por distintos dueños (Discriminar con marca, color, año y cilindrada) → Posible problema mecánico
   - patente se encuentra más de una vez en vehículos diferentes (Discriminar con marca, color, año y cilindrada) → Inconsistencia (posible fraude)

   Notar como el tipo de información que forma el antecedente de cada regla resulta muy simple de conseguir en una Base de Datos Relacional mediante consultas SQL estándar, por esta razón además de proporcionar una lista de predicados (hechos) para Bousi~Prolog, se proporción un Base de Datos Relacional equivalente.

5.1.5. Capa 5 (*Interface de consultas*)

Cada sistema de base de datos de la capa anterior incorpora su propia interface para realizar consultas sobre los datos, no obstante en esta capa se proporciona un único elemento:

a) *Interface de consultas:* Se trata de una interface general para realizar consultas sobre la base de conocimiento construida en la capa 4 (BD Relacional + BD Bousi~Prolog). Esta es una de las interfaces no implementadas aun y dada su importancia será parte del trabajo futuro de esta investigación.

Sin tener la intención de presentar el modelo como un Sistema Experto Basado en Reglas, es posible identificar en él los componentes de un sistema tal: *datos* o *base de hechos*, *conocimiento* y *motor de inferencia*; pues los datos extraídos transformados a predicados BPL en la capa 3 no son otra cosa que los *hechos*. Y Bousi~Prolog por otra lado, es un lenguaje de programación lógica basado en

proximidad que incorpora un *mecanismo de inferencia* que entre otras cosas utiliza *reglas* a partir de de las cuales es posible hacer deducciones, es decir, datos, base de conocimiento y motor de inferencia, de modo tal que una vez extraídos los datos y estableciendo las reglas y consultas apropiadas, el sistema podrá ser empleado como un sistema experto basado en reglas. No obstante si se omiten las reglas, mediante las interfaces de consulta (propias de la capa 4 o específicamente la capa 5) el usuario podrá realizar consultas directas sobre la información extraída. En ambos casos lo que se obtiene es nueva información, y con ella, conocimiento para apoyar los procesos de toma de decisiones en un área de conocimiento específica.

Un aspecto importante de las consultas mencionadas y la razón de utilizar un componente lógico-deductivo como Bousi~Prolog, es que al estar basado en proximidad (80), las consultas pueden incluir términos lingüísticos (83), y de esa forma, dotarlas de flexibilidad, lo que no sucede con las consultas SQL tradicionales sobre base de datos relacionales. Estos lenguajes de consulta (SQL) se han diseñado para recuperar información a partir de bases de datos que contienen datos precisos. Para que el usuario pueda efectuar consultas sobre los datos, debe proporcionar un conjunto de condiciones de selección que requieren comandos estrictos con parámetros y valores precisos. En muchas situaciones, los usuarios quienes requieren consultar los datos no están completamente familiarizados con estos, por lo tanto, es habitual que una consulta sea formulada por un experto que traduce una pregunta en lenguaje natural en restricciones precisas sobre valores rígidos, lo que implica la posibilidad de omitir respuestas de interés. Por otro lado, una parte de los conocimientos no se incorpora de una manera apropiada en los sistemas de almacenamiento de hoy en día (77), ya que la información del mundo real a menudo es permeada por la vaguedad y/o imprecisión. La rigidez de los mecanismos de consulta dificulta la recuperación de información de manera flexible. En este sentido, las técnicas basadas en la teoría de conjuntos difusos (74) que incorpora BPL son muy útiles para modelar la vaguedad e imprecisión, permitiendo la utilización de mecanismos que posibilita la recuperación de los datos de estos sistemas mediante consultas que contienen términos lingüísticos (81), es decir, consultas flexibles sobre información extraída desde la web usando el estándar XML como mecanismo de transporte de datos y el modelamiento de la imprecisión mediante la construcción de una Base de Datos Borrosa (Hechos y reglas de BPL).

5.2. Implementación del modelo (Caso de estudio)

Las pruebas del modelo se realizan en el contexto de la compra y venta de vehículos usados. En general, la implementación del modelo concentra varias tareas:

a) *Identificación de la o las fuentes de datos:*

Esta tarea se debe realizar manualmente sobre la web. Se requiere explorar el tipo y la calidad del contenido de algunos sitios como:

1. www.chileautos.cl
2. www.elrastro.cl
3. www.demotores.cl
4. www.mercadolibre.cl/vehiculos
5. www.yapo.cl

Para esta prueba se escogió www.chileautos.cl por ser un directorio que mantiene información bastante detallada y además existe una cantidad importante de vehículos a la venta. Una vez seleccionada la fuente de información, es necesario seleccionar la información que se quiere extraer del sitio. Este sitio concentra la información de cada vehículo en dos páginas, la primera es un listado paginado de vehículos donde se muestran 26 vehículos por página y aproximadamente 1650 páginas, lo que resulta en una cantidad de 41250 vehículos.

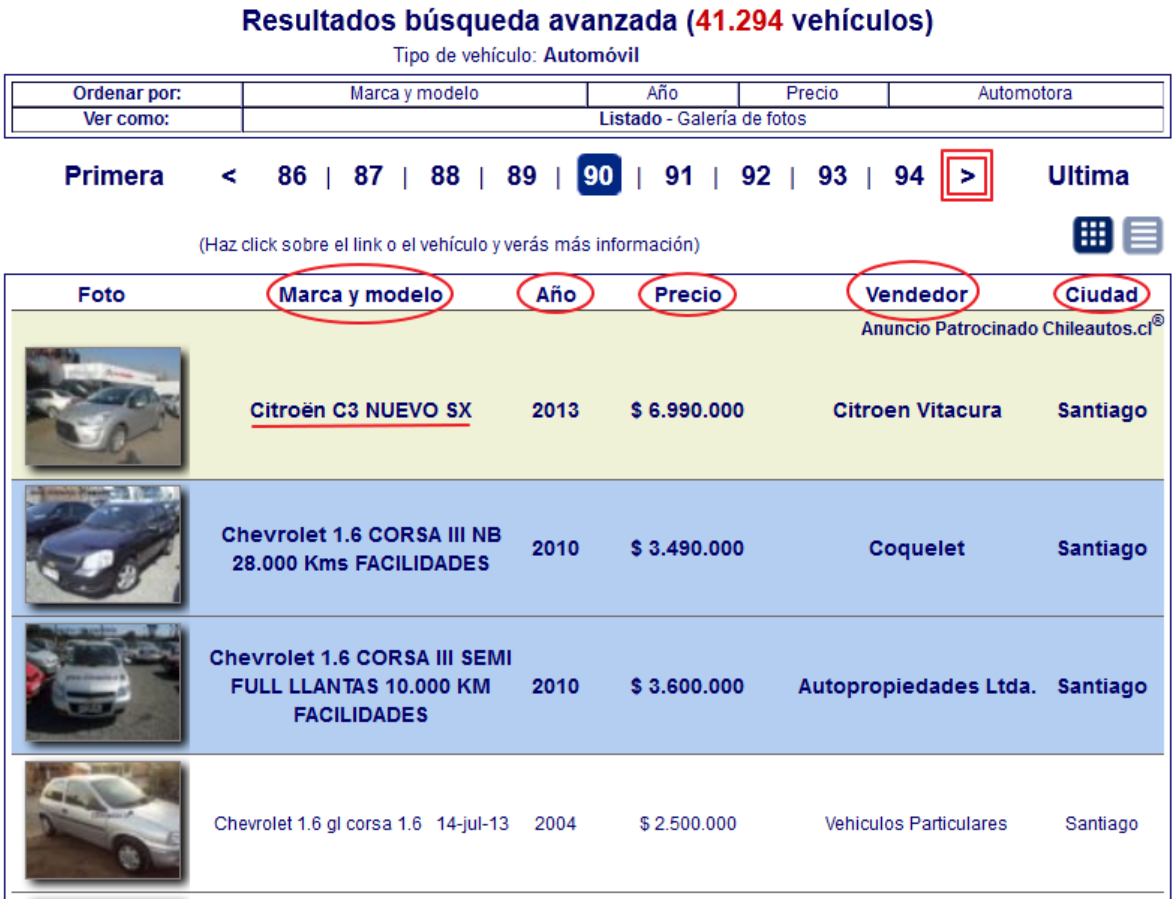


Figura 30. Listado paginado de vehículos

La figura 30 muestra el listado principal de vehículos con información de ellos, no obstante, al hacer click en marca y modelo (Ej. Citroen C3 NUEVO SX) se carga una segunda página con información más de tallada (Figura 31, 32 y 33).

Figura 31. Información de tallada de vehículo seleccionado (a)

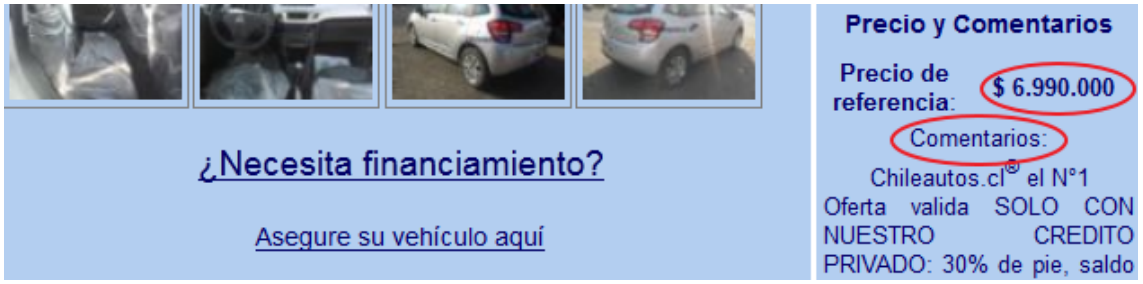


Figura 32. Información de tallada de vehículo seleccionado (b)

Figura 33. Información de tallada de vehículo seleccionado (c)

La información de ese sitio puede ser enriquecida si se cruza con información de otro sitio, por ejemplo, la url de más abajo corresponde a un sitio que permite saber si un determinado vehículo mantiene multas impagas, como se muestra en la figura 34.

```
http://consultamultas.srcei.cl
```

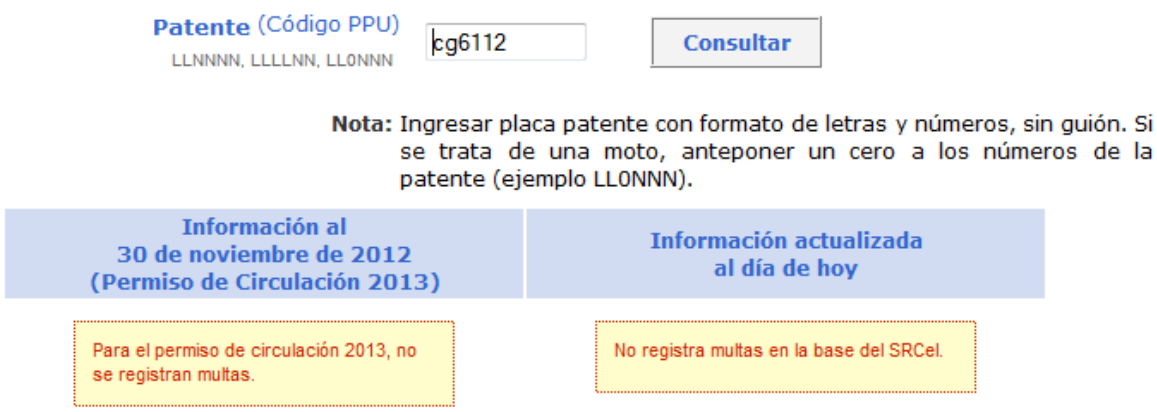


Figura 34. Registro de multas de tránsito no pagadas

Luego de identificar la información que se desea extraer, el paso siguiente es diseñar el o los robots que realicen el proceso de extracción.

b) *Extracción y conversión de los datos:*

La construcción del o los harvesters requieren de una planificación y diseño donde se establezca claramente cuáles son los elementos de información que se desean extraer y como se llevará a cabo la extracción. En particular resulta indispensable saber qué consulta XPath permite obtener los elementos de interés. Por otra parte, el harvester es un programa estructurado formado por estructuras de control simples (selección e iteración) en donde un diseño se hace necesario en la medida que su complejidad aumenta.

En las figuras 30, 31 y 32 todos los elementos encerrados en un óvalo rojo corresponden a contenido que se desea extraer. En la figura 30, el elemento subrayado corresponde a un enlace, que al seleccionar conduce a una segunda página con información detallada del vehículo.

Todos los vehículos particulares tienen publicada su patente y con este dato podemos diseñar el robot para que tome este dato y lo utilice para consultar otra información que se encuentre disponible en otro sitio. Por ejemplo, la url de más abajo proporciona información sobre multas impagas del vehículo con patente CG6112.

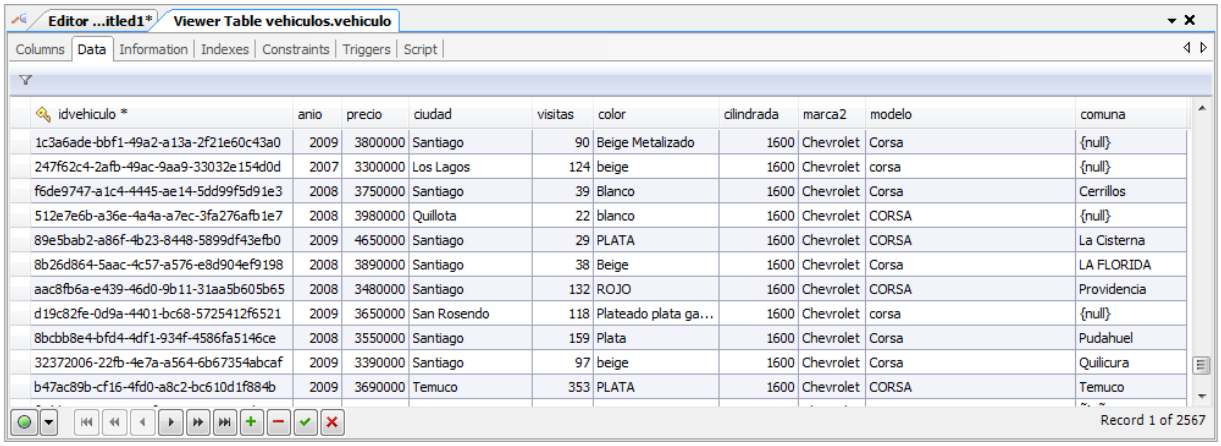
```
http://consultamultas.srcei.cl/ConsultaMultas/buscarConsultaMultasExterna.do?ppu=cg6112
```

Otro aspecto interesante que debe ser considerado es que, como se mencionó, existen alrededor de 1650 páginas a las cuales el robot debe acceder sistemáticamente. Para ello utilizamos el enlace ">" (marcado con un doble cuadrado rojo) el cual conduce a la página siguiente. De la misma forma en que accedimos a la página con información detallada del vehículo, lo hacemos para acceder a la página siguiente del directorio. Lo relevante es que cada elemento que se desea extraer se obtiene como resultado de una consulta XPath, por lo tanto, los elementos que deseamos extraer, las consultas XPath asociadas y la utilización de otros sitios para complementar la información, deben ser el resultado de una planificación y diseño del harvester.

El harvester completo se puede apreciar en el anexo 2. El resultado final de la ejecución del harvester es un documento xml con toda la información seleccionada en la etapa de diseño y construcción.

c) *Construcción de la Base de conocimientos:*

La construcción de la base de conocimiento se compone de dos tareas principales: Primero, la generación de SQL y predicados BPL para ser importados en la base de datos relacional y Bousi~Prolog respectivamente y segundo, el modelamiento de la imprecisión. La primera parte como ya se ha mostrado en la descripción del modelo (Capa 3 (Transformación)) se consigue utilizando las herramientas XML2BD(Ver en el anexo 5) y XML2Pred (Ver en el anexo 4). Una vez obtenidos el SQL y los Predicados, estos deben ser cargados en una Base de Datos Relacional (figura 35) y Bousi~Prolog respectivamente (figura 36).



idvehiculo *	anio	precio	ciudad	visitas	color	cilindrada	marca2	modelo	comuna
1c3a6ade-bbf1-49a2-a13a-2f21e60c43a0	2009	3800000	Santiago	90	Beige Metalizado	1600	Chevrolet	Corsa	{null}
247f62c4-2afb-49ac-9aa9-33032e154d0d	2007	3300000	Los Lagos	124	beige	1600	Chevrolet	corsa	{null}
f6de9747-a1c4-4445-ae14-5dd99f5d91e3	2008	3750000	Santiago	39	Blanco	1600	Chevrolet	Corsa	Cerrillos
512e7e6b-a36e-4a4a-a7ec-3fa276afb1e7	2008	3980000	Quillota	22	blanco	1600	Chevrolet	CORSA	{null}
89e5bab2-a86f-4b23-8448-5899df43efb0	2009	4650000	Santiago	29	PLATA	1600	Chevrolet	CORSA	La Cisterna
8b26d864-5aac-4c57-a576-e8d904ef9198	2008	3890000	Santiago	38	Beige	1600	Chevrolet	Corsa	LA FLORIDA
aac8fb6a-e439-46d0-9b11-31aa5b605b65	2008	3480000	Santiago	132	ROJO	1600	Chevrolet	CORSA	Providencia
d19c82fe-0d9a-4401-bc68-5725412f6521	2009	3650000	San Rosendo	118	Plateado plata ga...	1600	Chevrolet	corsa	{null}
8bcbb8e4-bfd4-4df1-934f-4586fa5146ce	2008	3550000	Santiago	159	Plata	1600	Chevrolet	Corsa	Pudahuel
32372006-22fb-4e7a-a564-6b67354abcaf	2009	3390000	Santiago	97	beige	1600	Chevrolet	Corsa	Quilicura
b47ac89b-cf16-4fd0-a8c2-bc610d1f884b	2009	3690000	Temuco	353	PLATA	1600	Chevrolet	CORSA	Temuco

Figura 35. SQL cargado en SGBD

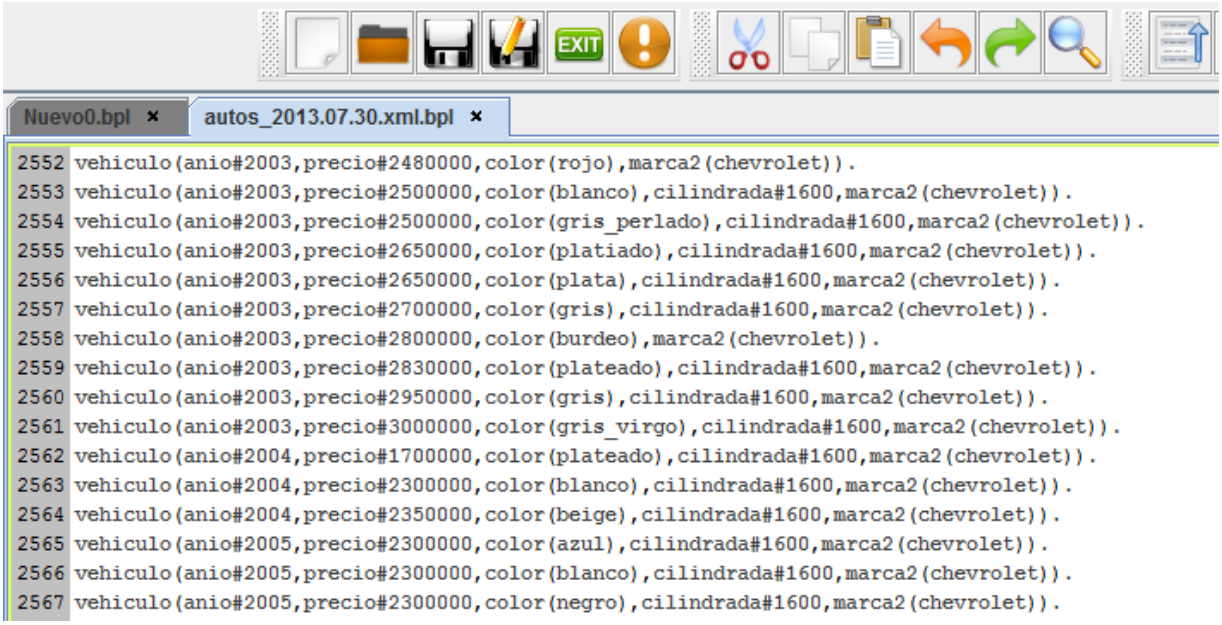


Figura 36. Predicados cargados en Bousi~Prolog

La segunda tarea tiene principalmente el propósito dar soporte a las consultas flexibles mediante la utilización de conjuntos borrosos y ecuaciones de proximidad (Explicado en el punto 1.4 del capítulo V).

```
% CONJUNTOS DIFUSOS PARA MODELAR EL TERMIN OLINGUISTICO AÑO

:-domain(anio(1970,2013,years)).
:-fuzzy_set(anio,[
      viejo(1970,1970,1980,1987),
      medio_nuevo(1982,1985,1998,2004),
      nuevo(2000,2005,2013,2013)]).

% ECUACIONES DE PROXIMIDAD PARA COLORES
rojo ~ negro = 0.6.
rojo ~ amarillo = 0.4.
amarillo ~ celeste = 0.8.
blanco ~ negro = 0.1.
```

d) Obtención de información para la toma de decisiones:

Una vez que la Base de Conocimiento (BD-Relacional + BD-BPL) están constituidas es posible interactuar con el sistema para obtener nueva información, como por ejemplo:

1. ¿Cuál es el color de vehículo que más se vende?
2. ¿Cuáles son los vehículos semi nuevos?
3. ¿Cuáles vehículos mantienen multas impagas?

4. ¿En qué rango de precios se venden más vehículos?
5. ¿Cuáles son los vehículos claros a la venta?

Algunas de estas preguntas se pueden responder con la BD relacional mientras que otras con BPL. Al responder por ejemplo la pregunta 1, mediante un par de consultas SQL obtenemos los datos de la tabla 12, representados también gráficamente en la figura 37.

Color	Cantidad
Blanco	580
Otros	539
Gris	500
Rojo	290
Azul	190
Plateado	155
Negro	126
Verde	116
Plomo	22
Amarillo	17

Tabla 12. Cantidad de ventas por color

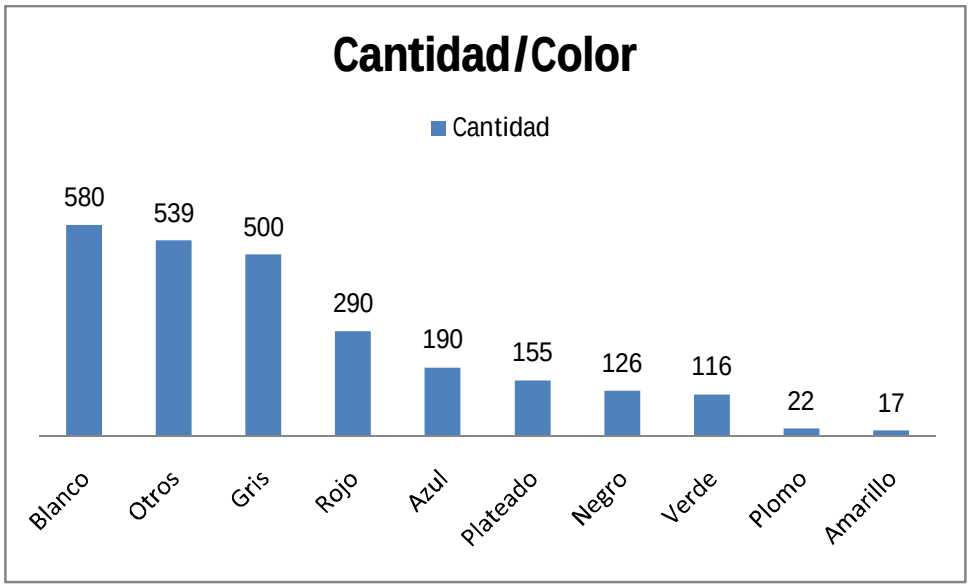


Figura 37. Cantidad de ventas por color

De manera similar, para la pregunta 4 obtenemos los datos de la tabla 13 y figura 38, donde es posible apreciar datos resúmenes de acuerdo según intervalos de precios. Señalar además que el mayor precio registrado fue $36.500.000.

Millones	Cantidad
[0-2]	450
[2-4]	1382
[4-6]	519
[6-8]	61
[8-10]	53
[10-15]	53
[15-20]	25
[20-25]	14
[25-50]	10

Tabla 13. Cantidad de ventas por rango-precio

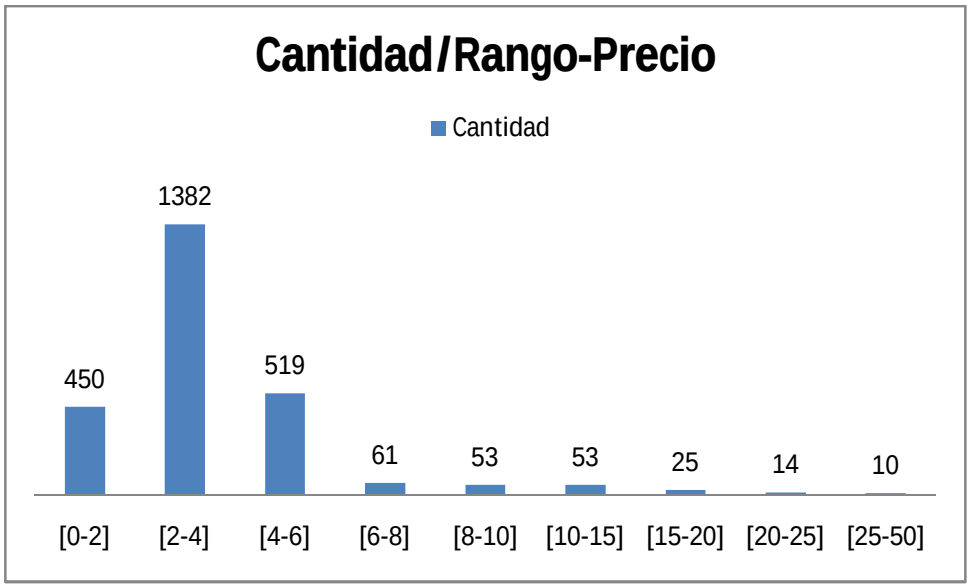


Figura 38. Cantidad de ventas por rango-precio

Para responder la pregunta 2 “¿Cuáles son los vehículos semi nuevos?” usamos BPL, pues requiere de soporte flexible, ya que al utilizar el concepto “semi nuevo” es necesario manejar la imprecisión.

```
% CONJUNTOS DIFUSOS PARA MODELAR EL TERMINO LINGUISTICO AÑO

:-domain(anio(1970,2013,years)).
:-fuzzy_set(anio,[
      viejo(1970,1970,1980,1987),
      medio_nuevo(1982,1985,1998,2004),
      nuevo(2000,2005,2013,2013)]).

% PREDICADOS CARGADOS (FRAGMENTO)
vehiculo(marca('Chevrolet ASTRA FULL
AIRE'),anio#1999,precio#2890000,color(blanco)).
vehiculo(marca('Chevrolet ASTRA FULL AIRE
IMPECABLE'),anio#1999,precio#2890000,color(blanco)).
vehiculo(marca('Chevrolet ASTRA FULL SIN
AIRE'),anio#2002,precio#2470000,color(rojo)).
vehiculo(marca('Chevrolet astra gl
18-mar-13'),anio#2005,precio#850000,color(rojo)).
vehiculo(marca('Chevrolet Astra gl
21-mar-13'),anio#1999,precio#2600000,color(gris)).
vehiculo(marca('Chevrolet Astra GL
26-feb-13'),anio#2010,precio#2000000,color(gris_perla)).
```

Para utilizar el concepto “semi nuevo” hemos definido un conjunto difuso con el término lingüístico “medio_nuevo”. Luego podemos hacer la pregunta “¿Cuáles son los vehículos semi nuevos?” de la forma:

```
QUERY> vehiculo(marca(X),medio_nuevo,Y,color(Z)).
```

Obteniendo los siguientes resultados:

```
Yes with 0.8333333333333334
X = Chevrolet ASTRA FULL AIRE, Y = precio#2890000, Z = blanco.

Yes with 0.8333333333333334
X = Chevrolet ASTRA FULL AIRE IMPECABLE, Y = precio#2890000, Z = blanco.

Yes with 0.3333333333333333
X = Chevrolet ASTRA FULL SIN AIRE, Y = precio#2470000, Z = rojo.

Yes with 0.8333333333333334
X = Chevrolet Astra gl 21-mar-13, Y = precio#2600000, Z = gris.
```

Ambos soportes de consultas, en conjunto permiten responder una amplia cantidad de interrogantes, cuyas respuestas serán de ayuda al momento de tomar una decisión en este contexto, ya sea para compra o venta particular, como también para iniciar algún negocio.

CAPÍTULO VI: Conclusiones

6.1. Análisis de los objetivos propuestos

El objetivo propuesto "Proponer un modelo para la extracción, procesamiento y consulta de información semiestructurada proveniente de la web para apoyar la toma de decisiones en un área de conocimiento específico" se cumple cabalmente (capa 1 y 2). El modelo presentado permite extraer información desde la web mediante la implementación de herramientas y técnicas de harvesting para luego contenerla en un documento XML (Objetivo específico 1) multipropósito. La segunda parte del objetivo principal contenida en el objetivo específico 2 relacionado con los mecanismos para interactuar con la información, también es cubierto con éxito. De esto se encarga la capa 3, 4 y 5 del modelo mediante la construcción de una base de conocimiento formada por una Base de Datos Relacional y una Base de Datos Bousi~Prolog. En virtud de lo anterior podemos decir que los objetivos generales y específicos se cumplieron en su totalidad.

6.2. Principales aportes

No es el propósito de este trabajo proponer nuevos mecanismos, técnicas o metodologías, sino integrar las existentes en un modelo que permita la construcción de sistemas dedicados a la obtención de información a partir de bases de conocimiento construidas a con información extraída desde la web.

Como principal aporte podemos destacar un modelo multicapa y escalable que provee orientación sobre las tecnologías y técnicas existentes, como así también la idoneidad y pertinencia de estas en la integración del modelo con vista a una implementación. De tal manera que el lector pueda seleccionar los componentes más apropiados para integrar el modelo de acuerdo a las necesidades, exigencias y expectativas de su implementación.

Hacer énfasis en que la implementación de este modelo permite hacer uso de la información web de manera casi inmediata con la posibilidad de disponer de la información obtenida de manera práctica.

6.3. Análisis general y Trabajo futuro

Sobre los temas planteados podemos hacer varias reflexiones. Una de las más importantes probablemente se relaciona con la gran cantidad de información existente en la web que se desaprovecha, esto no es una opinión personal sino una realidad. El valor que adquiere la información

es cada vez mayor, pues con ella se puede apoyar los procesos de toma de decisiones. A mayor información menor incertidumbre, lo que se traduce en decisiones tomadas con una alta probabilidad de acierto y esto, claramente se puede cuantificar en un ahorro deducible de la reducción de costos a nivel de gestión. La abundante información existente en la web puede, mediante los procesos apropiados, producir nueva información con valor agregado, la cual hoy en día es perfectamente comercializable. Son muchas las empresas que se dedican exitosamente al procesamiento de datos y a construir información para ser vendida a empresas que, por ejemplo, la emplean en estudios de mercado o análisis de prospectos. Nos referimos a verdadera información construida, no listas de correo ni base de datos de contactos, sino información del tipo "¿Qué marca de celular prefiere la gente?, ¿Cuáles prefieren los hombres o las mujeres?", "¿Qué color de vehículo se vende más o menos, etc.?", lo maravilloso de esto, es que materia prima existe en abundancia en la web y lo mejor, es gratis.

Este trabajo pretende acercar un poco las posibilidades y promesas del webminig, en donde hemos tratado de abstraer al máximo la complejidad subyacente para presentar un modelo atractivo que permita tanto la obtención de información desde la web como la posibilidad de realizar consultas sobre ella con un enfoque particular. Lo particular de este enfoque es que permite incorporar flexibilidad en las consultas mediante la utilización de términos lingüísticos. Esto implica que es necesario manejar y modelar la ambigüedad que reviste cada concepto (alto, bajo, chico, barato, joven, etc), lo cual demanda la existencia de conocimiento experto o experiencia que debe ser plasmada en el modelo mediante conjuntos difusos o grados de proximidad. Por ejemplo, si hablamos de joven, para mí es una persona entre 15 y 30 años, pero para otra puede estar entre 17 y 37. Lo mismo sucede con el precio de un vehículo, ¿Cuánto es caro o barato?. Información como esta proviene necesariamente de la experiencia de las personas, lo cual no necesariamente es un inconveniente, pero es algo que debe tenerse en cuenta para implementar el modelo. Una forma simple de sintetizar esta experiencia o conocimiento es mediante un pequeño estudio estadístico sobre el dominio de información. De modo tal que sea posible conocer muestralmente, en un contexto estadístico, cuanto es caro, barato, joven, etc.

Nuestro planteamiento, si bien es cierto cumple con el objetivo de formar bases de conocimientos con información extraída desde la web para luego consultarla de manera tradicional o agregando flexibilidad en la consulta, no es una solución definitiva para un usuario común, pues reviste algunos tecnicismos y bastante laboriosidad que obliga, ante una implementación escalable, contar con competencias muy específicas sobre todo en materia de modelamiento y construcción de los agentes de software encargados de la extracción, pues estos tienen una fuerte dependencia con la estructura de los

sitios web. Lo mismo sucede con el modelamiento de la imprecisión para dotar de flexibilidad a las consultas.

Si bien es cierto esta complejidad no puede ser eliminada de momento, como trabajo futuro, resulta tremendamente necesario encapsularla o proveer mecanismos de asistencia mediante una capa adicional de abstracción donde operen interfaces de software que asistan en las tareas de planificación, análisis y generación de los robots. Particularmente nos referimos a la interface de creación de la capa 1, pues este grupo de tareas reviste mucha laboriosidad que puede ser asistida con el artefacto de software señalado.

Como parte del trabajo futuro también está la Interface de Intercambio y producción de Información, que viene a ser un mecanismo que en definitiva asiste en la construcción de reglas más complejas que requieren de datos resúmenes u otra información que pueda ser obtenida a partir de los datos precisos. Por último, es deseable una Interface Gráfica de Usuario (GUI) para que la formulación de preguntas a la base de conocimiento resulte simple. Estos tres elementos, en el mismo orden mencionado, concentran a partir este momento nuestra atención como trabajo futuro.

A pesar de las dificultades mencionadas y del trabajo futuro requerido, es una propuesta que posibilita obtener resultados en el muy corto plazo, prácticamente de forma inmediata, en comparación con otras alternativas en el contexto de la web semántica donde se requieren cambios profundos en la web y por ende mayor tiempos para obtener resultados en materia de utilización de información web para apoyar los procesos de toma de decisiones.

Referencias bibliográficas

1. **M. Langer, Arthur.** *Analysis and Design of Information Systems.* New York : Springer, 2008. 978-1-84628-654-4.

2. **Gunasekaran, Angappa y Sandhu, Maqsood.** *Handbook on Business Information Systems.* Toh Tuck Link, Singapore : World Scientific, 2010. 13 978-981-283-605-2.

3. **Clarke, Steve.** *Information system Strategic Management.* London, New York : Routledge, 2001. 0-203-16045-2.

4. *Data Mining Process & Data Preprocessing.* **Dr. MOHD NAWI, NAZRI BIN.** April 2, 2012.

5. *A Framework of Business Intelligence-driven Data Mining for e-Business.* **Hang, Yang y Fong, Simon.** Macau, SAR : s.n., 2009.

6. **Barnes, S.** *Comercio Electrónico y Negocios Virtules. Segunda ed.* s.l. : Elsevier Ltd., 2007.

7. **Mika, Peter.** *Social Networks and the SemanticWeb.* Barcelona, Spain : Springer, 2007. 978-0-387-71000-6.

8. *Mobile Commerce: Assessing New Business Opportunities.* **Hsieh, C.** 2005.

9. *A Mobile Commerce Challenge Model.* **P.D. Bermudez, Manuel E.**

10. *La sobrevaloración de las redes sociales en Internet.* **Katz, Raúl L. y Wu, Haley.** 2008.

11. *Situación de Internet en Latinoamérica.* **Fosk, A.** 2010.

12. *Data mining: torturando a los datos hasta que confiesen.* **L., Molina.** 2002.

13. *El web como Sistema de Información.* **Rodríguez, K y Ronda, R.** 1, Ciudad de La Habana : s.n., Ene - Feb de 2006, ACIMED, Vol. 14.

14. **Taniar, David y Rahayu, Johanna Wenny.** *Web Information Systems.* s.l. : Idea Group Publishing, 2004. 1-59140-208-5.

15. **Kambayashi, Yahiko, y otros.** *Nontraditional Database Systems.* USA and Canada : Taylor & Francis, 2002. 0-203-30194-3.

16. *Redes Sociales en Internet, Redes móviles y Localización.* **V., E.S.M.** 2010.

17. *Aplicación de técnicas de la web semántica.* **Castells, Pablo.** Madrid, España : s.n., 2007.

18. **Pollock, Jeff rey T.** *Semantic Web For Dummies.* Hoboken, EEUU : Wiley, 2009. 978-0-470-39679-7.

19. *A Concise Guide to the Major Internet Bodies.* **Simonelis, Alex.** New York, NY, USA : ACM, 2005.

20. *Organizaciones de normalización en Internet.* **Merlo Vega, José Antonio y Rojo, Ángela Sorli.** 2, 2000, Vol. 23.

21. **Jacobs, Ian y Walsh, Norman.** Architecture of the World Wide Web, Volume One. [En línea] 5 de Diciembre de 2004.

22. **Gralla, Preston y Troller, Michael.** *How the Internet Works, Eighth Edition.* Indianapolis : Que Publishing, 2006. 978-0-7897-3626-0.

23. **W3C.** HTML 4.01 Specification. [En línea] 24 de December de 1999. [Citado el: 30 de 4 de 2013.] http://www.w3.org/TR/1999/REC-html401-19991224/.

24. **Berners-Lee, Tim.** Re: status. Re: X11 BROWSER for WWW. *http://lists.w3.org/Archives/Public/www-talk/1991SepOct/0003.html.* [En línea] 29 de 10 de 1991.

25. **W3C.** Index of Elements. [En línea] W3C, 30 de 4 de 2013. http://www.w3.org/TR/1999/REC-html401-19991224/index/elements.

26. —. XHTML™ Basic 1.1 - Second Edition. [En línea] 23 de November de 2010. [Citado el: 30 de 5 de 2013.] http://www.w3.org/TR/2010/REC-xhtml-basic-20101123/.

27. **Rodríguez, Joaquín Adiego.** *La estructura de los documentos en el ámbito de recuperación de información: propuestas para su compresión, indexación y recuperación.* Valladolid, España : s.n., 2004.

28. **Castro, Elizabeth.** *HTML, XHTML, & CSS, Sixth Edition: Visual QuickStart Guide.* Berkeley, EEUU : Peachpit Press, 2006. 0-321-43084-0.

29. **Funk, Tom.** *WEB 2.0 AND BEYOND: Understanding the New Online Business Models, Trends, and Technologies.* Westport, Connecticut, London : PRAEGER, 2009. 978–0–313–35187–7.

30. **Governor, James, Hinchcliffe, Dion y Nickul, Duane.** *Web 2.0 Architectures.* s.l. : O'Reilly, 2009. 978-0-596-51443-3.

31. *¿Web 2.0, Web 3.0 o Web Semántica?: El impacto en los sistemas de información de la Web.* **Codina, Lluís.** Fabra : s.n., 2009.

32. **W3C.** Extensible Markup Language (XML) 1.0 (Fifth Edition). *W3C.* [En línea] [Citado el: 23 de 05 de 2013.] http://www.w3.org/TR/REC-xml/.

33. *WEB: SEMANTICA: Principios y Estándares.* **Cueva, Samanta Patricia.** 2008.

34. *aAUTOMATOR: HERRAMIENTA FLEXIBLE PARA LA EXTRACCIÓN DE INFORMACIÓN EN SITIOS WEB BIOINFORMÁTICOS.* **Glez-Peña, Daniel, Méndez, José R. y Fdez-Riverola, Florentino.** Ourense, España : Conferencia IADIS Ibero-Americana, 2007. 978–972–8924–45-4.

35. *Revisión: tecnologia de agentes de software.* **Tolosa, Gabriel Hernán y Alfredo Bordignon, Fernando Raul.** 3, Brasília : s.n., 1999, Vol. 28.

36. *Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents.* **Stan Franklin, A.G.,.** s.l. : Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996.

37. *Intelligent Agents: Theory and Practice.* **Michael Wooldridge, N.R.J.** 1995.

38. *Agent-based engineering, the web, and intelligence.* **PETRIE, C.J.** 1996. IEEE Expert. Vol. 280, págs. 98-100.

39. *A meta-model for the analysis and design of organizations in multi-agent systems.* **FERBER, Jacques y GUTKNECHT, Olivier.** Laboratoire d'Informatique, Robotique et Micro-électronique de Montpellier. Université Montpellier II, France : s.n.

40. *Developing multi-agent systems with a FIPA-compliant agent framework.* **Bellifemine, Fabio, Poggi, Agostino y Rimassa, Giovanni.** 2001. SOFTWARE—PRACTICE AND EXPERIENC SP&E.

41. *Mobile Agents: Intelligent Assistants on the Internet.* **Reddy, Parineeth M Reddy M.**

42. *Mobile Agents and the Future of the Internet.* **Kotz, David y Gray, Robert S.** s.l. : ACM, 1999.

43. *Adaptive Multi-agent System: Cooperation and Structures Emergence.* **BOUSSEBOUGH, Imane, MAAMRI, Ramdane y SAHNOUN, Zaïdi.** 2010. JOURNAL OF SOFTWARE. Vol. 5.

44. *A MODEL OF ADAPTATION IN COLLABORATIVE MULTI-AGENT SYSTEMS.* **Lerman, Kristina.**

45. *Problemas Inversos, Tecnicas Evolutivas y Agentes inteligentes: Estrechando las Fronteras.* **MONETT, DAGMAR y BURKHARD, HANS-DIETER.** 2004.

46. *An agent model combining reactive and cognitive capabilities.* **Bussmann, S y Demazeau, Y.** Munich, Alemania : IEEE, 1994.

47. *Intelligent E-Marketing with Web Mining, Personalization and User-adpated Interfaces.* **Fiss, P.P.a.G.** 2001.

48. *Smart Shopper: An Agent-Based Web-MiningApproach to Internet Shopping.* **James Liu, J.Y.** 2003.

49. *A Conversational Agent Based on a Conceptual Interpretation of a Data Driven Semantic Space.* **Francesco Agostaro, A.A., Giovanni Pilato, Giorgio Vassallo and Salvatore Gaglio.** 2005. AI 2005: ADVANCES IN ARTIFICIAL INTELLIGENCE.

50. *Using Agents fot Secure Access to Data in the Internet.* **Zahir Tari, R.M.I.o.T.** 1997.

51. *A Conversational Agent Based on a Conceptual Interpretation of a Data Driven Semantic Space. AI 2005: ADVANCES IN ARTIFICIAL INTELLIGENCE.* **Francesco Agostaro, A.A., Giovanni Pilato, Giorgio Vassallo and Salvatore Gaglio.** 2005.

52. *Tendencias en minería de datos de la Web.* **Baeza-Yates, Ricardo.** 2009. El profesional de la información. Vol. 18.

53. *Formalización de Web Mining como Conocimiento Estructurado.* **Filocamo, Gabriel R. y Chesñevar, Carlos I.** Argentina : s.n.

54. *A web crawler design for data mining.* **Thelwall, Mike, [ed.].** s.l. : Sage Publications, 2001. Journal of Information Science. Vol. 27.

55. *SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers.* **Miller, Robert C. y Bharat, Krishna.** Brisbane, Australia : s.n., April 1998.

56. **Daniel Glez-Peña, José R. Méndez, Florentino Fdez-Riverola.** jARVEST (Java web harvesting library) . [En línea] [Citado el: 21 de 5 de 2013.] http://sing.ei.uvigo.es/jarvest/.

57. *Anatomía de un motor de búsqueda a gran escala de web hipertextual.* **Brin, Sergey y Page, Lawrence.** 1998.

58. *Web Crawler On Client Machine.* **Shettar, Rajashree y Shobha, G.** Hong Kong : s.n., Marzo de 2008, Proceedings of the International MultiConference of Engineers and Computer Scientists.

59. **developers, Scrapy.** *Scrapy Documentation.* 2013.

60. —. Scrapy Official Site. [En línea] http://scrapy.org/.

61. **Schrenk, Michael.** *Webbots, spiders, and screen scrapers: a guide to developing Internet agents with PHP/CURL.* San Francisco : No Starch Press, Inc., 2012. 1-59327-397-5.

62. **Department of Computer Sciences, University of Chile.** Center for Web Research. [En línea] [Citado el: 10 de 6 de 2013.] http://www.cwr.cl/projects/WIRE/.

63. *WIRE: an Open Source Web Information Retrieval Environment.* **Castillo, Carlos y Baeza-Yates, Ricardo.**

64. *Effective Web Crawling. PhD thesis, University of Chile.* **Castillo, C.** 2004.

65. **Nikic, Vladimir y Wajda, Alexander.** Web Harvest. [En línea] [Citado el: 11 de 11 de 2012.] http://web-harvest.sourceforge.net/manual.php.

66. **Berglund, Anders, y otros.** XML Path Language (XPath) 2.0 (Second Edition). *W3C Recommendation 14 December 2010 (Link errors corrected 3 January 2011).* [En línea] 2011. [Citado el: 22 de 5 de 2013.] http://www.w3.org/TR/2010/REC-xpath20-20101214/.

67. **Katz, Howard y Chamberlin, Don.** *XQuery from the Experts: A Guide to the W3C XML Query Language.* s.l. : Addison Wesley, 2003.

68. **Brundage, Michael.** *XQuery: The XML Query Language.* s.l. : Addison Wesley, 2004.

69. **W3C.** XML Path Language (XPath). *XPath language official reference.* [En línea] 1999. [Citado el: 12 de 1 de 2013.] http://www.w3.org/TR/xpath.

70. **Micaelo, Aitor.** *Extracción de información a partir de recursos web semi-estructurados.* 2009.

71. *BOUSI~PROLOG: A Fuzzy Logic Programming Language for Modeling Vague Knowledge and Approximate Reasoning.* **Julián Iranzo, Pascual y Rubio Manzano, Clemente.** 2010. International Conference on Fuzzy Computation (ICFC 2010).

72. *Bousi~Prolog: a Prolog Extension Language for Flexible Query Answering.* **P. Julián, C. Rubio, and J. Gallardo.** 2009, Electronic Notes in Theoretical Computer Science, págs. 131-147.

73. *A similarity-based WAM for Bousi~Prolog.* **Rubio, P. Julián and C.** 2009, In Proceedings of IWANN 2009, Springer LNCS 5517, págs. 245-252.

74. *Fuzzy Sets.Information and Control.* **Zadeh., L. A.** 1965.

75. *A Framework for Computing Linguistic Hedges in Fuzzy Queries.* **Kannan, V. Balamurugan and K.S.** 1, 2010, The Int. J. of Database Management Systems, Vol. 2.

76. *A fuzzy query language for relational databases.* **Takahashi, Y.** Physica-Verlag, Berlin : s.n., 1995, Fuzziness in Database Management Systems.

77. *A conceptual framework for fuzzy query processing - a step toward very intelligent databases systems. Information Processing and Management.* **Tahami, V.** s.l. : Pregamon Press, 1977, Vol. 13.

78. *A fuzzy model for relational databases.* **Petry, B. Buckles and F.** 1985, Fuzzy Sets and Syst., págs. 213-226.

79. *FuzzyClips Version 6.04A. User's Guide.* **Orchard, R.A.** Canadá : s.n., 1998, Integrated Reasoning.Institute for Information Technology.

80. *Proximity relations in the fuzzy relational database model.* **Melton., S. Shenoi and A.** 1999.

81. *A New Approach for Fuzzy Query Processing Based on Automatic Clustering Techniques.* **Hsiao, S.M. Chen and H.R.** 2007, Information and Management Sciences, págs. 223-240.

82. *Fuzzy Query using Clustering techniques Information Processing and Management.* **al, M. Kamel et.** 2, 1990, Vol. 26, págs. 279-293.

83. *The Concept of a Linguistic Variable and its Applications to Approximate Reasoning I, II and III.* **Zadeh, L. A.** 1975, J. of Information Sciences 8 and 9, Elsevier.

84. *Inclusión de Conjuntos Borrosos en el Núcleo del Sistema Bousi~Prolog.* **Julián-Iranzo, Pascual, Rubio-Manzano, Clemente y Gallardo-Casero, Juan.** Huelva : s.n., 2010. XV Congreso Español Sobre Tecnologías y Lógica Fuzzy.

85. *Sistemas Expertos Basados en Reglas.* **Gutiérrez, josé Manuel.**

86. **Bassiliades, Nick, Gvernatori, Guido y Paschke, Adrian.** *Rule-Based Reasoning, Programming, and Applications.* Barcelona, España : Springer, 2011.

Anexos 1 – Ejemplo de código HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
<TITLE>Documento con Tabla</TITLE>
</HEAD>
<BODY>
<TABLE border="2" frame="hsides" rules="groups"
          summary="Code page support in different versions
                   of MS Windows.">
      <CAPTION>CODE-PAGE SUPPORT IN MICROSOFT WINDOWS</CAPTION>
      <COLGROUP align="center">
      <COLGROUP align="left">
      <COLGROUP align="center" span="2">
      <COLGROUP align="center" span="3">
      <THEAD valign="top">
      <TR>
      <TH>Code-Page<BR>ID
      <TH>Name
      <TH>ACP
      <TH>OEMCP
      <TH>Windows<BR>NT 3.1
      <TH>Windows<BR>NT 3.51
      <TH>Windows<BR>95
      <TBODY>
      <TR><TD>1200<TD>Unicode (BMP of ISO/IEC-10646)<TD><TD><TD>X<TD>X<TD>*
      <TR><TD>1250<TD>Windows 3.1 Eastern European<TD>X<TD><TD>X<TD>X<TD>X
      <TR><TD>1251<TD>Windows 3.1 Cyrillic<TD>X<TD><TD>X<TD>X<TD>X
      <TR><TD>1252<TD>Windows 3.1 US (ANSI)<TD>X<TD><TD>X<TD>X<TD>X
      <TR><TD>1253<TD>Windows 3.1 Greek<TD>X<TD><TD>X<TD>X<TD>X
      <TR><TD>1254<TD>Windows 3.1 Turkish<TD>X<TD><TD>X<TD>X<TD>X
      <TR><TD>1255<TD>Hebrew<TD>X<TD><TD><TD><TD>X
      <TR><TD>1256<TD>Arabic<TD>X<TD><TD><TD><TD>X
      <TR><TD>1257<TD>Baltic<TD>X<TD><TD><TD><TD>X
      <TR><TD>1361<TD>Korean (Johab)<TD>X<TD><TD><TD>**<TD>X
      <TBODY>
      <TR><TD>437<TD>MS-DOS United States<TD><TD>X<TD>X<TD>X<TD>X
      <TR><TD>708<TD>Arabic (ASMO 708)<TD><TD>X<TD><TD><TD>X
      <TR><TD>709<TD>Arabic (ASMO 449+, BCON V4)<TD><TD>X<TD><TD><TD>X
      <TR><TD>710<TD>Arabic (Transparent Arabic)<TD><TD>X<TD><TD><TD>X
      <TR><TD>720<TD>Arabic (Transparent ASMO)<TD><TD>X<TD><TD><TD>X
</TABLE>
</BODY>
</HTML>
```

Anexos 2 – Código del Harvester

```
# Este ejemplo extrae un listado de vehículos en venta. Se accede al hipervínculo
# de cada vehículo para extraer más información del mismo.

# Trnsformador padre que entrega una copia de la página a cada hijo

# pag 1
append("http://www2.chileautos.cl/cemagic.asp?pag=1&dea=25&sort=&ciudad=0&tipo=A&co
mbustible=0&maresp=0&modelo=&ai=1920&af=2013&pi=0&pf=1000000000&disp=1&tipo_cambio=
0&aire_acondicionado=0&unico_dueno=0&cierre_centralizado=0&airbag=0&frenos_Abs=0&ki
lom=&c_otros=&fecha_ini=&fecha_fin=&vendedor=0&formulario=Busqueda_Avanzada&region=
0&Carroceria=&cili=0&cilf=99999&bus=&goo=0")

wget{
   branch(:BRANCH_DUPLICATED, :SCATTERED){ # Branch 1
       # Hijo 1
       # Extrae la marca del vehículo de la cual se buscarán más datos
       # visitando su hipervínculo (xpath('//table[4]//*/td[2]/a/@href'))

       pipe {
         xpath('//table[4]//*/td[2]/a/@href') # Muestra hipervínculo del vehículo
         decorate(:head=>"\n\n</vehiculo>\n\n<vehiculo>\n<link-detalle>",
         :tail=>"</link-detalle>")
       }

       pipe {
         xpath('//table[4]//*/td[2]')         # Muestra marca del vehículo
         decorate(:head=>"<marca>", :tail=>"</marca>")
       }
       pipe {
         xpath('//table[4]//*/td[3]')         # Muestra año del vehículo
         decorate(:head=>"<anio>", :tail=>"</anio>")
       }
       pipe {
         xpath('//table[4]//*/td[4]')         # Muestra precio del vehículo
         decorate(:head=>"<precio>", :tail=>"</precio>")
       }
       pipe {
         xpath('//table[4]//*/td[5]')         # Muestra vendedor del vehículo
         decorate(:head=>"<vendedor>", :tail=>"</vendedor>")
       }
```

```
# Hijo 2
# Extrae más antecedentes accediendo a cada hipervínculo
pipe{ #pipe hijo 2
     xpath('//table[4]//*/td[2]/a/@href')     # hipervínculo del vehículo
     decorate(:head=>"http://www2.chileautos.cl/")
   wget
   one_to_one{
     branch(:BRANCH_DUPLICATED, :COLLAPSED){ #Branch 2

       # Extrae la ciudad donde se encuentra el vehículo
       pipe {
         #xpath("//table[2]//*//td[text()='Ciudad:']")
         # Muestra etiqueta ciudad
         xpath("//table[2]//*//td[text()='Ciudad:']/following-sibling::*")
         # Muestra ciudad donde se encuentra el vehículo
         decorate(:head=>"<ciudad>", :tail=>"</ciudad>")
       }

       # Extrae cantidad de visitas del vehículo
       pipe {
         xpath("//div[2]/div[2]/div[3]/font[1]")
         # Muestra cantidad de visitas
         decorate(:head=>"<visitas>", :tail=>"</visitas>")
       }

       # Vende
       pipe {
         xpath("//table[2]//*//td[text()='Vende:']/following-sibling::*")
         decorate(:head=>"<vende>", :tail=>"</vende>")
       }
       # Teléfono
       pipe {
         xpath("//table[2]//*//td[text()='Telefono:']/following-sibling::*")
         decorate(:head=>"<telefono>", :tail=>"</telefono>")
       }
       # Contacto
       pipe {
         xpath("//table[2]//*//td[text()='Contacto:']/following-sibling::*")
         decorate(:head=>"<contacto>", :tail=>"</contacto>")
       }
       # Dirección
       pipe {
         xpath("//table[2]//*//td[text()='Direccion:']/following-sibling::*")
         decorate(:head=>"<direccion>", :tail=>"</direccion>")
       }
       # Comuna
       pipe {
         xpath("//table[2]//*//td[text()='Comuna:']/following-sibling::*")
         decorate(:head=>"<comuna>", :tail=>"</comuna>")
       }
```

```
            # Comentarios
            pipe {
              xpath("//table[2]//*//td[text()='Comentarios:']")
              decorate(:head=>"<comentarios>", :tail=>"</comentarios>")
            }
            # Color
            pipe {
              xpath("//table//td[text()='Color:']/following-sibling::*")
              decorate(:head=>"<color>", :tail=>"</color>")
            }
            # Cilindrada
            pipe {
              xpath("//table/*//td[text()='Cilindrada :']/following-sibling::*")
              decorate(:head=>"<cilindrada>", :tail=>"</cilindrada>")
            }
            # Marca2
            pipe {
              xpath("//table/*//td[text()='Marca:']/following-sibling::*")
              decorate(:head=>"<marca2>", :tail=>"</marca2>")
            }
            # Modelo
            pipe {
              xpath("//table/*//td[text()='Modelo:']/following-sibling::*")
              decorate(:head=>"<modelo>", :tail=>"</modelo>")
            }
            # Extrae la patente del vehículo
            pipe { # pipe patente
              xpath("//table[2]//*//td[text()='Patente:']/following-sibling::*")
              branch(:BRANCH_DUPLICATED, :COLLAPSED){ # branch patente y multas
                 decorate(:head=>"<patente>", :tail=>"</patente>")
                     pipe { # pipe multas
                         decorate(:head=>"http://consultamultas.srcei.cl/ConsultaMu
                         ltas/buscarConsultaMultasExterna.do?ppu=")
                        wget{
                          xpath("//table//tr[2]/td/table//tr[3]/td[2]")
                          replace(:sourceRE=>"\\r|\\n", :dest =>"")
                          decorate(:head=>"<multas>", :tail=>"</multas>")
                        }
                     } # pipe multas
               } # branch patente y multas
            } # pipe patente
          }#Branch 2
        } # one_to_one
      } # pipe hijo 2
   } # Branch 1

}.repeat?{
   xpath("//table//tr//td[2]//div[2]//a[@id='sig']/@href")
   decorate(:head=>"http://www2.chileautos.cl/")
}
```

Anexos 3 – Fragmento de XML producido por harvester

Fragmento XML con 4 registros de aproximadamente 2160 registros obtenidos.

```
<lista-vehiculos>
<vehiculo>
<link-detalle>auto.asp?codauto=3121802</link-detalle>
<marca>Chevrolet astra gl 4-jul-13</marca>
<anio>2000</anio>
<precio>$ 3.100.000</precio>
<vendedor>Vehiculos Particulares</vendedor>
<ciudad>ChillÃ¡n</ciudad>
<visitas>517</visitas>
<vende>daniloavendaÃ±o reyes
</vende>
<telefono>6-1742713</telefono>
<comentarios>Comentarios:Chileautos.cl&#174; el N&#176;1el auto esta en exselemte
estado lo vendo solo por apuro tiene llantas 17 buen audio xenxon y muchas cositas
el presio es conbersable cualquier duda llamar .Chileautos.cl&#174; el portal
chileno N&#176;1 de vehiculos en Internet. Todo este material pertenece a
Chileautos.cl&#174;. Prohibida su reproduccion total o parcial.</comentarios>
<color>azul perlado</color>
<cilindrada>1.600 c.c.</cilindrada>
<marca2>Chevrolet</marca2>
<modelo>astra</modelo>
<patente>TG6585</patente><multas>No registra multas en la base del SRCeI.</multas>
</vehiculo>

<vehiculo>
<link-detalle>auto.asp?codauto=3174159</link-detalle>
<marca>Chevrolet ASTRA GL 28-jul-13</marca>
<anio>2000</anio>
<precio>$ 3.200.000</precio>
<vendedor>Vehiculos Particulares</vendedor>
<ciudad>Concepcion</ciudad>
<visitas>15</visitas>
<vende>jonathan andres sandoval sandoval
</vende>
<telefono>7-1204400</telefono>
<comentarios>Comentarios:Chileautos.cl&#174; el N&#176;1esta en excelente estado en
mecanica detalles pocos
.Chileautos.cl&#174; el portal chileno N&#176;1 de vehiculos en Internet. Todo este
material pertenece a Chileautos.cl&#174;. Prohibida su reproduccion total o
parcial.</comentarios>
<color>BLANCO</color>
<cilindrada>1.800 c.c.</cilindrada>
<marca2>Chevrolet</marca2>
<modelo>ASTRA</modelo>
<patente>UB7214</patente><multas>La patente posee 2 multas.Juzgado PolicÃ-a
LocalRol Causa1 JPL SAN BERNARDO214918 - 20121 JPL SAN BERNARDO220531 - 2012
Registra multas en la base del SRCeI. </multas>
</vehiculo>
```

<vehiculo>
<link-detalle>auto.asp?codauto=3140650</link-detalle>
<marca>Chevrolet ASTRA GL 13-jul-13</marca>
<anio>2001</anio>
<precio>$ 2.250.000</precio>
<vendedor>Vehiculos Particulares</vendedor>
<ciudad>Santiago</ciudad>
<visitas>1.164</visitas>
<vende>ignacio javier CASTRO ESCOBAR
</vende>
<telefono>9-8790562</telefono>
<comentarios>Comentarios:Chileautos.cl® el N°1VEHICULO EN BUEN ESTADO LO VENDO POR RENOVACION, HACE COMO DOS MESES SE LE REMPLAZO LA CORREA DE DISTRIBUCION, A TODA PRUEBA .Chileautos.cl® el portal chileno N°1 de vehiculos en Internet. Todo este material pertenece a Chileautos.cl®. Prohibida su reproduccion total o parcial.</comentarios>
<color>BLANCO</color>
<cilindrada>1.800 c.c.</cilindrada>
<marca2>Chevrolet</marca2>
<modelo>ASTRA</modelo>
<patente>CDZX96</patente><multas>No registra multas en la base del SRCeI.</multas>
</vehiculo>

<vehiculo>
<link-detalle>auto.asp?codauto=3170555</link-detalle>
<marca>Chevrolet astra gl 26-jul-13</marca>
<anio>2001</anio>
<precio>$ 2.500.000</precio>
<vendedor>Vehiculos Particulares</vendedor>
<ciudad>Concepcion</ciudad>
<visitas>176</visitas>
<vende>cristianolivares
</vende>
<telefono>6-21272996-2104034</telefono>
<comentarios>Comentarios:Chileautos.cl® el N°1Radio con pantalla muy buen auto no conversable con todo sus papeles al dia segundo dueÃ±o..Chileautos.cl® el portal chileno N°1 de vehiculos en Internet. Todo este material pertenece a Chileautos.cl®. Prohibida su reproduccion total o parcial.</comentarios>
<color>gris</color>
<cilindrada>1.800 c.c.</cilindrada>
<marca2>Chevrolet</marca2>
<modelo>astra</modelo>
<patente>UD9884</patente><multas>No registra multas en la base del SRCeI.</multas>
</vehiculo>
<lista-vehiculos>

Anexos 4 – Fragmento de predicados producidos por XML2Pred

Fragmento con 4 predicados BPL de un total de 2160 generados aproximadamente. La transformación se realizo considerando las etiquetas lingüísticas *anio*, *precio*, *visitas*, *color*, *cilindrada*, *marca2* y *modelo*.

```
vehiculo(anio#2000,precio#3100000,visitas#517,color('azul
perlado'),marca2('Chevrolet'),modelo('astra')).

vehiculo(anio#2000,precio#3200000,visitas#15,color('BLANCO'),marca2('Chevrolet'),mo
delo('ASTRA')).

vehiculo(anio#2001,precio#2250000,visitas#1164,color('BLANCO'),marca2('Chevrolet'),
modelo('ASTRA')).

vehiculo(anio#2001,precio#2500000,visitas#176,color('gris'),marca2('Chevrolet'),mod
elo('astra')).
```

Anexos 5 – Fragmento de SQL producidos por XML2BD

```
# Se uso como modelo una sola tabla. Y de todos los datos extraídos (Anexo 3) se
# utilizaron para el análisis solo los que se ven en la estructura de la tabla.

CREATE TABLE vehiculo (
idvehiculo VARCHAR(36),
anio NUMERIC,
precio NUMERIC,
ciudad VARCHAR(200),
visitas NUMERIC,
color VARCHAR(200),
cilindrada NUMERIC,
marca2 VARCHAR(200),
modelo VARCHAR(200),
comuna VARCHAR(200),
PRIMARY KEY (idvehiculo)
);

# INSERTS de 2567 registros procesados

INSERT INTO vehiculo
(idvehiculo,anio,precio,ciudad,visitas,color,cilindrada,marca2,modelo) VALUES
('3076c675-f661-4606-81d7-4c3478c3dbfc','2011','7400000','Temuco
','189','Blanco','1800','Chevrolet','ASTRA');
INSERT INTO vehiculo
(idvehiculo,anio,precio,ciudad,visitas,comuna,color,cilindrada,marca2,modelo)
VALUES ('77dd3546-c79d-42b0-ab8b-74b52e43f889','2010','2750000','Santiago
','8068','La Cisterna','GRIS','1600','Chevrolet','1.6 CORSA IIIDIRECCION');
INSERT INTO vehiculo
(idvehiculo,anio,precio,ciudad,visitas,comuna,color,cilindrada,marca2,modelo)
VALUES ('bd1044d3-90ae-4e82-91b2-079ed2aff9c9','2011','3800000','Santiago
','1100','La Cisterna','GRIS','1600','Chevrolet','1.6 CORSA III AIRE FULL LLANTA');
INSERT INTO vehiculo
(idvehiculo,anio,precio,ciudad,visitas,comuna,cilindrada,marca2,modelo) VALUES
('4f24f5f5-d4ff-4cd2-9dcd-a9688095921f','2010','3490000','Santiago
','1992','EstaciÃ³n Central','1600','Chevrolet','1.6 CORSA III NB 28.000 Kms');
```

Anexos 6 – Papers enviados a congresos

En este anexo se adjuntan dos papers parte de esta tesis que fueron enviados a congresos:

1. "*Declarative Fuzzy Linguistic Queries on Relational Databases*" enviado a FQAS 2013 - 10th International Conference on Flexible Querying-Answering Systems. LNAI 8132, pp. 413–424, 2013. Springer-Verlag Berlin Heidelberg 2013.

2. "*Consultas Flexibles sobre bases de conocimiento extraídas desde la web*" enviado al XXV Encuentro Chileno de Computación 2013.