کالج پروژه

www.collegeprozheh.ir

دانلود پروژه های دانشگاهی

بانک موضوعات پایان نامه

دانلود مقالات انگلیسی با ترجمه فارسی

آموزش نگارش پایان نامه ، مقاله ، پروپوزال

دانشگاه آزاد اسلامی

واحد علوم و تحقیقات

پایان نامه کارشناسی ارشد

رشته مهندسی کامپیوتر – نرم افزار

موضوع :

استخراج اتوماتیک اطلاعات بر اساس آنتالوژی

استاد راهنما

دکتر فریدون شمس

استاد مشاور

دکتر مهرنوش شمس فرد

نگارش

مهدی طالبیان کوچکسرایی

سال تحصیلی ۱۳۸۶

سپاسگزاری

در نگارش این پایان نامه از راهنمایی های استاد گرامی دکتر فریدون شمس بهره برده ام که جا دارد در اینجا از ایشان بخاطر کمک های بی دریغشان قدر دانی و تشکر نمایم.

فهرست مطالب

فهرست شکل ها

فهرست لیست ها

فهرست جداول

چکیده

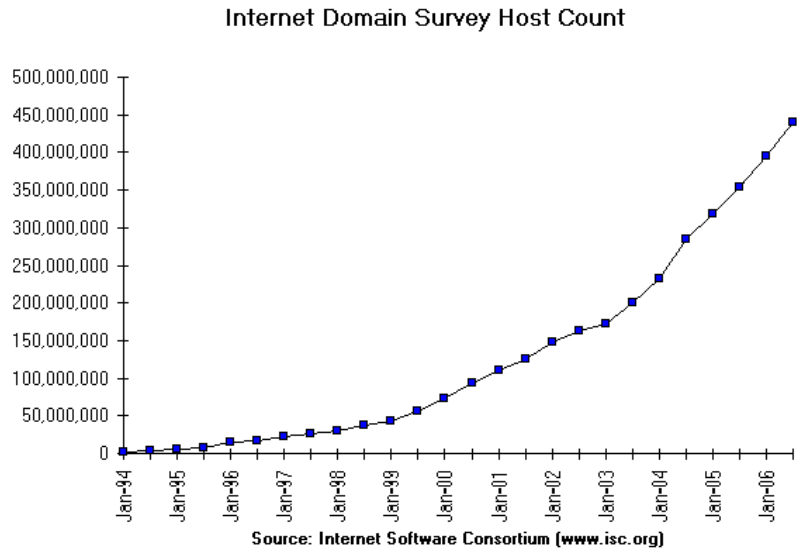
لازمه توسعه و گسترش وب معنایی، ازدیاد هر چه بیشتر آنتالوژی ها می باشد. آنتالوژی ها با ایجاد فهم مشترک از مفاهیم موجود در دامنه دانش، امکان همکاری و تبادل اطلاعات بین ماشین ها را بوجود می آورند. اما ایجاد دستی آنتالوژی ها سخت و طاقت فرسا بوده و احتمال بروز خطای انسانی نیز بسیار زیاد می باشد. از این رو، استخراج اطلاعات از منابع وب بصورت نیمه اتوماتیک و یا تمام اتوماتیک مورد توجه بسیاری از محققان قرار گرفته است. اولین ایده در استخراج اطلاعات، تهیه برنامه ای بنام **Wrapper** است که اطلاعات موجود در صفحات وب را در یک قالب ساختیافته، استخراج می کند. ابزار های متنوعی جهت تولید **Wrapper** ها بصورت دستی، نیمه اتوماتیک و اتوماتیک مورد پژوهش قرار گرفته اند. برخی از این ابزار ها ابتدا قواعد استخراج را آماده کرده و در هنگام استخراج اطلاعات با آگاهی از محل داده ها، به سرعت اطلاعات مورد نیاز را استخراج می نمایند. برخی دیگر نیز با استفاده از آنتالوژی استخراج، داده های موجود در صفحه وب را پردازش کرده و مفهوم مرتبط با هر مقدار متنی را از آنتالوژی دامنه یافته و در نهایت با توجه به الگوهای موجود، اطلاعات را استخراج می نمایند.

در این تحقیق، روش پیشنهادی ما که منجر به تولید یک سیستم نرم افزاری نیز خواهد شد، توصیف **Wrapper** با استفاده از زبان نشان گذاری **WDML** می باشد. این زبان با ترکیب مکانیزم های استخراج مبتنی بر مکان و مبتنی بر آنتالوژی، از مزایای هر دو روش بهره می برد. قواعد استخراج توصیف شده در فایل های **WDML** بصورت درختی تعریف شده و به آن الگوی استخراج گفته می شود از این رو نسبت به سایر روش ها دارای انعطاف پذیری بسیار بالایی می باشد. همچنین در این تحقیق روشی را جهت تولید اتوماتیک فایل های **WDML** با استفاده از یک آنتالوژی استخراج ارائه خواهیم کرد.

کلمات کلیدی: وب معنایی، آنتالوژی، استخراج اتوماتیک اطلاعات، WDML، Wrapper

۱- مقدمه

تا ده سال پیش کسی هرگز فکر آن را نمی کرد که روزی اینترنت بتواند تا این حد در زندگی انسان ها رسوخ نموده و در شیوه زندگی، اخبار، اطلاع رسانی، ارتباطات وغیره تاثیر گذار باشد. امروزه با رشد روز افزون وب و با وجود ۴۰۰ میلیون وب سایت [ISC06] و چندین میلیارد منابع اطلاعاتی (شامل صفحات HTML، فایل های صوتی، ویدئویی، تصاویر، XML، RSS و غیره)، عملا وب جاری به یک کتابخانه بزرگ از منابع و اطلاعات مختلف تبدیل شده است.



شکل ۱ : نمودار رشد سایت های اینترنتی [ISC06]

موتور های جستجوگر مانند Google، Yahoo، AltaVista و غیره، کمک شایانی در یافتن صفحات مورد نظر به کاربران اینترنت نموده اند. این موتور های جستجوگر به کاربران خود امکان آن را می دهند تا تنها با ورود چند کلمه کلیدی منظور خود را از آنچه که بدنبال آن هستند، مشخص نمایند. سپس موتور های جستجو گر این کلمات کلیدی را در محتوای صفحاتی که از قبل در بانک های اطلاعاتی خود ذخیره نموده اند، جستجو کرده و صفحات یافت شده را با استفاده از الگوریتم های موجود امتیاز دهی[1] می نمایند. در نهایت صفحات یافت شده، به ترتیب امتیاز به کاربر نشان داده می شود.

[1] Ranking

یکی از بزرگترین مشکلات در ارتباط با این شیوه سنتی جستجو، که به آن جستجوی کور هم گفته می شود، یافتن کلمات کلیدی مناسب می باشد. هرچه کلمات کلیدی مورد جستجو متناسب با موضوع مورد نظر باشد، پاسخ ها نیز به احتمال قویتر، مرتبط با موضوع خواهد بود. اما متاسفانه یافتن کلمات کلیدی مناسب در اکثر جستجو ها مشکل می باشد. لذا کاربر مجبور است با وارد کردن مجموعه ای از کلمات کلیدی اولیه، ابتدا صفحات یافت شده را خوانده تا کلمات کلیدی جدیدتری را بیابد. کاربر این شیوه را آنقدر ادامه خواهد داد تا به پاسخ مورد نظر خود برسد.

در صورتیکه ماشین ها از مفاهیم[۲] و اطلاعات موجود در صفحات وب آگاهی داشتند، آنگاه می توانستیم، بجای جستجوی کورکورانه‌ی کلمات کلیدی، صفحات وب را بصورت مفهومی جستجو کنیم. اما مشکل اساسی در ارتباط با وب جاری آن است که اطلاعات موجود در آن برای استفاده انسان ها آراسته شده است. مانند تصاویر، صدا، فیلم و حتی صفحات وب. بنابراین برای جستجوی معنایی، نیاز است تا ابتدا معنی و مفهوم اطلاعات موجود، به طریقی برای کامپیوتر ها قابل فهم گردد.

۱-۱- وب معنایی

برای مقابله با مشکلات فوق، آقای تیم برنرز لی[۳]، مبدع وب، برای اولین بار ایده وب معنایی را به عنوان نسل جدید وب جاری مطرح نمود. از نظر ایشان، وب معنایی دارای دو هدف اساسی و مهم می باشد [Lee01]:

۱. قابل فهم کردن اطلاعات موجود در وب جاری برای ماشین ها
۲. افزایش قابلیت همکاری[۴] بین افراد و عامل های درگیر با داده های مشترک

باید به این نکته توجه داشت که وب معنایی هرگز قصد ندارد تا جایگزین وب جاری گردد. بلکه همانند یک لایه بر روی وب جاری قرار می گیرد و کاربران همچنان از طریق اینترفیس ها و مرورگر های جاری، صفحات وب را مشاهده می کنند. در مقابل، وب معنایی تکنولوژی هایی را ارائه می کند که وب سایت ها و عامل های ماشینی می توانند با استفاده از آنها در سطح انسان ها، با کاربران ارتباط برقرار کنند [Dac03]. به عنوان مثال هنگام جستجوی یک موضوع، کاربران می توانند بجای ورود مجموعه ای از کلمات کلیدی، جمله پرسشی خود را به زبان طبیعی مطرح کنند. سپس سیستم با پردازش متن سوال، معنی و مفهوم مورد جستجو را یافته و آن را در بین مفاهیم بازیابی شده از صفحات وب جستجو می کند.

---

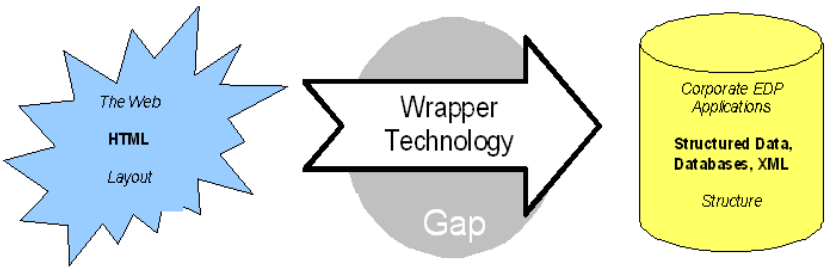
[2] Concepts
[3] Tim Berners Lee
[4] Interoperability

بدین ترتیب کلید اصلی موفقیت وب معنایی دسترسی کاملتر به مفاهیم موجود در صفحات وب می باشد. در وب معنایی جهت توصیف اطلاعات و مفاهیم یک دامنه به صورتی که برای ماشینها قابل فهم باشد، از آنتالوژی ها استفاده می گردد. در واقع آنتالوژی ابزار و روشی جهت رسمی سازی توصیف مفاهیم دامنه می باشد و باعث ایجاد یک فهم مشترک از مفاهیم و اطلاعات موجود می گردد. ایجاد دستی آنتالوژی ها عملی سخت و طاقت فرسا بوده و احتمال خطا های انسانی نیز در آن بسیار زیاد می باشد. از طرفی دیگر با توجه به تنوع اطلاعات در دامنه های گوناگون، جهت ایجاد دستی آنتالوژی ها، نیاز به چندین فرد متخصص در هر یک از دامنه های مورد بحث می باشد. جمع کردن چنین افراد متخصصی در کنار هم و رسیدن به یک توافق در توصیف تمامی مفاهیم، تقریبا امری غیر ممکن است. لذا ایجاد اتوماتیک آنتالوژی ها یکی از چالش ها و مشکلات موجود در وب معنایی می باشد.

۱-۲- بازیابی و استخراج اطلاعات

دسترسی آسان به منابع وب و حجم کثیر اطلاعات و مفاهیم موجود در صفحه وب، آن را به یک گنجینه ارزشمند از آنتالوژی های غیر رسمی تبدیل کرده است. از این رو استخراج آنتالوژی های رسمی سازی شده و یا حتی اطلاعات ساختیافته و یکپارچه از صفحات وب یکی از موضوعات مورد تحقیق پژوهشگران می باشد.

همانطور که گفته شد، صفحات وب برای نمایش و استفاده انسان ها طراحی شده اند و براحتی برای ماشین ها قابل فهم نمی باشند. لذا برای استخراج اطلاعات از صفحات وب لازم است تا ابتدا با استفاده از روش های داده کاوی، آماری و غیره، قواعد استخراج را یافته سپس با اعمال این قواعد بر روی صفحات مشابه، اطلاعات مورد نظر را استخراج نمود. عملیات مربوط به اعمال قواعد استخراج بر روی صفحات وب و بازیابی اطلاعات مورد نظر توسط نرم افزاری بنام Wrapper انجام می گیرد.



شکل ۲: عملکرد Wrapper ها

تاکنون Wrapper ها و چارچوب های مختلفی برای استخراج اطلاعات تهیه شده است که از نظر میزان مکانیزه بودن (عدم نیاز به دخالت انسان) و دقت در بازیابی اطلاعات و نیز انعطاف پذیری نسبت به تغییرات ساختار نمایش در منبع اطلاعاتی اولیه، با هم متفاوت می باشند. امروزه با وجود ابداع روش ها و ابزار های مختلف، هنوز مشکلات زیادی در تولید و اجرای اتوماتیک Wrapper ها وجود دارد. برخی از مشکلات و چالش های پیش رو در استخراج اطلاعات عبارتند از:

۱. ساختار صفحات وب ثابت نبوده و با گذشت زمان دستخوش تغییرات زیادی می گردند. با تغییر این صفحات وب، قواعد استخراج قبلی نامعتبر شده و اجرای Wrapper ها با شکست مواجه خواهد شد. از این رو نگهداری[۵] و اعتبار سنجی[۶] Wrapper ها باید مورد توجه خاص قرار گیرد.
۲. یکی از مشکلات مرتبط با Wrapper ها عدم وجود یک استاندارد مشخص در ساخت آنها است. متاسفانه با وجود Wrapper ها و الگوریتم های استخراج مختلف هنوز روشی جهت توصیف یکپارچه Wrapper ها وجود ندارد و به همین دلیل امکان استفاده همزمان از دو یا چند سیستم استخراج اطلاعات و یا حتی مقایسه عملی Wrapper ها وجود ندارد.
۳. یکی دیگر از مشکلات و چالش های پیش روی استخراج اطلاعات حذف و یا کاهش دخالت انسان و اتوماتیک نمودن فرایند استخراج می باشد. با توجه به حجم عظیم اطلاعات موجود در وب و نیز تغییرات متعدد در صفحات وب، نظارت بر کلیه فرایند استخراج امری غیر ممکن است. از این رو اتوماتیک و یا حداقل نیمه اتوماتیک کردن این فرایند یکی از چالش های اساسی در استخراج اطلاعات می باشد.
۴. سیستم های استخراج اطلاعات با فراهم کردن داده های مورد استفاده در سایر سیستم های کاربردی (از جمله سیستم های وب معنایی)، یکی از ارکان اساسی آنها را تشکیل می دهند. لذا دقت و سرعت دو پارامتر مهم در موفقیت سیستم های استخراج اطلاعات می باشد. متاسفانه این دو پارامتر در مقابل هم قرار دارند. به عبارتی با افزایش دقت، سرعت استخراج اطلاعات کاهش می یابد و برعکس.

ما در این تحقیق، قصد داریم تا در حوزه مشکلات فوق به بررسی پرداخته و در ارتباط با هر یک از آنها راه حل های جدیدی را ارائه نماییم.

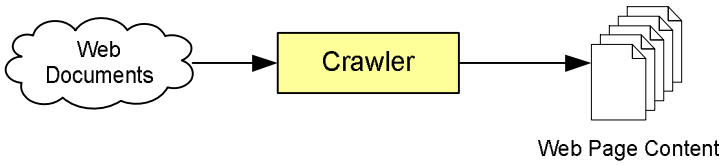
---

[5] Wrapper Maintenance
[6] Wrapper Validation

۱-۳- محدوده بحث و دستاورد تحقیق

روش های مورد استفاده در یک سیستم استخراج اطلاعات، با توجه به نوع منبع ورودی متفاوت می باشد. برای استخراج اطلاعات از منابع متنی غیر ساختیافته معمولا از تکنیک های پردازش زبان طبیعی، برای منابع نیمه و تمام ساختیافته از Wrapper ها به همراه قواعد استخراج، برای منابع تصویری از پردازش تصویر و غیره استفاده می شود. ما در این تحقیق به بررسی منابع ساختیافته و نیمه ساختیافته پرداخته و جهت حل مشکلات مطرح شده در ارتباط با Wrapper ها، راه حل هایی را پیشنهاد خواهیم داد.

دستاورد این تحقیق معماری سیستم جدیدی بنام OntoByOnto خواهد بود که قادر است مسائل مطرح شده فوق را بپوشاند. این سیستم شامل سه مولفه اساسی؛ خزشگر[۷]، استخراج کننده[۸] و تولید کننده Wrapper[۹] می باشد. مولفه خزشگر با دریافت مجموعه ای از آدرس های اولیه، در دوره های زمانی مشخص محتویات صفحات وب را جمع آوری کرده و در یک مخزن اطلاعاتی نگهداری می کند (شکل زیر).

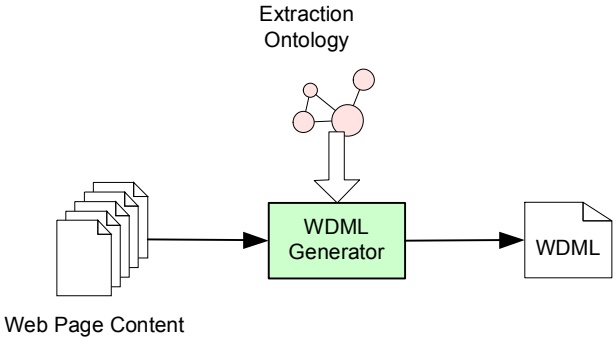


شکل ۳: مولفه خزشگر

ایده اصلی شامل معرفی زبانی بنام WDML [۱۰] جهت توصیف Wrapper ها می باشد. این زبان مبتنی بر XML بوده و به راحتی با استفاده از یک پردازشگر متنی قابل تهیه می باشد. این زبان جهت استخراج اقلام اطلاعاتی از ترکیبی از روش های مبتنی بر مکان، جهت جستجوی سریع محل اقلام اطلاعاتی، و روش مبتنی بر آنتالوژی جهت افزایش دقت در شناسایی اقلام اطلاعاتی استفاده می نماید. همچنین در این زبان، قواعد استخراج بسیار مطابق با ساختار درختی صفحه وب ورودی می باشد و به آن الگوی استخراج گفته می شود. استفاده از WDML در معماری سیستم پیشنهادی ما، بر نحوه طراحی دو مولفه باقیمانده تاثیر زیادی دارد.

---

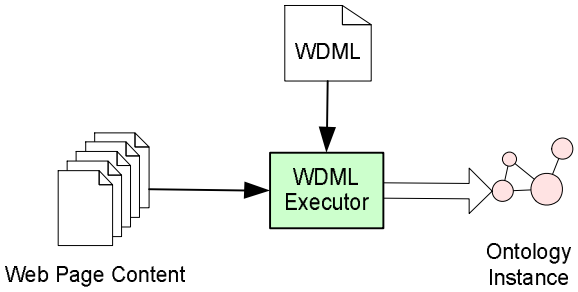
[7] Crawler
[8] Extractor
[9] Wrapper Generator
[10] Wrapper Difinition Markup Language



شکل ۴: مولفه تولید کننده Wrapper

مولفه تولید کننده Wrapper، جهت اتوماتیک کردن فرایند استخراج، با دریافت محتویات صفحات ورودی و با استفاده از آنتالوژی استخراج[11]، الگو های موجود در صفحات وب را شناسایی کرده و فایل های WDML را تولید می نماید. آنتالوژی استخراج یک نمونه از مدل مفهومی دامنه مورد بحث است که مشخصات مقادیری که هر یک از خصوصیات[12] می توانند دارا باشد را توصیف می نماید. این آنتالوژی نقش مهمی در شناسایی اقلام اطلاعاتی دارد به گونه ای که هر چه این آنتالوژی دقیق تر و کامل تر باشد، دقت و صحت الگوهای شناسایی شده نیز بیشتر خواهد بود.

در این تحقیق با توجه به اهمیت آنتالوژی استخراج، روش جدیدی جهت توصیف مقادیر اطلاعاتی را معرفی خواهیم کرد. این روش مبتنی بر ماشین آتاماتای غیر قطعی[13] بوده که با دریافت رشته ای از کلمات ورودی قادر است لیستی از خصوصیات و مفاهیم موجود در آن را استخراج نماید. این روش سبب ایجاد حالت بازگشتی در شناسایی اقلام اطلاعاتی خواهد شد. بدین معنی که هنگام تشخیص یک قلم اطلاعاتی روتین تشخیص قلم اطلاعاتی دیگر بصورت بازگشتی فراخوانی می گردد.



شکل ۵: مولفه استخراج کننده

---

[11] Extraction Ontology
[12] Properties
[13] Non Deterministic Finite State Automata

مولفه استخراج کننده، با توجه به الگو های استخراج تعریف شده در فایل WDML ورودی، اقلام اطلاعاتی موجود صفحه وب متناظر را استخراج می کند. در طی فرایند استخراج، محل قرار گرفتن هر یک از اقلام اطلاعاتی، توسط الگو های استخراج مشخص می گردد. همچنین جهت اطمینان از صحت اقلام اطلاعاتی استخراج شده، در صورت نیاز می توان به آنتالوژی استخراج مربوطه رجوع نمود. در نهایت، اقلام اطلاعاتی استخراج شده می توانند تحت قالب یک آنتالوژی نمونه[14] و یا جداول بانک اطلاعاتی در خروجی نوشته شوند.

۱-۴- طرح کلی بحث

در ادامه این پایان نامه، ابتدا در فصل دوم به بررسی مفاهیم و کلیات مرتبط با موضوع این تحقیق می پردازیم. در این فصل، با توجه به اهمیت آنتالوژی ها و نقشی کلیدی آنها در این تحقیق، ابتدا به معرفی اجمالی آنتالوژی ها می پردازیم. از آنجا که هدف این تحقیق تنها استخراج آنتالوژی های نمونه می باشد بنابراین لازم است تا در این فصل با سطوح مختلف تعریف آنتالوژی نیز آشنا شویم. در ادامه این فصل لازم است تا با برخی مفاهیم پایه ای مانند؛ بازیابی اطلاعات، استخراج اطلاعات، بازیابی داده و تفاوت های میان آنها آشنا شویم.

فصل سوم به بررسی فعالیت های انجام شده در زمینه بازیابی اطلاعات و البته محدود به حوزه این تحقیق، اختصاص دارد. در این فصل ابتدا روش های موجود را دسته بندی کرده و سپس بعد از تشریح جزئیات هر یک از روش ها، یک نمونه از ابزار های پیاده سازی شده در آن روش را تشریح خواهیم کرد. در پایان این فصل به جمع بندی و بررسی نقاط ضعف و قدرت هر یک از روش های بحث شده می پردازیم.

در فصل چهارم، معماری سیستم پیشنهادی خود را معرفی خواهیم کرد. این معماری بر اساس یک زبان پیشنهادی جدید جهت توصیف Wrapper ها می باشد. در ادامه این فصل به معرفی جزئیات زبان WDML پرداخته و سپس الگوریتم استخراج را تشریح خواهیم کرد. در پایان این فصل یک مثال نمونه از نحوه اجرای الگوریتم را ارائه خواهیم نمود.

---

[14] Instance Ontology

فصل پنجم، به بررسی موضوع استخراج اتوماتیک الگو های موجود در صفحات وب اختصاص دارد. از آنجا که در این بخش از معماری پیشنهادی، آنتالوژی استخراج نقشی کلیدی ایفا می کنند در ابتدای فصل، برخی اجزای پیشنهادی که باید یک آنتالوژی استخراج نمونه داشته باشد را معرفی می کنیم. مهمترین جزء پیشنهادی ما، الگو های NFA می باشد. در این فصل ضمن تشریح کامل این الگو، چند مثال نیز برای فهم بیشتر آن مطرح خواهیم نمود. در ادامه این فصل، یک الگوریتم استخراج ابتدایی را جهت تشخیص الگو های موجود در صفحات وب، ارائه کرده و با نشان دادن گام های اجرای الگوریتم بر روی یک صفحه وب نمونه، آخرین بخش از سیستم پیشنهادی را به پایان می رسانیم.

در نهایت، فصل ششم به جمع بندی و ارائه طرح هایی جهت ادامه و پیشرفت آتی این تحقیق اختصاص خواهد داشت.

۲- مفاهیم و کلیات

قبل از شروع بحث خود در مورد استخراج اطلاعات از صفحات وب، ابتدا به معرفی مفاهیم مهم مورد استفاده در این پایان نامه می پردازیم تا پیش زمینه ای مناسب از دامنه مورد تحقیق داشته باشیم. در این فصل، ابتدا به معرفی آنتالوژی پرداخته و برخی تکنولوژی های بکار رفته در آن را به اختصار شرح می دهیم. با توجه به آنکه آنتالوژی ها اطلاعات مورد نیاز در وب معنایی را سازماندهی می کنند، در بخش بعدی به بررسی ارتباط میان سیستم های استخراج اطلاعات و سیستم های مبتنی بر وب معنایی می پردازیم. در نهایت با توجه به برخی مفاهیم کلیدی و ظاهرا مشابه (نظیر استخراج اطلاعات و بازیابی اطلاعات) لازم است تا با تفاوت های آنها نیز آشنا خواهیم شد.

۲-۱- معرفی آنتالوژی

ریشه اصلی کلمه آنتالوژی برگرفته از علم فلسفه می باشد. آنتالوژی ترکیب دو کلمه یونانی "Onto" به معنی "هستی" و کلمه "Logy" به معنای "مطالعه" می باشد و در فارسی به آن هستان شناسی نیز اطلاق می شود. هستان شناسی (یا همان Ontology) شاخه‌ای از فلسفه است که به مطالعهٔ بودن، هستی و یا وجود[۱۵] می‌پردازد. دانش هستی‌شناسی در پی تشخیص و شرح رده های بنیادین و ارتباطات در هستی یا وجود برمی‌آید تا بدین وسیله بتوان به تعریف موجودات و انواع آنها درآن چارچوب پرداخت.

اما در حوزه فن آوری اطلاعات (IT)، معنای آنتالوژی تا حدی متفاوت است. بطور خلاصه می توان گفت؛ آنتالوژی عبارت است از توصیف رسمی واژگانی که در یک دامنه خاص با آنها برخورد می کنیم. تعریف دقیق تر آنتالوژی در حوزه IT بصورت زیر است:

۱. یک آنتالوژی کلمات و مفاهیم (مورد استفاده جهت توصیف و نمایش یک دامنه[۱۶]) را تعریف می کند.

---

[15] Existence

[16] در اینجا منظور از دامنه، محتوای یا مضمون یک ناحیه ای از دانش می باشد. مانند پزشکی، تعمیرات اتومبیل، طرح ریزی مالی، مدیریت اقتصادی و غیره.

۲. آنتالوژی یک محصول مهندسی شامل یک لغتنامه مشخص جهت توصیف بخشی از واقعیات، به همراه مجموعه ای از فرضیه های واضح و شفاف (با توجه به معنای مرتبط در یک لغتنامه) می باشد. به عبارتی دیگر، تعیین و توصیف مفاهیم ذهنی[۱۷].

تعریف اول دارای دو بخش است: ۱) آنتالوژی یک حوزه از دانش (دامنه) را توصیف کرده و نمایش می دهد. ۲) کلمات و مفاهیم توصیفات را تعریف می کند.

منظور از توصیف یک حوزه ای از دانش، هنر تشریح و بیان نوشتاری یا گفتاری بخش های مهم آن حوزه از دانش می باشد. برای مثال هنگام توصیف حوزه تعمیرات اتومبیل، احتمالا درباره موارد زیر صحبت می کنیم. ۱) انواع ماشین های موجود (سدان، بنز، استیشن واگن و غیره)، ۲) انواع موتورها (گاز سوز، بنزینی، گازوئیلی دیزلی، برقی و مختلط) ۳) مشخصات موتور (V-6 Ford و غیره) ۴) سازنده (ولو، هیوندای و غیره) ۵)اجزای تشکیل دهنده ماشین (موتور ها، سیستم های الکتریکی، سیستم های سرما ساز، شاسی و بدنه و غیره)

هنگام توصیف یک حوزه از دانش (دامنه) اشیاء مهم آن دامنه، خصوصیات آنها و روابط میان آنها را توصیف می کنیم. همچنین ممکن است قواعدی را درباره آن دامنه مطرح کنیم. به عنوان مثال؛ قاعده یافتن نقص های فنی ۱) اگر ماشین استارت نمی زند، باطری و اتصالات آن را چک کنید و ۲) ...

بنابراین می توان گفت که حتی یک توصیف ساده نیز می تواند یک آنتالوژی باشد. بطور کلی، یک آنتالوژی شامل بخش های زیر است: ۱) کلاس ها ۲) نمونه ها ۳) روابط میان کلاس ها و اشیاء ۴) خصوصیات اشیاء ۵) عملیات و فرایندهای در ارتباط با اشیاء ۶) شرایط و قواعد مرتبط با اشیاء

علاوه بر توصیف یک حوزه از دانش، نیاز است تا آن توصیف را نمایش داد. بدین معنی که یک توصیف را بگونه ای کد کنیم تا دیگران نیز بتوانند از آن استفاده نمایند. یک توصیف، شامل کلمات و عباراتی به زبان طبیعی، مانند؛ انگلیسی، چینی یا فارسی می باشد. لذا در صورت نیاز، ابتدا کلمات و اصطلاحات خود را تعریف می نماییم. سپس آنها را جهت توصیف حوزه دانش خود با هم ترکیب می کنیم.

در حوزه IT، مفهوم نمایش کمی پیچیده تر می باشد. در واقع، در حوزه IT، باید مدلی را خلق کنیم که نرم افزار قادر به استفاده از آن باشد. بدین منظور، یک حوزه از دانش را با کلاس ها، نمونه ها،

---

[17] Specification of Conceptualization

خصوصیات، روابط و قواعد آن نمایش می دهیم. ما از اصطلاحات توصیفی زبان طبیعی، به عنوان برچسب یا نام مفاهیم اساسی (کلاس ها، خصوصیات، روابط) استفاده می کنیم.

معمولا جهت نمایش آنتالوژی ها بجای زبان طبیعی از زبان منطقی بازنمایی دانش، استفاده می کنیم. چرا که می خواهیم توصیفات خود را تا حد امکان، دقیق و شفاف و غیر مبهم ارائه کنیم در حالیکه زبان های طبیعی خیلی مبهم می باشند. همچنین می خواهیم که مفاهیم و توصیفات مورد نظر جهت استفاده در تکنولوژی اطلاعات آماده باشد.

۲-۱-۱- سطوح بازنمایی آنتالوژی

وقتیکه از آنتالوژی ها صحبت می کنیم، لازم است تا با تفاوت های میان سطوح نمایش دانش آشنا باشیم. این شناخت لازم است، چرا که آنتالوژی ها می توانند در قالب زبان ها و یا لغتنامه نحوی به همراه سمانتیک، نشان داده شوند. از طرفی از آنجا که آنتالوژی ها محتوی هستند و محتوی تنها می تواند توسط زبان محتوی (که معمولا به آن زبان بازنمایی دانش گفته می شود) نمایش داده شوند. بنابراین حداقل دو سطح برای بازنمایی وجود دارد. یک سطح زبان بازنمایی دانش که با توجه به سطوح پایینی آن (سطح شی) به آن سطح متا (Meta) نیز گفته می شود. سطح دوم، سطح شیء (Object Level) است که در آن آنتالوژی ها توصیف می گردند. واضح است که یک سطح سوم که همان سطح نمونه[۱۸] ها است نیاز داریم. در این سطح سوم، نمونه هایی از کلاس های آنتالوژی وجود دارند. از طرفی اگر در سطح نمونه ها تمرکز کنیم، می توان آن را سطح شیء در نظر گرفت و سطح متا سطح بالایی آن یعنی سطح دوم خواهد بود که در آن، آنتالوژی ها تشریح می گردند. همچنین اگر در سطح دوم تمرکز کنیم می توان آن را بشکل سطح اشیاء دید و سطح بالایی آن یعنی سطح یک که در آن زبان بازنمایی دانش تعریف می گردد، سطح متا خواهد بود. پس می توان گفت که سطوح متا و شیء نسبی می باشند. لیست زیر سطوح نمایش آنتالوژی ها را به همراه یک مثال نشان می دهد[Dac03].

- سطح اول: سطح بازنمایی دانش است. این سطح، سازه های مورد استفاده در سطح مفاهیم آنتالوژی را تعریف می کند. این سازه ها، شامل مفاهیم؛ کلاس، رابطه، خصوصیات و غیره است. مثال هایی از این سطح عبارتند از زبان های CYCL، ONE، Ontolingua، KIF، UML، OWL و RDF/S و DAML+OIL

[18] Instance

LEVEL	EXAMPLE CONSTRUCTS
Knowledge representation (KR) language (Ontology Language) level: Meta level to the ontology concept level	Class, Relation, Instance, Function, Attribute, Property, Constraint, Axiom, Rule
Ontology concept (OC) level: Object level to the KR language level, meta level to the instance level	Person, Location, Event, Parent, Hammer, River, FinancialTransaction, BuyingAHouse, Automobile, TravelPlanning, etc.
Ontology instance (OI) level: Object level to the ontology concept level	Harry X. Landsford III, Ralph Waldo Emerson, Person560234, PurchaseOrder TransactionEvent6117090, 1995-96 V-6 Ford Taurus 244/4.0 Aerostar Automatic with Block Casting # 95TM-AB and Head Casting 95TM

جدول ۱: سطوح بازنمایی آنتالوژی [Dac03]

- سطح دوم: سطح مفهوم آنتالوژی (Ontology Concept) است. در این سطح مفاهیم آنتالوژی ها با استفاده از سازه های سطح یک ایجاد می شوند. در این سطح باید مفاهیم کلی جهان را مدل سازی کنیم. دانش دامنه درباره Persons، Locations، Events و Parents و Hammers و تراکنش ها مالی و غیره
- سطح سوم: سطح نمونه های آنتالوژی (Ontology Instances). در این سطح، با توجه به سازه های تعریف شده در سطح فوق، نمونه های موجود در دامنه مورد بحث تعریف می گردد. به عبارتی می توان گفت، این سطح شامل پایگاه دانش یا پایگاه حقایق از ادعاها و گزاره هایی درباره نمونه ها می باشد. مانند؛ علی و رضا به عنوان نمونه هایی از کلاس Person.

۲-۱-۲- زبان های تعریف آنتالوژی

همانطور که گفته شد، هر توصیفی در ارتباط با اطلاعات یا مفهوم و معنی آنها را آنتالوژی گوییم. هر چه این توصیف غنی تر باشد، آنتالوژی مورد نظر نیز قوی تر خواهد بود. از طرفی دیگر لازم است تا آنتالوژی ها را بگونه ای کد کنیم تا برای ماشین ها هم قابل فهم باشند. بدین ترتیب هر چقدر آنتالوژی ها قوی تر باشند، نیاز به دخالت انسان ها نیز کمتر می شود. به عنوان مثال یک سیستم آژانس مسافرتی می تواند بصورت اتوماتیک در میان صفحات وب کاوش کرده و یک برنامه سفر با کمترین هزینه ممکن را ارائه نماید.

گفتیم که هر توصیفی می تواند یک آنتالوژی باشد، بنابراین زمانیکه یک سیستم را با استفاده از ابزار های Rational Rose مدل می کنیم، در واقع یک آنتالوژی را ایجاد می کنیم. اصولا دانش را می توان با استفاده از زبان های نماد گرافیکی (مانند شبکه های معنایی[19]، نقشه های موضوعی[20]، گراف های مفهومی[21]، UML و غیره) و یا با استفاده از زبان های منطق (مانند منطق توصیفی[22]، منطق مرتبه اول[23] و دوم و غیره) نمایش داد.

یکی از مهمترین تلاش ها در زمینه بازنمابی دانش خوانا، مربوط می شود به تاسیس چارچوب ارتباطی مشترک بین عامل های مختلف. هدف از تاسیس چارچوب ارتباطی مشترک، آن است تا عامل ها بتوانند با استفاده از یک زبان مشترک با یکدیگر گفتگو کنند. ذیلا به معرفی برخی استاندارد ها و زبان های تعریف آنتالوژی می پردازیم:

- **قالب مبادله دانش (KIF):** این یک زبان بازنمایی دانش می باشد که قبل از ظهور وب معنایی بوجود آمده است. KIF مبتنی بر منطق گزاره ای مرتبه اول (به همراه برخی خصوصیات منطق مرتبه دوم) می باشد. KIF از نظر نحوی، بسیار شبیه به زبان LISP می باشد.
- **Ontolingua:** یک زبان بازنمایی دانش، مبتنی بر KIF می باشد. این زبان جهت توصیف آنتالوژی ها طراحی شده است. Ontolingua به عنوان اولین زبان تعریف آنتالوژی شناخته شده است و در حال حاظر در دانشگاه استنفورد سیستمی با عنوان Ontolingua طراحی شده است که یک محیط همکاری توزیع شده را جهت ساخت و مدیریت و استفاده از آنتالوژی ها را بوجود آورد ه است. (http://ksl.stanford.edu/software/ontolingua)
- **اتصال باز پایگاه دانش[24] (OKBC):** زبان دسترسی به دانش و تبادل دانش مبتنی بر پروتکل Frame می باشد.این زبان در اوایل دهه ۹۰ توسط مهندسان بازنمایی دانش تحت حمایت سازمان DARPA ایجاد گشت. با استفاده از این زبان می توان بین انواع مختلف پایگاه های دانش، بدون اطلاع از جزئیات پیاده سازی آنها، ارتباط برقرار کرد.
- **Protégé:** یک ابزار مدیریت آنتالوژی می باشد که در آزمایشگاه انفورماتیک پزشکی دانشگاه استنفورد طراحی شده است. Protégé مبتنی بر مدل دانش OKBC بوده و یک ابزار بسیار مفید جهت طراحی و مدیریت آنتالوژی ها می باشد. http://protege.stanford.edu

---

[19] Semantic Network
[20] Topic Maps
[21] Conceptual Graph
[22] Description Logics
[23] First Order Logic
[24] Open Knowledge Base Connectivity

- **زبان تعریف آنتالوژی در وب (OWL)** : یک زبان استاندارد معرفی شده توسط کنسرسیوم جهانی وب (W3C) جهت تعریف و آنتالوژی ها می باشد. این زبان تا کنون به عنوان بهترین و کاملترین زبان جهت توصیف آنتالوژی ها شناخته شده است.

۲-۲- استخراج اطلاعات و وب معنایی

موضوع استخراج اطلاعات یکی از بحث های مرتبط با داده وب کاوی[۲۵] می باشد. وب کاوی ریشه در داده کاوی[۲۶] [Wtn00] داشته و عبارت است از استفاده از روش های داده کاوی جهت کشف و استخراج اطلاعات از منابع وب بصورت اتوماتیک [Ksl00]. در حالت کلی می توان فرایند وب کاوی را در مراحل زیر خلاصه کرد:

- **اکتشاف منابع**[۲۷]: یافتن مستندات و سرویس های موجود در وب. این فرایند معمولا با استفاده از یک خزشگر انجام می گیرد.
- **استخراج اطلاعات:** عبارت است از استخراج اطلاعات از منابع صفحات جدید یافت شده در فاز قبل.
- **عمومیت دادن**[۲۸]: عبارت است از یافتن الگو های مشترک بکار رفته در میان صفحات یک وب سایت و نیز در میان چند وب سایت.

موضوع وب کاوی دارای زیر شاخه های متعددی می باشد که از این میان کاوش محتویات وب[۲۹] مرتبط با تحقیق ما می باشد و عبارت است از کشف منابع اطلاعاتی در وب و استخراج اطلاعات از آنها. در اینجا این سوال پیش می آید که اساسا استخراج اطلاعات چه ارتباطی با وب معنایی دارد؟

باید توجه داشت که موفقیت وب معنایی به شدت وابسته به رشد و گسترش آنتالوژی ها در انواع مختلف دامنه ها می باشد و لذا آنتالوژی ها نقشی اساسی را در مبادله اطلاعات و توسعه وب لغوی به سمت وب معنایی ایفا می کنند. در واقع آنتالوژی ها یک فهم مشترک از یک دامنه را برای تسهیل

---

[25] Web Mining
[26] Data Mining
[27] Resource Discovery
[28] Generalization
[29] Web Content Mining

ارتباطات بین افراد و سیستم های نا همگون و توزیع شده در وب را فراهم می کنند. استفاده از تکنیک های استخراج اطلاعات جهت تسریع و کاهش خطا های انسانی در ساخت انواع آنتالوژی ها می تواند بسیار موثر باشد.

همانطور که در بخش قبل نیز گفته شد، آنتالوژی ها در سه سطح (شامل سطح نمایش دانش، سطح مفهوم آنتالوژی و سطح نمونه آنتالوژی) تعریف می گردند. بنابراین باید مشخص گردد که بازیابی اطلاعات در هر یک از این سطوح، تا چه حد می تواند کمک نماید.

سطح اول، سطح نمایش دانش می باشد. در این سطح اجزای زبان بازنمایی آنتالوژی نشان داده می شود. زمانیکه یک زبان مشخص را برای نمایش آنتالوژی های دامنه انتخاب می کنیم، در واقع مفاهیم تعریف شده در این سطح از آنتالوژی ها، شامل مفاهیمی مانند؛ کلاس، خصوصیت، زیر کلاس، نوع داده و کاردینالیتی[30] و غیره را بصورت پیش فرض می پذیریم. بنابراین نیازی به استفاده از روش های بازیابی اطلاعات در این سطح از آنتالوژی ها نداریم.

سطح دوم، سطح مفهوم آنتالوژی می باشد. در این سطح، با استفاده از سازه های سطح یک (یعنی کلاس ها، خصوصیات، نوع داده و غیره)، به توصیف خصوصیات هر یک از مفاهیم موجود در دامنه مورد نظر خود می پردازیم. به عنوان مثال می توانیم جهت توصیف مفهوم موبایل از المان Class استفاده کنیم و با استفاده از المان Super Class مشخص کنیم که موبایل زیر مجموعه مفهوم Electronic-Products می باشد. همچنین می توانیم با استفاده از المان Property خصوصیات مربوط به گوشی موبایل (مانند نام، ابعاد، وزن، امکان اتصال به کامپیوتر، امکان گرفتن عکس و غیره) را تعریف نماییم. آیا می توان انتظار داشت سیستم های بازیابی اطلاعات، بتوانند مفاهیم جدید را در میان مستندات وب جستجو کرده و خصوصیات آنها و نیز روابط بین آنها و سایر مفاهیم دیگر را بیابند؟ بررسی انواع مختلف مقالات نشان داده است که هنوز استفاده از روش های بازیابی اطلاعات در این سطح از آنتالوژی ابتدایی و ناچیز می باشد و باید تحقیقات و پیشرفت های بیشتری در این زمینه انجام گیرد.

سطح سوم، شامل تعریف نمونه هایی[31] از مفاهیم تعریف شده در سطح قبل می باشد. به عنوان نمونه در مثال آنتالوژی گوشی تلفن همراه، نمونه هایی مانند Nokia 1110، SonyErricson W800 و غیره را می توان ذکر نمود. به ازای هر یک از نمونه های فوق باید خصوصیات و روابط موجود را نیز مشخص نمود. تعریف دستی نمونه های آنتالوژی بسیار وقت گیر و کاری طاقت فرسا می باشد. لذا روش های بازیابی اطلاعات در این سطح می تواند در یافتن نمونه های مفاهیم موجود در آنتالوژی کمک شایانی نماید.

---

[30] Cardinality
[31] Instances

۲-۳- استخراج اطلاعات از انوع مختلف منابع متنی وب

با توجه به گسترش روز افزون وب و تبدیل آن به یک کتابخانه بزرگ از صفحات و اطلاعات مختلف، وب به یک گنجینه ارزشمند تبدیل شده است. لذا امروزه تلاش های زیادی جهت استفاده هر چه بیشتر از این گنجینه صورت می گیرد. با توجه به این که هر نوع تفسیری از مفاهیم موجود، یک آنتالوژی بوده و از آنجا که وب مملو از تفاسیر، توضیحات و انواع بازنمایی ها می باشد، می توان گفت که وب شامل انبوهی از آنتالوژی های غیر رسمی می باشد. البته اکثر این آنتالوژی ها به زبان طبیعی توصیف شده اند و جهت استفاده در ماشین ها باید با توجه به یک زبان تعریف آنتالوژی مشخص، آن توصیفات را رسمی سازی نمود. در وب انواع مختلفی از منابع متنی وجود دارد که می توان آنها را به سه دسته زیر تقسیم بندی نمود:

- **ساختیافته:** مانند مستندات XML یا جداول بانک های اطلاعاتی رابطه ای، که در آن اقلام مختلف داده با استفاده از برچسب ها یا قواعدی معین و ثابت، مشخص شده اند.
- **نیمه ساختیافته**[۳۲] : مانند مستندات HTML که در آن اقلام داده، جهت نمایش به انسان ها قالب بندی شده اند. معمولا این نوع ساختار ها، هم از نظر محتوایی و هم از نظر ساختاری، مرتبا در حال تغییر می باشند. صفحات نمایش لیست محصولات، کاتالوگ و مشخصات فنی محصول، لیست قیمت ها و غیره نمونه هایی از این نوع مستندات می باشند.
- **غیر ساختیافته** : در این نوع ساختارها اقلام اطلاعاتی در قالب زبان طبیعی و به شکل متن ارائه شده اند و یافتن اقلام اطلاعاتی بدون دخالت انسان بسیار مشکل می باشد. اطلاعات موجود در متن خبر، یک مقاله، آگهی و غیره از جمله مثال هایی هستند که بصورت غیر ساختیافته ارائه می شوند.

استخراج اطلاعات از مستندات غیر ساختیافته معمولا پیچیده و غیر قطعی بوده و نیازمند به روش های پردازش زبان طبیعی می باشد. لذا Wrapper ها نمی توانند در استخراج اطلاعات از این صفحات کمک شایانی نمایند. در مقابل صفحات نیمه ساختیافته و ساختیافته، بدلیل نظم نسبی موجود در آنها، موضوع مناسبی جهت اجرای Wrapper ها می باشند. جهت تولید آنتالوژی با استفاده از Wrapper ها باید اقلام اطلاعاتی استخراج شده را بطریقی با بخش های موجود در آنتالوژی، منطبق نمود.

---

[32] Semi Structured

۲-۴- بازیابی اطلاعات[۳۳] و استخراج اطلاعات[۳۴]

ایده استفاده از کامپیوتر ها جهت جستجوی اقلام اطلاعاتی مرتبط، در مقاله ای با عنوان We May Think توسط آقای Vannevar Bush در سال ۱۹۴۵ ارائه شد[۳۵]. بعد از آن اولین سیستم های بازیابی اطلاعات در دهه های ۵۰ و ۶۰ میلادی پیاده سازی شد. تا سال ۱۹۹۰ چندین روش مختلف جهت بازیابی اطلاعات ابداع شد که معمولا بر روی یک محدوده متنی کوچک، بسیار خوب عمل می نمودند.

در سال ۱۹۹۲، سازمان دفاع آمریکا با همکاری انجمن بین المللی استاندارد ها و تکنولوژی[۳۶] (NIST)، مجموعه کنفرانس های بازیابی متن[۳۷] (TREC) را پایه نهادند. این کنفرانس در تسریع جستجو و تحقیقات در زمینه بازیابی اطلاعات نقش بسیار مهمی را ایفا کرده است. با ظهور موتورهای جستجو گر در اواخر دهه ۹۰ و توسعه و پیشرفت سریع آنها، امروزه نیاز به یک سیستم بازیابی اطلاعات، بیش از پیش احساس می شود.

فرایند بازیابی اطلاعات از زمانیکه کاربر متن پرس و جوی خود را در سیستم وارد می کند، شروع می گردد. البته یک سیستم بازیابی / استخراج اطلاعات با یک موتور جستجوی ساده مانند Google بسیار متفاوت می باشد. استخراج اطلاعات (IE)، یک فرایند مبتنی بر تحلیل زبان طبیعی به منظور استخراج اقلام اطلاعاتی[۳۸] می باشد. این فرایند متن ها را به عنوان ورودی دریافت کرده و داده های غیر مبهم را با یک فرمت ثابت، به عنوان خروجی تولید می کند. این داده ها ممکن است مستقیما جهت نمایش به کاربران نشان داده شده و یا جهت ذخیره در یک بانک اطلاعاتی یا صفحه گسترده[۳۹]، به منظور تجزیه و تحلیل های آتی، مورد استفاده گیرند. حتی ممکن است از این داده ها به منظور شاخص گذاری در برنامه های بازیابی اطلاعات (IR)، مانند موتور های جستجوگر وب؛ به عنوان مثال Google، استفاده شود.

علیرغم تشابه ظاهری عبارت های بازیابی اطلاعات و استخراج اطلاعات، این دو با هم یک تفاوت مهم نیز دارند:

---

[33] Information Retrieval
[34] Information Extraction
[35] منبع http://www.wikipedia.com
[36] National Institute of Standards and Technology
[37] Text Retrieval Conference
[38] Snippets of Information
[39] Spread Sheet

- یک سیستم بازیابی اطلاعات، اطلاعات مورد نیاز کاربر را بر نمی گرداند. بلکه مجموعه ای از مستندات را بازیابی می کند که احتمالا اطلاعات مورد نیاز کاربر را شامل می شوند.
- یک سیستم استخراج اطلاعات، متن ها را تجزیه و تحلیل کرده و تنها بخش هایی از آن را که مورد نظر کاربر می باشد، به آن نشان می دهد.

به عنوان مثال، یک کاربر سیستم IR را در نظر بگیرید که جهت یافتن اطلاعاتی در ارتباط با تشکیلات گروه های تجاری در بازار های تجهیزات کشاورزی، تعدادی کلمه مرتبط را در یک موتور جستجو وارد کرده و مجموعه ای از صفحات که احتمالا مرتبط با موضوع مورد نظر می باشد را دریافت می کند. کاربر صفحات دریافتی را خوانده و اطلاعات مورد نظر خود را از آنها استخراج می کند. سپس کاربر این اطلاعات را در یک صفحه گسترده (به عنوان مثال در برنامه Excel) ثبت کرده و شاید یک چارت نیز از آنها درست نموده و در نهایت یک گزارش تهیه کند. در مقابل یک سیستم IE، بطور اتوماتیک اطلاعات مورد نظر، شامل نام شرکت ها و گروه های کاری مرتبط را از صفحات وب دریافت کرده و آن ها را بصورت اتوماتیک در یک صفحه گسترده، ارائه می کند.

البته سیستم های IE در قیاس با سیستم های IR دارای مزایا و معایبی نیز می باشند. ساخت سیستم های IE بسیار سخت بوده و وابستگی آنها به دامنه دانش مورد بحث بسیار زیاد می باشد. همچنین این سیستم ها در مقایسه با سیستم های IR، عملیات محاسباتی بسیار زیادی را انجام می دهند. اما در مقابل، هنگامیکه حجم متون مستندات مرتبط بسیار زیاد باشد، سیستم های IE دارای کارایی بالقوه بیشتری می باشند. چرا که این سیستم ها نسبت به IR حجم کمتری از اطلاعات را جهت مطالعه در اختیار کاربران خود قرار می دهند. همچنین در مواردی که نیاز باشد تا اطلاعات در زبان های مختلف و یا با یک فرمت ثابت ارائه گردد، سیستم های IE بخاطر طبیعت غیر مبهم خود این کار را بهتر و دقیق تر از سیستم های IR انجام می دهند.

اولین قدم در استخراج اطلاعات، واکشی محتویات منابع وب می باشد. برای این منظور یک خزشگر[40] وب، با داشتن مجموعه ای از آدرس صفحات اولیه در وب، محتویات هر صفحه وب را واکشی کرده و در آن به جستجوی لینک به سایر صفحات می پردازد. خزشگر آدرس صفحات جدید را به لیست صفحات مورد جستجوی خود اضافه کرده و عملیات واکشی را برای سایر صفحات ادامه می دهد.

قدم بعدی در استخراج اطلاعات، استفاده از الگوریتم های شناسایی الگو ها و اقلام اطلاعاتی می باشد. ما برخی از این الگوریتم ها را با توجه به دامنه تحقیق خود، در فصل های بعدی معرفی خواهیم

---

[40] Crawler

کرد. در ادامه لازم است جهت آشنایی بیشتر با موضوع تحقیق برخی تفاوت را بین اصطلاحات مشابه و پر کاربرد در این زمینه بپردازیم.

۲-۴-۱- "بازیابی داده" در مقابل "بازیابی اطلاعات"

در محدوده این بحث، بازیابی داده[41] اساسا شامل تعیین این موضوع است که کدامیک از اسناد شامل کلمات کلیدی موجود در متن پرس و جوی کاربر می باشند. این نوع بازیابی در اکثر مواقع نیازهای اطلاعاتی کاربران را مرتفع نمی سازد.

در سیستم های بازیابی داده اگر یکی از هزاران خروجی سیستم دارای خطا باشد، نقص بزرگی به حساب می آید. اما در سیستم های بازیابی اطلاعات، خروجی ها غالبا غیر دقیق بوده و از خطا های کوچک آن چشم پوشی می شود. علت این تفاوت آن است که سیستم های بازیابی اطلاعات اغلب با متون زبان طبیعی سروکار دارند که معمولا دارای ساختار منظم و پیش بینی شده ای نمی باشند و مهمتر از همه اینکه زبان طبیعی از نظر مفهومی و معنایی مبهم و پیچیده می باشد.

بازیابی داده هرگز قصد ندارد که مشکل بازیابی اطلاعات در مورد یک موضوع مشخص را حل کند. اما در مقابل یک سیستم بازیابی اطلاعات، سعی می کند تا حد امکان محتوای منابع اطلاعاتی (مستندات وب) را تفسیر کرده و آنها را بر اساس میزان ارتباط شان با نیاز های اطلاعاتی کاربر رتبه بندی[42] نمایند. در اینجا منظور از تفسیر محتوای یک سند، آن است که اطلاعات نحوی[43] و معنایی[44] موجود در متن مستندات بازیابی شده و با متن پرس و جوی کاربر انطباق داده شود. بنابراین مشکل تنها استخراج اطلاعات نمی باشد. بلکه باید میزان ارتباط آنها را با نیاز های اطلاعاتی کاربر نیز سنجید.

از این رو مفهوم میزان ارتباط[45] در مرکز توجه سیستم های بازیابی اطلاعات قرار دارد. در واقع می توان گفت که هدف بازیابی اطلاعات، بازیابی تمام مستنداتی است که به نحوی با موضوع مورد نظر کاربر ارتباط داشته باشند [Baz99].

---

41 Data Retrieval
42 Ranking
43 Syntactic
44 Semantic
45 Relevance

۲-۴-۲- "بازیابی اطلاعات برای وب معنایی" در مقابل "بازیابی اطلاعات مبتنی بر آنتالوژی"

بازیابی اطلاعات مبتنی بر آنتالوژی، الزاما با بازیابی اطلاعات برای وب معنایی یکسان نمی باشد، به عبارتی هر سیستمی که از آنتالوژی ها استفاده کند الزاما یک برنامه وب معنایی نمی باشد. بسیاری از برنامه های کاربردی، برای افزایش کارایی، در بازیابی اطلاعات از آنتالوژی ها استفاده می کنند. از طرفی دیگر برخی سیستم های نوشته شده برای وب معنایی از روش هایی استفاده می کنند که در آن آنتالوژی ها هیچ کاربردی ندارد. به عنوان مثال پردازش آماری سه تایی های RDF.

بنابراین زمانیکه ادعا می شود یک سیستم بازیابی اطلاعات برای وب معنایی نوشته شده است، باید از خروجی آن مستقیما در تکنولوژی های وب معنایی استفاده گردد. به عنوان مثال یک سیستم استخراج اطلاعات که متن پرس و جو را از کاربر گرفته و داده های موجود در یک صفحه وب را استخراج می کند، نمی تواند یک سیستم برای وب معنایی باشد.

۲-۵- ایجاد آنتالوژی برای وب معنایی

ایجاد، مدیریت و بطور کلی چرخه حیات آنتالوژی ها مربوط به حوزه بحث مهندسی آنتالوژی می شود. در اینجا ما قصد نداریم در مفاهیم فوق کنکاش کنیم ولی با توجه به نقش کلیدی آنتالوژی ها در وب معنایی و توانایی روش های استخراج اطلاعات در جمع آوری آنتالوژی های نمونه، در ادامه به بررسی انواع روش های تولید آنتالوژی خواهیم پرداخت.

۲-۵-۱- ایجاد دستی آنتالوژی ها

اولین روش جهت ساخت آنتالوژی ها، روش دستی در توصیف مفاهیم موجود در دامنه می باشد. البته در این روش ابزارهایی وجود دارد که برخی از فعالیت های مربوطه مانند مدیریت آنتالوژی، استنتاج، تست اعتبار سنجی آنتالوژی و غیره را بر عهده می گیرد. لذا لازم نیست تمام فعالیت ها بصورت دستی انجام شود. بطور کلی، این روش شامل مراحل زیر می باشد:

۱. گروهی از کارشناسان خبره دامنه، ساختار مفهومی آنتالوژی را (احتمالا با استفاده از ابزارهـای موجود) طراحی می کنند.
۲. گروهی از متخصصان علوم کامپیوتر، مستندات طراحی در مرحله قبل را تحویل گرفتـه و آن را بطریقی که برای کامپیوتر ها قابل فهم باشد، مدل می کنند.
۳. گروهی از خبره های مهندسی کامپیوتر، سیستم ها و ابزار هایی را جهت بازنمایی، مدیریت و نگهداری آنتالوژی ها طراحی می کنند.

البته در مراحل فوق، یک فرد می تواند عضو بیش از یـک گـروه نیـز باشـد و نیـازی بـه جـدا بـودن متخصصان هر گروه نیست. این روش ساخت آنتالوژی ها، هنوز در چارچوب های مهندسی آنتالوژی نظیر Protégé و WebODE و OntoEdit و غیره استفاده می گردد. خلاصه ای از این چارچوب ها و متدولوژی های مرتبط در [Arp03] و [Crs05] آمده است. در این جا قصد نداریم، موارد فوق را به تفـصیل بررسـی کنیم. ولی لازم است جهت آشنایی با مشکلات موجود یک مرور کلی از آن داشته باشیم.

چارچوب های ذکر شده اغلب بر تولید یک محیط جامع و فراگیر که بتواند فازهای تولید آنتالوژی را به یک روش مجتمع و یکپارچه کنترل نماید، متمرکز می باشند. نتیجه تلاش های فوق ابـزار یـا ابزارهـایی است که شامل موارد زیر می باشند:

- طراحی ادراکی دامنه
- رسمی سازی (فرموله کردن) ادراک[۴۶] فوق الذکر
- پیاده سازی ساختار مفهومی فرموله شده
- ایجاد مفاهیم
- پردازش موازی (و احتمالا اصلاح) آنتالوژی توسط افراد خبره در خلال عملیات فوق
- اعتبار سنجی آنتالوژی
- ادغام آنتالوژی ها
- مدیریت آنتالوژی ها (مانند بروز رسانی، نسخه گذاری[۴۷] و غیره)

سیستم هایی که از روش دستی در تولید آنتالوژی ها استفاده می کنند، ممکن است از تمام موضوعات فوق بطور کامل استفاده نمایند. اما حتی با وجود تکنیک ها، الگوریتم ها و ابزار های پیشرفته و حرفـه ای، هنوز مهمترین فعالیت فوق، یعنی ساخت آنتالوژی ها، باید توسط انسان انجام گیرد.

---

[46] Formalisation of the conceptualisation
[47] Versioning

مشکل ترین بخش در تولید آنتالوژی، یافتن مفاهیم، خصوصیات و روابط بین آنها می باشد. حتی با وجود ماهرترین کارشناسان دامنه، ممکن است دو گروه جداگانه، طراحی های متفاوتی را ارائه نمایند. از طرفی دیگر، ساخت دستی آنتالوژی ها عملی خسته کننده و وقت گیر می باشد. همچنین بخاطر نقش زیاد انسان در تعریف آنتالوژی، احتمال خطای انسانی زیاد شده و نتایج آن معمولا بسیار وابسته به فرد طراح می باشد. بعلاوه به سختی می توان گروهی از افراد خبره را برای طراحی یک دامنه مشخص گرد هم آورد. این مشکلات سبب گرایش به سمت استخراج اتوماتیک آنتالوژی ها از منابع موجود در وب شده است.

۲-۵-۲- ایجاد اتوماتیک آنتالوژی ها

ایجاد اتوماتیک آنتالوژی از صفحات وب هرگز دقت و اطمینان لازم که در روش ایجاد دستی آنتالوژی ها وجود دارد را نداشته و به همین دلیل استخراج آنتالوی ها (در سطح مفهوم آنتالوژی دامنه) از صفحات غیر ساختیافته موضوع این تحقیق نمی باشد. در مقابل تولید نمونه های موجود از یک یا چند مفهوم آنتالوژی، بدلیل کثرت و تعدد آنها عملی خسته کننده بوده و لازم است تا از Wrapper های موجود برای این منظور استفاده نماییم.

بدین منظور لازم است تا Wrapper ها، قالب یکسانی را جهت تعریف قواعد استخراج و نگاشت اقلام اطلاعاتی استخراج شده با بخش هایی از آنتالوژی دامنه، رعایت نمایند. ما این استاندارد را در غالب زبان تعریف Wrapper[۴۸] یا به اختصار WDML، پیشنهاد می کنیم. WDML یک زبان نشانه گذاری جهت تعریف الگو ها، قواعد استخراج و نگاشت بین اقلام اطلاعاتی و آنتالوژی مقصد می باشد. الگو ها و قواعد استخراج موجود در یک فایل WDML توسط ماژولی بنام Extractor تجزیه تحلیل شده و آن را بر روی یک صفحه وب اعمال می کند تا اقلام داده استخراج گردند. ماژول فوق با توجه به دستورات نگاشت در WDML، اقلام داده استخراج شده را به آنتالوژی نمونه نگاشت می کند.

در واقع ماژول Extractor با استفاده از فایل ورودی WDML، جایگزین Wrapper ها می گردد و قادر است علاوه بر استخراج اطلاعات از صفحات وب، داده های استخراج شده را به یک آنتالوژی نمونه تبدیل نماید. اما مسئله مهم نحوه ساخت فایل های WDML می باشد. چرا که ایجاد دستی این فایل ها

---

[48] Wrapper Difinition Markup Language

بدلیل نیاز به آگاهی از ساختار صفحات وب پر هزینه می باشد. از این رو در فصل ۵، جهت استخراج الگوی های WDML از صفحات وب، روشی را در این خصوص ارائه خواهیم کرد.

در فصل بعد به معرفی معماری سیستم پیشنهادی خود می پردازیم. این سیستم از زبان WDML جهت تعریف Wrapper ها استفاده نموده و قادر است تا با استفاده از آنتالوژی استخراج الگو های موجود در صفحات وب را بصورت اتوماتیک شناسایی کرده و با استفاده از این الگو ها، فایل های WDML را تولید نماید. یکی دیگر از مزایای سیستم پیشنهادی گروه بندی اتوماتیک صفحات وب می باشد. بدین معنی که صفحاتی که از نظر ساختاری و مفهومی دارای یک الگو می باشند (به عنوان مثال، صفحات معرفی قابلیت های گوشی های تلفن همراه)، شناسایی کرده و الگویی جهت یافتن سایر صفحات مشابه ایجاد می نماید.

۲-۶- خلاصه مطالب و نتیجه گیری

آنتالوژی ها، روشی جهت توصیف اطلاعات و مفاهیم موجود در یک دامنه می باشند. جهت توصیف آنتالوژی ها می توان از هر روشی حتی یک زبان محاوره ای طبیعی نیز استفاده نمود. اما جهت رسمی سازی این توصیفات به گونه ای که برای ماشین ها نیز قابل فهم باشد، باید آن ها را با استفاده از یک زبان منطق (مانند First Order Logic، Conceptual Graph، Description Logic، OWL و غیره)، توصیف نمود. آنتالوژی ها بدلیل ایجاد یک فهم مشترک از مفاهیم موجود در دامنه، جهت سازماندهی داده ها و اطلاعات مورد استفاده در وب معنایی استفاده می گردند، لذا موفقیت و پیشرفت وب معنایی به شدت وابسته به تولید هر چه بیشتر آنتالوژی ها می باشد.

در این فصل دیدیم که آنتالوژی ها را می توان با توجه به سطوح بازنمایی دانش، به سه دسته تقسیم کرد که از میان آنها آخرین سطح، یعنی سطح آنتالوژی نمونه، بسیار به موضوع تحقیق ما نزدیک می باشد. همانطور که گفته شد، وب مملو از اطلاعات مختلف در دامنه های مختلف می باشد. به عنوان مثال در دامنه ای مانند گوشی های تلفن همراه، به جرات می توان گفت که وب بزرگترین مرجع توصیف کننده برای انواع مختلف گوشی های تلفن همراه در زمینه هایی مانند اطلاعات تکنیکی، نظرات کاربران، مزایا و معایب گوشی و غیره می باشد. در واقع این اطلاعات، آنتالوژی های نمونه ای می باشند که بصورت غیر رسمی در صفحات وب توصیف شده اند. استفاده از روش های استخراج اطلاعات، جهت جمع آوری اطلاعات مرتبط با مفاهیم یک آنتالوژی و ذخیره سازی آن در قالب یک آنتالوژی نمونه می تواند کمک شایانی به توسعه و گسترش وب معنایی نماید.

البته باید به این موضوع نیز توجه داشت که بازیابی اطلاعات و استخراج اطلاعات دو مقوله متفاوت می باشند. در بازیابی اطلاعات، مجموعه ای از مستندات و صفحات که احتمالا با پرس و جوی کاربر مرتبط می باشند یافت می گردد ولی در فرایند استخراج بخش هایی از صفحه وب که با مفهوم یا موضوع مورد نظر مرتبط باشد، استخراج می گردد. از این رو موضوع استخراج اطلاعات بیشتر با اهداف این تحقیق مرتبط می باشد.

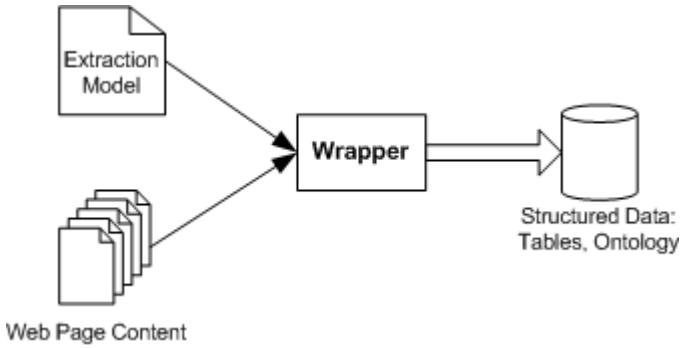
بنابراین بطور کلی می توان گفت هدف اصلی این تحقیق استخراج اقلام اطلاعاتی موجود در صفحات وب و ذخیره سازی آن تحت یک آنتالوژی نمونه می باشد. البته این هدف بسیار کلی بوده و در فصل آتی، بعد از مطالعه سوابق کاری مرتبط با این تحقیق، زیر موضوعات دقیق تری را جهت بررسی و طرح ایده های خود ارائه خواهیم نمود.

۳- بررسی سوابق مرتبط با تحقیق

ما در این فصل ابتدا سوابق و فعالیت های مرتبط با این تحقیق، را دسته بندی کرده و با برخی از ابزار های آنها آشنا می شویم. در پایان این فصل به جمع بندی مطالب پرداخته و نقاط ضعف و قوت هر یک را در یک جدول نشان می دهیم تا بتوانیم مقایسه جامع تری نسبت به آنها داشته باشیم.

۳-۱- مقدمه

معمول ترین روش در استخراج اطلاعات از منابع ساختیافته و یا نیمه ساختیافته، استفاده از Wrapper ها می باشد. Wrapper برنامه ای است که یک صفحه وب را به عنوان ورودی دریافت کرده و با استفاده از یک مدل استخراج مشخص، داده های موجود در صفحه وب را استخراج می کند.



شکل ۶: Wrapper

تا کنون فعالیت های مختلفی در زمینه ایجاد Wrapper ها، صورت گرفته است که بطور کلی می توان آنها را به پنج دسته زیر تفکیک نمود [Sno02 و Bar05 و Lae02]:

- روش های مبتنی بر زبان پرس و جو
- روش های آگاه از ساختار HTML
- روش های مبتنی بر استنتاج
- روش های مبتنی بر پردازش زبان طبیعی
- روش های مبتنی بر آنتالوژی

در ادامه این فصل به تشریح هر یک از روش های فوق پرداخته و در ارتباط با هر یک از آنها ابزار پیاده سازی شده مرتبط با آن را معرفی می کنیم. در انتها نیز به جمع بندی کلی این روش ها پرداخته و نقاط قوت و ضعف آن ها را برجسته می نماییم.

۳-۲- روش های مبتنی بر زبان های پرس و جو

یکی از اولین گام ها، جهت تولید Wrapper ، ایجاد زبان هایی خاص، جهت واکشی اقلام اطلاعاتی از صفحات وب می باشد. این زبان ها نسبت به زبان های همه منظوره مانند جاوا و C# و VB و ... خیلی ساده تر بوده و به عنوان یک ابزار در تولید Wrapper ها به تولید کنندگان آن، کمک می نمایند. در واقع می توان گفت این روش، کاملا بصورت دستی انجام می گیرد، چرا که کاربر باید با آگاهی از ساختار صفحات وب و زبان های مذکور تنها در جهت کمک به واکشی اطلاعات از صفحات HTML، استفاده می گردند.

در این روش قواعد استخراج بصورت دستورات پرس و جو تهیه می شوند. بدین منظور کاربر باید با توجه به ساختار درختی تگ های HTML و نیز محل قرار گرفتن اقلام اطلاعاتی، دستوراتی همانند دستور واکشیش Select در زبان SQL را تهیه نماید. در هنگام واکشی اطلاعات، این دستورات به واحد پردازشگر زبان ارسال شده تا با توجه به صفحه وب ورودی اقلام اطلاعاتی موجود در آن واکشی گردد.

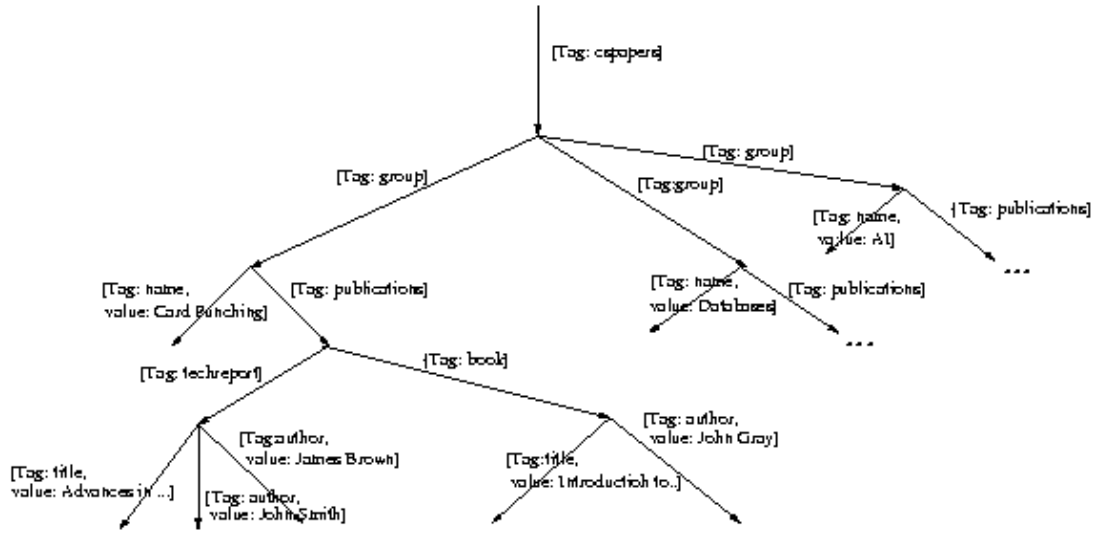
یکی از معایب این روش نیاز به دخالت زیاد انسان در یافتن اقلام اطلاعاتی موجود (به ازای هر صفحه وب) می باشد. عیب دیگر این روش آن است که Wrapper های ایجاد شده بسیار وابسته به ساختار صفحات می باشد و در صورت تغییر ساختاری و محتوایی صفحات وب، اجرای این Wrapper ها با خطا همراه خواهد شد. از جمله این زبان ها می توان FLORID [Fro97] و WebOQL [Aro98] را نام برد. در اینجا، جهت آشنایی بیشتر، ذیلا به معرفی اجمالی WEBOQL می پردازیم.

WEBOQL –

WebOQL [Aro98] یک زبان پرس و جوی بیانی، جهت توصیف Wrapper ها می باشد. این زبان، دارای دو خصوصیت مهم زیر می باشد:

۱. دسترسی به ساختار داخلی اشیای موجود در صفحات وب
۲. امکان ایجاد ساختار های پیچیده جدید به عنوان خروجی پرس و جو

از آنجا که اکثر داده های موجود در وب نیمه ساختیافته می باشند، این زبان نیز بر استخراج داده های نیمه ساخت یافته تاکید دارد. ساختار داده ای اصلی در این زبان Hyper Tree نام دارد. Hyper Tree، یک درخت برچسب دار و مرتب شده می باشد. در این درخت دو نوع کمان؛ داخلی و خارجی، وجود دارد. کمان های داخلی جهت نمایش اشیاء ساخت یافته و کمان خارجی جهت نمایش Hyperlink ها استفاده می شود. شکل زیر یک نمونه از درخت Hyper Tree که می تواند توسط یک Wrapperتولید شود را نشان می دهد.



شکل ۷: یک درخت Hyper Tree نمونه در WebOQL [Aro98]

این درخت می تواند در نتیجه تجزیه و تفسیر یک فایل، مانند فایل pubs.xml زیر، ایجاد گردد:

```
<?xml version="1.0"?>
<cspapers>
 <group>
   <name>Card Punching</name>
   <publications>
      <techreport>
         <title>Advances in Card Punching</title>
         <author>John Smith</author>
         <author>James Brown</author>
      </techreport>
      <book>
         <title>Introduction to Card Punching</title>
```

```
            <author>John Gray</author>
          </book>
      </publications>
    </group>
  <group>
      <name>Databases</name>
      <publications>
         ...
      </publications>
   </group>
   ...
  </cspapers>
```

مجموعه ای از HyperTree های مرتبط در یک ساختار بنام web جمع آوری می گردد. هر دوی HyperTree و Web، می توانند توسط WebOQL مدیریت شده و به عنوان خروجی ایجاد گردند.

WebOQL یک زبان تابعی می باشد، اما پرس و جوها در قالب آشنای Select-From-Where نگهداری می گردد. برای مثال فرض کنید که فایل pubs.xml در شکل فوق، شامل اطلاعات نشریه ها باشد. پرس و جوی زیر تمام نشریه هایی که توسط Smith نوشته شده است را نشان می دهد:

```
[Tag:"result" /
   select y
   from x in browse("file:pubs.xml") via ^*[tag = "publications"],
        y in x',
        z in y'
   where z.tag = "author" and z.value ~ "Smith"
]
```

پرس و جوی فوق یک HyperTree را در HyperTree دیگر نگاشت می کند. به عبارتی، در واقع، پرس و جوی فوق، تابعی است که یک وب را به وب دیگر نگاشت می کند. به عنوان مثال، پرس و جوی زیر یک صفحه وب جدید برای هر گروه تحقیقاتی که شامل نشریه های آن گروه می باشد، ایجاد می کند.

```
select x' as x'.value
 from x in browse("file:pubs.xml")'
```

۳-۳- روش های مبتنی بر پردازش زبان طبیعی

این روش، معمولا جهت استفاده در متون بدون ساختار (متن های آزاد نوشته شده به زبان طبیعی) استفاده می شود. پردازش زبان طبیعی توسط برخی ابزار ها مانند OntoWrapper [Dav03 و Eng02] و KIM [Pop03] و غیره، مورد استفاده قرار می گیرد. در این روش، نرم افزار قواعد استخراج را با استفاده از یک آنتالوژی دامنه فرا گرفته و سپس اقدام به استخراج اطلاعات می نماید. در این روش جهت یافتن روابط بین جملات، باید عملیاتی مانند فیلتر گذاری[۴۹] و نشانه گذاری بخش هایی از صحبت(POS) [۵۰] و نشانه گذاری سمانتیک لغوی[۵۱] اجرا گردند تا به این ترتیب قواعد استخراج کشف گردند. این نوع قواعد مبتنی بر محدودیت های سمانتیکی و نحوی می باشند که در یافتن اطلاعات مرتبط در یک سند کمک می کنند.

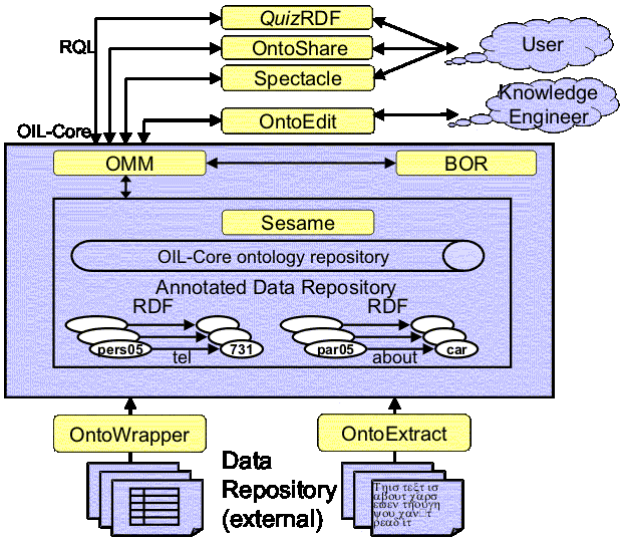
در حال حاضر ابزار ها و ایده های مطرح شده در زمینه پردازش زبان طبیعی قدرت و دقت مورد نیاز جهت تشخیص محتوای جملات و متن ها را ندارند [Sno02]. از این روش تنها در مواردی محدود استفاده می شود (به عنوان مثال در متن هایی که حالت تلگرافی دارند) مانند : لیست کارها و آگهی های اجاره آپارتمان ها و اطلاعیه های برگزاری سمینار ها و غیره. ذیلا به معرفی برخی ابزارها و پلتفورم های مرتبط در این روش می پردازیم.

OntoKnowledge –

پروژه On-to-Knowledge، مجموعه ای از ابزار ها را جهت پردازش اطلاعات معنایی، طراحی و پیاده سازی کرده است [Onto07]. بخشی از این ابزار ها ماژول های OntoWrapper [Eng02-7] برای استخراج اطلاعات از داده ها نیمه ساخت یافته و OntoExtract [Eng02-6] برای استخراج اطلاعات از داده های غیر ساخت یافته می باشد. شکل زیر معماری پیشنهادی جهت مدیریت دانش در وب معنایی را نشان می دهد [Dav03].

---

[49] Filtering
[50] Part of Speech Tagging
[51] Lexical Semantic Taging



شکل ۸: معماری OntoKnowledge [Dav03]

این معماری تمام مراحل چرخه مدیریت دانش را که ذیلا به بررسی اجمالی آن می پردازیم، نشان می دهد [Dav03]:

- **اکتساب دانش** : بدلیل وجود مقدار بسیار زیادی از داده ها در منابع وب (بصورت نیمه ساخت یافته و یا غیر ساخت یافته) ، استخراج این داده ها بصورت اتوماتیک یا حداقل نیمه اتوماتیک الزامی می باشد. این نیازمندی در بخش پایینی معماری فوق، توسط ابزار های OntoWrapper [Eng02-7] برای استخراج اطلاعات از منابع نیمه ساخت یافته و OntoExtract [Eng02-6] برای استخراج اطلاعات از منابع غیر ساخت یافته، پشتیبانی شده است.
- **نمایش دانش** : بعد از استخراج اتوماتیک اطلاعات از منابع موجود در وب، می بایست آن را توسط یک زبان تعریف آنتالوژی نمایش داد تا بتوان جهت استدلال و یا (احتمالا) اعمال تغییرات در آینده، مورد استفاده قرار داد. همانطور که در شکل نیز نشان داده شده، در این پروژه جهت نمایش دانش از معماری SESAME و زبان RDF استفاده می گردد.
- **نگهداری دانش:** برای نگهداری، مدیریت و کار با آنتالوژی ها نیاز به یک میان افزار[52] می باشد. بدین منظور، در این پروژه از ماژول میان افزار آنتالوژی[53] (OMM) استفاده می شود.
- **استفاده از دانش:** در نهایت نیاز به ابزاری جهت دسترسی به اطلاعات می باشد تا به کاربران امکان استفاده از دانش ذخیره شده در سیستم را ارائه کرد. بدین منظور ابزارهایی مانند

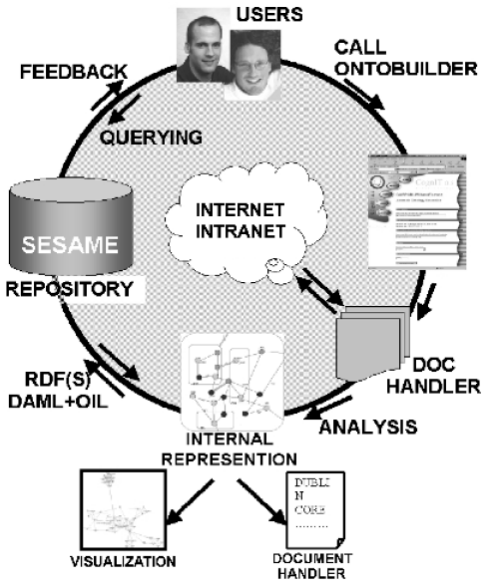
---

[52] MiddleWare
[53] Ontology Middleware Module

QuizRDF و OntoShare و Spectacle ایجاد شده اند. البته OntoEdit نیز برای استفاده توسط مهندس و خبره دانش، جهت تعریف آنتالوژی ها و ایجاد دانش اولیه، پیاده سازی شده است.

همانطور که گفته شد OntoKnowledge دارای دو ماژول مهم جهت استخراج اطلاعات می باشد.

OntoExtract: OntoExtract مبتنی بر موتور تحلیل زبان طبیعی ساخته شده است. این موتور متون زبان طبیعی را تحلیل کرده و آنتالوژی های سبک وزن دامنه را تولید می کند و در این بین از دانش موجود در مخزن مرکزی نیز استفاده می کند. همانطور که گفته شد، پلتفورم SESAME جهت مدیریت مخزن استفاده می شود و توسط ابزار OntoEdit ویرایش و بروز رسانی می گردد. OntoExtract، جهت ارتباط با SESAME از زبان پرس و جوی RQL[54] استفاده می کند. شکل زیر فرایند کلی استخراج اطلاعات را نشان می دهد.



شکل ۹: فرایند استخراج در OntoExtract [Dav03]

این فرایند، تکرار پذیر[55] می باشد. کاربر با تعیین دامنه اینترنتی مورد تحلیل (به عنوان مثال یک آدرس URL) ، فرایند را آغاز می نماید. به ازای هر صفحه موجود در دامنه تعیین شده که دارای متونی نوشته شده به زبان طبیعی می باشند، بصورت اتوماتیک، آنتالوژی های سبک وزن تولید می گردند و جهت ذخیره سازی به SESAME منتقل می گردند.

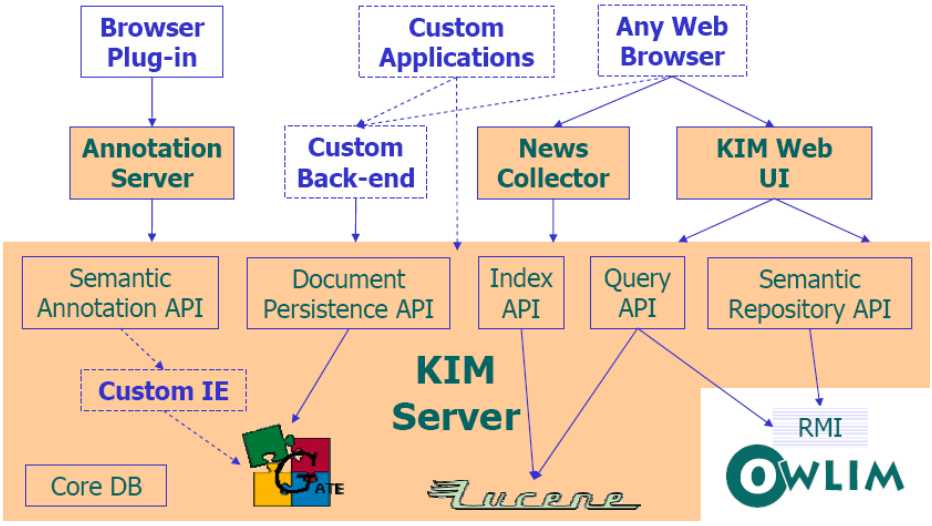
---

[54] RDF Query Language
[55] Iterative

OntoWrapper و TableAnalyzer: اغلب، مهندس آنتالوژی نیاز دارد تا نمونه های موجود در یک دامنه اینترنتی را که بصورت نیمه ساخت یافته یا تمام ساخت یافته ارائه شده اند، جمع آوری نماید. بدین منظور می توان از TableAnalyzer و OntoWrapper استفاده نمود. OntoWrapper، جهت استخراج موجودیت های موجود در یک صفحه از مجموعه ای از قواعد که بزبان RDF نوشته شده است. استفاده می نماید. بدین منظور مهندس آنتالوژی، مجموعه ای از الگوها، متغیر ها و قواعد را به زبان XML/RDF می نویسد. سپس این ماژول با استفاده از آنها، صفحات موجود را مرور کرده و اطلاعات مورد نظر را بازیابی می نماید. اطلاعات استخراج شده به فرمت سه تایی های RDF در مخزن SESAME ثبت و نگهداری می گردد.

KIM –

KIM مخفف Knowledge and Information Management بوده و پلتفورمی جهت حاشیه نویسی معنایی، جستجو و تحلیل می باشد [KIM07]. شکل زیر معماری KIM را نشان می دهد [Pop03]:



شکل ۱۰: معماری KIM [KIM07]

از این پلت فورم در موارد زیر استفاده می شود:

۱. حاشیه نویسی معنایی متن
۲. شاخص گذاری و بازیابی اطلاعات
۳. پرس و جو و اکتشاف دانش رسمی

۴. پیگیری وقایع همزمان و رتبه بندی موجودیت ها
۵. تحلیل محدوده زمانی محبوبیت و شهرت موجودیت

KIM، ترکیبی از سه معماری منبع باز[۵۶] زیر می باشد:

- GATE[۵۷] (معماری عمومی جهت مهندسی متن) مشهورترین پلتفورم نرم افزاری جهت مهندسی زبان می باشد که توسط گروه پردازش زبان طبیعی (NLP) در دانشگاه شفیلد طراحی و پیاده سازی شد. GATE یک زیرساختی را جهت تولید نرم افزارهای پردازش زبان طبیعی ارائه می کند. مولفه های GATE بسیاری از فعالیت های سطح پایین، مانند ذخیره سازی و بازیابی داده ها و پردازش زبان طبیعی را انجام می دهند [Cun02].
- SESAME : یک چارچوب منبع باز [SEA07 و Bro01] برای ذخیره سازی و پرس و جو و استنتاج از اطلاعات ذخیره شده در قالب RDF و RDF Schema می باشد. SESAME در فصل سوم معرفی شد.
- Apache Lucene: موتور جستجوی متن با کارایی بسیار بالا می باشد که بصورت یک کتابخانه از توابع به زبان جاوا، ارائه شده است. تکنولوژی Lucene تقریبا، برای هر برنامه کاربردی که نیاز به جستجوی کامل متن و خصوصا نیاز به اجرا در چند پلتفورم مختلف، مانند ویندوز و لینوکس و غیره دارد، مناسب می باشد [Luc07].

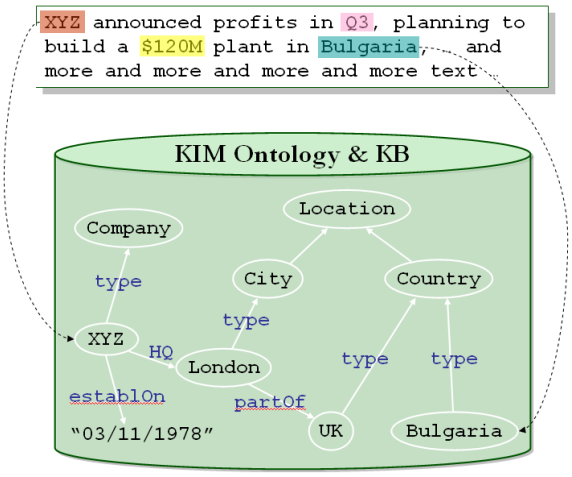
پلتفورم KIM شامل آنتالوژی KIM، پایگاه دانش، سرور KIM (شامل API هایی جهت دسترسی و ادغام و یک پارچه سازی از راه دور)، نرم افزار های نهایی[۵۸] (پلاگین برای مرورگر ها و رابط کاربری وب و مرورگر دانش). آنتالوژی KIM و پایگاه دانش آن در یک مخزن که مبتنی بر معماری SESAME [Dav03] می باشد، نگهداری می گردند. KIM یک زیر ساخت جدید برای استخراج داده ها که مبتنی بر پلتفورم GATE [Cun02] می باشد ارائه می کند. همچنین از پلتفورم Apache Lucene در شاخص گذاری مستندات بر اساس موجودیت های آنها و میزان ارتباط آنها با هم جهت بازیابی معنایی اطلاعات استفاده می کند.

برای آنکه برنامه های کاربردی کاملا اتوماتیک باشند، KIM دارای یک آنتالوژی سطح بالا و سبک وزن، بنام PROTON به همراه ۲۵۰ کلاس و ۱۰۰ خصوصیت می باشد. PROTON یک نمونه پیاده سازی

---

[56] Open Source
[57] General Architecture for Text Engineering
[58] Front-End

شده از آنتالوژی KIMO می باشد. بعلاوه پایگاه دانش KIM شامل ۲۰۰،۰۰۰ توصیف موجودیت به عنوان دانش اولیه می باشد.



شکل ۱۱: حاشیه گذاری در KIM [Sno02]

شکل فوق کاربرد آنتالوژی سطح بالای Proton و نیز دانش اولیه موجود در KIM را جهت پردازش یک جمله نشان می دهد.

ذکر این نکته مهم است که KIM یک پلتفورم است و می تواند توسط نرم افزار های دیگر مورد استفاده قرار گیرد تا در انجام حاشیه گذاری معنایی، بازیابی محتوی، پرس و جو های معنایی و غیره، از قابلیت های آن استفاده نمایند.

۳-۴- روش های آگاه از HTML[۵۹]

این نوع ابزار ها از روش های مبتنی بر خصوصیات ذاتی و ساختاری مستندات HTML استفاده می کنند. در این روش قبل از انجام عمل استخراج داده ابتدا برای یک فایل HTML درخت تجزیه ایجاد می گردد و برچسب های آن بصورت سلسله مراتبی (به شکل یک درخت) در حافظه ایجاد و نمایش داده می شود. سپس قواعد استخراج که مبتنی بر خصوصیات نمایشی داده ها است، بصورت اتوماتیک و یا نیمه اتوماتیک اجرا می گردند. فرق این روش نسبت به روش استفاده از زبان های پرس و جو آن است که در این روش سیستم نیازی به پردازش قواعد بازیابی (دستورات پرس و جو) را ندارد و قواعد بازیابی به صراحت محل وجود اقلام اطلاعاتی را مشخص می کند. در ضمن در این روش، امکان بکار گیری از ابزار های ویژوال در هنگام مشخص نمودن قواعد بازیابی، امکان پذیر می باشد. ولی تشخیص قواعد استخراج، همچنان بر عهده کاربر می باشد.

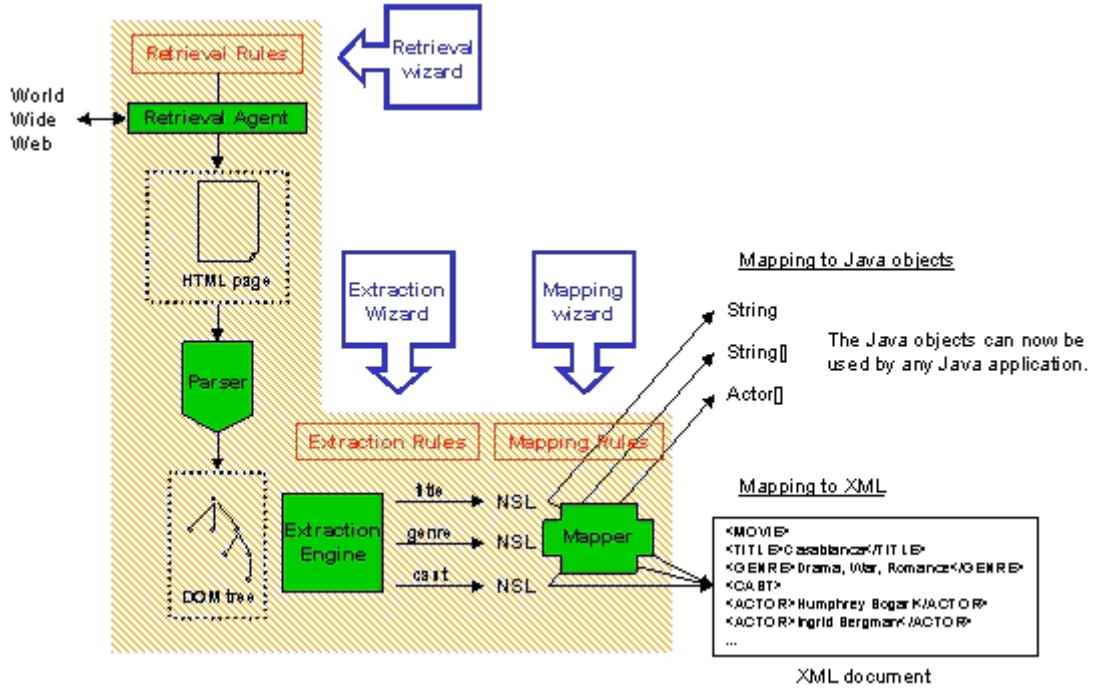
از جمله ابزارهایی که از این روش استفاده می کنند می توان به W4F [Sah00]، XWRAP اشاره کرد. که ذیلا به معرفی W4F می پردازیم.

W4F – (World Wide Web Wrapper Factory)

W4F [Sah00] یک جعبه ابزار[۶۰] برای ساخت Wrapper ها می باشد. W4F فرایند تولید نرم افزار را به سه مرحله تقسیم می کند. در ابتدا کاربر مشخص می کند که چگونه باید به مستندات دست یافت. به عبارتی آدرس URL سند مورد نظر را می دهد. سپس کاربر اقلام اطلاعاتی که باید استخراج شود را توصیف می کند. در نهایت باید ساختار مقصد، جهت ذخیره سازی و نگهداری داده های استخراج شده، مشخص گردد. شکل زیر معماری این سیستم را نشان می دهد.

---

[59] HTML Aware
[60] Toolkit

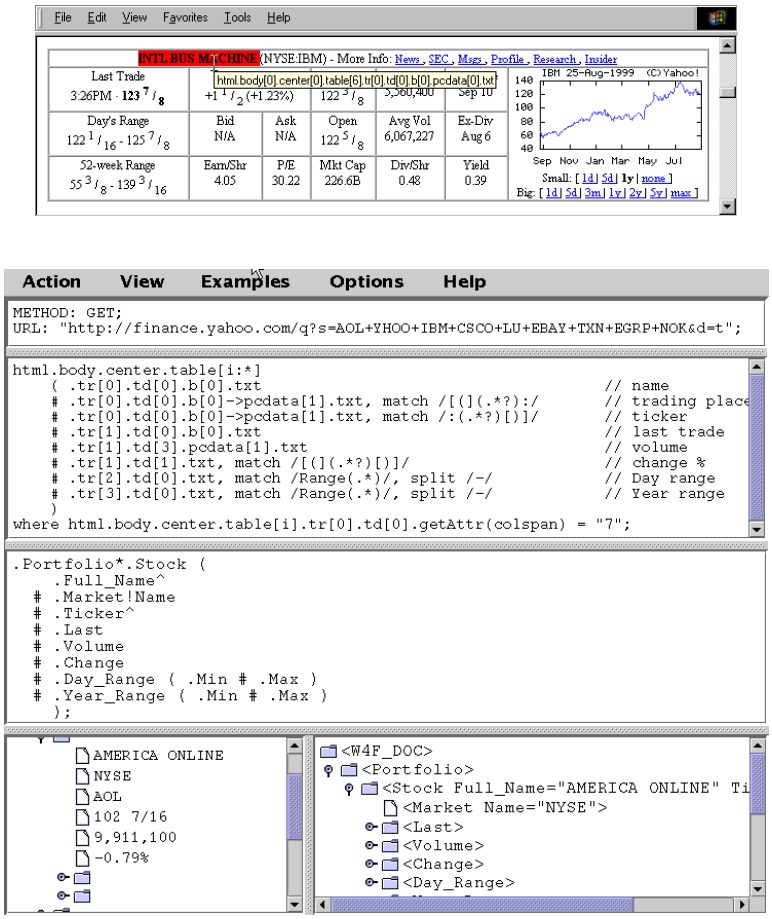


شکل ۱۲: معماری W4F [Sah00]

ابتدا سند مورد نظر مطابق با یک یا چند قاعده استخراج، بازیابی می شود. سند مورد نظر به محض بازیابی، به بخش تجزیه کننده HTML داده می شود تا یک درخت تجزیه مطابق با مدل شئ سند[۶۱] (DOM) ایجاد گردد. سپس کاربر می تواند قواعد استخراج را جهت یافتن اقلام اطلاعاتی در درخت تجزیه را مشخص نماید. این قواعد استخراج با فرمت داخلی W4F، با نام لیست رشته های تودرتو[۶۲] (NSL)، ثبت می گردد. در نهایت سازه های NSL، می توانند مطابق با تعیین قواعد نگاشت، به یک برنامه کاربردی سطح بالاتر منتقل گردند. زبان استفاده شده جهت تعریف قواعد استخراج HEL (HTML Extraction Language) نام دارد.

---

[61] Document Object Model
[62] Nested String List



شکل ۱۳: W4F Wizard [Sah00]

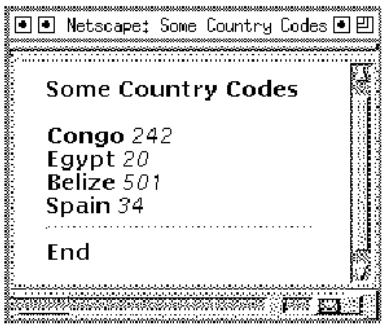
شکل فوق، بخشی از رابط کاربری مورد استفاده در W4F را نشان می دهد که جهت کمک به کاربران در نوشتن قواعد استخراج، تهیه شده است. ابتدا، کاربر بر روی اقلام اطلاعاتی مورد نظر خود، کلیک می کند (قسمت قرمز رنگ در شکل فوق). سپس سیستم، با توجه به محل قرار گرفته اطلاعات مورد نظر کاربر در ساختار درختی HTML، قواعد استخراج مربوطه را می سازد. در نهایت، کاربر می تواند، در صورت نیاز، آن قاعده را جهت تعمیم به مجموعه ای از رکورد های مشابه، ویرایش نماید.

۳-۵- روش های استنتاج Wrapper

این دسته از ابزار ها مجموعه ای از نمونه های آموزشی را دریافت کرده و بر اساس آنها قواعد استخراج را تولید می کنند. تفاوت اصلی این دسته از ابزارها با ابزار هایی از نوع NLP آن است که در اینجا به شرایط زبانی داده ها متکی نیستیم، بلکه به خصوصیات قالب بندی ای که بصورت غیر صریح خصوصیات ساختاری بخش هایی از داده های دریافت شده را طرح می کنند، متکی است.

– Wrapper Induction Environment (WIEN)

WIEN یکی از پیشتازان در تولید Wrapper به روش استنتاج می باشد [Ksh00]. در این روش مجموعه ای از صفحات به عنوان صفحات نمونه مثبت[۶۳]، که داده های مورد نظر در آن برچسب گذاری شده اند، جهت آموزش سیستم، جمع آوری می گردند. به عنوان نمونه در شکل زیر یک صفحه وب به همراه کد، HTML آن، نشان داده شده است. سیستم این صفحه را با سایر صفحات آموزشی دیگر مقایسه کرده و سعی می کند قواعدی را جهت استخراج اقلام متغیر در بین صفحات بیابد. در این مثال نام کشور ها و کد های آنها، مقادیر متغیر می باشند.




```
<HTML><TITLE>Some Country Codes</TITLE>
<BODY><B>Some Country Codes</B><P>
<B>Congo</B> <I>242</I><BR>
<B>Egypt</B> <I>20</I><BR>
<B>Belize</B> <I>501</I><BR>
<B>Spain</B> <I>34</I><BR>
<HR><B>End</B></BODY></HTML>
```

شکل ۱۴ : نمونه ای از یک صفحه وب جهت آموزش سیستم WIEN [Ksh00]

[63] Positive Sample Pages

یکی از فرضیات اولیه این سیستم آن است که تمام صفحات دارای ساختار از پیش تعریف شده HLRT می باشند در این ساختار صفحات دارای یک سربرگ[64] بوده و اقلام اطلاعاتی با مولفه های مشخصی در سمت چپ و راست آنها قابل تشخیص می باشند. در نهایت هر صفحه دارای یک بخش پایانی[65] می باشد.

در مثال فوق سربرگ صفحات، رشته زیر می باشد:

```
<HTML><TITLE>Some Country Codes </TITLE><BODY>
```

بخش پایانی صفحات نیز رشته زیر می باشد:

```
<HR><B>END</B></BODY></HTML>
```

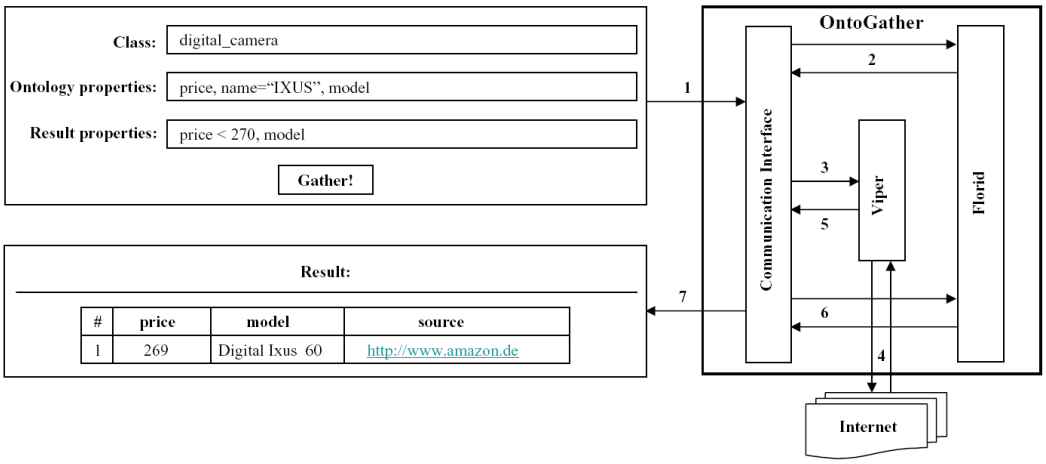
رشته سمت چپ نام کشور <B> و رشته سمت راست آن </B> و نیز رشته سمت چپ کد کشور <I> و رشته سمت راست آن </I> می باشد. سیستم WIEN با مقایسه صفحات نمونه مثبت، رشته های منتسب به ساختار HLRT را شناسایی کرده و بدین ترتیب قواعد استخراج را مشخص می کند.

OntoGather –

OntoGather [Hmn06] یک موتور پرس و جوی منابع وب مبتنی بر آنتالوژی می باشد. یکی از خصوصیات بارز این سیستم، استخراج لحظه ای اطلاعات می باشد. به عبارتی، پرس و جو ها بر روی داده های آنلاین انجام می گیرد و کاربر می تواند آخرین اطلاعات را حداکثر با چند دقیقه تاخیر مشاهده نماید. OntoGather از یک سیستم کاملا اتوماتیک جهت استخراج اطلاعات موجود در وب، بنام ViPER [Sim05] استفاده می کند و در کل دو فعالیت مهم را به موازات هم انجام می دهد. اول انتخاب منابع مرتبط با پرس و جوی مورد نظر و دوم استدلال بر روی داده های استخراج شده توسط سیستم ViPER. شکل زیر یک دید کلی از این سیستم را ارائه می کند.

---

[64] Header
[65] Tail



شکل ۱۵: مرور کلی سیستم OntoGather [Hmn06]

OntoGather شامل دو کامپوننت اساسی است:

- **FLORID** : FLORID [Fro97] یک سیستم بانک اطلاعاتی شی گرا و استنتاجی است که از F-Logic [FL07] جهت تعریف داده ها و زبان پرس و جو، استفاده می نماید. FLORID هسته مرکزی OntoGather را تشکیل می دهد. در واقع FLORID موتور استنتاج OntoGather جهت انتخاب منابع وب و ارزیابی پرس و جو های کاربر می باشد.

- **ViPER**[٦٦]: ViPER [Sim05] ابزاری جهت استخراج اتوماتیک داده ها از منابع وب می باشد. این ابزار جهت استخراج داد ها، در صفحات وب، به دنبال تکرار الگوی رکورد ها می گردد و برای پیدا کردن این الگو ها به شیوه نمایشی داده های موجود در صفحات وب تمرکز می کند. از این رو برای آنکه ViPER بتواند درست کار کند حداقل نیاز است تا دو رکورد وجود داشته باشد. این الگو ها همان قواعد استخراج را مشخص می کنند که در OntoGather جهت جمع آوری اطلاعات از منابع وب استفاده می گردد.

[66] Visual Perception-based Extraction of Records

در OntoGather منابع اطلاعاتی باید مرتبط با یک آنتالوژی دامنه باشند. این آنتالوژی شامل توصیف اطلاعات موجود در منابع وب می باشد و با زبان F-Logic [FL07] نشان داده شده و در سیستم FLORID ذخیره می گردند. با رجوع به شکل فوق، فرایند پرس و جو، در هفت مرحله انجام می گیرد:

۱. ابتدا کاربر از طریق یک رابط کاربری، مشخصات موضوع مورد جستجو را وارد می کند. در این مثال کاربر بدنبال دوربین های دیجیتالی است که قیمت آنها کمتر از ۲۷۰ باشد. دوربین در ضمن دوربین دیجیتال یکی از کلاس های موجود در آنتالوژی دامنه می باشد.
۲. سپس درخواست مورد نظر به واسط ارتباطی ارسال می گردد. این واسط از FLORID درخواست پرس و جوی مورد نظر را ارسال می کند. FLORID نیز لیستی از منابع موجود را که حاوی اطلاعات مورد نظر می باشند به همراه سایر مشخصات پرس و جو، به عنوان جواب ارسال می کند.
۳. این اطلاعات برای ViPER ارسال می گردد.
۴. ViPER منابع وب را کاوش کرده و اطلاعات درخواستی را استخراج می کند.
۵. اطلاعات استخراج شده برای واسط ارتباطی ارسال می گردد.
۶. این نتایج به Fact های F-Logic تبدیل شده و نمونه های جدید در آنتالوژی ذخیره می شوند.
۷. نتایج نهایی برای ارائه به کاربر، برای واسط کاربری ارسال می گردد.

۳-۶- روش های مبتنی بر آنتالوژی

اکثر روش های قبلی مبتنی بر ساختار و نحوه نمایش اقلام اطلاعاتی بوده اند و بـر اسـاس آن قواعـد، الگو های استخراج داده ها مشخص می شدند. اما علاوه بر ساختار نمایش داد ها، می توان اقلام اطلاعـاتی را با تکیه بر خود داده ها نیز استخراج نمود. بدین منظور یک آنتالوژی برای یک دامنه مشخص ایجـاد مـی شود. تا بوسیله آن ثابت های موجود در یک صفحه وب شناسایی شده و سپس بر اساس آن، اشیاء موجود استخراج گردند.

از آنجا که داده های مورد نظر، به روش های مختلف سازماندهی می شود، بهتر است تا آنها را مطابق با یک مدل عمومی[۶۷] و مستقل از منابع اطلاعاتی سازماندهی مجـدد نمـاییم. بـه ایـن ترتیـب، اسـتخراج و ترکیب داده ها از منابع اطلاعاتی مختلف ساده تر و قابل اعتماد تر خواهد بود. آنتـالوژی هـا بیـشتر جهـت برای استخراج اطلاعات از منابع غیر ساختیافته استفاده می شوند. متاسـفانه ابزارهـایی کـه آنتـالوژی هـا را جهت استخراج از منابع نیمه ساختیافته بکار ببرند بسیار محدود می باشد. ذیلا به معرفـی سیـستم پیـشنهاد شده در دانشگاه Brigham Young، بنام BYU-Ontos می پردازیم.

BYU-Ontos –

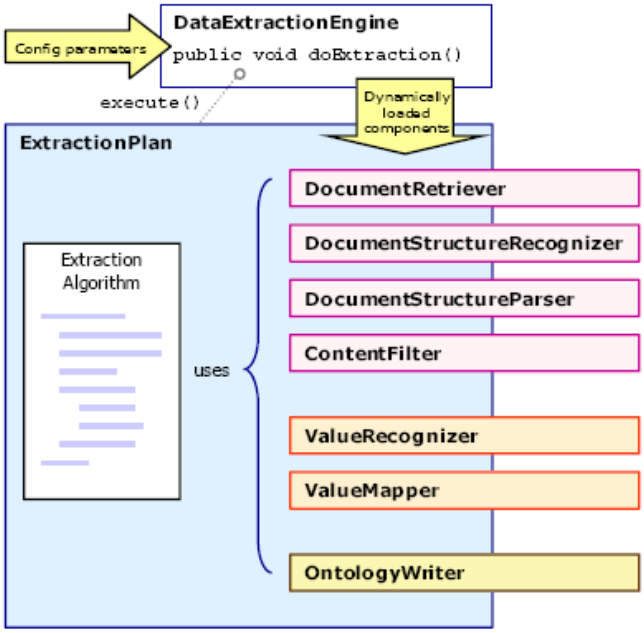
این ابزار توسط گروه استخراج داده در دانشگاه Brigham Young سـاخته شـده اسـت [Wes05]. در این ابزار، آنتالوژی ها جهت توصیف داده های مورد نظر شامل رابطه ها، واژگان مـورد اسـتفاده و کلمـات کلیدی زمینه، استفاده می گردد. این ابزار می تواند جهت استخراج، با تجزیه و تحلیل آنتالوژی دامنه، اقـلام داده ای مورد نظر را تشخیص داده و یک بانک اطلاعاتی از اطلاعات استخراج شـده ایجـاد نمایـد. قبـل از استفاده از آنتالوژی دامنه، باید توسط یک روتین اتوماتیک، بخش هایی از سند را کـه شـامل اقـلام داده ای مورد نظر می باشند، مشخص نمود.

شکل زیر، چارچوب استخراج اطلاعات در سیستم BYU-Ontos را نشان می دهد. فراینـد اسـتخراج از ماژول DataExtractionEngine شروع می شود. این ماژول، مسئولیت مقدار دهی اولیه سیستم و شروع

---

[67] Common Model

اجرای فرایند استخراج را بر عهده دارد. این ماژول، به اینترفیس هایی مانند ویرایشگر آنتالوژی ها (Ontology Editor) اجازه می دهد تا در یک سطح بالاتری با سایر اجزا و مولفه های سیستم ارتباط برقرار سازند.



شکل ۱۶: چارچوب استخراج اطلاعات در سیستم BYU-Ontos [Wes05]

واسط[۶۸] Document Retriever وظیفه جمع آوری محتویات صفحات وب را برعهده دارد. این کامپوننت، به آدرس یک صفحه وب را به عنوان ورودی دریافت کرده و سپس محتویات آن را از اینترنت خوانده و به عنوان خروجی بر می گرداند. سپس محتوای صفحه وب به واحد Document Structure Recognizer داده می شود. این واحد محتوای صفحه را تحلیل کرده و تعیین می کند که کدامیک از تجزیه کننده[۶۹] های موجود برای پردازش صفحه مورد نظر مناسب می باشد. سپس محتویات صفحه به واحد Document Structure Parser مربوطه ارسال می گردد. در این ماژول یک سند وب ممکن است به چند سند کوچکتر تقسیم شود تا عملیات پردازش بر روی بخش های کوچکتر را بتوان به راحتی انجام داد.

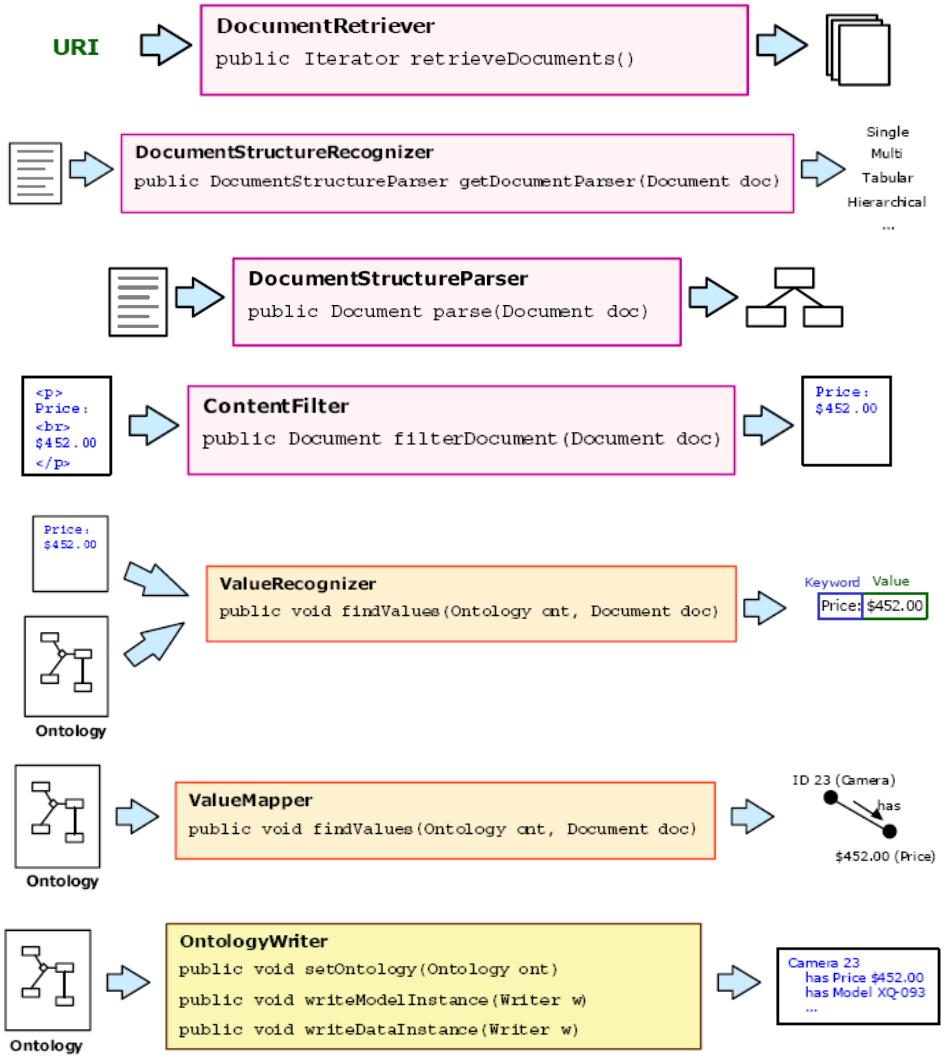
اکثر صفحات وب شامل داده های معنی دار و با اطلاعات قالب بندی[۷۰] می باشند. یک صفحه وب شامل تگ های بسیار زیادی است که مشخص می کند یک صفحه وب چگونه باید در یک مرورگر نشان داده شود. اما این تگ ها معمولا اطلاعات بیشتری در ارتباط با مفهوم آنچه که نمایش داده می شود، ارائه

---

[68] Interface
[69] Parser
[70] Formatting Information

نمی کنند. از این رو ماژول Content Filter اقدام به حذف تگ های HTML می نماید. البته این ماژول اختیاری بوده و سیستم بدون حذف تگ ها نیز می تواند کار کند.

خروجی ماژول Content Filter که یک متن مسطح[۷۱] می باشد به ماژول Value Recognizer ارسال می گرد. این ماژول نقش کلیدی را در استخراج اطلاعات ایفا کرده و مسئولیت اصلی آن اعمال قواعد تشخیص مرتبط با آنتالوژی استخراج و یافتن مجموعه از کاندید ها جهت استخراج می باشد. از آنجا که به ازای هر بخش از صفحه وب ممکن است چند کاندید در آنتالوژی استخراج یافت شود، لذا، درصورت یافتن دو یا چند کاندید، باید مشخص نمود که آن بخش از صفحه مربوط به کدام بخش از آنتالوژی می باشد.



شکل ۱۷: ماژول های سیستم BYU-Ontos [Wes05]

[71] Plain Text

یکی از مهمترین بخش های این سیستم ماژول ValueMapper می باشد. وظیفه اصلی این ماژول، نگاشت مقادیر کاندید به اجزای آنتالوژی می باشد. این ماژول چهار عملیات اصلی زیر را انجام می دهد:

۱. حذف آن اجزایی از آنتالوژی موجود در لیست کاندید، که با مقدار مورد بحث ناسازگاری دارند.
۲. تبدیل مقادیر لغوی به اشیاء (در اینجا منظور از شی، یک نمونه از مفاهیم موجود در آنتالوژی می باشد)
۳. اجرای فرایند استنتاج و یافتن اشیایی که بطور مستقیم در وجود ندارند.
۴. استنتاج روابط میان اشیاء

خروجی این ماژول، یک نمونه داده، شامل مجموعه ای از اشیاء و روابط بین آنها، خواهد بود. در نهایت ماژول Ontology Writer، خروجی داده های استخراج شده را در قالب یک آنتالوژی نمونه، پیاده سازی می کند.

برای آن که این سیستم بدرستی کار کند، باید آنتالوژی اولیه بدرستی و با دقت زیاد ایجاد گردد. ساخت این آنتالوژی بر عهده فرد خبره در دامنه آن آنتالوژی می باشد. در صورتیکه آنتالوژی اولیه به اندازه کافی بیانگر و تشریحی باشد، روال استخراج می تواند کاملا اتوماتیک انجام شود. ضمن آن که Wrapper های تولید شده به این روش، ذاتا نسبت به تغییرات ساختاری مستندات مقاوم می باشند. در واقع این مزیت ها به دلیل استفاده از آنتالوژی ها در تشخیص اقلام داده ای می باشد.

۳-۷- خلاصه مطالب و نتیجه گیری

در این فصل به بررسی انواع روش های موجـود در تولیـد Wrapper هـا پـرداختیم. اولـین روش در تولید Wrapper ها استفاده از یک زبان پرس و جو می باشد که قادر است المان ها موجود در یک یا چنـد صفحه وب را بازیابی نماید. ایجاد دستی دستورات پرس و جو بسیار سخت می باشـد از ایـن رو در روش های آگاه از HTML، از نماد های ساده تری جهت آدرس دهـی محـل اطلاعـات موجـود در صـفحه وب استفاده شده است. نیاز به آگاهی از ساختار صفحات یکی از مشکلات در اتوماتیک سازی فرایند اسـتخراج اطلاعات می باشد. از این رو در روش های مبتنی بـر اسـتنتاج، ابتـدا سیـستم بـا اسـتفاده از چنـد صـفحه برچسب گذاری شده، به عنوان صفحات آموزشی، قواعد استخراج را شناسایی کرده و سپس، از آن قواعـد جهت استخراج اطلاعات از صفحات مشابه استفاده می نماید.

اغلب در روش های مبتنی بر استنتاج، تهیه صفحات آموزشی مناسب بسیار دشوار می باشد. از این رو روش های مبتنی بر آنتالوژی، با استفاده از یک آنتالوژی استخراج اقدام بـه شناسـایی اقـلام اطلاعـاتی مـی نمایند. آنتالوژی استخراج یک مدل مفهومی نمونه از دامنه مورد نظر می باشد که در آن بیشتر بـه توصـیف مقادیر اطلاعاتی و نظم موجود در آنها پرداخته شده است. شکل زیر مقایسه ای بین روش هـای موجـود را ارائه می کند.

ابزار ها		میزان اتوماتیک سازی استخراج	سهولت استفاده	پشتیبانی از منابع غیر HTML	روش استخراج
روش های مبتنی بر زبان پرس و جو	WebOQL	دستی	متوسط	خیر	Location-Based
روش های آگاه از HTML	W4F	نیمه اتوماتیک	زیاد	خیر	Location-Based
	Road Runner	اتوماتیک	بسیار زیاد	خیر	Location-Based
روش های مبتنی بر استنتاج	WIEN, STALKER	نیمه اتوماتیک	زیاد	خیر	Location-Based
روش های مبتنی بر آنتالوژی	BYU-Ontos	اتوماتیک	زیاد	بله	Ontology-Based

جدول ۲: مقایسه میان روش های تولید Wrapper

با توجه به این موضوع که استخراج اطلاعات روشی جهت دسترسی به اطلاعات و پـردازش آنهـا در سیستم هایی چون وب معنایی می باشد، لذا سرعت، دقت و کارایی این زیر سیـستم بـسیار دارای اهمیـت

می باشد. ولی متاسفانه سرعت و دقت در مقابل هم قرار دارند بدین معنی که هرچه سیستم دارای سرعت بیشتری باشد کارایی آن کمتر بوده و برعکس افزایش کارایی باعث کاهش سرعت در استخراج اطلاعات می گردد.

در بین روش های بررسی شده فوق، روش هایی که با آگاهی از محل وقوع اطلاعات، اقدام به بازیابی اطلاعات می نمایند[۷۲]، دارای سرعت استخراج بیشتری می باشند. در این روش ها از آنجا که عملیات شناسایی محل وقوع اقلام اطلاعاتی قبلا صورت گرفته است زمانی برای آن صرف نمی شود لذا سرعت استخراج بسیار بالا است. ولی از آنجا که جهت شناسایی اقلام اطلاعاتی نیاز به دخالت و نظارت انسان می باشد، درجه خودکار سازی فرایند بسیار پایین می باشد.

روش های مبتنی بر آنتالوژی و پردازش زبان طبیعی نیز از آنجا که در حین استخراج اطلاعات، فرایند شناسایی اقلام اطلاعاتی را اجرا می نمایند دارای زمان استخراج بیشتری می باشند. این در حالی است که بسیاری از صفحات وب دارای ساختاری ساده و منظم بوده و نیازی به اجرای فرایند های سنگین تشخیص اقلام اطلاعاتی ندارند.

علاوه بر مشکلات فوق، بدلیل فقدان یک زبان مشترک در توصیف Wrapper ها، در روش های جاری امکان به اشتراک گذاری آن ها وجود ندارد. بدین ترتیب قابلیت نقل و انتقال Wrapper ها نیز وجود ندارد. از طرفی دیگر، جهت یکپارچه سازی سیستم ها و نیزآنتالوژی هایی که از Wrapper های متفاوت استفاده می کنند، سریع ترین و بهترین گزینه ترکیب Wrapper ها می باشد ولی بدلیل عدم وجود یک تعریف مشترک جهت توصیف Wrapper ها، عملیات یکپارچه سازی بسیار مشکل و پیچیده می گردد.

روش پیشنهادی ما جهت غلبه بر این مشکلات، ایجاد یک زبان مشترک جهت توصیف Wrapper ها می باشد. استفاده از این زبان جهت تعریف Wrapper ها ، سبب افزایش قابلیت جابجایی[۷۳]، قابلیت استفاده مجدد[۷۴] و نیز افزایش سرعت و دقت در فرایند استخراج خواهد شد. در ادامه این پایان نامه به معرفی جزئیات طرح خود خواهیم پرداخت.

---

[72] روش های مبتنی بر مکان با Location Based

[73] Portability

[74] Reuseability

۴- معرفی سیستم OntoByOnto

در این فصل با ایده های کلی روش پیشنهادی آشنا خواهیم شد. این روش، منتهی به معرفی یک سیستم جدید به نام OntoByOnto خواهد شد که معماری آن را در این فصل بررسی خواهیم کرد. برای تشریح این سیستم آن را به دو بخش استخراج اطلاعات و شناسایی اتوماتیک الگوها تقسیم می کنیم. در این بخش، ضمن آشنایی با ایده های کلی روش پیشنهادی، نحوه استخراج اطلاعات را نیز تشریح خواهیم کرد. در فصل بعد به معرفی نحوه شناسایی اتوماتیک الگوها خواهیم پرداخت.

۴-۱- مقدمه

مهمترین چالش های مرتبط با روش های بررسی شده عبارتند از؛ سرعت استخراج اطلاعات، دقت و صحت اقلام اطلاعاتی استخراج شده، امکان انتقال Wrapper ها به سایر سیستم ها و استفاده مجدد از آنها و همچنین امکان یکپارچگی سیستم های استخراج اطلاعات. روش های مبتنی بر استنتاج، قواعد استخراج را با توجه به محل قرار گیری اقلام اطلاعاتی مشخص می کنند. از این رو در هنگام استخراج اطلاعات با آگاهی از محل قرار گیری اطلاعات به سرعت اقلام اطلاعاتی را استخراج می کنند. اما متاسفانه این سیستم ها هیچ کنترلی بر صحت اقلام استخراج شده نداشته و در صورت تغییر در صفحات وب از کار می افتند.

در مقابل، روش مبتنی بر آنتالوژی با استفاده از آنتالوژی استخراج، قادرند تا بصورت اتوماتیک اقلام اطلاعاتی موجود در صفحات وب را شناسایی نموده و آنها را در یک آنتالوژی نمونه و یا یک مخزن اطلاعاتی ذخیره نمایند. این روش بدلیل پردازش صفات وب با استفاده از آنتالوژی استخراج سرعت اجرای بسیار پایینی دارد و بکار گیری آن در یک محیط عملیاتی شامل میلیون ها صفحه وب، به هیچ وجه مناسب نمی باشد. اما در مقابل این کاهش سرعت، دقت استخراج اطلاعات در این روش بسیار بالا می باشد و همچنین می توان با تهیه یک آنتالوژی استخراج کامل، فرایند شناسایی و استخراج اقلام اطلاعاتی را اتوماتیک نمود.

ایده اصلی ما شامل ترکیبی از روش های مبتنی بر آنتالوژی و مبتنی بر استنتاج می باشد. بدین ترتیب که ابتدا با استفاده از آنتالوژی استخراج، ساختار درختی تگ های صفحه وب را به یک ساختار درختی از مفاهیم موجود تبدیل نموده و سپس با استفاده از روش های استنتاج، اقدام به شناسایی الگو ها و محدوده

رکورد ها می نماییم. در ادامه و بعد از شناسایی الگو های استخراج، جهت ایجاد قابلیت استفاده مجدد و نیز قابلیت نقل و انتقال Wrapper، الگوهای بدست آمده را با استفاده از زبان پیشنهادی جدیدی بنام WDML، به یک ساختار مشخص و منظم تبدیل می کنیم.

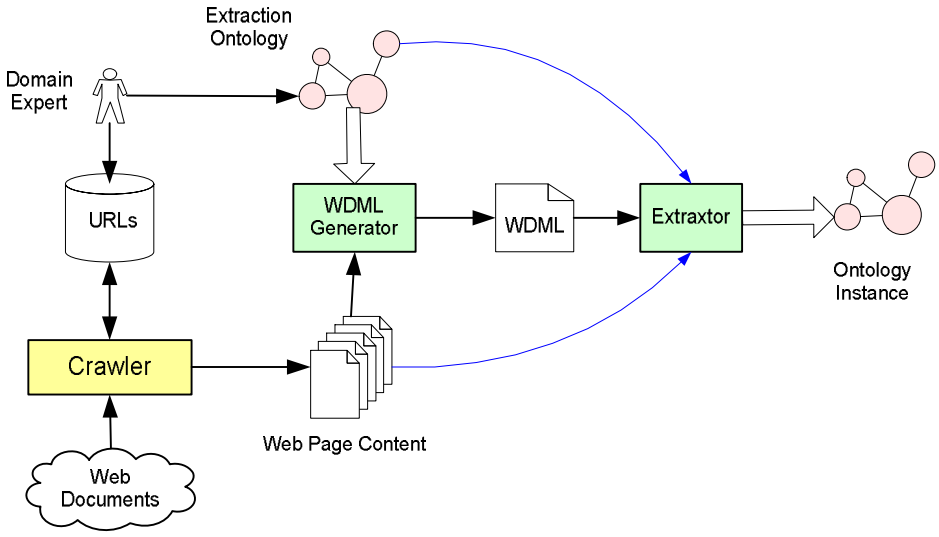
ما این روش پیشنهادی را در قالب یک سیستم بنام OntoByOnto تشریح نموده و عملکرد آن را به دو ماژول مجزا، یکی جهت استخراج اطلاعات با استفاده از الگو های تعریف شده در فایل های WDML و دیگری شناسایی و تولید اتوماتیک الگو ها از صفحات وب تقسیم می کنیم. در این فصل به معرفی زبان WDML و مشخصات ماژول استخراج پرداخته و سپس در فصل بعد به معرفی الگوریتم های مورد نیاز جهت شناسایی اتوماتیک الگو های استخراج می پردازیم.

۴-۲- معماری سیستم OntoByOnto

اولین قدم در سیستم OntoByOnto تعریف آنتالوژی استخراج توسط مهندس دامنه می باشد. آنتالوژی استخراج یک نمونه مفهومی از مقادیر دامنه مورد بحث است. در واقع آنتالوژی استخراج توصیف کننده مجموعه مقادیری است که می توان به خصوصیات تعریف شده در آنتالوژی دامنه نسبت داد. بدین ترتیب هرچه این آنتالوژی دقیق تر و کامل تر باشد، سیستم قادر خواهد بود با دقت بسیار بالاتری اطلاعات موجود را شناسایی کند.

یکی از ویژگی های بارز این سیستم، بهره گیری از زبان توصیف Wrapper می باشد. این زبان مبتنی بر XML بوده و Wrapper های نوشته شده به وسیله آن به راحتی قابل نقل و انتقال به سایر سیستم ها می باشند. در این زبان، الگو های استخراج بصورت سلسله مراتبی تعریف می گردد و محل وقوع هر یک از اقلام اطلاعاتی، با استفاده از یک المان خاص، دقیقا مشخص می گردد. بدین ترتیب فرایند استخراج با آگاهی از محل وقوع اقلام اطلاعاتی به سرعت اقدام به استخراج اطلاعات می نماید. البته در مواقعی که نیاز به کنترل اقلام اطلاعاتی باشد، می توان جهت کنترل صحت عملیات استخراج، به آنتالوژی استخراج مربوطه ارجاع داد. در نتیجه علاوه بر سرعت، صحت و دقت اقلام اطلاعاتی استخراج شده نیز افزایش چشمگیری خواهد داشت.

شکل زیر معماری سیستم OntoByOnto را نشان می دهد.



شکل ۱۸: معماری سیستم OntoByOnto

مولفه های اصلی این سیستم عبارتند از:

- **خزشگر**
- **ماژول WDML Generator**
- **ماژول Extractor**

کارشناس دامنه بعد از معرفی آنتالوژی استخراج، لیستی از صفحات آغازین از وب سایت هایی که شامل اطلاعات مورد نظر می باشند را تهیه می کند. سپس خزشگر لیست صفحات استخراج نشده را خوانده و به ازای هر از آنها ، محتویات آن صفحه را بررسی کرده و لیست پیوند ها به سایر صفحات را از آن استخراج می نماید. خزشگر آدرس صفحات یافت شده جدید را بررسی کرده و در صورتیکه قبلا این صفحات بازیابی نشده اند، به لیست صفحات بازیابی نشده اضافه می کند.

بعد از عملیات جمع آوری صفحات وب توسط خزشگر، ماژول WDML Generator به ازای هر یک از وب سایت های بازیابی شده، فایل های WDML مرتبط با آن را تولید می نماید. این ماژول جهت یافتن اطلاعات موجود در یک صفحه وب، از مفاهیم تعریف شده در صفحه وب و نیز از ساختار نمایشی اطلاعات استفاده خواهد نمود. یکی از عملیات کلیدی این ماژول، انطباق بین اقلام داده ای خام (یافت شده در صفحات وب) و مفاهیم موجود در آنتالوژیِ استخراج می باشد. ما این بخش از سیستم را پیاده سازی کردیم تا راه حل هایی عملی، برای آن مشخص سازیم. در روش پیشنهادی ما انطباق بین اقلام داده ای و مفاهیم آنتالوژی بصورت نسبی می باشد. به این معنی که نتیجه انطباق عددی بین صفر تا صد خواهد بود. بدین ترتیب می توان مفاهیم مرتبط با یک داده خام را برحسب درجه انطباق آنها مرتب نزولی کرد تا اولین عنصر لیست به عنوان بهترین کاندید در ادامه فرایند استخراج مورد استفاده قرار گیرد.

فایل WDML تولید شده در قسمت قبل باید توسط ماژول Extractor خوانده شده و قواعد استخراج آن مشخص گردد. سپس این قواعد استخراج، بر روی صفحات وب مربوطه اعمال گشته و اقلام اطلاعاتی آن مشخص می گردد. همچنین این ماژول قادر است تا اقلام اطلاعاتی را در قالب یک آنتالوژی نمونه، ارائه نماید.

در بخش های بعد به تفصیل جزئیات بیشتر هر یک از مولفه های فوق می پردازیم. اما ابتدا جهت آشنایی اولیه، به معرفی زبان WDML و نیز آنتالوژی استخراج خواهیم پرداخت.

۴-۳- WDML[۷۵]: زبان تعریف Wrapper

زبان تعریف Wrapper (WDML) یک زبان نشانه گذاری و مبتنی بر XML می باشد. هدف از تولید این زبان، جدا کردن قواعد استخراج از روتین اجرای Wrapper ها می باشد. بدین ترتیب، در صورت نیاز کاربران نیز قادر خواهند بود با استفاده از این زبان نشانه گذاری، به راحتی قواعد استخراج را مشخص نمایند. البته برای ایجاد دستی قواعد استخراج لازم است تا با ساختار صفحات وب آشنا باشیم. ولی ما در این فصل الگوریتمی را ارائه خواهیم کرد که بدون نیاز به آگاهی قبلی از ساختار صفحات و نیز بدون نیاز به آموزش اولیه، تنها استفاده از آنتالوژی استخراج و ساختار نمایشی داده ها بتوان الگو ها و نیز اقلام اطلاعاتی موجود را شناسایی نمود.

ما در اینجا قصد نداریم یک نسخه نهایی شده از این زبان پیشنهادی را معرفی کنیم بلکه تا آنجا پیش خواهیم رفت که نیازمندی های ما را پوشش دهد. از طرفی دیگر، از آنجا که WDML قواعد استخراج را در قالب ساختار درختی فایل های HTML نشان می دهد، ممکن است شباهت هایی بین آن و زبان هایی مانند XSD و XSTL یافت شود. ما این موضوع را به تحقیقات بیشتر در این زمینه محول می کنیم. اما آنچه که در اینجا می توان گفت آن است که WDML نیازها و ویژگی هایی دارد که در زبان های جاری وجود ندارد. در ادامه به معرفی ویژگی های این زبان می پردازیم.

WDML از فرمت XML پیروی می نماید. فایل های WDML دارای ساختاری مشابه با قالب زیر می باشد:

```
<?xml version="1.0"?>
<WDML ontology="">
  ...
</WDML>
```

گاهی اوقات ممکن است لازم باشد تا در یک فایل WDML به برخی اطلاعات موجود در آنتالوژی استخراج، ارجاع دهیم. بدین منظور باید خصوصیات ontology از المان WDML را با آدرس آنتالوژی استخراج مقداردهی نماییم. البته این قابلیت اختیاری می باشد و در صورت عدم نیاز می توان آن را حذف نمود.

---

[75] Wrapper Difinition Markup Language

هر فایل WMDL می تواند، شامل یک یا چند Wrapper باشد. هر Wrapper بـا المـان <wrapper> مشخص شده و باید دارای یک نام مشخص باشد. این نام می تواند مرتبط با اطلاعاتی باشد کـه اسـتخراج می نماید.

```
<WDML>
  <wrapper id="CellPhone">
        <resources>
        ...
        </resources>

        <output>
        ...
        </output>

        <htmlbody>
        ...
        </htmlbody>

  </wrapper>
  ...
</WDML>
```

هر Wrapper، شامل ۳ بخش مهم زیر می باشد:

- معرفی مستندات ورودی
- معرفی قالب خروجی
- معرفی الگوی استخراج

ذیلا به معرفی هر یک از بخش های فوق می پردازیم.

۴-۳-۱- معرفی مستندات ورودی

منظور از مستندات ورودی، مجموعه ای از صفحات وب می باشد که باید اقلام اطلاعاتی را از آن هـا استخراج نمود. در WDML، بخش مستندات ورودی با المان < resources> مشخص می شود. هر صفحه وب نیز با استفاده از آدرس url آن مشخص می شود. آدرس صفحات باید توسـط المـان <url> مـشخص شود. از آنجا که هر Wrapper می تواند بر روی چند صفحه وب با ساختار و مفاهیم مشابه اجـرا گـردد، و نیز با توجه به اینکه معمولا آدرس صفحات مشابه دارای نظـم خاصـی مـی باشـد، لـذا، در تعریـف آدرس صفحات مشابه می توان از حروف ? و * استفاده نمود. کد زیر ساختار کلـی معرفـی مـستندات ورودی را نشان می دهد.

```
<wrapper>
   ...
   <resources>
      <url> web page url (include wildcards) </url>
      ...
   </resources>
   ...
</wrapper>
```

باید توجه داشت که آدرس صفحات باید بصورت کد شده باشد. برای اطلاعات بیشتر می توانید به مراجع XML[۷۶] مراجعه نمایید. ذیلا مثالی از تعریف صفحه K750 به عنوان ورودی Wrapper ارائه می گردد.

```
<wrapper id="CellPhone">
    ...
    <resources>
          <url>http://www.gsmarena.com/sony_ericsson_*.php</url>
          <url>http://www.gsmarena.com/nokia_*.php</url>
    </resources>
    ...
</wrapper>
```

در مثال فوق، صفحاتی که در سایت www.gsmarena.com وجود داشته و نام صفحه با عبارت sony_ericsson_ یا nokia_ شروع شده و به .php ختم می شوند را به عنوان ورودی Wrapper تعریف می کند.

۴-۳-۲- معرفی قالب خروجی

Wrapper ها پس از اجرای فرایند استخراج، باید اطلاعات را تحت یک ساختار دقیق و مشخص در خروجی بنویسند. در WDML می توان خروجی Wrapper ها را یک فایل xml، owl و یا یک جدول بانک اطلاعاتی تعریف نمود. این بخش با المان <output> ، بصورت زیر، تعریف می گردد.

```
<wrapper id="CellPhone">
   ...
   <output type="owl/xml/database">
      <connectionString> A connection string </connectionString>
      <schema> existing schema file name </schema>
      <destination> output file / table </destination>
   </output>

   ...
</wrapper>
```

[76] http://www.w3c.com

همانطور که دیده می شود، نوع خروجی را با خصوصیت type مشخص می کنیم و می تواند یکی از مقادیر زیر باشد:

- xml : جهت تعریف یک فایل xml به عنوان خروجی Wrapper از این نوع خروجی استفاده می گردد و در صورت استفاده از آن باید، اطلاعات مربوط به فایل اسکیما (xsd) را توسط المان <schema> و فایل خروجی را توسط المان <destination> تعیین نمود.
- owl : جهت تعریف آنتالوژی به عنوان خروجی Wrapper از این نوع خروجی استفاده می گردد و در صورت استفاده از آن باید، اطلاعات مربوط به فایل آنتالوژی دامنه را توسط المان <schema> و فایل خروجی را توسط المان <destination> تعیین نمود.
- database : جهت تعریف یک بانک اطلاعاتی جهت خروجی Wrapper استفاده می شود و در صورت استفاده از آن باید، اطلاعات مربوط به رشته اتصال[۷۷] را توسط المان <connectionString> تعیین نمود.

در مثال زیر جدول CellPhones از بانک اطلاعاتی Products به عنوان خروجی Wrapper مشخص شده است:

```
<output type="database">
      <connectionString>Provider=SQLOLEDB.1;User ID=?; password=?;Data
Source=?;Initial Catalog=Products</connectionString>
</output>
```

۴-۳-۳- معرفی الگوی استخراج

در WDML قواعد استخراج در یک ساختار درختی مشابه با ساختار صفحات ورودی، تعریف می گردد. عملیات نگاشت بین اقلام داده و مفاهیم آنتالوژی دامنه (و یا فیلد های یک جدول)، نیز در همین ساختار درختی قابل تعریف است. در این روش از المان های <element> و <attribute> جهت توصیف تگ های HTML استفاده می شود. توصیف ساختار یک صفحه وب با استفاده از المان <htmlbody> و از بخش بدنه صفحه وب(تگ <body> در ساختار HTML) شروع می شود.

```
<wrapper id="CellPhone">
   ...
```

[77] Connection String

```
    <htmlbody>
        ...
    </htmlbody>
</wrapper>
```

به ازای هر یک از تگ های HTML آن را با استفاده از المان هـای <element> و <attribute> مـدل سازی می کنیم. جهت نمایش ساختار تودرتوی تگ ها مـی تـوانیم از المـان <elements> در درون المـان <element> استفاده می نماییم. همچنین، متن ها در ساختار HTML آزادانه می توانند هر جایی بـین تـگ ها قرار گیرند ولی در WDML متن ها را باید توسط المان <text> نشان دهیم.

```
<htmlbody or elements or record>
    ...
    <element  name=""  cardinality="">
        <attribute  name=""  value="" />
        ...
        <elements>
            ...
        </elements>
        ...
    </element>
    ...
    <text  bind="" pattern="">  a free text  </text>
    ...
</htmlbody or elements or record>
```

المان <element> دارای دو خصوصیت name و cardinality می باشد. خـصوصیت name نـام تـگ در صفحه وب را معرفی کرده و cardinality تعداد تکرار های مجاز را برای تـگ مـورد بحـث نـشان مـی دهد. مقدار کاردینالیتی دارای دو بخش حداقل و حداکثر می باشد. این دو بخش با سمبل ; از هم جدا مـی شوند (در صورت تعیین تنها یک مقدار برای کاردینالیتی، مقادیر حداقل و حداکثر کاردینـالیتی بـه آن عـدد مقدار دهی می شوند). مقدار پیش فرض برای کاردینالیتی مینیمم صفر و مقدار حداکثر برای آن بـی نهایـت می باشد. بنابراین عبارت 1;5 نشان می دهد که کاردینالیتی حداقل یک و حداکثر ۵ می باشـد و عبـارت5; نشان می دهد که کاردینالیتی حداقل صفر و حداکثر ۵ می باشد.

قبل از ادامه بحث، فرض می کنـیم کـه در آدرس http://www.MySite.com/CountryCodes1.htm لیست کد کشور ها، همانند شکل زیر، ارائه شده است.

Country Codes

Iran: *100*
Turkey: *120*
Japan: *130*
UK: *140*

---

end of list.

شکل ۱۹: یک صفحه نمونه وب – لیست کد کشور ها

همچنین فرض کنید که صفحات دیگری مانند CountryCodes2.htm و CountryCodes3.htm و غیره نیز مشابه با صفحه فوق وجود دارند که کد سایر کشور ها را نشان می دهد. اکنون می خواهیم لیست کشور ها را به همراه کد آنها از صفحات فوق جدا کرده و در یک جدول بانک اطلاعاتی مانند شکل زیر نگهداری کنیم:

Country Name	Country Code
Iran	100
Turkey	120
Japan	130
UK	140
...	...

شکل ۲۰: جدول نمونه جهت ذخیره سازی کد کشور ها

بدین منظور، ابتدا به کد HTML صفحه فوق توجه نمایید:

```
01: <html>
02: <head>
03:     <title> Country Codes</title>
04: </head>
05: <body>
06:         <h3> Country Codes</h3>
07:         <B>Iran: </B><i>100</i><br />
08:         <B>Turkey: </B><i>120</i><br />
09:         <B>Japan: </B><i>130</i><br />
10:         <B>UK: </B><i>140</i><br />
11:         <hr />
12:         end of list.
13: </body>
14: </html>
```

همانطور که گفته شد برای ایجاد الگوی استخراج، تنها محتویات درون تگ body (از خط ۶ تا خط ۱۲) را در نظر می گیریم. خط ۶ شامل تگ h3 می باشد که درون آن رشته متنی ثابت Country Codes قرار دارد. این تگ در بین سایر صفحات مشابه، تکراری خواهد بود. کد زیر، تگ فوق را در زبان WDML توصیف می کند.

```
<element name="h3" cardinality="1">
      <elements>
            <skip />
      </elements>
</element>
```

تعبیر کد فوق به این صورت است که در هنگام تجزیه ساختار صفحه وب ورودی، یکبار به المانی با نام h3 برخورد می کنیم که در درون آن متنی با محتوای Country Codes وجود دارد. از آنجا که این المان دارای هیچ نوع اطلاعاتی جهت استخراج نمی باشد. بنابراین می توانیم با استفاده از المان <skip> از محتویات درون آن صرف نظر کنیم. این عمل باعث افزایش قابلیت انعطاف پذیری سیستم خواهد شد. چرا که با تغییر محتویات این بخش از صفحه اجرای Wrapper همچنان موفقیت آمیز خواهد بود.

```
<element name="h3" cardinality="1">
      <elements>
            <text> Country Codes</text>
      </elements>
</element>
```

خط ۷ شامل تگ های B و i می باشد که درون آن یک رشته متنی قرار دارد. می توان این تگ ها را بصورت زیر کد نمود.

```
<element name="B">
      <elements>
            <text>Iran</text>
      </elements>
</element>
<element name="i">
      <elements>
            <text>100</text>
      </elements>
</element>
<element name="br" />
```

کد خطوط ۸ تا ۱۰ نیز معادل با کد فوق خواهد بود. اما از آنجا باید که نام و کد کشور ها به فیلدهای جدول CountryCodes نگاشت شوند، بنابراین کد فوق را بصورت زیر اصلاح می نماییم.

```
<element name="B">
      <elements>
            <text bind="CountryCodes.CountryName" />
      </elements>
</element>
<element name="i">
      <elements>
            <text bind="CountryCodes.CountryCode" />
      </elements>
</element>
<element name="br" />
```

در واقع خطوط ۷ تا ۱۰ رکورد های جدول مورد نظر ما را تشکیل می دهند. هر یک از این رکورد ها، دقیقا دارای کد فوق می باشند. در چنین مواردی با استفاده از المان <record> شروع محدوده رکورد را برای یک جدول و یا یک کلاس از آنتالوژی مقصد، مشخص می کنیم. البته باید به این نکته نیز توجه داشت که رکورد های یک جدول (یا مفهوم آنتالوژی) را نمی توان بصورت تودرتو تعریف نمود.

```
<elements>
      <record bind="">
            ...
      </record>
</elements>
```

بنابراین با استفاده از المان <record> می توان خطوط ۷ تا ۱۰ را بصورت زیر کد نمود.

```
<record bind="CountryCodes">
      <element name="B">
            <elements>
                  <text bind="CountryCodes.CountryName" />
                  <element name="i">
                        <elements>
                              <text bind="CountryCodes.CountryCode" />
                        </elements>
                  </element>
            </elements>
      </element>
</record>
```

در نهایت، از آنجا که باقیمانده خطوط اطلاعاتی جهت استخراج ندارند، می توان از آن صرف نظر کرد. کد نهایی WDML صفحه مورد بحث بصورت زیر می باشد:

```
<?xml version="1.0"?>
<WDML>

<wrapper id="CountryCodes">
    <resources>
          <url>http://www.MySite.com/CountryCodes*.php</url>
    </resources>

    <output type="database">
      <connectionString>Provider=SQLOLEDB.1; ... </connectionString>
    </output>

    <htmlbody>
       <element name="h3" cardinality="1">
          <elements>
             <skip />
          </elements>
       </element>
```

```
        <record bind="CountryCodes">
            <element name="B">
                <elements>
                    <text bind="CountryCodes.CountryName" />
                </elements>
            </element>
            <element name="i">
                <elements>
                    <text bind="CountryCodes.CountryCode" />
                </elements>
            </element>
            <element name="br" />
        </record>
        <skip />
    </htmlbody>
</wrapper>

</WDML>
```

همانطور که دیده می شود، ایجاد WDML بصورت دستی نیاز به آگاهی کامل از ساختار فایل HTML دارد. لذا یکی از فعالیت های مهم در این پروژه آن است که فرایند تولید WDML را اتوماتیک نماییم.

قبل از ادامه بحث لازم است تا یک نکته کوچک ولی خیلی مهم در ارتباط با مثال فوق ذکر گردد. در مثال فوق، دیده می شود متنی که در داخل المان B قرار دارد شامل نام کشور و بدنبال آن سمبل : می باشد. این مساله بیانگر این واقعیات است که در هنگام طراحی صفحات وب ممکن است برخی "مقادیر ثابت" به اقلام اطلاعاتی اضافه گردد و یا حتی ممکن است این اقلام اطلاعاتی بطریقی با هم ترکیب شوند. به عنوان مثال معمولا ابعاد یک محصول بصورت؛ طول x عرض x ارتفاع، نشان داده می شود.

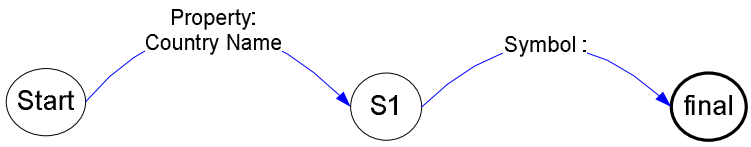
یکی از وظایف روتین استخراج آن است که از رشته های فوق الذکر اقلام اطلاعاتی اولیه را بازیابی نماید. پیشنهاد ما استفاده از آتاماتای غیر قطعی[۷۸] (NFA) می باشد. این آتاماتا در هنگام تعریف آنتالوژی استخراج، معرفی می گردد[۷۹] و نیازی به معرفی مجدد آن در فایل WDML نمی باشد و تنها کافی است تا از فایل WDML (با استفاده از خصوصیت ontology) به فایل آنتالوژی استخراج، ارجاع داده و سپس در صورت نیاز خصوصیت pattern در المان text را با نام الگوی مورد نظر در آنتالوژی استخراج مقدار دهی کنیم.

---

[78] Non-deterministic Finite state Automate

[79] توضیحات بیشتر در ارتباط با تعریف الگو ها با استفاده از NFA را در بخش های بعدی هنگام معرفی آنتالوژی استخراج، ارائه خواهیم کرد

در این مثال، برای تعریف الگو از NFA شکل زیر استفاده می گردد:



شکل ۲۱: یک NFA نمونه، جهت استخراج نام کشور ها

جزئیات مربوط به تعریف الگوها با استفاده از NFA را در بخش معرفی آنتالوژی استخراج بیان خواهیم کرد. در اینجا فقط جهت آشنایی اولیه، بخشی از آنتالوژی استخراج که NFA شکل فوق را پیاده سازی می کند ارائه می کنیم:

```
<pattern id="CountryName">
	<NFA>
		<node id="start">
			<edge propertyvalue="CountryName" target="S1" />
		</node>
		<node id="S1">
			<edge symbol=";" />
		</node>
	</NFA>
</pattern>
```

الگوی فوق در فایل آنتالوژی استخراج ایجاد خواهد شد. لذا تنها تغییری که در فایل WDML لازم است ایجاد شود آن است که اولا در المان WDML، به فایل آنتالوژی استخراج، ارجاع دهیم:

```
<WDML ontology="MyOntology.owl">
```

و ثانیا، هنگام نگاشت المان text به فیلد جدول مقصد (و یا یک مفهوم در آنتالوژی مقصد)، باید الگوی استخراج را هم مشخص نماییم:

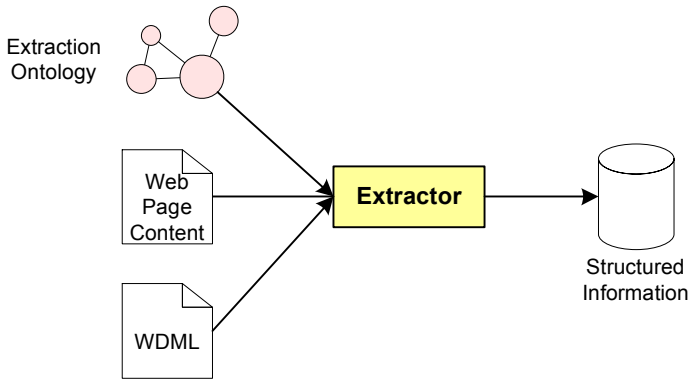
```
<text bind="CountryCodes.CountryName" pattern="CountryName" />
```

بدین ترتیب، ماژول استخراج، ابتدا آنتالوژی استخراج را خوانده سپس در فایل WDML هنگامیکه به الگوی Company Name را می یابد، رشته ورودی (به عنوان مثال :Iran) را به واحد پردازشگر NFA می دهد. واحد فوق، ابتدا رشته ورودی را در سطح کلمات و سمبل ها تقسیم بندی کرده و مجموعه {"Iran", ":"} را تشکیل می دهد. سپس با پردازش NFA، عبارت Iran را برای Country Name می یابد و نتیجه را به واحد استخراج بر می گرداند.

در بخش بعد به تشریح روند استخراج با استفاده از فایل های WDML می پردازیم.

۴-۴- استخراج با استفاده از WDML

در سیستم OntoByOnto، ماژول Extractor سه ورودی؛ فایل WDML، محتویات صفحه وب و آنتالوژی استخراج را دریافت کرده و با استفاده از قواعد استخراج تعریف شده در WDML، اطلاعات مورد نظر را استخراج کرده و تحت قالب مشخصی، در خروجی می نویسد.



شکل ۲۲: ماژول Extractor

الگوی استخراج در WDML بصورت درختی تعریف می شوند لذا ماژول Extractor ابتدا ساختار درختی صفحه وب ورودی را تشکیل می دهد. این ساختار درختی بر اساس درخت تجزیه HTML صورت می گیرد. سپس این ماژول بر اساس الگوی استخراج WDML، ساختار درختی فوق را می پیماید و هرگاه در این پیمایش، در الگوی استخراج به المان text برخورد نماید، در صورت نگاشت آن به یک فیلد از جدول مقصد (یا یک مفهوم از آنتالوژی مقصد)، مقدار معادل آن را از ساختار درختی بازیابی کرده و در مجموعه اطلاعات استخراج شده اضافه می کند. در ادامه، ابتدا الگوریتم استخراج را تشریح کرده و سپس به اجرای یک مثال می پردازیم.

۴-۴-۱- الگوریتم استخراج

الگوریتم استخراج که در اینجا تشریح خواهد شد، یک الگوریتم بازگشتی می باشد. در این الگوریتم، ابتدا براساس صفحه وب ورودی یک درخت (درخت تجزیه HTML) ایجاد شده و سپس این درخت ورودی، جهت استخراج اقلام اطلاعاتی موجود در آن به یک تابع بازگشتی ارسال می گردد. در صورت موفقیت، اطلاعات استخراج شده در یک مجموعه از زوج مرتب <bind, value>، ذخیره می گردد که می توان آن را با فرمت مشخص شده در WDML به خروجی منتقل نمود.

```
public class ResultValue
{
    public string Bind;
    public string Value;
}
```

ساخت درخت تجزیه، خارج از موضوع بحث ما می باشد و در صورت نیاز می توانید به سایت CodeProject [80] مراجعه نمایید. در اینجا فقط به معرفی برخی کلاس های پایه که در ساخت درخت تجزیه مورد استفاده قرار می گیرد، می پردازیم:

```
abstract public class HtmlNode
{
    ...
}
public class HtmlElement : HtmlNode
{
    public string Name;
    public HtmlElementCollection Nodes;
    public Attributes Attributes;
    ...
}
public class HtmlText : HtmlNode
{
    public string Text
    ...
}
```

تمام گره های درخت تجزیه Html یا از نوع HtmlElemen و یا از نوع HtmlText می باشند. HtmlElemen برای نمایش تگ های Html و HtmlText برای نمایش متن های آن استفاده می شود. درخت ایجاد شده در با ساختار بنام ParsTree نشان داده می شود. بخش آغازین الگوریتم به شکل زیر است:

```
01  public bool Extract(WDML W, string WebPageContent)
02  {
03      ResultValues Result = new ResultValues();
04      ParsTree P;
05      P = Build_ParsTree(WebPageContent);
06      if(!DoExtract(W.htmlBody, P.body, 0, Result))
07          return false;
08      W.WriteOutput(Result);
09  }
```

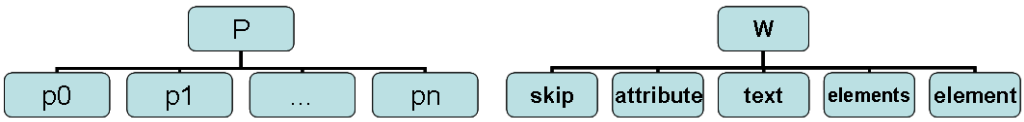
لیست ۱: الگوریتم استخراج – قسمت اول

تابع فوق، پس از تعاریف اولیه در خط ۵، دستور ساخت درخت تجزیه را اجرا می کند. عملیات اصلی استخراج توسط تابع DoExtract (خط ۶) انجام می شود. در صورت موفقیت، اطلاعات استخراج شده توسط پارامتر W (یک نمونه از کلاس WDML) در خروجی نوشته می شود.

---

[80] http://www.codeproject.com/info/search.asp

تابع DoExtract، چهار پارامتر ورودی؛ گره جاری در الگوی استخراج (W)، گـره جـاری در درخـت ورودی (P)، محل شروع پردازش در بین گره هـای فرزنـد P (pIndex) و مجموعـه ای جهـت نگهـداری اطلاعات استخراج شده را دریافت می کند. وظیفه اصلی این تابع، پیمـایش فرزنـدان گـره P بـا اسـتفاده از الگوی تعریف شده در W می باشد. شکل زیر انواع گره هایی که فرزندان گره W می توانند داشته باشند را در کنار فرزندان گره P، نشان می دهد.



شکل ۲۳ : مقایسه فرزندان P و W

با استفاده از گره های فرزند W، که در اینجا آن را wi می نامیم، صـحت گـره فرزنـد P [۸۱] را (کـه در اینجا آن را pi می نامیم) سنجیده و هر یک از حالت های زیر را بررسی می کنیم:

- اگر گره wi، از نوع skip باشد، از بررسی الباقی گره های فرزند P صـرف نظـر مـی کـرده و مقدار pIndex را به n (آخرین فرزند P) مقدار دهی می کنیم.
- اگر گره wi، از نوع attribute باشد، مقدار صفت مشخص شده باید با صفت مربوطه در گـره pi برابر باشد.
- اگر گره wi، از نوع text باشد، گره pi باید از نوع HtmlText بوده و مقدار متنـی pi بایـد بـا مقدار متنی wi (غیر تهی) برابر باشد. همچنین در صورتیکه برای گره wi صفت bind تعریف شده باشد، آنگاه مقدار متنی pi و مقدار صفت bind به مجموعه جواب اضافه می گردد.
- اگر گره wi، از نوع elements باشد، با توجه به اینکه نوع گره حاوی یک زیر الگو می باشـد، باید این بار تابع DoExtract را برای الگوی wi و گره P با شـروع از محـل جـاری pIndex، بصورت بازگشتی فراخوانی کنیم.
- اگر گره wi، از نوع elements باشد، صفت نام (name) این گـره بایـد بـا نـام گـره pi برابـر باشد. در اینصورت باید با توجه به حداقل و حداکثر کاردینالیتی، مراحل زیر انجام گیرد:
  - تابع DoExtract جهت اطمینان از تطابق کامل الگـوی wi بـا گـره pi (بـا شـروع از اولین گره فرزند pi) و استخراج اطلاعات موجود در فرزندان pi، فراخوانی می شود.
  - در P، به گره فرزند بعدی می رویم (pIndex را یک واحد افزایش می دهیم)
  - pi بعدی را محاسبه می کنیم. (pi = P[pIndex])

[81] با شروع از pIndex اُمین فرزند

در صورتی تابع DoExtract با موفقیت تمام می شود که اول،ا در صورت بروز هریک از موارد فـوق، نتیجه مقایسه ها درست ارزیابی گردد و ثانیا، در هنگام خروج از تابع، تمام گره های فرزند P بررسی شـده باشند. به عبارتی pIndex برابر n (تعداد فرزندان P) باشد.

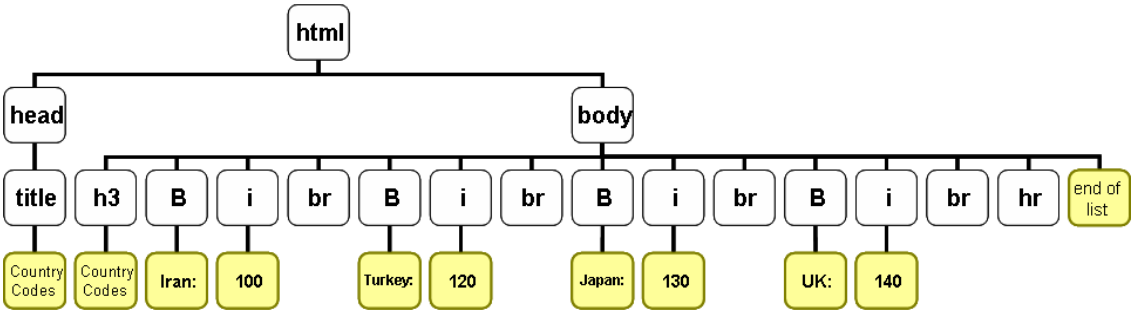
```
01 private bool DoExtract(ElementCollection W, ElementCollection P,
                          ref int StartIndex, ResultValues Result)
02 {
03     int pIndex = StartIndex;
04     Element pi = P.Count > pIndex ? P[pIndex] : null;
05     ResultValues tempResult = new ResultValues();
06     foreach (Element wi in W)
07     {
08         switch (wi.Name)
09         {
10             case "skip":
11                 pIndex = P.Count;
12                 return true;
13             case "attribute":
14                 string atrName = wi.Attributes["name"].Text;
15                 string atrValue = wi.Attributes["Value"].Text;
16                 if (pi.Attributes[atrName].Text != atrValue)
17                     return false;
18                 break;
19             case "Text":
20                 if (pi == null || !(pi is HtmlText)) return false;
21                 if(wi.Text!=null && wi.Text!=pi.Text) return false;
22                 string bind = wi.Attributes["bind"].Value;
23                 if (bind != null) tempResult.Add(bind, pi.Text);
24                 break;
25             case "record":
26                 while (DoExtract(wi, P, ref pIndex, tempResult))
27                     ;
28                 break;
29             case "Elements":
30                 if (!DoExtract(wi, P, ref pIndex, tempResult))
31                     return false;
32                 break;
33             case "Element":
34                 if(pi == null || wi.Attributes["name"].Text != pi.Name)
35                     return false;
36                 for (int i = 0; i < wi.MaxCardinality; ++i)
37                 {
38                     if (!DoExtract(wi, pi, 0, tempResult))
39                         if (i < wi.MinCardinality) return false;
40                         else break;
41                     pIndex += 1;
42                     pi = P.Count > pIndex ? P[pIndex] : null;
43                 }
34                 break;
45         }
46     }
47     if (pIndex < P.Count) return false;
48     Result.Add(tempResult);
49 }
```

لیست ۲: الگوریتم استخراج – قسمت دوم

لیست فوق، کد تابع DoExtract() را به زبان C# نشان می دهد. برای فهم بیشتر این الگوریتم در ادامه مثال لیست کشور ها که در بخش قبل مطرح شده است را ادامه می دهیم. این بار سعی خواهیم کرد تا با استفاده از این الگوریتم و کد WDML ایجاد شده در بخش قبل، لیست کشور ها و کد آنها را استخراج نماییم.

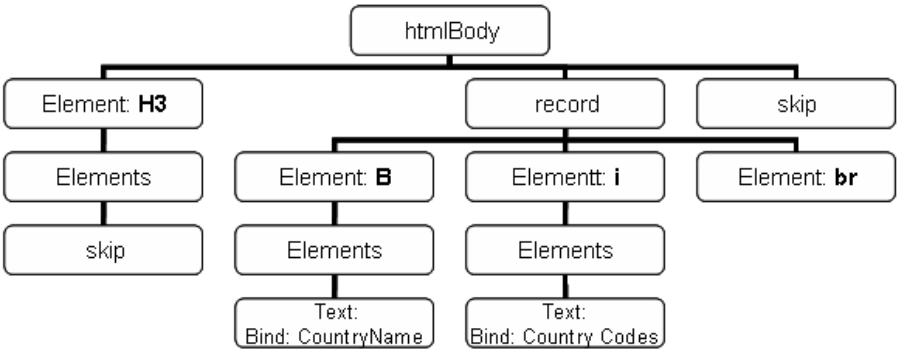
۴-۴-۲- اجرای یک مثال

همانطور که گفته شد، جهت استخراج اطلاعات باید ابتدا ساختار درختی صفحه وب ورودی را ایجاد نمود. شکل زیر این ساختار را به ازای صفحه وب لیست کشور ها (مطرح شده در بخش قبل) را نشان می دهد:



شکل ۲۴: ساختار درختی صفحه لیست کشور ها

همانطور که در الگوریتم نیز آورده شد، تنها فرزندان گره body در این الگوریتم شرکت خواهند داشت. بنابراین از سایر بخش های درخت فوق صرف نظر می شود. از طرفی دیگر، فایل ورودی WDML نیز خوانده شده و الگوی استخراج موجود در آن در یک ساختار درختی نگهداری می گردد. شکل زیر ساختار درختی فایل WDML در مثال قبل را نشان می دهد.



شکل ۲۵: الگوی استخراج در فایل WDML

الگوریتم استخراج، با ارسال الگوی استخراج فوق و گره body از درخت ورودی به تابع DoExtract آغاز می گردد. این الگوریتم گره های فرزند htmlBody را خوانده و با توجه به آن ها گره های فرزند body از درخت ورودی را پردازش می کند.

سیستم ابتدا در سطح صفر بازگشتی قرار دارد. در این سطح هنوز هیچ فراخوانی بازگشتی صورت نگرفته است. سیستم ابتدا گره [Element: h3] را در wi می خواند. لذا انتظار دارد، المانی با نام h3 در محل جاری از درخت ورودی بیابد (خط ۳۴ الگوریتم استخراج) از طرفی اولین فرزند از گره body، المان h3 می باشد (مقدار pi)، سیستم، با توجه به آنکه گره جاری از نوع المان می باشد، بنابراین باید الگوی موجود در درون این المان با فرزندان المان h3 از درخت ورودی مطابقت داشته باشد. لذا سیستم با ارسال پارامتر های [Element: h3] و h3 به تابع DoExtract، آن را بصورت بازگشتی فراخوانی می کند (خط ۳۸).

در این اجرای بازگشتی (سطح اول بازگشتی)، اولین فرزند [Element: h3]، گره Elements می باشد. از آنجا که این گره یک گره مجازی می باشد، در خط ۳۰، مجددا تابع DoExtract با ارسال گره Elements و h3 به عنوان پارامتر های ورودی، فراخوانی می گردد. در این اجرای مجدد (سطح دوم بازگشتی)، wi (گره فرزند Elements) برابر المان skip می باشد. لذا در خط ۱۱، از سایر گره های فرزند، صرف نظر شده و اندیس pi به آخرین گره فرزند h3 (از درخت ورودی) اشاره گر شده و کنترل به سطح اول بازگشتی و از آنجا به سطح صفر بازگشتی منتقل می شود.

بعد از بازگشت به سطح صفر بازگشتی، در اجرای دوم حلقه (خط ۶) مقدار wi برابر با record می گردد. در خط ۲۶ تا زمانیکه استخراج رکورد ها امکان پذیر است، تابع DoExtract بصورت بازگشتی فراخوانی می گردد. در اولین دور این حلقه، تابع فوق با ورودی های record برای پارامتر W و body برای پارامتر P و اندیس شروع StartIndex برابر با ۱ (اشاره گره به المان B) فراخوانی می گردد.

در سطح اول بازگشتی، مقدار wi ابتدا برابر با [Element: B] و pi برابر با المان B می گردد. در خط ۳۸ جهت انطباق فرزندان wi و pi گره فوق مجددا بصورت بازگشتی فراخوانی می گردد. در سطح دوم بازگشتی، wi برابر با elements خواهد شد در نتیجه جهت انطباق الگوی تعریف شده در آن با گره های فرزند المان B، مجددا فراخوانی بازگشتی صورت می گیرد.

در سطح سوم بازگشتی، wi برابر با المان Text و pi برابر با مقدار "Iran" خواهد بود. در خط ۲۰ ابتدا تست می شود که pi از نوع المان متنی باشد. سپس از آنجا که wi دارای صفت bind می باشد، زوج مرتب زیر به مجموعه جواب اضافه می گردد (خط ۲۳).

```
<Bind="CountryCodes.CountryName", Value="Iran">
```

در ادامه، کنترل به سطح دوم و سپس به سطح اول بازگشتی عودت داده مـی شـود. در اجـرای بعـدی حلقه (خط ۶)، wi برابر با [Element: i] می گردد. با اجرای عملیات فـوق سـایر اقـلام اطلاعـاتی نیـز بـه مجموعه جواب اضافه می گردند.

کلیه اطلاعات استخراج شده، بصورت زوج مرتب <bind, value> در مجموعه Result نگهداری مـی گردد. بعد از اجرای موفقیت آمیز این الگوریتم، می توان مجموعه فوق را با توجه به قالب تعیـین شـده در فایل WDML در خروجی نوشت.

۴-۵- خلاصه مطالب و نتیجه گیری

در این فصل، ابتدا به معرفی ایده اصلی خـود پـرداختیم و روش جدیـدی را جهـت اسـتخراج اقـلام اطلاعاتی مطرحی کردیم. روش پیشنهادی ما ترکیبی از روش های مبتنی بر آنتالوژی و مبتنی بر استنتاج می باشد. ما از آنتالوژی استخراج جهت تشخیص اتوماتیک الگو ها و اطمینان از صـحت داده هـای اسـتخراج شده استفاده می کنیم. در نهایت الگو های استخراج بدست آمده در قالـب فایـل هـای WDML در مخـزن خروجی نوشته می شوند.

فایل های WDML از یک طرف با ارائه الگو های استخراج در قالب یک ساختار درختی، محل دقیـق وقوع اقلام اطلاعاتی را در صفحه وب مشخص می سـازند و در نتیجـه باعـث افـزایش سـرعت در زمـان اجرای فرایند استخراج می گردند. از طرفی دیگر این فایل ها با ارجاع به آنتالوژی استخراج مربوطه تعیـین می نمایند که هر یک از مقادیر اطلاعاتی باید چه ویژگی هایی داشته باشند و بـدین ترتیـب صـحت اقـلام اطلاعاتی را چک می کنند.

ایده استفاده از زبان WDML می تواند در استاندارد کردن فرایند ایجاد Wrapper ها بسیار موثر باشد. در این روش، بدلیل استفاده از ساختار درختی صفحات و امکان گذر از گره هایی که فاقـد اطلاعـات مـی باشند، Wrapper هایی خواهیم داشت که نسبت به تغییرات جزئی در صفحات وب بـسیار انعطـاف پـذیر بوده و به راحتی بین سیستم های مختلف قابل نقل و انتقال می باشند. از طرفی دیگر، سـاختار فایـل هـای WDML بسیار ساده و گویا بوده و در استخراج سریع و بی عیـب و نقـص اطلاعـات کمـک شـایانی مـی نماید.

ما ایده خود را در قالب سیستمی بنام OntoByOnto مطرح نموده ایم. این سیستم دارای سه مولفه اصلی؛ خزشگر، ماژول استخراج کننده (Extractor) و ماژول تولید کننده خودکار WDML می باشد. در این سیستم، برای پردازش فایل های WDML از ماژول Extractor استفاده می گردد. این ماژول می تواند در هر سیستم و یا پلتفورمی مطابق با الگوریتم تشریح شده در این فصل به راحتی نوشته شود.

باید توجه داشت که علیرغم مزایای ناشی از توصیف Wrapper ها با استفاده از WDML، هنوز ایجاد دستی فایل های WDML سخت و وقت گیر بوده و نیازمند به آگاهی قبلی از ساختار صفحات وب می باشد. لذا در سیستم پیشنهادی خود، ماژولی جهت تولید اتوماتیک فایل های WDML ارائه در نظر گرفته شده است. ایده اصلی، استفاده از آنتالوژی استخراج و در نظر گرفتن ساختار نمایش درختی صفحات وب جهت اتوماتیک فایل های واسط WDML می باشد. در فصل بعد به تشریح کامل الگوریتم مربوطه خواهیم پرداخت.

۵- تولید اتوماتیک فایل های WDML

همانطور که در بخش قبل دیده شد، ایجاد دستی فایل های WDML بسیار خسته کننده بوده و نیازمند به داشتن آگاهی از ساختار صفحات و محل قرار گیری اقلام اطلاعاتی می باشد. از این رو در این بخش سعی خواهیم کرد تا روشی را جهت ایجاد اتوماتیک Wrapper ها از صفحات وب ایجاد نماییم.

۵-۱- مقدمه

روش پیشنهادی ما آن است که ابتدا با استفاده از یک آنتالوژی استخراج، مقادیر داده ای خام موجود در صفحه وب را شناسایی کرده و خصوصیات[۸۲] مرتبط با هریک از آنها را بیابیم. البته استفاده از آنتالوژی استخراج، ممکن است به ازای یک مقدار داده ای مشخص، دو یا چند خصوصیات مرتبط را بیابد. لذا در قدم بعدی سعی خواهیم کرد تا با در نظرگرفتن محدودیت ها و شرایط زمینه ای موجود، مناسب ترین خصوصیت را انتخاب نماییم. یکی از این شرایط زمینه ای، ساختار درختی صفحه وب می باشد به عبارتی می توانیم با در نظر گرفتن اصل همجواری داده و برچسب[۸۳]، مناسب ترین صفت را انتخاب نماییم. در این بخش، علاوه بر موارد فوق، روش های متنوع دیگری را برای شناسایی اقلام داده ای موجود در صفحه وب پیشنهاد خواهیم نمود.

در قدم سوم، جهت شناسایی اتوماتیک Warpper ها، محدوده رکورد و الگو های موجود در صفحه را جستجو خواهیم نمود. اما قبل از تشریح روش ایجاد اتوماتیک واسط های WDML، لازم است تا ابتدا با آنتالوژی استخراج که نقش مهمی در فهم صحیح ماشین از مقادیر مربوط به مفاهیم موجود در دامنه مورد بحث دارد، آشنا شویم.

---

[82] Properties
[83] Label

۵-۲- آنتالوژی استخراج[۸۴]

آنتالوژی استخراج، یک نمونه از مدل مفهومی یا مدل سیستم شیء گرایی[۸۵] (OSM) در دامنه مورد بحث، می باشد [Emb98]. برخلاف آنتالوژی دامنه که از آن جهت توصیف مفاهیم و نمونه های موجود در دامنه و نیز ارتباط بین آنها بکار می رود، این آنتالوژی جهت توصیف مقادیر داده ای استفاده می شود. به عبارتی زمانیکه لازم باشد تشخیص دهیم یک مقدار داده ای مرتبط با کدامیک از خصوصیات یا مفاهیم موجود در آنتالوژی دامنه می باشد، نیاز داریم تا در ارتباط با ویژگی های مقداری هر یک از خصوصیات تعریف شده در دامنه اطلاعات کاملی داشته باشیم.

برخی از این اطلاعات (مانند نوع داده ای و محدوده مقادیر مجاز و غیره) در آنتالوژی دامنه خواهد آمد. اما برخی از آنها ارتباطی با مفاهیم دامنه ندارند و از آنها تنها در شناسایی اقلام اطلاعاتی استفاده می شود. به عنوان مثال، الگوی مورد استفاده در نمایش شماره تلفن، تاریخ، ساعت و غیره. ما این اطلاعات تکمیلی را در یک آنتالوژی استخراج که توسط مهندس دامنه بصورت دستی ایجاد می شود، ثبت و نگهداری می کنیم. در این شیوه، آنتالوژی دامنه جهت استنتاج و استدلال از اطلاعات و مفاهیم موجود و پاسخ گویی به سوالات کاربران استفاده می شود در حالیکه آنتالوژی استخراج جهت شناسایی داده های خام موجود در صفحات وب استفاده می شود.

هر چند که آنتالوژی دامنه و آنتالوژی استخراج از هم جدا بوده و در دو محیط جداگانه طراحی و پیاده سازی شده و حتی دارای زبان های متفاوتی نیز می باشند ولی باید توجه داشت که آنتالوژی استخراج همواره به خصوصیات و مفاهیم موجود در آنتالوژی دامنه اشاره داشته و به آن وابسته است. به عبارتی در صورت تغییر آنتالوژی دامنه (به عنوان مثال تغییر نام یک خصوصیت) باید آنتالوژی استخراج را بررسی کرده و تغییرات لازم را در آن اعمال نمود. آنتالوژی دامنه خارج از حوزه بحث این تحقیق می باشد و به آن نخواهیم پرداخت ولی ذکر این نکته نیز لازم است که آنتالوژی استخراج را می توان در آنتالوژی دامنه ادغام نمود. این ادغام سبب می شود از دوگانگی در تعریف مفاهیم آنتالوژی جلوگیری شود و مدیریت آنتالوژی راحت تر گردد.

اما از طرفی، ادغام آنتالوژی دامنه و آنتالوژی استخراج سبب می شود حجم اطلاعات موجود در آنتالوژی دامنه که معمولا جهت استنتاج در پاسخگویی به پرس و جوی کاربران استفاده می شود، افزایش

---

[84] Extraction Ontology
[85] Object System Model

یافته و فرایند پردازش را کند نماید. این درحالی است که اطلاعات مربوط به آنتالوژی استخراج تنها یکبار و آن هم هنگام استخراج اطلاعات مورد استفاده قرار می گیرد ولی آنتالوژی دامنه مکررا در پاسخ به پرس و جو های کاربر مورد استفاده قرار می گیرد. علاوه بر موارد فوق می توان به افزایش مصرف حافظه، پیچیدگی در تعریف آنتالوژی دامنه و محدودیت در انتخاب زبان جهت توصیف آنتالوژی استخراج، به عنوان سایر معایب ادغام آنتالوژی دامنه و آنتالوژی استخراج اشاره کرد. با توجه به مطالب فوق، بهتر است آنتالوژی دامنه و آنتالوژی استخراج جداگانه تعریف شده ولی همواره به ارتباط میان آن دو نیز توجه داشت.

در سیستم پیشنهادی ما، آنتالوژی استخراج، بصورت دستی و توسط مهندس دامنه ایجاد می شود. این آنتالوژی یکی از ارکان سیستم OntoByOnto را تشکیل می دهد. بگونه ای که هرچه این آنتالوژی دقیق تر تعریف شده باشد، عملیات استخراج نیز دقیق تر و کاملتر خواهد بود. در ادامه به معرفی این آنتالوژی و اجزای مختلف آن می پردازیم.

۵-۲-۱- انتخاب زبان

اولین قدم در ساخت آنتالوژی استخراج، انتخاب یک زبان مناسب جهت توصیف مفاهیم و خصوصیات دامنه مورد بحث می باشد. از میان زبان های آنتالوژی موجود ، زبان OWL دارای قابلیت های فراوانی جهت توصیف مفاهیم و روابط موجود در آنتالوژی را دارا است. بخشی از قابلیت های این زبان مربوط به توصیفات مفاهیم و خصوصیات می باشد. برخی دیگر نیز جهت استنتاج از قواعد و حقایق [۸۶] موجود کاربرد دارد.

با توجه به ضرورت پیاده سازی این بخش از پروژه و پیچیدگی پارسر زبان OWL، تصمیم گرفته شد که در این تحقیق، از یک زبان ابتدایی که شامل سازه های از پیش تعریف شده می باشد، استفاده گردد. ما این زبان را ابتدا با قابلیت تعریف کلاس ها و خصوصیات آنها آغاز کردیم و سپس رفته رفته با توجه به پیشرفت پروژه و نیاز به تعاریف جدید در آنتالوژی استخراج، آن را تکمیل نمودیم.

ما در اینجا قصد نداریم تا این زبان را توصیف نماییم بلکه هدف تنها معرفی قابلیت هایی است که یک آنتالوژی استخراج باید داشته باشد. آنتالوژی استخراج مورد استفاده در این پروژه دارای چهار بخش؛

---

[86] Fact

کلاس ها، خصوصیات، الگو ها و فهرست واژگان می باشد. ما در اینجا با توجه به دامنه گوشی های تلفن همراه، هنگام تعریف هر یک از بخش های این آنتالوژی مثال هایی از نحوه پیاده سازی آن مطرح خواهیم کرد.

۵-۲-۲- تعریف کلاس ها

کلاس ها نشان دهنده مفاهیم موجود در دامنه مورد بحث بوده و می توانند مجازی یا غیر مجازی باشند. تا آنجا که به آنتالوژی استخراج مرتبط می شود، مجازی یا غیر مجازی بودن یک کلاس به معنای وجود یا عدم وجود نمونه های آن در بین صفحات وب می باشد. به عنوان مثال در تکه آنتالوژی زیر کلاس Product از نوع مجازی می باشد. به این معنی که سیستم در میان صفحات وب، بدنبال نمونه های این کلاس نخواهد بود. در مقابل از آنجا که سیستم، نمونه های موجود گوشی تلفن همراه را جمع آوری می نماید، خصوصیت abstract برای کلاس CellPhone به مقدار false تنظیم شده است.

```
<class id="Product" abstract="true">
  <instancename property="ProductName"/>
  <properties>
    <property resource="Brand" maxcardinality="1"/>
    <property resource="ProductName" cardinality="1"/>
  </properties>
</class>


<class id="CellPhone" abstract="false">
  <subclassof resource="Product"/>
  <labels>
    <label>موبایل</label>
    <label>گوشي</label>
    <label>mobile</label>
    <label>cell phone</label>
  </labels>
  <properties>
    <property resource="NetworkType" cardinality="1" />
    <property resource="Announced" maxcardinality="1" />
    <property resource="OS" maxcardinality="1"/>
    <property resource="Size" />
    <property resource="Weight" />
     ...

  </properties>
</class>
```

در هر کلاس امکان تعریف بخش های زیر وجود دارد:

- **برچسب**: قابلیت تعریف برچسب برای یک کلاس در حال حاضر کاربرد عملی ندارد و تنها جهت استفاده های آتی در نظر گرفته شده است.
- **خصوصیات**: خصوصیات بصورت جداگانه تعریف خواهند شد. در هنگام تعریف یک کلاس تنها مشخص می شود که هر کلاس شامل چه خصوصیاتی می باشد. به ازای هر خصوصیت باید کاردینالیتی حداقل و کاردینالیتی حداکثر آن را هم مشخص نمود. در صورت عدم تعیین کاردینالیتی، مقدار حداقل آن صفر و مقدار حداکثر آن بینهایت می باشد. به عنوان مثال همانطور که در تکه آنتالوژی فوق آمده است، کلاس CellPhone دارای خصوصیاتی مانند نوع شبکه، سیستم عامل، وزن و غیره دارد.
- **نام کلاس نمونه** : نمونه هایی که توسط این سیستم یافت خواهند شد همگی از نوع موجودیت های نام دار[۸۷] می باشند. هر یک از موجودیت های نام دار دارای یک نام منحصر به فرد بوده و هیچ دو موجودیتی از آن کلاس دارای نام مشابه نخواهند بود. به عنوان مثال، در دامنه گوشی های تلفن همراه هر محصول باید دارای یک نام منحصر به فرد باشد. مانند N70، K750 و W810 و غیره.
- **خاصیت ارث بری**: یک کلاس می تواند تمام خصوصیات و محدودیت های سایر کلاس ها را به ارث ببرد. ارث بری یکی از قابلیت های مهم در هنگام تعریف آنتالوژی دامنه می باشد ولی در آنتالوژی استخراج، با توجه به شرایط می توان آنتالوژی پدر را در آنتالوژی فرزند ادغام نمود.

باید توجه داشت که یک کلاس نمی تواند به تنهایی دارای یک مقدار باشد بلکه شامل خصوصیاتی است که هر یک از آنها به تنهایی نمایشگر یک مقدار مشخص در صفحه وب می باشند.

۵-۲-۳- تعریف خصوصیات

خصوصیت (یا Property) بصورت جداگانه و خارج از بدنه کلاس تعریف می شود. خصوصیت ها مهمترین بخش آنتالوژی استخراج بوده و همواره معادل با یک مقدار مشخص در صفحه وب می باشند. جهت توصیف خصوصیت ها باید از نحوه نمایش اطلاعات در صفحات وب اطلاعات کامل در اختیار داشت. لذا بهتر است در این مورد از یک مهندس دامنه مشاوره گرفت. در تکه آنتالوژی زیر تعریف برخی خصوصیت های گوشی تلفن همرا نشان داده شده است:

---

[87] Named Entity

```
...

<property id="NetworkType" datatype="">
    <labels>
        <label>network</label>
        <label>band</label>
        <label>فرکانس باند</label>
        <label>باند</label>
        <label>شبکه</label>
    </labels>
    <disjointwith>
        <property resource="" />
    </disjointwith>
    <pattern resource="NetworkType"/>
</property>

<property id="DisplayType" datatype="">
    <labels>
        <label>Type</label>
        <label>Display Type</label>
        <label>نوع</label>
        <label>نمایش صفحه نوع</label>
    </labels>
    <regularexpressions>
        <pattern>.*(CSTN|STN|TFT|TFD|OLED).*</pattern>
    </regularexpressions>
    <disjointwith>
        <property resource="" />
    </disjointwith>
</property>

<property id="Size" datatype="SizeClass">
    <labels>
        <label>Phone Size</label>
        <label>Size</label>
        <label>ابعاد</label>
        <label>سایز</label>
    </labels>
    <pattern resource="Size"/>
</property>

...
```

این دو خصوصیت فوق تقریبا شامل تمام ویژگی هایی هستند که تا به حال در سیستم OntoByOnto تعریف شده است. این ویژگی ها عبارتند از:

- **برچسب**: برچسب ها عناوینی هستند که معمولا برای توصیف مقادیر موجود در یک صفحه وب بکار برده می شود. این مقادیر به نوع زبان طبیعی (انگلیسی، فارسی، عربی و غیره) جهت توصیف اطلاعات وابسته می باشد. در این سیستم جهت تشخیص برچسب ها، مقادیر داده ای موجود در صفحه وب را با برچسب های تعریف شده به ازای هر خصوصیت مقایسه می کنیم. این ویژگی، تنها بخشی است که جهت تشخیص برچسب ها در نظر گرفته شده است و شاید نسبت به سایر بخش ها کمترین میزان تحقیقات را به خود اختصاص داده

است. به هر حال در صورت نیاز، جهت تشخیص بهتر برچسب ها، می تـوان از روش هـای مقایسه نسبی رشته ها استفاده نمود.

- **نوع داده** : یکی از اولین قدم ها در توصیف خصوصیت، تعیـین یـک نـوع داده ی مـشخص برای آن می باشد. در ایـن سیـستم سـه نـوع داده ای اولیـه، شـامل integer، float و string تعریف شده است. البته طراحی آنتالوژی می تواند بـا ایجـاد یـک کـلاس مجـازی، آن را بـه عنوان نوع داده ای پیچید تعریف نماید. به عنوان مثال نوع داده ای Size در دامنه گوشی های تلفن همراه، یک کلاس مجازی است که شـامل ترکیـب سـه خـصوصیت Width, Height, Length می باشد. این کلاس مجـازی را مـی تـوان بـه عنـوان نـوع داده ای پیچیـده جهـت توصیف خصوصیت Size بکار برد.
- **داد های گسسته**: گاهی اوقات مجموعه مقادیر ممکن برای دو یا چند خصوصیت بطور کامل از هم جدا می باشد. به عبارتی اگر مجموعه مقادیر خصوصیات P1 و P2 جدای از هم باشـد و در صورتیکه بدانیم یک مقدار مشخص مربوط به خصوصیت P1 می باشد می توانیم نتیجه بگیریم که آن مقدار هیچگاه نمی تواند P2 باشد. به عنوان مثال مجموعه مقادیر ممکـن بـرای نوع شبکه[88] (مانندGSM 1800, GSM 900, GSM 800 و غیره)، در سایر خصوصیات دامنه گوش های تلفن همراه وجود ندارد. این موضوع می تواند باعث افزایش سـرعت و دقـت در هنگام جستجوی خصوصیات مرتبط با یک داده خام موجود در صحفه وب گردد.
- **عبارت باقاعده**[89]: این واقعیت که برخی مقادیر خصوصیات دارای نظم بخصوصی هستند نیز می تواند در یافتن خصوصیت مرتبط بسیار موثر باشـد. بـه عنـوان مثـال مقـادیری کـه بـرای خصوصیت "نوع نمایش" بکار می روند حتمـا شـامل یکـی از مقـادیر CSTN، STN، TFT، TFD، OLED می باشند. این موضوع را می توان با استفاده از عبارت منظم و با تعریف یـک الگو (Pattern) نشان داد.
- **الگو ها**: در خلال این تحقیق مشاهده شد که بسیاری از داده ها دارای مقداری نظم نسبی می باشند که تنها با استفاده از عبارات منظم نمی توان آنها را توصیف نمـود. در ایـن خـصوص، پیشنهاد سیستم ما استفاده از آتاماتای غیر قطعی[90]، جهت توصیف نظم موجود مـی باشـد. در بخش بعد به معرفی جزئیات این روش می پردازیم.

---

[88] Network Type
[89] Regular Expression
[90] Non-deterministic Finite state Automata

۵-۲-۴- تعریف الگو ها

معمولا داده های موجود در صفحات وب پیچیده تر از آن هستند که بتوان تنها با استفاده از عبارات با قاعده، آنها را توصیف نمود. عبارات با قاعده برای نمایش نظم موجود در سطح حروف یک کلمه مناسب می باشند. اما گاهی اوقات لازم است تا نظمی را در سطح کلمات سمانتیک و یا حتی در سطح خصوصیات را بیان کنیم و به عنوان مثال این واقعیت که در دامنه گوش های تلفن همراه، ابعاد گوشی رشته ای متشکل از طول و عرض و ارتفاع است را نمی توان با استفاده از عبارات با قاعده نشان داد. گاهی اوقات نیز لازم می شود تا نظم سمانتیکی موجود را نشان دهیم. به عنوان مثال اگر در یک رشته نام کشور، نام شهر وجود داشت، آن را به عنوان آدرس محل می پذیریم.

ما در سیستم OntoByOnto لازم داریم تا بطریقی هر نوع نظم موجود را بطریقی برای سیستم قابل فهم نماییم. روش پیشنهادی ما استفاده از آتاماتای غیر قطعی (NFA) می باشد. مادر اینجا نشان خواهیم داد که چگونه می توان با اندکی تغییرات در مفاهیم برچسب های هر لبه از یک ماشین آتاماتای پایان پذیر، می توان از آن برای توصیف انواع مختلف نظم مقادیر موجود در دامنه استفاده نمود. اما از آن جهت آتاماتای غیر قطعی را انتخاب کرده ایم که ایجاد آن برای کاربران ساده تر بوده و نیز به تعداد وضعیت[۹۱] کمتری نیاز دارد.

در این روش ورودی مجموعه ای از کلمات و سمبل ها (مانند ، : ؛ ' $! و غیره) بوده و سیستم در ابتدا در وضعیت آغازین قرار دارد. عامل انتقال از یک وضعیت به وضعیت دیگر در صورت مشاهده یکی از موارد زیر رخ خواهد داد:

۱. **کلمه**: ماشین NFA ما می تواند در صورت مشاهده یک کلمه خاص، از وضعیت جاری به وضعیت دیگر منتقل شود. کلمه می تواند یک رشته متنی بدون جدا کننده و یا یک عدد صحیح یا اعشاری باشد. همچنین درصورت نیاز می توان با استفاده از عبارت باقاعده تعیین نمود که کلمه مورد نظر باید دارای یک الگوی مشخصی باشد.
۲. **سمبل ها** : سمبل ها نیز می توانند باعث انتقال NFA از وضعیت جاری به وضعیت دیگر شود. در عمل مشاهده شده است که در اکثر مواقع نیازی به پردازش جدا کننده ها نمی باشد. بدین منظور می توان در صورت نیاز سمبل ها را از مجموعه ورودی حذف نمود.

---

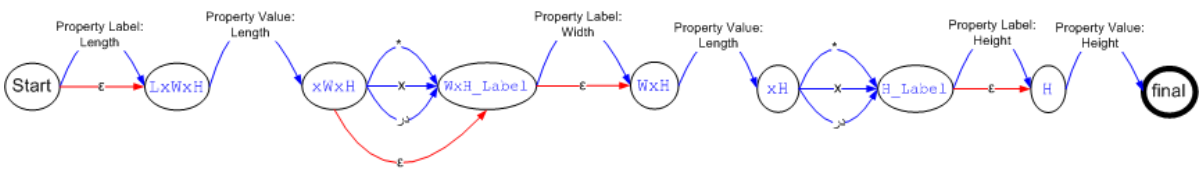
[۹۱] State

۳. **سمانتیک:** گاهی اوقات لازم است تا در صورت مشاهده یک سمانتیک مشخص در ورودی، NFA از وضعیت جاری به وضعیت دیگر منتقل گردد. به عنوان مثال جهت تشخیص آدرس لازم است تا ابتدا سمانتیک کشور و سپس سمانتیک نام استان و سپس سمانتیک نام شهر و غیره را در ورودی مشاهده نماییم. سیستم پیشنهادی ما جهت تشخیص سمانتیک از یک لغت نامه مفهومی استفاده می نماید. البته می توان جهت توسعه سیستم از WordNet که شامل مجموعه کاملی از کلمات انگلیسی به همراه سمانتیک، لغات هم معنی و غیره می باشد، استفاده نمود.

۴. **خصوصیت:** علاوه بر موارد فوق در برخی موارد لازم است تا در صورت مشاهده یک خصوصیت مشخص در ورودی NFA از وضعیت جاری به وضعیت دیگر منتقل گردد. این ویژگی باعث فراخوانی بازگشتی تشخیص الگو ها خواهد شد. چرا که در هنگام تشخیص خصوصیت جدید، الگوریتم تشخیص الگو برای خصوصیت مورد بحث فراخوانی گردد. روشن است که در هنگام فراخوانی بازگشتی تنها ورودی های پردازش نشده به روتین تشخیص خصوصیت ورودی ارسال می گردد.

همانطور که در مثال زیر نیز خواهیم دید، کلا دو نوع ورودی از نوع خصوصیت وجود دارد. یک نوع از ورودی برچسب خصوصیت و دیگری مقدار خصوصیت می باشد. هنگام تشخیص برچسب خصوصیت تنها بخش برچسب های خصوصیت که در آنتالوژی استخراج تعریف شده، بررسی خواهند شد. اما هنگام تشخیص وجود مقدار خصوصیت باید شرایط و محدودیت های تعریف شده برای مقادیر خصوصیت (مانند نوع داده ای، مقادیر حداقل و حداکثر، عبارات باقاعده و غیره) باید با کلمه یا کلمات ورودی منطبق باشد.

شکل زیر تعریف NFA مورد نیاز جهت توصیف الگوی Size را نشان می دهد. این الگو ترکیبی از خصوصیات Width، Length و Height می باشد. لازم به ذکر است که در برخی سایت ها خصوصیت فوق به شکل 105 x 48 x 19 و در برخی دیگر به شکل Length:105, Width:48, Height:19 مشاهده شده است. الگوی NFA زیر حالت های فوق و برخی دیگر از حالات یافت شده توسط مهندس دامنه را به عنوان خصوصیت Size می پذیرد.



شکل ۲۶ : الگوی NFA جهت توصیف خصوصیت Size

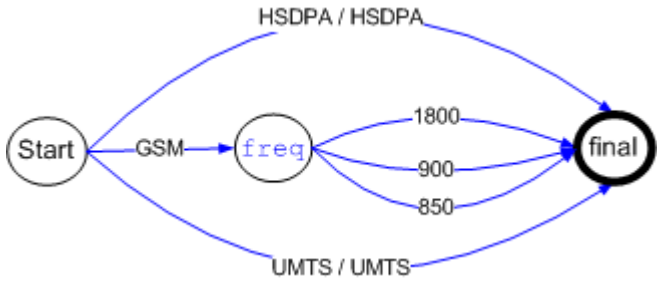
تکه آنتالوژی زیر، کد مرتبط با NFA شکل فوق را نشان می دهد:

```
<pattern id="Size">
  <NFA>

    <node id="start">
      <edge propertylabel="Length" target="LxWxH" />
      <edge target="LxWxH" />
    </node>

    <node id="LxWxH">
      <edge propertyvalue="Length" target="xWxH_Value" />
    </node>

    <node id="xWxH_Value">
      <edge word="*" target="WxH_Label" />
      <edge word="x" target="WxH_Label" />
      <edge word="در" target="WxH_Label" />
      <edge target="WxH_Label" />
    </node>

    <node id="WxH_Label">
      <edge propertylabel="Width" target="WxH" />
      <edge target="WxH" />
    </node>

    <node id="WxH">
      <edge propertyvalue="Width" target="xH" />
    </node>

    <node id="xH">
      <edge word="*" target="H_Label" />
      <edge word="x" target="H_Label" />
      <edge word="در" target="H_Label" />
      <edge target="H_Label" />
    </node>

    <node id="H_Label">
      <edge propertylabel="Height" target="H" />
      <edge target="H" />
    </node>

    <node id="H">
      <edge propertyvalue="Height" />
    </node>

  </NFA>
</pattern>
```

همانطور که در کد فوق دیده می شود، گره ای با عنوان گره پایانی مشخص نشده است. در این کد، خصوصیت Target بیان کننده گره مقصد می باشد. در واقع گره پایانی یک گره تهی می باشد و هر لبه ای که خصوصیت Target نداشته باشد وضعیت جاری را به گره پایانی منتقل خواهد کرد و باعث پذیرفته شدن رشته ورودی خواهد شد.

یکی از ویژگی های منحصر به فرد الگوهای NFA، امکان تعریف یک مقدار مشخص و یکه برای شیوه های مختلف نمایش مقادیر می باشد. به عنوان مثال مشاهده شده است که برخی سایت ها "نوع شبکه" مورد پشتیبانی یک گوشی همراه را بصورت GSM 1800, GSM 1900, GSM 900 و برخی دیگر بصورت GSM 1800, 1900, 900 نمایش داده اند. بدیهی است که مقادیر رشته ای فوق هیچگاه با هم برابر نمی باشند در حالیکه از نظر مفهومی هر دو به یک مجموعه از انواع شبکه قابل پشتیبانی اشاره دارند.

در الگو های NFA، به ازای یک ورودی مشخص می توانیم یک خروجی مورد انتظار را نیز تعریف نماییم. شکل زیر تعریف NFA مورد نیاز جهت توصیف الگوی نوع شبکه را نشان می دهد.



شکل ۲۷ : الگوی NFA جهت توصیف خصوصیت NetworkType

تکه کد آنتالوژی زیر کد الگوی فوق را نشان می دهد:

```
<pattern id="NetworkType">
  <NFA removesymboles="true">

    <node id="start" >
      <edge word="UMTS" value="UMTS"  />
      <edge word="HSDPA" value="HSDPA"/>
      <edge word="GSM" target="freq"/>
      <edge target="freq"/>
    </node>

    <node id="freq">
      <edge word="1900" value="GSM 1900"/>
      <edge word="1800" value="GSM 1800"/>
      <edge word="900" value="GSM 900"/>
      <edge word="850" value="GSM 850"/>
    </node>

  </NFA>
</pattern>
```

همانطور که مشاهده می شود، سیستم پس از پردازش هر دو نوع ورودی فوق الذکر مربوط به نوع شبکه های قابل پشتیبانی، مجموعه یکسان {GSM_1800, GSM_1900, GSM_900} را در خروجی تولید می کند.

۵-۲-۵- تعریف فهرست واژگان آنتالوژی[۹۲]

فرهنگ لغات، شیوه ای جهت توصیف دانش صریح موجود می باشد. در این بخش، مهندس دامنه، اطلاعات اولیه خود از مقادیر و واژگان ممکن برای یک خصوصیت را بیان می کند. به عنوان مثال کد زیر لیست برندهای شناسایی شده توسط مهندس دامنه را نشان می دهد.

```
<vocabulary>

    <property resource="Brand">
      <domains>
        <domain>CellPhone</domain>
      </domains>

      <words>
        <word means="SonyEricsson">SonyEricsson</word>
        <word means="SonyEricsson">Sony-Ericsson</word>
        <word>Nokia</word>
        <word>Samsung</word>
        <word>Motorola</word>
        <word>LG</word>
        <word>Alcatel</word>
        <word>Panasonic</word>
        <word>Simense</word>
        <word>BenQ</word>
        <word>Philips</word>
        <word>NEC</word>
        <word>Sagem</word>
        <word>Pantech</word>
        ...
      </words>

    </property>
  </vocabulary>
```

همانطور که در کد فوق دیده می شود، فهرست واژگان، روشی جهت ترجمه داده های موجود نیز می باشد. به عبارتی به ازای هر کلمه می توان واژه ای جدید و معادل با آن را نیز معرفی نمود. به عنوان مثال زمانیکه سیستم عبارت Sony-Ericsson را در ورودی می بیند، آن را به واژه SonyEricsson تبدیل می نماید. همچنین هنگام توصیف لغات باید مشخص نمود که مربوط به چه کلاس هایی می باشد. البته باید توجه داشت که مقادیر ممکن برای خصوصیت مورد بحث، محدود به واژگان توصیف شده نمی باشد و این مقادیر تنها برای تشخیص سریع خصوصیت مربوط به یک داده خام کاربرد دارد. در کد فوق، به صراحت ذکر شده است که واژه های SonyErricson، Nokia و غیره مربوط به خصوصیت Brand از کلاس CellPhone می باشد.

---

[92] Vocabulary

بدین ترتیب، زمانیکه روتین تشخیص دهنده خصوصیت به یک واژه مانند Nokia برخـورد مـی کنـد، بلافاصله تشخیص می دهد که این واژه مرتبط با خصوصیت Brand می باشد و از آنجا که مجموعه مقادیر خصوصیت Brand مستقل از سایر خصوصیات می باشد، لذا تنها خصوصیت مـرتبط بـا واژه فـوق Brand خواهد بود. به عبارتی دیگر، هیچ یک از خصوصیت ها بغیر از Brand، دارای مقدار Nokia نخواهـد بـود. همچنین با توجه به این موضوع که هر نمونه از کـلاس CellPhone حـداکثر مـی توانـد یـک واژه Brand داشته باشد، لذا این خصوصیت از مجموعه خصوصیات ممکن برای سایر اقلامـی کـه در رکـورد مربوطـه قرار دارند، حذف می گردد.

۵-۲-۵- آنتالوژی استخراج برای یک صفحه وب نمونه

با توجه به مثال لیست کشور ها که در بخش های قبلی مطرح شد و از آنجا که از این مثال در اجرای الگوریتم جستجوی الگو های استخراج نیز استفاده خواهد شد، ذیلا آنتالوژی استخراج مرتبط با آن را نشان می دهیم. البته این آنتالوژی استخراج تنها مقادیر مجاز برای کد کشور ها و نام کشور ها را تعریف می کند و لذا از توصیف مقادیر سایر خصوصیت های احتمالی در دامنه مورد بحث خودداری شده است.

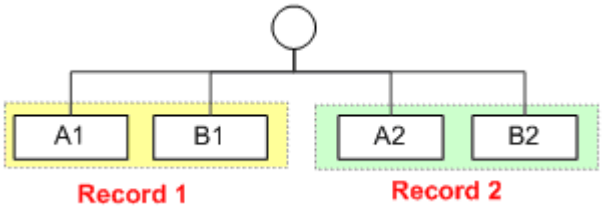
```
<?xml version="1.0"?>

<ontology about="CountryCodes">

  <classes>

    <class id="CountryCodes" abstract="false">
      <properties>
        <property resource="CountryName" cardinality="1"/>
        <property resource="CountryCode" cardinality="1"/>
      </properties>
    </class>

  </classes>


  <properties>

    <property id="CountryName" datatype="string" maxwordscount="3">
      <labels>
        <label>Name</label>
        <label>Country</label>
        <label>Country Name</label>
      </labels>
      <disjointwith>
```

```
        <property resource="" />
      </disjointwith>
      <pattern resource="CountryNamePettern"/>
    </property>

    <property id="CountryCode" datatype="int"
                               minvalue="0" maxvalue="1000">
      <labels>
        <label>Code</label>
        <label>Country Code</label>
      </labels>
      <disjointwith>
        <property resource="CountryName" />
      </disjointwith>
    </property>


  </properties>



  <patterns>

    <pattern id="CountryNamePettern">
      <NFA>
        <node id="start">
          <edge semnatic="Country Name" target="S1"/>
        </node>
        <node id="S1">
          <edge symbol=":" />
        </node>
      </NFA>
    </pattern>

  </patterns>


  <vocabulary>

    <property resource="CountryName">
      <domains>
        <domain resource="CountryCodes" />
      </domains>
      <words>
        <word>Iran</word>
        <word means="Iran">ایران</word>
        <word means="Iran">Islamic republic of iran</word>
      </words>
    </property>

  </vocabulary>


</ontology>
```

۵-۳- الگوریتم ایجاد اتوماتیک WDML

اکنون که با معماری سیستم ، ساختار WDML و کاربرد آنتالوژی استخراج آشنا شدیم لازم است تا روشی جهت تولید اتوماتیک الگوی WDML ارائه نماییم. الگوریتم پیشنهادی ما براساس مفاهیم آنتالوژی دامنه که در یک صفحه وب یافت می گردد و با توجه به ساختار درختی نمایش اطلاعات اقدام به یافتن الگو های موجود در صفحه وب می نماید . روتین Create pattern الگوهای موجود در صفحه را به روش پایین به بالا (Bottom up) یافته و به صورت یک ساختار درختی ساده در حافظه ایجاد می کند . سپس روتین WritePattern الگوهای یافت شده را با توجه به ساختار wdml در خروجی می نویسد.

قبل از ادامه بحث و تشریح الگوریتم لازم است تا یک پیش فرض اساسی را بیان کنیم . در الگوریتم پیشنهادی ما فرض می گردد که صفحات، دارای رکورد های همپوشان نمی باشند. منظور از عدم همپوشانی رکورد ها آن است که در گره های همجوار و دارای یک گره پدر، باید ابتدا اطلاعات یک رکورد بصورت کامل و سپس اطلاعات رکورد بعدی ظاهر گردد(شکل زیر)



شکل ۲۸: عدم همپوشانی رکورد ها

باید توجه داشت که نیاز به عدم همپوشانی رکورد ها یکی از محدودیت های ساختار Wdml است. در این ساختار از آنجا که الگوهای درختی تنها برای توصیف یک رکورد استفاده می گردند، لذا امکان توصیف دو یا چند رکورد با یک الگو وجود ندارد. البته از آنجا که این پیش فرض با شیوه طراحی بسیاری از برنامه های وب سازگاری دارد، بنابراین انتظار می رود محدودیت قابل توجهی را ایجاد ننماید.

الگوریتم های پیشنهادی ما که در اینجا تشریح خواهد شد ، برای مجموعه صفحات یک وب سایت و به ازای یک کلاس مشخص از آنتالوژی استخراج، الگوها را تشخیص می دهند. لذا به راحتی می توان آن را برای صفحات سایر وب سایت ها و نیز سایر کلاس های آنتالوژی استخراج بسط داد. در ادامه ابتدا به تشریح روتین سطح بالای GeneratWDML و سپس روتین های فرعی مورد استفاده در آن می پردازیم.

۵-۳-۱- تولید WDML

از آنجا که هدف اصلی، تولید اتوماتیک فایل ها WDML می باشد، لذا در بالاترین سطح ابتدا صفحات موجود را خوانده و سپس یک Wrapper مناسب برای آن ایجاد می کنیم. Wrapper های ایجاد شده در یک فایل Wdml نگهداری خواهند شد. این عملیات توسط روتین GneratWdml برای یک وب سایت و یک کلاس مشخص از آنتالوژی استخراج ، انجام خواهد شد . ذیلا شبه کد مربوط به روتین مذکور آورده شده است .

```
public void GenerateWDML(WebSite, class)
{
W = An Empty List of WDML
P = List of Pages in WebSite
for each ( Page pi in P )
{
    Done = false;
    for each ( Wrapper wi in W.Wrappers )
    {
        if( wi.Match(pi, class) )
        {
            wi.Resources.IncludeURL( pi.URL );
            Remove pi from P
            Done = True;
            break;
        }
        else
            wi.Resources.ExcludeURL( pi.URL );
    }
    if( !Done )
    {
        wrapper = CreateWrapper( pi, class );
        wrapper.Resources.IncludeURL( pi.URL );
        for each ( Wrapper wi in W.Wrappers )
            for each ( url in wi.Resources )
                wrapper.Resources.ExcludeURL( url );
        wdml.Wrappers.Add( wrapper );
    }
}
}
```

لیست ۳: الگوریتم تولید WDML

نکته مهم در این روتین آن است که مجموعه ای از صفحات می توانند دارای یک WDML مشترک باشند. لذا ، قبل از ایجاد Wrapper برای یک صفحه آن را بر روی wrapper های موجود آزمایش کرده و تنها در صورتیکه هیچ یک از Wrapper ها جهت استخراج اطلاعات از صفحه مورد نظر مناسب نباشند، یک Wrapper جدا گانه برای آن ایجاد می نماییم .

از طرفی دیگر ، این موضوع را هم در نظر خواهیم داشت که ممکن است بدلایلی (مانند ضعف در آنتالوژی استخراج و عدم رعایت شرایط و محدودیت های سیستم و غیره) سیستم نتواند الگو های موجود را شناسایی کند. لذا ما اینگونه صفحات را به عنوان صفحات مردودی[۹۳] علامت گذاری می کنیم تا بعدا توسط مدیر سایت بررسی شده و بهبود های لازم در سیستم صورت گیرد.

یکی از قابلیت های مهم این الگوریتم، امکان دسته بندی اتوماتیک آدرس صفحات مرتبط با یک Wrapper می باشد. این قابلیت می تواند در بهبود کارایی سیستم بسیار موثر باشد. از آنجا که مشکل اصلی در ارتباط با الگوریتم فوق، تعداد مقایسه های آن می باشد، در صورت دسته بندی اتوماتیک آدرس صفحات،به محض ایجاد یک Wrapper برای یک صفحه مشخص، سایر صفحات مشابه با آن، جهت آزمایش انطباق با آن کاندید می شوند. لذا انتخاب یک الگوریتم جهت تعیین روش دسته بندی، بسیار مهم می باشد. دسته بندی می تواند بر اساس یکی از موارد زیر صورت گیرد:

- بر اساس پارامتر ورودی به صفحه، مانند www.mysite.com/MyPage.aspx?q=*
- بر اساس صفحه، مانند www.mysite.com/Products/*
- بر اساس تشابه بخشی از نام صفحه، مانند www.mysite.com/**Noki**_ *

اتخاذ یک روش مناسب بسیار مهم بوده و باعث افزایش کارایی و هوشمندی سیستم در برخورد با صفحات جدید می باشد. البته شبه کد فوق باید جهت استفاده از قابلیت های دسته بندی بهینه گردد. ولی از آنجا که قصد نداریم بیش از این وارد مقوله دسته بندی صفحات شویم، بهینه سازی های لازم را به تحقیقات بیشتر در این خصوص واگذار می نماییم.

---

[93] Rejected Pages

۵-۳-۲- ایجاد Wapper

اولین گام در تولید یک Wrapper ایجاد ساختار درختی از صفحه وب ورودی و اجرای آنتالوژی استخراج بر روی گره های متنی آن می باشد. تجربه ما در این تحقیق نشان داده است که ایجاد یک آنتالوژی استخراج کامل، عملی پر زحمت و طاقت فرسا بوده و علیرغم تمام تلاشی که صورت می گیرد باز هم ممکن است برای یک مقدار داده ای مشخص، دو یا چند خصوصیت (از آنتالوژی استخراج) یافت گردد.

راه حل پیشنهادی ما جهت مقابله با وابستگی بیش از حد به آنتالوژی استخراج و افزایش قابلیت انعطاف پذیری سیستم، رتبه بندی میزان انطباق هر خصوصیت، استفاده از سایر خصوصیت های منطبق شده و در نهایت استفاده از قاعده نزدیکترین برچسب می باشد. البته عملیات فوق تنها بر روی گره های تشکیل دهنده یک رکورد قابل اجرا می باشند بنابراین لازم است بعد از اجرای آنتالوژی استخراج، ابتدا محدوده هر رکورد تعیین شده و بر اساس آن، الگوی اولیه استخراج مشخص گردد. در هنگام تشکیل الگوی اولیه استخراج تا حد امکان بخشی از مشکلات مربوط به تعدد خصوصیات کاندید برای یک قلم اطلاعاتی بر طرف خواهد شد. در نهایت پس از مشخص شدن محدوده رکورد، روش های پیشنهادی فوق را بکار خواهیم بست تا هر قلم داده تنها دارای یک خصوصیت کاندید باشد.

شبه کد زیر، الگوریتم سطح بالای CreateWrapper را نشان می دهد:

```
public Wrapper CreateWrapper( PageContents, class )
{
    tt = Create_HtmlTagTree( PageContents );
    Apply_ExtractionOntology( tt );
    w = New instance of Wrapper
    pat = CreatePattern( HtmlBody );
    w.SetHtmlBody( pat );
    return w;
}
```

لیست ۴: الگوریتم تولید Wrapper

همانطور که در این الگوریتم مشاهده می شود، فرایند ایجاد Wrapper، را می توان به مراحل کوچکتر زیر تقسیم کرد:

۱. ایجاد ساختار درختی صفحه وب: در این مرحله، بر اساس محتوای صفحه وب ساختار درختی آن ایجاد می شود. هر گره از درخت مزبور دارای اعضای زیر است:

- Nodes : مجموعه ای[۹۴] از گره های فرزند
- Name: نام المان معادل در صفحه وب (مانند Div، Table، Span و غیره)
- Attributes: مجموعه ای از خصوصیات تعریف شده برای المان مورد بحث.

از آنجا که در ساختار درختی هر یک از المان های موجود در صفحه وب نهایتا به یک گره در ساختار درختی تبدیل می شود، لذا برای نمایش متن ها نیز یک گره از نوع HtmlText ایجاد می گردد.

۲. اجرای آنتالوژی استخراج بر روی ساختار درختی فوق: با توجه به آنتالوژی استخراج، در ساختار درختی، متن موجود در گره های متنی (گره های از نوع HtmlText) را با توجه به خصوصیات موجود در آنتالوژی استخراج، پردازش می کنیم. به ازای هر گره متنی مشخص می کنیم کدامیک از خصوصیات با آن سازگار می باشد.

خصوصیات یافت شده به مجموعه Properties (یکی از اعضای گره ها در ساختار درختی) اضافه می گردد. در مراحل بعد از این اطلاعات جهت یافتن الگو های موجود استفاده می گردد.

۳. شناسایی الگوی موجود در ساختار درختی فوق: با توجه به ساختار درختی صفحه وب و لیست خصوصیاتی که هر گره و گره های فرزند آن می تواند شامل باشد، اقدام به یافتن الگو های می نماییم. در بخش بعدی به تشریح جزئیات این الگوریتم خواهیم پرداخت.
۴. ایجاد الگوی WDML بر اساس الگوهای یافت شده در مرحله قبل: ساختار الگو های یافت شده در مرحله قبل با ساختار درختی صفحات وب ادغام شده خواهد بود. لذا جهت ایجاد یک الگو مطابق با زبان WDML، باید این ساختار را به فرمت توصیف شده در آن زبان تبدیل کنیم. به عنوان مثال، هر المان از ساختار درختی باید با یک المان از نوع Element و

---

[94] Collection

صفت name مشخص می شوند و خصوصیت ها با المان Attribute و صفت های name و value مشخص می گردند.

مهمترین بخش این الگوریتم ایجاد الگوها می باشد. نکته مهم در هنگام شناسایی الگو ها، توجه به کاربردی بودن الگوریتم می باشد. چرا که برنامه نویس هنگام تولید نرم افزار با یک الگوی مشخص جهت تولید مجموعه ای مشخص از صفحات وب کار می کند. وی همچنین می تواند با کد نویسی الگوی فوق را کاملا به هم بریزد. بنابراین در حالت کلی نمی توان روش مشخص و ثابتی جهت یافتن الگو های اولیه ارائه کرد. اما با توجه به این موضوع که اکثر صفحات دارای نظم مشخص می باشند، می توان روشی ارائه کرد که در بسیاری از صفحات قابل استفاده باشد.

در این تحقیق هدف ما ارائه یک الگوریتم جامع و بی نقص جهت شناسایی الگوها نبوده است. لذا این پیش بینی هم شده است که الگوریتم ما در مواردی با شکست مواجه شود. در این صورت سوپروایزر سیستم می تواند با نظارت بر روند اجرای اتوماتیک سیستم، در صورت نیاز، تغییرات لازم را در سیستم جاری سازد.

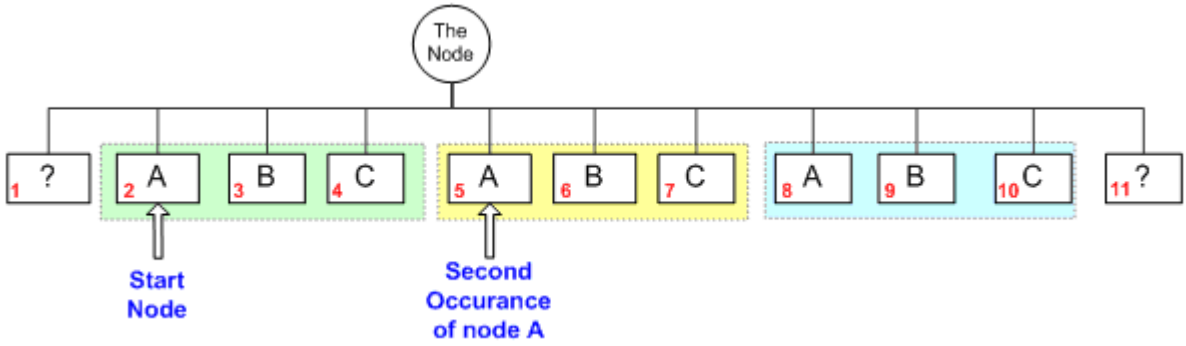
۵-۳-۳- یافتن الگو ها

الگوریتم پیشنهادی ما مبتنی بر ساختار و خصوصیات یافت شده در یک گره یا گره های فرزند آن گره می باشد. البته ممکن است این الگوریتم بسیار کامل نباشد و بتوان الگوریتم های بهبود یافته دیگری نیز ارائه نمود که در این صورت با جایگزینی آن می توان بخش جستجوی الگو ها را ارتقاء داده و کلیت سیستم را حفظ نمود.

الگوریتم پیشنهادی ما در ساختار درختی، از پایین به بالا اقدام به یافتن الگو ها می نماید. در این الگوریتم جهت یافتن الگو از بین فرزندان یک گره، باید مراحل زیر طی گردد:

۱. اگر گره مورد بحث، یک گره متنی (از نوع HtmlText) باشد، خود گره به عنوان الگو برگردانده می شود.
۲. به ازای هر یک از فرزندان گره مورد نظر، با فراخوانی بازگشتی همین الگوریتم، الگوی آن گره را ایجاد می کنیم. این الگو در صفت Pattern از گره مورد نظر نگهداری می گردد.

۳. یافتن اولین گره فرزند (گره آغازین) که خود یا فرزندان آن، حداقل با یکی از خصوصیات موجود در آنتالوژی استخراج مطابقت داشته باشد.
۴. یافتن اولین تکرار گره آغازین در بین گره های فرزند و ایجاد یک الگوی ساده با شروع از گره آغازین تا گره یافت شده در این مرحله. در صورتیکه این الگو مربوط به یک رکورد باشد، بررسی خواهیم کرد که آیا این الگو بصورت پیوسته در ادامه الگوی قبلی، تکرار می گردد یا نه؟ (شکل زیر)
۵. در صورتیکه یافتن تکرار های الگوی فوق در بین رشته گره های فرزند، با شکست مواجه شود، سعی خواهیم کرد تا تکرار دیگری را برای گره آغازین بیابیم. سپس الگوی جدیدی را با شروع از گره آغازین و این گره جدید، ایجاد خواهیم کرد. سپس در بین رشته گره های فرزند، تکرار های الگوی فوق را جستجو خواهیم کرد. این عملیات را تا زمان یافتن یک الگوی کامل که تمام گره ها را شامل گردد، ادامه خواهیم داد.



شکل ۲۹: جستجوی الگو در گره های فرزند

در شکل فوق، TheNode گره ورودی به الگوریتم است. در این شکل، اعداد قرمز رنگ نشان دهنده شماره گره فرزند بوده و نیز حروف حک شده بر روی هر گره، خصوصیت مرتبط با آن گره (یا گره های فرزند آن) را نشان می دهد. سمبل ؟ نشان می دهد که هیچ نوع خصوصیت مرتبط با گره مورد نظر در آنتالوژی استخراج یافت نشده است. بدیهی است که چنین گره هایی را می توان از الگوی رکورد مورد جستجو حذف نموده و در خارج از الگوی رکورد قرار داد. بنابراین گره آغازین در این الگوریتم، گره شماره ۲ (مرتبط با خصوصیت A) خواهد بود.

در بین رشته گره های فرزند TheNode اولین تکرار گره آغازین، گره شماره ۵ می باشد. بنابراین یک الگوی ساده با شروع از گره آغازین (گره شماره ۳) تا گره شماره ۵، ایجاد می کنیم (مانند شکل زیر). اکنون سعی خواهیم کرد تا تکرار های این الگو را در ادامه رشته گره های فرزند بیابیم.

شکل ۳۰: جستجوی الگو در گره های فرزند

در ادامه، گره آغازین را با شروع از گره ۶، جستجوی کرده و گره شماره ۸ را خواهیم یافت. بنابراین الگوی مورد نظر بین گره های ۵ تا ۷ وجود خواهد داشت، لذا الگوی ساده ای را بر اساس این گره های ایجاد کرده و آن را با الگوی اولیه انطباق می دهیم. در صورت انطباق این دو الگو، عملیات فوق را جهت یافتن الگو های بعدی ادامه می دهیم. ذیلا، شبه کد الگوریتم جستجوی الگو ها نشان داده شده است:

```
public HtmlElement CreatePattern( theNode )
{
if(  theNode is HtmlText ) return theNode;
nodes = theNode.nodes;
for( int i=0; i<nodes.count; ++i)
    theNode[i].Pattern = CreatePattern( theNode[i] );

i = IndexOfFirstDataNode( nodes );
j = i
pattern = null;
while( j < nodes.count )
{
    j = IndexOfMatchedNode( nodes, i, j+1 );
    pattern = MakeSimplePattern(nodes, i, j);

    while( n < nodes.count && IsRecordPattern(pattern, class) )
    {
            n = IndexOfMatchedNode( nodes, i, m+1 );
            P = MakeSimplePattern(nodes, m, n);
            pattern = ExpandPattern( pattern, P );
            m = n;
    }
    if( pattern != null ) break;
}

if( pattern == null )
    pattern = MakeSimplePattern(nodes, 0, nodes.count);
else
```

```
    {
        if( IsRecordPattern(pattern, class) )
            MarkAsRecordPattern( pattern );
        P = MakeSimplePattern(nodes, 0, i);
        pattern = CombinePattern( P, pattern );
    }
    return AdjustPattern( pattern );
    }
```

لیست ۵: الگوریتم تولید الگو

همانطور که دیده می شود در این الگوریتم، مقادیر داده ای و متنی بصورت مستقیم در فرایند جستجوی الگوها شرکت ندارند. بلکه ابتدا تمام متن ها با اعمال آنتالوژی ماژول استخراج پردازش شده و خصوصیات مرتبط با هر یک از آنها در گره مربوطه از ساختار درختی نگهداری می گردد. بدین ترتیب الگو ها بر اساس تکرار منظم خصوصیات تعریف شده در آنتالوژی استخراج یافت می شوند.

همانطور که در الگوریتم فوق دیده می شود، از تعدادی روتین اولیه جهت کار با الگو ها استفاده شده است که در ادامه به تشریح آنها خواهیم پرداخت

۵-۳-۴- روتین های اولیه مورد نیاز در پردازش الگو ها

همانطور که در بخش قبل دیده شد جهت استخراج الگو های موجود در ساختار درختی ورودی، به برخی توابع و روتین جهت انجام پردازش های اولیه، مانند مقایسه الگو ها و ترکیب الگو ها و غیره لازم داریم. در این بخش قصد داریم تا به تشریح این روتین ها بپردازیم.

۵-۳-۴-۱- تشخیص اولین گره داده ای - IndexOfFirstDataNode

گره داده ای، گره ای است که خود یا یکی از فرزندان آن به یکی از خصوصیات موجود در آنتالوژی استخراج مقید[۹۵] شده باشند. هر گره دارای صفتی بنام Properties می باشد که لیست خصوصیات مرتبط با آن گره را نشان می دهد. با استفاده از Properties تشخیص گره های داده ای بسیار ساده می گردد. ذیلا کد مربوط به این روتین به زبان C# آورده شده است.

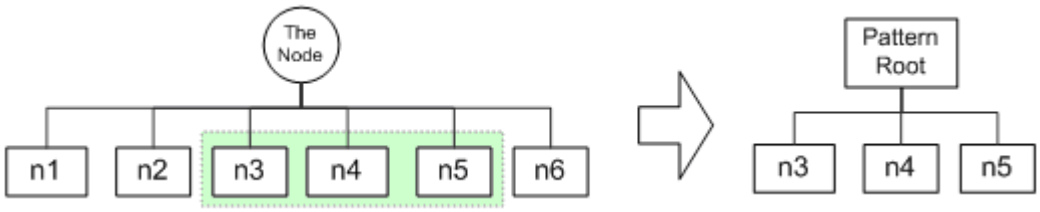
```
public int IndexOfFirstDataNode( nodes )
{
    for(int i=0; i<nodes.Count; ++i)
        if( nodes[i].Properties.Count>0 )
            return i;
    return nodes.Count;
}
```

۵-۳-۴-۲- ایجاد الگوی ساده از گره های همجوار- MakeSimplePattern

گاهی اوقات لازم می شود از مجموعه ای از گره های همجوار یک الگوی ساده ایجاد نماییم. هدف آن است که به این مجموعه از گره ها تنها با استفاده از یک گره ریشه (گره مجازی) دسترسی داشته باشیم. به عنوان مثال، در شکل زیر جهت ایجاد الگو از گره های n3 و n4 و n5 یک گره مجازی ایجاد کرده و گره های مذکور را در لیست فرزندان آن اضافه می کنیم.

---

[95] bind



شکل ۳۱: ایجاد الگوی ساده

منظور از گره مجازی، گره ای بدون نام و صفت می باشد. همانطور که دیده می شود از این گره جهت ایجاد ساختار سلسله مراتبی الگو ها استفاده می گردد. ذیلا کد مربوط به این روتین به زبان C# آورده شده است.

```
public HtmlElement MakeSimplePattern(NodeCollection nodes,
                                     int StartIndex, int ToIndex)
{
    HtmlElement root = new HtmlElement();
    for(int i=StartIndex; i<ToIndex; ++i)
        root.Nodes.Add( nodes[i] );
    return root;
}
```

۵-۳-۴-۳- تشخیص گره داده ای منطبق با گره دیگر- IndexOfMatchedNode

گاهی اوقات لازم می شود که تکرار یک گره مشخص را در میان سایر گره ها بررسی کنیم. برای این کار لازم است تا دو گره را با هم مقایسه نماییم.

```
public int IndexOfMatchedNode(NodeCollection nodes, int SourceIndex,
                                                    int StartIndex)
{
    HtmlNode TheNode = nodes[ SourceIndex ];
    for(int i=StartIndex; i<nodes.Count; ++i)
        if( IsEqualPattern(TheNode, nodes[i]) )
            return i;
    return nodes.Count;
}
```

تابع IsEqualPattern بررسی می کند که آیا الگو های دو گره با هم منطبق می باشد یا نه. الگوی های دو گره n1 و n2 زمانی با هم منطبق می باشند که دارای شرایط زیر باشند:

۱. نام گره n1 و نام گره n2 با هم برابر باشد
۲. گره های فرزند از الگوی n1 نظیر به نظیر با گره های فرزند از الگوی n2 برابر باشد. برای این منظور تابع IsEqualPttern را بصورت بازگشتی فرا خوانی می کنیم.

ذیلا کد مربوط به روتین IsEqualPattern، به زبان C# آورده شده است.

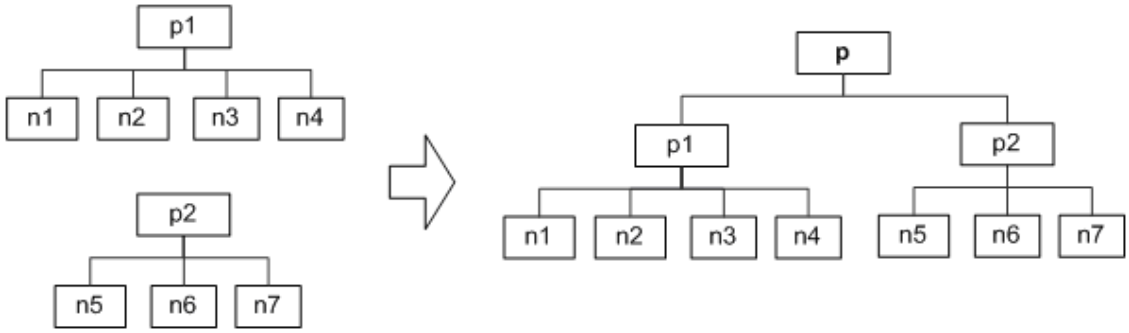
```
public bool IsEqualPattern( HtmlNode n1, HtmlNode n2 )
{
    if( n1.Name != n2.Name ) return false;
    HtmlNode p1 = n1.Pattern;
    HtmlNode p2 = n2.Pattern;
    if( p1.Name != p2.Name ) return false;
    if( p1.Nodes.Count != p2.Nodes.Count ) return false;
    for( int i=0; i<p1.Nodes.Count; ++i )
        if (!IsEqualPattern(p1.Nodes[i], p2.Nodes[i]))
            return false;
    return true;
}
```

همانطور که مشاهده می شود، الگوریتم فوق تنها بر اساس نام المان و ساختار الگوها امکان انطباق را بررسی می کند. در صورت نیاز می توان پارامتر های دیگر، را هم در نظر گرفت. به عنوان مثال می توان علاوه بر بررسی تساوی نام المان ها، برخی خصوصیات نمایشی المان ها، مانند طول، عرض، نوع فونت و غیره را هم در مقایسه نمود. باید توجه داشت که هرچه پارامتر های مورد بررسی بیشتر باشد، دقت سیستم در هنگام استخراج اطلاعات بالا رفته و در مقابل از انعطاف پذیری سیستم کاسته خواهد شد، و برعکس هرچه پارامتر های مورد بررسی کمتر باشد، دقت سیستم در هنگام استخراج اطلاعات بالا کاهش یافته و در مقابل از انعطاف پذیری سیستم افزایش می یابد.

از این رو باید با انتخاب پارامتر های مناسب، بین دقت و انعطاف پذیری سیستم تعادل برقرار کرد. در محدوده این پروژه با توجه به دامنه گوشی های تلفن همراه، پیشنهاد ما بررسی نام المان و ساختار الگو ها می باشد. بدیهی است در تحقیقات آتی می توان با توجه به نیاز، پارامتر هایی مانند اندازه قلم، رنگ متن، نام فونت و غیره را هم در مقایسه ها شرکت داد.

۵-۳-۴-۴- ترکیب الگو ها - CombinePattern

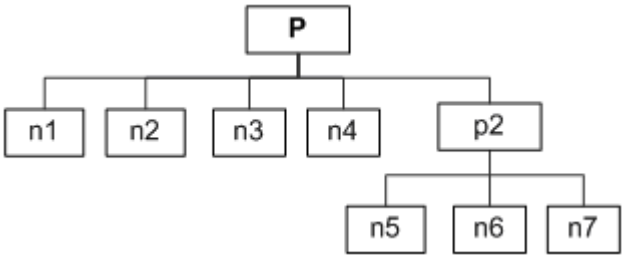
گاهی اوقات لازم می شود که دو الگو را با هم ترکیب نماییم. ترکیب دو الگوی p1 و p2 یک الگوی جدید بنام p خواهد بود که ابتدا الگوی p1 و سپس الگوی p2 را می پذیرد. شکل زیر این موضوع را به خوبی نشان می دهد.



شکل ۳۲: ادغام الگوها

جهت جلوگیری از افزایش بی رویه تعداد سطوح، باید ساختار الگو ها را نرمال نمود. بدین معنی که در هنگام ادغام الگو ها، در صورتیکه گره ریشه هریک از الگو های اولیه (در اینجا p1 یا p2)، یک گره مجازی بوده و کاردینالیتی آن یک باشد، دیگر نیازی به آن گره مجازی نخواهد بود. بنابراین می توان آن را حذف کرده و فرزندان آن را به لیست فرزندان گره مجازی جدید اضافه نمود.

به عنوان مثال اگر در شکل فوق کاردینالیتی گره p1، یک باشد، ترکیب دو الگوی فوق بصورت زیر خواهد بود.



شکل ۳۳: نرمال سازی در ادغام الگوها

ذیلا کد مربوط به روتین CombinePattern، به زبان C# آورده شده است.

```
public HtmlElement CombinePattern( HtmlElement P1, HtmlElement P2 )
{
    HtmlElement root = new HtmlElement();
    if( P1.Name=="" && P1.MinCardinality==1 && P1.MaxCardinality==1 )
        for(int i=0; i<P1.Nodes.Count; ++i)
            root.Nodes.Add( P1.Nodes[i] );
    else
        root.Nodes.Add( P1 );

    if( P2.Name=="" && P2.MinCardinality==1 && P2.MaxCardinality==1 )
        for(int i=0; i<P2.Nodes.Count; ++i)
            root.Nodes.Add( P2.Nodes[i] );
    else
        root.Nodes.Add( P2 );

    return root;
}
```

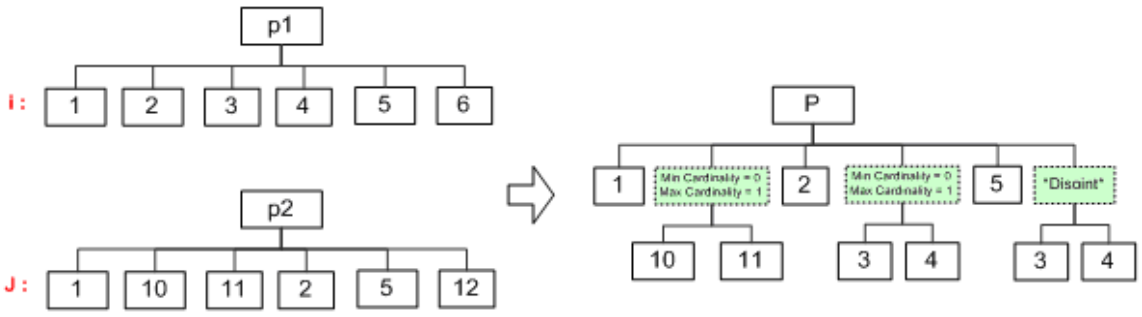
۵-۳-۴-۵- بسط الگو ها – ExpandPattern

هنگام استخراج الگو ها گاهی اوقات لازم خواهد شد تا الگو های کوچک یافت شده را با هم مقایسه کرده و در صورت امکان آنها را بسط دهیم تا الگوی جدیدی ایجاد گردد که بتواند هر یک از الگو های اولیه را بصورت جداگانه بپذیرد. در صورتیکه تمام رکورد های اطلاعاتی موجود در یک صفحه وب تحت یک قالب و الگوی ثابت ایجاد شده باشند، در روتین CreatePattern، تمام الگو های یافت شده یکسان خواهند بود. لذا خروجی روتین بسط الگو ها معادل با الگوی ورودی می باشد.

از آنجا که صفحات مورد تحقیق ما دارای قالب های ثابتی جهت نمایش اطلاعات بوده اند، بنابراین روتین بسط الگو ها بصورت زیر پیاده سازی شده است:

```
public HtmlNode ExpandPattern( HtmlNode P1, HtmlNode P2 )
{
    if( IsEqualPattern(P1, P2) )
        return P1;
    return null;
}
```

همانطور که دیده می شود تنها شرط ممکن برای الگو های ورودی P1 و P2، برابر بودن آن دو می باشد. از آنجا که اغلب رکورد های اطلاعاتی موجود در یک صفحه کاملا از نظر ساختار و محتوی مشابه به هم می باشند، روش فوق در بیشتر حالت ها موفقیت آمیز خواهد بود. با این وجود لازم است تا روشی نیز برای زمانیکه الگو ها با هم متفاوت می باشند، ارائه نماییم. در شکل زیر، P1 و P2 الگو های ورودی به روتین ExpandPattern می باشند. هدف یافتن یک الگوی گسترش یافته ای مانند الگوی P است که الگو های P1 و P2 زیر الگوی آن می باشند.



شکل ۳۴: ساخت الگو های پیچیده

روش های مختلفی جهت بسط میان دو یا چند الگو وجود دارد. روش پیشنهادی ما آن است که بخش هایی از یک الگو که در دیگری وجود ندارد را به عنوان بخش های انتخابی[۹۶]، علامت گذاری نماییم این بخش ها دارای حداقل کاردینالیتی صفر و حداکثر کاردینالیتی یک می باشند. همانطور که در شکل فوق نیز دیده می شود، گره های ۱۰ و ۱۱ در الگوی p2 وجود دارند که در الگوی P1 وجود ندارند، لذا در الگوی P، این دو گره در بخش فرزندان یک گره مجازی که حداقل کاردینالیتی آن صفر می باشد، قرار گرفته اند.

جهت ایجاد الگوی بسط یافته ابتدا یک گره مجازی معادل با P1 و P2 ایجاد کرده و با توجه به فرزندان P1 و P2، فرزندان P را ایجاد می کنیم. در اینجا فرض می کنیم متغیر های i و j به ترتیب شمارنده های گره های فرزند P1 و P2 می باشند. سپس مراحل زیر را دنبال می کنیم:

۱. گره موجود در محل i را با گره موجود در محل j مقایسه می کنیم. اگر این دو با هم برابر بودند یکی از آنها را در لیست فرزندان P درج کرده و سپس یک واحد به مقدار i و j اضافه می کنیم.
۲. اگر گره های فوق با هم متفاوت بودند، با شروع از گام یک، شرایط زیر را بررسی می کنیم
   a. گره موجود در محل i ام را با گره موجود در محل j+step مقایسه می کنیم[۹۷]. اگر این دو با هم برابر بودند، ابتدا از گره های موجود در فاصله j تا j+step یک الگوی ساده ایجاد کرده و سپس آن را به عنوان گره انتخابی (Optional) به لیست فرزندان P درج کرده درج می نماییم. مقدار j را هم به مقدار step (اندازه گام افزایش می دهیم)
   b. در صورت مغایرت در مقایسه فوق، عملیات فوق را برای گره j ام و گره موجود در محل i+strp انجام می دهیم.
   c. در صورتیکه با افزایش گامها به انتهای لیست i و j برسیم، به مرحله ۴ می رویم.
۳. مراحل فوق را تا رسیدن به انتهای یکی از لیست های فرزندان انجام می دهیم.
۴. باقیمانده لیست های فرزندان، را به عنوان فرزندان گره مجازی فصلی (Disjoint) در لیست فرزندان P اضافه می نماییم.

الگوریتم فوق جهت تولید الگو های پیچیده بکار می رود و لازمه استفاده از آن اعمال برخی تغییرات در روتین مقایسه تساوی الگو ها (IsEqualPattern) می باشد.

---

[96] Optional

[97] در اینجا Step برابر با مقدار گام جاری می باشد

ذیلا کد مربوط به روتین ExpandPattern، به زبان C# آورده شده است:

```
public HtmlNode ExpandPattern( HtmlNode P1, HtmlNode P2 )
{
    HtmlNode P = new HtmlNode();
    P.MinCardinality = min( P1.MinCardinality, P2.MinCardinality );
    P.MaxCardinality = min( P1.MaxCardinality, P2.MaxCardinality );

    int i = 0, j = 0;
    int Count1 = P1.Nodes.Count;
    int Count2 = P2.Nodes.Count;

    while( i < Count1  &&  j < Count2 )
        if( IsEqualPattern(P1.Nodes[i], P2.Nodes[j]) )
            P.Nodes.Add( P1.Nodes[i] );
        else
        {
            int rem1 = Count1 - i;
            int rem2 = Count2 - j;
            bool Done = false;
            for(int step=1; step<min(rem1, rem2); ++step)
                if( IsEqualPattern(P1.Nodes[i], P2.Nodes[j+step]) )
                {
                    HtmlNode tmp = MakeSimplePattern(P2.Nodes, j, j+step);
                    tmp.MinCardinality = 0;
                    P.Nodes.Add(tmp);
                    j += step;
                    Done = true;
                }
                else if( IsEqualPattern(P1.Nodes[i+step], P2.Nodes[j]) )
                {
                    HtmlNode tmp = MakeSimplePattern(P1.Nodes, i, i+step);
                    tmp.MinCardinality = 0;
                    P.Nodes.Add(tmp);
                    i += step;
                    Done = true;
                }
            if( !Done ) break;
        }

    if(i<P1.Nodes.Count || j<P2.Nodes.Count)
    {
```

```
        HtmlNode root = new HtmlNode();
        root.IsDisjointNode = true;
        if(i<Count1)
            root.Nodes.Add( MakeSimplePattern(P1.Nodes, i, Count1) );
        if(j<Count2)
            root.Nodes.Add( MakeSimplePattern(P2.Nodes, j, Count2) );
        if( root.Nodes.Count == 1 )
            root.MinCardinality = 0;
        P.Nodes.Add( root );
    }

    return P;
}
```

لیست ۶: الگوریتم بسط الگوها

۵-۳-۴-۵- تنظیم خصوصیات الگو – AdjustPatternProperties

همانطور که قبلا هم گفته شد، حتی با وجود یک آنتالوژی استخراج قوی، ممکن است برای یک گره متنی چند خصوصیت متفاوت منطبق گردد. لذا روتین AdjustPatternProperties سعی خواهد نمود تا مشکل تعدد خصوصیت ها برای یک داده متنی را بر طرف نماید. اجرای این روتین آزادانه و انتخابی می باشد. ولی باید توجه داشت که در WDML هر داده متنی تنها می تواند به یک خصوصیت مقید (bind) گردد بنابراین باید یا با اجرای این روتین یا با بهبود آنتالوژی استخراج، مشکل تعدد خصوصیت را برطرف نمود.

در اولین قدم سعی خواهیم کرد تا با استفاده از خصوصیت هایی که در یک گره تنها خصوصیت منطبق می باشند، مجموعه خصوصیت های منطبق با سایر گره ها را کاهش دهیم:

۱. تعیین لیست گره هایی که تنها دارای یک خصوصیت می باشند
۲. حذف خصوصیات فوق از مجموعه خصوصیت های مرتبط با سایر گره ها و بروز رسانی لیست گره هایی که دارای یک خصوصیت می باشند.
۳. در مرحله فوق در صورتیکه لیست گره های دارای یک خصوصیت بروز رسانی گردید، مرحله دوم مجددا اجرا می گردد.

در قدم بعدی، با توجه به آنکه این الگوریتم بعد از شناسایی الگوی رکورد فراخوانی می گردد، می توان از محدودیت های تعریف شده در آنتالوژی استخراج (به ازای یک کلاس) و نیز میزان انطباق خصوصیت استفاده نمود.

۱. به ازای هر یک از خصوصیت هایی که دارای کاردینالیتی یک بوده و در بیش از یک گره منطبق شده اند، لیست گره هایی که با آن منطبق می باشند را تهیه می کنیم.
۲. از لیست فوق آنکه بیشترین مقدار انطباق را داشته باشد، انتخاب کرده و خصوصیت مورد بحث را از لیست انطباق سایر گره ها حذف می کنیم.
۳. در صورت اجرای مرحله فوق، مراحل مربوط به قدم اول، را اجرا می کنیم.

تشخیص دقیق خصوصیات منطبق با یک گره بسیار مهم بوده و تعیین کننده دقت سیستم در شناسایی و فهم اتوماتیک محتویات صفحات وب می باشد. هر چند موارد فوق، جهت اعمال بر روی صفحات مورد تحقیق در این پروژه، بسیار موثر بوده اند، اما لازم است تا با بررسی های بیشتر بر روی نمونه های مختلفی از صفحات وب، این بخش را تکمیل تر نمود.

۵-۴- اجرای الگوریتم در یک صفحه نمونه

مهمترین بخش الگوریتم مطرح شده در این بخش، یافتن الگو ها می باشد، لـذا بـرای فهـم بهتـر ایـن الگوریتم، اجرای آن را بر روی مثال لیست کشور ها که در بخش های قبلی مطرح شـد، نـشان مـی دهـیم. جهت یاد آوری این مثال، ذیلا کد HTML آن را می آوریم:

```
01: <html>
02: <head>
03:     <title> Country Codes</title>
04: </head>
05: <body>
06:         <h3> Country Codes</h3>
07:         <B>Iran: </B><i>100</i><br />
08:         <B>Turkey: </B><i>120</i><br />
09:         <B>Japan: </B><i>130</i><br />
10:         <B>UK: </B><i>140</i><br />
11:         <hr />
12:         end of list.
13: </body>
14: </html>
```

مطابق با روتین CreateWrapper، ابتدا ساختار درختی آن را ایجاد کرده و سپس آنتـالوژی اسـتخراج را بر روی آن اعمال مـی کنـیم. بـا اعمـال آنتـالوژی اسـتخراج، بـه ازای هـر گـره متنـی، مجموعـه ای از خصوصیات منطبق با آن مشخص می گردد. برای توصیف آنتالوژی استخراج فرض مـی کنـیم کـه کـلاس CountryCodes دارای دو خصوصیت زیر می باشد:

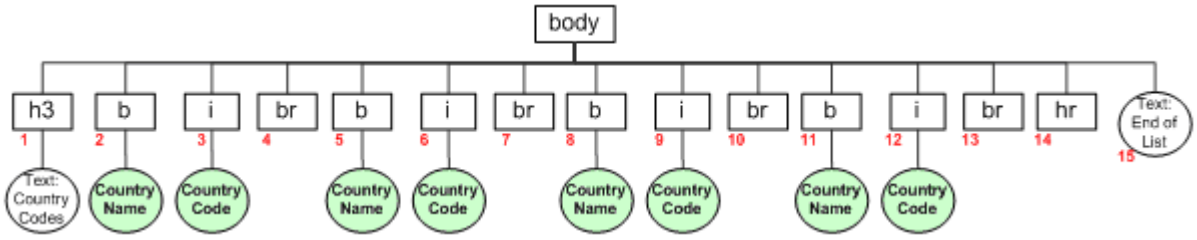
- نام کشور (CountryName) : نوع داده ای این خصوصیت، رشته ای بوده و می تواند از یـک تا ۳ کلمه باشد. مقادیر داده ای این خصوصیت، منحصرا مربوط به این خصوصیت بوده و در سایر خصوصیت ها یافت نخواهد شد. مانند کلمه Iran (بدین ترتیب هرگاه با کلمـه "ایـران" مواجه شدیم بدون هیچ پردازش اضافی مـی تـوان خـصوصیت CountryName را بـرای آن انتخاب نمود). همچنین مقادیر مرتبط با این خصوصیت دارای الگوی زیر می باشند:

شکل ۳۵: الگوی مقادیر خصوصیت CountryName

الگوی فوق نشان می دهد که تنها کلماتی که در لغتنامه مفهومی دارای سمانتیک "نام کشور" می باشند و احتمالا به سمبل : ختم می گردند می توانند پذیرفته گردند.

- کد کشور (CountryCode) : نوع داده ای این خصوصیت، عدد صحیح بوده و می تواند بین ۰ تا ۱۰۰۰ باشد. مقادیر داده ای این خصوصیت، به هیچ عنوان در خصوصیت نام کشور یافت نخواهند شد.

توصیفات فوق در قالب یک آنتالوژی استخراج قابل فهم برای ماشین ها در ضمائم آمده است. شکل زیر، ساختار درختی صفحه وب را پس از اعمال آنتالوژی استخراج بر روی آن، نشان می دهد. در این شکل دایره ها نشان دهنده گره های متنی می باشند.

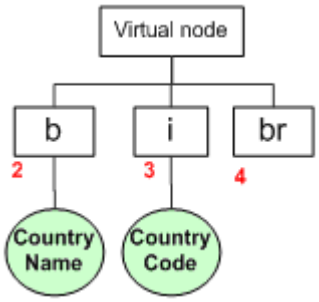


شکل ۳۶: نمایش درختی صفحه کد کشور ها، بعد از اعمال آنتالوژی استخراج

متن نوشته شده در درون دایره های سبز رنگ، نام خصوصیت منطبق با آن گره می باشد. متن نوشته شده بر روی سایر گره ها نیز نام المان مرتبط در صفحه وب را نشان می دهد. روتین CreatePattern با ارسال گره body (از ساختار درختی فوق)، فرایند ایجاد الگو ها را آغاز می کند.

در این روتین ابتدا به ازای هر یک از گره های فرزند روتین ایجاد الگو (CreatePattern) بصورت بازگشتی فراخوانی می گردد. سپس گره شماره ۲ به عنوان اولین گره داده ای[۹۸] (گره آغازین) انتخاب می شود. سپس سیستم بدنبال گره داده ای مشابه با گره آغازین می گردد و گره ۵ را می یابد (گره ۲ و گره ۵ هر دو دارای تگ b بوده و یک گره متنی از جنس نام کشور، گره فرزند آنها می باشد). سیستم با فراخوانی روتین MakeSimplePattern(nodes, 2, 5) الگوی ساده شکل زیر را ایجاد می کند.

[98] گره داده ای گره ای است که خود یا فرزندان آن منطبق با یکی از خصوصیت های تعریف شده در آنتالوژی باشد.



شکل ۳۷: الگوی ساده جهت ترکیب نام کشور ها

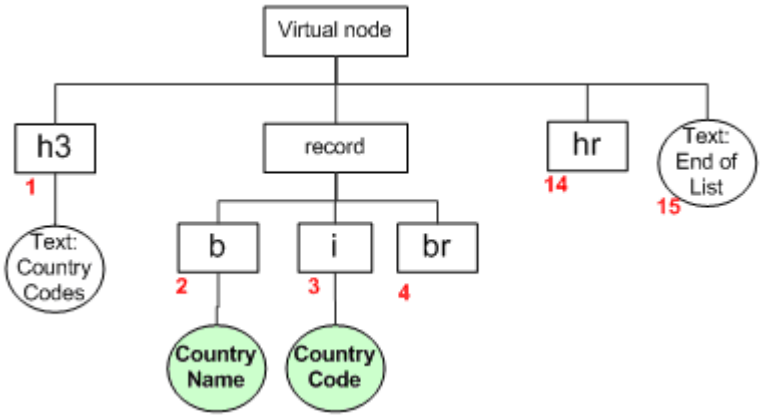
در ادامه، الگوی فوق با شرایط مطرح شده در آنتالوژی استخراج بررسی می شود تا مشخص شود که آیا این الگو می تواند نشان دهنده یک رکورد باشد یا نه؟ با توجه به تعریف کلاس CountryCodes در آنتالوژی استخراج[۹۹]، مشاهده می شود که هر یک از نمونه های این کلاس باید دقیقا دارای یک خصوصیت نام کشور و یک خصوصیت کد کشور باشند. در نتیجه الگوی بدست آمده می تواند الگوی یک رکورد از کلاس CountryCodes باشد.

اکنون با مشخص شدن الگوی رکورد، با شروع از گره پنجم، از گره های متناظر با الگوی رکورد فوق گذر می کنیم. در اولین تکرار گره های ۵ و ۶ و ۷ و در تکرار دوم گره های ۸ و ۹ و ۱۰ و در تکرار سوم گره های ۱۱ و ۱۲ و ۱۳، پردازش می گردند. تکرار پنجم در گره ۱۴ با شکست مواجه می شود. لذا سیستم بررسی می کند که آیا رکورد داده ای دیگری وجود دارد یا نه؟ از آنجا که رکورد داده ای دیگری نمی یابد سیستم، گره های فرزند ۱ تا ۱۵ را به سه بخش زیر تقسیم می کند:

- **بخش قبل از شروع رکورد ها**: این بخش تنها شامل گره اول می باشد.
- **بخش رکورد ها**: این بخش شامل گره های ۲ تا ۱۳ می باشد.
- **بخش بعد از رکورد ها**: این بخش شامل گره های ۱۴ و ۱۵ می باشد

سیستم برای هر بخش یک الگو ایجاد کرده و آنها را بصورت زیر با استفاده از یک گره مجازی با هم ترکیب می کند.

[99] رجوع شود به بخش ضمائم – آنتالوژی استخراج کد کشور ها



شکل ۳۸: الگوی نهایی تولید شده برای مثال کد کشور ها کد کشور ها

بعد از تولید الگوی فوق، متد SetHtmlBody آن را به قالب تعریف شده در زبان WDML تبدیل کرده و در ساختار داخلی Wrapper ذخیره می نماید.

۵-۵- خلاصه مطالب و نتیجه گیری

در ادامه ی معرفی سیستم پیشنهادی خود، ماژول الگو ها را بررسی کردیم. این ماژول استخراج اتوماتیک با استفاده از یک آنتالوژی استخراج ابتدا اقدام به شناسایی الگو های موجود در صفحه وب نموده و سپس آنها را در قالب فایل های WDML در خروجی می نویسد. فایل های واسط WDML اگر چه باعث ایجاد شکاف بین فرایند استخراج اطلاعات و فرایند شناسایی الگو ها می گردند ولی این شکاف در مواردی که اطلاعات موجود در یک صفحه وب، از بانک های اطلاعاتی خوانده شده و پیاپی در حال تغییر می باشد، بسیار سودمند می باشد.

فرایند شناسایی الگو های موجود در صفحه وب نسبتا پیچیده می باشد. پیشنهاد ما جهت افزایش کارایی و دقت در شناسایی الگو ها، یافتن نظم موجود در ساختار درختی صفحه وب با توجه به خصوصیات موجود در آنتالوژی دامنه می باشد. در این روش بجای یافتن نظم موجود در بین داده های متنی خام، ابتدا با اعمال آنتالوژی استخراج مشخص می کنیم هر داده متنی خام به چه خصوصیتی مرتبط می باشد. سپس در ساختار درختی به دنبال نظم موجود در بین خصوصیت های یافت شده می پردازیم. جهت یافتن نظم موجود، ابتدا در بین فرزندان یک گره اولین گره داده ای و نیز تکرار آن را می یابیم. سپس یک الگوی ساده با شروع از اولین گره داده ای تا تکرار آن ایجاد کرده و این الگو را بر روی گره های باقیمانده بررسی می کنیم. در صورت انطباق تمامی گره ها با الگوی مذکور، این الگو را به عنوان الگوی رکورد انتخاب می نماییم. ما در عمل مشاهده کردیم که اعمال آنتالوژی استخراج و یافتن خصوصیات مرتبط با یک قلم داده خام، بسیار دشوار بوده و نیاز به داشتن یک توصیف دقیق می باشد. اما از طرفی دیگر هر چه توصیف مقادیر موجود در دامنه دقیق تر باشد، سیستم محدود تر گشته و نیاز به داشتن دانش بیشتر از مقادیر موجود در دامنه افزایش می یابد.

در این خصوص پیشنهاد ما این است که برای برخی خصوصیات کلیدی آنتالوژی استخراج، توصیفات را با دقت و جزئیات فراوان وارد نموده و میزان انطباق خصوصیت و داده متنی خام را تعیین نماییم. بدین ترتیب، به محض مشخص شدن محدوده یک رکورد، با توجه به میزان انطباق، مناسب ترین خصوصیت را انتخاب می کنیم. این روش، کمک شایانی در کاستن از وابستگی شدید سیستم به آنتالوژی استخراج خواهد نمود.

۶- ارزیابی و جمع بندی مطالب

۶-۱- مقدمه

ما در این تحقیق، موضوع استخراج اطلاعات را جهت جمع آوری آنتالوژی های نمونه از میان صفحات وب مورد بررسی قرار داده ایم. سیستم های استخراج اطلاعات، سند ورودی و مدل استخراج اطلاعات را به عنوان ورودی پذیرفته و پس از پردازش سند ورودی اطلاعات بازیابی شده را در یک ساختار منظم و مشخص قالب بندی می کنند. مدل استخراج شامل قواعد و مفاهیم مورد نیاز جهت استخراج اقلام اطلاعاتی می باشد. بررسی فعالیت های مرتبط با این تحقیق نشان داد که قواعد استخراج فعلی یا مبتنی بر مکان و یا مبتنی بر آنتالوژی می باشند.

ایده اصلی ما در این تحقیق استفاده ترکیبی از روش مبتنی بر آنتالوژی و روش مبتنی بر استنتاج بوده است. بدین ترتیب که ابتدا صفحه وب ورودی خوانده شده و ساختار درختی تگ های HTML آن ایجاد می گردد، سپس با اعمال آنتالوژی استخراج بر روی آن، ساختاری درختی از مفاهیم موجود در صفحه وب فراهم می گردد. ما برای افزایش قابلیت نقل و انتقال Wrapper ها اطلاعات مورد نیاز هر یک از آنها را با استفاده از یک زبان جدید بنام WDML (زبان تعریف Wrapper ها) معرفی نمودیم. این عمل باعث افزایش قابلیت استفاده مجدد از Wrapper ها نیز خواهد شد.

همچنین در این تحقیق به تشریح سیستمی بنام OntoByOnto پرداختیم که از این روش پیشنهادی جهت استخراج اطلاعات بهره می برد. این سیستم، با هدف قرار دادن مشکلات سیستم های جاری که در بخش سوم مطرح شده است، سبب بهبود در موارد زیر خواهد شد:

۱. سرعت در استخراج اطلاعات: در روش پیشنهادی استخراج اطلاعات با تکیه بر محل قرارگیری اطلاعات صورت می گیرد که با حذف فرایند شناسایی محل قرار گیری اطلاعات به ازای هر صفحه سبب افزایش سرعت استخراج اطلاعات خواهد شد.
۲. دقت در استخراج اطلاعات: در روش پیشنهادی فرایند شناسایی اقلام اطلاعاتی با تکیه بر یک آنتالوژی استخراج صورت می گیرد که این خود باعث افزایش دقت در استخراج اتوماتیک اقلام اطلاعاتی می گردد.

۳. قابلیت نقل و انتقال Wrapper ها: یکی از ایده های مطرح شده در این تحقیق استفاده از یک زبان استاندارد و مبتنی بر XML، جهت معرفی کلیه اطلاعات مورد نیاز برای یک Wrapper می باشد. بطوریکه هم از قابلیت تعریف الگوی استخراج بصورت سلسله مراتب درختی بهره ببرد و هم از قابلیت آنتالوژی ها در تشخیص صحیح اطلاعات. ما در این تحقیق زبان WDML را معرفی نموده ایم که شامل تمام ویژگی های فوق می باشد. Wrapper های توصیف شده با این زبان، به راحتی قابل نقل و انتقال به سایر سیستم ها بوده و سبب افزایش قابلیت استفاده مجدد آنها می گردد.

۴. مقاومت در مقابل برخی تغییرات جزئی در صفحات وب: در این سیستم، گروه بندی تگ های مشابه که از نظر نحوه نمایش اطلاعات، در یک خانواده قرار دارند باعث افزایش مقاومت سیستم نسبت به تغییرات جزئی در ساختار نمایش اطلاعات می گردد.

در ادامه این فصل به ارزیابی سیستم پیشنهادی خود خواهیم پرداخت. در ابتدا شاخص های Precision و Recall را به عنوان معیار هایی جهت سنجش سیستم خود معرفی کرده و سپس با کمک یک سیستم ابتدایی، به جمع آوری نمونه های آزمایشگاهی حاصل از اجرای الگوریتم پیشنهادی می پردازیم. سپس با توجه به داده های بدست آمده و نیز شاخص های محاسبه شده، به ارزیابی سیستم می پردازیم.

۶-۲- شاخص های ارزیابی

در این بخش شاخص های ارزیابی روش پیشنهادی خود را تشریح می کنیم. مهمترین شاخص هایی که جهت ارزیابی سیستم های بازیابی و استخراج اطلاعات استفاده می شود [Eik99] عبارتند از Precision (خلوص و صحت استخراج[100]) و Recall (تمامیت استخراج[101]).

در حوزه بحث استخراج اطلاعات Recall برابر است با نسبت تعداد اطلاعات استخراج شده صحیح به تعداد کل اطلاعات قابل استخراج. همچنین Precision برابر است با نسبت اطلاعات استخراج شده صحیح به کل اطلاعات استخراج شده. در واقع Precision نشان می دهد که کاربر چه مقدار اطلاعات را دریافت کرده است و مقیاس Recall بیانگر آن است که کاربر باید به چه اندازه تلاش نماید تا به الباقی اطلاعات دسترسی پیدا کند.

---

[100] Purity of retrieval
[101] Completeness of retrieval

برای آشنایی بیشتر با این دو شاخص ابتدا لازم است تا یک دسته بندی از مجموعه جواب سیستم های استخراج اطلاعات داشته باشیم. از یک دیدگاه می توان اطلاعات موجود در صفحات وب را به دو گروه استخراج شده و استخراج نشده تقسیم کرد. اما همواره نمی توان انتظار داشت که تمام اطلاعات استخراج شده، مرتبط با موضوع مورد نظر باشد، لذا می توان اطلاعات استخراج شده را به دو دسته اطلاعات مرتبط و اطلاعات غیر مرتبط تقسیم بندی کرد. جدول زیر مجموعه جواب سیستم های استخراج اطلاعات را بر اساس گروه های فوق، دسته بندی می کند.

	مرتبط	غیر مرتبط	
استخراج شده	A True positive	B False positive	A+B کل داده های استخراج شده
استخراج نشده	C False Negative	D True Negative	C+D کل داده های استخراج نشده
	A+C کل داده های مرتبط	B+C کل داده های غیر مرتبط	

جدول ۳: دسته بندی مجموعه جواب سیستم های استخراج اطلاعات

در جدول فوق، A آن مجموعه از جواب های درست را نشان می دهد که به درستی استخراج شده اند (True Positive)، B آن دسته از جواب های درست می باشند که به اشتباه استخراج شده اند (False Positive)، C آن دسته از جواب های نادرست می باشند که به درستی استخراج نشده اند (False Negative) و در نهایت D نشان گر آندسته از جواب های درستی است که به اشتباه استخراج نشده اند (True Negative). با توجه به این جدول، Recal و Precision طبق فورمول های زیر بدست می آیند:

$$\mathrm{Re}call = \frac{A}{A+c}$$

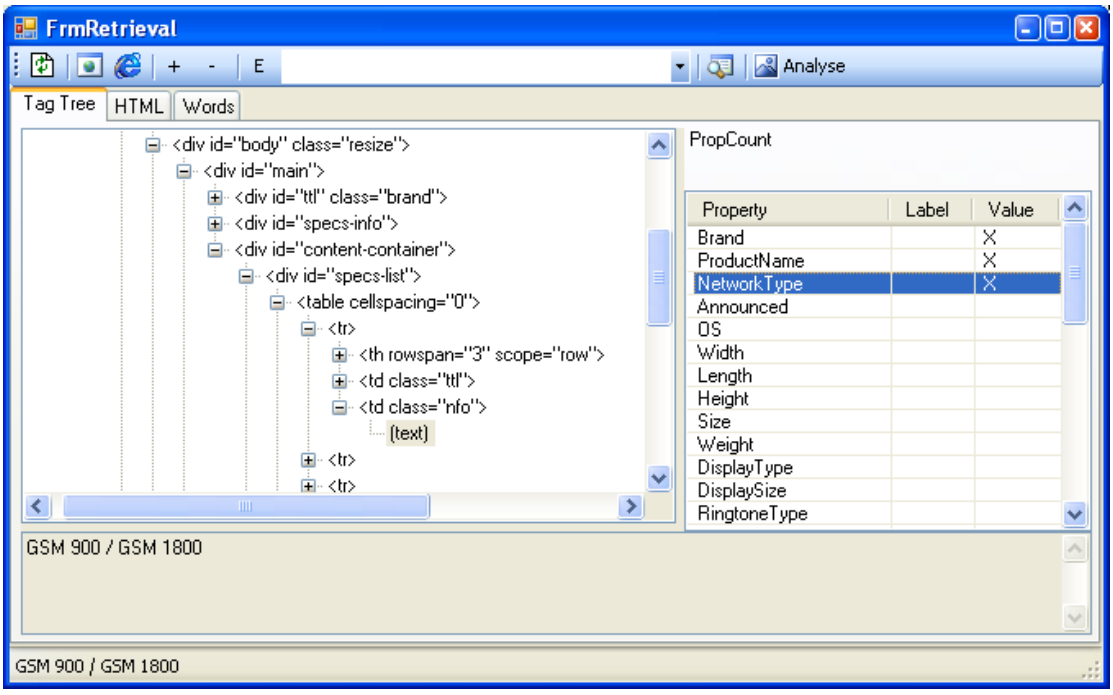
$$\mathrm{Pr}ecision = \frac{A}{A+B}$$

باید توجه داشت که جهت ارزیابی روش استخراج، هر یک از شاخص های فوق به تنهایی قابل استفاده نمی باشند. به عنوان مثال در یک سیستم استخراج اطلاعات در صورت استخراج تمام اطلاعات موجود مقدار Recall برابر با ۱۰۰٪ می گردد. از طرفی دیگر این دو شاخص با هم نسبت عکس دارند و افزایش یکی باعث کاهش دیگری خواهد شد.

۶-۳- روش و نتایج ارزیابی

جهت ارزیابی روش پیشنهادی خود، ابتدا در دامنه گوشی های تلفن همراه تحت عنوان یک مهندس دامنه به معرفی آنتالوژی استخراج مربوطه پرداختیم. مفهوم اصلی که در این آنتالوژی مورد توجه قرار، کلاس گوش های تلفن همراه بوده است. برای این کلاس ۱۲ خصوصیت (Property) شامل؛ نام گوشی، ابعاد، وزن، سیستم عامل و غیره را در نظر گرفته ایم. سپس از میان وب سایت های موجود در زمینه گوشی های تلفن همراه سه وب سایت GSMArena.com و Shop.ir و GSM.ir را انتخاب نموده و در هریک از این سایت ها ۴۰ صفحه از صفحات مرتبط با معرفی مشخصات تکنیکی گوشی های تلفن همراه و ۵ صفحه غیر مرتبط را بصورت تصادفی انتخاب نموده ایم.

ما جهت سهولت در تست و ارزیابی سیستم، دو نرم افزار جداگانه تولید کرده ایم. اول نرم افزار خزشگر که با شروع از یک لیست اولیه از آدرس صفحات وب، محتویات آنها را از اینترنت خوانده سپس ضمن ذخیره آن در یک بانک اطلاعاتی، با توجه به لینک های موجود در آن صفحه به صفحات دیگر آدرس صفحات جدید را بدست آورده و لیست اولیه خود را بروزرسانی می نماید. نرم افزار دوم، محتویات صفحات که در جداول بانک اطلاعاتی ذخیره شده است را خوانده، ابتدا ساختار درختی تگ های html را ایجاد کرده و سپس با اعمال آنتالوژی استخراج، به ازای هر یک از تگ های موجود در آن، مشخص می سازد که با کدام یک از مفاهیم موجود در آنتالوژی استخراج، مرتبط می باشد. شکل زیر یک صفحه نمونه از این نرم افزار را در حال نمایش مفهوم نوع شبکه نشان می دهد:



شکل ۳۹: ایجاد درخت تگ های Html و اعمال آنتالوژی استخراج

البته فرایند اعمال آنتالوژی استخراج در نرم افزار فوق، بطور کامل پیاده سازی نشده است و جهت استفاده صحیح از آن باید بصورت دستی سایر قواعد آنتالوژی استخراج را نیز اعمال نمود. نتیجه بررسی صفحات مورد بحث، در جدول زیر نشان داده شده است.

	ویژگی گوشی	تعداد موارد استخراج شده	تعداد موارد مرتبط	Recall	Precision
۱	برند	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۲	نام گوشی	۳۸	۳۷	٪۹۲،۵۰	٪۹۷،۳۷
۳	نوع شبکه	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۴	نوع سیستم عامل	۳۹	۳۹	٪۹۷،۵۰	٪۱۰۰،۰۰
۵	اندازه (طول در عرض)	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۶	وزن	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۷	نوع صفحه نمایش	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۸	ابعاد صفحه نمایش	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۹	نوع زنگ	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۱۰	نوع باطری	۳۸	۳۵	٪۸۷،۵۰	٪۹۲،۱۱
۱۱	حداکثر زمان مکالمه مداوم	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
۱۲	حداکثر زمان حالت انتظار	۴۰	۴۰	٪۱۰۰،۰۰	٪۱۰۰،۰۰
			میانگین	۹۸.۱۳	۹۹.۱۲

جدول ۴: نتایج حاصل از اجرای الگوریتم شناسایی اتوماتیک الگو ها

همانطور که در جدول فوق نیز مشاهده می شود، جهت آگاهی از جزئیات، نتایج ارزیابی را به تفکیک خصوصیات کلاس گوشی ها نشان داده ایم. نتایج بدست آمده از این ارزیابی نشان می دهد که عملیات استخراج بسیار رضایت بخش و قانع کننده می باشد. اما در برخی خصوصیات مانند نام گوشی، سیستم عامل و نوع باطری عملیات استخراج با اشکال مواجه شد که علت آن کمبود اطلاعات مورد نیاز در آنتالوژی استخراج می باشد. عملیات استخراج برای خصوصیات سیستم عامل و نوع باطری، با تکمیل و بهبود آنتالوژی استخراج، بطور کامل اصلاح می گردد ولی نام گوشی نیاز به ترفند های بیشتری نسبت به اطلاعات ارائه شده توسط مهندس دامنه دارد. به عنوان مثال مشاهده شده است که اغلب، نام گوشی به همراه نام شرکت سازنده آن آورده می شود. لذا با تعریف یک Pattern بنام `Brand&ProductName` این موضوع را نشان دادیم.

مهمترین نتیجه حاصل از این ارزیابی آن است که در این سیستم، آنتالوژی استخراج نقش مهمی در کیفیت استخراج اطلاعات ایفا می نماید. بطوریکه هر چه آنتالوژی استخراج دقیق تر و کاملتر باشد، اطلاعات استخراج شده نیز به همان اندازه دقیقتر خواهد بود.

۶-۴- نقاط گسترش تحقیق

جهت استخراج اطلاعات از متون زبان طبیعی گاهی اوقات لازم می شود فایل های WDML را به آنتالوژی استخراج مربوطه ارجاع داد. این وابستگی WDML به آنتالوژی استخراج، هر چند هم ضروری و یا اندک باشد ولی بخاطر درگیر شدن آن با مفاهیم آنتالوژی به هیچ وجه خوشایند نمی باشد. یکی از دلایل دوری از این وابستگی آن است که ماژول استخراج به زبان پیاده سازی آنتالوژی، وابسته می گردد. این موضوع که آیا می شود این وابستگی را کاهش داد و یا آن را حذف نمود، می تواند موضوع تحقیقات بیشتر در این زمینه باشد. به عنوان مثال جهت حذف برخی سمبل های ساده[۱۰۲]، به الگوهای NFA که حافظه و زمان بیشتری نیز مصرف می کنند، نیاز نمی باشد و شاید در سایر بخش ها نیز بتوان تسهیلاتی را جهت استقلال بیشتر WDML و یا افزایش کارایی آن، اعمال کرد که این موضوع نیاز به تحقیقات بیشتر در این زمینه دارد.

الگوریتم توصیف شده در این تحقیق، با وجود سادگی آن، مجموعه بسیار بزرگی از صفحات وب را در بر می گیرد. اما با این حال برخی صفحات وب دارای الگوی های پیچیده ای می باشند که طبیعتا به الگوریتم های پیچیده تری جهت یافتن الگو های استخراج نیاز دارند. در صورت نیاز می توان با بهبود روتین CreatePattern، این نوع صفحات پیچیده را نیز پوشش داد.

یکی دیگر از نقاط گسترش این تحقیق، دسته بندی صفحات بر اساس آدرس url آنها می باشد. تقریبا تمامی وب سایت ها بین آدرس صفحات و محتوای ارائه شده در آن صفحات، ارتباط مشخصی را ارائه می کنند. یافتن نظمی دقیق و مشخص بین آدرس صفحات و الگو های WDML، کارایی و انعطاف پذیری سیستم را به شدت افزایش می دهد. در این تحقیق هنگام توصیف روتین CreateWDML اشاره ای به این موضوع داشته ایم ولی استفاده عملی از این قابلیت پر کاربرد، نیازمند به تحقیقات بیشتر در ارتباط با این موضوع می باشد.

---

[102] مانند سمبل : که در این تحقیق از روش الگو های NFA برای حذف آن استفاده شد

۷- منابع

[ISC06] Internet Software Consortium, Last Visited: 05/05/2007, www.isc.org, Jannuary 2006

*[Lee01] Berners-Lee, T, Hendler, J & Lassila, O, 2001. "**The semantic web**", Scientific American.*

*[Dac03] Michael C. Daconta, Leo J. Obrst, Kevin T. Smith, 2003. "**The Semantic Web: A Guide to the Future of XML, Web Services and Knowledge Management**", Wiley publishing.*

*[Dav03] John Davies, Dieter Fensel, and Frank van Harmelen, editors. "**Towards the Semantic Web: Ontology-Driven Knowledge Management**". John Wiley & Sons, 2003.*

*[Bro01] Broekstra, J. and Kampman, A., 2001. "**Sesame: A generic Architecture for Storing and Querying RDF and RDF Schema**". Deliverable 10, On-To-Knowledge project, October. http://www.ontoknowledge.org/downl/del10.pdf*

*[Wes05] Wessman, A., Liddle, S.W., Embley, D.W., "**A generalized framework for an ontology-based data-extraction system**". In Proc. 4th Int. Conference on Information Systems Technology and its Applications, 239-253, 2005.*

*[Sha01] Shah, U., T. Finin, J. Mayfield. "**Information retrieval on the Semantic Web**" In Proceedings of the Eleventh International Conference on Information and Knowledge Management, McLean, Virginia, 4-9 November 2002, pp. 461-468.*

*[Sim05] Simon, K., and Lausen, G. 2005. "**ViPER: augmenting automatic information extraction with visual perceptions**". In Proc. CIKM'05, 381--388. ACM.*

*[Sno02] Snoussi H., Magnin L. and Nie J.-Y, "**Toward an Ontology-based Web Data Extraction**", The AI-2002 Workshop on Business Agents and the Semantic Web (BASeWEB) held at the AI 2002 Conference (AI-2002), Calgary, Alberta, Canada, May 26, 2002.*

*[Eng02-6] Robert Engels, 2002. "**CORPORUM-OntoExtract: Extraction of structured information from web based resources**", OnToKnowledge Project Report, Delivery 6.*

*[Eng02-7] Robert Engels, 2002. "**CORPORUM-OntoWrapper: Extraction of structured information from web based resources**", OnToKnowledge Project Report, Delivery 7.*

[Onto07] The Onto-Knowledge Toolset, Last Visited: 05/05/2007, http://www.ontoknowledge.org/tools/toolrep.shtml.

[KIM07] KIM Online reference, Last Visited: 05/05/2007, http://www.ontotext.com

*[Lae02] Laender, A., Ribeiro-Neto, B., Silva, A. and Teixeira, J. "**A Brief Survey of Web Data Extraction Tools**", in: SIGMOD Record, Volume 31, Number 2, June 2002.*

*[Bar05] Bartlett W., "**A Comparison of Techniques for Exposing Legacy Data to Semantic Web Technologies**", In Proc. 21st Annual Computer Science Conference, 2005.*

[SEA07] SESAME Online Open Source, Last Visited: 05/05/2007, http://www.openrdf.org/documentation.jsp.

*[Pop03] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, A. Kirilov, M. Goranov, "**Towards Semantic Web Information Extraction**", Human Language Technologies Workshop at the 2nd International Semantic Web Conference (ISWC2003), Florida, USA, 2003.*

*[Cun02] Cunningham H., Maynard D., Bontcheva K. and Tablan V., "**GATE: A Framework, Graphical Development Environment for Robust NLP Tools and Applications**". In Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics, 2002.*

[Luc07] Lucene - high performance full text search engine, Last Visited: 05/05/2007. http://jakarta.apache.org/lucene.

*[Fro97] Frohn, J., Himmer¨oder, R., Kandzia, P.T., Lausen, G., Schlepphorst, C., "**FLORID: A Prototype for F-Logic**", In: ICDE'97, IEEE Computer Society (1997) 583.*

*[FL07] "**How to Write F-Logic Programs**," Online. Internet. Last Visited: 05/05/2007, http://ontoprise.de/documents/tutorial_flogic.pdf*

*[Aro98] Arocena, G., Mendelzon, A., "**WebOQL: Restructuring Documents, Databases and Webs**", Proceedings of International Conference on Data Eengineering, 1998.*

*[Sah00] A. Sahuguet and F. Azavant. "**Building Intelligent Web Applications Using Lightweight Wrappers**". to appear in: Data and Knowledge Engineering, 2000.*

*[Hmn06] Thomas Hornung, Kai Simon, Georg Lausen, "**Information Gathering in a Dynamic World**". PPSWR 2006: 237-241, Principles and Practice of Semantic Web Reasoning, 4th International Workshop, PPSWR 2006, Budva, Montenegro, June 10-11, 2006, Revised Selected Papers.*

*[Chr07] Christopher D. Manning, P. Raghavan, H. Schütze."**An Introduction to Information to Information Retrieval**", Cambridge University Press, 2007.*

*[Wtn00] Ian H. Witten, Eibe Frank, "**Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations**", Morgan Kaufman Publishers, 1-55860-552-5, 2000.*

*[Ksl00] Kosala R., Blockeel H. "**Web Mining Research: A Survey**", SIGKDD Explorations, July 2000.*

*[Baz99] Baeza-Yates, R. & Ribeiro-Neto, B. "**Modern Information Retrieval**", Addison Wesley, 1999.*

*[Arp03] Arpirez J. C., Corcho O., Fernandez-Lopez M., and Gomez-Perez A. "**Webode in a nutshell**". AI Magazine, 24(3):37-47, 2003*

*[Ksh00] Kushmerick, N. 2000. "**Wrapper induction: Efficiency and expressiveness**". Artificial Intelligence J. 118(1-2):15-68 (special issue on Intelligent Internet Systems).*

*[Crs05] M. Cristani, R. Cuel, "**A Survey on Ontology Creation Methodologies**", International Journal on Semantic Web and Information Systems, Vol. 1, No. 2, 49 – 69, 2005*

*[Emb98] D.W. Embley. "**Toward semantic understanding - an approach based on information extraction ontologies**". In Proceedings of the the Fifteenth Australasian Database Conference, Dunedin, New Zealand, January 2004.*

*[Eik99] Line Eikvil. "**Information Extraction from World Wide Web - A Survey**". Technical Report 945, Norweigan Computing Center, 1999.*

*[Hau03] Y. Qi, A. Hauptmann, and T. Liu: "**Supervised classification for video shot segmentation**" in Proc. IEEE Conf. Multimedia Expo (ICME03) vol. 2, 2003, pp. 689-692.*

۸- ضمائم

۸-۱- آنتالوژی استخراج نمونه برای گوش های تلفن همراه

آنتالوژی استخراج زیر، جهت شناسایی و استخراج گوشی های تلفن همراه، هنگام ارزیابی سیستم در بیشتر از ۱۰۰ صفحه وب نمونه استفاده شده است

```
<?xml version="1.0"?>

<ontology about="E-Catalogue">


  <classes>

    <class id="SizeClass" abstract="true">
      <properties>
        <property resource="Length" cardinality="1"/>
        <property resource="Width" cardinality="1"/>
        <property resource="Height" cardinality="1"/>
      </properties>
    </class>

    <class id="DisplaySizeClass" abstract="true">
      <properties>
        <property resource="DisplayWidth" cardinality="1"/>
        <property resource="DisplayHeight" cardinality="1"/>
      </properties>
    </class>

    <class id="Mobile" abstract="false">
      <subclassof resource="Product"/>
      <labels>
        <label>موبایل</label>
        <label>گوشي</label>
        <label>mobile</label>
        <label>cell phone</label>
      </labels>+
      <instancename property="ProductName"/>
      <properties>
        <property resource="Brand" maxcardinality="1"/>
        <property resource="ProductName" cardinality="1"/>
        <property resource="NetworkType" />
        <property resource="OS" maxcardinality="1"/>
        <property resource="Size" cardinality="1" />
        <property resource="Weight" cardinality="1" />
        <property resource="DisplayType" />
        <property resource="DisplaySize" />
        <property resource="RingtoneType" />
        <property resource="BatteryStandbyTime" />
```

```
        <property resource="BatteryTalkTime" />
        <property resource="BatteryType" />
      </properties>
    </class>

  </classes>



  <properties>

    <property id="Brand" datatype="string" maxwordscount="3">
      <labels>
        <label>brand</label>
        <label>برند</label>
        <label>مارك</label>
      </labels>
      <disjointwith>
        <property resource="" />
      </disjointwith>
      <pattern resource="Brand"/>
    </property>

    <property id="ProductName" datatype="string" maxwordscount="4">
      <labels>
        <label>phone name</label>
        <label>name</label>
        <label>گوشي نام</label>
        <label>نام</label>
      </labels>
      <disjointwith>
        <property resource="" />
      </disjointwith>
      <pattern resource="Brand&ProductName"/>
    </property>

    <property id="NetworkType" datatype="">
      <labels>
        <label>network</label>
        <label>band</label>
        <label>فركانس باند</label>
        <label>باند</label>
        <label>شبكه</label>
      </labels>
      <disjointwith>
        <property resource="" />
      </disjointwith>
      <pattern resource="NetworkType"/>
    </property>

    <property id="OS" datatype="">
      <labels>
        <label>عامل سيستم</label>
        <label>operating system</label>
        <label>os</label>
      </labels>
      <disjointwith>
        <property resource="" />
      </disjointwith>
      <pattern resource="OS" />
    </property>

    <property id="Width" datatype="int" minvalue="50" maxvalue="250"
abstract="true">
```

```
      <labels>
        <label>Width</label>
        <label>عرض</label>
      </labels>
    </property>

    <property id="Length" datatype="int" minvalue="60" maxvalue="600"
abstract="true">
      <labels>
        <label>Length</label>
        <label>طول</label>
      </labels>
    </property>

    <property id="Height" datatype="int" minvalue="5" maxvalue="40"
abstract="true">
      <labels>
        <label>Height</label>
        <label>ارتفاع</label>
        <label>عمق</label>
        <label>ضخامت</label>
      </labels>
    </property>

    <property id="DisplayWidth" datatype="int" minvalue="50"
maxvalue="500" abstract="true">
      <labels>
        <label>Display Width</label>
        <label>Width</label>
        <label>صفحه عرض</label>
        <label>عرض</label>
      </labels>
    </property>

    <property id="DisplayHeight" datatype="int" minvalue="50"
maxvalue="500" abstract="true">
      <labels>
        <label>Display Height</label>
        <label>Height</label>
        <label>صفحه ارتفاع</label>
        <label>ارتفاع</label>
      </labels>
    </property>

    <property id="Size" datatype="SizeClass">
      <labels>
        <label>Phone Size</label>
        <label>Size</label>
        <label>ابعاد</label>
        <label>سایز</label>
      </labels>
      <pattern resource="Size"/>
    </property>

    <property id="Weight" datatype="int" minvalue="40" maxvalue="600">
      <labels>
        <label>Phone Weight</label>
        <label>Weight</label>
        <label>گوشي وزن</label>
        <label>وزن</label>
      </labels>
      <pattern resource="Weight"/>
    </property>
```

```
    <property id="DisplayType" datatype="">
      <labels>
        <label>Display Type</label>
        <label>Type</label>
        <label>نمایش صفحه نوع</label>
        <label>نوع</label>
      </labels>
      <regularexpressions>
        <pattern>.*(CSTN|STN|TFT|TFD|OLED).*</pattern>
      </regularexpressions>
      <disjointwith>
        <property resource="" />
      </disjointwith>
    </property>

    <property id="DisplaySize" datatype="DisplaySizeClass">
      <labels>
        <label>Display Size</label>
        <label>Size</label>
        <label>ابعاد</label>
        <label>سایز</label>
      </labels>
      <pattern resource="DisplaySize"/>
    </property>

    <property id="RingtoneType">
      <labels>
        <label>Ringtone Type</label>
        <label>Type</label>
        <label>نوع</label>
        <label>زنگ نوع</label>
      </labels>
      <regularexpressions>

<pattern>.*(channel|Polyphonic|Monophonic|Tone|كانال|فونيك).*</pattern>
      </regularexpressions>
      <disjointwith>
        <property resource="" />
      </disjointwith>
    </property>

    <property id="BatteryStandbyTime" datatype="float" minvalue="50"
maxvalue="5000">
      <labels>
        <label>Stand by</label>
        <label>Stand by time</label>
        <label>Battery Talk Time</label>
        <label>کار به آماده زمان</label>
        <label>انتظار حالت</label>
        <label>انتظار زمان</label>
      </labels>
      <pattern resource="BatteryTime"/>
    </property>

    <property id="BatteryTalkTime" datatype="float" minvalue="1"
maxvalue="500">
      <labels>
        <label>Talk</label>
        <label>Talk Time</label>
        <label>Battery Talk Time</label>
        <label>مکالمه حالت</label>
        <label>مکالمه</label>
        <label>مداوم مکالمه زمان</label>
        <label>مکالمه زمان</label>
```

```
    </labels>
    <pattern resource="BatteryTime"/>
  </property>

  <property id="BatteryType" >
    <labels>
      <label>Battery</label>
      <label>Battery Type</label>
      <label>Type</label>
      <label>باتري</label>
      <label>باتري نوع</label>
      <label>نوع</label>
    </labels>
    <disjointwith>
      <property resource="" />
    </disjointwith>
  </property>

</properties>



<patterns>

  <pattern id="Brand">
    <NFA removesymboles="true">
      <node id="start" >
        <edge semantic="Company Name" />
      </node>
    </NFA>
  </pattern>

  <pattern id="Brand&ProductName">
    <NFA removesymboles="true">
      <node id="start" >
        <edge propertyvalue="Brand" target="S1" />
        <edge target="S1" />
      </node>
      <node id="S1">
        <edge maxwordscount="3"  value="#"/>
      </node>
    </NFA>
  </pattern>

  <pattern id="NetworkType">
    <NFA removesymboles="true">
      <node id="start" >
        <edge word="UMTS" value="UMTS"  />
        <edge word="HSDPA" value="HSDPA"/>
        <edge word="GSM" target="freq"/>
        <edge target="freq"/>
      </node>
      <node id="freq">
        <edge word="1900" value="GSM 1900" target="next" />
        <edge word="1800" value="GSM 1800" target="next" />
        <edge word="900" value="GSM 900" target="next" />
        <edge word="850" value="GSM 850" target="next" />
      </node>
      <node id="next">
        <edge target="next" />
        <edge />
      </node>
    </NFA>
  </pattern>
```

```
<pattern id="OS">
  <NFA removesymboles="true">
    <node id="start">
      <edge word="windows" value="#" target="mobile" />
      <edge word="symbian" value="#" target="ver" />
    </node>
    <node id="mobile">
      <edge word="mobile" value="#" target="ver"/>
    </node>
    <node id="ver">
      <edge value="#" />
    </node>
  </NFA>
</pattern>

<pattern id="Size">
  <NFA>
    <node id="start">
      <edge propertylabel="Length" target="LxWxH" />
      <edge target="LxWxH" />
    </node>
    <node id="LxWxH">
      <edge propertyvalue="Length" target="xWxH_Value" />
    </node>
    <node id="xWxH_Value">
      <edge word="*" target="WxH_Label" />
      <edge word="x" target="WxH_Label" />
      <edge word="در" target="WxH_Label" />
      <edge target="WxH_Label" />
    </node>
    <node id="WxH_Label">
      <edge propertylabel="Width" target="WxH" />
      <edge target="WxH" />
    </node>
    <node id="WxH">
      <edge propertyvalue="Width" target="xH" />
    </node>
    <node id="xH">
      <edge word="*" target="H_Label" />
      <edge word="x" target="H_Label" />
      <edge word="در" target="H_Label" />
      <edge target="H_Label" />
    </node>
    <node id="H_Label">
      <edge propertylabel="Height" target="H" />
      <edge target="H" />
    </node>
    <node id="H">
      <edge propertyvalue="Height" />
    </node>
  </NFA>
</pattern>

<pattern id="Weight">
  <NFA>
    <node id="start">
      <edge propertyvalue="Weight" target="unit"/>
    </node>
    <node id="unit">
      <edge word="gram"/>
      <edge word="gr"/>
      <edge word="g"/>
      <edge word="گرم"/>
```

```
        <edge />
      </node>
    </NFA>
  </pattern>

  <pattern id="DisplaySize">
    <NFA>
      <node id="start">
        <edge propertylabel="Width" target="WxL" />
        <edge propertylabel="Length" target="WxL" />
        <edge target="WxL" />
      </node>
      <node id="WxL">
        <edge propertyvalue="Width" target="xL" />
      </node>
      <node id="xL">
        <edge word="*" target="L_Label" />
        <edge word="x" target="L_Label" />
        <edge word="در" target="L_Label" />
        <edge target="L_Label" />
      </node>
      <node id="L_Label">
        <edge propertylabel="Length" target="L" />
        <edge target="L" />
      </node>
      <node id="L">
        <edge propertyvalue="Length" target="Pixels"/>
      </node>
      <node id="Pixels">
        <edge word="pixels" />
        <edge word="pixel" />
        <edge word="پیکسل" />
      </node>
    </NFA>
  </pattern>

  <pattern id="BatteryTime">
    <NFA>
      <node id="start">
        <edge word="up" target="ToHour"/>
        <edge word="حداکثر" target="Hour"/>
        <edge target="Hour"/>
      </node>
      <node id="ToHour">
        <edge word="to" target="Hour"/>
        <edge />
      </node>
      <node id="Hour">
        <edge propertyvalue=""  target="Saat"/>
        <edge />
      </node>
      <node id="Saat">
        <edge word="ساعت"/>
        <edge word="hour"/>
        <edge />
      </node>
    </NFA>
  </pattern>

</patterns>
```

```
<vocabulary>
  <property resource="Brand">
    <domains>
      <domain>Mobile</domain>
    </domains>
    <words>
      <word means="">Nokia</word>
      <word means="">Samsung</word>
      <word means="">Motorola</word>
      <word means="">Sony Ericsson</word>
      <word means="">LG</word>
      <word means="">Alcatel</word>
      <word means="">Panasonic</word>
      <word means="">BenQ Simense</word>
      <word>Simense</word>
      <word>BenQ</word>
      <word>Philips</word>
      <word>Sagem</word>
      <word>Sharp</word>
      <word>Toshiba</word>
      <word>Pantech</word>
      <word>Palm</word>
      <word>HTC</word>
      <word>Qtek</word>
      <word>i-mate</word>
      <word>BlackBerry</word>
      <word>Haier</word>
      <word>Eten</word>
      <word>HP</word>
      <word>Asus</word>
    </words>
  </property>
</vocabulary>


  </ontology>
```

ISLAMIC AZAD UNIVERSITY

SCIENCE AND RESEARCH BRANCH

FACULTY OF GRADUATE STUDIES

DEPARTMENT OF COMPUTER ENGINEERING

"M.Sc" THESIS

SUBJECT :

Information Retrievel for Semantic Web

THESIS ADVISOR:

Freidon Shams Ph.D

CONSULTING ADVISOR:

Mehrnoush Shamsfard Ph.D

BY:

Mahdi Talebian

DATE:

(January 2008)