دانشگاه آزاد اسلامی

واحد نجف آباد

دانشکده تحصیلات تکمیلی - گروه کامپیوتر

پایان نامه برای دریافت درجه کارشناسی ارشد *M.Sc.*

گرایش نرم افزار

عنوان

داده کاوی با سرویسهای وب با استفاده از الگوریتمهای وب معنایی

استاد راهنما : دکتر ناصر قاسم آقایی

استاد مشاور : دکتر احمدرضا نقش نیلچی

نگارش

عبدالرضا رسولی کناری

تابستان ۸۵

ای خدا ای فضل تو حاجت روا
با تو یاد هیچ کس نبود روا
قطره دانش که بخشیدی ز پیش
متصل گردان به دریاهای خویش

اللهم عجل لولیک الفرج

اللهم عجل لولیک الفرج

تقدیم به پدر، مادر و همسر عزیزم

چکیده

آنچه مسلم است داده کاوی به عنوان مبحثی نو و مهم نقش بسزایی را در تحلیل اطلاعات و تصمیم گیریهای تجاری مهم سازمانها، شرکتها و افراد بازی میکند. اما پیاده سازیهای پیچیده و خاص ارایه کنندگان آن و نیاز به داشتن دانش خاص برای استفاده از آنها این امکان را از افراد جامعه برای استفاده از آن گرفته است، به نحوی که کمتر کسی، حتی آنان که با کامپیوتر آشنایی دارند داده کاوی را اصطلاحی نامفهوم برمیشمارند. با توجه به رشد روزافزون تجارت الکترونیک و اینترنت، ارایه این ابزار در وب میتواند تصمیم گیریهای تجارتهای کوچک و بزرگ را وارد عرصه ای نوین نماید. به نظر میاید سرویسهای وب بستر مناسبی برای اجرای ابزارهای داده کاوی در اینترنت باشد. این پایان نامه کوششی است برای پیاده سازی ابزارهای داده کاوی توسط سرویسهای وب، به نحوی که براحتی توسط کاربران قابل استفاده باشد.

فهرست مطالب

فهرست شکل ها

فهرست جداول

Introduction and General Concepts

- مقدمه
- طرح موضوع تحقیق
- اهداف تحقیق
- اهمیت موضوع و ضرورت آن
- کاربرد نتایج تحقیق
- محدودیت های تحقیق

فصل اول
مقدمه و کلیات طرح

در این فصل مقدمه مختصری از تحقیق ارایه خواهیم کرد و بطور اجمالی نگاهی به فصول آتی خواهیم داشت. در ادامه پرسش اصلی و موضوع تحقیق را که پایان نامه بر اساس آن شکل گرفته شده طرح و به اهمیت آن اشاره خواهیم کرد. سپس حوزه هایی را که در آن طرح قابل استفاده میباشد معرفی مینماییم. در انتها نیز با عنوان محدودیت ها مشکلات و موانع بر سر راه تحقیق را ذکر خواهیم نمود...

۱-۱. مقدمه

رشد روزافزون و انفجاری داده‌ها در عصر حاضر، پایگاه‌های داده را به عنوان جز لاینفکی در همه‌ی زمینه‌های کامپیوتر قرار داده است. اما با این سیل عظیم اطلاعات و نیازهای گسترده‌ی امروزی تنها نمی‌توان به اطلاعات بازیابی شونده‌ای از بانک‌های اطلاعاتی که تنها یک کپی از اطلاعات ذخیره شده در پایگاه داده هستند، دل، خوش کرد، بلکه باید راه‌هایی برای استخراج دانش موجود در این داده‌ها پیدا کرد.

از هنگامی که رایانه در تحلیل و ذخیره سازی داده ها بکار رفت (۱۹۵۰) پس از حدود ۲۰ سال، حجم داده ها در پایگاه داده ها دو برابر شد. ولی پس از گذشت دو دهه و همزمان با پیشرفت فن آوری اطلاعات هر دو سال یکبار حجم داده ها، دو برابر شد. همچنین تعداد پایگاه داده ها با سرعت بیشتری رشد نمود. این در حالی است که تعداد متخصصین تحلیل داده ها و آمارشناسان با این سرعت رشد نکرد. حتی اگر چنین امری اتفاق می افتاد، بسیاری از پایگاه داده ها چنان گسترش یافته اند که شامل چندصد میلیون یا چندصد میلیارد رکورد ثبت شده هستند و امکان تحلیل و استخراج اطلاعات با روش های معمول آماری از دل انبوه داده ها مستلزم چند روز کار با رایانه های موجود است. حال با وجود سیستم های یکپارچه اطلاعاتی، سیستم های یکپارچه بانکی و تجارت الکترونیک، لحظه به لحظه به حجم داده ها در پایگاه داده های مربوط اضافه شده و باعث به وجود آمدن انبارهای عظیمی از داده ها شده است به طوری که ضرورت کشف و استخراج سریع و دقیق دانش از این پایگاه داده ها را بیش از پیش نمایان کرده است (چنان که در عصر حاضر گفته می شود "اطلاعات طلاست").

به این منظور پروسه‌ی کشف دانش از پایگاه داده مطرح شد که یک پروسه‌ی علمی برای شناسایی الگوهای معتبر، نوین، بالقوه مفید و قابل فهم از داده‌ها می‌باشد. مهم‌ترین بخش این پروسه، کاوش داده‌ها می‌باشد که با استفاده از الگوریتم‌های مشخصی یک سری الگوها را از پایگاه داده استخراج می کند.

هم اکنون در هر کشور، سازمان ها، شرکت ها و . . . برای امور بازرگانی، پرسنلی، آموزشی، آماری و . . . پایگاه داده ها ایجاد یا خریداری شده است، به طوری که این پایگاه داده ها برای مدیران، برنامه ریزان، پژوهشگران و . . . جهت تصمیم گیری های راهبردی، تهیه گزارش های مختلف، توصیف وضعیت جاری خود و . . . می تواند مفید باشد. داده کاوی[1] یا استخراج و کشف سریع و دقیق اطلاعات با ارزش و پنهان از این پایگاه داده ها از جمله اموری است که هر کشور، سازمان و شرکتی به منظور توسعه علمی، فنی و اقتصادی خود به آن نیاز دارد.

در کشور ما نیز سازمان ها، شرکت ها و مؤسسات دولتی و خصوصی به طور فزاینده ولی آهسته در حال ایجاد یا خرید نرم افزارهای پایگاه داده ها و مکانیزه کردن سیستم های اطلاعات خود هستند، همچنین با توجه به فصول دهم و یازدهم قانون برنامه سوم توسعه در خصوص داد و ستدهای الکترونیکی و همچنین تأکید بر برخورداری کشور از فن آوری های جدید اطلاعات برای دستیابی آسان به اطلاعات داخلی و خارجی، دولت مکلف شده است امکانات لازم برای دستیابی آسان به اطلاعات، زمینه سازی برای اتصال کشور به شبکه های جهانی و ایجاد زیر ساخت های ارتباطی و شاهراه های اطلاعاتی فراهم کند. واضح است این امر باعث ایجاد پایگاه های عظیم داده ها شده و ضرورت استفاده از داده کاوی را بیش از پیش نمایان می سازد.

از سوی دیگر، کسانی که با صنعت *IT* آشنایی دارند حتماً نام وب سرویس را شنیده اند .برای مثال، بیش از ۶۶ درصد کسانی که در نظر سنجی مجله *InfoWorld* شرکت کرده بودند بر این توافق داشتند که وب سرویس ها مدل تجاری بعدی اینترنت خواهند بود. به علاوه گروه گارتنر پیش بینی کرده است که وب سرویس ها کارآیی پروژه های *IT* را تا ۳۰ در صد بالا می برد. اما وب سرویس چیست و چگونه شکل تجارت را در اینترنت تغییر خواهد داد؟

---

[1] - Data mining

برای ساده کردن پردازش های تجاری، برنامه های غیرمتمرکز باید با یکدیگر ارتباط داشته باشند و از داده های اشتراکی یکدیگر استفاده کنند. قبلاً این کار بوسیله ابداع استانداردهای خصوصی و فرمت داده ها به شکل مورد نیاز هر برنامه انجام می شد. اما دنیای وب و *XML* تکنولوژی آزاد برای انتقال داده انتقال اطلاعات بین سیستم ها را افزایش داد. وب سرویس ها نرم افزارهایی هستند که از *XML*برای انتقال اطلاعات بین نرم افزارهای دیگر از طریق پروتکل های معمول اینترنتی استفاده می کنند. به شکل ساده یک وب سرویس از طریق وب اعمالی را انجام می دهد و نتایج را به برنامه دیگری می فرستد. این یعنی برنامه ای که در یک کامپیوتر در حال اجراست اطلاعاتی را به کامپیوتردیگری می فرستد و از آن درخواست جواب می کند. برنامه ای که در آن کامپیوتر دوم است کارهای خواسته شده را انجام می دهد و نتیجه را بر روی ساختارهای اینترنتی به برنامه اول برمی گرداند.

پیشرفت های جدید در زمینه وب سرویس ها باعث شده است تا از آن به عنوان امکانی درجهت کاربردهای انتشار ، فراخوانی و مکان یابی در وب استفاده گردد . به همین دلیل است که امروزه بیشتر سازمانها وشرکتها اساس کسب وکار و سایر سرویس های کاربردی خود را بر روی اینترنت پیاده سازی می نمایند . بنابراین توانایی انتخاب و یکپارچه سازی مفید سرویس های درون سازمانی برروی وب و در زمان اجرا ، مبحث مهم تهیه وب سرویس ها را بوجود آورد.

اما آنچه متاسفانه امروزه دیده میشود جدایی این دو مقوله مهم از یکدیگر میباشد. پیچیدگی و تخصصی بودن ابزارهای داده کاوی تا حدی است که کمتر کسی از آن با خبر است. به نظر میاید با توجه به اقبال روزافزون افراد و سازمانها برای استفاده از اینترنت بخصوص در کشور ما، ترکیب این دو مقوله و پیاده سازی ابزارهای داده کاوی با سرویسهای وب این امکان را به سازمانها و افراد میدهد تا اطلاعات خود را به طور موشکافانه ای مورد تحلیل و بررسی قرار داده و تصمیم گیریهای بهتری را در عرصه تجارت شاهد باشیم.

این پایان نامه قصد دارد با بررسی جوانب کار و ارایه راهکارهایی برای حل مشکلات از سرویسهای وب به عنوان بستر پیاده سازی مناسبی برای ابزارهای داده کاوی استفاده و این ایده نو را تحلیل و پیاده سازی نماید. ان شاء ا...

در ادامه این بخش موضوع تحقیق را به عنوان یک پرسش اساسی مطرح و اهداف، اهمیت، کاربرد، موانع و محدودیتهای آن را بیان خواهیم نمود. در فصل دوم با عنوان داده کاوی به بررسی دقیقتر داده کاوی و پیشینه آن میپردازیم. در فصول بعدی دو ابزار مهم داده کاوی – که مورد استفاده نویسنده در پیاده سازی تحقیق قرار گرفته است – را معرفی مینماییم. در این راستا در فصل سوم درخت تصمیم و در فصل چهارم سیستم پشتیبانی از تصمیم را موشکافانه تحلیل مینماییم. بخش بعدی پایان نامه مربوط به بررسی و تعریف سرویسهای وب میباشد. در فصل پنجم سرویسهای وب را تعریف میکنیم و پیشینه آنرا مورد بررسی قرار میدهیم. در ادامه فصل به تعریف زبان تعریف سرویسهای وب[2] و تکنولوژی توصیف، کشف، مجتمع سازی منابع[3] میپردازیم. در بخش نهایی پایان نامه به تشریح مسایل و نکات مربوط به پیاده سازی عملی کار میپردازیم. فصل ششم اختصاص به ارایه دلایل انتخاب ترکیب وب سرویس و ابزارهای داده کاوی دارد. فصل هفتم نیز به بررسی مسایل و مشکلات مربوط به پیاده سازی و ارایه راه حلهای مناسب اختصاص دارد. اکتیوایکس به عنوان تکنولوژی انتخاب شده برای پیاده سازی سمت مشتری وب سرویس و در سمت دیگر *CGI* برای سمت کارگزار و همچنین انتخاب پروتکل دسترسی ساده شی[4] برای ارتباط این دو و تبادل اطلاعات مورد بررسی قرار میگیرد. در فصل هشتم نیز الگوریتم ساخت درخت تصمیم و پیاده سازی آن با سرویسهای وب مورد بررسی قرار خواهد گرفت.

امیدوارم که این پایان نامه مورد استفاده خوانندگان و علاقه مندان به این زمینه قرار بگیرد. ان شاء ا...

---

[2] Web Service Definition Language (WSDL)
3 Universal Description , Discovery , and Integration (UDDI)
[4] Simple Object Access Protocol (SOAP)

۱-۲. طرح موضوع تحقیق

داده کاوی به معنای کشف اطلاعات ناشناخته در داده ها است. تاثیر مهم داده کاوی تا آنجاست که امروزه در تمام شاخه های علوم کامپیوتر از پایگاه داده تا هوش مصنوعی و شبکه های عصبی و ... ردپای آن و تاثیر بسزای آن را میتوان یافت، تا حدی که یکی از ارکان تصمیم گیریهای بزرگ تجاری داده کاوی اطلاعات در اختیار سازمان میباشد. در بیشتر پیاده سازیها آنچه مشهود است گران بودن استفاده، پیچیده بودن واسط مربوطه و شیوه کار با آنهاست که داده کاوی را به امری تخصصی تبدیل میکند به نحوی که کمتر کسی از مزایا و امکانات این ابزار بالقوه باخبر است.

از سوی دیگر، امروزه اینترنت جزء لاینفک زندگی بشر گشته است، به نحوی که آشنایی با آن به عنوان یکی از معیارهای سواد جامعه محسوب میشود. در بسیاری از کشورها سرویسهای وب کلیه کارهای افراد از امور ساده تا تجارتهای بزرگ را انجام میدهند. با توجه به این، به نظر میرسد سرویسهای وب میتوانند به عنوان بستر مناسبی برای پیاده سازی ابزارهای داده کاوی مورد استفاده قرار بگیرند.

پرسش اصلی این تحقیق این است که آیا میتوان ابزارهای داده کاوی را بطور موثری توسط سرویسهای وب پیاده سازی نمود بطوریکه افراد و سازمانها بتوانند از آن برای تجارت استفاده کنند ؟

۱-۳. اهداف تحقیق

هدف پیاده سازی ابزارهای داده کاوی با استفاده از سرویسهای وب با طراحی واسطهای ساده و کارامد برای استفاده کاربران اینترنت میباشد.

این امر باعث خواهد شد سازمانها و تجارتهای کوچک بتوانند در تصمیم گیریهای خود از داده کاوی بهره گیرند و تجارت الکترونیک که شاید تنها تجارت بزرگ آینده خواهد بود، وارد عرصه نوینی از ایده های نو گردد.

۱-۴. اهمیت موضوع و ضرورت آن

با توجه به موارد زیر میتوان به اهمیت موضوع پی برد :

- حجم بالای اطلاعات موجود در سازمانها که تحلیل دستی داده ها را عملا غیر ممکن میسازد
- جدا بودن مقوله های تحلیل و مدیریت بطوریکه نتایج تحلیل باید به شکل قابل درکی برای مدیران سازمان به تصویر درایند
- رشد روزافزون و دور از تصور اینترنت و تجارت الکترونیک
- وب به عنوان تنها منبع دانش و نیاز سازمانها
- ارایه قابلیتهای تجارتی سازمان به صورت وب سرویس به عنوان تنها شکل آینده تجارت

دو مورد اول لزوم توجه سازمانها به ابزارهای داده کاوی و موارد بعدی اهمیت حضور این ابزار و بازار گسترده آنها در وب را بیان میکند.

۱-۵. کاربرد نتایج تحقیق

نتایج تحقیق برای مراکز پژوهشی و شرکتهای فعال در زمینه وب و پایگاه داده برای پیاده سازی ابزارهای داده کاوی تحت وب میتواند مفید باشد. همچنین بسیاری از تحلیلگران و مدیران سازمانها میتوانند براحتی از نتایج تحقیق و پیاده سازیهای مربوطه استفاده نمایند. بسیاری از شرکتهای فعال در زمینه جمع آوری اطلاعات و دانشهای مفید میتوانند با برپایی چنین سرویسهایی و جمع آوری نتایج مفیدی

که سازمانها از سرویسهای مورد نظر دریافت میکنند، دست به آمارگیری و تحلیل زده و اطلاعات سودآور و ارزشمند تجاری را از آنها استخراج نمایند.

۱-۶. محدودیت های تحقیق

مشکل اصلی و اولیه تحقیق نو بودن ایده است و اینکه تا کنون کارهایی که در این زمینه انجام شده بسیار ناچیز است و استانداردهای مناسب در این زمینه طراحی نشده است. همچنین این ترکیب مورد آزمایش قرار نگرفته تا معایب آن شناخته شود و چون مورد پیاده سازی وسیعی قرار نگرفته مسایل و مشکلات آن مشخص نگردیده است.

یکی دیگر از مشکلات عمده وسعت عملیاتی طرح برای پیاده ساز میباشد که باید از داده کاوی و الگوریتمهای مختلف موجود برای ابزارهای آن مطلع باشد. همچنین باید دید و تجربه خوبی از کار با پایگاه داده به عنوان منبع ابزارها و الگوریتمهای داده کاوی داشته باشد و از سوی دیگر باید با برنامه نویسی سرویسهای وب و پروتکلهای مختلف ازتباطی آن آشنا باشد. باید توجه داشت که هر کدام از موارد بالا برای پیاده سازی لازم به شمار میاید در حالیکه هرکدام یک رشته تخصصی است و دارای وسعت بسیار میباشد.

The secret of success is to know something that nobody else knows. (Aristotle Onassis)

داده کاوی و ابزارهای آن

Data Mining and Tools

راز موفقیت در دانستن چیزی است که هیچ کس دیگری آن را نمیداند. ارسطو

Data Mining

فصل دوم
داده کاوی

در این فصل مقدمه مختصری از داده کاوی ارایه خواهیم کرد و بطور اجمالی نگاهی به پیشینه آن خواهیم داشت. در ادامه داده کاوی را تعریف و قوانین وابسته سازی را که اساس بسیاری از الگوریتمهای داده کاوی است شرح و سعی خواهیم کرد آنرا در غالب دستورات *SQL* بنویسیم...

- مقدمه
- پیشینه داده کاوی
- پروسه‌ی کشف دانش از پایگاه داده
- قوانین وابسته سازی
- پیاده سازی با *SQL*
- خلاصه

۲-۱. مقدمه

ابزارهای داده کاوی با جستجوی حجم عظیم داده های ما می توانند تکه طلای کوچکی را که در گوشه ای پنهان شده بیابند. بازگشت هزینه صرف شده در این ابزارها غالبا بسیار سریع است. مثلا در بررسی داده های یک واحد از یک فروشگاه متوجه شدند که میزان سرقت حین فروش از باتریها و فیلمها و قلم های با قیمت متوسط ماهانه حدود ۶۰۰۰۰ دلار برای فروشگاه هزینه داشته است که به این ترتیب با جابجاکردن اقلام و قرار دادن در قسمتهای با دید بهتر سالانه حدود ۷۰۰۰۰۰ دلار صرفه جویی بدنبال داشته است.

ابزارهای داده کاوی بدنبال طرحها و گروه بندی هایی در داده ها می گردد که ممکن است از دید ما پنهان مانده باشد. این ابزار فرض می کند که شما خود نیز دقیقا نمی دانید که چه می خواهید. بیشتر این ابزار ها از روش های جستجوی زیر استفاده میکنند :

۱. ارتباطات که اصطلاحا تحلیل سبد بازار خوانده می شود. ابزار بدنبال اثبات این موضوع است که وجود چیزی بمعنی وجود چیز دیگریست. مثلا بیشتر خریداران لوازم غواصی به تعطیلات تابستانی در استرالیا می روند. یا مصرف کننده یک کالای مشخص مصرف کرده خریدار کا لای دیگری نیز هست .

۲. ارتباطات متوالی ابزار بدنبال روابط متوالی بین موضوعات می گردد مثلا وقتی قیمت طلا ۱۰ درصد بالا می رود یک هفته بعد قیمت سهام ۱۵ درصد پایین می آید .

۳. دسته بندی بدنبال دسته بندی و طبقه بندی سطح بالای اطلاعات هستند. مثلا ۷۰ درصد رای دهندگانی که تصمیم نگرفته اند به که رای دهند درآمدی بالای ۶۰۰۰۰ دلار دارند، بین ٤٠ تا ٥٠ سال سن دارند و در منطقه X اقامت دارند.

این ابزار ها در زمینه های مختلف کاربرد یافته اند.از جمله محققین بهداشت برای کشف میزان موفقیت جراحیها . بانکها برای ارزیابی اعتبار مشتریان، بورس بازان برای تشخیص جابجایی قیمتهای سهام و تشخیص طرحهای تجاری ، شرکتهای بیمه برای تشخیص ریسک مشتریان و رفتارهایشان و هتل ها برای تشخیص مشتریان بازگشتی خود از آن استفاده میکنند. همانطوریکه بنظر می آید ابزارهای داده کاوی از مجموعه ابزارهای یک رده بالاتر هستند که استفاده های قابل توجهی برای آنها در صنعت قابل تصور است.

۲-۲. پیشینه داده کاوی

داده کاوی و کشف دانش در پایگاه داده ها از جمله موضوع هایی هستند که همزمان با ایجاد و استفاده از پایگاه داده ها در اوایل دهه ۸۰ برای جستجوی دانش در داده ها شکل گرفت. شاید بتوان لوول (۱۹۸۳) را اولین شخصی دانست که گزارشی در مورد داده کاوی تحت عنوان « شبیه سازی فعالیت داده کاوی » ارائه نمود. همزمان با او پژوهشگران و متخصصان علوم رایانه، آمار، هوش مصنوعی، یادگیری ماشین و . . . نیز به پژوهش در این زمینه و زمینه های مرتبط با آن پرداخته اند.

پژوهش جدی روی موضوع داده کاوی از اوایل دهه ۹۰ شروع شد. پژوهش ها و مطالعه های زیادی در این زمینه صورت گرفته، همچنین سمینارها، دوره های آموزشی و کنفرانس هایی نیز برگزار شده است. نتایج پایه های نظری داده کاوی در تعدادی از مقاله های پژوهشی آورده شده است. مثلاً سال ۱۹۹۱ پیاتتسکی و شاپیرو [1] « استقلال آماری قاعده ها در داده کاوی» را بررسی نموده اند. سال ۱۹۹۵ هافمن و نش استفاده از داده کاوی و داده انبار [2] توسط بانک های آمریکا را بررسی نموده و بیان کردند که چگونه این سیستم ها برای بانک های آمریکا قدرت رقابت بیشتری ایجاد می کنند. چت فیلد مشکلات ایجاد شده توسط داده کاوی را بررسی نمود و همچنین مقاله ای تحت عنوان « مدل های خطی غیر دقیق داده کاوی و استنباط آماری » ارایه نمود. هندری نیز دیدگاه اقتصاد سنجی روی داده کاوی را تهیه کرد. در این سال انجمن داده کاوی همزمان با اولین کنفرانس بین المللی «کشف دانش و داده کاوی»

[1] Piatetsky-shapiro
[2] Data warehouse

شروع به کار کرد. این کنفرانس توسعه یافته چهار دوره آموزشی بین المللی در پایگاه های داده درسال ۱۹۸۹ تا ۱۹۹٤ بود. انجمن مذکور، یک سازمان علمی به نام *ACM- SIGKDD* را ایجاد نمود. سال ۱۹۹٦ ایمیلنسکی[۳] و منیلا[٤] دیدگاهی از داده کاوی به عنوان «پرس و جو کننده از پایگاه های استنتاجی[٥]» را پیشنهاد کردند. فایاد، پیاتتسکی – شاپیرو، اودورأسامی پیشرفت های کشف دانش و داده کاوی را عنوان کردند. در سال ۱۹۹۷ منیلا خلاصه ای از مطالعه روی اساس داده کاوی ارایه نمود. باربارا و همکاران نیز دیدگاه کاهش داده ها روی داده کاوی را در گزارش کاهش داده های نیوجرسی ارایه نمودند.

در مدیریت مالی می توان، تحلیل داده های مالی و مدل سازی مالی بنینگاه و چاچ کز و هیگینز [٦] را ملاحظه کرد. فریدمن نیز مقاله ای در ارتباط با مفهوم آمار و داده کاوی ارایه نمود. سال ۱۹۹۸ هند [۷] مقاله ای تحت عنوان « داده کاوی : آمار یا بیشتر؟ » ارائه نمود. کلینبرگ[۸] پائودیمیتریو و راغان [۹] دیدگاه اقتصاد سنجی روی داده کاوی و عملکرد داده کاوی به عنوان یک مسئله بهینه را ارایه نمودند. در این سال نیز کنفرانس های ناحیه ای و بین المللی در مورد داده کاوی برگزار شد که از جمله می توان به کنفرانس آسیا و اقیانوسیه درباره کشف دانش و داده کاوی اشاره کرد. سال ۲۰۰۰ هند و همکاران و اسمیت بحث های مقایسه ای بین آمار و داده کاوی را ارایه کردند. سری و استاوا، کولی، رش پاند و تن استفاده از وب در کاوش داده ها و کاربردهای آن را ارایه کردند. سال ۲۰۰۲ کلادیو کانورسانو و همکاران « مدل آمیخته چندگانه جمع پذیر تعمیم یافته » برای داده کاوی را بررسی نمودند. پائلو و گیانلوکاپاسرون، « داده کاوی ساختارهای پیوند برای مدل رفتار مصرف کننده » را ارایه نمودند.

۲-۳. پروسه‌ی کشف دانش از پایگاه داده[۱۰]

یک پایگاه داده یک ذخیره‌سازی اطلاعات قابل اطمینان است، یکی از اهداف اولیه و اصلی این ذخیره‌سازی بازیابی موثر اطلاعات می‌باشد. این اطلاعات بازیابی شونده لزوما یک کپی از اطلاعات ذخیره شده در پایگاه داده نیستند، بلکه اطلاعات استنتاجی از آن می‌باشند. دو نوع استنتاج از اطلاعات یک پایگاه داده داریم: *[Holsheimer99]*

۱. استنتاج قیاسی[۱۱]: یک تکنیک برای استنتاج اطلاعات است که یک سلسله مراتب منطقی از اطلاعات پایگاه داده می‌باشد. اکثر سیستم‌های مدیریت پایگاه داده‌ها[۱۲]، مانند سیستم‌های مدیریت پایگاه داده‌های رابطه‌ای، اپراتورهای ساده‌ای را برای استنتاج اطلاعات در اختیار می‌گذارند. برای مثال یک اپراتور پیوند بین دو جدول *Employee-Department* و *Manager-Department* در مجموع یک ارتباط بین کارمندان و مدیران را نتیجه می‌دهند.

۲. استنتاج استقرایی [۱۳]: یک تکنیک برای استنتاج اطلاعاتی است که از اطلاعات موجود در پایگاه داده استنباط[۱٤] می‌شود. برای مثال از جداول *Employee-Department* و *Department-manager* مثال بالا، ممکن است این نتیجه‌گیری حاصل شود که هر کارمند یک مدیر دارد.

جستجو برای این اطلاعات سطح بالا یا در اصطلاح دانش، هدف پروسه‌ی *KDD* می‌باشد. در پروسه‌ی *KDD* ما به دنبال الگوهایی با ساختار عبارات منطقی هستیم.

---

[3] Imielnski
[4] Mannila
[5] Inductive databases
[6] Benninga, Czaczkes, Higgins
[7] Hand
[8] Kleinberg
[9] Paodimitriou , Raghavan
[10] Knowledge Discovery in Database Process
[11] Deduction
[12] DBMSs
[13] induction
[14] generalize

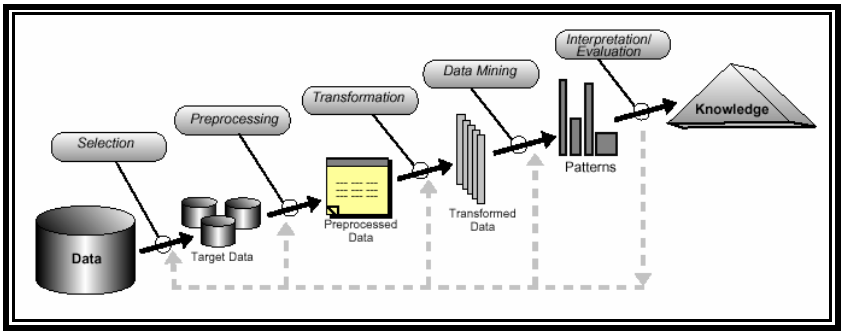
تعریف: ***KDD*** یا کشف دانش از پایگاه داده یک پروسه‌ی علمی برای شناسایی الگوهای معتبر، نوین، بالقوه مفید و قابل فهم از داده‌ها می‌باشد. *[Breiman 2003]*

داده کاوی: یک مرحله از پروسه‌ی ***KDD*** می‌باشد که با استفاده از الگوریتم‌های کاوش مشخصی یک سری الگوها را از پایگاه داده استخراج می‌کند.

۲-۳-۱. ویژگی‌های KDD

ویژگی‌های زیادی برای یک پروسه‌ی *KDD* در نوشتجات مختلف ذکر شده است. در اینجا مراحل این پروسه را بر اساس یکی از این نوشته‌ها بصورت زیر عنوان می‌کنیم: *[John97]*

- استخراج داده‌ها
- آماده کردن داده‌ها
- مهندسی داده‌ها
- مهندسی الگوریتم و تعیین استراتژی‌های کاوش
- اجرای الگوریتم کاوش



شکل ۲-۱ . مراحل پروسه‌ی کشف دانش

۲-۳-۲. استخراج داده‌ها

موقعی که یک مساله تعریف شد، باید داده‌های وابسته به آن مساله جمع‌آوری شوند. در بسیاری از موارد، داده‌های مورد نیاز از یک پایگاه داده یا انبار داده‌ها [15] استخراج می‌گردند. معمولا الگوریتم‌های داده‌کاوی نمی‌توانند مستقیما روی داده‌های بانک اطلاعات اجرا شوند، در اینگونه موارد باید داده‌ها قبل از اعمال الگوریتم‌ها، آماده شوند.

۲-۳-۳. آماده کردن داده‌ها

از آنجایی که داده‌های مورد نظر جمع‌آوری شده‌اند، خیلی مهم است که آگاه باشیم که این داده‌ها نیاز به پاکسازی دارند و ما باید زمان زیادی را صرف این کار بکنیم. زیرا بسیاری از منابع خطا، می‌توانند هنگام جمع‌آوری داده‌ها از چندین پایگاه داده در یک پایگاه داده تحلیلی، موجود باشند و یک تحلیلگر خوب ناچار است بسیاری از بررسی‌های اعتبار داده‌ای را بر روی داده‌های استخراج شده، انجام دهد. بسیار نادر اتفاق می‌افتد که داده‌های جمع‌آوری شده دارای مشکل نباشند. *[Berger2001]*

[15] Data Warehouse

۲-۳-۴. مهندسی داده‌ها

بعد از انجام مراحل فوق سه مشکل اصلی هنوز وجود دارد:*[Berger2001]*

۱. ممکن است چشمپوشی از تعدادی از ویژگی‌های داده‌ها برای ما سودمند باشد. چون ممکن است که بخواهیم خطاها را از داده‌ها جدا کنیم.

۲. ممکن است مجبور باشیم به دلیل ناکارآمدی الگوریتم‌های داده‌کاوی داده‌هایمان را به‌یک مجموعه داده‌ی ساده‌تر تقلیل دهیم.

۳. بسیاری از داده‌ها ممکن است که بصورت دیگری غیر از این شکل بیان شوند که بسیار مفیدتر باشند. مثلا برای خصوصیت جنسیت، می‌توانیم اگر این ویژگی متنی باشد، آن‌را به دودویی تبدیل کرده و زن را مثلا صفر و مرد را یک در نظر بگیریم.

۲-۳-۵. مهندسی الگوریتم و تعیین استراتژی‌های کاوش

تعداد الگوریتم‌های بسیار زیادی که در زیر هر کدام از متدها قرار می‌گیرند، می‌توانند حتی برای افراد حرفه‌ای هم گیج‌کننده باشند. این نکته باید ذکر شود که همه‌ی این الگوریتم‌ها دارای تکنیک‌های پایه‌ای یکسانی هستند.*[Fayyad 2005]*

انواع مختلفی از الگوریتم‌های داده‌کاوی وجود دارند. مدل‌های مشهور، از قوانین و درخت‌های تصمیم، رگرسیون و کلاسه‌بندی غیرخطی، متدهای مبتنی بر مثال (شامل متدهای نزدیکترین همسایه و استنتاج بر اساس مورد[16])، مدل‌های آماری (نظیر شبکه‌های بیز، تابع توزیع احتمال نرمال، تابع توزیع احتمال $\chi 2$ و...) و مدل‌های یادگیری رابطه‌ای (نظیر برنامه‌نویسی منطقی استنتاجی)، استفاده می‌کنند. لازم به ذکر است که هر تکنیک برای انواع مختلفی از مسایل بهتر از سایر تکنیک‌ها عمل می‌کند. برای مثال کلاسه‌بندی‌های درخت تقسیم برای پیدا کردن ساختار در فضاهای با ابعاد بالا و همچنین در مسایل با داده‌هایی که می‌توانند پیوسته‌یا طبقه بندی شده باشند، مفید خواهد بود (زیرا متدهای بر پایه‌ی درخت نیاز به فواصل متریک ندارند)، در حالی‌که این درخت‌ها برای مسایلی که نیاز به مرزبندی دقیق بین کلاس‌ها دارند، مفید نمی‌باشد. بنابراین هیچ متد داده کاوی واحدی نمی‌تواند وجود داشته باشد و انتخاب یک الگوریتم مشخص برای کاربردهای ویژه نیاز به مهارتهای خاصی دارد.

در عمل به علت گستردگی، پیچیدگی و حجم بسیار بالای داده‌های موجود برای یک کاربرد خاص و نیاز به بهینه‌سازی انها، دامنه متدهای مفید محدود شده است.*[Fayyad 2005]*

این مرحله یعنی اینکه مفیدتر است که برای یک منظور خاص و بر روی یک سری داده‌های ویژه، چه الگوریتمی اعمال شود که بهترین کارایی را برای ما داشته باشد؟

۲-۳-۶. اجرای الگوریتم کاوش و ارزیابی نتایج

مرحله‌ی اجرای الگوریتم مرحله‌ای است که کافی است الگوریتم به اجرا کننده داده شود و ما راحت به صندلی تکیه دهیم و نتایج را نگاه کنیم. مساله‌ای که در این مرحله قابل توجه است نحوه‌ی تقسیم کردن داده‌ها به مجموعه‌ی آموزشی و مجموعه‌ی تست می‌باشد.

۲-٤. قوانین وابسته سازی

هدف از کاوش قوانین از نوع وابسته‌سازی، یافتن ارتباطات بین اجزا یک مجموعه می‌باشد. به این ترتیب، جستجو و یافتن وابستگی‌ها، همبستگی‌ها و ساختارهای علی و معلولی موجود بین یکسری اجزا و یا اشیای موجود در بانک‌های اطلاعات تراکنشی رابطه‌ای و سایر انبارهای اطلاعاتی در این مقوله جای می‌گیرد. *[HK 2000 ch6]*

---

[16] Case-base reasoning

فرض کنید مجموعه‌ای از داده‌ها داریم که در آن، هر رکورد داده شامل اجزا کوچک‌تری می‌باشد. فرآیند کاوش قوانین وابسته‌سازی، قوانین وابستگی‌ای را تولید می‌نماید که انجام یک جز به انجام جزیی دیگر منوط است. شکل این قوانین به صورت زیر می‌باشد:

[ضریب اطمینان، ضریب پشتیبانی] تالی ⟶ مقدم

ضریب پشتیبانی، احتمال اینکه شمول مقدم و تالی در یک تراکنش می‌باشد. این ضریب را با علامت S نمایش می‌دهیم.

ضریب اطمینان، احتمال اینکه اگر تراکنشی، شرایط مقدم را ارضا کند، آنگاه شرایط تالی را نیز ارضا می‌نماید، می‌باشد. این ضریب را با علامت C نمایش می دهیم. به عنوان مثال قانون زیر را در نظر بگیرید:

[۰/۶٪ , ۶۵٪] (″شیر″ , X) خرید ⟶ (″نان″ , X) خریدن

این قانون نشان می دهد که اگر شخصی ″نان″ خریداری کند به احتمال ۶۵ درصد ″شیر″ هم خریداری می‌نماید.

قوانین وابسته‌سازی به فرم $X \rightarrow Y$ و یا $A_1 \wedge A_2 \wedge \dots \wedge A_m \rightarrow B_1 \wedge \dots \wedge B_n$ می‌باشند که Ai و Bj ها زوج‌های خصیصه –ارزش می‌باشند. قانون وابسته سازی $X \rightarrow Y$ به مفهوم زیر است :

«یک رکورد بانک اطلاعات که شرایط X را ارضا کند آنگاه شرایط Y را نیز ارضا می نماید.»

در سالهای اخیر در زمینه کاوش کارآیی قوانین وابسته سازی تحقیقات و الگوریتم‌های زیادی بررسی شده است. *[HK2000ch1]*

۲-۴-۱. اصول کاوش قوانین وابسته سازی

در ابتدا این نکته را یادآور شویم که در اینجا، توجه ما بر قوانین وابسته‌سازی تک‌بعدی می‌باشد. فرض می گردد ورودی، یک بانک از تراکنش‌ها بوده، هر تراکنش لیستی از عناصر[۱۷] می‌باشد. هدف یافتن کلیه‌ی قوانین وابسته به یکدیگراست. به عنوان مثال در یک مجموعه‌ی بزرگ از داده‌های یک فروشگاه، قوانین مشابه با قانون زیر جستجو و کاوش می‌شوند:

«۹۸ درصد افرادی که لاستیک و فرمان خودرو خریداری می کنند، سرویس و خدمات نیز نیاز دارند»

کاوش اجزا وابسته و مرتبط با یکدیگر در دوگام به شرح زیر انجام می‌پذیرد:

یافتن مجموعه عناصر تکراری[۱۸] :در این گام می‌بایست مجموعه اجزایی که حداقل ضریب پشتیبانی را دارا هستند با توجه به نکات زیرتعیین شوند:

اول اینکه زیر مجموعه‌ای از مجموعه عناصر تکراری، خود یک مجموعه عناصر تکراری می‌باشد، یعنی اگر $\{A,B\}$ یک مجموعه عنصر تکراری باشد آنگاه $\{A\}$ و $\{B\}$ نیز هر کدام مجموعه عنصر تکراری مجزا هستند.

دوم اینکه به صورت تکراری[۱۹] می توان مجموعه عناصر تکراری باکاردینالیتی بالاتر، تولید نمود.

استفاده از مجموعه عناصر تکراری : در این گام برای تولید قوانین وابسته‌سازی از مجموعه عناصر تکراری حاصل آمده استفاده می گردد. *[DK 2000]*

[۱۷] item
[۱۸] Frequent Itemset
[۱۹] Itterative

۲-۴-۲. اصول استقرا در کاوش قوانین وابسته سازی

جمع‌آوری تعدادی از اجزا و ایجاد بخش‌های بزرگتر، بعنوان مثال یافتن عناصر دوعضوی و مبنا قرار دادن آنها، سپس تهیه عناصر دو عضوی دیگر و عناصر سه عضوی و بهمین ترتیب تهیه عناصر چند عضوی. نکته قابل ذکر اینکه می بایست تمامی زیر مجموعه‌های یک مجموعه عناصر تکراری نیز منظور گردند.

هدف، در یک روش کاوش قوانین وابسته سازی، پیدا کردن تمامی قوانین وابسته‌سازی‌ای است که دارای شرایط حداقل ضریب پشتیبانی و حداقل ضریب اطمینان تعریف شده توسط کاربر باشند.

مثالی از این مساله را در اینجا می‌آوریم ، فرض کنید در جدول «*People*» که در زیر آورده شده است،هدف پیدا کردن کلیه‌ی قوانین وابسته‌سازی با حداقل ضریب پشتیبانی *40%*، و حداقل ضریب اطمینان *50%* باشد: *[Sikant95]*

RecordID	*Age*	*Married*	*NumCars*
100	*23*	*No*	*0*
200	*25*	*Yes*	*1*
300	*29*	*No*	*1*
400	*34*	*Yes*	*2*
500	*38*	*Yes*	*2*

جدول ۲-۱. جدول *People*

در اینجا دو ویژگی کمی وجود دارند، *Age* و *NamCars*. فرض کنید که در اولین مرحله ما فرض کنیم که تصمیم گرفته‌ایم که *Age* را به ٤ بازه تقسیم‌بندی کنیم:

Interval
20 .. 24
25 .. 29
30 .. 34
35 .. 39

جدول ۲-۲. افراز *Age*

نتایج کار در زیر نشان داده شده است:

RecordID	*Age*	*Married*	*NumCars*
100	*20 .. 24*	*No*	*0*
200	*25 .. 29*	*Yes*	*1*
300	*25 .. 29*	*No*	*1*
400	*30 .. 34*	*Yes*	*2*
500	*35 .. 39*	*Yes*	*2*

جدول ۲-۳. جدول *People* بعد از افراز *Age*

حال نوبت به نگاشت مقادیر می‌رسد. نحوه‌ی نگاشت را بصورت زیر در نظر می گیریم:

Interval	*Integer*
20 .. 24	*1*
25 .. 29	*2*
30 .. 34	*3*
35 .. 39	*4*

Value	*Integer*
Yes	*1*
No	*2*

جدول ۲-۴. نگاشت *Age* و *Married*

نتایج اعمال این نگاشت بصورت زیر می‌باشد:

RecordID	*Age*	*Married*	*NumCars*
100	*1*	*2*	*0*
200	*2*	*1*	*1*
300	*2*	*2*	*1*
400	*3*	*1*	*2*

500	*4*	*1*	*2*

جدول ۲-۵. جدول *People* بعد از اعمال نگاشت

برای این مثال نمونه‌ای از مجموعه عناصر تکراری به شکل زیر می‌باشد:

Itemset	*Support*
{< Age: 20 .. 29>}	*3*
{< Age: 30 .. 39>}	*2*
{< Married: Yes>}	*3*
{< Married: No>}	*2*
{< NumCars: 0 .. 1>}	*3*
{< Age: 30 .. 39>,< Married: Yes>}	*2*

جدول ۲-۶. نمونه‌هایی از مجموعه عناصر تکراری

با فرض اینکه حداقل پشتیبانی ٤٠٪ و حداقل اطمینان ٥٠٪، مد نظر ما بوده است، جدول زیر نتایج را به ما نشان می‌دهد:

Rule	*Support*	*Confidence*
< Age: 30 .. 39 > and < Married: Yes > → < NumCars: 2 >	*40%*	*100%*
< Age: 20 .. 29 > → < NumCars: 0 .. 1 >	*50%*	*66.6%*

جدول ۲-۷. نمونه‌هایی از قوانین تولید شده

۲-۴-۳. کاوش اجزای وابسته

مساله‌ی کاوش قوانین وابسته‌سازی می‌تواند به دو زیر مساله تقسیم شود:*[Srawagi]*

۱. پیدا کردن همه‌ی ترکیب‌های عناصر که پشتیبانی آن‌ها از حداقل پشتیبانی بیشتر باشد که به آن مجموعه عناصر تکراری[۲۰] گفته می‌شود. این بخش زمانبرترین بخش الگوریتم می‌باشد.

۲. استفاده از مجموعه عناصر تکراری برای تولید قوانین موردنظر.

ایده‌ی اصلی در این‌جا این است که اگر *AB* و *ABCD* عضو این مجموعه باشند، در این‌صورت قانون $AB \rightarrow CD$ در صورتی برقرار است که نسبت پشتیبانی *ABCD* به پشتیبانی *AB* بزرگتر یا مساوی مقدار اطمینان باشد. توجه داشته باشید که این قانون حداقل پشتیبانی را دارد، زیرا *ABCD* عنصر مجموعه‌ی تکراری بوده است.

۲-۴-۴. الگوریتم Apriority

مفاهیم پایه در یافتن قوانین وابستگی یکسان می‌باشد. الگوریتم‌های متفاوتی برای این کار پیشنهاد شده‌اند که خروجی همه‌ی آن‌ها یکسان می‌باشد. به‌همین دلیل اصرار به استفاده از الگوریتم خاصی نداریم. برای این منظور از معروف‌ترین و پرکاربردترین آن‌ها یعنی الگوریتم *Apriority* استفاده می‌کنیم. حال سوالی که مطرح است این است که ما به چه نحو به بهترین شکل می‌توانیم ساختار الگوریتم *Apriority* را با یک پایگاه داده مجتمع کنیم؟ الگوریتم *Apriority* برای پیدا کردن مجموعه عناصر تکراری، چندین گذر بر روی داده‌ها انجام می‌دهد. در K اُمین گذر، این الگوریتم همه‌ی مجموعه عناصر دارای k عنصر را پیدا می‌کند که به آن *k-itemset* می‌گویند. هر گذر از دو فاز تشکیل می‌شود. با فرض این‌که F_k نشان‌دهنده‌ی عناصر موجود در *k-itemset* و C_k مجموعه‌ی کاندیداهای *k-itemset* باشد، داریم:

فاز اول: فاز تولید کاندیداها که مجموعه‌ی همه‌ی عناصر *(k-1)-itemset* یعنی F_{k-1} می‌باشد که در *(k-1)* اُمین فاز پیدا شده‌اند. این عناصر برای تولید مجموعه کاندیداهای C_k به کار می‌روند. روال تولید کاندیدا ما را مطمین می‌سازد که C_k یک *Superset* برای

[20] Frequent item set

عناصر *k-itemset* می‌باشد، یک ساختار داده‌ی خاص از درخت درهم سازی که در حافظه نگهداری می‌شود برای ذخیره C_k مورد استفاده قرار می‌گیرد.

فاز دوم: پیمایش کردن داده‌ها در فاز شمارش پشتیبانی[21]. برای هر تراکنش، کاندیداهای موجود در C_k که در آن تراکنش نیز وجود دارند، برای الحاق به درخت درهم سازی معین می‌شوند و سپس شماره‌ی پشتیبانی[22] آن‌ها افزایش پیدا می کند. در آخرین فاز برای این‌که داده‌های موجود در C_k، که می‌توانند در F_k نیز باشند شناسایی شوند،این مجموعه مورد جستجو قرار می گیرد.

شرط پایانی الگوریتم این است که F_k یا C_{k+1} خالی شوند.

فرمت ورودی: جدول تراکنش‌های T دارای دو ویژگی می باشد: شناسه‌ی تراکنش *(tid)* و شناسه‌ی عنصر *(item)*. تعداد عناصر هر *tid* متغیر بوده و در زمان ساخت جدول مشخص نیست.

۲-۵. وابسته سازی در SQL

هر گذر k ازالگوریتم *Apriority* ابتدا یک مجموعه *itemset* کاندیدا (C_k) از F_{k-1} مرحله‌ی قبل تولید می‌کند.

در مرحله‌ی پیوند یک *Supreset* از C_k بوسیله پیوند کردن F_{k-1} با خودش تولید می‌گردد:

```
Insert into Ck select I1.item1, …, I1.itemk-1, I2.itemk-1
From Fk-1 I1, Fk-1 I2
Where  I1.item1 = I2.item1 and
.
.
.
I1.itemk-2 = I2.itemk-2 and
I1.itemk-1 < I2.itemk-1
```

بعنوان مثال فرض کنید $F_3 = \{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{1,3,5\},\{2,3.4\}\}$ باشد. بعد از مرحله‌ی پیوند مجموعه‌ی C_4 برابر خواهد بود با: $\{\{1,2,3,4\},\{1,3,4.5\}\}$.

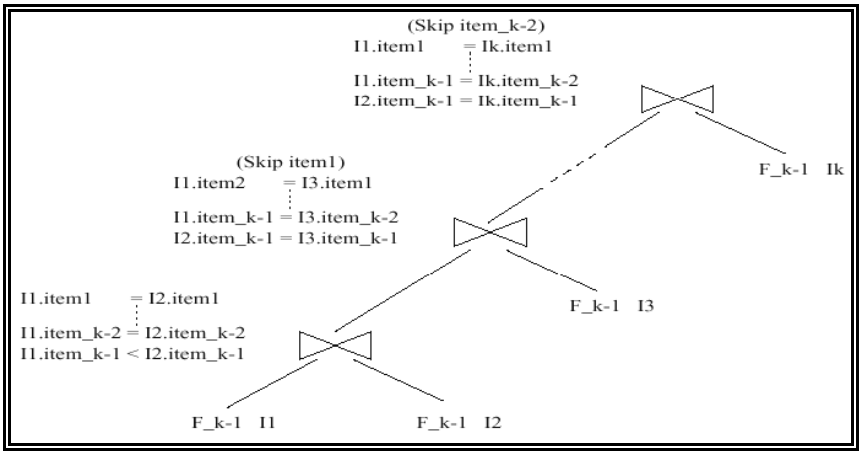
مرحله‌ی بعدی مرحله‌ی هرس می‌باشد. هر مجموعه عنصر $c \in C_k$ که بعضی از زیر مجموعه‌های *k-1* عنصری آن‌ها در F_{k-1} نباشد حذف خواهد شد. مثلا در مثال فوق *{1,3,4,5}* بدلیل این‌که زیر مجموعه‌ی *{1,4,5}* آن در F_3 نیست حذف خواهد شد. بنابراین تنها مجموعه عنصر عضو C_4 مجموعه *{1,2,3,4}* خواهد بود.

ما می‌توانیم مرحله‌ی هرس را نیز در همان پرس‌وجوی فوق بگنجانیم و آن را به صورت یک *k-way*پیوند به صورت زیر نشان دهیم. بعد از پیوند کردن *I1* و *I2*، ما k مجموعه عنصر که از *(I1.item1, …, I1.itemk-1, I2.itemk-1)* تشکیل شده‌اند را داریم. برای این مجموعه عناصر، دو تا از زیرمجموعه‌های با طول *k-1*، از آنجایی که از دو مجموعه عنصر موجود در F_{k-1} ساخته شده‌اند، *Frequent* می‌باشد. در نتیجه ما با پیوندهای اضافی *k-2* زیر مجموعه‌ی باقیمانده را چک می‌کنیم. اساس کار بر این است که در هر لحظه از یکی از عناصر، از مجموعه‌ی *k-itemset* صرف‌نظر می‌کنیم. ابتدا از *item1* صرف‌نظر کرده و چک می‌کنیم که آیا زیر مجموعه‌ی *(I1.item2, …, I1.itemk-1, I2.itemk-1)* به F_{k-1} تعلق دارد یا نه (در پرس‌وجوی زیر ما این‌کار را با پیوند کردن با I_3 انجام می‌دهیم: در حالت کلی هنگام پیوند کردن با I_r ما از عنصر *r-2* صرف‌نظر می‌کنیم. ساخت یک ایندکس اولیه بر روی *(item1, …, itemk-1)* از F_{k-1} به کاراتر کردن این عملیات، خیلی کمک می‌کند.

---

[21] Support Counting
[22] Support count

بعضی از مراحل مختلف فوق توسط همروندهای سیستمهای پایگاه داده به صورت همزمان اجرا می‌گردند.



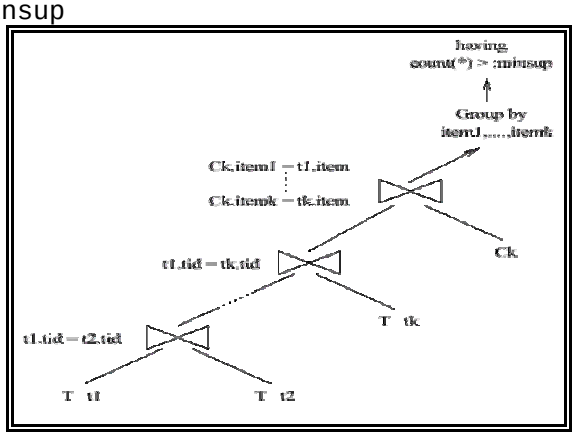
شکل ۲-۲. ساختن کاندیداها برای هر K

۲-۵-۱. شمارش پشتیبانی[۲۳] برای پیدا کردن مجموعه عناصر تکراری

این بخش زمانبرترین قسمت از الگوریتم قوانین وابسته‌سازی می‌باشد. ما برای شمارش پشتیبانی عناصر C_k از مجموعه عناصر C_k و داده‌های جدول T استفاده می‌کنیم ما در اینجا برای پیاده سازی *SQL*، دو رهیافت بر پایه‌ی *SQL-92* ارایه می‌دهیم:

k-way Joins:

```
Insert into Fk select item1,. .., itemk, count (*)
From Ck, T t1,. .., T tk
Where t1.item = Ck.item and

tk.item = Ck.item and
t1.tid = t2.tid and

tk-1.tid = tk.tid
Group by item1, item2,. .., itemk
Having count(*) > :minsup
```



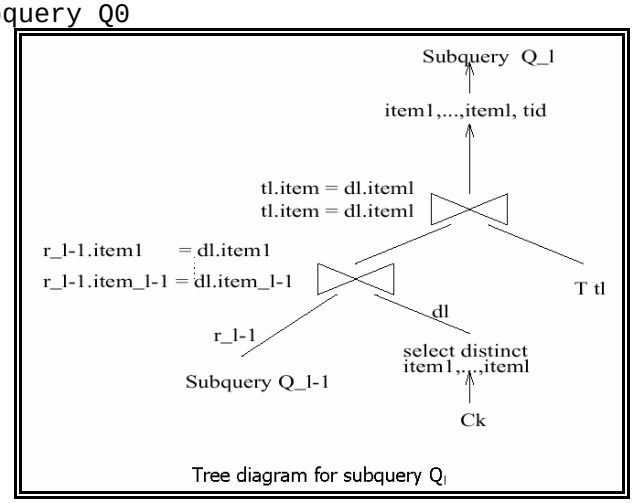
شکل ۲-۳. شمارش پشتیبانی با روش *K-way* پیوند

Subquery-based:

```
Insert into fk Select item1,. .., itemk, count(*)
From (Subquery Qk) t
Group by item1, item2,. .., itemk
Having count (*) > :minsup

Subquery Q1 (for any l between 1 and k):
       Select item1,. .., iteml, tid
       From T tl, (Subquery Ql-1) as rl-1
```

[23] Counting Support

```
                (Select distinct item1, …, iteml from Ck ) as dl
        Where rl-1.item1 = dl.item1 and

rl-1.iteml-1 = dl.iteml-1 and
rl-1.tid = tl.tid and
tl.item = dl.iteml

Subquery Q0: No Subquery Q0
```



شکل ۲-۴. شمارش پشتیبانی با روش *Subquery-based*

در *[Srawagi98]* برای این روشِ پیاده‌سازی علاوه بر دو رهیافت فوق، دو رهیافت دیگر و ۶ رهیافت هم برای پیاده‌سازی بر اساس *SQL-OR* پیشنهاد و توضیح داده شده است. جهت اطلاعات بیشتر می‌توانید به مقاله‌ی فوق مراجعه نمایید.

۲-۶. خلاصه

امروزه با حجم عظیمی از داده ها روبرو هستیم. برای استفاده از آنها به ابزارهای کشف دانش نیاز داریم. داده کاوی به عنوان یک توانایی پیشرفته در تحلیل داده و کشف دانش مورد استفاده قرار می گیرد. داده کاوی در علوم (ستاره شناسی،...)، در تجارت (تبلیغات، مدیریت ارتباط با مشتری،...)، در وب (موتورهای جستجو،...)، در مسایل دولتی (فعالیتهای ضد تروریستی،...) کاربرد دارد. عبارت داده کاوی شباهت به استخراج زغال سنگ و طلا دارد. داده کاوی نیز اطلاعات را که در انبارهای داده مدفون شده است، استخراج می کند. در واقع هدف از داده کاوی ایجاد مدل هایی برای تصمیم گیری است. این مدلها رفتارهای آینده را براساس تحلیلهای گذشته پیش بینی می کنند.

فصل سوم
سیستم پشتیبانی از تصمیم

Decision Support System

در این فصل مقدمه مختصری از سیستمهای پشتیبانی از تصمیم به عنوان یکی از مهمترین ابزارهای داده کاوی ارایه خواهیم کرد. همچنین به بررسی فرایند آماده سازی داده ها که مهمترین مرحله هر ابزار داده کاوی میباشد خواهیم پرداخت. در ادامه به فرایند تحلیل روی خط که امروزه ابزارهای خاص خود را در پایگاههای داده بزرگ دارد میپردازیم و ...

- مقدمه
- مخزن داده ها *Data Warehouse*
- پردازش تراکنشهای روی خط *OLTP*
- فرایند تحلیل روی خط *OLAP*
- پرس و جو های *OLAP*
- سیستمهای پشتیبانی از تصمیم *DSS*
- پایگاههای داده آماری
- خلاصه

۳-۱. مقدمه

سیستم های مدیریت پایگاه[1] داده بطور گسترده ای توسط سازمانها برای نگهداری اطلاعاتی که اعمال روزمره آنها را مستند میکند ، بکار میرود. در برنامه هایی که برخی اطلاعات عملیاتی را دستکاری میکنند تراکنشها معمولا تغییرات اندکی بوجود می آورند اما در عین حال تراکنشهای بسیار فراوانی باید بطور مطمئن و موثر پردازش شوند. برخی برنامه های پردازش تراکنشهای روی خط[2] توسط شرکتهای بزرگ مدیریت پایگاه داده در دهه های اخیر بوجود آمدند و بدون شک اهمیت آنها رو به رشد است. سیستمهای مدیریت پایگاه داده تمام تلاشها و تجارت خویش را برای بهتر شدن این ابزار بکار گرفته اند.

اخیرا سازمانها بطور فزاینده ای تاکید بر استفاده از نرم افزارهایی دارند که قابلیت تحلیل جامع واکتشاف داده های حال و گذشته سیستم برای بدست آوردن سیر حرکتی داده ها[3] و پیش بینی آینده را دارا باشند. همچنین تمایل بسیار زیادی برای نرم افزارهایی دیده میشود که قادرند خلاصه کاملی از اطلاعات - مجموع، میانگین، تعداد و دیگر مقیاسهای آماری- را خصوصا با نمایش گرافیکی آن در اختیار تحلیلگر قرار دهند. این گونه نرم افزارها سیستم را به یک نگرش جامع و به اصطلاح علمی قدرت تصمیم گیری سطح بالا[4] سوق میدهد. از چنین ابزارهایی تحت عنوان کلی سیستمهای پشتیبانس از تصمیم[5] یاد میشود.تعدادی از شرکتهای سازنده ی پایگاه داده ابزارهای تحقیق را برای ساده تر کردن پیاده سازی کار با ابزارهای پشتیبانی از تصمیم به سیستم خود افزوده اند. مجامع استانداردسازی بین المللی نیز برای استانداردسازی و ایجاد توافق روی مسائل پشتیبانی از تصمیم مانند زبان و معماری طراحی میکوشند.

ویژگیهای خاصی در پرس و جوهای مربوط به سیستم پشتیبانی از تصمیم وجود دارد که *SQL* فعلی را ناکارامد میکند این ویژگیها بقرار زیر است :

۱. شرط موجود در قسمت *where* اغلب شامل *and* و *or* های زیادی است. در بسیاری از سیستمهای پایگاه داده ای رابطه ای *or* از پشتیبانی ضعیفی برخوردار است.

۲. برنامه ها نیاز بسیاری به استفاده از ابزارهای آماری مانند انحراف معیار و ... دارند که *SQL-q2* از آن پشتیبانی نمیکند. همچنین پرس و جوهای *SQL* معمولا در قالب یک زبان برنامه نویسی میزبان قابل اجرا است.

۳. بسیاری از پرس وجوها منوط به شرایطی از زمان هستند. یا در فواصل زمانی تغییر میکنند. *SQL* پشتیبانی ضعیفی از تحلیل همچنین سریهای زمانی دارد.

٤. کاربران معمولا نیاز به مطرح کردن چند پرس و جوی وابسته دارند. در حالیکه هیچ راه معمولی برای شرح همچنین خانواده ای از پرس و جو ها ندارند. کاربران مجبورند آنها را بصورت مجموعه ای از پرس و جو های مستقل بنویسند که میتواند خسته کننده باشد. علاوه بر این سیستم های پایگاه داده هیچ راهی برای تشخیص و بهبود مناسب در اجرای بسیاری از پرس و جو های وابسته به یکدیگر ندارند.

سه نوع از ابزارهای تحلیل در دسترس است. اولین دسته سیستمهایی هستند که از مجموعه پرس و جوهایی که غالبا درگیر با گروهبندی فیلدها هستند پشتیبانی میکنند و اطلاعات مفیدی برای شرایط بولی پیچیده ، توابع آماری و تحلیل سریهای زمانی آماده میکنند. برنامه هایی که وابسته به همچنین پرس و جوهایی هستند پردازش تحلیلی روی خط[6] نامیده میشود. این سیستمها معمولا روی پرس و جوهایی

---

[1] Database Management System (DBMS)
[2] OnLine Transaction Processing (OLTP)
[3] Trends
[4] High - Level Decision Support Making
[5] Decision Support Tools
[6] OnLine Analytic Processing

که بر روی داده های بیان شده به صورت آرایه های چند بعدی هستند،کارایی بسیار خوبی دارند و بسیار برای کاربران کـه بـا ابزارهـای مانند صفحه گسترده ها کار میکنند مناسب هستند.

دومین، دسته سیستمهای مدیریت پایگاه داده ای هستند که از پرس و جوهـایی بـه سـبک تجـاری پـشتیبانی میکننـد. امـا همچنـین بـرای پشتیبانی از پرس و جو های *OLAP* طراحی گشته اند. این سیستم ها میتوانند در کنار سیستمهای رابطه ای برای برنامـه هـای پـشتیبانی از تصمیم بهینه شوند. بسیاری از شرکتهای سازنده ی پایگاه داده رابطه ای محصولات جدید خود را در این زمینه بهبود می بخـشند کـه بـه نظر می آید فاصله بین سیستم های مخصوص *OLAP* و پایگاه داده رابطه ای بهبود یافته به سرعت در حال کم شدن است.

سومین دسته از ابزارهای تحلیل در صدد یافتن تمایلات جذاب یا غیـر قابـل انتظـار و الگوهـای موجـود در مجموعـه داده هـای بـزرگ هستند. معمولا دا رای پرس و جوهای پیچیده ای با مشخصات یاد شـده در بـالا هـستند. در تحلیـل داده هـای اکتـشافی[۷] اگـر چـه یـک تحلیلگر میتواند یک الگوی جالب را تشخیص دهد اما بسیار مشکل است که یک پرس و جو مناسب که قابلیت اکتـشاف ایـن الگـوی جذاب در داده ها را داشته باشد به صورت یک فرمول کلی بیان کند. بعنوان مثال یک تحلیلگر بدنبال استفاده های غیر معمـول از یـک کارت اعتباری برای کشف سوء استفاده های احتمالی از یک کارت گم شده یـا دزدیـده شـده میگـردد. تحلیلگـر میتوانـد بـا نظـاره بـر الگوهای خرید مشتری محدوده ی علایق و سلایق مشتری و مانند اینها به یک استفاده غیر معمول دست یابد. میزان اطلاعات در بسیاری از کاربردها بسیار حجیم و بزرگ است بطوریکه اجازه ی تحلیل دستی و یا حتی تحلیل آماری تجاری را میسر نمیکند. در اینجـا هـدف از داده کاوی که نوع سوم ابزارهای تحلیل است پشتیبانی از تحلیل روی مجموعه داده های بسیار بزرگ میباشد.

واضح است انجام *OLAP* یا داده کاوی برروی داده های توزیع شده بطور مشقت باری آهسته تر از این خواهد بود. یک راه حل طبیعی ایجاد یک مخزن مرکزی از کلیه داده هاست. بنابراین میزتن دسترسی پذیری مخزن میتواند کمـک شـایانی بـه ابزارهـای *OLAP* وداده کاوی کند و بالعکس نتیجه رضایت بخش از ابزارهای تحلیل بسیار در گرو ساخت مخزن مناسب خواهد بود.

۳-۲. مخزن داده[۸]

پردازش تراکنشهای روی خط چیست؟ فقط در سال ۲۰۰۰ میزان ظرفیـت نـصب شـده جهـت ذخیـره سـازی اطلاعـات از کـل ظرفیـت موجود در دهه ۱۹۹۰ بیشتر بوده است .

حیات بازرگانی نوین مبتنی بر داده هاست. در حال حاضر تقریبا حجم کل اطلاعات در کامپیوترهـا هـر ۵ سـال دو برابـر مـی شـود و بـا توجه به سرعت ایجاد برنامه های چند رسانه ای و بانکهای اطلاعاتی پیش بینی می شود که شـتاب رشـد اطلاعـات بـه دو برابـر در سـال برسد .

تولید کنندگان این اطلاعات موسسات و شرکت های جدیدی هستند که امور خود را توسط کامپیوترها هدایت می کنند. سیـستم هـای تولید مکانیزه ای که داده ها را جمع آوری نموده و به مصرف می رسانند سیستم های پردازش تراکنش روی خط نامیده می شوند. ایـن سیستم ها تولید کنندگان واقعی داده ها هستند .

برنامه های کاربردی مشتری و کارگزار بدو دسته تقسیم می شوند :

۱. سیستم های پشتیبانی تصمیم گیری *(DSS)*
۲. سیستم های پردازش زنده *(OnLine)* اطلاعات

[7] Exploratory Data Analysis
[8] Data Warehouse

این دو دسته هر یک راه های کاملا متفاوتی را جهت حل مسائل تجاری ارائه می کنند. قبل از آنکه به ارزش انباره های داده پی ببریم لازم است تفاونهای این دو را بشناسیم .

سیستم های *OLTP* در کلیه خدمات بازرگانی دیده می شوند از جمله سیستم های رزرواسیون، دستگاه های فروش ، کنترل انبار، سهام و فروش و ... . این سیستم ها غالبا به زمان پاسخی بین ۱ تا ۳ ثانیه در ۱۰۰ در صد اوقات نیاز دارند. تعداد کاربران آنها در ساعات مختلف روز ، هفته و ماه می تواند بشدت متغیر باشد و درتمامی این اوقات به همان زمان پاسخ قبلی نیاز دارند. در این گونه سیستم ها معمولا مشتریان بجای ارتباط با بانکهای اطلاعاتی[9] به کارگزاران تعاملی[10] متصل می شوند. البته این گونه ارتباط لازمه دستیابی به سرعت مورد نیاز مشتریان است .

*OLTP*خود نیز به دو نوع عادی و قوی تقسیم می گردد. کارگزاران عادی قادرند تعامل را در غالب پردازش های ثبت شده در بانک اطلاعاتی[11] به اجرا بگذارند و کارگزاران قوی از پروتکل ناظر تراکنش[12] برای اجرای دستورات استفاده میکنند.در *OLTP* برای دستیابی به سرعت، سربار ارتباطی شبکه ها در حداقل ممکن نگاه داشته می شود و غالبا ارتباطات در حد انتقال یک دستور *SQL* هستند .

امروزه حتی کوچکترین تجارتها هم قادرند بسرعت پایگاه های اطلاعاتی بزرگی را با جمع آوری اطلاعات صندوق های فروش ایجاد کنند چه رسد به وب سرور ها که می توانند ظرف مدت بسیار کوتاهی چندین گیگا بایت اطلاعات جمع آوری نمایند .

زمانی برای هر کار مکانیزه ای نیاز به میلیونها پول و ده ها متخصص بود . اما امروزه هر کسی بسادگی با خرید چند کامپیوتر شخصی و استخدام یک برنامه نویس می تواند از امکانات رایانه ای بهره مند گردد. بعبارت دیگر دسترسی به خدمات رایانه ای برای ایجاد پایگاه های خصوصی از داده ها برای همگان آسانتر شده است .

در مجموع داده هایی که توسط سیستم های *OLTP* جمع آوری می شود مستقیما مورد استفاده افراد ایجاد کننده آن قرار دارد. آنها دقیقا می دانند این داده ها چیستند و همچنین می دانند چگونه نیاز های اطلاعاتی لحظه ای خود را که بطور روزمره بوجود می آید حل کنند .

سوالی که مطرح است اینست که اگر کسی خارج از مجموعه *OLTP* به این اطلاعات نیاز داشته باشد چه باید کرد. این افراد از کجا می دانند چه داده ای موجود است؟ کجا بایستی آنرا پیدا کرد و چگونه به آن دسترسی پیدا کنند؟ داده ها به چه شکلی است ؟ چه معنایی دارد؟ آخرین چیزی که افراد *OLTP* به آن رضایت خواهند داد آنست که اجازه دهند دیگران به اطلاعات گرانبهای آنان دسترسی داشته باشند. کسانی که حتی نمی دانند چه می خواهند، درخواستهای *SQL* زمانگیری را بر روی بانکهای اطلاعاتی اجرا می کنند که سرعت و قابلیت سیستم تولید کننده داده ها را پایین می آورد .

در گذشته افراد بیرون از سیستم ، از همکاران اطلاعاتی خود می خواستند با همکاران مشابه خود در سیستم مربوطه تعامل داشته و نهایتا اطلاعات مورد نظر را از سیستم استخراج نمایند. اما امروزه حتی مجموعه مدیران اطلاعاتی خود هم بدرستی نمی دانند چه اطلاعاتی در سازمان موجود است. اطلاعات بشدت توزیع شده و پراکنده است و تقریبا روی هر کامپیوتری بخشی از اطلاعات سازمان وجود دارد .

یکی از ویژگیهای کامپیوتر های شخصی و همچنین معماری مشتری/کارگزار موجب شده است که افراد اکثرا به اطلاعات سازمانی و کاربرد اطلاعات در سازمان علاقه ای نداشته و ترجیح می دهند اطلاعات را تحت مالکیت شخصی اداره کنند به این ترتیب بین اطلاعات سازمان و شخص (یا واحد های متشکله) شکاف وجود خواهد داشت. از طرف دیگر بین داده های سیستمی و اطلاعات استخراج شده نیز شکاف دیگری مشاهده می شود. کسانی که از بیرون به این اطلاعات نگاه می کنند افرادی هستند که بدنبال یافتن

---

[9] Database Server
[10] Transaction Server
[11] Stored Procedures
[12] TP Monitor

طرحها، روالها و تمایلات در داده ها هستند بطوریکه بتوانند تصمیمات بهتری بگیرند. تنیدن حصار بدور اطلاعات بمعنی تنیدن حصار در برابر تجارت دیگران است و خیلی زود همگان بازنده جنگ این حصارها خواهند بود .چه خواهد شد اگر بدیگران اجازه دسترسی به اطلاعات خود را بدهیم .

سولات زیادی مطرح هستند که بایستی پاسخ داده شوند و از آن جمله اند :

چگونه مطمئن شویم که عملکرد بیرونی ها (غریبه ها) عملکرد سیستم ما را کند نمی کند؟

چه اطلاعاتی را بایستی در اختیار بیرونی ها قرار دهیم؟

چه اطلاعاتی درونی و شخصی (فقط مربوط به سیستم تولید کننده داده) است؟

چه کسی مالک اطلاعات به اشتراک گذاشته شده است؟

چه کسی این اطلاعات را بروز میکند؟

آیا بایستی بگذاریم دسترسی به اطلاعات مستقیم باشد یا آنرا در بانک دیگری کپی کنیم؟

اطلاعات استخراج شده چگونه نگهداری شده و چگونه بروز می شود؟

برای پاسخ به سوالات فوق بایستی نیاز های استفاده کنندگان از این اطلاعات را بشناسیم و تفاوتهای میان سیستمهای پشتیبان تصمیم گیری و *OLTP* را درک کنیم .

چه کسانی از این داده ها استفاده میکنند؟

بیایید نامی برای این دسته از افراد انتخاب کنیم. این افراد مصرف کنندگان اطلاعات هستند(کسانی هستند که تصمیمات استراتژیک می گیرند) فعلا نام این افراد را شکارچی اطلاعات می گذاریم چون این نام معرف هر کسیست که به یک *PC*دسترسی دارد و نیازمند اطلاعات است. البته بازرگانان و صنعتگران اولین دسته از این افراد هستند .

۳-۳. سیستم پشتیبانی از تصمیم گیری چیست؟

این سیستم یک سیستم کارآمد و ابزاریست برای تحلیل داده ها ، یافتن ارتباط بین داده ها، تولید گزارش های کارآمد، دسترسی منعطف به داده ها، راهکار های نمایش اطلاعات در انواع ممکن، قابلیت پاسخ به سوالات، چاپ اطلاعات،انتقال داده ها به صفحات گسترده و ...

در مقایسه با سیستم های تولید داده، این ابزارها از انعطاف بیشتری در زمان پاسخگویی برخوردار هستند. معمولا کنترل یکپارچگی در آنها رعایت نشده است و قابلیت دسترسی همزمان کاربران به آن غالبا محدود است. جستجوی اطلاعات و یا بروز رسانی اطلاعات غالبا بمعنی پردازش روی تمامی اطلاعات خواهد بود. این برنامه ها برای غیر برنامه نویسان تهیه شده و بیشتر فعالیت ها در آن از طریق نشان بده و کلیک کن انجام می شود .

۳-۳-۱. سیستم های اطلاعات مدیران اجرایی[۱۳]

این دسته از برنامه ها از ابزارهای *DSS* قوی تر، ساده تر و کار آمدتر هستند. همچنین به یک زمینه تجاری خاص نزدیکتر و طبیعتا گرانتر هم هستند. البته اختلاف بین *DSS* و *EIS* بتدریج کم رنگ شده است. ابزارهای *EIS* بتازگی دامنه عمل خود را گسترش داده و در سطح سازمان خود را مطرح کرده اند بطوریکه مدیران و تحلیلگران نیز از این ابزار ها استفاده می کنند.

[13] Executive Information Systems

ابزارهای *DSS/ESS* بطور خلاصه ابزارهای پردازش تحلیلی روی خط یا *OLAP*[14] و یا ابزارهای تحلیل چند بعدی[15] نامیده می شوند و در لایه های بالاتر به آنها ابزارهای داده کاوی گفته می شود .

۳-۳-۲. مقایسه سیستم های DSS و OLTP

در جدول زیر تفاوت های دو نوع سیستم *DSS* و *OLTP* را می بینیم :

	OLTP	*DSS*
قابلیت نیاز بانک اطلاعاتی	نیاز به بانک اطلاعاتی دارد	ندارد
چه کسی از آن استفاده می کند	کارکنان سیستم تولید کننده اطلاعات	شکارچی اطلاعات
ارزش زمانی اطلاعات	به مقدار فعلی اطلاعات نیاز دارد و گزارش ها قابل باز سازی نیستند	به اطلاعات پایدار نیاز دارد. اطلاعات هر از گاه به وقت می شوند.
تعداد دسترسی ها به اطلاعات	پیوسته در طول روز کاری	با نقاط پیک کاری هر از گاه
شکل داده	داده خام است. استخراج و تبدیلی صورت نگرفته	در چندین لایه تبدیل صورت گرفته است. استخراج و فشرده سازی داده ها انجام شده
جمع آوری داده ها	از یک برنامه	از چندین محل داخلی و خارجی
آیا محل تولید داده مشخص است	بلی بیشتر داده توسط یک برنامه تولید می شود	خیر از برنامه های مختلف و بانک های اطلاعات و وب می آید
آیا اطلاعات نگارش بندی شده هستند	خیر. داده ها پیوسته و در یک نگارش هستند	بلی هر مجموعه از داده دارای تاریخ برداشت است
نوع دسترسی به داده	چندین کاربر اطلاعات را به وقت می کنند	بیشتر اوقات یک کاربر
آیا داده قابل به وقت رسانی است	مقدار کنونی مدام در حال تغییر است	فقط خواندنیست
انعطاف در دسترسی	انعطاف ندارد. فقط از طریق برنامه ها ممکن است.	منعطف از طریق یک تولید کننده درخواست و*OLAP*
راندمان	سرعت پاسخ بالا مورد نیاز است	فعالیت ها همگی مکانیزه و سریع نسبتا کند
نیازهای اطلاعاتی بخوبی فهمیده شده اند	ناپایدار و نسبی.	به مقدار زیادی کار کشف و تحقیق و جستجوی موضوعی نیاز است
دامنه اطلاعات	محدود. آن چیزی که در بانک موجود است	داده ها ممکن است از هر جایی بیایند

[14] OnLine Analytical Processing
[15] Multidimensional Analysis

۳-۳-۳. انباره داده

در محیط مشتری/کارگزار انباره داده یعنی انباره اطلاعات برای مصرف سیستم های پشتیبانی تصمیم گیری. انباره داده بک مخزن فعال و هوشمند از اطلاعات است که قادر است اطلاعات را از محیط های گوناگون جمع آوری و مدیریت کرده و نهایتا پخش نماید و در صورت لزوم نیز سیاست های تجاری را روی آنها اجرا نماید .

۳-۳-۴. عناصر انباره داری

انباره یک محل است و انباره داری یک فرآیند. این فرآیند از عناصر زیر تشکیل شده است :

۱. مدیریت انتشار اطلاعات انباره که وظیفه نسخه برداری و توزیع اطلاعات را بر روی بانک های مختلف (آنگونه که شکارچی اطلاعات تعریف می کند) به عهده دارد. شکارچی اطلاعاتی را که بایستی کپی شود، مبدا و مقصد اطلاعات، تعداد بوقت رسانی ها و تبدیلات لازم روی اطلاعات را تعریف می کند. اصطلاح تازه سازی بمفهوم کپی کامل آخرین وضعیت اطلاعات و اصطلاح بوقت رسانی بمفهوم اعمال آخرین تغییرات بکار گرفته شده اند. همه کارها می تواند بصورت خودکار و یا دستی انجام پذیرد. اطلاعات ممکن است از بانکهای رابطه ای و غیر رابطه ای تهیه شود. توجه کنید که کلیه اطلاعات خارجی قبل از ورود به سیستم، تبدیل شده و پاک سازی می شوند .

۲. بانک اطلاع رسانی یک بانک اطلاعاتی رابطه ایست که وظیفه سازماندهی و ذخیره نمودن یک نسخه از اطلاعات و همچنین تبدیلات و جمع بندی و افزودن ارزش به اطلاعات حاصله از منابع مختلف و با فرمت های مورد نظر را بعهده دارد. نگهداری فراداده نیز به عهده این بانک است . فراداده های سیستمی روابط بین جداول و ایندکس ها و غیره را بیان می کنند و فراداده های محتوایی ارزش اطلاعات را برای یک شکارچی اطلاعات روشن می سازند .

۳. راهنمای اطلاعات ترکیبی از یک راهنمای فنی و راهنمای تجاری و یک پویشگر اطلاعات است. هدف اصلی این راهنما کمک به شکارچی برای دانستن محل وجود اطلاعات ،شکل آن و روش دسترسی به آن است.

٤. پشتیبانی ابزارهای *DSS/EIS* از طریق انواع دستورات *SQL* انجام می گیرد. بسیاری از فروشندگان پروتکل *ODBC* و سایرین انواع دیگر پروتکل ها را سرویس می دهند .

۳-۳-۵. سلسله مراتب انباره ها

انواع کوچکتری از انباره های داده هستند. در عمل غرفه های داده دپارتمانی و غرفه های داده همراه از ابتدا برنامه ریزی نمی شوند بلکه در ابتدا بوجود آمده و در صورت موفقیت تکثیر شده و در نهایت مدیر بانک اطلاعاتی سازمان ممکن است بتواند یک فدراسیون آزاد ار این غرفه ها تشکیل دهد و نهایتا یک انباره داده را پایه گذاری نماید .

۳-۳-۶. ابزارهای گزارش گیری

ابزارهای تحلیل داده و خواسته پردازها بما اجازه ساختن یک دستور*SQL* را می دهند بدون آنکه مجبور باشیم برنامه ای بنویسیم یا*SQL* یاد بگیریم. با چند نشانه و کلیک عبارت های*SQL* مناسب برای گرد آوری اطلاعات و نمایش آن بشکل یک گراف / جدول و یا گزارش آماده می شود. ابزارهای برجسته تر در این زمینه امکان کنترل میزان نتایج برگشته از یک خواسته را می دهند و به این ترتیب می توان جلوی درخواستهایی را که ممکن است میلیونها رکورد را برگردانند گرفت. در سال ۱۹۹۸ بیش از ۱۵۰ نوع از این ابزارها در بازار وجود داشته است که *Microsoft Access, Oracle Reports, Business Objects* از آن جمله اند .

۳-٤. OLAP و اطلاعات چند بعدی

به ساختار *OLAP* مثل یک مکعب روبیک از داده ها نگاه کنید که می توانید آنرا در جهات مختلف بچرخانید تا بتوانید سناریو های قبلا چه شده و چه می شد اگر ... را بررسی کنید. این ابزارها دیدگاههای چند بعدی از داده ها را توسط بانکهای اطلاعاتی دو بعدی (و یا بانکهای خاص چند بعدی) تولید کرده و در اختیارما می گذارند. توان دسترسی چند بعدی به داده ها در *OLAP* قدرت فرموله کردن خواسته های پیچیده تر را بما می دهد .

برای سادگی فرض کنید *OLAP* یک صفحه گسترده با چند محور است (در صفحات گسترده متعارف فقط دو محور افقی با اختصار *A, B, C, ...* و عمودی با ایندکس های ۱و۲و۳,... داریم) در این صورت مثلا میتوانیم اطلاعات فروش یک سازمان را از دیدگاه های منطقه فروش، تاریخ، مشتری، فروشگاه، قیمت و میزان فروش بررسی کنیم. و پاسخ سولاتی نظیر میزان فروش به ازائ یک محصول و فروشگاه در یک ماه مشخص را خواهیم داشت .

مدل چند بعدی *OLAP* طریقه نمایش دادن داده ها را در مقایسه با بانک های اطلاعاتی رابطه ای تسهیل میکند. *ROLAP* با ایجاد یک لایه محافظ روی یک بانک اطلاعاتی رابطه ای سرویس فوق را ارائه میدهد. از دیدگاه فنی *OLAP* فقط راهی برای ذخیره سازی و محاسبه اطلاعات چند بعدی برای پاسخوگویی به سناریوهای کاربر است .یک مشتری*OLAP* ، داده ها را از پیش روی چندین محور جمع می زند. توجه کنید که اطلاعات قبل از وارد شدن به *OLAP* بایستی پاک سازی شوند . غالبا *OLAP* داده ها را از یک انباره داده استخراج می کند .

ابزارهای *OLAP* را به چند دسته تقسیم می کنند :

۱. *OLAP* رو میزی : ابزارهای ساده و مستقل که روی کامپیوتر های شخصی نصب شده و مکعب های کوچکی می سازند و آنها را نیز بر روی سیستم به شکل فایل ذخیره می کنند. بیشتر این ابزارها با صفحات گسترده ای نظیر *Excel* کار می کنند.به این ترتیب کسانی که در سفر هستند قادر به استفاده از این دسته از محصولات هستند.(در حال حاضر *Web OLAP* در حال جایگزین کردن این محصولات است)

۲. *MOLAP*چند بعدی : بجای ذخیره کردن اطلاعات در رکورد های کلید دار، این دسته از ابزارهای بانکهای اطلاعاتی خاصی را برای خود طراحی کرده اند بطوریکه داده ها را به شکل آرایه های مرتب شده بر اساس ابعاد داده ذخیره می کنند. در حال حاضر نیز دو استاندارد برای این تیپ ابزار وجود دارد. سرعت این ابزار بالا ولی اندازه بانک اطلاعاتی آن نسبتا کوچک است .

۳. *OLAP*رابطه ای *(ROLAP)* : این ابزار ها با ایجاد یک بستر روی بانکهای رابطه ای اطلاعات را ذخیره و بازیابی می کنند . بطوریکه اساس بهینه سازی برخی بانکهای اطلاعاتی رابطه ای مانند*Red Brick, MicreoStrategy* بر همین اساس استوار است .اندازه بانک اطلاعاتی این ابزار قابل توجه می باشد .

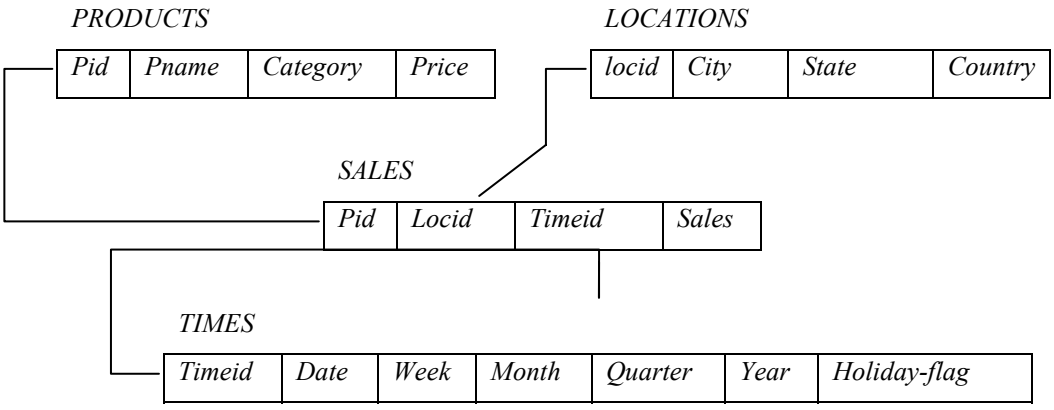
٤. *Hybrid OLAP (HOLAP)*: در اینجا منظور از *hybrid* ترکیبی از *ROLAP* و*DBMS* طرح شده در *MOLAP* است. ابزار دارای بانک اطلاعاتی بزرگ و رادمان بالاتر نسبت به *ROLAP* می باشد .

۳-۴-۱. استانداردهای OLAP

جامعه *OLAP* با دو استاندارد مواجه است، از یک طرف گروه *OLAP* با استاندارد *MD-API* و از طرف دیگر *Microsoft* با استاندارد *OLE DB for OLAP (Tensor)*. اولی از حمایت *Oracle* و دومی از حمایت فروشندگان کوچکتری برخوردار است که امیدوارند فروش *MS-SQL7* برنامه های آنان را در ابعاد فروش ویندوز مطرح کند.

۳-۵. طراحی پایگاه داده برای OLAP

شکل ۳-۱ جداول فروش یک شرکت را نشان میدهد. این شکل یک مدل ستاره ای را پیشنهاد میکندکه در مرکز آن جدول حقیقت[۱۶] فروش وجود دارد. جدول حقیقت ارتباط میان ابعاد ومقیاس مورد علاقه را به صورت یک رابطه بیان مینماید. این چنین ترکیبی از جدول حقیقت و ابعاد آن شمای ستاره ای نامیده میشود. این الگوی شما در طراحی پایگاه داده ای *OLAP* بسیار معمول و متداول است. انباره اصلی داده ها معمولا در جدول حقیقت نشان داده میشود جاییکه هیچ تکراری درآن وجود ندارد و معمولا در سطح نرمال *BCNF* قرار دارد و در حقیقت برای کم کردن اندازه جدول حقیقت شماره مشخصه ابعاد فقط در سیستم استفاده میشود.



شکل ۳-۱ . شمای ستاره ای

اطلاعات مربوط به مقادیر ابعاد در جداول ابعاد نگهداری میشود. جداول ابعاد معمولا نرمال شده نیستند. دلیل این امر اینست که جداول ابعاد در پایگاه داده مورد استفاده در *OLAP* ایستا هستند و آنومالی بروز رسانی، درج و حذف خیلی مهم نبستند. بدلیل اندازه پایگاه داده مربوط به جداول حقیقت فضای ذخیره شده توسط جداول نرمال شده ابعاد، قابل اغماض است. بعلاوه به حداقل رساندن سرعت محاسبه برای ترکیب واقعیتها در جدول حقیقت با اطلاعات ابعاد مهمترین معیار طراحی است که پیشنهاد میشود جداول ابعاد به جداول کوچکتری شکسته شوند.

زمان پاسخ کوتاه برای پرس و جوهای *OLAP* از اهمیت بسیاری برخوردار است و بیشتر سیستمها از روشهای خلاصه برداری جداول پشتیبانی میکنند. (معمولا با گروه بندی) در مواردی برخی از پرس و جوهای کاربران از جداول اصلی با خلاصه های پیش پردازش شده، قابل پاسخدهی میباشند. نکته بسیار مهم در طراحی این نکته است که کدام جدول خلاصه شده باید برای رسیدن به بهترین قابلیت دسترسی به حافظه مورد استفاده قرار بگیرد. در سیستمهای *OLAP* فعلی تصمیم برای انتخاب جدول یکی از مهمترین تصمیم گیریهای طراحی است.

ساختار ذخیره سازی و تکنیکهای فهرست بندی برای پشتیبانی از *OLAP* گسترش یافته است و به طراح پایگاه داده انتخابهای طراحی فیزیکی مختلفی را ارایه میدهد. مانند فهرست $B+$ ، فهرست بیت مپی، فهرست اتصالی، سازماندهی فایل و ...

۳-۵-۱. پرس و جوهای Nتای برتر

تحلیلگران معمولا میخواهند برترین محصولات خود را شناسایی کنند. میتوان با مرتب کردن میزان فروش هر محصول به پاسخ این پرسش دست یافت. اما اگر ما یک میلیون محصول داشته باشیم و تحلیلگر به ده تای برتر آنها علاقه داشته باشد مطمئنا این استراتژی

[16] Fact Table

بسیار ضعیف عمل خواهد کرد. پرس و جوی مثالی زیر در صدد یافتن ده محصول برتر از نظر میزان فروش در محل و زمان داده شده، است.

```
SELECT P.pid, P.pname, S.Sales
FROM Products P, Sales S
WHERE S.pid = P.pid and S.locid = 1 and S.timeid = 3
ORDER BY S.sales DESC
OPTIMIZE FOR 10 ROWS
```

جمله آخر پرس و جو در *SQL* وجود ندارد اما در محصول *DB2* از شرکت *IBM* و همچنین محصولات *ORACLE* دستور ات مشابهی یافت میشود. درنبود همچنین دستوری پایگاه داده میزان فروش را برای همه محصولات محاسبه میکند و به ترتیب نزولی آنها را مرتب مینماید. برنامه میتواند مکان نمای جواب را بعد از یافتن ده سطر ببندد. (اجرای پرس و جو را خاتمه دهد.) اما همچنان هزینه محاسبه مجموع فروش برای همه محصولا ت و مرتب کردن آن انجام گرفته است.

اکنون اجازه دهید که به بررسی این نکته بپردازیم که چگونه یک سیستم پایگاه داده میتواند پرس و جوی بالا را بصورت موثرتری انجام دهد. کلید قضیه محاسبه فروش برای محصولاتی است که بالقوه میتوانند در ردیف ده تای بالایی فروش قرار بگیرند. فرض کنید ما توزیع میزان فروش را با نگهداری یک هیستوگرام روی ستون در رابطه فروش داریم. میتوان یک مقدار فروش به عنوان مثال C را انتخاب کرد بطوریکه فقط ده محصول دارای میزان فروش بالاتر از آن باشد. برای رکوردهای فروشی که این شرط را ارضا میکنند میتوان شرط مکان و زمان را اعمال و براحتی نتیجه را مرتب کرد. پرس و جوی زیر این مسئله را بیان میکند :

```
SELECT P.pid, P.pname, S.Sales
FROM Products P, Sales S
WHERE S.pid = P.pid and S.locid = 1 and S.timeid = 3 and S.sales > C
ORDER BY S.sales DESC
```

مطمئنا این پرس و جو بسیار سریعتر از پرس و جوی دیگری که فروش کلیه محصولات را محاسبه و سپس آنها را مرتب میکرد، عمل خواهد کرد. اما چند مسئله وجود دارد :

۱. چگونه مقدار C را انتخاب کنیم ؟ هیستو گرام ودیگر سیستمهای آماری برای این هدف میتواند بکار رود. اما نکات دیگری نیز وجود دارد. بعنوان مثال بسیاری از آمارهای نگهداری شده تخمینی هستند و حتی اگر میزان C را که متناظر یا ده تای بالایی فروش است را بتوان دقیقا انتخاب نمود، این احتمال وجود دارد که اعمال شرایط دیگر نتیجه را کمتر از ده محصول برتر ارائه دهد.

۲. چه باید کرد اگر نتیجه بیشتر از ده سطر باشد ؟ از آنجاییکه انتخاب مقدار C تقریبی است میتوان این احتمال را مد نظر گرفت که رکوردهای بیش از حد انتظار در پاسخ نمایان شود. این مسئله براحتی قابل حل است و کافیست فقط ده تای برتر را به کاربر نمایش دهیم. ما هنوز هزینه کمتری را نسبت به موردی که همه محصولات را مورد محاسبه قرار میداد، مصرف کرده ایم.

۳. چه باید کرد اگر کمتر از ده مورد در نتیجه وجود داشته باشد ؟ حتی اگر میزان C را دقیقا بدست آوریم احتمال اینکه پاسخ کمتر از حد انتظار باشد وجود دارد. در این موارد میتوان پرس و جو را با C کوچکتری دوباره اجرا کرد ویا حداکثر بدون استفاده از C آنرا مورد اجرا قرار داد.

تاثیر این دستاورد بستگی به تخمین و محاسبه درست میزان برش C دارد و بویژه کم کردن دفعاتی که تعداد نتیجه ها کمتر از تعداد مورد انتظار باشد.

۳-۵-۲. محاسبات تخمینی روی خط[17]

به پرس و جوی زیر که میانگین فروش استانها را محاسبه میکند، توجه کنید :

```
SELECT L.state, AVG(S.sales)
FROM Sales S, Locations L
WHERE S.locid = L.locid
GROUP BY L.state
```

این پرس و جو میتواند بسیار هزینه بر باشد اگر رابطه فروش و مکان بسیار بزرگ باشند و ما نخواهیم توانست زمان پاسخ سریعی را برای محاسبه این هدف تجاری در اختیار کاربر قرار دهیم. یک راه حل، استفاده از پیش پردازشها و پاسخهایی است که از قبل آماده شده است. راه حل دیگر، محاسبه پاسخ پرس و جو، اما نه پاسخ دقیق بلکه یک پاسخ تقریبی با سرعت بیشتر میباشد. در این چنین فرایندی کیفیت پاسخ امر مهمی است. چنین راه حلی محاسبه تخمینی نامیده میشود. این تکنیک بسیار کارامد و جذاب میباشد، از آنرو که شیوه های موثری برای محاسبه و تخمین پاسخها در دسترس میباشد.

محاسبه تخمینی در شکل ۳-۲ نشان داده شده است. برای هر استان (معیار گروه بندی در پرس و جوی مثال) مقدار فعلی میانگین فروش نمایش داده شده است. برای هر مقدار یک فاصله اطمینان در نظر گرفته شده است. رکورد مربوط به الاسکا به ما میگوید که تخمین فعلی از میانگین فروش فروشگاهها در الاسکا ۲۸۳۲,۵ دلار میباشد و اینکه این مقدار در محدوده ۲۷۰۰,۳ دلار و ۲۹۶۴,۷ دلار با احتمال ۹۳٪ متغیر میباشد. نمای وضعیت در ستون اول مشخص میکند که تا چه اندازه به میزان واقعی مقدار میانگین نزدیک هستیم و ستون دوم مشخص میکند که آیا محاسبه میانگین برای این استان دارای اولویت میباشد یا نه ؟ محاسبه میانگین فروش برای آلاسکا دارای اولویت نیست اما محاسبه برای آریزونا دارای اولویت است. همانگونه که شکل ۳-۲ نمایان میکند، سیستمهای پایگاه داده منابع سیستمی بیشتری را برای محاسبه میانگین در استانهای اولویت دار اختصاص داده اند، محاسبات مربوز به آریزونا قویتر از آلاسکا میباشد و احتمال بالاتری را به خود اختصاص میدهد. درجه بالای به روز رسانی همراه با یک برگشت به عقب پیوسته و یک نمایش تصویری مناسب، محاسبه تخمینی را به یک تکنیک بسیار جذاب تبدیل کرده است.

Status	Periority	State	AVG	Conf	Int.Val
▓▓▓▓	●	Alabama	5232.5	97%	103.4
▓▓▓▓	○	Alaska	2832.5	93%	132.2
▓▓▓▓▓	●	Arizona	6432.5	98%	52.3
▓▓▓▓	○	Wyoming	4243.5	92%	152.3

شکل ۳-۲ . محاسبه تخمینی

برای پیاده سازی محاسبه تخمینی، پایگاه داده باید تکنیکهای آماری را که محاسبات مربوط به فاصله اطمینان را انجام میدهند، در خود سیستم فراهم نمایند. همچنین باید برای پاسخهای تخمینی از الگوریتمهای غیر انسدادی برای عملگرهای رابطه ای استفاده کرد. یک الگوریتم انسدادی نامیده میشود، اگر تا مصرف نکردن تمام ورودیها نتواند خروجی تولیدکند یا به عبارتی تا پایان محاسبات قادر به تولید خروجی نباشد و بر عکس الگوریتم غیر انسدادی الگوریتمی است که در آن همزمان با مصرف ورودیها خروجیها تولید میشوند.

[17] OnLine Aggregation

۳-٦. خلاصه

تصمیم گیری سازمانی نیاز به دانش جامع از وضع موجود سازمان دارد و بهمین دلیل بسیاری از سازمانها اقدام به تشکیل یک مخزن اطلاعات جامع و یکنواخت میکنند که شامل اطلاعات جمع آوری شده از پایگاه های داده مختلف سازمان - که هر کدام اطلاعات یک واحد کاری را نگهداری میکنند - با اطلاعات زمانی و خلاصه ی اطلاعات آنها مانند میانگین و ... میباشد. آینده تجاری سازمان با افزایش تاکید بر ابزارهای تحلیل قوی روی مخزن اطلاعات تعیین میشود.

سیستمهای مدیریت پایگاه داده رابطه ای اصلی این بازار مهم را کشف نموده اند و ابزارهای خود را برای پشتیبانی از آن توسعه داده اند بویژه شیوه ی فهرست بندی های نو و بهبود تکنینکهای جستجو برای پشتیبانی از پرس و جو های پیچیده تحت نظر قرار گرفته اند. سیستمها همچنین خصوصیات جدیدی را برای تعریف و استفاده از نماها تدارک دیده اند. استفاده از نماها بدلیل کاربرد آنها در حل پیچیدگیهای تحلیل داده بسیار شهرت یافته است. استفاده از نماها پاسخ به پرس و جو های پیچیده را بسیار سریع تر کرده است.

فصل چهارم
درخت تصمیم

Decision Tree

در این فصل مقدمه مختصری از انواع روشهای کلاسه بندی و خوشه بندی به عنوان یکی از مهمترین ابزارهای داده کاوی ارایه خواهیم کرد. همچنین نگاهی به درخت تصمیم - یکی از ابزارهای داده کاوی که نویسنده برای پیاده سازی آنرا برگزیده است - خواهیم داشت. در ادامه خوشه بندی را شرح و سعی خواهیم کرد الگوریتم *EM* در خوشه بندی را در غالب دستورات *SQL* بنویسیم...

- مقدمه
- کلاسه بندی داده ها
- درخت تصمیم و انواع آن
- پیاده سازی با *SQL*
- خوشه بندی داده ها
- الگوریتم *EM*
- پیاده سازی با *SQL*
- خلاصه

٤-١. مقدمه

هدف کلاسه‌بندی داده‌ها، سازماندهی و تخصیص داده‌ها به کلاس‌های مجزا می‌باشد. در این فرآیند بر اساس داده‌های توزیع شده، مدل اولیه‌ای ایجاد می‌گردد. سپس این مدل برای طبقه‌بندی داده‌های جدید مورد استفاده قرار می‌گیرد، به این ترتیب با بکارگیری مدل بدست آمده، تعلق داده‌های جدید به کلاس معین قابل پیشگویی می‌باشد. کلاسه‌بندی در مورد مقادیر گسسته و پیشگویی آنها به‌کار می‌رود. *[HK 2000 chV]*

در کلاسه‌بندی، مدل ایجاد شده بر پایه‌ی یک سری داده‌های آموزشی، (اشیا داده‌هایی که بر چسب کلاس آنها مشخص و شناخته شده است) حاصل می آید. مدل بدست آمده در اشکال گوناگون مانند قوانین کلاسه‌بندی، درخت‌های تصمیم، فرمول‌های ریاضی و شبکه‌های عصبی قابل نمایش می‌باشد. یک درخت تصمیم، یک ساختار درختی سلسله مراتبی است که هر گره‌ی آن نشان دهنده‌ی تست مقدار خصیصه بوده، هر انشعاب یک خروجی تست را نمایش می‌دهد. برگ‌های درخت نیز، کلاس‌ها یا توزیع‌های کلاس را مشخص می‌نمایند. این نوع درخت تصمیم قابل تبدیل به قوانین کلاسه‌بندی می‌باشد. از کلاسه‌بندی می توان برای پیشگویی کلاس اشیا داده‌ها استفاده کرد.

در برخی موارد نیز افراد ترجیح میدهند مقدار یک خصیصه و نه کلاس آن را پیشگویی نمایند که به یافتن مقدار یک خصیصه، پیشگویی اطلاق می‌گردد. در هر حال پیشگویی، تخمین مقدار و برچسب کلاس را با هم در بر می گیرد. کلاسه‌بندی و پیشگویی با استفاده از تحلیل ارتباط[1]، خصیصه‌هایی را که در فرآیند مورد نظر، بی‌تاثیر و قابل حذف می‌باشند، شناسایی می‌کنند.

به عنوان مثال فرض کنید مدیر فروشگاهی در نظر دارد مجموعه‌ی بزرگی از داده‌ها را بر اساس میزان فروش به زیاد، متوسط و کم طبقه‌بندی کند. وی می‌بایست مدلی ایجاد کند که بر اساس خصیصه‌های کالا مانند قیمت، مارک، محل ساخت و نوع کالا، کلاس مربوط به آن نوع کالا را تعیین نماید. طبقه‌بندی نهایی می‌بایست به طور ماکزیمال هر کلاسی را از دیگری تشخیص داده، تصویر سازماندهی شده‌ای از داده‌ها را به نمایش در آورد. اگر خروجی به صورت یک درخت تصمیم باشد، خصیصه‌ها به ترتیب اولویت بررسی شده، بقیه خصیصه‌ها در اولویت بعدی بررسی می‌گردند. چنین مدلی در طرح‌های آتی فروشگاه می تواند بسیار موثر واقع شود. *[HK2000ch1]*

از کاربردهای کلاسه‌بندی می توان بازاریابی، تشخیص بیماری، تحلیل اثرات معالجه، تشخیص خرابی در صنعت و تعیین اعتبار را نام برد. *[HK 2000 ch7]*

٤-٢. انواع روش‌های کلاسه‌بندی

کلاسه‌بندی به روش‌های زیر انجام‌پذیر است:

- استنتاج بر اساس درخت تصمیم
- طبقه‌بندی بیز
- الگوریتم‌های ژنتیک
- شبکه‌های عصبی
- مجموعه‌های فازی
- طبقه‌بندی بر اساس وابسته‌سازی

[1] Relevance Analysis

٤-٣. مراحل یک الگوریتم کلاسه‌بندی

الگوی عمومی برای الگوریتم‌های آموزش از طریق مثال با فرایند کلاسه‌بندی به سه مرحله تقسیم می‌شوند: *[Gams 2006]*

۱. پیش‌پردازش داده‌ها

۲. ساخت و ارزیابی قوانین کلاسه‌بندی و هرس کردن قوانین اضافی که هدف ما می‌باشد.

۳. کلاسه‌بندی نمونه‌های جدید.

٤-٤. ارزیابی روش‌های کلاسه‌بندی

ارزیابی روش‌های کلاسه‌بندی با معیارهای زیر انجام می‌پذیرد:

- میزان دقت[۲] درپیشگویی، از جمله معیارهای ارزیابی روش‌های مذکور در کلاسه‌بندی می‌باشد که میزان قابلیت و توانایی یک مدل را در پیشگویی صحیح برچسب یک کلاس، مشخص می‌کند.
- سرعت[۳] و توسعه‌پذیری از نظر زمانی که برای ایجاد یک مدل و زمان استفاده از آن مدل لازم می‌باشد، از معیارهای دیگر ارزیابی روش در کلاسه‌بندی می‌باشد.
- قوی بودن[٤] معیار مهمی است که میزان توانایی یک مدل را در برخورد با نویز و مقادیر حذف شده تعیین می کند.
- توسعه‌پذیری[٥] معیار دیگری است که از نقطه نظر میزان کارایی در بانک‌های اطلاعاتی بزرگ و نه داده‌های مقیم در حافظه مورد بررسی قرار می گیرد.
- قابل تفسیر بودن[٦] یعنی میزان و سطح درک ایجاد شده توسط مدل از دیگر مواردی است که می‌بایست در بررسی روش‌های کلاسه‌بندی در نظر گرفت.
- شکل قوانین و نحوه نمایش آنها از جمله سایز درخت تصمیم و فشردگی و پیوستگی قوانین، معیار دیگری است که در بررسی روش‌های کلاسه‌بندی موثر می‌باشد. *[HK 2000ch7]*

٤-٥. روش درخت تصمیم در کلاسه‌بندی

یک درخت تصمیم یک ساختار سلسله مراتبی می‌باشدکه در آن، گره‌های میانی برای تست یک خصیصه[۷] به کار می روند. شاخه‌ها نشانگر خروجی تست بوده، برگ‌ها برچسب کلاس[۸] و یا توزیع بر چسب کلاس را مشخص می‌نمایند. نکات اساسی برای هر درخت تصمیم به شرح زیر هستند: *[Kennedy97]*

ملاک استفاده شده برای ساخت درخت چه عواملی هستند؟ یعنی کدام متغیر باید برای شکستن انتخاب گردد و این متغیر چگونه باید شکسته شود؟

ملاک برای متوقف کردن رشد درخت کدام‌ها هستند؟ یعنی چه موقعی باید عمل شاخه شاخه شدن یک نود باید متوقف شود؟

---

[2] Accuracy
[3] Speed
[4] Robustness
[5] Scalability
[6] Interpreted
[7] Attribute
[8] Class Label

چگونه باید شاخه‌های درخت بدست آمده هرس شوند تا بیشترین کارایی را در کلاسه‌بندی داشته باشیم؟

۴-۵-۱. انواع درخت‌های تصمیم

درخت‌های تصمیم بر دو نوعند:

۱. درختان تصمیم دودویی که در هر نود، فقط دو شاخه‌ی انشعابی از آن را داریم، مانند *CART*.

۲. درختان تصمیم خطی[9]، که هر نود می‌تواند به چند شاخه منشعب شود، مثل *CHAID*. که اگر ضریب انشعاب دو باشد به درخت تصمیم دودویی تبدیل می‌شود.

۴-۵-۲. درختهای رگرسیون و کلاسه بندی[10] [Kennedy97]

بهترین ملاک شکست در یک نود مشخص از این درخت، از رابطه‌ی زیر بدست می‌آید:

$$S = \mathrm{Max}_i \left(2P_l P_R \sum_{j=1}^{\text{Classes}} \left| P(j/\text{LeftChild}) - P(j/\text{RightChild}) \right| \right)$$

که در آن S بهترین قانون شکست می‌باشد و

$$P_L = \frac{\text{مجموع تعداد الگوهای موجود در بچه‌ی چپ}}{\text{مجموع تعداد الگوهای موجود در داده‌های آموزشی}}$$

$$P_R = \frac{\text{مجموع تعداد الگوهای موجود در بچه‌ی راست}}{\text{مجموع تعداد الگوهای موجود در داده‌های آموزشی}}$$

$$P\ (j/LeftChild) = \frac{\text{مجموع تعداد الگوهای کلاس } j \text{ در بچه‌ی چپ}}{\text{مجموع تعداد الگوهای در نود فعلی}}$$

$$P\ (j/RightChild) = \frac{\text{مجموع تعداد الگوهای کلاس } j \text{ در بچه‌ی راست}}{\text{مجموع تعداد الگوهای درنود فعلی}}$$

۴-۵-۳. نحوه‌ی هرس کردن درخت:

تابع $g(t)$ (تابع قوت) رابرای هر نود غیر برگ حساب می‌کنیم. سپس می‌توان زیر درختی را که دارای کمترین $g(t)$ می‌باشد از درخت هرس کرد:

$$g(t) = \frac{R(t) - \sum_{i \in T'} R(i)}{|T'| - 1}$$

که در آن:

$$R(t) = \left(\frac{\text{مجموع الگوهای موجود در گره‌ی } T}{\text{مجموع الگوهای موجود در داده‌های آموزشی}} \right) \left(\frac{\text{تعداد الگوهای با کلاس } j \text{ در گره‌ی } T}{\text{مجموع الگوهای موجود در گره‌ی } T} \right) \quad Maxi$$

[9] Linear Decision Tree
[10] Classification and Regression Trees (CART)

تعداد گره‌های برگ در زیر درخت با ریشه‌ی $T' = T$

۴-۵-۴. درخت تصمیم مربع خی[11] [Kennedy97]

نحوه‌ی شکستن گره‌ها: کدام متغیر باید شکسته شود و مقدار شکست چقدر است؟ در این الگوریتم جواب این سوال بر پایه‌ی توزیع $\chi 2$ متغیرهای ورودی و خروجی می‌باشد. صورت کلی این شکست به صورت زیر می‌باشد:

```
if Variable < SplitValue1 then branch#1
    else if  Variable < SplitValue2 then branch#2
```

اگر فقط دو شاخه داشته باشیم، به همان درخت دودویی تبدیل می‌شود، در غیر این‌صورت درخت خطی را داریم. توجه داشته باشیم که مقادیر، باید به صورتی تقسیم شوند که هر مقدار به طور یکتا در یک شاخه جای بگیرد و هر مقدار حتما به‌یک شاخه بتواند نسبت داده شود. (البته ممکن است که چندین مقدار به‌یک شاخه نسبت داده شوند) حال باید به این سوال جواب داده شود:

حال که نود کاندیدا برای شکست انتخاب شد، نحوه‌ی تقسیم‌بندی مقادیر به طبقه یا طبقه‌ها (البته توجه داشته باشیم که تعداد شاخه‌ها مشخص است) چگونه باید باشد؟ ما این‌کار را بر اساس روش تقسیم یک بازه به زیر بازه‌های مساوی[12] انجام می‌دهیم.

۴-۵-۵. نحوه محاسبه‌ی $\chi 2$:

$$\chi^2 = \sum_{i \in \text{categories}} = \frac{(t_0[i] - e_0[i])^2}{e_0[i]^2} + \frac{(t_1[i] - e_1[i])^2}{e_1[i]^2}$$

که در آن:

t_0، تعداد کل صفرها، t_1، تعداد کل یک‌ها، $t[i]$، تعداد اعضای طبقه‌ی i، $e_0[i]$، صفرهای طبقه‌ی i و $e_1[i]$، یک‌های طبقه‌ی i می‌باشد.

۴-۵-۶. شرط پایان[13]

زمانی که همه‌ی متغیرها در یک گره، مقادیر زیر یک مقدار آستانه‌ای[14] خاصی را بگیرند، الگوریتم را متوقف می‌کنیم. (این مقدار اگر بخواهد بهینه باشد باید از طریق آزمایش و خطا آن را پیدا کنیم.)

درخت *CHAID* نیز می‌تواند هرس شود.

٤-٦. خوشه‌بندی

پدیده‌ی خوشه‌بندی که یکی دیگر از اهداف داده‌کاوی می‌باشد، به فرآیند تقسیم مجموعه‌ای از داده‌ها (یا اشیا) به زیر کلاس‌هایی با مفهوم خوشه اتلاق می‌شود. به این ترتیب یک خوشه، یک سری داده‌های مشابه می‌باشد که همانند یک گروه واحد رفتار می‌کنند. لازم به ذکر است خوشه‌بندی همان کلاسه‌بندی است، با این تفاوت که کلاس‌ها از پیش‌تعریف‌شده و معین نمی‌باشند و عمل گروه‌بندی داده‌ها بدون نظارت انجام می‌گیرد. *[HK 2000 ch8]*

فرض کنید که مجموعه داده‌های X موردنظر ما از نقاط داده‌ای (یا مترادف آن اشیا، موارد، الگوها، تراکنش‌ها، گروه‌ها یا رکوردها)، در فضای ویژگی A تشکیل شده باشند. یعنی $x_i = (x_{i1}, x_{i2}, \ldots x_{id}) \in A$ که در ان $i=1..N$ و هر جز $x_{il} \in A_{ll}$ یک داده عددی یا

---

[11] Chi - Squared Automatic Iteration Decision tree) CHAID
[12] Equal Population Bins
[13] Stop Condition
[14] threshold

ویژگی طبقه‌بندی شده‌ی اسمی باشد. این فرمت داده- ویژگی مفهوما متناظراست با یک ماتریس $N \times D$. هدف خوشه‌بندی پیدا کردن سگمنت‌هایی در ماتریس فوق می‌باشد، که اجتماع همه‌ی آن‌ها کل ماتریس باشد و دو بدوی آن‌ها نقطه اشتراکی نداشته باشند.

$$X = C_1 \upsilon C_2 \upsilon \ldots \upsilon C_k \quad , C_{j1} \cap C_{j2} = \emptyset$$

بر خلاف کلاسه‌بندی و پیشگویی که اشیا داده‌ها را براساس کلاس‌ها تحلیل می کنند، خوشه‌بندی اشیا داده‌ها را بدون در نظر گرفتن برچسب‌های کلاس، تحلیل و آنالیز می نماید. عمدتا برچسب کلاس‌ها در داده‌های آموزشی به آسانی مشخص نیست زیرا این کلاس‌ها شناخته شده نمی‌باشند. خوشه‌بندی گاهی برای تعیین و تولید چنین برچسب هایی بکار می رود. اشیای خوشه‌بندی شده بر اساس اصل ماکزیمم شباهت بین اعضای هر کلاس و مینیمم شباهت بین کلاس‌های مختلف گروه‌بندی می‌شوند، یعنی خوشه‌ها به‌گونه‌ای تنظیم می‌شوند که اشیای داخل هر خوشه بیشترین شباهت را با یکدیگر داشته باشند. هر خوشه به عنوان یک کلاس می‌باشد که قوانین از آن مشتق می شوند. ضمنا خوشه‌بندی می تواند امکان طبقه‌بندی تشکیلات[۱۵] را فراهم کند، یعنی سازماندهی مذکور، به صورت سلسله‌مراتبی از کلاس‌هاست که هر کلاس شامل حوادث مشابه یکدیگر می‌باشد.

۴-۶-۱. تعریف فرآیند خوشه‌بندی

خوشه‌بندی فرآیند تقسیم یک‌سری از داده‌ها یا اشیا به زیر کلاس‌هایی به نام خوشه می‌باشد که توسط این رویه فهم ساختار داده‌ها و گروه‌بندی آنها ساده می گردد. هر خوشه، شامل یک‌سری داده‌های مشابه می‌باشد که به صورت یک گروه رفتار می کنند. می‌توان خوشه‌بندی را به صورت کلاسه‌بندی تعریف کرد، با این تفاوت که کلاس‌ها و برچسب آنها از پیش تعریف شده نبوده، عمل کلاسه‌بندی بدون نظارت انجام می گیرد.

۴-۶-۲. کیفیت خوشه‌بندی

کیفیت خوشه‌بندی بر اصول زیر متکی است:

- داده‌های داخل یک کلاس بیشترین تشابه وداده‌های کلاس‌های متمایز بیشترین تفاوت را دارا باشند.
- استفاده از معیارهای صحیحِ یافتن تشابهات در متدها و پیاده‌سازی آن روش‌ها.
- کیفیت خوشه‌بندی، بر توانایی استخراج برخی الگوهای مخفی موجود در داده‌ها متکی است.
- چگونگی تعریف و نمایش خوشه‌های انتخابی نیز، معیار مهم دیگر در کیفیت خوشه‌بندی می‌باشد. *[HK2000ch8]*

در داده‌کاوی و در زمینه‌ی خوشه‌بندی می‌بایست ملاحظات زیر را منظور کرد:

- توسعه‌پذیری
- بتوان خوشه‌ها را با هر شکل دلخواه استخراج نمود.
- در تصمیم گیری پارامترهای داخلی، نیاز به دانش زمینه‌ی حداقل باشد.
- عدم آسیب‌پذیری در مواجهه با خطا.
- عدم حساسیت به ترتیب رکوردهای ورودی و یا عدم تاثیر مخرب رکوردهای آتی بر رکوردهای فعلی.
- پشتیبانی انواع مختلف خصیصه‌ها.
- پشتیبانی ابعاد بزرگ داده‌ها.

---

[۱۵] Taxonomy Formation

۴-۶-۳. روش ها و الگوریتم‌های خوشه‌بندی

دو دسته‌ی کلی از الگوریتم‌های خوشه‌بندی، سلسله‌مراتبی[16] و تفکیکی[۱۷] می‌باشند. الگوریتم‌های سلسله مراتبی خوشه‌ها را به تدریج می‌سازند (مانند کریستال‌ها رشد می‌کنند) ولی الگوریتم‌های تقسیم کننده مستقیما خوشه‌بندی را انجام می‌دهند. آنها سعی می‌کنند که خوشه‌ها را با جایگذاری مجدد نقطه‌ها بین زیرمجموعه‌ها کشف کنند. در یک تقسیم‌بندی جزیی‌تر این الگوریتم‌ها به صورت زیر دسته‌بندی می‌گردند:

۴-۶-۴. الگوریتم‌های تفکیک

یکی از انواع الگوریتم‌های خوشه‌بندی است که درابتدا مجموعه‌ی داده‌ها را به بخش‌هایی تبدیل کرده، سپس با استفاده از برخی معیارها آن دسته‌بندی را مورد ارزیابی قرار می‌دهد و در صورت لزوم در دسته‌بندی اولیه تغییراتی ایجاد می‌نماید. رایج‌ترین الگوریتم‌های خوشه‌بندی در این دسته قرار می گیرند.

این الگوریتم‌ها داده‌ها را به چندین زیر مجموعه تقسیم می‌کنند. به علت این‌که چک کردن همه‌ی زیر مجموعه‌های ممکن، امکانپذیر نیست. تابع‌های مکاشفه‌ای حریصانه‌ی خاصی به کار گرفته می‌شوند. در این الگوریتم‌ها به صورت تکراری نقاط بین k خوشه جابجا می‌شوند تا در نهایت به بهترین خوشه ممکن نسبت داده شوند. بر خلاف متدهای سلسله مراتبی که خوشه‌ها بعد از ساخته شدن بازبینی نمی‌شود، این الگوریتم‌ها مرتبا خوشه‌ها را به منظور بهبود بخشیدنشان تغییر می‌دهند، به همین دلیل در این روش‌ها نهایتا خوشه‌هایی با کیفیت بالا خواهیم داشت.

نقش اصلی را در این متدها روش‌های آماری و احتمالی دارند، به همین دلیل خوشه‌های تولید شده دارای قابلیت تفسیرپذیر بالایی بوده و بسیار مورد قبول می‌باشند.

بهینه‌سازهای تکراریِ الگوریتم‌های افراز به دو نوع تقسیم می‌شوند: *k-means , k-medoids*. که در اولی یکی از نقاط خوشه، معرفی کننده‌ی آن خوشه است ولی در دومی، مرکز ثقل آن خوشه. ما در اینجا تمایل داریم که با حالت کاراتر این دو، یعنی *k-means* کار کنیم. الگوریتم *Expectation-Maximization* که در سال ۱۹۹۷ ارایه شده است *[Mitchell97]*، یک چارچوب[۱۸] کلی برای الگوریتم‌های آماری می‌باشد که‌یکی از حالت‌های خاص آن الگوریتم *k-means* محبوب می‌باشد. این الگوریتم با توزیع نرمال کار می‌کند.

۴-۶-۵. الگوریتم‌های سلسله‌مراتبی[۱۹]

نوع دیگری از الگوریتم‌های خوشه‌بندی است که در ابتدا با در نظر گرفتن برخی معیارها به تجزیه‌ی سلسله مراتبی داده‌ها می‌پردازد و سپس با روش‌های اجماع و تقسیم تغییراتی در دسته‌بندی اولیه ایجاد می‌نماید. الگوریتم‌های *BIRCH* و *CHRE* در این گروه جای دارند. الگوریتم‌های سلسله مراتبی به دو گونه تقسیم می‌شوند. الگوریتم‌های تجمیعی[۲۰] (پایین به بالا) و تقسیمی[۲۱] (بالا به پایین). در خوشه‌بندی تجمیعی، کار با خوشه‌هایی با یک داده شروع می‌شود (تعداد خوشه‌ها در ابتدا به اندازه‌ی تعداد داده‌های موجود می‌باشد). در هر مرحله دو یا چند خوشه‌ی مناسب با هم ترکیب شده و خوشه‌ی جدیدی را بوجود می‌آورند. در خوشه‌بندی تقسیمی عمل خوشه‌بندی با یک خوشه شروع می‌شود. این خوشه به صورت بازگشتی به دو یا چند خوشه تقسیم می‌گردد و به همین ترتیب عمل خوشه‌بندی ادامه پیدا می‌کند.

---

[16] hierarchical
[17] Partitioning
[18] Framework
[۱۹] Hierarchical Algorithm
[20] agglomerative
[21] devise

برای هر دو نوع از الگوریتم‌های بالا ما نیاز به‌یک شرط پایانی داریم این شرط اغلب رسیدن به k خوشه می‌باشد.

ادغام یا تقسیم خوشه‌ها به شباهت یا عدم شباهت عناصرخوشه‌ها وابسته است یکی از مهمترین ملاک‌های شباهت، فاصله‌ی بین عناصر دو خوشه باشد. یعنی، فاصله‌ی دو زیر مجموعه از یک خوشه (برای هر ترکیب دوتایی از عناصر آن زیرمجموعه از خوشه‌ها) محاسبه می‌گردد.

برای بدست آوردن فاصله‌ی دو خوشه، ابتدا x درصد از اعضای خوشه اول و y درصد از اعضای خوشه دوم انتخاب می‌گردند (معمولا x و y را با هم برابر می‌گیریم) سپس این دو زیر مجموعه را با هم *Cross-join* کرده و فاصله‌ی تک تک عناصر را محاسبه می‌کنیم. فاصله‌ی دو زیر مجموعه از رابطه‌ی زیر بدست می‌آید:

$$d(C_1, C_2) = \text{Operation}\{d(x, y) | x \in C_1, y \in C_2\}$$

که در آن *Operation* می‌توانند یکی از عملگرهای *minimum* یا *maximum* یا *average*باشد که مثلا اگر *Operation = minimum* باشد، الگوریتم *SLINK* را خواهم داشت، اگر *Operation = maximum* باشد، الگوریتم *CLINK* را خواهیم داشت و اگر *Operation = average* باشد، الگوریتم *Voorhees* را خواهیم داشت.

روش‌هایی که مبتنی بر فاصله‌ی اقلیدسی می‌باشند (مثل روش‌های بالا) خوشه‌هایی با شکل منظم[۲۲] می‌سازند، ولی در عمل بسیاری از خوشه‌ها از منحنی‌هایی در اشکالشان استفاده می‌شود. برای این منظور چند الگوریتم بوجود آمده است ولی از آنجایی که هر کدام از این الگوریتم‌ها ورودی خاص خود را دارند و بر روش خاصی تکیه می‌کنند، قابل کلی[۲۳] کردن نیستند. از این قسمت نیز به دلیل کلی نبودن الگوریتم‌های آن صرف‌نظر می‌شود. (لااقل در نسخه‌های اولیه).

۴-۶-۶. روش‌های متکی برچگالی[۲۴]

یکی از روش‌های خوشه‌بندی است که مجموعه‌ی داده‌ها را بر اساس معیارهایی همچون توابع همسایگی[۲۵] و چگالی دسته‌بندی کرده، مورد ارزیابی قرار می‌دهد *OPTICS , CLIQUE , DBScon* از جمله مثال‌های این نوع خوشه‌بندی می‌باشند. این الگوریتم‌ها از لحاظ تشکیل شکل‌های نامنظم الگویی، انعطاف پذیرتر هستند ولی اغلب فقط بر روی داده‌های عددی و با ابعاد پایین کار می‌کنند.

۴-۶-۷. روش‌های متکی بر گرید[۲۶]

در این متدها، داده‌ها ابتدا خلاصه شده و سگمنت می‌شوند، سپس عمل افراز[۲۷] روی فضاهای بدست آمده انجام می‌پذیرد و نهایتا فضاهای خوشه‌بندی شده منجر به داده‌های خوشه‌بندی شده می‌شود. هدف از این عملیات بالا بردن کارایی می‌باشد چون دیگر لازم نیست که با کل فضاهای داده‌ای کار کنیم و فقط باید به سگمنت‌ها کار کنیم. بعد از بدست آمدن سگمنت‌ها، کل کار به همان صورت قبلی دنبال می‌شود. یعنی می‌توان این متدها را معادل سایر متدها منتها با یک مرحله‌ی پیش‌پردازش[۲۸] در نظر گرفت.

۴-۶-۸. روش‌های متکی بر مدل[۲۹]

یک نوع از روش‌های خوشه‌بندی است که برای هر خوشه، مدلی فرضی را در نظر می‌گیرد و هدف آن یافتن مناسب‌ترین مدل برای هر خوشه می‌باشد. *Cod Web Denclue* و *AutoClass* مثال‌هایی از این نوع روش خوشه‌بندی می‌باشند. *[Berkhin2002]*

---

[22] Proper Shape
[23] Generalize
[۲۴] Density –Based method
[۲۵] Connectivity
[۲۶] Grid-Based method
[27] Partitioning
[28] Preprocessing
[۲۹] Model - Based method

۴-۶-۹. تکنیک‌های خوشه‌بندی دیگر

الگوریتم‌های بسیار دیگری وجود دارند، تعدادی از انها به منظور برطرف کردن نیازهای کاربردی خاصی بوجود امده‌اند. خوشه‌بندی‌های بر اساس محدودیت[۳۰] از این گونه‌اند. تعدادی فقط از لحاظ تئوری مطرح شده‌اند و کاربردهای ان بیشتر در زمی‌نه‌های دیگر بوده و به این قسمت نیز کشانده شده‌اند.

٤-۷. دسته‌بندی ویژگی‌های الگوریتم‌های خوشه‌بندی

در حالت کلی ویژگی‌هایی که این الگوریتم‌ها را از همدیگر متمایز می‌سازند عبارتند از: *[berkhin2002]*

- نوع ویژگی‌هایی که الگوریتم می‌تواند با آن‌ها کار کند. ما علاقمند به الگوریتم‌هایی هستیم که با داده‌های عددی و داده‌های طبقه‌بندی شده (در حالت کلی در همان دسته‌ی داده‌های عددی می‌گنجند)، بتوانند کار کنند.
- حجم داده‌هایی که می‌توانند با آن کار کنند. ما علاقمند به الگوریتم‌هایی هستیم که بتوانند با حجم بسیار بالای داده‌های انفجاری عصر حاضر بتوانند کار کنند.
- ابعاد داده‌ای که می‌توانند با آن کار کنند. الگوریتم‌هایی مطلوب ما هستند که داده‌های با ابعاد بالا را نیز پشتیبانی می‌کنند.
- پیچیدگی زمانی. تعدادی از الگوریتم‌ها با توجه به حجم بالای اطلاعات امروزه کارایی خود را از دست داده و در مدت زمان معقول و قابل قبول قادر به جوابگویی نیستند و بایگانی شده‌اند.
- پیچیدگی‌الگوریتم : ما مایل به استفاده از الگوریتم‌هایی هستیم که با اعمال رابطه‌ای بانک‌های داده‌ای با موتور *RDBMS* قابل پیاده‌سازی باشند.

۴-۸. پیاده‌سازی چارچوب[۳۱] کلی الگوریتم خوشه‌بندی تفکیکی، بر پایه‌ی SQL

در اینجا الگوریتم *EM*[۳۲] را تشریح کرده و به ترجمه‌ی آن به *SQL* استاندارد می‌پردازیم *[Ordonez2000]*

۴-۸-۱. ورودی‌های الگوریتم

ورودی‌های این الگوریتم بشرح زیر می‌باشند :

- K: تعداد خوشه‌ها
- Y : مجموعه‌ای از داده‌های P بعدی که الگوریتم را روی آنها اجرا می‌کنیم : $Y= \{y1,y2,...,yn\}$
- P: تعداد ابعاد هر داده
- E : یک تلورانس برای *LogLikelihood*.
- *MaxIteration* : ماکزیمم مقدار تکرار حلقه‌ی خارجی.

۴-۸-۲. خروجی‌های الگوریتم

انتظار داریم که این الگوریتم خروجی‌های زیر را به ما بدهد:

- $C[p\times k]$: ماتریس میانه‌های خوشه‌ها (k خوشه p بعدی)

---

[30] Constraint-based
[31] Framework
[32] Expectation–Maximization

- $R[p \times p]$: کوواریانس هر کدام از ابعاد (کواریانس‌ها بین خوشه‌ها مشترک است.) این ماتریس قطری است.
- $W[k \times 1]$: وزن‌های خوشه‌ها.
- $X[n,k]$: ماتریس احتمال عضویت هر داده در خوشه‌ها.

۴-۸-۳. مدل احتمال به کار رفته

تابع چگالی احتمال که در این الگوریتم بکار می‌رود به صورت زیر است:

: تابع چگالی احتمال توزیع نرمال در یک بعد $$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{-(x-\mu)^2}{2\sigma^2}}$$

که در آن :

$$\mu = E[x] = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma^2 = E[(x-\mu)^2] = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu)^2$$

که این تابع برای داده‌های P بعدی (x تعداد بعدهایش، p تاست)، بصورت زیر درمی‌آید:

$$P(x) = \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

که در آن :

یک بردار P بعدی است: $\mu = E[x]$

ماتریس کوواریانس‌ها که $P \times P$ بعدی است $\Sigma =$

ما در الگوریتم‌های EM با پارامترهای p بعدی سر و کار داریم، پس از تابع چگالی فوق استفاده می‌کنیم. بصورت ساده‌تر، این تابع را می‌توان بصورت زیر نوشت :

$$P(x) = \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} e^{-\frac{1}{2}\delta^2}$$

که δ را فاصله‌ی ماهانوبیس گوییم و داریم :

$$\delta^2 = (x-\mu)^T \Sigma^{-1}(x-\mu)$$

و چون در اینجا داده‌های ما ترکیبی از k خوشه از داده‌های p بعدی هستند، در نتیجه تابع احتمال مدل ترکیبی نرمال، بصورت زیر خواهد بود :

$$P(x) = \sum_{i=1}^{k} w_i P(x \mid i)$$

$P(x|i)$: توزیع نرمال برای هر کلاستر

wi: وزنی از داده‌های کلی که این کلاستر آن را معرفی می‌کند. $\left(\sum_{i=1}^{k} w_i = 1\right)$

۴-۸-۴. الگوریتم EM

حال که شناختی از پارامترهای مختلف EM بدست آوردیم، خود الگوریتم را در اینجا ذکر می‌کنیم :

```
1. INITIALIZE: Set Initial values for C,R,W (Rendom or approximate solution
from sample).
2. WHILE change in Log Likelihood llh is greater than ϵ and MaxIteration has
not been reached Do E and M Steps.
```

```
E step
C'=0, R'=0, W'=0, llh=0
For i =1 to n
      Sumpi = 0
      For j=1 to k
            δij = (yi - cj)T R-1 (yi - cj)
            Pij = [wi/((2π)p/2|R|1/2)] exp [-1/2 δij]
            Sumpi = sumpi + pij
      End for
      Xi = pi/sumpi
      llh = llh + ln(sumpi)
      C' = C'+ yi xiT
      W' = W' + x i
End for

M step
For j=1 to k
      Cj = C'j / W'j
      For i=1 to n
            R' = R' + (yi - cj) xij (yi - cj)T
End for
R = R' / n
W = W' / n
```

الگوریتم محبوب و مشهور *k-Means* یک حالت خاص از *EM* است وقتی که *R,W* ثابت بوده و برابر باشند با :

$$W=1/k$$
$$R=I$$

۴-۹. خلاصه

هدف این فصل پیشگویی، پیش‌بینی و دریافت مقدار یک خصیصه بر اساس خصیصه‌های دیگر می‌باشد. بر اساس داده‌های توزیعی، در ابتدا یک مدل ایجاد می گردد، سپس از این مدل در پیشگویی مقادیر ناشناخته استفاده می‌شود. در داده‌کاوی، کلاسه‌بندی به پیش بینی مقادیر گسسته، و پیشگویی به تخمین مقادیر پیوسته اطلاق می‌شود. در فرآیند کلاسه‌بندی، اشیا موجود به کلاس‌های مجزا با مشخصه‌های تفکیک‌شده (ظروف جداگانه) طبقه‌بندی و به صورت یک مدل معرفی می گردند. سپس با در نظر گرفتن ویژگی‌های هر طبقه، شی جدید به آنها تخصیص یافته، برچسب و نوع آن پیشگویی می گردد.

No man is an island; entire of itself, every man is a piece of continent a part of main.

- John Donne

تکنولوژیهای وب سرویس

Web Service Technology

هیچ کس یک جزیره نیست، به تنهایی کامل باشد، هر فرد تکه ای از قاره است، قسمتی از کل.

Web Services

فصل پنجم
سرویسهای وب

در این فصل نگاهی کلی به پیشرفت های اخیر در تکنیک های وب سرویس ها خواهیم داشت. در ادامه به معرفی برخی تکنولو ژیهای مهم وب سرویس مانند *UDDI, WSDL, ebXML, SOAP, ...* خواهیم انداخت و ...

- مقدمه
- وب سرویس
- توصیف، کشف و مجتمع سازی یکنواخت منابع *UDDI*
- زبان توصیف وب سرویس *WSDL*
- *ebXML*
- *XML*
- خلاصه

۵-۱. مقدمه

کسانی که با صنعت *IT* آشنایی دارند حتماً نام وب سرویس را شنیده اند . برای مثال ، بیش از ٦٦ درصد کسانی که در نظر سنجی مجله *InfoWorld*شرکت کرده بودند بر این توافق داشتند که وب سرویس ها مدل تجاری بعدی اینترنت خواهند بود . به علاوه گروه گارتنر پیش بینی کرده است که وب سرویس ها کارآیی پروژه های *IT* را تا ۳۰ در صد بالا می برد . اما وب سرویس چیست و چگونه شکل تجارت را در اینترنت تغییر خواهد داد ؟

برای ساده کردن پردازش های تجاری ، برنامه های غیر متمرکز باید با یکدیگر ارتباط داشته باشند و از داده های اشتراکی یکدیگر استفاده کنند . قبلاً این کار بوسیله ابداع استاندارد های خصوصی و فرمت داده ها به شکل مورد نیاز هر برنامه انجام می شد . اما دنیای وب و – *XML* تکنولوژی آزاد برای انتقال داده – انتقال اطلاعات بین سیستم ها را افزایش داد . وب سرویس ها نرم افزارهایی هستند که از *XML* برای انتقال اطلاعات بین نرم افزارهای دیگر از طریق پروتکل های معمول اینترنتی استفاده می کنند . به شکل ساده یک وب سرویس از طریق وب اعمالی را انجام می دهد و نتایج را به برنامه دیگری می فرستد . این یعنی برنامه ای که در یک کامپیوتر در حال اجراست ، اطلاعاتی را به کامپیوتری می فرستد و از آن درخواست جواب میکند ، برنامه ای که در آن کامپیوتر دوم است کارهای خواسته شده را انجام می دهد و نتیجه را بر روی ساختارهای اینترنتی به برنامه اول بر می گرداند . وب سرویس ها می توانند از پروتکل های زیادی در اینترنت استفاده کنند اما بیشتر از *HTTP* که مهم ترین آنهاست استفاده می شود .

وب سرویس هر نوع کاری می تواند انجام دهد . برای مثال در یک برنامه می تواند آخرین عنوان های اخبار را از وب سرویس *Associated Press* بگیرد یا یک برنامه مالی می تواند آخرین اخبار و اطلاعات بورس را از وب سرویس بگیرد . کاری که وب سرویس انجام می دهد می تواند به سادگی ضرب ۲ عدد یا به پیچیدگی انجام کلیه امور مشترکین یک شرکت باشد .

وب سرویس دارای خواصی است که آن را از دیگر تکنولوژی و مدل های کامپیوتری جدا می کند ، *Paul Flessner*، نایب رییس مایکروسافت در *dot NET Enterprise Server* چندین مشخصه برای وب سرویس در یکی از نوشته هایش ذکر کرده است :

- وب سرویس ها قابل برنامه ریزی هستند . یک وب سرویس کاری که می کند را در خود مخفی نگه می دارد وقتی برنامه ای به آن اطلاعات داد وب سرویس آن را پردازش می کند و در جواب آن اطلاعاتی را به برنامه اصلی بر می گرداند .
- وب سرویس ها بر پایه *XML* بنا نهاده شده اند. *XML* و *XML* های مبتنی بر پروتکل دسترسی ساده شی[1] تکنولوژی هایی هستند که به وب سرویس این امکان را می دهند که با دیگر برنامه ها ارتباط داشته باشد حتی اگر آن برنامه ها در زبانهای مختلف نوشته شده و بر روی سیستم عامل های مختلفی در حال اجرا باشند .
- همچین وب سرویس ها خود ، خود را توصیف می کنند . به این معنی که کاری را که انجام می دهند و نحوه استفاده از خودشان را توضیح می دهند . این توضیحات به طور کلی در زبان تعریف سرویس[2] نوشته می شود. *WSDL* یک استاندارد بر مبنای *XML* است . به علاوه وب سرویس ها قابل شناسایی هستند به این معنی که برنامه نویس می تواند به دنبال وب سرویس مورد علاقه در دایرکتوری هایی مثل توصیف، کشف و مجتمع سازی یکنواخت منابع[3] جستجو کند. *UDDI* یکی دیگر از استاندارد های وب سرویس است.

[1] Simple Object Access Protocol (SOAP)
[2] Web Service Definition Language (WSDL)
[3] Universal Description, Discovery and Integration (UDDI)

۲-۵. وب سرویس ها : تکنولوژیهای استاندارد و وابسته

امروزه واژه وب سرویس بسیار بکار می رود. طبق تعریف *IBM*، وب سرویس ها برنامه های کاربردی و جامع هستندکه توسط زبانهای استاندارد باز ازطریق وب قابل دسترسی هستندکه مجموعه ای از وظیفه مندیها جهت کسب وکار یا استفاده شخصی را فراهم می آورند . تأکید این تعریف بر دو نکته می باشد :

نخست اینکه یک وب سرویس بصورت یک برنامه کاربردی قابل دسترس توسط سایر برنامه های کاربردی تحت وب دیده می شود .

دوم اینکه وب سرویس ها "باز" هستند بدان معنا که وب سرویسها واسط هایی را منتشرمی کنند که بتوانند توسط استانداردهای *Message Passing*فراخوانی گردند .

این تعریف بسیار ساده است اما کافی نیست . برای نمونه منظوراز برنامه های کاربردی پیمانه ای و جامع را مشخص نمی کند. *W3C* تعریف مناسب تری از وب سرویس ارائه داده است :

وب سرویس ، سیستم نرم افزاری شناسایی شده توسط یک *URL* است که واسط های عمومی و محدودیت های آن توسط *XML* تعریف و توصیف شده است. تعریف آن می تواند توسط سایر سیستم های نرم افزاری کشف گردد . سپس امکان دارد این سیستمها با وب سرویس در یک روش تعیین شده توسط تعریفش،با استفا ده از پیامهای مبتنی بر *XML* که توسط پروتکل های اینترنتی رسانده شده اند، تعامل برقرار نمایند.*[W3C 2004]*

تعریف *W3C* تأکید می کند که وب سرویس ها باید توانایی تعریف ، توصیف و اکتشاف و در نتیجه مشخص کردن چگونگی دسترسی به وب سرویس ها را داشته باشند *[Alonso 2004]*. همچنین تأکید می کنیم که وب سرویس ها نمی توانند صرفاً اطلاعات ایستا را فراهم کنند بلکه باید بر فعالیتها و تغییرات در جهان تأثیرگذار باشند ؛ به عنوان مثال فروش یک محصول ، کنترل یک دستگاه فیزیکی و امثال آن .

آنچه که باعث جذابیت وب سرویس ها می شود ، امکان یکپارچه سازی وب سرویس های تولید شده توسط سازمان های مختلف برای انجام درخواست های کاربر است . این یکپارچه سازی بر اساس استانداردهای مشترک واسط های وب سرویسها ، صرفنظر از زبان هایی که برای پیاده سازی وب سرویسها وجود دارد و پلاتفرم هایی که وب سرویس ها تحت آن اجرا می شوند ، می باشد .

بطور کلی برای آنکه یکپارچه سازی وب سرویس ها در محیط های ناهمگن ، بهتر ایجاد شود ، باید دارای

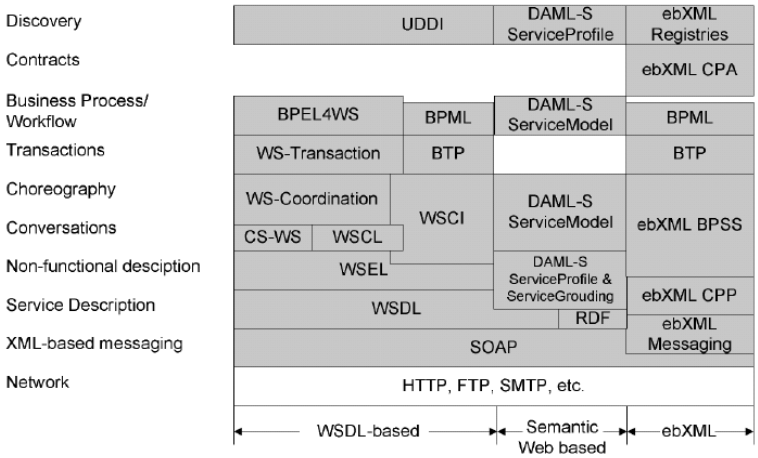
ویژگیهای زیر باشند :

- پیوند سست[4]: پیوند در تولید نرم افزار معمولاً بستگی به درجه وابستگی مؤلفه های نرم افزاری / پیمانه ها دارد . وب سرویس ها در مقایسه با مؤلفه های با پیوند محکم مانند *DCOM [MS COM]* یا *CORBA [OMG CORBA]* ، مستقل هستند یعنی می توانند بصورت مستقل از یکدیگر عمل کنند . این ویژگی وب سرویس ها امکان مکان یابی و ارتباط آنها بایکدیگر را بصورت پویا و در زمان اجرا فراهم می آورد .
- قابلیت دسترسی عمو می : وب سرویس ها می توانند از طریق وب که قابلیت دسترسی آسانی را فراهم می آورد ، تعریف ، توصیف و کشف گردند . علاوه بر امکان مکان یابی سرویس ها توسط کاربران ، سرویس ها نیز می توانند خودشان را توصیف و منتشر نمایند بطوریکه بتوانند با یکدیگر تعامل برقرار نمایند .

[4] Loosely Coupled

- زبانهای استاندارد : توصیف وب سرویس ها توسط زبانهای استاندارد *XML* که به عنوان بخشی از تکنولوژی وب محسوب می شوند ، انجام می شود . اگرچه هسته وب سرویس ها ممکن است توسط زبان های برنامه نویسی مختلفی پیاده سازی شده باشد ، واسط وب سرویس ها توسط زبان های استاندارد *XML* توصیف شده است.

استاندارد *XML* اهمیت بسیاری در موفقیت وب سرویس ها دارد .یکپارچه سازی وب سرویسها در چارچوب ها و زبانهای استاندارد ، جهت کمک به کاربران جهت نمایش یکسان سرویس ها می باشد. شکل ۵ – ۱ که از *[Turner 2003]* اقتباس شده است ، نگاهی به استانداردهای مشترک بکاررفته و موقعیت آنها در کاربرد وب سرویس دارد.



شکل ۵ – ۱ . نگاهی به استانداردهای مشترک بکاررفته و موقعیت آنها در کاربرد وب سرویس

در زیر در مورد زبانها از سه منبع سخن خواهیم گفت:

نخست ، زبانهای وب سرویس که تاکنون پیشنهاد شده است را به عنوان بهترین شبه استاندارد توصیف شده برای وب سرویس بررسی می کنیم . زبانهای وب سرویس *IBM* دیدگاه رایجی از زبانهای وب سرویس را ارائه می دهند.

دوم ، یک توصیف وب معنایی از سرویس ها را معرفی می کنیم که برخی اصلاحات دیگر را جهت تبدیل زبانهای رایج بسته تر به زبانهای بازتر و قابل اندرکنش فراهم می آورد . استدلال می شود که توصیف کنونی *DAML-S* یعنی *DAML + OIL* مبتنی برهستان شناسی وب سرویس معنایی ، نوعی از چنین زبانهایی است.

سوم ، زبانهای تعامل کسب و کاری است که توسط *ebXML* بیان می شوند.

درنهایت برخی از پلاتفرم های وب سرویس که از زبانهای فوق پشتیبانی می نمایند را معرفی خواهیم کرد.

۵-۳. زبانهای وب سرویس IBM

از دیدگاه *IBM* نظریه عمومی کاربرد وب سرویس بصورت زیر است :

یک فراهم کننده وب سرویس ، سرویس ها را برروی وب ارائه می کند . او ممکن است سرویس خود را در یک رجیستری *OnLine* یک دلال سرویس[5] ثبت نماید . این رجیستری نیز امکانات توصیف استا ندارد (مانند طبقه بندی هایی که امکان توصیف وظیفه مندی یک سرویس ، اطلاعات فراهم کننده سرویس و روشی برای دسترسی و تعامل با سرویس را می دهند) را فراهم می آورد . اطلاعات مربوط به یک سرویس خاص توسط فراهم کننده در یک دلال ثبت می گردد.

[5] Service Broker

درخواست کننده سرویس ، جستجو برای سرویس را در رجیستری انجام می دهد . در ابتدا این جستجو با کاوش و پرس و جو در رجیستری انجام می گردد . سپس از توصیف سرویس برای ایجاد محدودیت با توجه به کاربرد آن استفاده می شود . در نهایت با استفاده از زبان ارتباط استاندارد فراخوانی یا تعامل با وب سرویس انجام می گیرد . زبانهای وب سرویس استاندارد که از رویه فوق پشتیبانی می کنند ،عبارتند از : *SOAP, WSDL, UDDI* .

۵-۴. توصیف، کشف و مجتمع سازی یکنواخت منابع [UDDI 2002]

سومین استاندارد اصلی وب سرویس ها ، *UDDI*، به شرکتها و برنامه نویسان اجازه می دهد تا وب سرویس های خود را بر روی اینترنت معرفی کنند . این استاندارد در اصل بوسیله مایکروسافت ، *IBM*و *Ariba* و ۵۰ شرکت بزرگ دیگر ساخته شده است . با استفاده از *UDDI* شرکتها می توانند اطلاعات خود را در اختیار شرکت های دیگر قرار بدهند و مدل *B2B* ایجاد کنند . همان طور که از نام آن مشخص است شرکت ها می توانند وب سرویس خود را معرفی کنند ، با وب سرویس دیگران آشنا شوند و از آن در سیستم های خود استفاده کنند . این استاندارد جدیدی است و در سال ۲۰۰۰ ساخته شده ، کنسرسیومی از شرکتهای صنعتی در حال کار بر روی آن هستند ؛ نسخه دوم *UDDI* در ماه ژوئن سال ۲۰۰۱ ارائه شد و نسخه سوم آن در دست ساخت است. *UDDI* یک متن مبتنی بر *XML*را تعریف می کند که در آن شرکت ها توضیحاتی درباره چگونگی کار وب سرویس شرکتشان و امکانات خود می دهند . برای تعریف این اطلاعات از شکل خاصی که در *UDDI* توضیح داده شده استفاده می شود . شرکت ها می توانند این اطلاعات را در *UDDI*شرکت خود نگهداری کنند و تنها به شرکت های مورد نظرشان اجازه دستیابی به آنها را بدهند یا آنها را در مکان عمومی و در اینترنت قرار دهند . بزرگترین و مهمترین پایگاه *UDDI* ، *UDDI Business Registry* یا *UBR* نام دارد و توسط کمیته *UDDI* طراحی و اجرا شده است . اطلاعات این پایگاه در چهار نقطه نگهداری می شود ، مایکروسافت ، *IBM*، *SAP* و*HP*. اطلاعاتی که در یکی از چهار پایگاه تغییر کند در سه تای دیگر نیز اعمال می شود .

در حقیقت *UDDI* یک رجیستری است بطوریکه فراهم کنندگان سرویس بتوانند سرویس هایشان را ثبت و منتشر نمایند ، فراهم می آورد. این رجیستری حاوی سه بخش است :

صفحات سفید[6] : اطلاعات تماس و اطلاعات قابل خواندن توسط انسان در این صفحات ثبت می گردند.

صفحات زرد[7] : کلمات کلیدی که توصیف سرویس را انجام می دهند ، در این صفحات ثبت می گردند.

صفحات سبز[8] (تکنیکی) : قواعد و توصیفات سرویس برای فراخوانی ها ی کاربردی در این صفحات ثبت می گردند. اطلاعات تجاری و سرویس های شرکت ها کاملاً طبقه بندی شده است و اجازه می دهد که به راحتی در آنها جستجو کرد . سپس متخصصان *IT*می توانند از این اطلاعات استفاده کرده و شرکت ها را برای خدمات بهتر به هم متصل کنند . با این شرح *UDDI* امکان پیاده سازی مدل *B2B* را ایجاد می کند و شرکتها می توانند از سرویس های یکدیگر استفاده کنند .

شرکت هایی که به *UDDI* علاقه نشان داده اند قدرت مند هستند و خیلی از آنها از وب سرویس و استاندارد های آن در محصولات خود استفاده می کنند. *NTT Communications of Tokyo* یکی از شرکت هایی است که در حال اضافه کردن توضیحاتی به ساختار *UDDI*است . در هر حال شرکت ها هنوز کمی درباره وارد کردن خود در پایگاه های عمومی محتاط هستند . این چیز عجیبی نیست . شرکتها ابتدا این امکانات را فقط برای شرکای خود ایجاد می کنند . شرکتهای بزرگ نیز برای مدیریت بر سرویس های خود و اشتراک آنها بین قسمت های مختلف از این استاندارد استفاده می کنند . وقتی این استاندارد به حد بلوغ خود برسد و کاربران با آن

[6] White Paper
[7] Yellow Paper
[8] Green Paper

احساس راحتی بکنند استفاده از آن نیز در مکان های عمومی فراگیر خواهد بود. این تغییر رویه برای شرکت های بزرگی که *B2B* را به روش های قدیمی اجرا کرده بودند مشکل است . بعضی نیز اشکال امنیتی بر این روش می گیرند و مایل نیستند اطلاعاتشان را بدهند . اما با گذشت زمان و کامل شدن این تکنولوژی و درک لزوم استفاده از آن شرکت ها چاره ای جز استفاده از آن ندارند .

UDDI از توصیفات معنایی سرویس ها پشتیبانی نمی کند و فاقد زبان محتوایی برای انتشار آنچه فراهم شده است می باشد. *WSDL* یک گزینه مناسب برای چنین زبان محتوایی می باشد .

۵-۵. زبان تعریف سرویسهای وب [WSDL 1.2]

استاندارد دیگری که نقش اساسی در وب سرویس بازی می کند *WSDL* است . همانطور که قبلاً اشاره کردیم یکی از خواص وب سرویس ها توصیف خود آنهاست به این معنی که وب سرویس دارای اطلاعاتی است که نحوه استفاده از آن را توضیح می دهد . این توضیحات در *WSDL* نوشته می شود ، متنی به *XML* که به برنامه ها می گوید این وب سرویس چه اطلاعاتی لازم دارد و چه اطلاعاتی را بر می گرداند .

وقتی که سازندگان نرم افزار برای اولین بار *SOAP* و دیگر تکنولوژی های وب سرویس را ساختند دریافتند که برنامه ها قبل از اینکه شروع به استفاده از یک وب سرویس بکنند باید اطلاعاتی درباره آن را داشته باشند . اما هر کدام از آن سازندگان برای خودشان روشی برای ایجاد این توضیحات ابداع کردند و باعث شد که وب سرویس ها با هم هماهنگ نباشد . وقتی *IBM* و مایکروسافت تصمیم گرفتند تا استاندارد های خود را یکسان کنند *WSDL* بوجود آمد . در ماه مارس سال ۲۰۰۱ مایکروسافت ، *IBM*و *Ariba* نسخه ۱,۱ را به *W3C* ارائه کردند . گروهی از *W3C* بر روی این استاندارد کار کردند و آن را پذیرفتند . هم اکنون این تکنولوژی در دست ساخت است و هنوز کامل نشده . ولی هم اکنون اکثر سازندگان وب سرویس از آن استفاده می کنند .

هر وب سرویسی که بر روی اینترنت قرار می گیرد دارای یک فایل *WSDL* است که مشخصات ، مکان و نحوه استفاده از وب سرویس را توضیح می دهد . یک فایل *WSDL* نوع پیغام هایی که وب سرویس میفرستد و می گیرد را توضیح می دهد مانند پارامترهایی که برنامه صدا زننده برای کار با وب سرویس باید به آن بفرستد . در تئوری یک برنامه در وب برای یافتن وب سرویس مورد نظر خود از روی توضیحات *WSDL* ها جستجو می کند . در *WSDL* اطلاعات مربوط به چگونگی ارتباط با وب سرویس بر روی *HTTP*یا هر پروتکل دیگر نیز وجود دارد. این مهم است که بدانیم *WSDL* برای برنامه ها طراحی شده است نه برای خواندن آن توسط انسان . شکل فایلهای *WSDL* پیچیده به نظر می آید ولی کامپیوترها می توانند آن را بخوانند و نجزیه و تحلیل بکند . خیلی از نرم افزارهایی که وب سرویس می سازند فایل *WSDL* مورد نیاز وب سرویس را نیز تولید می کنند بنابراین وقتی برنامه نویس وب سرویس خود را ساخت به شکل خودکار *WSDL* مورد نیاز با آن نیز ساخته می شود و احتیاجی به آموزش دستورات *WSDL* برای ساختن و استفاده از وب سرویس نیست .

درحقیقت *WSDL* یک استاندارد پیشنهادی توسط *W3C* برای توصیف سرویس های شبکه می باشد . عمل توصیف بصورت دستورالعملی جهت خودکارسازی جزئیات ارتباط بین برنامه های کاربردی می باشد. *WSDL* ، یک گرامر *XML* که در آن سرویس ها مجموعه ای از پورتهامی باشند را تعریف می نمایند . پیام ها توصیفات انتزاعی داده ردوبدل شده می باشند . انواع پورتها مجموعه های انتزاعی عملیات پشتیبانی شده توسط پورتهای ارتباطی می باشند. معمولا *SOAP* جهت انتشار این عملیات مورد استفاده قرارمی گیرد.

۵-۶. پروتکل ساده دسترسی به شی [SOAP 1.1]

SOAP یکی از عمومی ترین استاندارد هایی است که در وب سرویس ها استفاده می شود . طبق شواهد اولین بار توسط *DeveloperMentor*، شرکت *UserLand* و مایکروسافت در سال ۱۹۹۸ ساخته شده و نسخه اول آن در سال ۱۹۹۹ ارایه شده است .

آخرین نسخه *SOAP* ، نسخه ۱,۲ بود که در دسامبر سال ۲۰۰۱ در *W3C* ارایه شد . نسخه ۱,۲ نشان دهنده کار زیاد بر روی آن و نمایانگر اشتیاق زیاد صنعت *IT* برای استفاده از *SOAP* و وب سرویس است .

هدف اصلی *SOAP* ایجاد روش برای فرستادن داده بین سیستم هایی است که بر روی شبکه پخش شده اند . وقتی یک برنامه شروع به ارتباط با وب سرویس می کند ، پیغام های *SOAP* وسیله ای برای ارتباط و انتقال داده بین آن دو هستند . یک پیغام *SOAP* به وب سرویس فرستاده می شود و یک تابع یا ساب روتین را در آن به اجرا در می آورد به این معنی که این پیغام از وب سرویس تقاضای انجام کاری می کند . وب سرویس نیز از محتوای پیغام *SOAP* استفاده کرده و عملیات خود را آغاز می کند . در انتها نیز نتایج را با یک پیغام *SOAP* دیگر به برنامه اصلی می فرستد .

به عنوان یک پروتکل مبتنی بر *XML* ، *SOAP* تشکیل شده از یک سری الگو های *XML* ی است . این الگو ها شکل پیغام های *XML* را که بر روی شبکه منتقل می شود را مشخص می کند ، مانند نوع داده ها و اطلاعاتی که برای طرف مقابل تفسیر کردن متن را آسان کند . در اصل *SOAP* برای انتقال داده بر روی اینترنت و از طریق پروتکل *HTTP* طراحی شده است ولی از آن در دیگر مدلها مانند *LAN*نیز می توان استفاده کرد . وقتی که وب سرویس ها از *HTTP* استفاده می کنند به راحتی می توانند از *Firewall* عبور کنند .

یک پیغام *SOAP* از سه بخش مهم تشکیل شده است : پوشش یا *Envelope* ، *Header*، بدنه یا *Body* . قسمت پوشش برای بسته بندی کردن کل پیغام به کار می رود . این بخش محتوای پیغام را توصیف و گیرنده آن را مشخص می کند . بخش بعدی پیغام های *SOAP* ، *Header* آن است که یک بخش اختیاری میباشد و مطالبی مانند امنیت و مسیریابی را توضیح می دهد . بدنه پیغام *SOAP* بخشی است که داده های مورد نظر در آن جای می گیرند . داده ها بر مبنای *XML* هستند و از یک مدل خاص که الگوها *(Schemas)* آن را توضیح می دهند تبعیت می کنند . این الگو ها به گیرنده کمک می کنند تا متن را به درستی تفسیر کند .پیغام های *SOAP* نوسط سرور های *SOAP* گرفته و تفسیر می شود تا در نتیجه آن ، وب سرویس ها فعال شوند و کار خود را انجام دهند .

برای اینکه از *SOAP* در وب سرویس استفاده نکنیم از تعداد زیادی پروتکل باید استفاده شود . برای مثال *XML-RPC* تکنولوژی قدیمی تری بود که همین امکانات را ایجاد می کرد . به هر حال ، خیلی از سازندگان بزرگ نرم افزار *SOAP* را بر تکنولوژی های دیگر ترجیح دادند . دلایل زیادی برای انتخاب *SOAP* وجود دارد که خیلی از آنها درباره پروتکل آن است که فراتر از این متن می باشد . سه برتری مهم *SOAP* نسبت به تکنولوژی های دیگر *Extensibility , Simplicity* و *Interoperability* است .

پیغام های *SOAP* معمولاً کدهای زیادی ندارند و برای فرستادن و گرفتن آن به نرم افزار های پیچیده نیاز نیست *SOAP* . این امکان را به برنامه نویس می دهد تا بنا به نیاز خود آن را تغییر دهد . در آخر بدلیل اینکه *SOAP* از *XML* استفاده می کند می تواند بوسیله *HTTP*اطلاعات را انتقال بدهد بدون اینکه زبان برنامه نویسی ، سیستم عامل و سخت افزار برای آن مهم باشد .

SOAP امکان فراخوانی از راه دور متدهای اشیاء را فراهم می آورد. *SOAP* شامل سه بخش است :

Envelop ، قواعد کدگذاری و *RPC* [9]

Envelop ، چارچوبی برای محتوای پیام و قابلیت پاسخ به آن را فراهم می آورد.

قواعد کدگذاری ، یک مکانیسم ترتیبی برای ردوبدل نمونه های انواع داده مشخص را فراهم می آورند.

RPC امکان کپسوله کردن و ارائه از راه دور فراخوانی ها و پاسخ های رویه را فراهم می آورد.

[9] Remote Procedure Call

۵-۷. توصیف وب سرویس معنایی توسط DAML-S

DAML-S [*Ankoekar 2001*] تکنولوژ ی است که ساختارهایی برای توصیف وب سرویسها فراهم می آورد. با استفاده از *DAML-S* ، ابهام کمتری در ویژگیها وقابلیت های سرویس ها بوجود می آید و به فرمت قابل تفسیر توسط کامپیوتر توصیف می گردند . این امر به عنوان تلاشی جهت توصیف ، انتشار و جریان سرویس تلقی می گردد. [*Sollazzo 2002*] توصیفات *DAML-S* امکان تطبیق بهبودیافته سرویس ها را فراهم می آورد. [*Paolucci 2002*]

سطح بالای هستان شناسی[10] بصورت ساختاری است که وابسته به سه مفهوم اساسی زیر است :

Service Profile : آنچه سرویسها انجام می دهند (فراهم می آورند) را توصیف می کنند.

ServiceModel : چگونگی کارکردن وب سرویسها (آنچه رخ میدهد) را توصیف می کنند.

Service Grounding : چگونگی دستیابی به وب سرویس ها (چگونگی استفاده) را توصیف می کنند.

با یک دید عمیق تر ، یک وب سرویس ترکیبی می تواند به عنوان فرآیندی درنظر گرفته شود که توسط زیرکلاسی از *Service Model* بنام هستان شناسی فرآیند ، مشخص می گردد . یک فرآیند می تواند دارای تعدادی از سرویس های شرکت کننده و همچنین جریان کنترل و جریان داده در سراسر سرویس ها باشد. *ServiceGrounding* بصورت نگاشتی بین *ServiceModel* و زبان توصیف سرویس، *WSDL* تعریف میگردد.

۵-۷-۱. ebXML :

ebXML [*ebXML 1.4*] کوشش استانداردی برای تعامل های کسب و کاری است که ابتکاری از *OASIS* و *UN/CEFACT* می باشد و می تواند جانشینی گسترده تر و قابل انعطاف تر برای تبادل داده الکترونیکی[11] تلقی گردد . شرکت ها جهت هدایت کسب و کاری با یکدیگر باید کارهای زیر را انجام دهند :

- کشف محصولات و سرویس های عرضه شده
- تعیین اینکه این محصولات چگونه می توانند با تعیین فرآیند مشترک و تبادل اطلاعات بدست آیند.
- توافق برسر فرم ارتباط و نقاط اتصال برای تبادل مستندات
- توافق برسر عبارات قراردادی همچون تراکنش ها ، طرح ها و امنیت

رجیستری *ebXML* سرور مرکزی است که داده های لازم را ذخیره می کند . هر شرکت باید پروفایل خود *Collaboration Protocol Profile (CPP)* ، که برخی از فرآیندهای کسب و کاری او را مشخص می کند و برخی *Business Service Interface* های پشتیبانی شده را ثبت نماید . فرآیندهای کسب و کاری همان فعالیت های شرکت هستند و واسط های سرویس ، چگونگی انجام تراکنش های لازم در فرآیندهایش را توصیف می کنند . شرکت ها از *CPP* جهت توافق برسر اصطلاحات قراردادی و ایجاد یک توافقنامه همکاری[12] استفاده مینمایند. *ebXML* برای توصیف تعاملات کسب و کاری دارای دو دید می باشد : دید عملیاتی کسب و کار و دید سرویس وظیفه مندی . اولی معناشناسی داده کسب وکار بعلاوه قراردادهای عملیاتی ، توافقنامه ها ، وظایف متقابل و نیازمندیها را مشخص می سازد . دومی ، با سرویس های پشتیبانی همچون قابلیتهای وظیفه مندی ، واسطهای سرویس کسب وکار و پروتکل ها سرو کار دارد.

---

[10] Ontology
[11] EDI - Electronic Data Interchange
[12] CPA - Collaboration Protocol Agreement

ebXML به عنوان مکملی برای سایر تکنولوژی ها تلقی می گردد [*ebXML*] .این قبیل استانداردها در یک معنای عام برای قابلیت اندرکنش معنایی بسیار مناسب هستند اما در این پایان نامه تأکید زیادی بر آن نداریم.

۵-۸. پلاتفرم ها

شرکت های دارای زیر ساخت تجارت الکترونیکی با مشخص ساختن پلاتفرم هایی برای پشتیبانی از برخی سطوح خودکارسازی وب سرویس کار را آغاز می کنند . از جمله چنین محصولاتی می توان موارد زیر را نام برد : *e-speak* شرکت هیولت پکارد که یک پلاتفرم توصیف ، ثبت و اکتشاف پویا برای سرویس های الکترونیکی است ؛ ابزار *BizTalk* و *NET*. شرکت مایکروسافت ؛ *Dynamic Services Framework* شرکت اوراکل ؛ *Application Framework* شرکت *IBM* برای تجارت الکترونیکی و *Open Network Environment* شرکت *SUN*. راه حل های *VerticalNet* که درپی تسریع علامتگذ اری سرویس ها جهت اکتشاف می باشند ، برای سازماندهی و سفارشی کردن اکتشاف وب سرویس ها ، هستان شناسی ها و ابزاری را ایجاد نموده اند و با پلاتفرم خود ، زیرساختی که وب سرویس ها را برای تبادلات تجاری عمومی و خصوصی هماهنگ می سازد ، ارائه می دهند . همچنین پلاتفرم هایی بصورت آکادمیک پیشنهاد شده اند(به [*Shriv 2000*] رجوع نمایید.)

۵-۹. زبان نمادگذاری توسعه یافته

نمیتوان از سرویسهای وب صحبت کرد اما چیزی از *XML* نگفت. با توجه به کثرت مطالب در زمینه *XML* مختصرا درباره آن اشاراتی خواهیم داشت و بقیه را به خواننده واگذار میکنیم.

XML یک تکنولوژی است که به شکل گسترده از آن پشتیبانی می شود ، همچنین این تکنولوژی *Open* است به این معنی که تعلق به شرکت خاصی ندارد . اولین بار در کنسرسیوم *WWW* یا *W3C* در سال ۱۹۹۶ برای ساده کردن انتقال داده ایجاد شده است . با گسترده شدن استفاده از وب در دهه ۹۰ کم کم محدودیت های *HTML* مشخص شد . ضعف *HTML* در توسعه پذیری (قابلیت اضافه و کم کردن خواص) و ضعف آن در توصیف داده هایی که درون خود نگهداری میکند برنامه نویسان را از آن نا امید کرد . همچنین مبهم بودن تعاریف آن باعث شد از توسعه یافتن باز بماند . در پاسخ به این اشکالات *W3C* یک سری امکانات را در جهت توسعه *HTML* به آن افزود که امکان تغییر ساختار متنهای *HTML* مهم ترین آن است . این امکان را *CSS* یا *Cascade Style Sheet* می نامند .

این توسعه تنها یک راه موقتی بود . باید یک روش استاندارد شده ، توسعه پذیر و داری ساختار قوی ایجاد می شد . در نتیجه *W3C* *XML* را ساخت. *XML* دارای قدرت و توسعه پذیری *SGML* یا *Standard Generalized Markup Language* و سادگی که در ارتباط در وب به آن نیاز دارد است .

استقلال اطلاعات یا جدا بودن محتوا از ظاهر یک مشخصه برای *XML* به حساب می آید . متنهای *XML* فقط یک داده را توصیف می کنند و برنامه ای که *XML* برای آن قابل درک است – بدون توجه به زبان و سیستم عامل – قادر است به اطلاعات درون فایل *XML* هر گونه شکلی که مایل است بدهد . متنهای *XML* حاوی داده هستند بدون شکل خاص بنابراین برنامه ای که از آن می خواهد استفاده کند باید بداند که چگونه می خواهد آن اطلاعات را نمایش دهد . بنابراین نحوه نمایش یک فایل *XML* در یک *PC* با *PDA* و تلفن همراه می تواند متفاوت باشد .

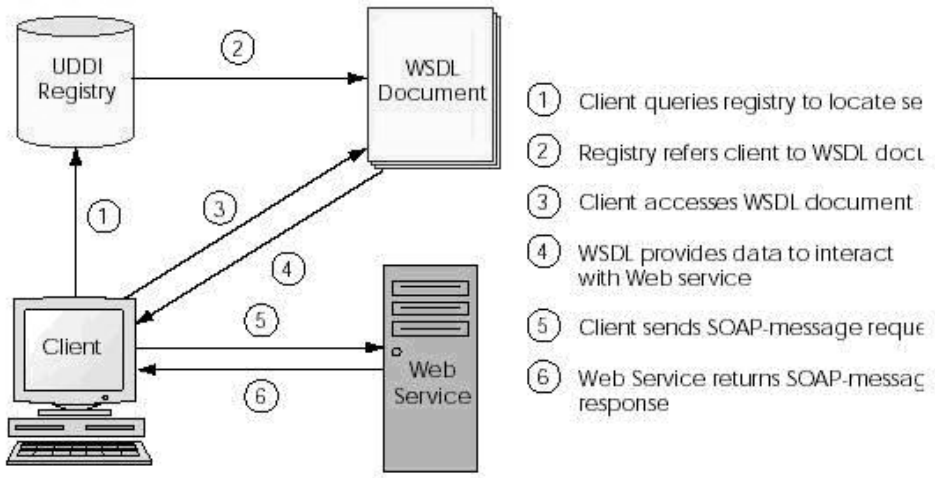
وقتی یک برنامه با متن *XML* مواجه می شود باید مطمئن باشد که آن متن حاوی داده های مورد نظر خود است . این اطمینان توسط برنامه هایی با نام *XML Parser* حاصل می شود . تجزیه کننده ها دستورات متن *XML* را بررسی می کنند . همچنین آنها به برنامه کمک می کنند تا متن های *XML* را تفسیر کند . به صورت اختیاری هر متن *XML* می تواند به متن دیگری اشاره کند که حاوی ساختار فایل *XML* اصلی باشد . به آن متن *XML* دوم *DTD* یا *Document Type Definition* گفته می شود .

وقتی فایل *XML* به *DTD* اشاره می کند برنامه تجزیه کننده فایل اصلی را با *DTD* بررسی می کند که آیا به همان ساختاری که در *DTD*توصیف شده شکل گرفته است یا خیر . اگر یک تجزیه کننده *XML* بتواند یک متن را به درستی پردازش کند متن *XML* نیز به شکل صحیحی فرمت شده است .

وقتی که اکثر نرم افزار ها امکانات وبی خود را افزایش دادند این طور به نظر می آید که *XML* به عنوان یک تکنولوژی جهانی برای فرستادن اطلاعات بین برنامه های انتخاب شود . تمامی برنامه هایی که از *XML* استفاده می کنند قادر خواهند بود که*XML* همدیگر را بفهمند . این سطح بالای تطابق بین برنامه ها باعث می شود که *XML* یک تکنولوژی مناسب برای وب سرویس باشد ، چون بدون اینکه احتیاج به سیستم عامل و سخت افزار یکسان باشد می تواند اطلاعات را جابجا کند.

۵-۱۰. خلاصه

همانطور که در ابتدا توضیح داده شد یکی از دلایل اینکه وب سرویس از دیگر تکنولوژی های موجود مجزا شده است استفاده از *XML*و بعضی استاندارد های تکنیکی دیگر مانند *SOAP* ، *WSDL* و *UDDI* است . این تکنولوژی های زمینه ارتباط بین برنامه ها را ایجاد می کند به شکلی که مستقل از زبان برنامه نویسی ، سیستم عامل و سخت افزار است. *SOAP* یک مکانیزم ارتباطی را بین نرم افزار و وب سرویس ایجاد میکند. *WSDL* یک روش یکتا برای توصیف وب سرویس ایجاد می کند و *UDDI* یک دایرکتوری قابل جستجو برای وب سرویس می سازد . وقتی اینها با هم در یک جا جمع می شود این تکنولوژی ها به برنامه نویس ها اجازه می دهد که برنامه های خود را به عنوان سرویس آماده کنند و بر روی اینترنت قرار دهند . شکل زیر نقش هر کدام از استاندارد ها را در ساختار وب سرویس نمایش می دهد.



شکل ۵-۲ . ساختار یک وب سرویس

Nothing is more difficult, and therefore more precious than to be able to decide.

-Napoleon Bonaparte

پیاده سازی

Implementation

هیچ چیز گرانبهاتر و باارزشتر از قدرت تصمیم گیری نیست.

فصل ششم

لزوم ترکیب داده کاوی با وب سرویس

Data Mining with Web Service Necessity

در این فصل ابتدا به این پرسش پاسخ میدهیم که نیاز به پیاده سازی ابزارهای داده کاوی تحت سرویسهای وب چیست، حال آنکه تحت *PC* وجود دارند...

۶-۱. مقدمه

پس از فراگیری موج اینترنت، پدیده جدیدی در محاسبات، شروع فناوری اطلاعات را سبب گردیده است. این پدیده مبتنی بر مفهوم محاسبات مبتنی بر سرویس است و ادعا شده که سبب میگردد وب برای ما کار کند، بیش از آنکه ما روی وب کار کنیم. این فکر مبنای سرویسهای وب و الگوی غالب سالهای آتی خواهد بود. این شاید مهمترین دلیل ما برای پیاده سازی ابزارهای مفید داده کاوی از طریق سرویسهای وب میباشد. در بخش اول این فصل به این پرسش پاسخ میدهیم که آیا هیچ نیازی برای پیاده سازی ابزارهای داده کاوی با سرویسهای وب وجود دارد یا نه ؟

۶-۲. دلائل پیاده سازی ابزارهای داده کاوی با وب سرویس

پس از فراگیری موج اینترنت، پدیده جدیدی در محاسبات، شروع تحولاتی در جهان فناوری اطلاعات را سبب گردیده است. پدیده سرویسهای وب الگوی غالب سالهای آتی خواهد بود. سرویسهای وب به ارتباط متقابل و پویای تجاری ربط دارند و مرزهای بین مشتریان، شریکان و طرفهای تجاری را محو میسازند. در منظر وسیعتری از این جهان گفتگوی الکترونیکی درباره اکوسیستمهایی است که این سرویسها در آن قرار میگیرند. این اکو سیستمها خود نیز در اینترنت مستقر میشوند.

ممکن است در نگاه اول حیرت‌انگیز به نظر برسد ولی بسیاری از شرکت‌ها، چه کوچک و چه بزرگ، باید به فکر توسعه‌ی یک راهکار منطقی در زمینه بازاریابی الکترونیکی باشند. با توجه به این که از سال ۱۹۹٤ تا کنون صنعت بازاریابی به شکل موثری از اینترنت بهره گرفته است، می‌توان این طور نتیجه گرفت که سازمان‌هایی که در فعالیت بازاریابی خود راهکاری برای استفاده از اینترنت درنظر نگرفته‌اند اشتباه بزرگی را مرتکب می‌شوند. در این بخش ده دلیل برای حضور ابزارهای داده کاوی در وب، آنچه تاکنون هیچ تلاش هدف‌مندی در مورد آن نشده است، ارایه می شود:

۱- مکانی برای جستجوی اطلاعات

احتمالا مهم‌ترین دلیل توجیه کننده‌ی نیاز شرکتها به استفاده از راهکار وب سرویس، تغییراتی است که در نحوه جستجوی اطلاعات توسط مشتریان صورت گرفته است. با آن که هنوز هم برخی مشتریان برای جمع آوری اطلاعات با مراجعه به مراکز خرید، مشورت با نمایندگان فروش، جستجو در مغازه‌ها و صحبت با دوستان نیازهای خود را رفع می‌کنند، شمار فزاینده‌ای از آنها نیز شبکه اینترنتی را به عنوان منبع آگاهی اولیه خود برمی‌گزینند.

از زمانی که سایت‌های جستجو به مقصد نهایی بسیاری از کاربران تبدیل شده است، مشتریان به طور خاص از موتورهای جستجو به عنوان پورتالهای اطلاعاتی مورد نیاز خود استفاده می‌کنند. بخش بازاریابی باید تشخیص دهد که اینترنت محلی است که مشتریان آن‌را برگزیده‌اند و این موضوع برای حضور و بقا سازمانها در دنیای تجارت، باید مورد توجه قرار گیرد.

۲- آنچه مشتری توقع دارد

امروزه نه تنها اینترنت به منبعی منتخب برای جستجوی اطلاعات تبدیل شده است، انتظار می‌رود در چند سال آینده افراد توقع خواهند داشت که مطالب مورد نظر خود درباره کالا را در آن یافته، معاملات خود را از طریق اینترنت انجام دهند. این مطلب به خصوص در مورد مشتریان زیر ۲۵ سال صدق می‌کند. در بیشتر کشورها تقریبا تمام کودکان و نوجوانان با آگاهی از نحوه‌ی استفاده از اینترنت، پرورش می‌یابند. با تسلط این نسل بر معاملات خانگی و تجاری، بدیهی است که آنها خواهان پررنگ‌تر بودن حضور شرکت ها در شبکه اینترنتی خواهند بود.

۳- ضبط اطلاعات گسترده درباره مشتری

شبکه اینترنت (به عنوان یک ابزار جمع آوری اطلاعات) در مقام تهیه اطلاعات درباره‌ی چگونگی فعالیت مشتری در وب ابزاری بی همتا است. هر بازدیدکننده با ورود به یک سایت، اطلاعاتی درباره چگونگی ورود خود به سایت، جهت‌یابی، موضوعاتی که روی آن کلیک کرده، جنس خریداری‌شده و جزییات بی‌شمار دیگری از این دست به جا می‌گذارد. با استفاده از روشی برای شناسایی هویت مشتری (مانند اطلاعات ورود به سایت)، سازمان قادر است فعالیت مشتری را در جریان بازدیدهای مکرر پی‌گیری کند. شناخت رفتار و اولویت‌های مشتری فرصت‌های زیادی را برای پاسخ‌گویی به نیازهای او ایجاد می‌کند و در صورت تشخیص صحیح، مشتری با وفاداری درازمدت خود پاسخ‌گوی این اقدام خواهدبود.

٤- بازاریابی هدف‌مند

بهترین راه هزینه کردن برای بازاریابها، سرمایه‌گزاری مستقیم روی افرادی است که بیش از بقیه احتمال می‌رود جذب تولیدات آنها بشوند. متاسفانه تلاش برای نشانه‌گرفتن مشتریانی با بیش‌ترین احتمال خرید چندان آسان نخواهد بود. به عنوان مثال میزان پولی را در نظر بگیرید که چگونه در تبلیغات تلویزیونی برای افرادی هزینه می‌شود که احتمالا کالای مورد نظر را نخواهند خرید. با این حال قابلیت بی‌همتای شبکه اینترنت در شناسایی و ردیابی رفتار مشتری، توانایی بخش بازاریابی را در هدف‌گیری افرادی با بیشترین پتانسیل برای خرید کالاهای مذکور افزایش داده است. این یکی از اهداف اصلی داده کاوی است.

٥- برانگیختن میل افراد به خرید

اینترنت، خواه خوشایند مشتری باشد و خواه نباشد، به بهترین عرصه برای تحریک افراد به خرید، تبدیل شده است. قسمت اعظم این اتفاق را می‌توان به بهره‌گیری بخش بازاریابی از پیشرفت‌های حاصل در حوزه فناوری مرتبط دانست که ۱) به وب‌سایت ها اجازه می‌دهد پیشنهادات خود را براساس رفتار برخط مشتری تنظیم کنند و ۲) فرایند معاملات برخط را بهبود می‌بخشد.

اما این نوع خریدها هم‌چنین از روش "حالا بخرید، بعد پرداخت کنید" ، رویکرد رایج جامعه‌ای که در آن مصرف کارت‌های اعتباری بیش از اندازه رایج شده است، بهره می‌گیرد. حال چگونگی تاثیر این مطلب در طول زمان و در حالی که بیشتر مشتریان در قروض خود غوطه ور خواهند بود، نیاز به مراقبت داشته، می‌تواند بر فعالیت سازمان موثر باشد.

٦- ارایه تولیدات و سرویسهای سفارشی

شرکت‌ها به خوبی به این نکته واقفند که با طراحی تولیدات و ارایه سرویسها به منظور برآورده کردن نیازهای اشخاص می‌توانند مشتریان دایمی خود را افزایش دهند. این موضوع بسیاری از بخش‌های بازاریابی آنلاین را به انجام راهکارهای سفارشی سازی در سطح عمده سوق داده است. به این‌ترتیب که آنها به مشتری گزینه های آنلاینی را برای طراحی تولیدات و خدمات ارایه می‌کنند. محاسبه تخمینی میتواند در این راستا موثر باشد.

ماهیت تعاملی شبکه اینترنت روش "به میل خودتان بسازید" را به گزینه ای تسهیل کننده در تحقق حق انتخاب در امر خرید تبدیل می کند. مشتریان مختار به انتخاب، وقتی احساس کنند شرکت طرف معامله‌ی آنها دقیقا کالای مورد نظر ایشان را توزیع خواهد کرد، در دراز مدت به مشتریان وفادار شرکت تبدیل خواهند شد.

۷- به دام انداختن فوری مشتریان در زمان حراج و تخفیف

هیچ شیوه ارتباطی با اینترنت در روند تاثیر تبلیغات به واکنش بی‌درنگ مشتری برابری نمی کند. چرا که در این شیوه مشتری بلافاصله بعد از تبلیغات شروع به خرید می‌کند. پیش از اینترنت، ثمربخش‌ترین روش "فراخوان برای خرید" از طریق برنامه های تلویزیونی صورت می‌گرفت.

این برنامه‌ها بینندگان را تشویق می‌کردند با شماره تلفن‌های رایگان تماس حاصل کنند و سفارش دهند. با این حال، تغییر مشتری از حالت منفعل (تماشاگر تلویزیون) به وضعیت فعال (برداشتن گوشی تلفن و برقراری تماس)، به اندازه واداشتن افراد به کلیک روی تبلیغات اینترنتی، ساده نیست.

۸- القا مفهوم "تامین‌کننده" تمام عیار در ذهن مشتری

اینترنت به سادگی توزیع‌کنندگان و خرده فروشان را به تامین کنندگانی همه‌جانبه تبدیل می‌کند. برخلاف تامین‌کنندگان تجارت سنتی که غالبا بر مبنای لیست موجودی در انبار یا خدماتی که در محل ارایه می‌دهند راجع به آنها قضاوت می‌کنیم، سایت‌های تجارت الکترونیکی این تصور را ایجاد می‌کنند که حجم زیادی کالای موجود در انبار و خدمات گسترده و متنوعی دارند. باید توجه داشته باشید که می توان این فعالیت را با قرار دادن اطلاعات مربوط به کالاها و خدمات در وب سایت تکمیل کرد.

در عین حال می‌توانید در پشت پرده این فعالیت را با برخی تامین‌کنندگان بیرونی، بر اساس قراردادهای حمل و نقل و توافق‌های خدماتی، همراه کنید. با چنین تمهیداتی، مشتریان احساس می کنند با تامین کنندگانی روبه رو هستند که خدمات همه جانبه‌ای را ارایه می‌کنند. این در حالی است که درصد خاصی از این تولیدات و خدمات از منابع دیگر تامین شده‌اند. کلاسه بندی اطلاعات میتواند دراین زمینه کمک شایانی به سازمانها کند.

۹- سربار کمتر، هزینه های پایین تر و خدمات بهتر

امروزه فناوری اینترنت جایگزین روشهای پرهزینه توزیع محصولات و ارایه خدمات و همچنین مدیریت نیازهای اطلاعاتی مشتری می شود. در توزیع تولیدات و ارایه خدمات به شیوه دیجیتالی(مانند موسیقی، نشر، گرافیک و غیره) مقادیر زیادی از هزینه ها صرفه جویی می‌شود، زیرا در این روش، هزینه‌های حمل و نقل از معادله هزینه اساسا پاک شده است.

در حوزه‌های دیگر بازاریابی مانند بخش خدمات مشتری نیز می‌توانیم شاهد چنین صرفه‌جویی‌هایی باشیم: شرکت ها با فراهم کردن امکان دسترسی آن‌لاین به اطلاعات مربوط به تولیدات از طریق سیستم های مدیریت دانش و پاسخ به سوالات تکراری ، قادرند از حجم تماس‌های تلفنی مشتری بکاهند.

فروشندگان نیز می‌توانند با تشویق مشتریان بالقوه به کسب اطلاعات برخط درباره‌ی کالای مورد نظر، قبل از جلسات رو در رو، آمادگی مشتری را برای خرید افزایش دهند. با این روش زمانی که پیش از این صرف توضیح اطلاعات ابتدایی شرکت و اطلاعات مربوط به کالاها می شد کاهش یافته، زمان بیشتری به درک و ارایه راه حل برای مشکلات مشتری اختصاص می‌یابد. با توجه به این مثال‌ها، اینترنت می‌تواند علاوه بر کاهش هزینه‌های عملیاتی و اجرایی، ارزش بیشتری به مشتری ها بدهد.

۱۰- جهانی کردن دامنه حضور

شبکه اینترنت کانالی ارتباطی و توزیعی است که قابلیت در دسترس بودن جهانی را برای تولیدات و خدمات یک شرکت فراهم می‌کند. یک بازاریاب محلی می‌تواند با داشتن یک وب‌سایت به بازاریابی جهانی تبدیل شود و به این ترتیب بازار هدف بالقوه خود را در مقایسه با میزان کنونی تا چند برابر افزایش دهد.

برخلاف زمانی که تجارت الکترونیکی هنوز باب نشده بود و بازاریابی بین المللی کاری وقت‌گیر و پرهزینه به شمار می‌رفت، اکنون بارگذاری کردن فایل‌ها در اینترنت برای ایجاد یک وب سایت تنها کاری است که باید برای حضور در عرصه جهانی انجام دهید. درحالی که تاسیس یک وب‌سایت به تنهایی فروش بین المللی را تضمین نمی کند (برای این که یک سایت از نظر بین المللی کارآمد باشد، به فعالیت بازاریابی زیادی نیازمند هست)، ولی اینترنت در مقایسه با روزهای قبل از همه گیرشدن آن، جهشی عظیم را به سمت تجارت جهانی فراهم می کند.

٦-٣. نکات تکنیکی

از بعد تکنیکی نیز، وب سرویس دارای خواصی است که آن را از دیگر تکنولوژی ها و مدل های کامپیوتری جدا می کند *Paul Flessner*، نایب رییس مایکروسافت در *dot NET Enterprise Server* چندین مشخصه برای وب سرویس در یکی از نوشته هایش ذکر کرده است.

١. وب سرویس ها قابل برنامه ریزی هستند. یک وب سرویس کاری که می کند را در خود مخفی نگه میدارد. وقتی برنامه ای به آن اطلاعات داد وب سرویس آن را پردازش می کند و در جواب آن اطلاعاتی را به برنامه اصلی بر می گرداند.

٢. وب سرویس ها بر پایه *XML* بنا نهاده شده اند *XML* .و *XML* های مبتنی بر *SOAP* تکنولوژی هایی هستند که به وب سرویس ها این امکان را می دهد که با دیگر برنامه ها ارتباط داشته باشد حتی اگر آن برنامه ها در زبانهای مختلف نوشته شده و بر روی سیستم عامل های مختلفی در حال اجرا باشند.

٣. همچین وب سرویس ها خود-توصیف هستند. به این معنی که کاری را که انجام می دهند و نحوه استفاده از خودشان را توضیح می دهند. این توضیحات به طور کلی در *WSDL* نوشته می شود *WSDL* . یک استاندارد بر مبنای *XML* است.

٤. وب سرویس ها قابل شناسایی هستند به این معنی که برنامه نویس می تواند به دنبال وب سرویس مورد علاقه در دایرکتوری هایی مثل *UDDI* جستجو کند. *UDDI*یکی دیگر از استاندارد های وب سرویس است.

٥. وب سرویس مستقل از زبان برنامه نویسی، سیستم عامل و سخت افزار است.

شرکتهای معتبر در این سالها اقدا م به ساخت ابزارهای حرفه ای داده کاوی نموده اند که از جمله میتوان از *Microsoft OLAP Browser(SQL Server), Oracle OLAP, DTREG, 3DMiner, ...* و همچنین یک ابزار رایگان تحقیقاتی مبتنی بر جاوای مشهور به نام *Weka* یاد کرد. در همه این موارد آنچه مشهود است گران بودن استفاده، پیچیده بودن واسط مربوطه و شیوه کار با آنهاست که داده کاوی را به امری تخصصی تبدیل میکند به نحوی که کمتر کسی از مزایا و امکانات این ابزار بالقوه باخبر است. لذا میتوان با پیاده سازی مناسب واسط از طریق وب سرویسها آنرا به یک ابزار قابل استفاده عام تبدیل نمود.

٦-٤. مشکلات و نیازها

اما این ترکیب به این سادگی نیز امکان پذیر نیست. حوزه بسیار وسیع مسئله از برنامه نویسی پایگاه داده گرفته تا برنامه نویسی تحت اینترنت و همچنین تجربه در پیاده سازی الگوریتمهای پیچیده و تحلیلی داده کاوی، پیاده سازی این مسئله را مشکلتر از آنچه به نظر میرسد، کرده است. ترکیب و هماهنگی دو مقوله جدا از هم با استانداردها و تکنیکهای کاملا متفاوت برنامه نویسی مسئله را کاملا جذاب و در خور کار اساسی مینماید.

مشکل دیگر بر سر راه پیاده سازی نو بودن ایده است و اینکه تا کنون کارهایی که در این زمینه انجام شده بسیار ناچیز است و استانداردهای مناسب در این زمینه طراحی نشده است. همچنین این ترکیب مورد آزمایش قرار نگرفته تا معایب آن شناخته شود و چون مورد پیاده سازی وسیعی قرار نگرفته مسایل و مشکلات آن مشخص نگردیده است.

در نهایت، آنچه در برنامه نویسی برای نیل به این هدف مورد نیاز است و کسی که بخواهد در این زمینه کار کند به آن نیاز دارد، از قرار زیر است :

١. آشنایی با پایگاههای داده معتبر مانند *Access, Oracle, SQL-Server* و ...

٢. آشنایی با برنامه نویسی پایگاه داده *SQL* و *PL/SQL* .

۳. آشنایی با *Data Warehousing*

۴. آشنایی با *OLAP* و ابزارهای آن

۵. آشنایی با الگوریتمهای داده کاوی

۶. آشنایی با یکی از تکنیکهای برنامه نویسی پویای مشتری مانند اکتیوایکس *ASP, PHP* , و ...

۷. آشنایی با یکی از بسترهای پیاده سازی وب سرویس مانند *CGI* و ...

۸. آشنایی با *SOAP* به عنوان مهمترین پروتکل ارتباطی وب سرویس

۶-۵. خلاصه

با توجه به آنچه گفته شد، لزوم پیاده سازی ابزارهای داده کاوی با سرویسهای وب بر هیچ کسی پوشیده نیست. سرویسهای وب به عنوان تکنولوژی اصلی آینده فناوری اطلاعات و تجارت الکترونیک هر سازمانی را ناچار به کسب و کار با خود خواهد کرد.

از سوی دیگر، آنچه امروزه در علم کامپیوتر شاهد آن هستیم، گواه بر این مدعاست که داده کاوی یک ابزار بسیار موثر، مهم و مفید در تحلیل و آنالیز داده هاست بطوریکه بینش و نگرشی فراتر از تصور و قدرت انسان را در اختیار تحلیلگر قرار میدهد، تا آنجا که امروزه اساس و بنیان ایده های نو وتصمیم گیریهای کلان آینده سازمانها و تجارتهای عظیم را بر آن استوار نموده اند. اما آنچه متاسفانه در این مقوله دیده میشود حاکی از آنست که پیاده سازیهای ابزار داده کاوی، راه را بر استفاده تجارتهای کوچک از این ابزار خلاقه بسته است. بنا بر باور بزرگان اقتصاد، تجارتهای کوچک و سودآور الکترونیکی عرصه را بر تجارتهای کلان قدیمی تنگ خواهند کرد. لذا جای خالی ابزارهای داده کاوی در دنیای مجازی اینترنت بشدت احساس میشود.

فصل هفتم
پیاده سازی
مشکلات و راه حلها

Implementation Problems And Resolves

در این فصل پیاده سازی را به دو بخش اساسی سمت مشتری و سمت کارگزار تقسیم میکنیم و مسایل هرکدام را بیان و به ارائه را ه حل میپردازیم. در سمت مشتری تکنولوژی اکتیوایکس را برگزیده و به تشریح آن و ذکر دلایل میپردازیم. همچنین در سمت کارگزار *CGI* را تشریح خواهیم کرد. در انتها نیز مشکلات ترکیب و ارتباط و ارسال داده ها را با *SOAP* حل خواهیم کرد و ...

۷-۱. مقدمه

اینترنت شبکه ای از شبکه های مرتبط به یکدیگر بوده که بر روی آن سرویس های متعددی بمنظور ارائه خدمات فعال می باشند. بدون شک سرویس وب یکی از مهمترین سرویس های موجود بر روی اینترنت است که بیشترین تاثیر را در عمومیت یافتن اینترنت در سطح

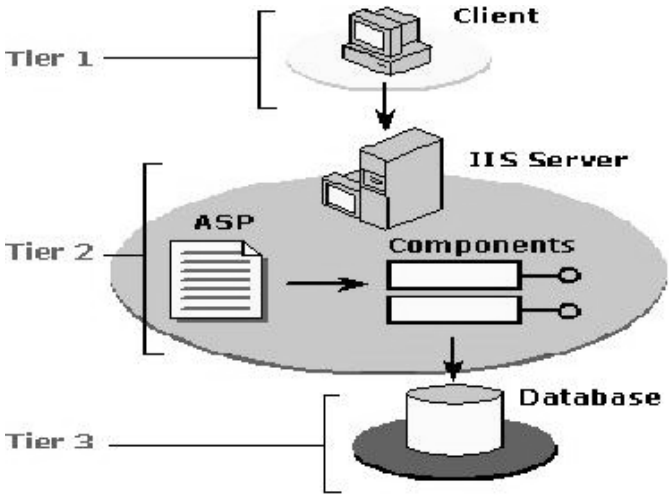
جوامع بشری برعهده داشته است. وب در نگاه اول مانند یک سیستم اطلاعاتی توزیع شده است . در سیستم فوق اطلاعات (با اشکال متفاوت) بر اساس استانداردهای مربوطه و با یک ساختار مشخص قرار گرفته و با استفاده از ارتباطات تعریف شده بین اطلاعات می توان بسرعت و بسادگی از اطلاعات موجود استفاده کرد. ارائه دهندگان اطلاعات با پیروی از اصول موجود اطلاعات خود را در سیستم فوق قرار داده و جستجو کنندگان اطلاعات با استفاده از ابزارهائی خاص قادر به جستجو و دستیابی به این اطلاعات خواهند شد.

برای تشریح بهتر برنامه های کاربردی تحت وب[1] مدرن و امروزی، معمولا آنها را به ٤ لایه مستقل تقسیم می کنند. لایه مشتری یا *Client*، لایه های میانی که عبارتند از *presentation* و *Business* و در نهایت آخرین لایه که عبارت است از لایه داده یا *Database*. معمولا لایه های میانی در یک لایه مجتمع شده و در نتیجه یک مدل ساده شده ۳ لایه ای به دست می دهند. (شکل ۷-۱)

لایه مشتری بخشی از این معماری است که توسط کاربر وب مشاهده می شود. بقیه لایه ها بر روی کامپیوترهایی قرار دارند که ممکن است صدها کیلومتر از مشتری فاصله داشته باشند. لایه مشتری در این مدل همان مرورگر وب است که بر روی کامپیوتر کاربر صفحات وب را نمایش می دهد. در این لایه علاوه بر امکان نمایش اطلاعات، امکان ورود و برخی پردازشهای اطلاعاتی نیز وجود دارد.

معمولا دو گونه پیاده سازی از لایه مشتری وجود دارد. در روش اول از هوشمند بودن کامپیوتر کاربر و قابلیتهای پردازشی آن استفاده ای نمی شود. هنگامی که کاربر اطلاعات یک فرم را تکمیل می کند و یا درخواست صفحه خاصی را دارد، کلیه عملیاتهای پردازشی در لایه های میانی انجام می گیردو صفحه های جدیدی که حاوی نتایج این پردازشها هستند برای او ارسال می گردند. این روش را اصطلاحا مشتری غیر هوشمند[2] می نامند.

اما در روش دوم از هوشمندی کامپیوتر کاربر به خوبی استفاده می شود. در این روش برخی پردازشهای خاص توسط کامپیوتر مشتری و مرورگر موجود بر روی آن صورت می گیرد. در این حالت کامپیوتر مشتری را مشتری نیمه هوشمند[3] می نامند. برای انجام این پردازشها معمولا از زبانهای اسکریپت نویسی نظیر *vbscript* و یا *javascript* استفاده می شود. توانایی مرورگها در اینجا اهمیت می یابد.



شکل ۷-۱. معماری سه لایه برنامه های کاربردی تحت وب

لایه *presentation* وظیفه ارائه محتویات دینامیک صفحات وب را بر عهده دارد. روشهای پیاده سازی این عملیات بر روی کامپیوترها متنوع است. تکنولوژیهای مختلف نظیر *CGI, PHP, ASP, JSP, Servlet, ...* لایه *presentation* معمولا درون برنامه های وب سرور نظیر *Apache webServer* و یا *Microsoft IIS* و ... پیاده سازی می شود. وب سرورها توانایی دریافت و اجرای چندین درخواست از

[1] Web based applications
[2] Dump HTML Client
[3] Semi Intelligent Client

سوی برنامه های کاربردی لایه های دیگر و همچنین کاربران را به طور همزمان دارند. این نرم افزارها بر اساس یک پیکربندی اولیه می دانند که درخواستهای کاربر را باید به کدام برنامه کاربردی ارجاع دهند.

لایه *Business* بیشترین حجم عملیات یک سایت مدرن امروزی را انجام می دهد. عملیاتهای نظیر مدیریت نشستها، مدیریت دستیابی به بانکهای اطلاعاتی و... از جمله این عملیاتها هستند. روشهای گوناگونی برای پیاده سازی این لایه وجود دارد نظیر استفاده از اشیای *COM*متعلق به شرکت میکروسافت و یا استفاده از متناظرهای آن متعلق به شرکت *SUN*. راه حلهای قدیمی نیز برای پیاده سازی این لایه وجود دارد نظیر استفاده از *CORBA Objects*. برنامه های لایه *Business* معمولا درون برنامه هایی موسوم به *Application Server* پیاده سازی می شوند نظیر *Microsoft MTS, Oracle Application Server*.

لایه داده مسئول نگهداری اطلاعات است و می تواند شامل مجموعه ای از اطلاعات به صورت بانکهای اطلاعاتی مدرن یا فایلهای ساده و ... باشد.

۷-۲. پیاده سازی سمت مشتری سرویس

با توجه به اینکه کاربر استفاده کننده از سرویس باید صورت مسئله داده کاوی را بیان و از سرویس انتظار پاسخ داشته باشد. این نکته را باید در نظر داشته باشیم که باید حداقل اطلاعات ورودی از کاربر در داده کاوی دریافت شود. صرف نظر از مقدار و نوع ورودیها، باید از یک تکنولوژی پویا سازی و یا برنامه نویسی وب استفاده نماییم. برای این منظور میتوان از *VBScript, Jscript, PHP, ASP,* اکتیوایکس و... استفاده نمود.

یکی از مشکلات اصلی در این پیاده سازی توجه به این مسئله است که بر خلاف بیشتر برنامه های وب، پایگاه داده مورد استفاده در سرور قرار ندارد، بلکه پایگاه روی سیستم مشتری قرار دارد و باید از انجا فراخوانی شود. مشکل ارسال قابل فهم داده ها نیز در همین جا به چشم میخورد. به این معضل در جای خود کامل رسیدگی خواهیم کرد اما باید آنرا در انتخاب تکنولوژی مشتری نیز مد نظر قرار دهیم.

مواردی که باید در پیاده سازی سمت مشتری در نظر گرفته و حل شود به قرار زیر است :

۱. تکنولوژی مورد استفاده باید قادر به اتصال به پایگاه داده در سیستم مشتری داشته باشد.
۲. تکنولوژی مورد استفاده باید قادر به خواندن و بازیابی جداول و اطلاعات پایگاه داده باشد.
۳. تکنولوژی مورد استفاده باید قادر به اتصال به هر نوع پایگاه داده ای باشد. چون مشتریان مختلف از پایگاههای متفاوت استفاده میکنند.
۴. تکنولوژی مورد استفاده باید قادر به طراحی یک واسط ساده و مبتنی بر کاربر باشد.
۵. تکنولوژی مورد استفاده باید قادر به نمایش و ارایه پاسخ داده کاوی بصورت تصویری و قابل فهم به مشتری مانند گراف و جدول باشد.
۶. تکنولوژی مورد استفاده باید قادر به برقراری امنیت لازم برای جلب مشتریان باشد.
۷. تکنولوژی مورد استفاده باید قادر به پشتیبانی از پروتکلهای ارتباطی سرویسهای وب برای برقرای ارتباط موثر و ارسال درست و معنایی داده ها باشد.

ازمیان تکنولوژیهای موجود ما برای پیاده سازی از اکتیوایکس استفاده نموده ایم که در ادامه به شرح آن و ذکر مزایا، معایب و دلائل این انتخاب خواهیم پرداخت.

۷-۳. آشنایی با اکتیوایکس

تکنولوژی اکتیوایکس یکی از قویترین ابزارهاییست که برنامه نویسان ویندوز در اختیار دارند. تقریبا هر برنامه ای که بنویسند از اکتیوایکساستفاده خواهند کرد . در این فصل با سر فصل ذیل آشنا خواهیم شد :

- اهمیت مدول های نرم افزاری
- تاریخچه و اصطلاحات اکتیوایکس
- درک مفهوم جاوا و کنترل اکتیوایکس
- مطالبی درباره امنیت و توزیع کنترل های اکتیوایکس

۷-۳-۱. انقلاب مدول های نرم افزاری

همانند هر صنعت دیگری ، کارآیی یکی از دغدغه های اصلی برنامه نویسان است و یکی از جنبه های مهم کارآیی اجتناب از دوباره کاری است . اگر کدی نوشته اید که کار خاصی را انجام می دهد ، چرا باید دفعه بعد همان کد را دوباره بنویسید ؟

اولین گام در راه استفاده مجدد از کدها ، مفهوم زیر مجموعه یا روال *Procedure* است . کدی را یکبار می نویسید و از آن به بعد در هر جای برنامه که لازم بود فقط کافی است آنرا احضار *Call* کنید . قرار دادن روال های کلی در یک مدول برنامه نویسی قدمی به پیش بود . با این تمهید می توان از آن روال ها در برنامه های مختلف استفاده کرد.

۷-۳-۲. برنامه نویسی شی گرا

به موازات رشد برنامه نویسی مدولار ، تکنولوژی دیگری در زمینه برنامه نویسی اختراع شد و توسعه یافت : برنامه نویسی شی گرا .[4]

محرک توسعه این تکنولوژی رشد و پیچیدگی روز افزون برنامه ها و مشکلاتی بود که به تیغ آن برنامه نویسان را درگیر خود کرده بود . مهمترین منبع این مشکلات بر هم کنش غیر قابل پیش بینی قسمتهای مختلف یک برنامه با یکدیگر بود . چون این قسمتها مانند دانه های یک زنجیر در هم بافته شده بوند و هر تغییری در یک قسمت به راحتی سایر قسمتها را متاثر می کرد.

را ه حل این مشکل آن بود که هر قسمت برنامه در یک بسته بنام شی[5] کپسوله[6] شود . ساز و کار درونی هر شی مطلقا از دید دنیای خارج مخفی است و آنها نمی توانند تاثیری بر عملکرد وی بگذارند . البته یک شی نمی تواند بکلی از دنیای اطراف خود ایزوله شود چون بدین ترتیب دیگر چیز بی فایده ای بیش نخواهد بود ! به همین دلیل برای ارتباط با دیگر قسمتهای برنامه ، هر شی از وسیله ای بنام واسط[7] استفاده می کند . واسط هر شی دو بخش دارد : خواص (داده ها) و متدها (کدها)ی آن.

۷-۳-۳. تاریخچه کوتاهی از اکتیوایکس

برنامه نویسی مدولار سالها برنامه نویسان را به خود مشغول کرده بود و در واقع یکی از محرکهای اصلی توسعه سیستم عامل ویندوز هم همین ایده کدهای قابل اشتراک و قابل استفاده مجدد بوده است . اولین گام در راه پیاده سازی عناصر مدولار تکنولوژی *OLE* [8] بود . هدف اولیه *OLE* ایجاد سندهای مرکب با استفاده از برنامه های مختلف بود.

---

[4] Object Oriented Programming
[5] Object
[6] Encapsulation
[7] Interface
8 Object linking and embedding

سندی که مقداری متن و مقداری نمودار دارد ، و هر کدام آنها با نرم افزار خاص خود ایجاد شده اند ، نمونه ای از یک سند مرکب است . وقتی با متن کار می کنید نرم افزارها واژه پرداز کنترل را بدست می گیرد و وقتی با نمودارها کار می کنید نرم افزار ترسیمی مسئولیت را به عهده می گیرد. *OLE* با وجود کندی و مشکلات دیگر مسلماً قدمی به پیش بود.

تکنولوژی *OLE* خود بر یک استاندارد کلی تر بنام *COM* [9] استوار است. بزودی *COM* از سندهای مرکب فراتر رفت و *OLE* را هم بدنبال خود کشاند و از آن به بعد *OLE* اصطلاحی شد برای هر چیزی که از تکنولوژی *COM* استفاده می کرد . سالها بعد که میکروسافت بطور جدی درگیر اینترنت شد ، اصطلاح اکتیوایکس هم وارد ادبیات کامپیوتری شد . ابتدا این اصطلاح فقط در رابطه با اینترنت و وب بود اما اوضاع بدین منوال باقی نماند و اکنون اکتیوایکس به آن بخش از تکنولوژیهای *COM* گفته می شود که در آنها یک قطعه نرم افزاری امکانات خود را در اختیار برنامه های دیگر می گذارد . یکی از ادعاهای اکتیوایکس (که بویژه به اینترنت مربوط می شود) پشتیبانی از نرم افزارهای توزیع شده است ، و این یعنی ، کنترل های اکتیوایکس به شما سرویس خواهند داد ، حتی اگر در کامپیوتری دیگر (و هزاران کیلومتر دورتر) باشند . اما چه بر سر *OLE* آمد ؟ این تکنولوژی دوباره به وضعیت اولیه اش برگشت و اکنون فقط با سندهای مرکب سروکار دارد.

۷-۳-۴. نقاط ضعف و قوت اکتیوایکس

تکنولوژی اکتیوایکس بهترین وسیله برای ایجاد محتویات فعال در وب است . البته برخی با این عقیده مخالفند و باید گفت که این تکنولوژی مسلما تنها ابزار محتویات فعال وب نیست . پس اجازه دهید نگاهی به جنبه های مثبت و منفی این تکنولوژی بیندازیم.

نقاط قوت

یکی از مهمترین نقاط قوت اکتیوایکس قدرت آن است . یک کنترل اکتیوایکس تقریباً از عهده هر کاری که یک برنامه معمولی بتواند انجام دهد ، برمی آید . از دیدگاه یک برنامه نویس وب این بهترین جنبه یک ابزار خلق محتویات دینامیک است . دیگر ابزارهای محتویات فعال ، مانند جاوا و *CGI* ، در این زمینه بسیار محدودتر از اکتیوایکس هستند . نکته مثبت دیگر ، حداقل برای بسیاری از افراد ، امکان استفاده از مهارتهای عادی برنامه نویسی در خلق کنترل های اکتیوایکس است .

روش انجام بارگیری های وب هم یکی از مزایای اکتیوایکس است . وقتی به یک صفحه وب که عناصر اکتیوایکس دارد می روید ، وقایع ذیل اتفاق می افتد :

۱. اطلاعات مختصری درباره کنترل و شماره ویرایش آن بار می شود.
۲. کاوشگر بررسی می کند که آیا این نرم افزار در کامپیوتر شما نصب شده یا خیر.
۳. اگر این نرم افزار در سیستم شما وجود نداشت (یا اینکه ویرایش آن قدیمی تر بود) ، کاوشگر آنرا بار کرده و سپس نصب و اجرا خواهد کرد.
٤. اگر نرم افزار در سیستم شما وجود داشت ، کاوشگر آنرا از همان جا اجرا خواهد کرد . حتما متوجه مزیت این روش شده اید : هر قطعه نرم افزاری فقط یکبار باید بار شود و دفعات بعد دیگر نیازی به بار شدن آن نخواهد بود .

بنابراین صرفه جویی زیادی در زمان نمایش آن صفحه وب خواهد شد . در ابزارهایی مانند جاوا ، این قبیل نرم افزارها باید هر دفعه مجددا بار شوند و این اجرای آنها را به مراتب کندتر خواهد کرد.

نقاط ضعف – هیچ چیز کامل نیست !

[9] Component Object Model

شاید مهمترین جنبه منفی اکتیوایکس ضعف امنیتی آن باشد . این ضعف ظاهرا یکی از تبعات اجتناب ناپذیر قدرت و انعطاف تکنولوژی اکتیوایکس است . یک برنامه نویس شرور می تواند براحتی با این تکنولوژی نرم افزاری بسازد که به کامپیوتر دیگران صدمه بزند . با توجه به این وضعیت ، آیا می توان هنگام برخورد با کنترل های اکتیوایکس روی اینترنت احساس راحتی کرد ؟ پاسخ این سئوال مثبت است و در ادامه توضیح خواهیم داد که چگونه میکروسافت اقدامات امنیتی خاص را برای این تکنولوژی پیاده سازی کرده است . با این وجود همواره این مطلب را در نظر داشته باشید که هیچ سد امنیتی ذاتاً نفوذ ناپذیر نیست.

یکی دیگر از نقاط ضعف تکنولوژی اکتیوایکس آن است که فقط برخی از کاوشگرهای امروزی از آن پشتیبانی می کنند . اگر در یک صفحه وب نرم افزارهای اکتیوایکس وجود داشته باشد و فردی با یک کاوشگر که از اکتیوایکس پشتیبانی نمی کند این صفحه را باز کند قادر به استفاده از مزایای آن نخواهد بود . خوشبختانه این مشکل آنچنان که به نظر می آید حاد نیست ، چون دو تا از مهمترین کاوشگرهای وب *(Netscape Navigator , Internet Explorer)* از این تکنولوژی پشتیبانی می کنند (اولی مستقیم و دومی از طریق یک افزودنی قابل نصب) به هر حال ، وقتی با برنامه نویسی اینترنت سروکار داریم ، این مشکل اساساً وجود ندارد چون این ما هستیم که شبکه را کنترل می کنیم.

مشکل دیگر رفتار کنترل های اکتیوایکس آن است که تمام آنها روی سیستم شما جا خوش کرده و فضای هارددیسک را اشغال خواهند کرد . اما بنظر میاید نگرانی در این مورد هم بیهوده است چون عناصر اکتیوایکس معمولا کوچکند و چند مگابایت جایی که احتمالا (صدها کنترل اکتیوایکس) اشغال خواهند کرد در هارد دیسک های بسیار بزرگ امروزی نمی توانند مشکلی ایجاد کند.

۷-۳-۵. وضعیت جاوا چگونه است ؟

جاوا یکی دیگر از تکنولوژیهای عمده ایجاد محتویات فعال در اینترنت است. اپلت های جاوا ، که توسط کاوشگر بار و اجرا می شوند ، دارای بسیاری از قابلیت های اکتیوایکس هستند ، اپلت های جاوا برای جلوگیری از اقدامات خرابکارانه ، فاقد توانایی های خاص (از قبیل دسترسی به سیستم فایل) هستند و این آنها را بسیار محدود کرده است . از نظر تئوری ، اپلت های جاوا ایمن هستند چون اساساً نمی توانند کارهای خطرناک انجام دهند !

۷-۳-۶. درک مفهوم جاوا و کنترل های اکتیوایکس

جاوا یک زبان برنامه نویسی کامپیوتر است که توسط شرکت *Sun* به بازار عرضه شده است تا به وسیله آن برنامه نویسان قادر باشند برنامه هایی را برای مرورگرهای وب بنویسند که کار با آنها برای استفاده کنندگان بسیار راحت باشد ابزارهای استاندارد شبکه وب به فعالیت ها و عملکردهایی که از صفحه وب قابل دسترسی هستند محدود میشوند . با یک نرم افزار کامل برنامه نویسی مانند جاوا محدودیت های کمی برای طراحی عملکردهای دورن یک صفحه وب وجود دارد.

برنامه های جاوا را با نام جاوا *Applet* نامگذاری کرده اند . *Applet* توسط صفحات وب بر روی یک سرویس دهنده وب نگهداری می شود . وقتی که شما یک صفحه وب را که دارای قسمت هایی است که بوسیله جاوا برنامه ریزی شده نگاه می کنید برنامه های جاوا بصورت خودکار در هنگام دیدن آن صفحه وب از روی اینترنت گرفته و بر روی کامپیوتر شما قرار می گیرند . برای انجام این کار ، احتیاجی نیست شما کاری انجام دهید.

نرم افزار *IE* خود می داند که چگونه برنامه های جاوا را اجرا کند . یک برنامه جاوا به همان سرعت که به دستگاه شما می رسد بر روی آن نیز اجرا می شود.

برای اینکه کامپیوتر خود را بصورت مطمئن نگهداری کنید نرم افزار *IE* مدلی از امنیت جاوا را به شما ارائه می کند که در آن اجازه اجرای کلیه اپلت های جاوا داده شده است . بدون اینکه در مورد آسیب دیدن برنامه های کامپیوتر تان یا نفوذ به اطلاعات شخصی خودتان که بر روی دستگاه شما موجود است نگران باشید.

Javascript ساده ترین زبان برنامه نویسی کامپیوتر است که برای طراحی اجزاء صفحه وب شما بکار می رود.

Javascript بوسیله شرکت *Netscape* به بازار عرضه شده است . برخلاف جاوا این نرم افزار نمی تواند برای نوشتن برنامه ها یا *Applet* بکار رود. *Javascript* فقط برای اعمال ساده نظیر روشن کردن یک کلمه وقتی که شما با ماوس بر روی آن می روید و یا تغییر شکل یک نشانه به شکلی که شما تصور کنید آن را فشار داده اید بکار می رود.

Internet Explorer قادر است که *Javascript* را اجرا کند و صفحاتی را که در آنها از اجزایی استفاده شده که بوسیله *Javascript* برای همگان قابل استفاده شده است. ممکن است در نوشتن آنها مشکلاتی پیش آید و در نتیجه امکان دارد شما در هنگام دیدن صفحات وب با اشکالاتی در رابطه با دستورات *Javascript* مواجه شوید و عملکردهای *Javascript* بر روی دستگاه شما به شکل درستی اجرا نشود. شرکت *Microsoft* دوست دارد که اکتیوایکس را یک مجموعه از امکانات جدید برای ساختن صفحات وب فعالتر معرفی کند بر خلاف جاوا *Microsoft*اکتیوایکس یک زبان برنامه نویسی نیست بلکه یک مجموعه از قطعات نرم افزاری است که توسط دیگر نرم افزارهای برنامه نویسی نظیر جاوا می تواند استفاده شود. اکتیوایکس مانند *Plug – ins* و *add-ons* قابلیت نرم افزار *IE* را گسترش داده است . همچنین اکتیوایکس قابلیت های نرم افزار جاوا را نیز بهبود بخشیده است . برنامه های اکتیوایکس کنترل نامیده می شوند . مانند اپلت های جاوا ، کنترل ها از روی اینترنت گرفته می شوند و بر روی دستگاه شما اجرا می گردد این عمل هنگامی صورت می گیرد که شما صفحات وبی را مشاهده می کنید که دارای کنترل های اکتیوایکس هستند در گوشه پایین سمت چپ پنجره *IE* شما می توانید جمله (نصب قطعات نرم افزاری) وقتی که کنترلهای اکتیوایکس در حال انتقال به دستگاه شما هستند را مشاهده می کنید.

مواقعی ممکن است شما با پنجره هایی در صفحه مانیتور کامپیوترتان مواجه شوید که بوسیله آنها از شما سئوال می شود که آیا می خواهید کنترل های اکتیوایکس به دستگاه شما منتقل شوند یا نه ؟ تکنولوژی اکتیوایکس در حقیقت قسمتی از مدلهای *COM* (مخفف مدلهای شی ء گرای برنامه نویسی میکروسافت) میکروسافت می باشد.

این مدل به برنامه نویسان اجازه می دهد تا نرم افزارهایی بصورت مستقل برای صفحات اینترنت خود ایجاد کنند و یا برنامه هایی را طراحی کنند که عملیات خاصی را انجام دهد . وقتی که این برنامه ها نوشته و طراحی می شوند می توان از آنها در جاهای دیگر هم دوباره استفاده کرد . در قسمت پایین تعدادی از کنترل های اکتیوایکس که در داخل *IE* بکار می روند آورده شده است :

- نمایشگر فایل های *Power Point*
- رابطه هایی برای بکارگیری بانک های اطلاعاتی
- ساعت های بین المللی
- نقشه راه ها که کار با آنها ساده است.

یکی از بزرگترین امتیازات جاوا و اکتیوایکس که در *Plug – ins* ،*add – ons* و نمایشگر ها وجود ندارد این است که آنها بصورت خودکار کار می کنند و شما احتیاجی ندارید که به پایگاه های اینترنت مراجعه کرده و آنها را به کامپیوتر خود منتقل کنید و مدتی وقت صرف کنید تا فرم های ثبت نام شما را پر نمایید . پس از انتقال آنها به دستگاه مدتی را برای نصب آنها وقت صرف کنید . از زمانی که نرم افزار *IE* توانست اپلت های جاوا و کنترل های اکتیوایکس را اجرا کند. کاربران اینترنت دیگر احتیاجی ندارند که به پایگاههای خاص برای گرفتن اطلاعات مراجعه کنند بلکه اپلت ها و کنترل ها به همان سرعت که به روی دستگاه شما متصل ی شوند در همان فاصله نیز به اجرا در خواهند آمد.

۷-۳-۷. بکار بردن کنترل های اکتیوایکس

کنترل های اکتیوایکس صفحات اینترنتی شما را بصورت زنده در می آورند بطوریکه شما قادر خواهید بود فایل های صوتی زنـده را اجرا کرده و یا نشانگرهای متغییر را ببینید و بسیاری کارهای مشابه دیگر.

اگر شما به یک پایگاه اینترنت که دارای یک کنترل اکتیوایکس هست مراجعه کنید نرم افزار *IE* چک می کند که کـدام کنتـرل هـا بصورت دیجیتالی تایید شده اند . یک کنترل تایید شده دیجیتالی برنامه ای است که بصورت مستقل تایید شده است که دارای ویروس های کامپیوتری نیست و تاثیرات منفی بر روی دستگاه شما ندارد . شما میتوانید پنجره ای بر روی صفحه مانیتور خود ببینید که به شـما اطلاع می دهد که آیا نصب کردن این نرم افزار بر روی دستگاه شما به امنیت آن صـدمه ای نمـی زنـد و یـا اخطـار مـی دهـد کـه در صورت نصب به دستگاه شما بصورت نرم افزاری آسیب میرساند.

نکته :

جریان داده : در اینترنت به جای اینکه یک فایل صوتی بزرگ را قبل از اینکه بشنوید به کامپیوترتان منتقل نکنید از جریان داده استفاده می شود . پایگاههای اینترنتی اطلاعات خود را بصورت یک جریان داده می فرستند ، کامپیوتر شما پس از چند ثانیه از شـروع جریـان داده آن را با خبر کرده و شروع به پخش آن می کند و به همین ترتیب آن فایل صوتی یا تصویری را تا انتها اجرا می کند.

۷-۳-۸. اکتیوایکس و امنیت

مطمئناً تا به حال درباره ویروس های کامپیوتری (برنامه هایی که با آلوده کردن سیستم ها صدماتی به آنها میزنند) چیزهایی شنیده اید . ویروس های مختلفی از انواع بی آزار تا بسیار مخرب وجود دارند و تلاش های زیادی صورت مـی گیـرد تـا جلـوی ایـن آلـودگی و انتشار ویروس ها گرفته شود . اما آیا این احتمال وجود ندارد که ویروسها (یا دیگر نرم افزارهای مخـرب) از طریـق اینترنـت پراکنـده شوند ؟

در روزهای اول وب این خطر چندان جدی نبود چون سندهای *HTML* و فایلهای گرافیکی و تصویری تنها انواع فایل هـایی بـود کـه روی اینترنت جابجا می شد . این قبیل فایلها می توانستند خراب باشند اما در ضمن هیچ خطری برای سیستمی که آنها را بار مـی کـرد نداشتند.

اما با روی کار آمدن محتویات فعال اوضاع دگرگون شد ، چون در این حالت برنامه ها هم جزیی از وب شده بودند و یک برنامه هـم قاعدتاً هر کاری می تواند انجام میدهد (از پاک کردن فایل های کامپیوتری مقصد گرفته تـا دزدیـدن اطلاعـات آن و انتقـال آنهـا بـه جاهای نامعلوم). در اینجا بود که شرکتهای تولید کننده ابزارهای برنامه نویسی وب به ضرورت تمهیدات امنیتی پـی بردنـد . وقتـی در وب گشت و گذار میکنید . مایلید مطمئن باشید کدهایی که بار می کنید سیستم تان را بهم نخواهند ریخت !

برای مقابله با این مشکل بالقوه دو روش اساسی ابداع شد.

روش اول آن بود که اساساً اجازه اعمال بالقوه خطرناک به برنامه های وب داده نشود . این راهی بود که جاوا در پیش گرفت . اپلـت های جاوا قادر به انجام افعال بالقوه مخرب (مانند دسترسی به سیستم فایل) نیستند و می توانید مطمئن باشید که آنهـا بـی خطـر هـستند درست همانگونه که یک چاقوی پلاستیکی بی خطر است !

در روش دوم ، که میکروسافت برای عناصر اکتیوایکس در پیش گرفت ، به جای محدود کردن ذاتی برنامه ها از یک تکنولوژی بنام کد تعیین اعتبار برای تعیین صحت و سقم نرم افزارها استفاده می شود . ایده نهفته در این روش این است که اگر شما بدانید کـه یـک عنصر اکتیوایکس یا هر نرم افزار دیگری از کجا آمده (خالق آن کیست) و تغییری هم در آن صورت نگرفته ، دیگر می توانید با خیال راحت از آن استفاده کنید . به این روش امضای دیجیتالی هم گفته می شود.

۷-۴. پیاده سازی سمت کارگزار سرویس

جنبه دیگر وب سرویس که باید مورد کنکاش و بررسی قرار بگیرد، پیاده سازی سمت کارگزار سرویس است که در این بخش مشکلات را بررسی و راه حلهای مناسب را ارایه خواهیم کرد.

با توجه به اینکه داده کاوی معمولا عملی وقت گیر است، یکی از جنبه های مهم قضیه سرعت کارگزار است که از اهمیت فوق العاده ای برخوردار میباشد. یکی از مشکلات مهم مسئله حضور کاربران مختلف با انبوهی از پایگاههای داده مختلف است که باید توسط کارگزار پشتیبانی شوند. برای حل این معضل ما از تکنولوژی نرم افزاری دسترسی داده اکتیوایکس[10] که یک استانداردکامل در تمام محیطهای برنامه نویسی میباشد و تمام کارگزاران و پلاتفرمهای مختلف از آن پشتیبانی میکنند، استفاده کردیم.

با توجه به انتخاب اکتیوایکس برای مشتری و نکات یاد شده بالا مسایل و مشکلاتی که در انتخاب و پیاده سازی سمت کارگزار باید مورد توجه قرارگیرد، به شرح زیر است :

- بهینه سازی الگوریتمهای داده کاوی سرویس – که در سمت کارگزار پیاده سازی میشود – به سمت سرعت بیشتر در وب و استفاده بهینه از پهنای باند
- انتخاب تکنولوژی سازگار با اکتیوایکس انتخاب شده در سمت مشتری
- پشتیبانی کارگزار از پایگاههای داده مختلف
- انتخاب کارگزار با هزینه پایین، قدرت سرویس بالا و سازگار با همه پلاتفرمها
- رعایت هرچه بیشتر استانداردهایی که در وب و سیستمهای مختلف قابل فعالیت باشند
- پشتیبانی از وب معنایی[11] برای فهم معنای داده های ارسالی و برگشت آنها

در زمینه استانداردها و هزینه پایین کارگزار و سرعت بالای آن، استفاده از تکنولوژیهای بر پایه *HTTP* میتواند راهکار موثری برای این منظور باشد. در ادامه به بررسی برخی تکنولوژیهای موجود و ذکر دلایل انتخاب *CGI* به عنوان تکنولوژی پیاده سازی میپردازیم.

۷-۴-۱. تکنولوژیهای مرتبط با طرف سرویس دهنده

PHP, ASP, Servelet سه نمونه از معروفترین تکنولوژی هایی است که در انتشار صفحات وب دینامیک در طرف سرویس دهنده به کار گرفته می شوند. جدای از اختلاف های موجود میان آنها، هر سه امکاناتی را برای اجرای یک برنامه درکارگزار و تولید خروجی *HTML*به صورت دینامیک فراهم می کنند. در این تکنولوژیها کد برنامه و کدهای *HTML* با هم ترکیب می شوند.

تولید دینامیک صفحات وب و برنامه نویسی به فرم اسکریپت نویسی از مزایای این تکنولوژیها به صورت عام می باشند. اما در نهایت ترکیب محتوا، قالب بندی و منطق برنامه پیچیدگی بیشتری را نسبت به *HTML* سبب میگردند.

۷-۴-۲. PHP[12]

PHP یک زبان اسکریپت نویسی و سمت کارگزار می باشد. یک اسکریپت به کدی از برنامه میگویند که بصورت ابتدایی ترجمه نمیشود و فقط در صورت فراخوانی و یا درخواستی برای اجرا توسط مفسر زبان مورد نظر ترجمه می شود و به خروجی فرستاده می

[10] ActiveX data object DataBase (ADODB)
[11] Web Semantic
[12] Perl HyperText PreProcessor

شود. این به این معنی میباشد که شما دقیقاً کد برنامه خود را در جای مورد نیاز به کار میبرید و رابط اجرایی برنامه آنـرا بـه مفـسر مـورد نظر ارجاع میدهد.

بعد از نصب مفسر *PHP* شما قادر خواهید بود که از کدهای *PHP* به همراه کدهای *HTML* خـود در صـفحات *Web* اسـتفاده کنیـد و همچنین میتوانید اسکریپت های صرفاً *PHP* خود را به عنوان ترمینالهای تفـسیر کننـده داده هـا و رابطهـای ورودی و خروجـی بـر روی سرور خود قرار دهید.

فکر اولیه *PHP*در پاییز سال ۱۹۹٤ توسط *Rasmus Lerdorf* شکل گرفت .در ابتدا نگارشـی از *PHP* در صـفحه شخـصی وی بـه کـار گرفته شد ، تا اطلاعاتی از کسانی که رزومه وی را می بینند ، نگاه داشته شود. اولین نگارش عمومی آن در اوایل سال ۹۵ ارایه شد و بـا نام*"Personal Home Page Tools"* معرفی گردید . که البته شامل پارسری بسیار ساده بود که ماکروهای خاصی را می شناخت و نیـز برخی کاربردهای مشترک در صفحات شخصـی مانند شمـارنده یا *Guestbook* وبرخـی ابزار های دیگر را شامل می شد.

پارســر در نیمه سال ۹۵ بازنویســی شد و با نام *PHP/FI* " نگارش ۲ " ارایه گردید *FI* . نام بسـته نرم افزاری دیگـری از *Rasmus* بـود که فرم های داده *HTML* را تفسیر می کرد . پس از آن بسیاری از *PHP* در کــد های خـود اسـتفاده کردنـد . در میانـه سـال ۹٦ میـزان اسـتـفاده کنندگان به حــدود ۱۵ هـزار سایت رسید . این میــزان در نیمه سال ۹۷ در حدود ۵۰ هزار سایت مختلف بود. در هـمین زمان *PHP*از حالت یک پروژه شخصی درآمد و توســط تیمی توسـعه یافت . این گروه نگارش جدیدی از *PHP* را ارایه دادند و پارسر آن را بازنویسی نمودند *PHP3* .به سرعت مورد استفاده قرار گرفـت . هم اکنـون نیز *PHP4* آخرین نگارش این محصول اسـت کـه در آن از موتور اسکریپت *Zend* برای بدست آوردن قابلیت های بیشتر استفاده شده است.

امروزه ، *PHP3*و *PHP4* بر روی بسیاری از محصولات تجاری مانند *"RedHat's Stronghold web* کارگزار" ارایـه مـی گـردد. هـم اکنون برآورد می شود بیش ۵,۱۰۰,۰۰۰ سایت از*PHP* در طراحی استفاده کرده اند.

دلایل زیادی در محبوبیت و استفاده از این زبان در حال حاضر وجو دارد برخی از این دلایل عبارتند از :

- سرعت: در مقایسه با تکنولوژی نظیر *ASP* به دلیل عدم اسـتفاده *PHP* از موتـور هـای مختلـف و متفـاوت کـدهای *PHP* از سرعت بالایی برخوردارند.
- نزدیک بودن نحو به *C/C++* و *Java:* از آنجائیکه اکثر برنامه نویسان از *C/C++* استفاده کرده اند و بخـاطر محبـوب بـودن بی حد *Java* معمولا با نحو های این دو زبان اکثرا آشنا هستند. *PHP* هم اکثر نحو های خود را شبیه بـه ایـن زبانهـا انتخـاب کرده است
- متن باز بودن*PHP*
- اجرا بر روی پلاتفرمهای مختلف: به دلیل آنکه توسط *GNU C Compiler* در همه پلاتفرمها قابل ترجمه شـدن اسـت و از موتور های خاص هیچ سیستم عاملی برای اجرای کدها استفاده نمی کند قابلیت اجرا بر روی تعداد زیادی از سیستم عامل ها را داراست که این یک مزیت برای توسعه دهندگان سیستم محسوب می شود .

۷-۴-۳. ASP[13]

یکی از این روشها کدنویسی معروف، *ASP* با زبان *VBScript* (یا *Jscript* به انتخاب برنامه نویس) در سمت سرویسگر است. گذشـته از امکانات کدنویسی سمت سرویسگر ، از انسجام و ارتباط بی مانندی با دیگر سرویسهای ویـندوز *NT* برخـوردار اسـت. بطـور مثـال کـد نویسی *Server Index Microsoft* در ارتباط تنگاتنگ با *IIS* و *Server SQL* و با زبان *VBScript* انجام میگیرد.

[13] Active Server Page

ASP یک مدل موفق و روبه رشد در این زمینه است که با سرعت مناسب ، امکانات و آبجکت های آماده بسیار ، قابلیت هـای بـسیار در ارتباط با سرویسهای *NT* ، سادگی برنامه نویسی و امکانات ویژوال برنامه نویس نظیـر آنچـه در *Visual Studio 6.0*آمـده ، قـادر اسـت بخوبی از پس طراحی کاربردهای متکی بر وب بانکهای اطلاعاتی فارسی برآید.از محاسن *ASP* میتوان به موارد زیر را نام برد :

- انعطاف پذیری و وفور صریح واژه های کلیدی برنامه نویسی است.
- روش *CGI* نیاز به کدنویسی کم و استفاده از مزایای آبجکتهای آماده و سرعت فراخوانی هنگام نیاز کاربر و نیـاز بـه ترجمـه برنامه بعد از هر تغییری است.
- دسترسی به توابع پایه سیستم عامل.

لازم به ذکر است که در صورت کد نویسی عادی در *ASP* ، سورس برنامه ها نسبتا حجیم میشود که افـت سـرعت واکـنش را بـه دنبـال دارد. از اینرو توصیه میکنیم که یک آبجکت در سمت سرویسگر (که ترجیحا با زبان *C++* نوشته شده باشـد و از قالـب *Server-Side ActiveX* تبعیت کند) نصب و ثبت شود تا برنامه های *ASP* براحتی با رجوع بدان یک نمونه آماده از آن را فراخوانی کرده و از صـفات و متدهای آن جهت پردازش استفاده نمایند.

۷-۴-۴. Servlet

یک کلاس در زبان جاوا است که در سرویس دهنده اجرا می شود. در ایـن کـلاس از دسـتورات *Printh* بـرای تولیـد یـک خروجـی *HTML*استفاده می شود. تولید دینامیک صفحات وب از مزیتهـای ایـن روش اسـت امـا پیچیـدگی تولیـد کـدهای *HTML* بـه کمـک دستورات *Printh* و در حقیقت ترکیب محتوا قالب و منطق بازهم از معضلات این روش محسوب می گردد.

امروزه نقطه تمرکز بسیاری از تلاشها جداسازی سه عنصر محتوا، قالب و منطـق برنامـه از یکـدیگر در طراحیهـای مبتنـی بـر وب اسـت. *XML* به عنوان یکی از اجزا بسیاری از این تکنولوژیها مطرح است.

۷-۵. CGI[14] پرکاربردترین محیط تولید برنامه های وب

یکی از نخستین سرویس دهنده های وب به نام *NCSA HTTPD* در مرکز ملی *NCSA* جـایی کـه اغلـب موسـسین *Netscape* کـار مـی کردند (و مرورگر *Mosaic* را نیز طراحی و تولید کرده اند) طراحـی شـد. از ویژگیهـای مهـم ایـن سـرویس دهنـده *CGI* بـود کـه در ویرایش ۱,۰ به آن اضافه شد.

از آنجاییکه *NCSA HTTPD* یکی از نخستین سرویس دهنده های وب بود بنابراین *CGI* نیز به استاندارد بالفعل پیاده سازی برنامـه هـای وب تبدیل شد. پشتیبانی *CGI* در آوریل سال ۹۸ به ویرایش ۱۵/۲ سرویس دهنده *CERN* اضافه شد.

زمانی که *NCSA*، *HTTPD* را ارائه کرد . *CGI* تنها راه پیاده سازی برنامه های وب بود. به دلیل سادگی *CGI* و رایگان بودن کـد منبـع آن برای عموم تقریبا تمامی سرویس دهنده های وب تولید شده پس از *HTTPD* از *CGI* پشتیبانی می کنند.

امروزه تمامی سرویس دهنده های وب معروف یونیکس و ویـندوز *NT* از *CGI* پـشتیبانی مـی کننـد ، حتـی سـرویس دهنـده هـای وب مکینتاش نیز نوعی روش مشابه با *CGI* را برای ارتباط با برنامه های خارجی بکار می برند.

۷-۵-۱. یک برنامه CGI چیست ؟

یک برنامه *CGI* بوسیله سرویس دهنده وب و در پاسخ به درخواست ایجاد شده توسط مرورگر وب اجرا میشود . سـرویس دهنـده وب به عنوان واسط میان مرورگر و برنامه *CGI* درخواست مرورگر را به برنامـه و خروجـی برنامـه *CGI* را بـرای پـردازش بـه مرورگـر وب

[14] Common Gateway Interface

ارسال می کند. برای مثال ممکن است برنامه ای نقطه نظرات یک کاربر را دریافت کرده و آن را به شکل یک پیام الکترونیکی برا مسئول وب سایت ارسال کند.

تقریبا هر نوع زبان برنامه نویسی را می توان برای نوشتن یک برنامه *CGI* بکار برد. *CGI* واسط تعریف شده ما بین سرویس دهنده وب و برنامه خارجی است که مایلید آن را بنویسید.

حالا ببینیم یک برنامه *CGI* چه کاری انجام نمی دهد.

این برنامه نمی تواند بطور مستقیم با کاربر ارتباط برقرار کند. همچنین نمی تواند اطلاعات را از خط اعلان منوها یا سایر بخشها تعاملی دریافت یا نمایش دهد. این برنامه تصاویر گرافیکی را نیز نمایش نمی دهد. اگر چه می توانند داده های باینری که در حقیقت تصویر هستند را تولید کنند. اما پنجره یا چیزهای مشابه آن را برای برقراری ارتباط از طریق یک واسط گرافیکی در اختیار قرار نمی دهند.

یک برنامه *CGI* برای عملکرد صحیح و مناسب نیازمند شرایط زیر است :

- برنامه با وارد کردن نام آن در خط فرمان قابل اجرا باشد.(برای مثال برنامه های جاوا از طریق ماشین مجازی جاوا (*vertual Machine*) و به شکل *Java Programname* اجرا می شوند اما این وضعیت مناسب برنامه های *CGI* نیست)
- برنامه باید یک سرآیند[15] معتبر از نوع محتویات[16] تولید کند. محتویات خروجی یک برنامه *CGI* می تواند کد *HTML* تصاویر *CGI* ، فایلهای متنی ، مستندات مایکروسافت وردو یا فایلهای صوتی باشد. سرآیند نوع محتویات که توسط برنامه *CGI* تولید می شود نوع محتویات بازگشتی را مشخص می کند و مرورگر می تواند عملیات مناسب را بر اساس آن انجام دهد.

مادامیکه سرویس دهنده وب بتواند برنامه ای را اجرا کند و این برنامه نیز خروجی معتبری تولید نماید کاربرد آن به عنوان برنامه *CGI* قابل قبول است.

۷-۵-۲. نقاط قوت و ضعف CGI

نوشتن برنامه های وب با استفاده از *CGI* دارای مزایا و معایبی است. مزایای برنامه نویسی *CGI* به شرح زیر است :

- مهمترین مزیت *CGI* پشتیبانی آن از محیطهای متفاوت است به عبارت دیگر بر روی انواع سرویس دهنده های وب با سیستم عامل یونیکس ، ویندوز *NT* و تقریبا هر نوع سرویس دهنده وب دیگر کار می کند.بنابراین با نوشتن یک برنامه *CGI* می توانید مطمئن باشید این برنامه به محیطهای دیگر نیز قابل انتقال و اجراست .
- مزیت دیگر آن استقلال از زبانه برنامه نویسی است. در بیشتر موارد می توانید برنامه های *CGI* را با هر زبانی که مایلید بنویسید. بنابراین نیازی به یادگیری یک زبان جدید برای نوشتن آنها وجود ندارد. اگر یک زبان چند محیطی مانند پرل را انتخاب کنید، براحتی می توانید برنامه های موجود را از سیستم عامل یونیکس و ویندوز *NT* یا بلعکس منتقل کنید.
- مزیت دیگر *CGI* سادگی آنست. برای ایجاد یک برنامه *CGI* نیازی به کتابخانه های خاص یا نوشتن برنامه هایی برای بکارگیری *API* مشخص وجود ندارد. برنامه های *CGI* متکی به مفاهیم استاندارد یونیکس در مورد ورودی استاندارد، خروجی استاندارد و متغییرهای محیطی برای برقراری ارتباط با سرویس دهنده وب می باشد.

حال به معایب *CGI* می پردازیم :

---

[15] Header
[16] Content

- بزرگترین عیب برنامه های *CGI* زمانی بروز می کند که توسط زبانهای اسکریپت نویسی نوشته شوند. با هر بار درخواست یک برنامه *CGI* مفسر زبان اسکریپت نویسی اجرا می شود و اسکریپت پس از ارزیابی اجرا می شود. این واقعیت که با هر درخواست اسکریپت *CGI* پرل لازم است تا مفسر پرل اجرا شود بسیار ناکار آمد است. البته این مسئله به توانمندی سرویس دهنده وب ، تعداد درخواستهای اسکریپت *CGI* و زمان لازم برای بارگذاری برنامه *CGI* بستگی دارد. افرادی که با زبانهای ترجمه ای نظیر *C* برنامه می نویسید با این مشکل روبرو نیستند. در حقیقت بسیاری از سرویس دهنده ها برنامه های *CGI* کوچک و سریع الاجرا را به عنوان پل ارتباطی[17] ما بین سرویس دهنده وب و فرآیند سرویس دهنده برنامه بکار می برند.
- یکی دیگر از ایراداتی که به برنامه های *CGI* گرفته می شود آنست که کار برنامه نویسان وب در آن بسادگی سایر محیطهای تولید برنامه وب نیست.

هنگام نوشتن یک برنامه *CGI* لازم است علاوه بر منوی برنامه که عملکرد موردنظر شما را ایجاد می کند که تولید *HTML* صفحه را نیز بنویسید.بسیاری از سرویس دهنده های برنامه معروف امکان ادغام منطبق برنامه در صفحه استاندارد *HTML* را می دهند. این سرویس دهنده های برنامه برای افرادی که با *HTML* آشنا هستند اما برنامه نویسی نمی دانند راحتتر است. اما نوشتن برنامه های سازمان یافته و ساختیافته با این تکنولوژی مشکل تر است. بنابراین بطور مطلق روشی که از همه بهتر باشد وجود ندارد و بیشتر سلیقه شخصی بستگی دارد.

۷-٦. برقرای ارتباط با وب سرویس برای ارسال و دریافت معنایی داده ها

مشکل اصلی و مهم پیاده سازی بحث انتخاب تکنولوژی ارتباط با وب سرویس برای ارسال و دریافت داده های پایگاه داده است. نکته مهمی که باید در این مورد در نظر گرفت انتخاب یک تکنولوژی بر پایه *HTTP* است که هزینه انتقال و سرعت تبادل اطلاعات را افزایش دهد. همچنین باید قادر به تبادل نحو و معنای داده ها باشد تا بتوان در دوسوی ارتباط مشتری / کارگزار داده ها را مورد تفسیر قرار داد. استانداردترین تکنولوژی موجود که در همه عرصه های وب سرویس تکامل کاملی یافته است پروتکل دسترسی ساده شی است، که ما نیز در پیاده سازی از آن استفاده نموده ایم. در فصل اینده هنگام تشریح برنامه بطور مفصلتر در مورد آن توضیح خواهیم داد. اما در ادامه برخی تکنولوژی های موجود را بررسی و پروتکل دسترسی ساده شی را نیز تشریح و راجع به مزایا و معایب آن بحث خواهیم کرد.

۷-۶-۱. ابزارهای داده کاوی جاوا[18]

یک ابزار قدرتمند جاوا برای تسهیل پیاده سازی برنامه های مربوط به داده کاوی است. همچنین این ابزار شامل یک بسته اضافی با نام *(JDM with Web Service Extensions)* میشود که بسیار مناسب برای کار با وب سرویسها است. *JDM* بر پایه *OMG* مدل شی گروه *CORBA* بنا نهاده شده است. اشیا موجود در موتور داده کاوی آن دارای نام یکتا و نوع هستند و از همین طریق قابل ذخیره سازی و بازیابی هستند.

علیرغم حسن بزرگ این بسته مانند تمام بسته های تحت جاوا مبنی بر متن باز بودن آن، استاندارد نشدن آن و ناسازگاری آن با *ActiveX* دو عامل رد شدن آن توسط نویسنده است. اما باید ذکر کرد که از ارزش تحقیقاتی بالایی برخوردار است. [برای مطالعه بیشتر نگاه کنید به *SUN, JDM,JDM 2006*]

---

[17] Gateway
[18] Java Specification Request 73: Java Data Mining (JDM) Version 1.0

۷-۶-۲. ebXML :

ebXML کوشش استانداردی برای تعامل های کسب و کاری است که ابتکاری از *OASIS* و *UN/CEFACT* می باشد و می تواند جانشینی گسترده تر و قابل انعطاف تر برای تبادل داده الکترونیکی[19] تلقی گردد . شرکت ها جهت هدایت کسب و کاری با یکدیگر باید کارهای زیر را انجام دهند :

- کشف محصولات و سرویس های عرضه شده
- تعیین اینکه این محصولات چگونه می توانند با تعیین فرآیند مشترک و تبادل اطلاعات بدست آیند.
- توافق برسر فرم ارتباط و نقاط اتصال برای تبادل مستندات
- توافق برسر عبارات قراردادی همچون تراکنش ها ، طرح ها و امنیت

رجیستری *ebXML* سرور مرکزی است که داده های لازم را ذخیره می کند . هر شرکت باید پروفایل خود *Collaboration Protocol Profile (CPP)* ، که برخی از فرآیندهای کسب و کاری او را مشخص می کند و برخی *Business Service Interface* های پشتیبانی شده را ثبت نماید . فرآیندهای کسب و کاری همان فعالیت های شرکت هستند و واسط های سرویس ، چگونگی انجام تراکنش های لازم در فرآیندهایش را توصیف می کنند . شرکت ها از *CPP* جهت توافق برسر اصطلاحات قراردادی و ایجاد یک توافقنامه همکاری[20] استفاده مینمایند. *ebXML* برای توصیف تعاملات کسب و کاری دارای دو دید می باشد : دید عملیاتی کسب و کار و دید سرویس وظیفه مندی . اولی معناشناسی داده کسب وکار بعلاوه قراردادهای عملیاتی ، توافقنامه ها ، وظایف متقابل و نیازمندیها را مشخص می سازد . دومی ، با سرویس های پشتیبانی همچون قابلیتهای وظیفه مندی ، واسطهای سرویس کسب وکار و پروتکل ها سرو کار دارد.

ebXML به عنوان مکملی برای سایر تکنولوژی ها تلقی می گردد [*ebXML*] .این قبیل استانداردها در یک معنای عام برای قابلیت اندرکنش معنایی بسیار مناسب هستند اما در این پایان نامه تأکید زیادی بر آن نداریم.

۷-۷. پروتکل ساده دسترسی شی[21]

SOAP یک پروتکل مبتنی بر *XML* برای رد و بدل کردن اطلاعات بین برنامه ها است. اطلاعات در *SOAP* به صورت پیام و از طریق پروتکل‌های موجود در اینترنت مانند *HTTP* منتقل می‌شود. (*SOAP* در سایر پروتکل ها، مانند *SMTP* یا *MIME* نیز قابل استفاده است). به زبان ساده‌تر، *SOAP* یک پروتکل است برای دستیابی به یک سرویس ارایه شده در وب. آخرین نسخه*SOAP* ، نسخه ۱,۲ می‌باشد.

۷-۷-۱. ویژگی های SOAP

- یک پروتکل ارتباطی است .
- برای ارسال پیام استفاده می‌شود .
- برای محیط اینترنت و شبکه طراحی شده است .
- وابسته به محیط پیاده سازی و اجرا نیست.

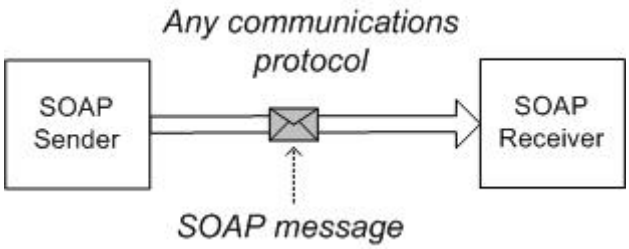
---

[19] EDI - Electronic Data Interchange
[20] CPA - Collaboration Protocol Agreement
[21] Simple Object Access Protocol

- مبتنی بر *XML* است .
- از دیوارهای آتش گذر می‌کند ودیوارهای آتش مانع آنها نمی شوند نمیشوند.

یکی از مسایلی که در دهه اخیر از اهمیت خاصی برخوردار بوده، چگونگی ارتباط برنامه های تحت اینترنت با یکدیگر بوده است. همانطور که می‌دانید برنامه های عادی از*RPC* [۲۲] برای فراخوانی اشیاء *DCOM* یا*CORBA* ، استفاده می‌کنند. اما مشکلی که در این نوع فراخوانی‌ها در بستر اینترنت وجود دارد، مسدود شدن این نوع ترافیک‌ها در کارگزارن پراکسی و دیوارهای آتش است .

در صورت استفاده از *SOAP* با این مشکل روبرو نخواهید بود *SOAP* .به راحتی شما را قادر خواهد کرد تا بین برنامه هایی که در بسترهای متفاوت طراحی شده اند و در بسترهای متفاوتی در حال سرویس دهی هستند، ارتباط برقرار کنید .

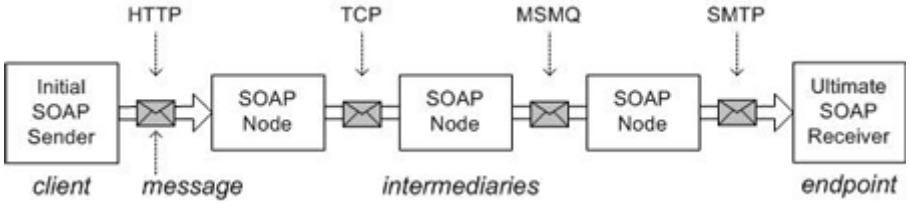


شکل ۷-۱. شیوه ارتباط *SOAP*

۷-۷-۲. ساختار SOAP

پیام ها در *SOAP* یک فایل *XML* هستند که از ساختار زیر پیروی می‌کنند :

- یک بخش ضروری که به آن پاکت نامه[۲۳] گفته می‌شود که مشخص می‌کند که این *XML* یک پیام*SOAP* است .
- قسمت سرآیند که اختیاری است. این بخش شامل اطلاعاتی در مورد خود برنامه است. در صورتی که از سرآیند استفاده شود، باید اولین عنصر در ساختار پاکت باشد .
- قسمت بدنه که ضروری است و شامل فراخوانی یا پاسخ است. در واقع مشخص کننده درخواستِ برنامه‌ی سرویس گیرنده یا پاسخ برنامه سرویس دهنده است .
- قسمت خطا که اختیاری است و اطلاعاتی درباره خطاهای بوجود آمده در هنگام پردازش پیام در خود دارد .



شکل ۷-۲. ساختار *SOAP*

قوانین مهمی که در ساختار پیام *SOAP* وجود دارند عبارتند از :

- پیام حتماً باید در قالب *XML* باشد .
- باید از *Namespace* تعریف شده در پاکت پیروی کند .

[22] Remote Procedure Call
[23] Envelope

- فقط باید از نوع داده های تعریف شده و مجاز استفاده کند .
- در قالب پیام، نباید از *DTD* استفاده شود .برای یک *XML* ، مانند نمای طراحی یک جدول در پایگاه داده است و مشخص می کند که فیلدهای آمده در *XML* از چه نوع هستند و با چه ترتیبی می‌آیند.
- نباید شامل دستورات پردازشی باشد .

۷-۷-۳. قالب کلی پیام

قالب پیام به صورت زیر است :

```
<?xml version="1.0" ?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
      soap:Header>
          ...
      </soap:Header>
      <soap:Body>
          ...
            <soap:Fault>
                ...
            </soap:Fault>
      </soap:Body>
</soap:Envelope>
```

۷-۷-۴. یک درخواست و پاسخ آن با SOAP

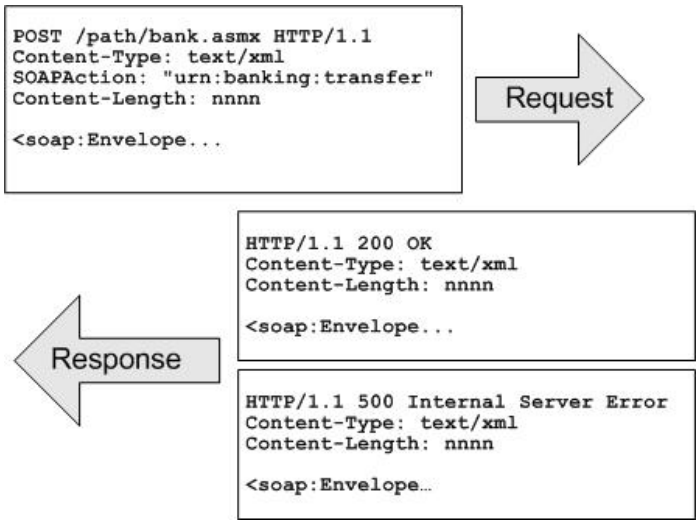
هنگام استفاده از پروتکل *HTTP* ، در هر درخواست باید *Content-Type* و *Content-Length* مشخص شود. که برای *SOAP* ، موارد ارسالی در مثال زیر، به طور معمول مورد استفاده قرار می گیرند .

در این مثال ، درخواست قیمت سیب و پاسخ آن آورده شده است. مشتری یک *XML* را به کارگزار میفرستد که در آن قالب مشخص شده توسط برنامه کارگزار رعایت شده است و در خواست مشتری در آن قرار دارد. در این مثال، قیمت سیب، موردنظر است که در برچسب *m:GetPrice* ، آمده است. در صورتی که قالب تعیین شده توسط کارگزار این اجازه را به شما بدهد که چند مورد را در یک در خواست بفرستید، می توانید این کار را انجام دهید .

برنامه کارگزار نیز، با استفاده از یک فایل *XML* پاسخ مشتری را می دهد و قیمت را در یک برچسب با عنوان *m:GetPriceResponse* به مشتری تحویل می دهد.

```
POST /InStock HTTP/1.1
Host: http://www.stock.org/
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn


<?xml version="1.0" ?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
    <soap:Body>
          <m:GetPrice xmlns:m="http://www.w3schools.com/prices">
                <m:Item>Apples</m:Item>
          </m:GetPrice>
    </soap:Body>
</soap:Envelope>

HTTP/1.1 200 OK
Content-Type: application/soap; charset=utf-8
Content-Length: nnn


<?xml version="1.0" ?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
```

```
    <soap:Body>
          <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
                <m:Price>1.90</m:Price>
          </m:GetPriceResponse>
    </soap:Body>
</soap:Envelope>
```



شکل ۷-۳. ارسال و دریافت پیام *SOAP*

۷-۷-۵. معایب SOAP

همانطور که می‌دانید اولین حرف از حروف تشکیل دهنده ی *SOAP*، *S*است که حرف اول *Simple* است. همین مورد، باعث شده است تا سادگی بر هرچیز در این سیستم، مقدم باشد .برای همین در *SOAP* بسیاری از کاستی ها دیده می‌شود، که یکی از مهمترین آنها امنیت و قابلیت اعتماد پایین در *SOAP* است.

همین کاستی باعث شده است که تولیدکنندگان نرم‌افزار به این فکر بیفتند تا *SOAP* را توسعه دهند و استانداردهای جدیدتری با امکانات بیشتری تولید کنند. استاندارد تولید شده توسط مایکروسافت با نام*GXA* [۲۴] ارایه شد. که یک پیاده‌سازی ازآن وب سرویس بهبود یافته[۲۵] است. این ابزار یک ابزار قدرتمند است که شما با استفاده از *DotNet Framework* و *WSE* می توانید وب سرویس های امن و قدرتمند بنویسید. به بیان ساده‌تر *WSE* ابزار شما برای طراحی و ساخت وب سرویس ها با *.NET* می باشد. *WSE* را می توانید از سایت مایکروسافت بارگیری کرده و نصب کنید.

۷-۷-۶. مزایای SOAP

آنچه *SOAP* را برای پیاده سازی مورد نظر ما جذاب کرده است عبارتست از :

- توسط *IEEE, W3C* کاملا استاندارد شده و مورد پذیرش همه میباشد.
- کار با *XML* به عنوان اصلی ترین و مهمترین سند دادهای روی وب.
- وب بر اساس *HTML* است و *SOAP* به خوبی به عنوان یک لایه بر روی *HTML* کار میکند و با آن سازگار است.
- دسترسی آسان و ارزان فیمت به سرورهای بر پایه *HTTP*

[24] Global XML Web Services Architecture
[25] Web Services Enhancements

- قدرت بالای *SOAP* در ارتباط بی نقص بین مشتری و کارگزار .
- قابلیت فوق العاده انتقال اطلاعات با استفاده از *Soap-Attachment* .

[برای مطالعه بیشتر نگاه کنید به *W3C, SOAP, SOAP 1.2*]

۷-۸. خلاصه

در این فصل در سه بخش کلی به بررسی حوانب پیاده سازی ابزارهای داده کاوی با وب سرویسها پرداختیم. در بخش اول آن مسایل مربوط به پیاده سازی سمت مشتری (نیاز به واسط ساده و کارامد، پایگاههای داده مختلف، جمع آوری ورودیهای لازم برای داده کاوی و ...) مورد بررسی قرار گرفت که از میان آنها اکتیوایکس به عنوان یک تکنولوژی خوب به دلیل قدرت کامل برنامه نویسی، ارایه کنترل کامل، سازگاری با پایگاههای داده، سازگاری با وب، امنیت دیجیتالی و ... پیشنهاد گردید.

در بخش دوم به مشکلات سمت کارگزار پرداخته شد. مشکلاتی از قبیل حجم داده ها، سرعت، هزینه پیاده سازی، استانداردسازی و ... . در این میان نیز با بررسی تکنولوژیهای موجود *CGI* بدلیل استقلال برنامه نویسی، استقلال پلاتفرم، سازگاری با محیطهای مختلف، سادگی و ... پیشنهاد گردید.

در بخش آخر نیز *SOAP* به عنوان بهترین استاندارد موجود در زمینه کار با وب سرویس برای تبادل اطلاعات حجیم و حفظ ساختاری آن تشریح و توسعه های پیشرفته تر آن معرفی گردید.

در فصل آتی پیاده سازی انجام شده برای ابزار داده کاوی مهم درخت تصمیم از طریق وب سرویسها مورد کنکاش کامل قرار خواهد گرفت.

فصل هشتم
پیاده سازی درخت تصمیم با وب سرویس

Implementation Problems And Resolves

در ابتدای این فصل درخت تصمیم و کلاس مربوط به آن که در پیاده سازی مورد استفاده قرار گرفته، را تشریح میکنیم. در ادامه الگوریتم مورد استفاده وب سرویس برای ساخت درخت تصمیم که با *SQL* پیاده سازی شده است را بررسی میکنیم. در انتها نیز مسایل مربوط به اتصال برنامه مشتری به وب سرویس و تبادل اطلاعات بین آنها از طریق *SOAP* را شرح خواهیم داد و ...

- مقدمه
- شی درخت تصمیم
- اتصال به وب سرویس
- ارسال پایگاه داده برای وب سرویس
- دریافت تصویر درخت تصمیم
- انتشار وب سرویس در اینترنت
- چند نکته کلیدی
- خلاصه

۸-۱. مقدمه

با توجه به آنچه در فصلهای قبل گفته شد و رشد سازمانها به سمت استفاده از وب برای کارهای روزمره سازمان و توسعه فراگیر تجارت الکترونیک و استفاده از وب سرویسها به عنوان بهترین راه حل برطرف کردن نیازها و همچنین اهمیت بی حد و حصر داده کاوی برای

تصمیمات حیاتی سازمان، به نظر میرسد وضوح مقولات تئوریک کار به حد کافی روشن گشته است و نیاز به یک پیاده سازی عملی از کار به عنوان شروع زمینه های تجاری و ادامه آن توسط دیگر علاقه مندان به این مقوله یک نیاز اساسی باشد.

دراین راستا نویسنده اقدام به پیاده سازی عملی ابزارهای داده کاوی با وب سرویس ها نموده است که خوشبختانه با موفقیت به پایان رسیده است و نتایج عملی وتحقیقی خوبی داشته است. یکی از ابزارهای مهم و حیاتی داده کاوی درخت تصمیم است که مورد استفاده نویسنده قرار گرفته است. در فصل سوم این ابزار بطور کامل تشریح و اهمیت آن در تجارت امروز به خوبی روشن شده است. در این پیاده سازی از بورلند دلفی استفاده شده است. کدهایی که در ادامه میایند نیز کد دلفی هستند و بر اساس پاسکال شی گرا نوشته شده اند.

در بخش اول این فصل ساختار استفاده شده برای درخت تصمیم -که در آن از ساختمان داده پویای درخت دودویی یا باینری با تکنیکهای برنامه نویس پویا استفاده شده است - را تشریح مینماییم. در ادامه به بررسی وب سرویس نوشته شده و الگوریتم ساخت درخت تصمیم در سرویس ذکر شده میپردازیم. این الگوریتم بر پایه *SQL* است و دلایل این انتخاب نیز شرح داده خواهد شد. انتهای فصل نیز به توضیح شیوه ارتباط برنامه مشتری با وب سرویس و ارسال جداول اطلاعاتی به سرویس از طریق *SOAP* و دریافت اطلاعات مربوط به درخت از آن تحت وب میپردازیم.

امید است این پیاده سازی راهگشای دیگر علاقه مندان به این ایده نو و مهم باشد. ان شاءا...

۸-۲. ساختمان داده درخت تصمیم

ساختار درخت تصمیم مورد استفاده در برنامه بر اساس ساختمان داده مربوط به درخت دودویی یا باینری شکل گرفته است. کد مربوط به تعریف آن در زیر آمده است :

```
type
  PDTNode = ^DTNode;
  DTNode = record
    Caption, REdjeCap, LEdjeCap: String;
    RChild, LChild: PDTNode;
  end;
```

در ابتدا یک گره از درخت تصمیم *TDTNode* تعریف شده است. در ساخت درخت تصمیم هر فیلد از جدول در یکی از دو دسته زیر قرار میگیرد :

۱. پیشگویی کننده

۲. وابسته یا هدف

فیلدهای پیش بینی کننده مقادیری هستند که در هر سطح از درخت از طریق آنها داده ها به دو بخش مجزا تقسیم میشوند. برچسب روی یالهای خارج شده از درخت معیار تقسیم بندی را نشان میدهند. هر گره از درخت نیز سه خصوصیت از نوع رشته دارد :

۱. عنوان *(Caption)* که نام فیلد پیش بینی کننده در آن ثبت میشود.

۲. عنوان یال سمت راست *(REdjeCap)* که معیار تقسیم بندی داده ها در سمت راست گره را در خود ذخیره میکند.

۳. عنوان یال سمت چپ *(REdjeCap)* که معیار تقسیم بندی داده ها در سمت گره را در خود ذخیره میکند.

همچنین دو اشاره گر *LChild* برای فرزند سمت چپ و *RChild* برای فرزند سمت راست برای هر گره در نظر گرفته شده است.

چنانچه هر دوی اشاره گرها دارای مقدار تهی باشند، آنگاه گره یک برگ است و عنوان گره، متناظر با مقدار فیلد وابسته یا هدف روی آن مسیر از ریشه تا برگ میاشد. برای اطلاعات بیشتر مربوط به گره های درخت تصمیم، فصل سوم را مطالعه کنید.

در ادامه به تعریف یک درخت تصمیم متشکل از این گره ها میکنیم. کد مورد نظر برای اینکار در پایین آمده است :

```
  TDecisionTree = class(TComponent)
  public
    Tree: PDTNode;
    Caption: String;
    BkImg: TImage;
    constructor Create;
    destructor Destroy;
    procedure EmptyTree;
    procedure AddNode(Cap, RCap, LCap: String);
    procedure DeleteAllNode(p: PDTNode);
    procedure ShowNode(p: PDTNode; top, width: Integer);
    procedure Show;
  end;
```

هر درخت تصمیم یک اشاره گر به اولین گره درخت را در خود نگهداری میکند. این اشاره گر در کد با نام *Tree* مشخص شده است. همچنین هر درخت دارای یک صفت عنوان است که این صفت نام فیلد وابسته یا هدف میباشد که درخت تصمیم قصد دارد داده ها را براساس مقادیر این فیلد تقسیم بندی نماید. یک شی تصویر نیز برای نمایش تصویر درخت و ذخیره سازی ان در صورت نیاز با فرمتهای تصویری مختلف در درخت تصمیم در نظر گرفته شده است.

شی مربوط به درخت تصمیم علاوه بر دو متد سازنده و مخرب دارای ۵ متد دیگر نیز میباشد که در ادامه هر کدام را بررسی میکنیم.

۸-۲-۱. متد سازنده درخت تصمیم

این متد وظیفه ایجاد و مقدار دهی اولیه به صفات درخت را انجام میدهد. کد متناظر با این متد در زیر آمده است :

```
constructor TDecisionTree.Create;
begin
  Tree := nil;
  BkImg := TImage.Create(Self);
  BkImg.Width := 500;
  BkImg.Height := 500;
end;
```

در خط اول اشاره گر به اولین گره درخت برای اجتناب از مقدار دهی های اشتباه حافظه ای به مقدار تهی نسبت داده میشود. سه خط بعدی مربوط به ایجاد شی تصویر درخت و مقدار دهی اولیه به طول و عرض آن میشود.

۸-۲-۲. متد مخرب درخت تصمیم

وظیفه این متد ازبین بردن درخت و پس دادن حافظه مربوط به آن به سیستم عامل میباشد. برای این منظور ابتدا کلیه گره ها از بین میروند و سپس فضای اشغال شده توسط درخت به سیستم عامل برمیگردد. کد این متد در زیر آمده است :

```
destructor TDecisionTree.Destroy;
begin
  EmptyTree;
  Dispose(Tree);
end;
```

برای رعایت اصول شی گرایی و پیمانه ای شدن کار حذف کلیه گره های درخت در یک متد جداگانه و مجزا نوشته شده است. کد مربوط به این عمل در زیر آمده است :

```
procedure TDecisionTree.DeleteAllNode(p: PDTNode);
begin
  if p = nil then exit;
  DeleteAllNode(p^.LChild);
  DeleteAllNode(p^.RChild);
  Dispose(p);
end;
```

برای سادگی این متد را بصورت بازگشتی نوشتیم. به این صورت که چنانچه گره تهی باشد کار خاتمه پیدا میکند و لی در غیر اینصورت فرزند سمت چپ و سپس فرزند سمت راست ازبین میروند. در انتها نیز حافظه اشغالی خود گره آزاد میگردد.

۸-۲-۳. اضافه کردن یک گره جدید به درخت

این متد وظیفه ایجاد و اضافه کردن یک گره جدید در سطح بعدی درخت را دارد. به عنوان پارامترهای ورودی عنوان گره متناظر با نام فیلد پیشگویی کننده و برچسب یال سمت چپ و راست گره متناظر با معیارهای تقسیم بندی به متد ارسال میشوند. به کد مربوط به این متد توجه کنید :

```
procedure TDecisionTree.AddNode(Cap, RCap, LCap: String);
var p, q: PDTNode;
begin
  New(p);
  p^.Caption := Cap;
  p^.REdjeCap := RCap;
  p^.LEdjeCap := LCap;
  p^.RChild := nil;
  p^.LChild := nil;


  if tree = nil then
    Tree := p
  else begin
    q := Tree;
    while q^.LChild <> nil do
      q := q^.LChild;
    q^.LCHild := p;
  end;
end;
```

شش خط اول کد مربوط به اختصاص فضای حافظه لازم برای گره و مفدارد هی به صفت گره از روی پارامترهای ورودی میباشد. در بخش دوم برای اضافه کردن گره جدید به درخت دو حالت پیش رو داریم :

۱. درخت تهی میباشد و این گره اولین گره درخت است. که در این حالت فقط کافیست مقدار صفت مربوط به اشاره گر به اولین گره از درخت تصمیم برابر با گره ایجاد شده شود.

۲. درخت پراست و این گره باید در انتهای درخت اضافه شود. لذا از ریشه شروع و تا زمانی که گره ای وجود دارد در درخت به جلو میرویم. بعد از رسیدن به انتها گره را به فرزند سمت چپ گره انتهایی اضافه میکنیم.

۸-۲-۴. متد نمایش یک گره بر روی تصویر

این متد وظیفه ترسیم یک گره تنها را در محل ارتفاع و عرض داده شده بر روی شی تصویر پس زمینه درخت تصمیم بر عهده دارد. برای این منظور باید ارتفاع مورد نظر – که متناظر با سطح گره در درخت تصمیم میباشد – و عرض گره به عنوان پارامتر ورودی به متد ارسال شوند.

```
procedure TDecisionTree.ShowNode(p: PDTNode; top, width: Integer);
begin
  if p = nil then exit;
  with BkImg.Canvas do begin


    Rectangle(trunc((width / 2) - 40), top, trunc((width / 2) + 40), top +
30);
    TextOut(trunc((width / 2) - TextWidth(p^.Caption) / 2), top + 10,
p^.Caption);
```

ابتدا در وسط عرض داده شده یک مستطیل به طول ۸۰ و عرض ۳۰ نقطه ترسیم میشود. در وسط این مستطیل عنوان گره که همان نام فیلد پیشگویی کننده است نمایش داده میشود. برای ترسیم و نمایش کلمه از توابع ویندوز *Rectanlge* و *TextOut* استفاده شده است.

```
    if p^.LEdjeCap <> '' then begin
      MoveTo(trunc(width / 2), top + 30);
      LineTo(trunc(width / 4), top + 80);
      TextOut(trunc((3 * width / 8) - TextWidth(p^.LEdjeCap) / 2), top + 50,
p^.LEdjeCap);
```

```
      if p^.LChild = nil then begin
        Ellipse(trunc(width / 4) - 40, top + 80, trunc(width / 4) + 40, top +
110);
        TextOut(trunc((width / 4) - TextWidth('True') / 2), top + 90,
'True');
      end;
    end;
```

سپس به فرزند سمت چپ گره نگاه میکنیم. چنانچه یال سمت چپ دارای برچسب باشد یال مربوطه را ترسیم و در وسط آن برچسب مورد نظر را نمایش میدهیم. در این حالت بررسی میکنیم که چنانچه گره دارای فرزند نمیباشد یا بعبارتی گره، یک گره برگ است درون یک بیضی در انتهای یال مربوطه مقدار *True* یعنی مقدار فیلد هدف را نمایش میدهیم. توجه کنید که در الگوریتم فرض شده است که فرزندان سمت چپ مقادیر درست قیلد هدف و فرزندان سمت راست نمایانگر مقدار نادرست از فیلد وابسته یا هدف را نمایش دهند.

```
    if p^.REdjeCap <> '' then begin
      MoveTo(trunc(width / 2), top + 30);
      LineTo(trunc(3 * width / 4), top + 80);
      TextOut(trunc(5 * width / 8), top + 50, p^.REdjeCap);

      Ellipse(trunc(3 * width / 4) - 40, top + 80, trunc(3 * width / 4) + 40,
top + 110);
      TextOut(trunc((3 * width / 4) - TextWidth('False') / 2), top + 90,
'False');
    end;
```

مانند عمل قبلی برای فرزند سمت راست نیز تکرار میشود. اما باید توجه داشت که فرزند سمت راست نمایانگر مقادیر نادرست فیلد وابسته یا هدف هستند.

همچنین برای نمایش کامل یک درخت متد دیگری در نظر گرفته شده است. این متد با پیمایش درخت از ریشه تا برگها و استفاده از تابع بالا برای نمایش تک تک گره ها کار نمایش کامل یک درخت تصمیم را بر عهده دارد.

```
procedure TDecisionTree.Show;
var p: PDTNode;
    top, width: Integer;
begin
  if Tree = nil then exit;
  BkImg.Canvas.FillRect(BkImg.ClientRect);
  p := Tree;
  BkImg.Canvas.TextOut(trunc((BkImg.Width / 2) -
BkImg.Canvas.TextWidth(Caption) / 2), 2, Caption);
  top := 20;
  width := BkImg.Width;
```

در ابتدا این متد تصویر پس زمینه را پاک میکند. سپس در وسط تصویر عنوان درخت که همان نام فیلد وابسته یا هدف است را نمایش میدهد.

```
  while p <> nil do begin
    ShowNode(p, top, width);
    p := p^.LChild;
    top := top + 80;
    width := trunc(width / 2);
  end;
end;
```

در انتها نیز متد تا جایی که گره وجود دارد آنها را با استفاده از متد قبلی که توضیح داده شد نمایش میدهد. هر بار با اضافه کردن ۸۰ نقطه به مقدار *top* گره ها را درسطح پایینتری نمایش میدهد.

۸-۳. الگوریتم ساخت درخت تصمیم

شاید مهمترین و سخت ترین متد مربوط به ساخت درخت تصمیم از یک جدول رابطه ای باشد. در پیاده سازی این الگوریتم از دستورات *SQL* استفاده کردیم. ابتدا به ذکر دلائل خود برای پیاده سازی الگوریتم با *SQL* میپردازیم و سپس کد اصلی را بطور مفصل شرح میدهیم.

۸-۳-۱. دلایل اقبال و رویکرد ما به روش‌ها و الگوریتم‌های بر پایه‌ی SQL

اولا ذکر این نکته ضروری است که تفاوت بین پیاده‌سازی‌ها وقتی خود را نشان می‌دهند که این واقعیت را بیاد آوریم که ما در داده‌کاوی معمولا با مجموعه‌ی داده‌هایی بسیار بزرگ با حجم و ابعاد بالا سر و کار داریم. در زیر مختصرا چند تا از دلایل خود را برای استفاده از *SQL* ذکر می‌کنیم :

- عدم نیاز به حافظه‌های اضافی. تعداد زیادی از الگوریتم‌ها هنگام مواجهه با حجم بسیار بالای داده‌ها، از لحاظ حافظه‌ی اصلی مورد، نیاز به مشکل برمی‌خورند. در حالیکه استفاده از رهیافت *SQL* به حافظه‌ی اضافی نیازی ندارند، زیرا الگوریتم‌ها در همان فضای *DBMS* اجرا می‌گردند.
- انتقال داده‌ها از انبارهای داده‌ای که بستر اصلی پیاده‌سازی این الگوریتم‌ها هستند به محیط‌های زبان‌های دیگر، کاری است بسیار زمانبر، با احتمال رخداد خطای بالا و در ضمن مشکل. برای مثال اگر در نظر بگیریم که ما می‌خواهیم یک الگوریتم خوشه‌بندی داده‌ها را روی یک جدول با بیش از یکصد میلیون رکورد انجام دهیم، مشاهده می‌کنیم که مدیریت صحیح این داده‌ها و نقل و انتقال آن‌ها، خود مساله‌ی پیچیده‌تر از اعمال الگوریتم می‌باشد. تولید کردن عبارات *SQL* در خروجی یعنی فقط تکیه کردن بر روی پایگاهها برای داده‌هایی که بر روی آنها هستند و همچنین یعنی جدا کردن مدیریت الگوریتم‌ها از مدیریت داده‌ها.
- بعلّت پشتیبانی بسیاری از پایگاهها از پرس و جو های موازی، پیاده‌سازی بروش *SQL* می‌تواند به آسانی بطور موازی اجرا گردد.
- امکان استفاده از توانایی‌های ایندکس گذاری پایگاه داده و پردازش پرس‌وجو. *[Sarawagi]* باعث می‌شود که سیستم‌ها قوی، قابل حمل و قابل گسترش شوند. *[Sarawagi, Rajamani98]*
- پشتیبانی پایگاههای داده برای علامتگذاری و مدیریت فضا برای الگوریتم‌های داده کاوی، می‌توانند بسیار ارزشمند باشند. *[Sarawagi]*
- در بسیاری از موارد ممکن است کدهای خروجی حاصل از الگوریتم‌ها، غیر بهینه بوده و از لحاظ هزینه‌ی اجرایی ناکارامد باشند، در حالی که اگر کلیه‌ی الگوریتم‌های ما با عبارات *SQL* بیان شده باشد و این خروجی‌ها برروی پایگاههایی اجرا گردند که دارای بهینه‌گرهای پرس‌وجوهای *SQL* باشند- امروزه اکثر پایگاههای داده مشهور دارای بهینه‌گرها می‌باشند- خودبه‌خود کدهای ما بهینه گردیده و سرعت کار بالا می‌رود.
- آخرین نکته اینکه، چون فاز اجرای الگوریتم‌های ما را پایگاههای داده بر عهده دارند، هرگاه امکانات و قابلیت‌های پرس و جوگیری این ابزارها اضافه گردد، خودبخود پرسو جو های ما نیز از کارایی بیشتری برخوردار خواهند شد *[Rajamani98]*

۸-۳-۲. مشکلات پیاده‌سازی با SQL

علیرغم کلیه‌ی خوبی‌هایی که در اینجا مشاهده کردیم، در این راه به مشکلات اساسی بر خواهیم خورد اهم مشکلاتی که پیش‌بینی می‌گردند، بصورت زیر لیست می‌گردند:

- *SQL* بر پایه‌ی مدل رابطه‌ای بوده، قادر به اجرای عملگرهای رابطه‌ای می‌باشد و از عملگرهای جبر خطی پشتیبانی نمی‌کند، در حالیکه در الگوریتم‌های داده‌کاوی این عملگرها شدیدا مورد نیاز بوده و بوفور استفاده می‌شوند.
- در پایگاههای رابطه‌ای که ما با آن‌ها سر و کار داریم، داده‌ها در جداول ذخیره می‌شوند، بنابراین موجودی به نام آرایه در این بین وجود نخواهد داشت در حالی‌که تصور پیاده‌سازی اکثر الگوریتم‌های داده کاوی، بدون استفاده از آرایه، بیشتر به یک کابوس شباهت دارند تا واقعیت.
- برای عبارت *SQL* یک محدودیت ماکزیمم طول برای پرس و جوها را داریم، در حالی‌که پرس و جوهایی که نتیجه‌ی کارِ ما می‌باشند، معمولا از این مقدار حدی تجاوز خواهند کرد.
- آخرین مشکل اساسی اینکه پایگاه داده، عبارات *SQL* را بصورت موازی اجرا کرده و ممکن است نتایجی را که برگشت می‌دهد، در یک ترتیب غیر قابل پیش بینی باشند.

۸-۳-۳. متد مربوط به الگوریتم ساخت درخت تصمیم

از آنجا که در این الگوریتم مدام به ساخت نماهای جدید نیاز است و ایجاد و حذف جداول و نماها نیاز به اتصال مداوم با پایگاه داده دارد، باید از یک موتور اتصال به پایگاه داده استفاده کنیم. باید توجه داشت که کاربران وب از پایگاههای داده گوناگونی استفاده میکنند. ما به دلایل زیر *ADODB*[1] را انتخاب کردیم :

- کاملا استاندارد شده و در تمام محیطها و پلاتفرمها کار میکند.
- قابلیت اتصال به هر نوع پایگاه داده را دارد.
- برپایه اکتیو ایکس است و با انتخاب ما برای سمت مشتری سرویس سازگارتر است.

یک درخت تصمیم داده های یک جدول را براساس مقادیر فیلد هدف به سطوح مختلف تقسیم میکند. این تقسیم بندی بر اساس فیلدهای پیش بینی کننده در هر سطح انجام میگیرد. با توجه به اینکه درخت تصمیم الگوریتم از نوع دودویی است، بنابراین فیلد هدف را باید بتوان به صورت یک فیلد دو مقداری یا بولینی تبدیل نمود. لذا ورودیهای الگوریتم به شرح زیر است :

۱. نام جدول داده ها
۲. فیلد وابسته یا هدف
۳. لیستی از فیلدهای پیش بینی کننده

الگوریتم بصورت بازگشتی برای هر سطح و هر فیلد پیش بینی کننده داده ها را به دو گروه

۱. داده هایی که فقط دارای مقدار نادرست در فیلد هدف هستند
۲. داده هایی که هم مقدار درست و هم مقدار نادرست یا فقط مقدار درست را در فیلد هدف دارند

تقسیم میشوند. این کار با گروه بندی روی فیلد پیش بینی کننده و فیلد هدف انجام میگیرد. تعداد سطوح درخت برابر با تعداد فیلدهای پیش بینی کننده است. کد زیر را در نظر بگیرید:

```
procedure TDecisionTreeService.BuildDT(DataTableName: String; GoalFieldName:
String; AVCList: TStringArray; Level: Integer);
var RCap, LCap, Temp: String;
    i: Integer;
begin
```

[1] ActiveX Data Object DataBase (MicroSoft)

```
if Level > Length(AVCList) - 1 then exit;
```

DataTableName نام جدول داده ها، *GoalFieldName* نام فیلد هدف و *AVCList* لیست فیلدهای پیش بینی کننده است. برنامه بصورت بازگشتی تا رسیدن تعداد سطوح درخت به تعداد فیلدهای پیش بینی کننده ادامه خواهد داشت.

برای درک هر چه بیشتر موضوع به جدول مثال زیر نگاه کنید :

Insurance		
age	**Cartype**	**highrisk**
23	saden	False
30	sports	False
36	saden	False
25	truck	True
30	saden	False
23	truck	True
30	truck	False
25	sports	True
18	Sadden	False

جدول ۸-۱. جدول اطلاعات یک شرکت بیمه

این جدول قسمتی از اطلاعات یک شرکت بیمه را نشان میدهد. ستون اول سن و ستون دوم نوع ماشین مشتریان را نشان میدهد. ستون سوم مشخص میکند که آیا این مشتری برای شرکت بیمه دارای ریسک است یا نه ؟ قصد داریم براساس فیلد هدف ریسک داده ها را با قیلدهای پیشگویی کننده سن و نوع ماشین بترتیب به دسته های مختلف تقسیم کنیم تا یک دید عمومی نسبت به همه مشتریان پیدا کنیم و قوانین جامعی بدست آوریم.

سطح اول با شماره صفر متناظر با فیلد سن است. جدول را براساس فیلد سن و فیلد ریسک مرتب میکنیم و یک نمای جدید با پیشوند *Group* میسازیم. دستور *SQL* این کار بصورت زیر است :

```
CREATE VIEW Group_0 as
SELECT Age, HighRisk
FROM Insurance
GROUP BY Age, HighRisk
```

پاسخ این پرس و جو در جدول زیر نمایش داده شده است :

Group_0	
Age	**HighRisk**
18	False
23	True
23	False
25	True
30	False
36	False

جدول ۸-۲ . جدول گروه بندی شده با سن و ریسک

در متد اصلی این کار از طریق کد زیر انجام میگیرد :

```
with ADOQuery.SQL do begin
  ADOQuery.Close;
  Clear;
  Add('CREATE VIEW Group' + IntToStr(Level) + ' as');
  Add('SELECT ' + AVCList[Level] + ',' + GoalFieldName);
  Add('FROM ' + DataTableName);
  Add('GROUP BY ' + AVCList[Level] + ',' + GoalFieldName);
  ADOQuery.ExecSQL;
```

در مرحله بعد نمای بالا را به دو نمای جدید تقسیم میکنیم :

۱. آندسته از مشتریان که در نمای بالا داری مقدار نادرست در فیلد هدف یا ریسک هستند.

۲. آندسته از مشتریان که در نمای بالا داری مقدار درست در فیلد هدف یا ریسک هستند.

نمای اول را با پیشوند *False* و نمای دوم را با پیشوند *True* نامگذاری میکنیم. دستور پرس و جوی مربوط به این کار در زیر آمده است :

```
CREATE VIEW False_0 as
SELECT Age
FROM Group_0
WHERE HighRisk = FALSE
```

و

```
CREATE VIEW True_0 as
SELECT Age
FROM Group_0
WHERE HighRisk = TRUE
```

دو نمای تشکیل شده را در جداول زیر ببینید.

False_0
Age
18
23
30
36

True_0
Age
23
25

جدول ۸-۳. نماهای مشتریان با ریسک و بدون ریسک

کد مربوط به ساخت این نماها در متد پیاده سازی شده در زیرآمده است.

```
ADOQuery.Close;
Clear;
Add('CREATE VIEW False' + IntToStr(Level) + ' as');
Add('SELECT ' + AVCList[Level]);
Add('FROM Group' + IntToStr(Level));
Add('WHERE ' + GoalFieldName + '=FALSE');
ADOQuery.ExecSQL;

ADOQuery.Close;
Clear;
Add('CREATE VIEW True' + IntToStr(Level) + ' as');
Add('SELECT ' + AVCList[Level]);
Add('FROM Group' + IntToStr(Level));
Add('WHERE ' + GoalFieldName + '=TRUE');
ADOQuery.ExecSQL;
```

اما این دو نما ممکن است اشتراکاتی با هم داشته باشند. به عنوان مثال ممکن است یک مشتری با سن ۲۳ سال و با ماشین سواری دارای ریسکی برای بیمه نباشد، اما مشتری دیگری با همان سن و ماشین سنگین دارای ریسک باشد. به این منظور ما با حذف اشتراکات از جدول نادرستها جدول حتما نادرست را میسازیم. دلیل این امر آنست که اینگونه افراد در این سطح صرف نظر از فیلدهای پیش بینی کننده بعدی همواره برای شرکت بیمه بدون ریسک باقی خواهند ماند و نیازی به تقسیم بندیهای بعدی وجود ندارد. دستور *SQL* مربوط به اینکار و نمای حاصل را در زیر ببینید. پیشوند استفاده شده برای این نما *OnlyFalse* در نظر گرفته شده است.

```
CREATE VIEW OnlyFalse_0 as
SELECT Age
FROM False_0
WHERE Age not in (SELECT * FROM True_0)
```

OnlyFalse_0
Age
18
30
36

جدول ۸-۴. نمای مشتریان دارای مقدار حتما نادرست در فیلد هدف یا ریسک

توجه کنید که تعداد رکوردهای نمای فوق از تعداد رکوردهای نمای *False* کمتراست. کد متناظر با این عمل در زیر آمده است :

```
    ADOQuery.Close;
    Clear;
    Add('CREATE VIEW OnlyFalse' + IntToStr(Level) + ' as');
    Add('SELECT ' + AVCList[Level]);
    Add('FROM False' + IntToStr(Level));
    Add('WHERE ' + AVCList[Level] + ' not in (SELECT * FROM True' +
IntToStr(Level) + ')');
    ADOQuery.ExecSQL;
```

نمای جدیدی با نام *Data* از جدول اصلی میسازیم که درآن فقط رکوردهایی وجود دارند که در نمای *OnlyFalse* وجود ندارند. این نما داده های لازم برای ساخت سطح بعدی درخت متناظر با فیلد پیشگویی کننده بعدی، به عنوان مثال نوع ماشین، را فراهم میکند. به دستور *SQL* زیر توجه کنید.

```
CREATE VIEW Data_1 as
SELECT *
FROM Insurance
WHERE Age not in (SELECT * FROM OnlyFalse_0
```

داده های این نما در جدول زیر آمده است.

Data		
age	cartype	highrisk
23	saden	False
25	truck	True
23	truck	True
25	sports	True

جدول ۸-۵. نمای ایجاد شده برای مرحله بعدی پردازش

چنانچه نمای فوق دارای رکورد باشد این نماها بصورت بازگشتی به مرحله بعد رفته و مراحل فوق برای تقسیم بندی در یک سطح بالاتر انجام میگیرد. کد مربوط به این قسمت در زیر آمده است.

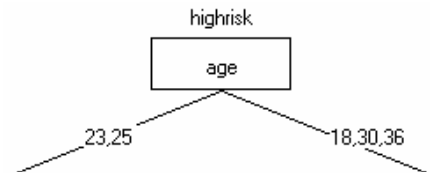
```
    ADOQuery.Close;
    Clear;
    Add('CREATE VIEW Data' + IntToStr(Level + 1) + ' as');
    Add('SELECT *');
    Add('FROM ' + DataTableName);
    Add('WHERE ' + AVCList[Level] + ' not in (SELECT * FROM OnlyFalse' +
IntToStr(Level) + ')');
    ADOQuery.ExecSQL;

    if LCap <> '' then
      BuildDT('Data' + IntToStr(Level + 1), GoalFieldName, AVCList, Level +
1);
```

گره جدیدی را با عنوان فیلد پیشگویی کننده میسازیم. با ترکیب مقادیر داده نمای *OnlyFalse* برچسب یال سمت راست و با ترکیب مقادیر داده نمای *Data* برچسب یال سمت چپ بدست میآید. کد زیر این عمل را پیاده سازی مینماید.

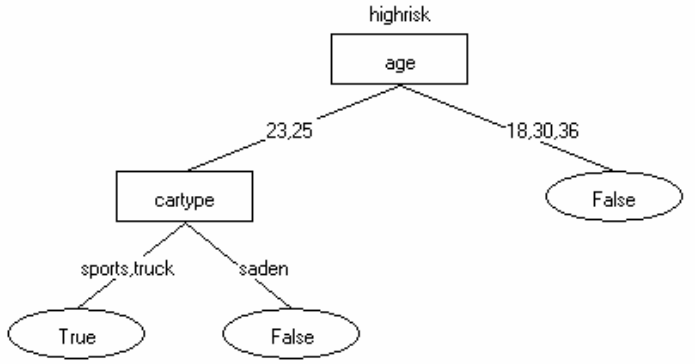
```
    RCap := '';
    ADODataSet.Close;
    ADODataSet.CommandText := 'SELECT DISTINCT ' +  AVCList[Level] + ' FROM
OnlyFalse' + IntToStr(Level);
```

```
    ADODataSet.Open;
    ADODataSet.First;
    if ADODataSet.RecordCount > 0 then
      for i := 1 to ADODataSet.RecordCount do begin
        Temp := ADODataSet.FieldValues[AVCList[Level]];
        RCap := RCap + Temp + ',';
        ADODataSet.Next;
      end;
    if RCap <> '' then RCap[Length(RCap)] := ' ';


    LCap := '';
    ADODataSet.Close;
    ADODataSet.CommandText := 'SELECT DISTINCT ' +  AVCList[Level] + ' FROM
Data' + IntToStr(Level + 1);
    ADODataSet.Open;
    ADODataSet.First;
    if ADODataSet.RecordCount > 0 then
      for i := 1 to ADODataSet.RecordCount do begin
        Temp := ADODataSet.FieldValues[AVCList[Level]];
        LCap := LCap + Temp + ',';
        ADODataSet.Next;
      end;
    if LCap <> '' then LCap[Length(LCap)] := ' ';


    DTree.AddNode(AVCList[Level], RCap, LCap);
```

در شکل های زیر درخت ساخته شده در سطح اول مثال بالا و سپس درخت کامل آن نمایش داده شده است.



شکل ۸-۱. گره ساخته شده در سطح اول با فیلد پیشگویی کننده *age*



شکل ۸-۲. درخت تصمیم مثال

با بررسی این درخت قوانین جالبی بدست میاید. به عنوان مثال افراد بین ۲۰ تا ۳۰ سال با ماشین سواری ریسکی برای بیمه ندارند، حال آنکه همین افراد با ماشینهای ورزشی و سنگین بالعکس عمل میکنند.

۸-٤. پیاده سازی وب سرویس

در این بخش به آن قسمت از کد که به ارتباط میان برنامه مشتری و وب سرویس کارگزار و ارسال و دریافت معنایی داده ها میشود، میپردازیم. بحث را در سه محور اصلی گسترش میدهیم. این سه محور عبارتند از :

۱. ارتباط و اتصال برنامه مشتری با وب سرویس

۲. طریقه ارسال معنایی داده و حفظ نوع داده ها در ارسال و دریافت پاسخ

۳. معرفی و انتشار وب سرویس در اینترنت برای استفاده مشتریان

در هر قسمت از کار مشکلات و نکاتی وجود دارد که به ذکر آنها در هر بخش میپردازیم.

۸-۴-۱. اتصال به وب سرویس

همانطور که در فصل قبلی بطور مفصل توضیح داده شد، برای اتصال به وب سرویس و انجام تبادلات از پروتکل ساده دسترسی به شی که برای استفاده در وب سرویسها استاندارد شده است و بطور وسیعی کاربرد دارد، استفاده میکنیم. برای این منظور نیاز است که قبل از ارسال داده ها یک اتصال به وب سرویس انجام گیرد. این کار با استفاده از شی اتصال *SOAP*[۲] انجام میگیرد.

با این شی میتوان به یک کارگزار وب سرویس اتصال پیدا کرد. اتصال *SOAP* در برنامه های پایگاه داده چند لایه ای مشتری، برای برقراری و نگهداری اتصال بین مشتری و کارگزار استفاده میشود. شی اتصال *SOAP* میتواند کارهای زیر را انجام دهد :

- برقراری و آغاز اتصال به یک کارگزار راه دور
- فراهم کردن یک واسط برای برنامه کارگزار
- فراهم کردن لیستی از سرویس دهنده ها در برنامه کارگزار
- قطع اتصال با کارگزار راه دور

اتصال *SOAP* یک اتصال بین برنامه مشتری و کارگزار راه دور را با استفاده از پروتکل *SOAP* برقرار و آغاز میکند. برای استفاده از این شی در ویندوز باید فایل *wininet.dll* در ماشین مشتری نصب شده باشد. این فایل پوشه سیستم ویندوز را شناسایی میکند، اگر شما از اینترنت اکسپلورر استفاده کنید.

شی اتصال *SOAP* از شی خارجی واسط راه دور برای فراهم کردن یک واسط از برنامه های وب سرویس استفاده میکند. صرف نظر از نوع واسط استفاده شده توسط اتصال *SOAP* ، این شی اطلاعات مورد نظر برای مجموعه داده های مشتری را در برنامه آن فراهم میکند. مجموعه داده های مشتری از این اطلاعات برای برقراری ارتباط با سرویس دهنده های روی کارگزار یا هر فراخوانی دیگر استفاده میکنند.

برای اتصال به سرورهایی که از *DCOM* استفاده میکنند کلاسهای اضافه تری وجود دارد. همچنین برای اتصال به کارگزاران *TCP/IP* میتوان از برنامه نویسی سوکت استفاده نمود.

همچنین قبل از ارسال داده ها و فراخوانی متدها باید یک نمونه از سرویس در سمت مشتری ایجاد شود تا بتوان از این نمونه برای دسترسی و فراخوانی متدهای سرویس استفاده نمود. اصطلاحا به این نمونه واسط سرویس[۳] گویند. برای ساخت این نمونه از شی واسط راه دور[۴] *SOAP* استفاده میکنیم.

این شی از پیغام *HTTP* برای فراخوانی واسطهای راه دور با کمک *SOAP* استفاده میکند. از واسط راه دور میتوان برای تولید فراخوانیهای ایستای واسط موجود در یک کارگزار روی سرویس راه دور استفاده نمود. هنگامیکه یک برنامه شی واسط راه دور را تولید میکند، کلیه متد های سرویس راه دور را در حافظه داخلی ثبت میکند تا هنگام اجرای این متدها با فراخوانی یک درخواست *SOAP* و ارسال یک پیغام درخواست *HTTP* برای برنامه وب سرویس، یک متد راه دور را اجرا کند. همچنین این شی قادر است پیغام

---

[2] SOAP Connection
[3] Service Interface
[4] HTTPRIO

پاسخ *HTTP* نتیجه را برای فراهم نمودن مقادیر برگشتی از دیگر پارامترهای خروجی جداسازی نماید و یا یک استثنا را برای پردازش استثنائات در کارگزار هنگام بروز خطا تولید کند.

واسط راه دور برای تشخیص برنامه وب سرویس نیاز به دانستن محل آن دارد. دو راه برای مشخص کردن محل کارگزار وجود دارد :

۱. میتوان ارل کارگزار وب سرویس را به صفت ارل[5] شی واسط راه دور نسبت داد.

۲. چنانچه بخواهیم اتصال راه دور را در زمان اجرا و به صورت پویا از یک سند *WSDL* بدست آوریم، میتوان مقدار صفت *WSDLLocation* را تنظیم نمود. پس از تنظیم این مقدار، باید مقادیر پورت و سرویس را تنظیم نمود.

چنانچه نیاز به کارگزار پراکسی[6] باشد یا سرور نیاز به تائید اعتبار دارد، باید اطلاعات لازم را به شی اضافه نمود. باید توجه داشت که قبل از تولید جدول متدها توسط واسط، این واسط باید در رجیستری ثبت شده باشد.

```
HTTPRIO.WSDLLocation := WSURL.Text + '/wsdl/IDecisionTreeService';
Service := HTTPRIO as IDecisionTreeService;
```

۸-۴-۲. ارسال، فراخوانی متد ساخت و دریافت درخت تصمیم از وب سرویس

در این مرحله باید پایگاه داده را برای وب سرویس ارسال کنیم بطوریکه فرمت آن و معنای داده ها از بین نرود. یکی از راه حلها ارسال داده ها با فرمت *XML* است. *XML* قادر است داده ها را با حفظ نوع و فرمت آن در بستر وب جابجا کند. یکی دیگر از راههای ارسال پایگاه با استفاده از شی ضمیمه *SOAP*[7] است. با توجه به اینکه *SOAP* برپایه *HTTP* است، لذا سرعت و سازگاری در کلیه محیطها حفظ میشود. همچنین شی ضمیمه *SOAP* فایلها را با فرمت آنها کاملا میشناسد و نیاز به هیچ تبدیلی در این زمینه وجود ندارد.

ضمیمه *SOAP* شامل پیغامهای *SOAP* فراخوانی کننده متد در یک وب سرویس کاربردی است. از ضمیمه *SOAP* میتوان برای ارسال یا دریافت اضافات به عنوان یک پارامتر یا مقدار برگشتی یک متد در یک واسط احضار شده استفاده نمود. هنگامیکه یک فراخوانی متد *SOAP* نیاز به اطلاعات ضمیمه پیدا کند بصورت یک فرم چند قسمتی *MIME* در میاید، بطوریکه هر ضمیمه در یکی از قسمتهای اضافی قرار گرفته است. باید توجه کرد که برخی وب سرویس ها مانند *Net*. از فرمهای چندقسمتی *DIME* بجای *MIME* استفاده مینمایند. این نکته نیز حائز اهمیت است که از ضمیمه *SOAP* فقط به عنوان پارامتر یک متد یا مقدار برگشتی آن میتوان استفاده نمود. ضمیمه *SOAP* نمیتواند قسمتی از انواع پیچیده مانند یک صفت از کلاس باشد.

هنگام ارسال یک ضمیمه میتوان از خاصیت *SourceString* یا متدهای *SetSourceFile* یا *SetSourceStream* برای ضمیمه کردن داده ها استفاده نمود. البته باید نوع داده های ضمیمه نیز مشخص گردد.

هنگام دریافت یک ضمیمه، ضمیمه بصورت اتوماتیک در یک فایل موقت ذخیره میشود. میتوان محتویات این فایل موقت را با استفاده از متدهای *SaveToFile* یا *SaveToStream* ذخیره نمود. همچنین میتوان نوع داده ها را تشخیص داد. با استفاده از صفت سرایند میتوان اطلاعات اضافی همراه با ضمیمه را مطالعه نمود. بطور پیش فرض هنگامیکه ضمیمه *SOAP* آزاد میشود، فایل موقت شامل داده های ضمیمه دریافت شده پاک میگردد. البته با تنظیم خصوصیات ضمیمه میتوان مانع این کار شد.

برنامه های کاربردی قبل از استفاده از این کلاس، باید انرا در رجیستری کلاسهای راه دور ثبت کنند. هنگامیکه شما یک سند *WSDL* را به برنامه خود اضافه میکنید، فراخوانی متد عمل ثبت ضمیمه *SOAP* بطور اتوماتیک انجام خواهد شد.

---

[5] URL
[6] Proxy Server
[7] SOAP Attachement

اگر یکی از واسطها از ضمیمه استفاده کند فراخواننده واسط احضار شده (مشتری) مسول ایجاد نمونه ضمیمه های *SOAP* که با عنوان پارامترهای ورودی ارسال میشوند، است و همچنین آزادسازی ضمیمه *SOAP* های ایجاد شده یا برگشت داده شده خروجی نیز بر عهده آن میباشد. در برنامه های وب سرویس آزادسازی نمونه های ضمیمه ایجاد شده یک امر ضروری به شمار میاید.

کد مربوط به ایجاد یک شی ضمیمه *SOAP* که پایگاه داده انتخابی کاربر را در خود جاسازی میکند در زیر نشان داده شده است.

```
Var DB: TSOAPAttachment;
…

  DB := TSOAPAttachment.Create;
  DB.SetSourceFile(DataBase.FileName);
```

اکنون با داشتن یک نمونه از سرویس در سمت مشتری و ساخت یک ضمیمه *SOAP* از پایگاه داده میتوان متد ساخت درخت تصمیم از سرویس را فراخوانی نمود.

برای اجرای متد باید نام جدولی که قرار است درخت تصمیم از روی آن ساخته شود، نام فیلد هدف یا وابسته و لیست فیلدهای پیشگویی کننده از کاربر توسط اکتیوایکس موجود در سمت مشتری گرفته و به عنوان پارا متر به متد وب سرویس ارسال شود. کد مربوط به فراخوانی در زیر آمده است.

```
  Tree := Service.BuildDataDTree(DB, TableList.Text, GoalFieldList.Text,
AVCCheckList);
```

پیام درخواست *SOAP* مربوط به این فراخوانی در زیر آمده است.

```
--MIME_boundaryB0R9532143182121
Content-Type: text/xml
SOAPAction: "urn:DecisionTreeServiceIntf-IDecisionTreeService#BuildDataDTree"
Content-ID: <http://www.borland.com/rootpart.xml>
Content-Location: http://www.borland.com/rootpart.xml
Content-Length: 736


<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Body SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"><NS1:BuildDataD
Tree xmlns:NS1="http://tempuri.org/"><DB href="cid:CE2FD3F8-10E2-488A-871D-
F2D5304F0FAA"/><TableName
xsi:type="xsd:string">insurance</TableName><GoalFieldName
xsi:type="xsd:string">highrisk</GoalFieldName><AVCList xsi:type="SOAP-
ENC:Array" SOAP-
ENC:arrayType="xsd:string[2]"><item>age</item><item>cartype</item></AVCList><
/NS1:BuildDataDTree></SOAP-ENV:Body></SOAP-ENV:Envelope>

--MIME_boundaryB0R9532143182121
Content-ID: <CE2FD3F8-10E2-488A-871D-F2D5304F0FAA>
Content-Length: 897024
Content-Type: application/binary
Content-transfer-encoding: binary
```

پس از فراخوانی و ارسال پایگاه داده و دیگر اطلاعات مورد نیاز، وب سرویس شروع به ترسیم درخت تصمیم میکند. برای این منظور ابتدا فایل موجود در ضمیمه در کارگزار ذخیره میشود. کد مربوط به این عمل در زیر آمده است.

```
function TDecisionTreeService.BuildDataDTree(DB: TSOAPAttachment; TableName,
GoalFieldName: String; AVCList: TStringArray): TSOAPAttachment;
begin
  if not DirectoryExists(DATA_DIR) then
    CreateDir(DATA_DIR);
  DB.SaveToFile(DATA_DIR + 'Base.mdb');
```

سپس یک نمونه از شی درخت تصمیم که در بخش قبلی مفصل توضیح داده شد ساخته میشود.

```
  if DTree = nil then DTree := TDecisionTree.Create;
  DTree.EmptyTree;
  DTree.Caption := GoalFieldName;
```

در ادامه اتصالات لازم برای کار با پایگاههای داده ذخیره شده در کارگزار توسط اشیا *ADODB* ایجاد و برقرار میشود.

```
 if ADOConnection = nil then
      ADOConnection := TADOConnection.Create(DTree);
  ADOConnection.ConnectionString := 'Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=D:\Inetpub\DMServices\Data\Base.mdb;Persist Security Info=False';
  ADOConnection.LoginPrompt := False;
  ADOConnection.Open;
  if ADOQuery = nil then ADOQuery := TADOQuery.Create(DTree);
  ADOQuery.Connection := ADOConnection;


  if ADODataSet = nil then ADODataSet := TADODataSet.Create(DTree);
  ADODataSet.Connection := ADOConnection;
```

متدهای لازم برای ساخت درخت تصمیم و ترسیم آن در شی تصویر درخت تصمیم فراخوانی میشوند. این متدها در بخش قبلی کاملا تشریح شده اند.

```
  BuildDT(TableName, GoalFieldName, AVCList, 0);
  DTree.Show;
```

تصویر موجود در شی درخت تصمیم در یک فایل تصویر ذخیره میشود. سپس با ایجاد یک شی ضمیمه *SOAP* و جاسازی قایل تصویر درخت در آن، ضمیمه ایجاد شده بعنوان پاسخ متد برای مشتری ارسال میشود. مشتری میتواند این فایل را برای تحلیلهای بعدی ذخیره نماید.

```
  DTree.BkImg.Picture.SaveToFile(DATA_DIR + 'Tree.bmp');
  Result := TSOAPAttachment.Create;
  Result.SetSourceFile(DATA_DIR + 'Tree.bmp');
  //Return to Client
```

همچنین پیغام پاسخ *SOAP* بصورت زیر است.

```
--MIME_boundaryB0R9532143182121
Content-Type: text/xml
Content-ID: <http://www.borland.com/rootpart.xml>
Content-Location: http://www.borland.com/rootpart.xml
Content-Length: 554

<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"><SOAP-ENV:Body SOAP-
ENC:encodingStyle="http://schemas.xmlsoap.org/soap/envelope/"><NS1:BuildDataD
TreeResponse xmlns:NS1="urn:DecisionTreeServiceIntf-
IDecisionTreeService"><return href="cid:B90B3DCD-8BAC-4FB0-A116-
44681A36C90C"/></NS1:BuildDataDTreeResponse></SOAP-ENV:Body></SOAP-
ENV:Envelope>

--MIME_boundaryB0R9532143182121
Content-ID: <B90B3DCD-8BAC-4FB0-A116-44681A36C90C>
Content-Length: 500066
Content-Type: application/binary
Content-transfer-encoding: binary
```

۸-۴-۳. انتشار وب سرویس بر روی اینترنت

برای استفاده از وب سرویس و شناسایی آن توسط مشتریان موجود در اینترنت، هر وب سرویس باید مشخصات، نحوه دسترسی و لیست متدهای خود را در اختیار سرویس گیرندگان قرار دهد. این همان مفهوم خود تعریف بودن وب سرویس است که در فصل وب سرویس به آن مفصلا اشاره کردیم. در برنامه این کار با استفاده از شی انتشار دهنده سند *WSDL* انجام میگیرد.

این شی لیستی از سندهای *WSDL* که بطور کامل یک وب سرویس را شرح میدهند، تولید میکند. با اضافه کردن این شی به پیمانه وب از یک وب سرویس، میتوان سندهای تشریح کننده *WSDL* را برای یک وب سرویس تولید نمود. این سندها طرز کار و فراخوانی متدهای وب سرویس را تشریح میکنند. این امر باعث میشود که هر مشتری که بخواهد، بتواند از وب سرویس استفاده نماید.

سند تولید شده توسط این شی، کلیه انواع، اشیا و کلاسهای راه دور ثبت شده را بطور دقیق توصیف میکند. همچنین کلیه متدها و طرز فراخوانی آنها را برای مشتریان وب سرویس تشریح میکند. این شی بصورت اتوماتیک خود را در رجیستری ثبت میکند، یعنی برای استفاده از آن نیاز به هیچ عمل اضافی تحت وب نیست.

کد مربوط به استفاده از این شی و تولید اسناد *WSDL* لازم در زیر آمده است.

```
procedure TWebModule1.WebModule1DefaultHandlerAction(Sender: TObject;
  Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);
begin
  WSDLHTMLPublish1.ServiceInfo(Sender, Request, Response, Handled);
end;
```

۸-۵. چند نکته کلیدی در مورد برنامه

سرعت داده کاوی با وب سرویسها : معمولا داده کاوی امری وقت گیر و زمان بر است که این نکته باید مورد توجه قرار بگیرد. چون قفل شدن برنامه مشتری و منتظر بودن برای رسیدن جواب شاید زیاد مطلوب نباشد.

بدون حافظه بودن[8] وب سرویسها : وب سرویسها اصطلاحا ناپایدار یا بدون حافظه اند. تنظیم برقراری یک جلسه مشترک از زمان ارسال داده ها تا گرفتن نتیجه داده کاوی ضروری به نظر میرسد. خصوصا این مسئله زمانی پیش میاید که چند کاربر همزمان بخواهند از سرویس استفاده کنند. آنگاه تنظیم یک جلسه منحصر به فرد میان هر مشتری با کارگزار از اهمیت خاصی برخوردار است. چون سرویس ذاتا اطلاعات مربوط به مشتری را نگهداری نمیکند و ممکن است در پاسخدهی دچار تداخل شود یا اصطلاحا طرف رابطه را گم کند.

۸-۶. خلاصه

در این فصل سعی شده است پیاده سازی عملی برنامه که مربوط به ساخت درخت تصمیم با استفاده از سرویسهای وب است، به طور مفصل شرح داده شود. برنامه مزبور با استفاده از بورلند دلفی نوشته شده است و کدهای آن به زبان پاسکال شی گرا میباشد.

در بخش اول ساختمان داده مربوط به درخت تصمیم تشریح و الگوریتم مورد استفاده در آن برای ساخت درخت تصمیم بررسی گردید. این الگوریتم با استفاده از زبان پرس و جوی استاندارد نوشته شده است. دلایل این انتخاب نیز ذکر گردید.

در بخشهای بعدی نحوه ارتباط برنامه مشتری با سرویس از طریق وب و پروتکل *SOAP* مورد کنکاش قرار گرفت. اشیاء ضمیمه *SOAP* برای ارسال پایگاه داده، شی اتصال *SOAP* برای برقراری یک ارتباط کامل با وب سرویس، شی واسط راه دور برای ساخت یک نمونه از سرویس در برنامه مشتری و دسترسی به آن کاملا تشریح گردید.

---

[8] Stateless

در انتها نیز مسایل مربوط به انتشار وب سرویس در وب و نکات کلیدی مورد توجه در پیاده سازی ذکر گردید. امید است مورد توجه قرار گرفته باشد. ان شا ا...

نتیجه گیری و پیشنهادات

Conclusion and Offers

۹-۱. نتیجه گیری

آنچه این پایان نامه بر محور آن شکل گرفته است پاسخ به این پرسش است که آیا میتوان ابزارهای داده کاوی را بطور موثری توسط سرویسهای وب پیاده سازی نمود بطوریکه افراد و سازمانها بتوانند از آن برای تجارت استفاده کنند ؟ و هدف غایی آن پیاده سازی ابزارهای داده کاوی با استفاده از سرویسهای وب با طراحی واسطهای ساده و کارامد برای استفاده کاربران اینترنت میباشد.

اهمیت موضوع در آنجاست که امروزه شرکت‌ها، سازمان‌ها و دانشگاه‌ها غرق در انبوه داده‌ها و اطلاعاتی هستند که استفاده از آنها در بیشتر موارد محدود به انجام کارهای جاری می‌باشد و هنوز از داده‌ها در تصمیم‌گیری استراتژیک استفاده نمی‌شود. داده‌کاوی که استفاده از آن روز به روز توسعه می‌یابد می‌تواند به استفاده از اطلاعات موجود در مؤسسات در زمینه‌های تصمیم‌گیری استراتژیک منجر شود. اما علیرغم رشد قابل ملاحظه ابزارهای داده کاوی، پیچیده بودن، تخصصی بودن و عدم حضور آنها در وب به عنوان مهمترین منبع براورده کردن نیاز امروز سازمانها به صورت یک سرویس ساده باعث شده است که این مقوله مهجور واقع شود.

برای رسیدن به این هدف ابتدا مبانی کار و اصول علمی آن تشریح شد. به همین منظور داده کاوی را تعریف و مورد بررسی کامل و مفصل قرار دادیم . به طور اجمال باید گفت، هدف از داده کاوی ایجاد مدل هایی برای تصمیم گیری است. این مدلها رفتارهای آینده را براساس تحلیلهای گذشته پیش بینی می کنند. داده کاوی به عنوان اهرمی برای آماده سازی داده ها، بهترین موقعیت را برای به دست آوردن برتریهای رقابتی ایجاد می کند.

اغلب تجارت ها به تصمیم گیریهای استراتژیک و یا اتخاذ خط مشی های جدید برای خدمت رسانی بهتر به مشتریان نیاز دارند. به عنوان مثال فروشگاهها آرایش مغازه خود را برای ایجاد میل بیشتر به خرید مجدداً طراحی می کنند و یا خطوط هواپیمایی تسهیلات خاصی را برای مشتریان جهت پروازهای مکرر آنها در نظر می گیرند. این دو مثال به داده هایی در مورد رفتار مصرفی گذشته مشتریان برای تعیین الگوهایی به وسیله داده کاوی، نیاز دارد. براساس این الگوها تصمیمات لازم اتخاذ می شود. در واقع ابزار داده کاوی، داده را می گیرد و یک تصویر از واقعیت به شکل مدل می سازد، این مدل روابط موجود در داده ها را شرح میدهد.

از نظر فرایندی فعالیتهای داده کاوی به سه طبقه بندی عمومی تقسیم می شوند:

اکتشاف : فرایند جستجو در یک بانک داده برای یافتن الگوهای پنهان، بدون داشتن یک فرضیه از پیش تعیین شده درباره اینکه این الگو ممکن است چه باشد. مانند تحلیلهایی که برحسب کالاهای خریداری شده صورت می گیرد، اینگونه تحلیلهای سبدی نشانگر مواردیست که مشتری تمایل به خرید آنها دارند. این اطلاعات می تواند به بهبود موجودی، استراتژی طراحی، آرایش فروشگاه و تبلیغات منجر گردد.

مدل پیش بینی : فرایندی که الگوهای کشف شده از بانک داده را می گیرد و آنها را برای پیش بینی آینده به کار می برد. مانند پیش بینی فروش در خرده فروشی، الگوهای کشف شده برای فروش به آنها کمک میکند تا تصمیماتی را در رابطه با موجودی اتخاذ کنند.

تحلیلهای داده ای : به فرایند به کارگیری الگوهای استخراج شده برای یافتن عوامل داده ای نامعقول و متناقض مربوط می شود. مانند شناسایی و تشخیص کلاهبرداری در موسسات مالی. کلاهبرداری به میزان زیادی پرهزینه و زیان آور است، بانکها می توانند با تحلیل دادوستدهای جعلی گذشته الگوهایی را برای تشخیص و کشف کلاهبرداری به دست آورند.

دو ابزار مهم داده کاوی درخت تصمیم و سیستم پشتیبانی از تصمیم هستند. از آنرو که این دو ابزار در پیاده سازی عملی پایان نامه مورد استفاده قرار گرفته اند، توضیح مفصلی از آنها و الگوریتمهای موجود برای پیاده سازی آنها را در پایان نامه آورده ایم. بطور خلاصه باید گفت درخت تصمیم پیشگویی، پیش‌بینی و دریافت مقدار یک خصیصه بر اساس خصیصه‌های دیگر می‌باشد. در این درخت هر گره میانی یک فیلد پیشگویی کننده است که بر اساس آن داده ها تقسیم میشوند. برچسب یالهای خروجی نیز معیار این تقسیم بندی را نشان میدهند. مقادیر برگها نیز مقدار فیلد هدف در آن مسیر را مشخص میکند. همچنین ابزارپشتیبانی از تصمیم ابزاریست برای تحلیل

داده ها ، یافتن ارتباط بین داده ها، تولید گزارش های کارآمد، دسترسی منعطف بـه داده هـا، راهکـار هـای نمـایش اطلاعـات در انـواع ممکن، چاپ اطلاعات،انتقال داده ها به صفحات گسترده و...

برای پیاده سازی این ابزار مفید در اینترنت وب سرویسها را به عنوان یک بستر مناسب تعریف و تـشریح کـردیم. اگـر بخـواهیم نگـاهی اجمالی به سرویسهای وب بیندازیم، باید بگوییم یک سرویس وب، سیستم نرم افزاری شناسایی شده توسط یک *URL* است کـه واسـط های عمومی و محدودیت های آن توسط *XML* تعریف و توصیف شده است. تعریف آن می تواند توسط سایر سیستم های نرم افـزاری کشف گردد . سپس امکان دارد این سیستمها با وب سرویس در یک روش تعیین شده توسط تعریفش، با اسـتفاده از پیامهـای مبتنـی بـر *XML* که توسط پروتکل های اینترنتی رسانده شده اند، تعامل برقرار نمایند. مهمترین دلیل ما برای پیاده سازی ابزارهای مفید داده کاوی از طریق سرویسهای وب اینست که به اعتقاد همه سرویسهای وب الگوی غالب سالهای آتی خواهد بود.

در بخش آخر هم سعی کردیم همه آنچه گفته شد، را بـه طریـق عملـی پیـاده سـازی کنـیم و مـشکلات آن را بیـان و بـه ارایـه راه حـل بپردازیم. به همین منظور در سه بخش کلی به بررسی حوانب پیاده سازی ابزارهای داده کاوی با وب سرویسها پرداختیم. مـسایل مربـوط به پیاده سازی سمت مشتری (نیاز به واسط ساده و کارامد، پایگاههای داده مختلف، جمع آوری ورودیهای لازم برای داده کـاوی و ...) مورد بررسی قرار گرفت که از میان آنها اکتیوایکس به عنوان یک تکنولوژی خوب به دلیل قدرت کامـل برنامـه نویـسی، ارایـه کنتـرل کامل، سازگاری با پایگاههای داده، سازگاری با وب، امنیت دیجیتالی و ... پیشنهاد و تشریح گردید.

در ادامه مشکلات سمت کارگزار از قبیل حجم داده ها، سرعت، هزینه پیاده سازی، استانداردسـازی و ... بررسـی و در ایـن میـان نیـز بـا بررسی تکنولوژیهای موجود *CGI* بدلیل استقلال برنامه نویسی، استقلال پلاتفرم، سازگاری با محیطهای مختلـف، سـادگی و ... پیـشنهاد گردید. همچنین *SOAP* به عنوان بهترین استاندارد موجود در زمینه کار با وب سرویس برای تبادل اطلاعات حجـیم و حفـظ سـاختاری آن تشریح و توسعه های پیشرفته تر آن معرفی گردید.

سعی کردیم در فصل نهایی کد پیاده سازی شده با بورلند دلفی مورد بررسی قرار گرفته و جزییات آن برای مشخص شدن هر چه بیـشتر جوانب کار تشریح گردد. ساختمان داده های مورد استفاده و ساختارهای مرتبط کاملا توضیح داده شـد. همچنـین الگوریتمهـای مـورد استفاده، متدهای لازم برای فراخوانی، ارسال و دریافت داده ها برای ساخت یک درخت تصمیم با ذکر یک مثـال عملـی بـا توضـیحات مفصل در اختیار خوانندگان قرار گرفت.

درنهایت هدف نهایی و غایی این پایان نامه اینست که سازمانها و تجارتهای کوچـک بتواننـد در تـصمیم گیریهـای خـود از داده کـاوی بهره گیرند و تجارت الکترونیک که شاید تنها تجارت بزرگ آینده خواهد بود، وارد عرصه نوینی از ایده هـای نـو گـردد. بایـد گفـت طیف وسیعی از کاربردها برای این تکنولوژی قابل تصور است که به عنوان نمونه میتوان به مراکـز پژوهـشی، شـرکتهای فعـال در زمینـه وب و پایگاه داده، تحلیلگران و مدیران سازمانها، تجارت (تبلیغات، مدیریت ارتبـاط بـا مـشتری،...)، وب (موتورهـای جـستجو،...) و ... اشاره کرد.

۹-۲. پیشنهادات

پیاده سازی عملی پایان نامه در سایت *http://www.itlecture.com/dm_ws/* قابل مشاهده اسـت. همچنـین فهرسـت کـاملی از مراجـع و منابع مورد نیاز در زمینه های داده کاوی، سرویسهای وب و تکنولوژیهای مرتبط در انتهای پایان نامه آمده است که میتواند مورد استفاده دانش پژوهان علاقه مند قرار گیرد. در ضمیمه پایان نامه نیز سورس کد کامل برنامه آورده شده است.

پیشنهاد میشود که پیاده سازی کاملی از کار با در نظر گرفتن محیطهای توسعه چند منظوری وب و توجه به مـسایل مربـوط بـه همزمـانی کاربران، ترافیک شبکه، قدرت کارگزاران اینترنتی مختلف و ... دریک سایت کامل در یک بازه زمانی نسبتا طولانی انجام گیـرد تـا بـا ثبت وقایع و مشکلات گزارش شده بتوان محدودیتهای کار را سنجید و به یک راه حل کاملا عملی از قضیه رسید.

همچنین میتوان با ذخیره نتایج حاصل از مراجعه کاربران پایگاه کاملی از نتایج کاربران مختلف در زمینه های مختلف گردآوری کرد که این پایگاه خود میتواند زمینه داده کاوی های بیشتر و کسب نتایج غیر منتظره باشد. حتی میتواند زمینه تجاری و کسب درامد را نیز فراهم نماید.

امیدوارم این کوشش هرچند ناچیز در این زمینه نو و ارایه ایده های نو راهگشای جویندگان طریق علم و معرفت باشد. ان شاء ا...

بر خود میدانم در پایان مطلب یکبار دیگر از زحمات بیدریغ استادان گرانقدرم جناب آقای دکتر قاسم آقایی، دکتر نقش نیلچی و پیشنهادات گرانقدر دکتر ابوالحسنی و همچنین همسر و همکار عزیزم خانم مهندس شمس و در نهایت پدر و مادرم که پشتیبان من در همه مراحل بوده اند، تشکر و قدردانی نمایم.

عبدالرضا رسولی کناری

تابستان ۸۵

مراجع و منابع

References

[W3C 2004] http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

[W3C 2004] http://www.w3.org/2004/02/wsa/

[W3C SOAP] http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[DAML] http://www.daml.org/services/swsf

[OSASIS] http://www.oasis-open.org/committees

[UDDI] http://www.uddi.org/

[DMGroup] http://www.dmg.org

[SOAP] http://www.w3.org/TR/2003/REC-soap12-part0-20030624/

[SUN] Java.sun.com

[JDM] http://jcp.org/aboutJava/communityprocess/edr/jsr247/index.html

[JDM 2006] http://jcp.org/aboutJava/communityprocess/mrel/jsr073/

[WEKA] Khoussainov, R., Zuo, X., Kushmerick, N.: Grid-enabled Weka: A Toolkit for Machine Learning on the Grid. ERCIM News, n. 59 (2004).

[Shaikh] Shaikh Ali, A., Rana, O. F., Taylor, I. J.:Web Services Composition for Distributed Data Mining. Workshop on Web and Grid Services for Scientific Data Analysis (2005 - to appear).

[Prez] Prez, M. S., Sanchez, A, Herrero, P, Robles, V., Pea. J. M.: Adapting the Weka Data Mining Toolkit to a Grid based environment. 3rd Atlantic Web Intelligence Conference (2005 - to appear).

[Agrawal 2005] Agrawal, Shim, Developing tightly-coupled Data Mining Applications on a Relational Database System, 2005.

[Berger 2004] Gideon Berger, Knowledge Discovery in Databases for Introsion Disease Classification and Beyoud, 2004.

[Berkhin2002] P. Berkhin, Survay of clustering Data Mining Techniques, Accrue Software, CA. 2002.

[Breiman 2003] Leo Breiman. Bias, Variance and Arcing Classifiers, 2003.

[C2001] C.Clifton, Security Issues in Data Mining, CS590 M Fall 2001.

[DK2000] D.Kondo, Data Mining and Data-Intensive Computing , CSE225, 2000.

[Fayyad 2005] Usama Fayyad, Shapiro, Smyth, Knowledge discovery and Data Mining: Towards a unifying framework, 2005.

[Gama] Gama, Bradzil, Characterization of Classification Algorithms.

[Gams 2006] Matjaz Gams, Nada Lavarc, Review Of Five Emperical Learning Systems Whitin a Proposal Schemata,2006 , EWSL06.

[Han 2000] Jiawey Han, Fu, Wang, Koperski, Zaiane, DMALm A Data Mining Query Language For Relational Database, 2000.

[HK 2000ch1] J.Han, M.Kamber, Date Mining: Goncepts and Techinqeus, JimGray, Series Editor Morgan Kaufmann Publishers, Augest 2000.

[Hogl2001] Hogl, Stoyan, Muller, The Knowledge Discovery Assistant: Making Data Mining Available for Business Users, 2001.

[Holsheimer99] Holsheimer, siebes, Data Mining: The Search for Knowledge in Databases, 1994.

[John97] G. H. John, Enhancements to the Data Mining Process, 1997.

[Ordonez2000] Carlos Ordonez, Paul Cereghini, SQLEM: Fast Clustering in SQL using the EM Algorithm, 2000.

[Rajamani98] Karthik Rajamani, Alan Cox, Efficient Mining For Association Rules With Relational Database Systems, 1998.

[RK 98] R. Rastogi, K. Shim, Public: A Decision Tree Classifier that Integrates Building and Pruning, Bell Laboratories. Murray Hill, NJ 07974, 1998.

[Sarawagi98] Sarawagi, Thomas, Agrawal, Integrating Association Rule Mining with Relational Database Systems: Alternative and Implecations.

[SHK98] A.Srivastave, E.Han, V. Kumar, Parallel Formulation of Decisio-Tree Classification Algorithms, 1998 .

[Thomas98] Shiby Thomas, Sunita Sarawagi, MiningGeneralaized Association Rules and Sequential Patterns Using SQL Queries, Ammerican Association for Artificial Intelligence, 1998.

[Alonso 2004] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. Web Services - Concepts, Architectures and Applications. Springer-Verlag, 2004.

[MS COM] Microsoft Corporation. The Component Object Model specification, October 1995. Draft Version 0.9.

[OMG CORBA] Object Management Group. The Common Object Request Broker Architecture (CORBA core specification, December 2002. Version 3.0.

[Turner 2003] M. Turner, D. Budgen, and P. Brereton. Turning software into a service. IEEE Computer, 36(10), October 2003.

[UDDI 2002] T. Beliwood et a!. Universal Description, Discovery and Integration specification (UDDI) 3.0. Online: http://uddi.org/pubs/uddi-v3.00- published-2002071 9.htm.

[WSDL 1.2] R. Chennai et al. Web Services Description Language (WSDL) 1.2. Online: http://www.w3.org/TR/wsdl/.

[SOAP 1.1] D. Box et . Simple Object Access Protocol (SOAP) 1.1. Online: http://www.w3.org/TR/SOAP/, 2001.

[Ankoekar2001] A. Ankolekar et al. DAML-S: Semantic markup for Web services. Proceedings of the International Semantic Web Workshop, 2001.

[Paolucci 2002] M. Paolucci, T. Kawamura, T. R. Payne, and K. Sycara. Semantic matching of web services capabilities. In First International Semantic Web Conference, Sardinia, Italy, June 2002.

[Sollazzo 2002] T. Sollazzo, S. Handschuh, S. Staab, and M. Frank. Semantic Web service architecture — evolving Web service standards toward the Semantic Web. In Proceedings of the 15th International FLAIRS Conference, Florida, USA, 2002.

[Shriv 2000] S. Shrivastava, L. Bellissard, D. Fliot, et a!. A workflow and agent based platform by service provisioning. In Proceedings 01 the 4th lithe/UMG International Enterprise Distributed Object Computing Conference (ED OC 2000), Makuhari, Japan, September 2000. IEEE Computer Society Press.

[ebXML] M. Siddalingaiah. Overview of ebXML. Online: http:// dcb.sun.com/ practices/ webservices/, August 2001.

[ebXML 1.4] ebXML Technical Architecture Team. ebXML technical architecture specification vl.O,4. online: http://www.ebxml.org, February 2001,

[Agrawal] R. Agrawal, A. Gupta, S. Sarawagi. Modeling Multidimensional Databases. ICDE 1997

[Wiley] L. English: Improving Data Warehouse and Business Information Quality. Wiley, New York et al.1999.

[WU] Wu, M-C., A.P. Buchmann. “Research Issues in Data Warehousing.” Submitted for publication.

[Shukla] A. Shukla, P.M. Deshpande, J.F. Naughton, and K. Ramasamy. Storage estimation for multidimensional aggregates in the presence of hierarchies. In Proc. of the 22nd Int’lConference on Very Large Databases, pages 522–531, Mumbai (Bombay), India, September 1996.

[Jarke] M. Jarke, M. Jeusfeld, C. Quix, P. Vassiliadis: Architecture and Quality in Data Warehouses: An Extended Repository Approach, in Information Systems, 24(1999), No. 3, p. 229-253.

[Chawathe] S. Chawathe, H. Garcia-Molina, J. Hammer,K. Ireland, Y.Papakonstantinou, J.Ullman, and J. Widom. The Tsimmis project: Integration of heterogeneous information sources. In Proceedings of 100th Anniversary Meeting of the Information Pro-cessing Society of Japan, pages 7{18, Tokyo, Japan,October 1994.

[Wiener] J. L. Wiener, H. Gupta, W. J. Labio, Y. Zhuge, H. Garcia-Molina, and J. Widom. A System Prototype for Warehouse View Maintenance. In Montreal, Canada, pp. 26-33 (1996).

[Hull] R. Hull, and G. Zhou. A Framework for supporting data integration using the materialized and virtual approaches.In ACM Press, pp. 481-492, Montreal (1996).

[Cabibbo] L. Cabibbo, and R. Torlone. A logical approach to multidimensional databases. In, Avignon, France, Springer Verlag, pp. 183-197 (1998).

[Bosworth] J. Gray, A. Bosworth, A. Layman, H. Pirahesh. Data cube: a relational aggregation operator generaizing group-by, cross-tab, and subtotals. New Orleans, IEEE Computer Society, pp.152-159 (1996).

[Wiederhold] G. Wiederhold. Mediators in the architecture of future information systems. IEEE Computer25(3):38-49 (1992).

[Curley 2004] K. McCurley and A. Tomkins. Mining and knowledge discovery from the Web. In *7th International Symposium on Parallel Architectures, Algorithms and Networks*, Hong Kong, 2004

[Zhuge 2004] H. Zhuge et al. An Automatic Semantic Relationships Discovery Approach. The 13th International World Wide Web Conference (WWW2004), New York, USA, May 2004,

[DM 2003] “THE HAND BOOK OF DATA MINING”, ARIZONA STATE UNIVERSITY, 2003.

[CHRIS 2002] CHRIS RYGIELSKI, “DATA MINING TECHNIQUES FOR CUSTOMER RELATIONSHIP MANAGEMENT ,”TECHNOLOGY IN SOCIETY, 2002.

[FREEMAN] FREEMAN M., “THE CUSTOMER LIFECYCLES, INTELLIGENT ENTERPRISE, 1999.

[HILL] HILL L., “CRM: EASIER SAID THAN DONE ,”INTELLIGENT ENTERPRISE, 1999.

[IDC & CAP] IDC & CAP GEMINI, “FOUR ELEMENTS OF CUSTOMER RELATIONSHIP MANAGEMENT”, CAP GEMINI WHITE PAPER.

[DMREVIEW] INFORMATION DISCOVERY INC., “A CHARACTERIZATION OF DATA MINING TECHNOLOGIES AND PROCESSES ,” HTTP://WWW.DMREVIEW.COM/PORTAL

Abstract

To be sure that Data Mining as new idea plays important role in data analysis and great decision of organizations, companies and people. But complex implementation of providers and need of special knowledge for using it, cause that this tools became unusable. Such as most peoples didn't know what is data mining. In attention to growth of E-commerce, offering this tool on the web, can induct the business on new arena. Seems web services are appropriate for executing data mining tools on the web. This thesis attempt implement data mining tools by web service to users simply use it.

AZAD ISLAMIC UNIVERSITY
NAJAF ABAD BRANCH
POST GRADGUATE FACULTY – COMPUTER SCIENCE

Thesis for grasping Master of Science (M.Sc.) degree
SOFTWARE

Subject

Data Mining with Web Service using Web Semantic Algorithm

Advisor
Dr. Naser Ghasem Aghayi

Consulting Advisor
Dr. Ahmad Reza Naghsh Nilchi

Written by
Reza Rasouli

Summer 84