A Web Mining Platform for Enhancing Knowledge Management on the Web

KOK-LEONG ONG WEE-KEONG NG EE-PENG LIM

Center for Advanced Information Systems, Nanyang Technological University
Nanyang Avenue, N4-B3C-14, Singapore 639798

awkng@ntu.edu.sg

Abstract

The ubiquity of the Web and content management tools has overcome several limitations of manual knowledge acquisition. Today, huge amount of expertise knowledge are readily available online. However, content creation tools concentrate on the publishing of information without considering the need for structural organization. As a result, the flat distribution of knowledge makes consuming these materials difficult. Web mining is believed to be one of the solutions to manage these materials and to uncover other hidden insights. In this paper, we present the design of a Web mining platform that attempts to bridge the differences between the Web, the data mining algorithms and the constellation of Web technologies.

Keywords: Architecture, Web Mining, Knowledge Management

1 Introduction

The traditional view of knowledge management has focused mainly on the acquisition of knowledge from human beings. For example, interviews are conducted between the knowledge analyst and the domain specialist to recorded knowledge in some representation suitable for processing by the knowledge management system. In the recent years, the ubiquity of the Web and the maturity of content management tools have brought about changes to the way knowledge are made available to the public. Domain experts can now express their wealth of experiences with the help of publishing tools. As a result, it is easy to find FAQs, knowledge-bases, support sites and discussion boards on the Web. For example, the *Microsoft Developer Network* (MSDN) provides an avenue for developers to share tools, source codes and discuss related programming problems. Such a site creates value by providing mechanisms for knowledge distribution and consumption. More importantly, this model of knowledge management is more effective than the traditional process. First, knowledge can be recorded without the presence of a knowledge analyst. Content management tools are able to guide the entry of knowledge into a form that can be queried and searched. In MSDN, information are indexed and categorized to make the search and the use of materials contributed by others easy. Second, the time invariant and global reach for such information is made possible with the Web as the media for storing and delivering knowledge to a large number of audiences not possible in the past.

However, knowledge expertise on the Web has problems of its own. With a flat distribution structure, the Web makes searching of information difficult. Using search engines, search queries are difficult to formulate and search results are often inaccurate. At the same time, managing the knowledge from different experts in the same domain is also difficult and time consuming. Data mining has been touted by the industry as a possible solution to the problems of the Web [2]. Using data mining on Web data, the efficiency and accuracy of search engines can be improved [3, 7]. Knowledge contained in Web documents can be organized hierarchically. Correlated documents can be identified and hence, improve the Web as a better repository of knowledge. More importantly, the wealth of information on the Web may be the seed to create knowledge out of knowledge automatically through algorithmic methods. The fusion of knowledge management and data mining has thus become an interesting subject of study. But the Web contains issues of its own that made data mining difficult to realize its full potential. This includes the following.

Web data is semi-structured at best – HTML pages, Web logs, multimedia streams etc., are all unstructured data that many algorithms are unable to mine directly. This is because most algorithms assume that data is available in some transactional form in the database or a text file. Mining Web data using existing algorithms therefore requires data transformation. Although the KDD process includes data transformation, the methodology for transforming Web data differs from transforming database tuples.

Web data is distributed – Web data usually exists across multiple machines and is located on different physical machines. Outdated copies of the same document may also reside on the proxy servers making the direct mining of Web documents a non-trivial affair. Data mining algorithms usually assume data is available at some central location. Therefore, there is a need to address the integration of information from multiple Web servers and to ensure that the documents used for mining are up-to-date.

Web data is owned by everyone – Traditionally, the domain experts are only concerned with mining "their own data". Every single piece of information belongs to some databases that are housed within the premises of the organization. The Web however is the contribution from the "public". As such, everyone owns some part of the data on the Web. If the global mining of data is to be done, the data gathering process will be more complex than the warehousing of corporate data.

Web data has privacy concerns – As users of e-Commerce gain awareness about the privacy of their data, the ability to perform data mining without "knowing" the exact details of each individual user becomes an important concern. To date, there is very little literature to address this issue. Hence, methodologies for mining e-Commerce data without violating privacy policies are an important issue.

Web data is online and interactive – Traditionally, data mining has been an offline and batch processing task. This is different on the Web where Web pages change continuously. This mismatch results in the need for online algorithms capable of processing data incrementally. Some studies on modifying algorithms to their online equivalent are already available. However, architectures supporting incremental additions and deletions in the database are missing.

While the traditional knowledge discovery process in databases (KDD) defines a general process to transform knowledge into the desired form, realizing such architecture can be time consuming, difficult and inefficient. Instead, we believe that by "loosening" the definition of the KDD process, existing architectures that are "experts in their own rights" can be used. Motivated by the above, we construct our platform using two earlier built systems. Using XML as the fabric for connecting the components, the architecture is open and flexible to support the construction of more complex knowledge systems. In the next section, we begin a survey of existing

```
<?xml version="1.0" ?> <PMML version="1.1">
    <Header copyright="www.dmg.org" description="sample model for association rules"/>
    <DataDictionary numberOfFields="1" >
        <DataField name="item" optype="categorical" />
    </DataDictionary>

    <AssociationModel>
        <AssocInputStats numberOfTransactions="4" numberOfItems="3"
                        minimumSupport="0.6"     minimumConfidence="0.5"
                        numberOfItemsets="3"     numberOfRules="2"/>

        <!-- We have three items in our input data -->
        <AssocItem id="1" value="Cracker" />
        <AssocItem id="2" value="Coke" />
        <AssocItem id="3" value="Water" />

        <!-- and two frequent itemsets with a single item -->
        <AssocItemset id="1" support="1.0" numberOfItems="1">
            <AssocItemRef itemRef="1" />
        </AssocItemset>
        <AssocItemset id="2" support="1.0" numberOfItems="1">
            <AssocItemRef itemRef="3" />
        </AssocItemset>

        <!-- and one frequent itemset with two items. -->
        <AssocItemset id="3" support="1.0" numberOfItems="2">
            <AssocItemRef itemRef="1" /><AssocItemRef itemRef="3" />
        </AssocItemset>

        <!-- Two rules satisfy the requirements -->
        <AssocRule support="1.0" confidence="1.0" antecedent="1" consequent="2" />
        <AssocRule support="1.0" confidence="1.0" antecedent="2" consequent="1" />
    </AssociationModel>
</PMML>
```

Figure 1: Using PMML to model association rules.

efforts in data mining standards by the industry. This will provide the reader with a general understanding on the state of data mining architectures in the industry. We than introduce the **Wiccap** and **Whoweda** systems, two existing research prototypes that we reused to complete the KDD platform. The **Wiccap** system is an information extraction tool for the Web while **Whoweda** is a warehouse that harvest information from the Web with support for a set of query operators. Section 4 introduces the architecture for Web platform mining and Section 5 discusses an application example. Finally, we conclude our discussion in Section6.

2 Current Mining Scenario

The current state of the data mining industry can be described as similar to the database industry in its infancy stage. Although the techniques already exist in numbers, the use of the technologies and the implementation remain fragmented among vendors. As a result, products are standalones, have unique user interfaces, and take on different KDD approaches. If the database industry is an indicator of where we should be heading, then efforts in addressing standardization by the industry are needed. Recently, we are seeing standardization on data mining practices.

For example, the Predictive Model Markup Language (PMML[1]) is one such effort. Proposed by Grossman et al. [1], at the National Center for Data Mining, PMML is an interchange format for data mining results among applications. Using XML, PMML inherits the features of the language, making it natural for the Web. Today, PMML has already gained support from the research community including ACM SIGKDD, `XML.org` and the University of Illinois at Chicago. Together with industry players including Microsoft, IBM, Oracle etc., PMML is beginning to

[1] *http://www.dmg.org*

```
CREATE MINING MODEL [Age Prediction] (
    [Customer ID]          LONG    KEY,
    [Gender]               TEXT    DISCRETE,
    [Age]                  DOUBLE  DISCRETIZED()   PREDICT,
    [Product Purchases]    TABLE (
        [Product Name]     TEXT    KEY,
        [Quantity]         DOUBLE  NORMAL CONTINUOUS,
        [Product Type]     TEXT    DISCRETE RELATED TO [Product Name]
    )
) USING [Decision Trees]
```

Figure 2: Using SQL-like constructs in OLE DB to create a decision tree.

establish its position. PMML supports seven models including association rules, polynomial regression, classification trees, center based clusters, general regression, neural networks, and distribution based clusters.

The motivation behind PMML was that existing applications store their models in proprietary formats. As a result, these models become dependent on the application, system and architecture. This limits the sharing of models among applications. Recognizing this problem, PMML was introduced as an attempt to push forward an open standard for modeling data mining results. This model is independent of the application, platform or the process (i.e., the data structure or the algorithm). Hence, PMML makes it possible to easily exchange data mining models among applications and across the Internet. Figure 1 shows a set of association rules represented using PMML for exportation to another application. Since PMML defines a standard for model exchange, it is completely independent of the implementation itself, making the sharing of models as easy as sharing HTML documents. An upcoming version of PMML is expected to arrive with support for additional models and other standards like the JSR 073[2] and OLE DB[3].

The JSR 073 is a standardization in process for the Java programming language. Its objective is to create a consistent application programming interface for Java developers to accessing data mining related technologies such as JDBC or PMML. The notion of a consistent API for Java developers is extremely attractive and is similar to our vision of a consistent platform for engineering Web mining research. The JSR 073 is in an infancy stage, and more work needs to be done before we are able to make further assessments of the technology. At the other end of the spectrum is the approach taken by Microsoft to extend SQL with data mining constructs. It allows a user to treat models as a special database table and hence, issue SQL-like statements to perform data mining tasks. Unlike the JSR 073, OLE DB is a complete implementation with native support to export and consume PMML strings. Figure 2 shows the SQL-like statement to create a decision tree using the technology.

Although both the Java Data Mining API and the OLE DB standards are attractive options, one is only available to the Java community and the other is strictly a Microsoft technology. The OMG's Common Warehouse Meta-data (CWM[4]) for data mining is an open technology that is independent of the vendor. The initiative for CWM came about in June 2000 with the objective of becoming the industry standard for describing meta-information across enterprise data. CWM is developed by independent standards body and a number of industry leaders including IBM, Unisys and Oracle. CWM reflects the real world allowing data to be shared without compromising uniqueness and facilitates a single point of reference throughout the enterprise. With this standard, it is believed that software vendors will have reduced development time to bring products to the market quickly, and the freedom of choice with reduced complexity for customers will also improve business intelligence as a whole. Like the JSR 073, CWM is relatively new with changes expected in its specifications.

[2] *The Java Data Mining Application Programming Interface, http://jcp.org/jsr/detail/73.jsp*

[3] *Microsoft Universal Data Access, http://www.microsoft.com/oledb*

[4] *Object Management Group (Common Warehouse Meta-Model), http://www.omg.org/cwm*

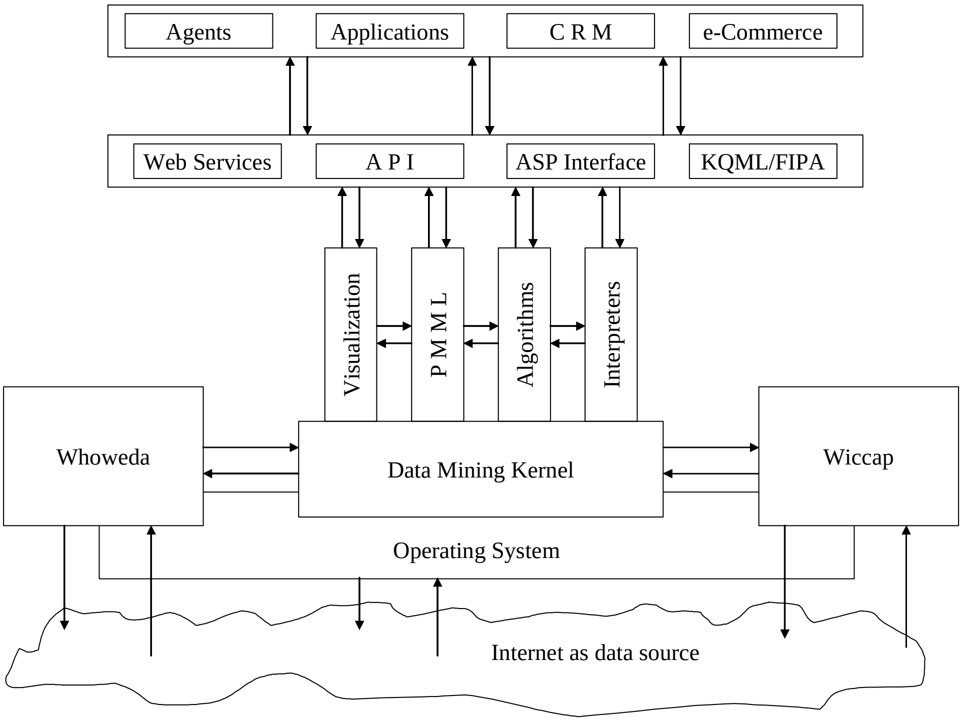

Figure 3: The general Web architecture.

The purpose of surveying these standards is to identify the directions taken and to consider a design that can inter-operate with the industry modules. We shall briefly discuss the existing systems used in the construction of the Web mining platform in the next section.

3 Existing Architectures

Figure 3 shows the high level architecture of our proposed platform to support applications with needs to mine Web data. The Internet represents the knowledge which we believe, when combined with other information such as Web logs, multimedia data, and transaction records, generates interesting insights. The kernel presents the environment for running data mining services. The arrows indicate the inter-API calls and data transfers between components. We begin by discussing the existing architectures used. The reader should note that while our discussion uses systems from our own work, the communication between components is achieved by XML. This means that it is possible to replace them with other components by importing and exporting the necessary information via XML.

3.1 Whoweda

The **Whoweda** [5, 6, 8, 9] system is a Web warehouse that materializes and manages useful information from the Web to support strategic decision making. The system is a meta-data repository of useful, relevant Web information available for querying and analysis. As relevant information becomes available on the Web, they are coupled from various sources, translated into a common Web data model (Web Information Coupling Model), and integrated with existing data in **Whoweda** (see Figure 4). At the warehouse, queries can be answered and Web data analysis can be performed locally. Accessing data at the warehouse does not incur costs that may be asso-

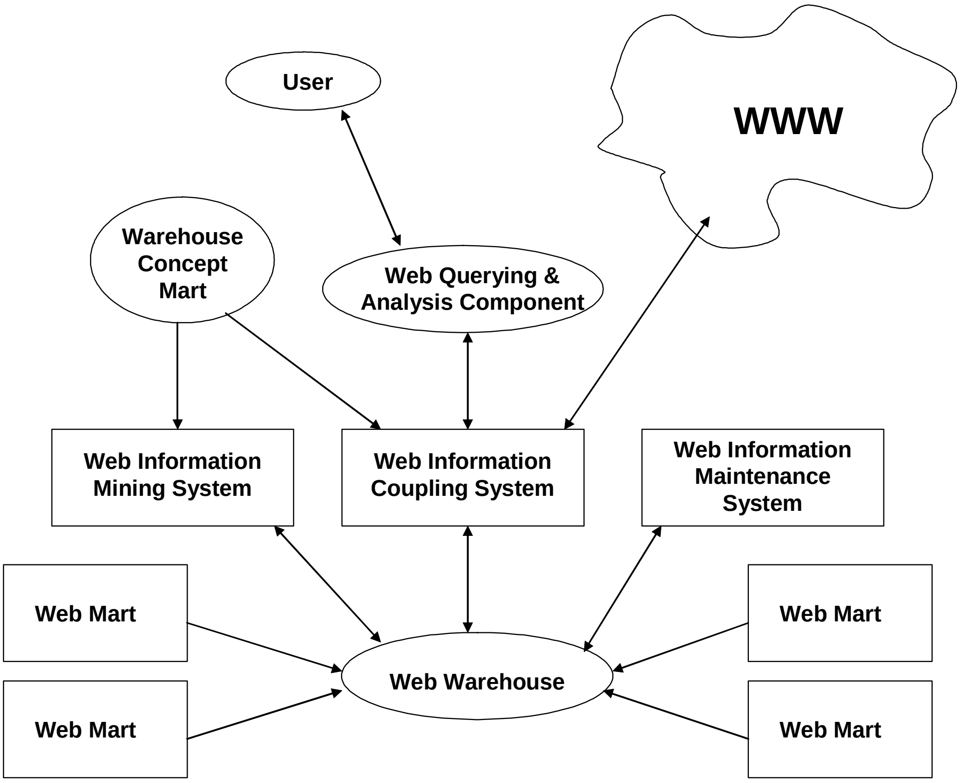
User
WWW
Warehouse Concept Mart
Web Querying & Analysis Component
Web Information Mining System
Web Information Coupling System
Web Information Maintenance System
Web Mart
Web Mart
Web Warehouse
Web Mart
Web Mart

Figure 4: The **Whoweda** Architecture.

ciated with accessing data from the information sources scattered at different geographical locations. In a Web warehouse data is available even when the Web is inaccessible.

The **Whoweda** system consists of two major components: a data manipulation module called Web Information Coupling System (WICS) and a data mining module called Web Information Mining System (WIMS). WICS focuses on the manipulation of information in the **Whoweda** system. It is concerned with the extraction and retrieval of information from the Web, the storage and organization of information in the warehouse, and the data manipulation via various Web operators such as *Web-select*, *Web-join*, and *Web-project*. WICS brings Web data into the warehouse and provides various operators for preprocessing and storing this information. This information is then fed into WIMS for various forms of mining and knowledge discovery. However, as we shift our focus on to the mining of Web information, we are expanding on the scope of the WIMS. Therefore, in the proposed architecture, the WIMS ascend to become the platform, while WICS forms a component that exists in the context of the WIMS.

3.2 Wiccap

The **Wiccap** [4] project investigates issues in information collection, collaging and programming Web data. The **Wiccap** system is a software project that aims to streamline and facilitate the process of content extraction and presentation. As shown in Figure 5, **Wiccap** consist of three main modules, namely, the Mapping Wizard, the Network Extraction Agent and the Presentation Toolkit. The Mapping Wizard generates mapping rules that are used by the Network Extraction Agent. These rules contain meta-information about the information source (e.g., a Web site) that tells the Network Extraction Agent where to find the required information. The Network Extraction Agent is responsible for the actual extraction of information from the Web using the mapping rules. Together, these three modules form a channel in which the information from the

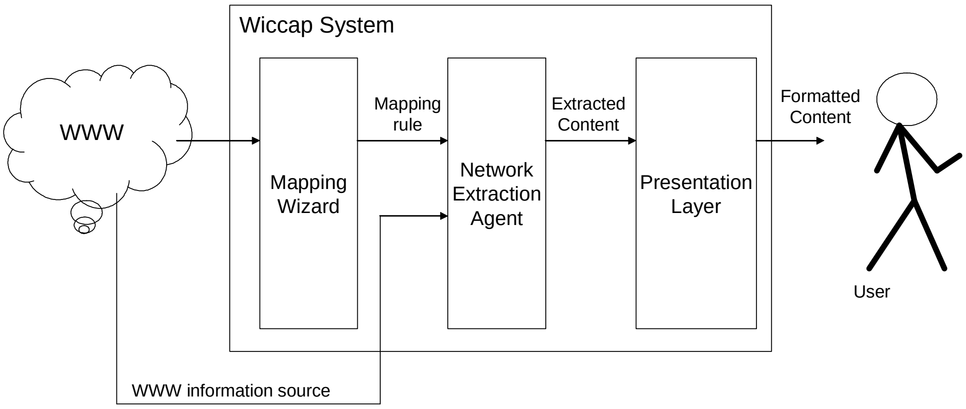

Figure 5: The **Wiccap** architecture.

Web flows through the **Wiccap** system, translating the unstructured information into XML documents.

Since XML is structured, it becomes easy to consume information and to perform data mining. In the context of the Web mining platform, we are interested in the first two modules of **Wiccap**, where the Mapping Wizard and Network Extraction Agent provide the interfaces to content extraction of Web data. While the **Whoweda** system provides comprehensive query facilities to locating Web pages, the **Wiccap** system provides the extraction facilities on HTML pages to transform content into structured manipulatable data. Combining the two systems, the primitive Web protocols and the interfaces in the data mining kernel, the platform is capable of providing access to all Web data in a structured and location independent manner. Together, they provide the solution to the concern that Web data is semi-structured and are distributed physically.

4 Platform for Web Mining

Combining the two existing architectures, we addressed the first four phases of the KDD process (i.e., extraction, cleaning, integration and transformation). What remains are the last three phases (i.e., mining, evaluation and presentation), making the construction of such platform easier.

4.1 The Data Mining Kernel

The role of the data mining kernel is most critical on the platform. First, it provides a consistent set of interfaces to the data mining components. As such, the complexity of the **Whoweda** and **Wiccap** system are shielded, and the specifics of the operating system abstracted. With the kernel, the data mining components can assume the availability of a set of core services such as Web protocols (e.g., HTTP and FTP), XML parsers etc., to be available. Compare this to the case of running the data mining components directly on top of the operating system. The presence of the kernel ensures that if in the event that the operating system does not provide the necessary facilities, the kernel can compensate for the absence. At the same time, the inconsistency of the programming interface among operating systems is also unified. Unlike the Java virtual machine, the data mining kernel contains mostly wrappers to the services on the operating systems. The rational for taking this approach is to balance the ease of programming on one hand against the speed of execution on the other. The virtual machine approach will be too expensive especially when the application requires a lot of computing power.

The second functionality of the kernel is to provide the necessary security features to the incoming data. Information obtained from the Web consists of user profiles and surfing patterns that may be protected by privacy policies. However, traditional data mining applications have access to 'naked' data making it inappropriate for Web mining. To compensate for this, the kernel is responsible for preprocessing some aspect of the data before passing it to the algorithm. This is made possible by moving the micro computations into the kernel instead of having it in the algorithm. There are two advantages in doing so. First, the algorithm has no direct access to the `naked' data and hence allows some form of security mechanisms to be put in place. Second, the data structure and process in the kernel can be shared consistently across all algorithms, making optimizations and improvements possible across all algorithms. This addresses the fourth concern that we raised at the beginning of the paper.

The fifth concern is also addressed by the kernel via the responsibility of monitoring the change in Web data. Using **Whoweda** and **Wiccap**, the data mining kernel is able to monitor the changes on a Web site. When an update occurs, the kernel is able to extract the changes using the facilities in **Whoweda** and **Wiccap**, and then intelligently invoke the incremental version of the data mining algorithm. This event driven approach is different from existing incremental algorithms where the mining takes place via user intervention. Here, the application can indicate to the kernel to perform an incremental mining upon a certain change in the data. For example, using **Whoweda**, we can specify that incremental mining should be done when the warehouse has accumulated a week of changes for a particular Web site.

4.2 Data Mining Components

Although the diagram shows four components running on top of the kernel, there can be other possibilities as the project materializes. The interpreter is the bridge between the application interface and the rest of the data mining facilities. Since the platform supports a variety of application interfaces (e.g., KQML/FIPA or ASP interface), there is a need to translate the request into the internal representation of the data mining components. At the same time, the interpreter also contains specific framework related languages that is interpreted to instantiate an algorithmic framework for data mining.

The set of data mining algorithms and their algorithmic frameworks are placed in the component labeled "Algorithms" in Figure 3. This is where algorithms developed within the group are placed and investigated. Currently, we have a variant of association rule [11] and classification [12] algorithm in place for the platform. It is of course possible to wrap existing code from other sources to quickly increase the platform's data mining capabilities. Each algorithm performs a specific data mining task, and potentially has an incremental version that the kernel can invoke. In addition to the above, each algorithm is developed or wrapped according to some skeleton framework to guarantee a set of interfaces required for inter-component communication. For example, the skeleton framework defines an interface for the kernel such that incremental mining can be activated. At the same time, it also defines an interface for exporting the model in the algorithm to its PMML counterpart.

The PMML component represents the interface for model exchange with other applications as well as to support inter component communication of models within the architecture. For example, the results of the algorithms are visualized by the visualization module. The visualization component represents the set of visualization algorithms that are capable of consuming PMML strings, and then converting them into graphical representations (e.g., generate a HTML file containing the graphic representation of the discovered data model). Instead of passing the results using the internal data structure, the algorithms will first translate the models to PMML and send this to the visualization unit. There are several advantages in doing so. First, this makes the components in the architecture self-contained and modular. This makes development easy and the components can be easily separated to run in a distributed fashion. Second, the use of XML-based representation facilitates reuse of each component. In the case of the visualization unit, it can expose its services such that an agent can actually use the visualization unit to display models created from an external data mining algorithm by simply sending the PMML string to the visualization unit.

4. 3 Application Interfaces

So far, we have described the components that constitute to a Web mining platform. In this section, we touch on the interface between the platform and the applications on the Web. This interface sits on top of the components as shown in Figure 3. In the figure, we illustrated with four possible interfaces on the Web, but more can be included in this layer. Briefly, we see that this includes "Web Services", "API", "ASP Interface" and "KQML/FIPA". With these interfaces, the complexity of data mining can be abstracted allowing more sophisticated knowledge-base systems to be built.

Web services are supported by a number of industry players including Microsoft, IBM, Sun Microsystems and the W3C, to create a simple Web-based application interface using SOAP[5]. The importance of Web services is in its proposition in creating a programmatic Web where existing services on the Web are given a programmatic interface. By taking this approach, ambiguity is removed making the development of Internet applications across different Web sites easy. To date, a number of Web services are already available (e.g., address books, booking of air tickets, book search etc). In line with the vision of having data mining as a service [10], the use of this technology in our design allows the exportation of specific programming interfaces for use by other applications.

On the more traditional approach, programs such as those developed to run on a particular platform will appreciate the application programming interface. This would be the most common approach to developing data mining applications that runs on the same machine as the architecture itself. A variation of this would be the ASP interface where Application Service Providers may need to use the data mining services on the platform. These service providers may not be using an open technology interface like Web services, but instead have some proprietary protocols riding on top of some existing standards such as HTTP. The inclusion of this component identifies the possibility of having proprietary support.

The last interface demonstrates the cross of data mining and agent technologies. Here, KQML[6]/FIPA[7] is standards for agent communication. With this interface, the platform can appear as a software agent to other real agents on the Internet. The rationale behind this is simple. Software agents, particularly mobile software agents, need to be small in size so as to navigate easily around the network. Carrying facilities like data mining of this scale may impede the movements of these agents. By providing such interfaces, an agent can move to the platform, negotiate for the service and then use the platform's facilities to achieve its task. Such computing model is becoming prevalent as software agents, the Web and the ubiquity of wireless devices become common.

[5] *Simple Object Access Protocol, http://www.w3.org/TR/SOAP*

[6] *Knowledge and Query Manipulation Language, http://www.csee.umbc.edu/kqml*

[7] *Foundation of Intelligent Physical Agents, http://www.fipa.org*

5 Application Example

To illustrate the motivation and importance of such a platform, we discuss an advanced application possible. Although the scenario described requires other research efforts, it is complete and sufficient to justify the need for knowledge management platforms such as the one proposed in this paper.

Software agents [13, 14] were touted by many as the candidate that will succeed where artificial intelligence failed. Software agents are essentially program entities that are capable of perform specific tasks in an autonomous manner. Consider a software agent that assists the user in finding information on the Web. Such agents are sometimes called "search agents". A user may specify a set of keywords related to a document that he or she is interested in. Instead of using search engines that tend to return many unrelated results, the keywords maybe specified to a search agent.

Traditionally, a search agent may take the keywords and attempts to perform a search using specific built in algorithms. A more effective way is to build the search agent around an execution framework that defines the task to perform without specifying the "how". In other words, the agent simply needs to know about where to find the "expert" on performing the task within the execution framework. A possibility is as follows. Upon obtaining the keywords specified by the user, the search agent initiates a connection a particular search engine to retrieve a group of related documents. Using its domain knowledge about the keywords specified, the agent may perform a first cut filtering and removal of unrelated documents or it may do so via another agent with the necessary domain knowledge. With the proposed platform, the agent then negotiates the service required via the agent communication protocol (i.e., FIPA/KQML). In our scenario, the agent may request for clustering services where documents are clustered according to their similarity. To do so, the interface component and the interpreter will initiate a protocol to negotiate the required service. Once approved, the interpreter translates the request to an execution plan. In the above scenario, the links of the related documents are passed as the input which the interpreter will schedule the **Whoweda** system to retrieve the documents into the warehouse. Using Whoweda, a snapshot of the documents are taken and stored in the warehouse. Algorithms are then instantiated to perform the data mining task. Once done, the internal data model is then translated by the PMML module in which the clustered results are then returned via the FIPA/KQML interface to the search agent.

As the user clicks on specific documents in the returned list, the agent tracks the documents that the user is interested in. Using association analysis, the frequent keywords in the group of documents are identified. This is done on the basis that documents clicked within a search are related and are therefore candidates for discovering keywords that describes the group of documents. Again, the agent can choose to engage the proposed platform to perform the task. Once the keywords are obtained, the agent can then recognize the differences between the original keywords against the set discovered. This knowledge can then be submitted to other search agents or other search engines as a form of knowledge contribution to the Web. In this case, when another user specifies a group of similar keywords, the search agent may submit the frequent keywords discovered instead of the original to "seed" better search results.

Notice that the above is only possible if there are contributions on a large scale involving many entities. To equip every entity with the knowledge discovery means is not only difficult but at the same time impractical. Therefore, a centralized platform with the capability and horsepower to perform the task on the behalf is needed to realize the above.

6 Summary

In this paper, we proposed the construction of a Web mining platform for the purpose of bridging the differences in data mining algorithms and knowledge on the Web (represented in the form of FAQs, discussion boards etc). We argue that knowledge is different compared to database records that are acquired mainly through automated techniques. To minimize this difference, the proposed platform uses information retrieval techniques such as those in **Whoweda** and **Wiccap**. With **Whoweda,** Web data can be harvested into structured records that can be queried using **Whoweda's** Web query language, while **Wiccap** materializes information on the Web into XML documents that can be structurally processed. The architecture is built using component concepts with XML as the fabric for inter-component communication. Interfaces are part of the platform in order to interoperate with a variety of technologies such as software agents and service provider models. Such an approach allows us to create a platform without starting from scratch and facilitates investigation of newly developed algorithms and techniques in different areas of the Web.

In addition, we also surveyed the maturity of the data mining industry with particular focus on the standardization efforts. We focused on these initiatives as we foresee the need for interoperability of data mining results, and the marshalling of data from distributed sources owned by different organizations. At this point, the proposed platform favors sharing of high level results among components and assumes an efficient communication channel. However, this may not be the case and ways to enhance the efficiency of data exchange in XML may be needed in order for the platform to become useful in general. However, for now, it is sufficient for the purpose of research and enhancements to make the platform suitable for large scale exchange maybe investigated if our resources allow.

To conclude, we believe the future platform for data mining will encompass technologies beyond those dictated within the KDD environment. In particular, as the industry makes the shift to focus on a Web centric computing model, the issue inherent in the Web requires new research directions to bring data mining up-to-date so as to realize the construction of more advanced knowledge systems. Till now, we have focused the reader on the user of different technologies that cross and overlap the boundaries of the data mining discipline. It can be seen that our approach to data mining is unique in the sense that we integrate disciplines from Web warehousing, Web technologies, software agents and databases as the supporting platform. We believe these individual disciplines have strong relationship managing knowledge on the Web. As each individual research efforts mature, it makes sense to bring these disciplines together to create a new level of abstraction for researching knowledge management tasks through data mining. A case for such a need is demonstrated in Section 5 and we believe our discussion in this paper will be a step towards this vision.

References

[1] R. Grossman, S. Bailey, A. Ramu, B. Malhi, P. Hallstrom, I. Pulleyn, and X. Qin. "The Management and Mining of Multiple Predictive Models Using the Predictive Modeling Markup Language", *Information and Software Technology*, **41**:489-595, 1999.

[2] R. Kohavi and F. Provost. "Applications of Data Mining to Electronic Commerce", *Applications of Data Mining to e-Commerce – a Special Issue of the Int. J. on Data Mining and Knowledge Discovery*, January 2001.

[3] C. Lee and H. Yang. "Developing an Adaptive Search Engine for e-Commerce Using a Web Mining Approach", in *Proc. Of the Int. Conf. on Information Technology: Coding and Computing*, pp. 604-608, 2001.

[4] F. Li, Z. Liu, Y. Huang, and W.-K. Ng. "An Information Concierge for the Web", in *Proc. of the 3rd Int. Conf. on Information, Communications and Signal Processing*, Singapore, October 2001.

[5] E.-P. Lim and W.-K. Ng. "A Relational Interface for Heterogeneous Information Sources", in *Proc. of the 2nd IEEE Int. Conf. on Advances in Digital Libraries*, Washington, D.C., May 1997.

[6] E.-P. Lim, C.-H. Tan, B.-W. Lim and W.-K. Ng. "Querying Structured Web Resources", in *Proc. of the 3rd ACM Int. Conf. on Digital Libraries*, Pittsburgh, Pennsylvania, June 1998.

[7] C. X. Liong, J. Gao, H. Zhang, W. Qian, and H. Zhang. "Mining Generalized Query Patterns from Web Logs", in *Proc. of the 34th Annual Hawaii Int. Conf. on System Sciences*, pp. 1816-1824, 2001.

[8] W.-K. Ng, E.-P. Lim, S. Bhowmick, and S. K. Madria. "Web Warehousing System: Design and Issues", in *Proc. of the Int. Conf. on Data Warehousing and Data Mining, in conj. with Int. Conf. on Conceptual Modeling*, Singapore, November 1998.

[9] W.-K. Ng, E.-P. Lim, C.-T. Huang, S. S. Bhowmick, and F. Qin. "Web Warehousing: An Algebra for the World Wide Web", in *Proc. of the 3rd IEEE Int. Conf. on Advances in Digital Libraries*, California, April 1998.

[10] S. Sarawagi and S. H. Nagaralu. "Data Mining Models as Services on the Internet", *ACM SIGKDD Explorations*, **2(1)**, June 2000.

[11] K.-L. Ong, W.-K. Ng, and E.-P. Lim. "Bringing Chaos to Order – A Framework for Association Rule Mining", *submitted to the SIAM 2nd Int. Conf. on Data Mining*, 2001.

[12] A. Sun and E.-P. Lim. "Hierarchical Text Classification and Evaluation", in *Proc. of the IEEE Int. Conf. on Data Mining*, San Jose, California, November 2001.

[13] M. R. Genesereth and S. P. Ketchpel. "Software Agents", *Communications of the ACM*, **37(7),** July 1994.

[14] P. Maes. "Agents that Reduce Work and Information Overload", *Communications of the ACM*, **37(7)**, July 1994.