Data Mining in Android Devices Through Web Services

K . Suresh
Student M.Tech (CSE)
KMM ITS
Tirupati, India

E . Chaitanya krishna
Software Engineer
TCS,Chenai,India

Dr . K .Venkataramana
Asso.Prof, Dept of CSE
KMM ITS,Tirupati

Prof . M . Padmavathamma
HOD,Dept of Comp.science
S.V.University,Tirupati

***Abstract*-- In this paper we discuss about pervasive data mining of databases for data related queries from mobile devices by using Web Services. By implementing mobile Web Services we allow remote users to view the results relating to data analysis large databasesby executing data mining tasksfrom a mobile phone or a PDA. A prototype based on an Android client will be presented, by specifying the data selection task, the server invocation mechanisms is created, and the result converted to format visible by a mobile device.As everyone is familiar with Android now-a-days which is an new open source software stack initiated by Google, and the possibilities of developing a mobile client applications, a service oriented architecture platform based upon SOAP messaging, JSON. The study focuses on the architectural alternatives, their impacts on the mobile client application with data mining applications, Android's performance on SOAP messaging and JSON, and how Web services' design can be optimized to give well performing Android clients.**

Keywords: Data mining, web services, SOA, SOAP, JSON etc.

I. INTRODUCTION

Analysis of data is a complex process that often uses remote resources like computers, software, databases, files, etc. and people (analysts, professionals, end users). Recently, distributed data mining techniques are used to analyze dispersed data sets. Advancement in this research area comes from the use of mobile computing technology for supporting new data analysis techniques and new ways to discover knowledge from every place in which people operate.

The availability of client programs on mobile devices that can invoke the remote execution of data mining tasks and show the mining results is a significant added value for nomadic users and organizations that need to perform analysis of data stored in repositories far away from the site where users are working, allowing them to generate knowledge regardless of their physical location.

A. *Mobile Data Mining*

The goal of mobile data mining is to provide advanced techniques for the analysis and monitoring of critical data from mobile devices.

Mobile data mining has to face with the typical issues of a distributed data mining environment, with in addition technological constraints such as low bandwidth networks, reduced storage space, limited battery power, slower processors, and small screens to visualize the results [1].

The mobile data mining field may include several application scenarios in which a mobile device can play the role of data producer, data analyzer, client of remote data miners, or a combination of them. More specifically, we can envision three basic scenarios for mobile data mining:

The mobile device is used as terminal for ubiquitous access to a remote server that provides some data mining services. In this scenario, the server analyzes data stored in a local or distributed database, and sends the results of the data mining task to the mobile device for its visualization. The system we describe in this chapter is based on this

approach.Data generated in a mobile context are gathered through a mobile device and sent in a stream to a remote server to be stored into a local database. Data can be periodically analyzed by using specific data mining algorithms and the results used for making decisions about a given purpose.

Mobile devices are used to perform data mining analysis. Due to the limited computing power and storage space of today's mobile devices, currently it is not realistic to perform the whole data mining task on a small device. However, some steps of a data mining task (i.e, data selection and preprocessing) could be run on small devices.

MobiMine [2] is an example of data mining environment designed for intelligent monitoring of stock market from mobile devices. MobiMine is based on a client server architecture. The clients, running on mobile devices such as PDAs, monitor a stream of financial data coming through a server. The server collects the stock market data from different Web sources in a database and processes it on a regular basis using several data mining techniques. The clients query the database for the latest information about quotes and other information. A proxy is used for communication among clients and the database.
Thus, when a user has to query the database, she/he sends the query to the proxy which connects to the database, retrieves the results and sends them to the client. To efficiently communicate data mining models over wireless links with limited bandwidth, MobiMine uses a Fourierbased approach to represent the decision trees, which saves both memory on mobile device and network bandwidth.

Another example of mobile data mining system is proposed in [3]. Such system considers a single logical database that is split into a number of fragments. Each fragment is stored on one or more computers connected by a communication network, either wiredly or wirelessly. Each site is capable of processing user requests thatrequire access to local or remote data.

Users can access corporate data from their mobile devices. Depending on the particular requirements of mobile applications, in some cases the user of a mobile device may log on to a corporate database server and work with data there. In other cases the user may download data and work with it on a mobile device or upload data captured at the remote site to the corporate database. The system defines a distributed algorithm for global association rule mining, which does not need to ship all of local data to one site, thereby not causing excessive network communication cost.
Another promising application of mobile data mining is the analysis of streams of data generated from mobile devices. Some possible scenarios are patient health monitoring, environment surveillance, and sensor networks. The Vehicle Data Stream mining (VEDAS) system [4] is an example of mobile environment for monitoring and mining vehicle data streams in real time. The system is designed to monitor vehicles using onboard PDAbased systems connected through wireless networks. VEDAS continuously analyzes the data generated by the sensors located on most modern vehicles, identifies the emerging patterns, and reports them to a remote control center over a low bandwidth wireless network connection. The overall objective of VEDAS is supporting drivers by characterizing their status, and helping the fleet managers by quickly detecting security threats and vehicle problems.

II. MOBILE WEB SERVICES

A Web service is a method of communication between two electronic devices over World Wide Web. A web service is a software function provided at a network address over the web or the cloud; it is a service that is "always on" as in the concept of utility computing. Web Services is not given by specific organization, working with the web service is all about working with multiple technologies, concepts, specifications protocols together to develop distributed applications. Web services are used to develop the interoperable components and also overcome the all above problems. It is also convert the existing business

components into as web services based business components and no need to develop the web application from the scratch level. While working with the hardware components we never bother about their compatibility with other devices and components because they are developed as interoperable components.

To get the communication between two incompatible software applications we take the support of web services. That means the server application can be developed in any language and the client application can be developed in any language. For example the server application developed in dotnet and the client application can be developed in java.

The distributed application development we can use four resources utilization. They are

1. *Service Provider:* is server application of distributed application that develops business methods of business object from remote or local object.
2. *Service client:* is client application of distributed application having capability to call the business methods of business object from remote or local client.
3. *Service Interface:* is a common understanding document between service provider and service client having the declaration of business methods. This will be used by service provider and service client (WSDL that is XML).
4. *Service registry* maintains the object (s) references along with related does for global visibility (to expose to client). Service registry is nothing but registry software of distributed application.

Android is a Linux based operating system which is used to develop touch screen mobiles like smart phones and tablets etc. The Android SDK consists of several tools to help Android application development. This includes an Eclipse IDE plug-in, emulator, debugging tools, visual layout builder, log monitor and more. It contrasts to the other mobile platforms, Android available as open source software. This enables restrictions and enables any device manufacturer to ship devices with Android. Likewise are developers able to distribute applications to any Android device trough the Android market. Unlike Apple's iPhone platform, application distribution does not require any external review or acceptance and multiple application markets exist.

Android web services technologies are JSON, XML, HTTP, SOAP (Structural).

III. SERVICE ORIENTED ARCHITECTURE

Service-oriented architecture (SOA) is a software design and software architecture design pattern based on discrete pieces of software providing application functionality as services to other applications. This is known as service-orientation. It is independent of any vendor, product or technology.

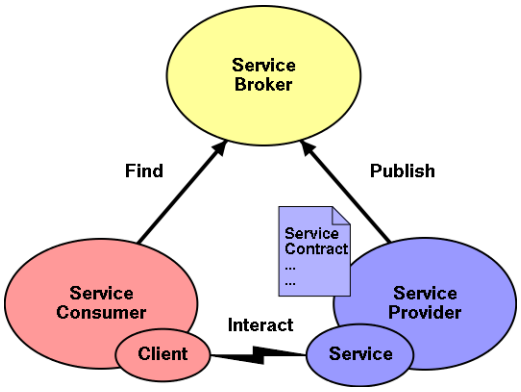
A service is a self-contained unit of functionality, such as retrieving an online bank statement. Services can be combined by other software applications to provide the complete functionality of a large software application. SOA makes it easy for computers connected over a network to cooperate. Every computer can run an arbitrary number of services, and each service is built in a way that ensures that the service can exchange information with any other service in the network without human interaction and without the need to make changes to the underlying program itself.

Provides an approach for building systems focused on a loosely coupled set of components (services) that can be dynamically composed. It promotes seamless software integration as a business benefit.

A. Designing Concept

SOA is based on the concept of a service. Depending on the service design approach taken, each SOA service is designed to perform one or more activities by implementing one or more service operations. As a result, each service is built as a discrete piece of code. This makes

it possible to reuse the code in different ways throughout the application by changing only the way an individual service interoperates with other services that make up the application, versus making code changes to the service itself. SOA design principles are used during software development and integration.



SOA generally provides a way for consumers of services, such as web-based applications, to be aware of available SOA-based services. For example, several disparate departments within a company may develop and deploy SOA services in different implementation languages; their respective clients will benefit from a well-defined interface to access them. SOA defines how to integrate widely disparate applications for a Web-based environment and uses multiple implementation platforms. Rather than defining an API, SOA defines the interface in terms of protocols and functionality. An *endpoint* is the entry point for such a SOA implementation.

Service-orientation requires *loose coupling* of services with operating systems and other technologies that underlie applications. SOA separates functions into distinct units, or services, which developers make accessible over a network in order to allow users to combine and reuse them in the production of applications. These services and their corresponding consumers communicate with each other by passing data in a well-defined, shared format, or by coordinating an activity between two or more services.

For some, SOA can be seen in a continuum from older concepts of distributed computing and modular programming, through SOA, and on to current practices of mashups, SaaS, and cloud computing (which some see as the offspring of SOA).

IV. WEB SERVICES IN ANDROID

A web service is any piece of software that makes itself available over the internet that can be remotely invoked using HTTP, that is, it can be activated using HTTP requests. XML is used to encode all communications to a web service. Web services allow you to expose the functionality of your existing code over the network but code is completely invisible to Web site surfers and software users. Their job is to run silently in the background, For example, a client invokes a web service by sending an XML message, and then waits for a corresponding XML response. Because all communication is in XML, web services are not tied to any one operating system or programming language--Java can talk with Perl; Windows applications can talk with UNIX applications, thus providing a way for applications to work with each other to get the user the information or functionality he needs.

The web services can be developed in android using the any one of the following technologies. They are

1. *REST (Representational State Transfer) Based Web Services.*
2. *SOAP Based Web Services*

A. *REST Based Web Services*

REST defines a set of architectural principles by which you can design Web services that focus on a system's resources, including how resource states are addressed and transferred over HTTP by a wide range of clients written in different languages. If measured by the number of Web services that use it, REST has emerged in the last few years alone as a

predominant Web service design model. In fact, REST has had such a large impact on the Web that it has mostly displaced SOAP-and WSDL-based interface design because it's a considerably simpler style to use.

REST didn't attract this much attention when it was first introduced in 2000 by Roy Fielding at the University of California, Irvine, in his academic dissertation, "Architectural Styles and the Design of Network-based Software Architectures," which analyzes a set of software architecture principles that use the Web as a platform for distributed computing. Now, years after its introduction, major frameworks for REST have started to appear and are still being developed because it's slated, for example, to become an integral part of Java™6 through JSR-311. REST Web service follows four basic design principles:

- Use HTTP methods explicitly.
- Be Stateless.
- Expose Directory Structure like URIs
- Transfer XML, JavaScript Object Notation (JSON) or both.

If you are going to build an android application (it can be any other mobile platform or web too) that manages all the user data on a central database, REST API will be good architectural option to do the communication between the app and the server.

If you consider Evernote, Wunderlist apps, these apps can uninstalled at anytime and once we install them back and login, all our data will be restored. This is because all the data will stored in a cloud database and communication b/w app and database will be done using a REST API.

REST architecture will be useful to build client/server network applications. REST represents Representational State Transfer. Implementing REST is very simple compared to other methods like SOAP, CORBA, WSDL etc., It basically works on HTTP protocol.

V. SYSTEM DESIGN AND MPLEMENTATION

In this section we describe the design and implementation of the system. As mentioned before, the goal of the system is supporting mobile data mining on small devices, such as cellular phones or PDAs, through the use of Web Services. First we introduce the system architecture and describe the design of system components. Then, we present the functionality of the system and its implementation.

The architecture includes three types of components:

1. *Data providers:* the applications that generate the data to be mined.

2. *Mobile clients*: the applications that require the execution of data mining computations on remote data.

3.*Mining servers*: server nodes used for storing the data generated by data providers and for executing the data mining tasks submitted by mobile clients.

As shown in Fig. 1.1, data generated by data providers is collected by a set of mining servers that store it in a local data store. Depending on the application requirements, data coming from a given provider could be stored in more than one mining server.

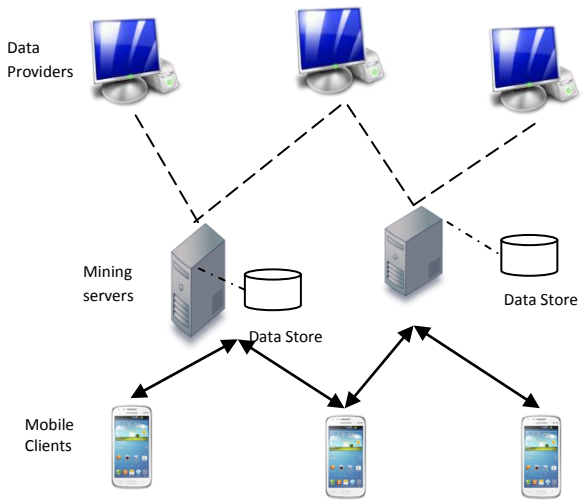


Fig. 5.1 General ***architecture of the system***

The main role of mining servers is allowing mobile clients to perform data mining on remote data by using a set of data mining algorithms. Once connected to a given server, the mobile client allows a user to select the remote data to

be analyzed and the algorithm to be run. When the data mining task has been completed on the mining server, the [redacted] lized on the user device

[redacted]

oftware components of

server exposes its

Services: the *Data Collection Service* (*DCS*) and the *Data Mining Service* (*DMS*). Fig. 1.2 shows the DCS and DMS and the other software components of a mining server.

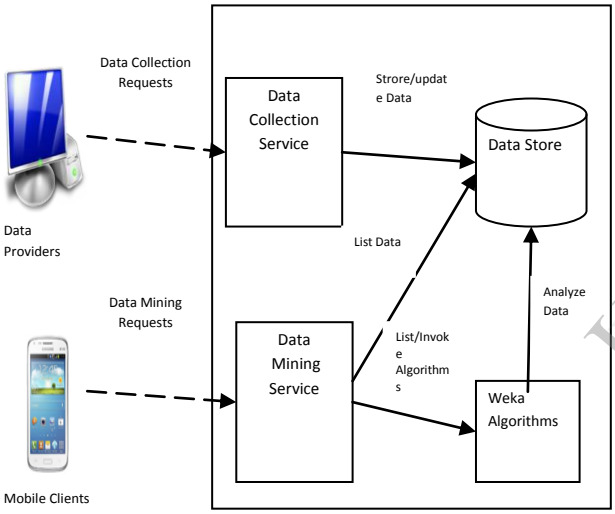


Fig. 5.2. ***The software components of a mining server***

The DCS is invoked by data providers to store data on the server. The DCS interface defines a set of basic operations for uploading a new data set, updating an existing data set with incremental data, or deleting an existing data set. These operations are cumulatively indicated as *DCS ops* in the figure. Data uploaded through the DCS is stored as plain data sets in the local file system. As shown in the figure, the DCS performs either store or update operations on the local data sets in response to data providers requests.

The DMS is invoked by mobile clients to perform data mining tasks. Its interface defines a set of operations (*DMS ops*) that allow to: obtaining the list of the available data sets and algorithms, submitting a data mining task, getting the current status of a computation, and getting the result of a given task. Table 1.1 lists the main operations implemented by the DMS.

The data analysis is performed by the DMS using a subset of the algorithms provided by the Weka library [16], which includes a large collection of machine learning algorithms written in Java for data classification, clustering, association rules discovery, and visualization. When a data mining task is submitted to the DMS, the appropriate algorithm of the Weka library is invoked to analyze the local data set specified by the mobile client.

- ***Mobile client*** The mobile client is composed by three components: the *Android App*, the *DMS Stub*, and the *Record Management System* (*RMS*) (see Fig. 1.3)

Table 1.1 Data Mining Service operations

Operation	Description
listDataSets	Returns the list of the local Data Sets
listAlgorithms	Returns the list of the available DM algorithms
submitTask	Submit a DM task for the analysis of a given dataset using a specific DM algorithm. Returns the unique id for the task.
getStatus	Returns the current status of the task with a given id. The status of a task can be running, done, or failed.
getResult	Returns the Result of the task with a given id, either in textual of visual form.

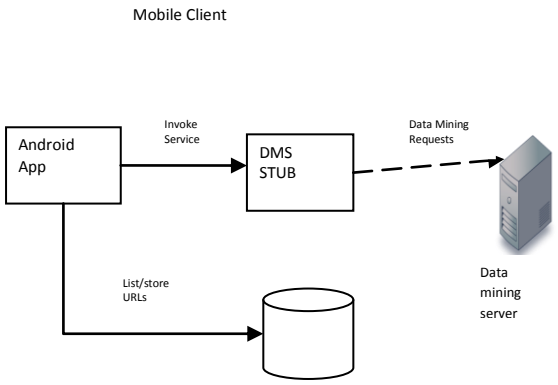


Fig. 5.3 The software components of a mobile client

The Android App allowing the user to perform data mining operations and visualize their results. The DMS Stub is aWeb Service stub allowing the Android App to invoke the operations of a remote DMS. The stub is generated from the DMS interface to conform with the JSR 172 specifications, introduced in the previous section. Even if the DMS Stub and the Android App are two logically separated components, they are distributed and installed as a single ANDROID application.

The RMS is a simple recordoriented database that allows Android applications to persistently store data across multiple invocations. In our system, the Android App uses the RMS to store the URLs of the remote DMSs that can be invoked by the user. The list of URLs stored in the RMS can be updated by the user using a Android App functionality.

C*Functionality of the system*

In the following we describe the typical steps that are executed by the client and server components, to perform a data mining task in our system:

1. The user starts the Android App on his/her mobile device. After started, the Android App accesses the RMS and gets the list of remote mining servers. The list is shown to the user who selects the mining server to connect with.

2. The Android App invokes the listDatasets and listAlgorithms operations of the remote DMS in order to get the lists of data sets and algorithms that are available on the server. The lists are shown to the user who selects the data set to be analyzed and the mining algorithm to be used.

3. The Android App invokes the submitTask operation of the remote DMS, passing the data set and the algorithm selected by the use with associated parameters. The task is submitted in a batch mode: whenever the task has been submitted, the DMS returns a unique *id* for it, and the connection between client and server is released.

4. After the task submission, the Android App monitors its status by querying the DMS. To this end, the Android App periodically invokes the getStatus operation, which receives the *id* of the task and returns its current status (see Table 1.1). The polling interval is an application parameter that can be set by the user.

5. Whenever the getStatus operation returns *done*, the Android App invokes the getResult operation to receive the result of the data mining analysis. Depending on the type of data mining task, the Android App asks the user to choose how to visualize the result of the computation (e.g., pruned tree, confusion matrix, etc.).

VI CONCLUSIONS

This paper discussed insidious data mining of databases from mobile devices through the use of Web Services. By implementing mobile Web Services we allow remote users to execute data mining tasks from a mobile phone or a PDA and receive on those devices the results of a data analysis task. A prototype based on an Android Application client has been discussed by describing the data selection task, the server invocation mechanisms and types of technologies used for web services in Android devices.

REFERENCES

1. S. Pittie, H. Kargupta, B. Park. Dependency detection in MobiMine: a systems perspective. Information Sciences, 155(34): 227243 (2003)
2. H. Kargupta, B. Park, S. Pitties, L. Liu, D. Kushraj, K. Sarkar. Mobimine: monitoring the stock marked from a PDA. ACM SIGKDD Explorations, 3(2): 3746 (2002)
3. F. Wang, N. Helian, Y. Guo, H. Jin. A Distributed and Mobile Data Mining System. Proc. Int. Conf. on Parallel and Distributed Computing, Applications and Technologies (2003)
4. H. Kargupta, R. Bhargava, K Liu, M. Powers, P. Blair. S. Bushra, J. Dull. VEDAS: A Mobile and Distributed Data Stream Mining System for RealTime Vehicle Monitoring. Proc. SIAM Data Mining Conference (2003)
5. K. Channabasavaiah, K. Holley, E. M. Tuggle. Migrating to a serviceoriented architecture. http://www106. ibm.com/developerworks/library/wsmigratesoa (visited: Jan. 2007)
6. Web Services Activity. http://www.w3.org/2002/ws (visited: Jan. 2007)
7. Web Services Description Language (WSDL). http://www.w3.org/TR/wsdl (visited: Jan. 2007)
8. Simple Object Access Protocol (SOAP). http://www.w3.org/TR/soap (visited: Jan. 2007)
9. Universal Description, Discovery, and Integration. http://www.uddi.org (visited: Jan. 2007)
10. M. Adaçal, A. B. Bener. MobileWeb Services: ANewAgentBased Framework. IEEE Internet Computing, 10(3): 5865 (2006)
11. M. Tian, T. Voigt, T. Naumowicz, H. Ritter, J. Schiller. Performance Considerations for MobileWeb Services. Computer Communications, 27(11): 10971105 (2004)
12. H. Chu, C. You, C. Teng. Challenges: Wireless Web Services. Proc. Int. Conf. Parallel and Distributed Systems (ICPADS 04), IEEE CS Press (2004)
13. W. Zahreddine, Q. H. Mahmoud. An Agentbased Approach to Composite Mobile Web Services. Proc. Int. Conf. on Advanced Information Networking and Applications (AINA'05), IEEE CS Press (2005)
14. JSR 172: J2ME Web Services Specification. http://jcp.org/en/jsr/detail?id=172 (visited: Jan. 2007)
15. Java Micro Edition. http://java.sun.com/javame (visited: Jan. 2007)
16. H. Witten, E. Frank. Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann (2000)
17. Sun java Wireless Toolkit http://java.sun.com/products/sjwtoolkit (visited: Jan. 2007)
18. Apache Axis. ws.apache.org/axis (visited: Jan. 2007)
19. The UCI Machine Learning Repository. http://www.ics.uci.edu/»mlearn/ MLRepository.html (visited: Jan. 2007)