Sādhanā Vol. 39, Part 6, December 2014, pp. 1259–1270. © Indian Academy of Sciences

Design and development of a web-enabled data mining system employing JEE technologies

VARUN KRISHNA*, JINTOMON JOSE and N N R RANGA SURI

Centre for Artificial Intelligence and Robotics, DRDO Complex, C V Raman Nagar, Bangalore 560 093, India
e-mail: varunkrishan@cair.drdo.in; jintojose@cair.drdo.in; nnrsuri@cair.drdo.in

MS received 24 February 2014; accepted 15 August 2014

Abstract. With the advent of cost effective storage systems and high speed network connectivity, the amount of data gathered by various transactional systems has increased manifold and processing the same has become a real challenge. A number of data mining systems have been developed for processing such huge data in a meaningful way. Most of these systems are stand-alone in nature except a few of them that facilitate a subset of their overall functionality for web-based usage. This is mainly due to the challenges involved in designing the architectural framework required for developing a web-enabled data mining system. Such a system is aimed at analysing the transactional data pertaining to various input domains employing some advanced graph mining techniques like subgraph matching, frequent subgraph discovery and graph visualization. In this paper, we present an innovative approach employing various Java Enterprise Edition (JEE) based technologies for developing a data mining system that operates in the web environment. This paper basically presents the architectural principles required to design a suitable framework for enabling the development of such a system and the implementation challenges faced in realising the same. The paper also discusses functional details of the system involving various graph mining techniques. A few general guidelines based on our understanding during the system implementation are also included in this paper for completeness.

Keywords. Data mining; graph mining; web-based system; JEE technologies.

1. Introduction

The emergence of various online communication systems (like Gmail, Skype), social networks (Scott 2012) (like Facebook, Twitter) and automation of various transactional systems (Brewer *et al* 2006) have led to a massive generation of data. Data mining has begun to play a pivotal role in the processing of these huge amounts of data thus generated. Data mining methods have been extensively applied in various application domains ranging from analysis of the simple

*For correspondence

market basket (transactional) data to great scientific discoveries dealing with astronomical data (Agrawal & Srikant 2004; Chaudhary *et al* 2002). It has been applied for analysing the web usage data to discover various online communities and potential anomalies/outliers. Data mining has also been used for analysing various social and communication networks for exploring their connectivity (link) structure leading to prediction of future links among the network objects (Nowell & Kleinberg 2003). It helps in product research and marketing campaigns based on the discovery of customer consumption patterns (Agrawal & Srikant 2004). It also plays a significant role for fraud detection in financial institutions and stock markets (Chandola *et al* 2009).

Processing of huge data connected with these application domains involves appropriate grouping (clustering) (Xu & Wunsch 2005; Jain *et al* 1999), characterization (classification) (Han *et al* 2006) of the data and finding relevant patterns (Krishna *et al* 2011) in the data that can be of use in various application scenarios. A number of data mining tools exist today that can process the data making use of various machine learning and pattern recognition techniques like Bayesian networks and Decision trees to analyse the data and provide valuable information to the users. Some tools provide the additional functionality of forecasting the trend based on the available data employing various predictive models.

Due to the large amount of data that has to be handled and the high computational resources required for processing the same, most of the existing data mining tools are designed to be stand alone systems installed on a system equipped with the required resources. The huge costs involved in setting up such a computing system make these data mining systems unavailable for academicians and independent researchers. Also, there exist many usage scenarios requiring a data mining system only for a short period of time. In such cases, systems over web or a network would be very much beneficial.

The requirement to have a data mining system operating over web is very much essential given the fact that internet connectivity across the globe has reached every nook and corner. The advantages of having a web-based data mining system are as follows:

- Users need not have to worry about high computational resources required for running such a system, as all the resource requirements are taken care at the server end.
- Web-enabled systems do not require any dedicated network as all the communication between server and client can take place through the existing internet gateways.
- Global accessibility of the application hosted by a company or an institution with geographically spread out branches without any additional costs.
- All the other advantages of a web-based application such as security, reliability and availability offered by the internet framework.

As per our understanding of the philosophy of designing web-based systems considering the specific application oriented functionality to be provided, we identify the following challenges in developing such a system:

- Designing a suitable framework for supporting the system requirements.
- Ability to handle large volume of data.
- Processing the data in a reasonable amount of time.
- Meaningful visualization of the processed data at the user-end.

Our attempt in this regard is to design and develop a web-enabled data mining system addressing the challenges brought out above employing suitable tools and techniques.

Of the various frameworks available for developing such a complex web application, we found JEE (Johnson 2005) to be the most relevant one suiting the above listed requirements. JEE is a

Java based enterprise oriented application environment meant for the development of systems employing the client-server architecture in web environment or over a network.The specific data mining system that we have developed is based on a hybrid architecture which is a combination of three JEE technologies namely JSF, Spring and Hibernate (Chao 2008) that facilitates graph based data mining functionality. It supports a set of advanced graph mining functions like graph matching (Conte *et al* 2000), frequent sub-graph discovery (Cook & Holder 1994) and has a graph visualization (Herman *et al* 2011) module. A detailed discussion on these graph mining functions is furnished in the following section.

An implementation of the graph based data mining system employing the chosen JEE technologies indicates the complexity involved in developing such a system. Architectural diagrams indicating the implementation details are also furnished in this paper with a short discussion wherever necessary.

The rest of the contents of this paper are organized into five different sections. Section 2 sheds light on the existing data mining systems and tools. Section 3 details the functional aspects of our data mining system. Section 4 focuses on the proposed design methodology and the architectural details of our system. Implementation details of the system are explained in section 5. The additional features that we incorporated in our system are brought out in section 6. Finally, we conclude by throwing light on the path forward in the field of web-based data mining systems development.

2. Survey of the existing systems

Most of the data mining systems available in the market today have implemented various popular data mining techniques like classification, clustering and association rule mining, etc. More recently, the text mining capability is also being included in these systems. Some of the well-known data mining systems are R (Zhao 2012), SAS (SAS 2013), IBM-SPSS (Statistics 2008), WEKA (Hall *et al* 2009) and STATISTICA (StatSoft 2001). R is a popular data mining tool which facilitates various mining tasks like classification, cluster discovery, regression, anomaly detection and network analysis. Since it is an open source system with an active development community, a number of new functionalities get added continuously making it the most extensively used data mining system as brought out in a recent survey of such systems (Rexer 2011). SAS is a commercial software that includes many data mining modules connected with predictive and descriptive modelling, forecasting, simulation and experimental design. It provides a lot of analysis capabilities for classification, clustering and interpretation of the data patterns. IBM-SPSS is another commercial data mining tool which is in use across various domains. SPSS Statistics has a server version which works in client–server architecture mode and supports few additional functionalities over and above its desktop version. WEKA is another open source data mining tool which supports many of the capabilities provided by the commercial tools. It is well in use among academicians and researchers. STATISTICA offers data mining capability across various product lines which include desktop and web-enabled solutions. STATISTICA web-enabled enterprise server is built based on the open architecture and includes .NET compatible development tools that enable future expansion of the basic system. It has a user interface which supports interactive data analysis and data mining capabilities along with database management and querying functionalities.

In summary, many of the tools discussed above provide stand alone or desktop based data mining capabilities, while some of them provide this functionality over the platforms employing the client–server architecture.

3. Functional details of our system

The specific data mining system that we have developed is meant for facilitating various graph based data mining tasks. It is basically designed to represent the input transactional data in the form of a graph structure and carry out different graph-based analysis operations like graph matching and frequent sub-graph discovery on such a representation as described below.

- Graph generation: This functionality of our system creates a graph representation by exploring the inherent link structure in the input data as per the perceived knowledge of the underlying transactional domain.
- Graph matching: This is the process of finding all instances of a given query graph in a large transactional graph. Typically, query graph is a small graph, which represents a particular pattern of interest to the user. We have developed a novel algorithm named as Multi-labelled Graph Matching Algorithm (MuGRAM) (Krishna *et al* 2012) for providing the graph matching functionality.
- Frequent sub-graph discovery: It is an important analysis technique used to find all the frequently occurring sub-graphs within a large transactional graph. A sub-graph is deemed frequent depending on the threshold value set by the user on the frequency count. We have used an implementation of SUBDUE (Ketkar *et al* 2005) for providing this functionality in our system.
- Graph visualization: This capability of our system has been developed by extending the JUNG (O'Madadhainr *et al* 2003) library and helps the user to visualize the transactional graphs. It provides various layout options for the user and helps him/her to find certain graph patterns through graph visualization. The visualization module has been equipped with other additional features that facilitate a better semantic understanding of the graph. The focus of our system has been mainly facilitating graph based data mining analysis functionality owing to the fact that the standard exploratory analysis features have been implemented in almost every data mining tool.

Development of a data mining system providing the above described functionalities in a user friendly way through a web browser is a challenging task. Addressing this challenge, the user interface developed as a part of our system implementation has a multi-component layout wherein the user can view/edit each record from the database tables. A view component in the user interface is reserved exclusively for involving the data mining operations interactively. The view components are arranged in a semantically appealing manner providing convenient use to the user.

4. Proposed design methodology

The choice of JEE as the preferred framework for developing our data mining system was due to its wide variety of features. JEE supports component based framework and consists of several API specifications, which include Java Database Connectivity (JDBC), Remote Method Invocation (RMI), web services, XML, etc. JEE also covers some unique specifications which include Enterprise JavaBeans, Connectors, Servlets, Java Server Pages and several other web service technologies. A JEE application server provides various features like transaction management, application security, concurrency management, application scalability, and hassle free inter-operability of the components deployed in it. JEE provides multi-tiered architecture thus supporting modular independence and enhanced scalability. Even though many systems have

been developed based on the multi-tier architecture, our system stands unique due to the fact that we have optimized the usage of various components depending on the functional requirements and added a security layer using Spring Security framework (Mularien 2010) which operates in a cohesive manner with the JEE framework.

Our system is modularized into four tiers, namely Presentation tier, Business logic tier, Persistence tier and Database tier as shown in the system architecture in figure 1.

- Presentation tier: Presentation tier represents the front end of the system. It is this tier with which user has direct interaction. All the communication to the system by the user happens through this tier. The Presentation tier takes the request from the user and loads the required web pages. It also transfers the user driven request to Business logic tier for processing and retrieves the result from Business logic tier and displays on the screen.
- Business logic tier: Business logic tier houses all the major data processing functionalities of the system. It inter-connects Presentation tier with Persistence tier. All database requests from Presentation tier pass through Business logic tier. This tier fetches the results and sends them to the application client.
- Persistence tier: The basic role of Persistence tier is in data access management and resource handling. Data access management mainly deals with accessing data in database. Persistence tier provides the abstraction for database access and thus insulates it from other tiers. Resource handling deals with establishing connections to the database as required by the application.
- Database tier: This tier lies outside the JEE Application server and it consists of a relational database server. We chose to use Postgre SQL as the relational database server in our application as it is one of the popular open source database management systems.

5. Implementation details

The first step in the implementation of our system was identification of the right technologies with the capability to provide the required functional support.

Presentation tier in our system is implemented using Java Server Faces (JSF) (Bergsten 2009) framework as shown in figure 2. We used an open source implementation of JSF, namely Prime Faces (Civici 2012) for realising the multi-component flexible layout boundary based user interface for our application. JSF is a server side specification for creating user interface for JEE based applications. It has a broad user interface (UI) component model which consists of various UI components, event handling model, validation model, navigation model and page navigation support. It has a rich tag library and a huge number of web page design options. A number of open source communities have developed their own versions of JSF based on core JSF specifications. The basic view templates in JSF are called Facelets. Facelets are configured using the parameter values specified in the faces configuration file written in XML. The display of a web page is rendered using the faces servlet class and the faces configuration file. The faces configuration file manages the navigation controls, bean scopes and theme settings of the application. Faces servlet processes all incoming requests and passes it on to Business logic tier. Managed beans are used to hold data from Presentation tier and hand them over to Business logic tier and also handle navigation control of the data.

Business logic tier is implemented using Spring framework (Johnson *et al* 2009) as shown in figure 3. This is due to the fact that Spring supports a number of advanced features like inversion of control (IOC) (Fowler 2004), data access, transaction management and authentication and

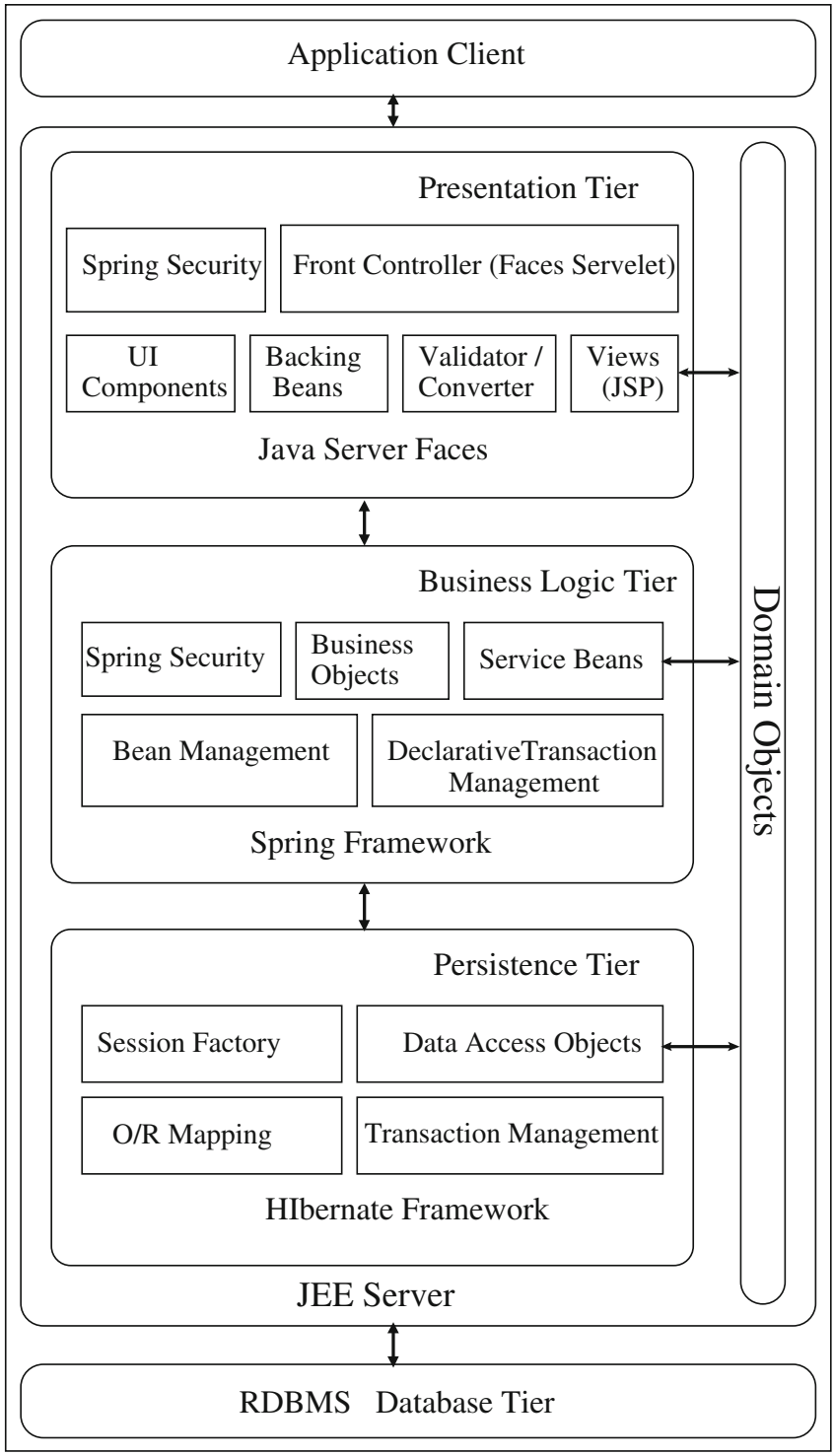

Figure 1. System architecture based on the JEE framework.

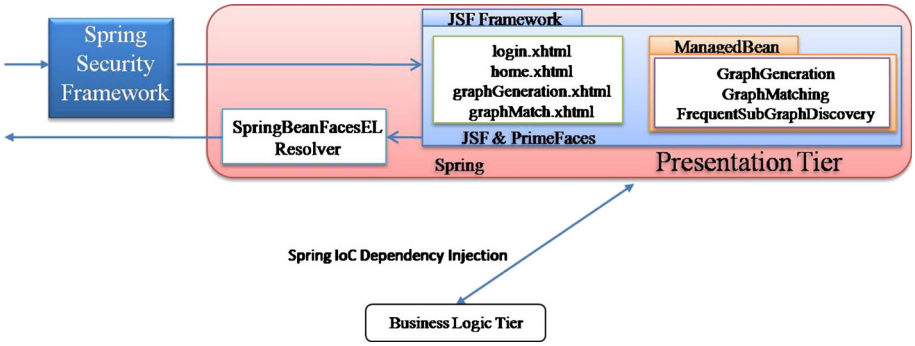

Figure 2. Implementation details of Presentation tier.

authorization. Spring framework also provides abstraction layers for transaction management, and integration capabilities with O/R mapping frameworks. It has the capability to interact with JSF in Presentation tier and Hibernate in Persistence tier. Dependency injection plays a major part in this interaction process. Dependency injection is the process of injecting a property or object of a class into another class (rather than the calling class itself creating an object of the called class) retaining the independence and modularity of the involved Java components. IOC feature of Spring is used for dependency injection.

The Persistence tier takes care of the database access management. The O/R framework named Hibernate (Bauer & King 2005) is used to manage database access. Hibernate makes use of persistent objects which are Plain Old Java Objects (POJOs). These POJOs represent the database tables in the Database tier. The mapping of POJO classes with the database tables can be done by using an XML configuration file or annotations in the respective POJO class. Annotations can be used in the case of small to medium sized applications. Due to scalability issues, we use XML configuration file for O/R mapping. A hibernate mapping file is created for each of the tables and a corresponding POJO class is also created. Thus, Hibernate ensures mapping between the class created for each table and the corresponding table in the database through the mapping file. Thus

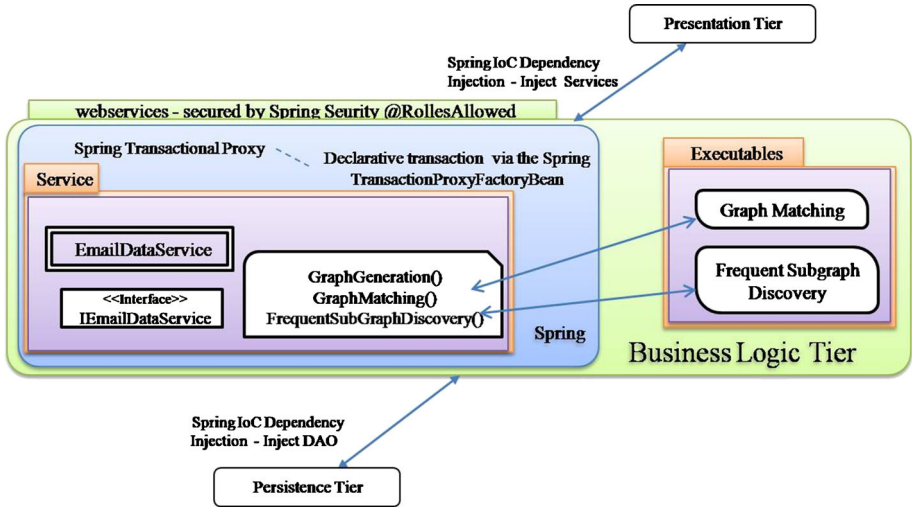

Figure 3. Implementation details of Business logic tier.

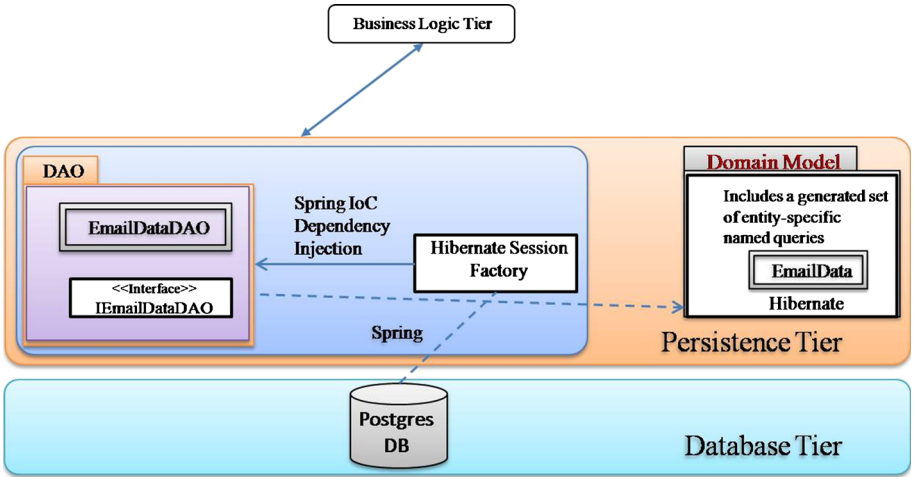

Figure 4. Implementation details of Persistence tier.

a database table can be accessed using an object of the class created for that table. Persistence tier is implemented in our application as shown in figure 4.

The main advantage of using Hibernate in Persistence tier is that it ensures database independence. Database independence is referred to the ease of switching from one database to other without having to change any major functionalities or code. It also supports lazy loading technique along with pagination which enables loading only a required number of rows from database at a time. This is a very important property for the display of a large number of records from the database, because retrieving all the records from database in one shot might be time consuming.

5.1 *Integrating the technologies*

The above discussed tiers and the corresponding technologies form the building blocks of our browser based system development. However, the real challenge lies in integrating these technologies in a seamless manner ensuring smooth and error free operation.

5.1a *Integration of JSF with spring*: The first step in the integration process is configuring the context listener of the application by invoking Spring's listener classes (ContextLoaderListener and RequestContextListener) in the deployment descriptor file. To integrate JSF with Spring, we have to ensure compatibility among the beans in Spring and managed beans in JSF. To ensure seamless integration, we define SpringBeanFacesELResolver as an 'el-resolver' parameter in the faces configuration file. We can implement the dependency injection using two methods, namely setter injection and constructor injection (Prasanna 2009). Due to the simplicity in implementation, we have considered setter injection for dependency injection.

5.1b *Integration of hibernate with spring*: Once the database is created and is made available for the application, we create a data source entry in the application-context file providing the necessary parameter values. The application context file is the configuration file for Spring which plays a key role in enabling integration between Spring and Hibernate. To integrate Spring with Hibernate, Hibernate SessionFactory is configured using SpringFactory Object-Relation Mapping (ORM) in the application context file. We use separate Data Access Object (DAO) classes

to interact with the database which encapsulates the underlying persistence mechanism. Spring provides LocalSessionFactoryBean which creates a SessionFactory object, which is directly injected as a property into the DAO beans. It supports Hibernate's SessionFactory and thus takes care of the Hibernate session per database transaction. Spring also has a 'TransactionProxyFactoryBean' which works along with the Hibernate transaction manager, thus making the tier transaction aware. The transaction management is assigned to a Hibernate transaction object that is retrieved from the Hibernate session by the Hibernate transaction manager. When a transaction is completed successfully, the transaction manager commits it and if it fails, the same is rolled back.

5.2 *Guidelines for building a web-based data mining system*

During the development of the browser based data mining system, we came across a number of issues and we followed certain design principles to overcome them. In this section, we focus on the aspects that need to be considered in developing any such browser based data mining system.

- Appropriate application framework: For developing a complex data mining tool, it is always preferred to opt for a well-supported application framework like JEE or .NET. We opted for JEE framework so as to ensure application compatibility with the data mining modules which were developed in Java.
- Code modularization: The designer should align the design with modularity concept of the framework for better maintainability and re-usability of the code in the application.
- Seamless integration of technologies: We have chosen three technologies for implementing the application so that each tier of the application is addressed in the best possible manner. When such hybrid approach is used, seamless integration of the technologies should be ensured and appropriate testing should be done before fielding the application.
- Coding standard: Appropriate naming conventions and packaging of the modules need to be done to avoid any kind of ambiguity. Following any of the existing coding standards is a must to ensure better code maintainability.
- Binding third-party modules: Even though the framework can provide flexibility to plug-in third party modules and executables, it is always preferred to bind in the modules written in the same programming language supported by the framework rather than plugging in an executable of the module because of the fact that it gives better control over calling an executable.
- Directory structures and file names: Appropriate directory structure and file naming notation need to be followed for user input files and for the intermediate output files generated.

These are some of the important aspects that the designer and the developer should keep in mind while developing a web-based data mining application.

6. Additional features

6.1 *Ergonomics*

Ergonomics of a web application refers to the ease of use of the application and the associated guidelines to be followed while designing the user interface to ensure the better usability of the

system. The user interface for the data mining application under consideration has been designed with two key points in mind:

- Maximum utility with minimum navigation.
- Enhanced viewability in the display area.

To provide a view of multiple functionalities without any clutter in a single screen was one of the design criteria of our user interface. We used multi-component layout to realize this, wherein the user had the option to perform various data mining operations along with providing the option to view a record from the database. The use of Primefaces, an open-source implementation of JSF helped in implementing the multi-component layout without having to compromise the JEE modularity concept.

The second criterion comes in view of the graph visualization aspect. Since the whole application is rendered through the browser, the display of large graphs would be more legible if the display area is put to maximum use. This has been facilitated by providing a button beside the menu bar in the screen which when clicked would expand the graph visualization pane to most of the display area by shrinking the other components in the screen.

The User Interface has been designed adhering to the standard user interface design principles like simplicity, clarity, visibility and flexibility. Uniformity in terms of colours and fonts has also been maintained throughout the screens and appropriate shortcuts in relevant screens have been incorporated to enable a hassle free user interaction with the system.

6.2 *Application security*

Ensuring security of the entire application is an important non-functional requirement, more in case of web-enabled applications. For data mining systems dealing with large amount of data, security is inevitable to prevent data loss, data misuse and unauthorised data manipulation. Typically, a web application requires multiple layers of security. In this section, we focus on the security oriented features implemented at the application level. The JEE application level security is handled in our system using the Spring Security framework.

Spring Security is an implementation of the JEE framework that primarily ensures authentication and access control for JEE applications. It is basically an extension of Java Authentication and Authorization Service (JAAS) (Authentication 2005), a standard authentication framework used in Java. Spring Security also provides the option for specifying, modifying, storing and retrieving access control lists. Spring Security provides credentials based authentication for the application. Spring Security authorisation module considers two aspects: assigning a role to the user and the privileges associated with that role.

Implementation of Spring Security for the Presentation tier involves three steps (Mularien 2010): creation of a Spring Security configuration file, adding the Spring DelegatingFilterProxy to the deployment descriptor file and adding a reference of Spring Security configuration file in the deployment descriptor file.

A Spring Security configuration file is created in which, we mention the different user roles or role categories and also different web-pages that can be accessed by the particular user group. According to the specified configuration, only administrator has the privileges to access graph mining functionalities. If any other user tries to access these functionalities, the system will display an 'access denied' message. The security for a particular element of a page instead of the entire page can be enforced using Spring Security tag library. An application is secured by Spring Security using a set of Servlet Request Filters. Every request to the application is thus

secured using these filters. The 'DelegatingFilterProxy' offered by Spring framework is used in our application which acts as a filter for every application request. In the final step, we add a reference of Spring Security configuration file in the deployment descriptor file.

Securing Business logic tier is very essential due to the fact that the Business logic tier can be exposed via the Presentation tier; i.e., if there is any security breach in the Presentation tier, it can percolate into the Business logic tier. In our application, Business logic tier is secured using Java Specification Request (JSR-250) (Mordani 2009) annotation. JSR-250 has four elements which are used in our implementation, these are: RunAs, RolesAllowed, PermitAll, DenyAll. RunAs allows us to execute a method as a particular user. RolesAllowed gives the option of mentioning the roles/users who can access the method. PermitAll allows every user, whereas DenyAll denies every user from accessing the method.

7. Conclusion and future work

Having explored the importance of a web-enabled data mining system and the challenges involved in developing the same, we have come up with a novel architectural design required for implementing such a system. In this paper, we have discussed the architectural features of our system along with the implementation details employing some latest JEE technologies namely, JSF, Spring and Hibernate. The system that we have developed has the capability to carry out graph-based mining of the data over the web. The web-enabled data mining application can even be hosted on a cloud environment with high performance computing infrastructure. The advancements in the network security domain and the enhancements in the network bandwidth availability are expected to facilitate further improvements in dealing with various data mining tasks in the web browser environment more effectively.

Acknowledgements

The authors would like to thank the Director, Centre for Artificial Intelligence and Robotics (CAIR) for supporting this work. The authors also thank Dr G Athithan, Scientist-'H' (formerly with CAIR) for providing valuable suggestions during this work.

References

Agrawal R and Srikant R 2004 Fast algorithms for mining association rules in large databases. In: *Proceedings 20th International Conference on Very Large Data Bases*, San Francisco, CA, USA, pp 487–499

Authentication J 2005 Authorization service (jaas). URL http://java.sun.com/j2se/1.5.0/docs/guide/security/jaas/JAASRefGuide.html 14

Bauer C and King G 2005 *Hibernate in action*. Manning Publications Co., Greenwich, CT

Bergsten H 2009 *Java Server Faces*. O'Reilly Media, Inc.

Brewer A, Neubauer B and Geiselhart K 2006 Designing and Implementing E-Government Systems Critical Implications for Public Administration and Democracy. *Administration and Society* 38(4): 472–499

Chandola V, Banerjee A and Kumar V 2009 Anomaly detection: A survey. *ACM Computing Surveys (CSUR)* 41(3): 15

Chao H G F 2008 Use JSF, Spring, Hibernate for Web Application Development. *Microcomputer Information* 9: 80

Chaudhary A, Szalay A S, Szalay E S and Moore A W 2002 Very fast outlier detection in large multidimensional data sets. In: *Proc. ACM SIGMOD Workshop in Research Issues in Data Mining and Knowledge Discovery*

Civici C 2012 PrimeFaces Users Guide. http://primefaces.org/documentation.html

Conte D, Pasquale F, Carlo S and Vento M 2000 Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recognition and Artificial Intelligence* 18(3): 265–298

Cook J and Holder B 1994 Substructure discovery using minimum description length and background knowledge. *J. Artificial Intel. Research* 1: 231–255

Fowler M 2004 Inversion of control containers and the injection pattern. http://martinfowler.com/articles/injection.html

Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P and Witten I H 2009 The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter* 11(1): 10–18

Han J, Kamber M and Pei J 2006 Data mining: Concepts and techniques. Morgan Kaufmann

Herman I, Melancon G and Scott M 2011 Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics* 100(2): 190

Jain A K, Murty M N and Flynn P J 1999 Data clustering: A review. *ACM Computing Surveys* 31(3): 264–323

Johnson R 2005 J2EE development frameworks. *Computer* 38(1): 107–110

Johnson R, Hoeller J, Arendsen A and Thomas R 2009 Professional Java Development with the Spring Framework. Wiley.com

Ketkar N, Holder B and Cook D 2005 Subdue: Compression-based frequent pattern discovery in graph data. In: 1st International workshop on open source data mining: Frequent pattern mining implementations, ACM, pp 71–76

Krishna V, Suri N and Athithan G 2011 A comparative survey of algorithms for frequent subgraph discovery. *Curr. Sci.* 100(2): 190

Krishna V, Suri N N R R and Athithan G 2012 MuGRAM: An approach for multi-labelled graph matching. In: *IEEE International Conference on Recent Advances in Computing and Software Systems (RACSS)*, 19–26

Mordani R 2009 JSR 250: *Common annotations for the Java platform*. Sun Mircrosystems Inc

Mularien P 2010 *Spring Security 3*. Packt Publishing

Nowell D L and Kleinberg J 2003 The Link Prediction Problem for Social Networks. In: *12th Annual ACM International Conference on Information and Knowledge Management (CIKM)*, 556–559

O'Madadhainr J, Scott W and Boey Y 2003 The Jung (Java Universal Network/graph) framework. University of California, Irwine, California

Prasanna D R 2009 *Dependency Injection*. Manning Publications Co., Greenwich, CT

Rexer K 2011 5th Annual Data Miner Survey-2011 Survey Summary Report. Rexer Analytics

SAS 2013 SAS: Business Analytics and Business Intelligence Software. http://www.sas.com

Scott J 2012 *Social network analysis*. SAGE Publications Limited, London

Statistics IS 2008 17.0 SPSS Inc., Chicago, IL

StatSoft I 2001 *Statistica: Electronic statistics textbook*. http://www.statsoft.com/textbook/

Xu R and Wunsch D 2005 Survey of clustering algorithms. *IEEE Transactions on Neural Networks* 16(3): 645–678

Zhao Y 2012 *R and data mining: Examples and case studies*. Access Online via Elsevier