

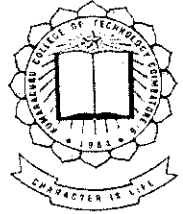
P-3111



# **RSS FEEDS FOR SOA SUITES**

**A PROJECT REPORT**

*Submitted by*



**VIGNESH.R**  
**PANNEER SELVAM.R**

**71206104055**  
**71206104031**

*In partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY**

**COIMBATORE - 641006.**



**ANNA UNIVERSITY: CHENNAI 600 025**

**MAY 2010**

**ANNA UNIVERSITY: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project “**RSS FEEDS FOR SOA SUITES**” is the bonafide work of

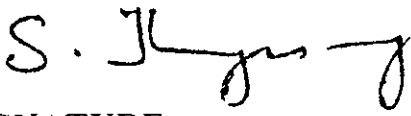
**VIGNESH.R**

**71206104055**

**PANNEER SELVAM.R**

**71206104031**

who carried out the project work under my supervision during the academic year 2009-2010.



**SIGNATURE**

**Dr.S.Thangasamy**

**DEAN**

Computer Science and Engineering,  
Kumaraguru college of Technology,  
Chinnavedampatti,  
Coimbatore-641006.



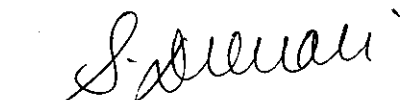
**SIGNATURE**

**Mr.K. Sivan Arul Selvan**

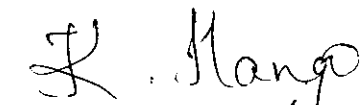
**PROJECT GUIDE**

Senior Lecturer,  
Computer Science and Engineering,  
Kumaraguru College of Technology,  
Chinnavedampatti,  
Coimbatore-641006.

The candidates with University Register Nos. 71206104055, 71206104031 were examined by us in the project viva-voice examination held on 15-04-2010.



**INTERNAL EXAMINER**



**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made this possible and whose constant guidance and encouragement crowns all efforts with success.

We express our profound gratitude to **Dr.S.Ramachandran**, Principal, Kumaraguru College of Technology, Coimbatore, for permitting us to undergo a project work.

We are greatly indebted to **Dr.S.Thangasamy**, Professor and Head of the Department Of Computer Science and Engineering, for the immense support he has provided us throughout our project.

We extend our sincere thanks to our project coordinator **Mrs. P.Devaki**, Associate Professor, Department of Computer Science and Engineering, for her constant support and encouragement.

We would like to express our heartfelt thanks to our guide, **Mr. K.Sivan Arul Selvan**, Senior Lecturer, Department of Computer Science and Engineering, who gave the initial idea and guidance to carry out every further step and for her everlasting counseling and untiring help throughout our project.

We sincerely thank our class advisor, **Ms. R.Kalaiselvi**, Senior Lecturer, Department of Computer Science and Engineering, for her unconditional support and motivation in helping us in completing the project.

We would also like to thank all the faculty members and lab technicians of Department of Computer Science and Engineering for their valuable guidance, support and encouragement during the course of project work.

## DECLARATION

We,

**R. VIGNESH**

**71206104055**

**R. PANNEER SELVAM**

**71206104031**

Declare that the project entitled “**RSS FEEDS FOR SOA SUITES**” submitted in partial fulfillment to Anna University, as the project work of Bachelor of Engineering ( Computer Science) degree, is a record of original work done by us under the supervision and guidance of Mr. K. Sivan Arul Selvan , Senior Lecturer, Department of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore.

Place: Coimbatore

Date: 15.04.2010

*R. Panneeselvam*

*R. Vignesh*

<b>CHAPTER</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	1
1	<b>INTRODUCTION</b>	2
2	<b>SYSTEM ANALYSIS</b>	3
	2.1 Existing system	3
	2.2 Proposed system	3
3	<b>REQUIREMENTS SPECIFICATION</b>	4
	3.1 Introduction	4
	3.2 Hardware and Software specification	5
	3.3 Tools and Technologies Used	5
	3.3.1 Java	6
	3.3.2 JSP	7
	3.3.3 Servlets	7
	3.3.4 Hibernate	8
	3.3.5 Oracle 10g	8
	3.3.6 Rome	9
	3.3.7 RSS Feeds	9
4	<b>SYSTEM DESIGN</b>	10
	4.1 Architecture	10
	4.2 Data Flow Diagram	
	4.2.1 Level 0 DFD	11

4.2.2	Level 1 DFD	12
4.2.3	Level 2 DFD AddSubscription	13
4.2.3	Level 2 DFD Delete	14
4.3	Use Case Diagram	
4.3.1	AddSubscription Usecase	15
4.3.2	Delete Usecase	16
4.4	Class Diagram	
4.4.1	FeedGenerator	17
4.4.2	FeedReader	18
4.5	Sequence Diagram	19
<b>5</b>	<b>SYSTEM DESIGN - DETAILED</b>	
5.1	Modules	20
5.2	Module explanation	20
<b>6</b>	<b>CODING AND TESTING</b>	
6.1	Coding	22
6.2	Coding standards	22
6.2.1	Naming conventions	23
6.2.2	Value conventions	24
6.2.3	Script writing and commenting standard	24
6.3	Testing	25
6.3.1	Unit Testing	25

6.3.2 Functional Tests	27
6.3.2.1 Performance Test	27
6.3.2.2 Stress test	27
6.3.3 Integration Testing	28

## **7 CONCLUSION AND FUTURE ENHANCEMENTS**

7.1 Conclusion and Future enhancements	30
--	----

## **8 APPENDIX**

8.1 Sample Source code	31
8.2 Snap Shots	50

## **9 REFERENCES**

56

## **ABSTRACT**

This project proposes a method for the enterprises developing Service Oriented Architecture (SOA) suites to achieve customer retention, and an increase in customer loyalty. Customer loyalty is achieved by communicating the details of the release of their latest versions immediately and provokes the customer to use the latest version.

This project uses Really Simple Syndication (RSS) feeds to communicate with the customers. Syndication is an act by which a website material is made available over multiple sites. In this project a feed reader is designed to read the feeds generated by the companies upon the update of their database, of the release of the new version of the suite. A customer may use many number of suites of any company. He may subscribe to many company's feeds he wish to have and get updates about their new versions immediately. The feeds are aggregated and displayed in the reader. As there are different types of feeds like RSS 2.0, Atom the reader is capable of aggregating all the feeds and displaying the feeds to the user in a way that is convenient to the user. Therefore, the customer retention can be ensured by making the user subscribe to company's feeds.



## INTRODUCTION

The key to a successful business is a steady customer base. After all, successful businesses typically see 80 percent of their business come from 20 percent of their customers. Too many businesses neglect this loyal customer base in pursuit of new customers. However, since the cost to attract new customers is significantly more than to maintain your relationship with existing ones, your efforts toward building customer loyalty will certainly payoff. The major task of retaining the customer base is done by maintaining a good communication with the customers. We explored various options for easy communication and finally went for RSS feeds.

RSS (Really Simple Syndication) is a format for delivering regularly changing web content. Many news-related sites, weblogs and other online publishers syndicate their content as an RSS Feed to whoever wants it. RSS solves a problem for people who regularly use the web. It allows you to easily stay informed by retrieving the latest content from the sites you are interested in. You save time by not needing to visit each site individually. You ensure your privacy, by not needing to join each site's email newsletter. The number of sites offering RSS feeds is growing rapidly and includes big names like Yahoo News. So, we generate RSS feeds for SOA suites developing companies when they update their database and the customers of the company can view the update immediately. It is profitable in the customer side also because he can subscribe to the feeds of all the products he is using instead of visiting each company's website for updates.

## **CHAPTER 2**

### **SYSTEM ANALYSIS**

#### **2.1 EXISTING SYSTEM**

Retaining the customer base, achieving customer loyalty are the major tasks of the enterprises nowadays. Browsing for updates is also a timing consuming task for the customers. The customers may utilize many products of many companies. So it is difficult for them to keep track of the updates of every product. The first most important thing is maintaining a good communication with the customers. The customer communication must be organized, relevant and flexible. The Service Oriented Architecture (SOA) suites developing companies inform their customers about their latest release by collecting the contact details of its customers and informing everyone. Some customers may not be interested in providing their personal contact details to the companies.

#### **2.2 PROPOSED SYSTEM**

We propose a system in which the enterprises developing Service Oriented Architecture (SOA) suites advice the customers to subscribe to the feeds. The feeds are generated by the company on a time basis or upon the update of the database based on the wish of the company. Whenever the user comes online the feed reader program in the client side searches for any new feeds on the subscribed Uniform Resource Locator (URL). If so it gets the feeds and store the feed details in the database and the user can view it whenever he is free.

## **CHAPTER 3**

### **REQUIREMENT SPECIFICATIONS**

#### **3.1 INTRODUCTION**

##### **3.1.1 Purpose**

The purpose of this project is to achieve customer retention, loyalty for the enterprise and easy viewing of the recent updates of the suites for the customers and to provide an easy method for the customers to keep track of the latest releases of the product they are using. This project benefits both the customers and the enterprises in saving time by making communication between them so easily.

##### **3.1.2 Project Scope**

The customers do not need to provide the personal information for contact and do not need to browse for knowing any recent updates in the suites they are using. The enterprise doesn't have to spend time in contacting the customers to inform the customer about the release of their latest version. Instead it is enough to advice the customers to subscribe for the feeds they generate and make them do it. This may help the enterprise to retain their customers as customer retention is the most important and competitive work in the corporate world and this may help the customers in knowing the latest releases of various products they are using.

## **3.2 HARDWARE AND SOFTWARE SPECIFICATION**

### **3.2.1 HARDWARE REQUIREMENTS**

- Processor : Pentium III and Above
- RAM : 512MB and Above
- Hard Disk : 10GB and Above

### **3.2.2 SOFTWARE REQUIREMENTS**

- Operating System : Windows XP/Vista/Windows 7
- Documentation Tool : Microsoft word 2000

## **3.3 TOOLS AND TECHNOLOGIES USED**

- **Java** - Used for writing the classes
- **JSP** - Used to design the user interface
- **Servlets** - Used to generate the feeds on the sever side
- **Hibernate** - Object relational mapping concept
- **Oracle 10g** - Database used to store the tables
- **Rome** - Parsing tool included as a jar file in Java Build path
- **RSS feeds**

## TECHNOLOGIES USED

### 3.3.1 JAVA

It is a Platform Independent. Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. The language, initially called Oak (named after the oak trees outside Gosling's office), was intended to replace C++, although the feature set better resembles that of Objective C.

### WORKING OF JAVA

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data and functions.

When the interpreter executes a class, it looks for a particular method by the name of **main**, which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (similar to the `argv[]` of C), and is declared as a static method.

To output text from the program, we execute the **println** method of **System.out**, which is java's output stream. UNIX users will appreciate the theory behind such a stream, as it is actually standard output. For those who are instead used to the Wintel platform, it will write the string passed to it to the user's program.

### 3.3.2 JSP

Java Server Pages (JSP) is a Java technology that helps software developers serve dynamically generated web pages based on HTML, XML, or other document types. JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver an HTML or XML document. JSP syntax is a fluid mix of two basic content forms: *scriptlet elements* and *markup*. Markup is typically standard HTML or XML, while scriptlet elements are delimited blocks of Java code which may be intermixed with the markup. When the page is requested the Java code is executed and its output is added, in situ, with the surrounding markup to create the final page. Because Java is a compiled language, not a scripting language, JSP pages must be compiled to Java bytecode classes before they can be executed, but such compilation generally only occurs once each time a change to the source JSP file occurs.

### 3.3.3 SERVLETS:

Servlets are modules of Java code that run in a server application (hence the name "Servlets", similar to "Applets" on the client side) to answer client requests. Servlets are not tied to a specific client-server protocol but they are most commonly used with HTTP and the word "Servlet" is often used in the meaning of "HTTP Servlet".

Servlets make use of the Java standard extension classes in the packages `javax.servlet` (the basic Servlet framework) and `javax.servlet.http`

(extensions of the Servlet framework for Servlets that answer HTTP requests). Since Servlets are written in the highly portable Java language and follow a standard framework, they provide a means to create sophisticated server extensions in a server and operating system independent way.

### **3.3.4 HIBERNATE:**

Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions. Mapping Java classes to database tables is accomplished through the configuration of an XML file or by using Java Annotation. When using an XML file, Hibernate can generate skeletal source code for the persistence classes. This is unnecessary when annotation is used. Hibernate can use the XML file or the annotation to maintain the database schema.

### **3.3.5 ORACLE10g :**

The **Oracle Database** (commonly referred to as *Oracle RDBMS* or simply as *Oracle*) is a relational database management system (RDBMS) produced and marketed by Oracle Corporation. As of 2009, Oracle remains a major presence in database computing. As of the Oracle Database 10g release, Oracle Corporation seems to have started to make an effort to standardize all current versions of its major products using the "10g" label.

### **3.3.6 ROME:**

Rome is an open source Java API for reading and publishing RSS feeds in a relatively format-neutral way. Originally developed by Sun and now a Java.net project, Rome is designed to provide a level of abstraction to mask the subtle differences between the various formats. This lets you concentrate on publishing and/or processing content, rather than wrestling with the particularities of each format. Rome was designed from the onset to be easy to use, flexible, and to support all existing RSS flavors.

### **3.3.7 RSS FEEDS:**

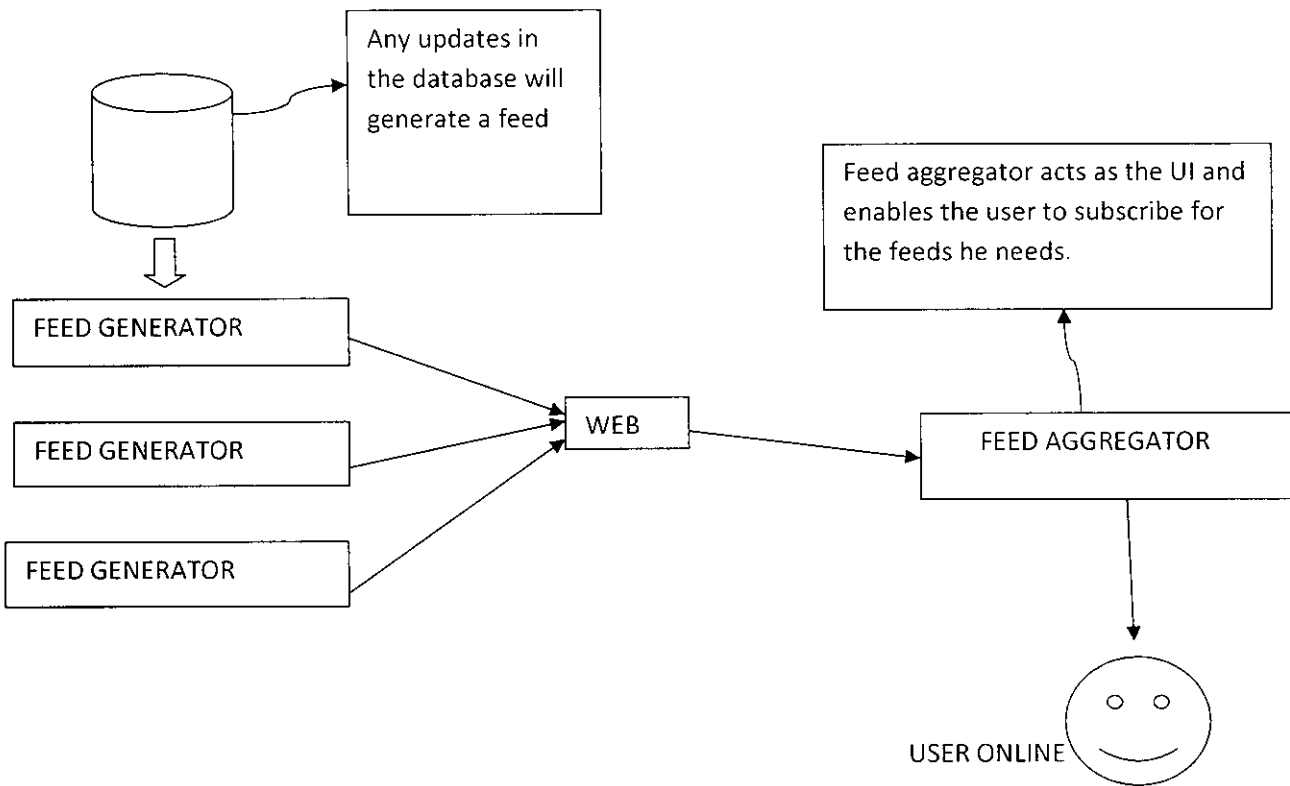
RSS (Really Simple Syndication) is an established way of publishing short snippets of information, such as news headlines, project releases, or blog entries. Modern browsers such as Firefox, and more recently Internet Explorer 7, and mail clients such as Thunderbird recognize and support RSS feeds; not to mention the a large number of dedicated RSS readers (also known as aggregators) out there.

RSS feeds aren't just for end-users, though. A variety of application scenarios could require you to read, publish, or process RSS feeds from within your code. Your application could need to publish information through a set of RSS feeds, or need to read, and possibly manipulate, RSS data from another source. For example, some applications use RSS feeds to inform users of changes to the application database that could affect them.



# CHAPTER 4

## SYSTEM DESIGN



**Fig 4.1 Structure Diagram**

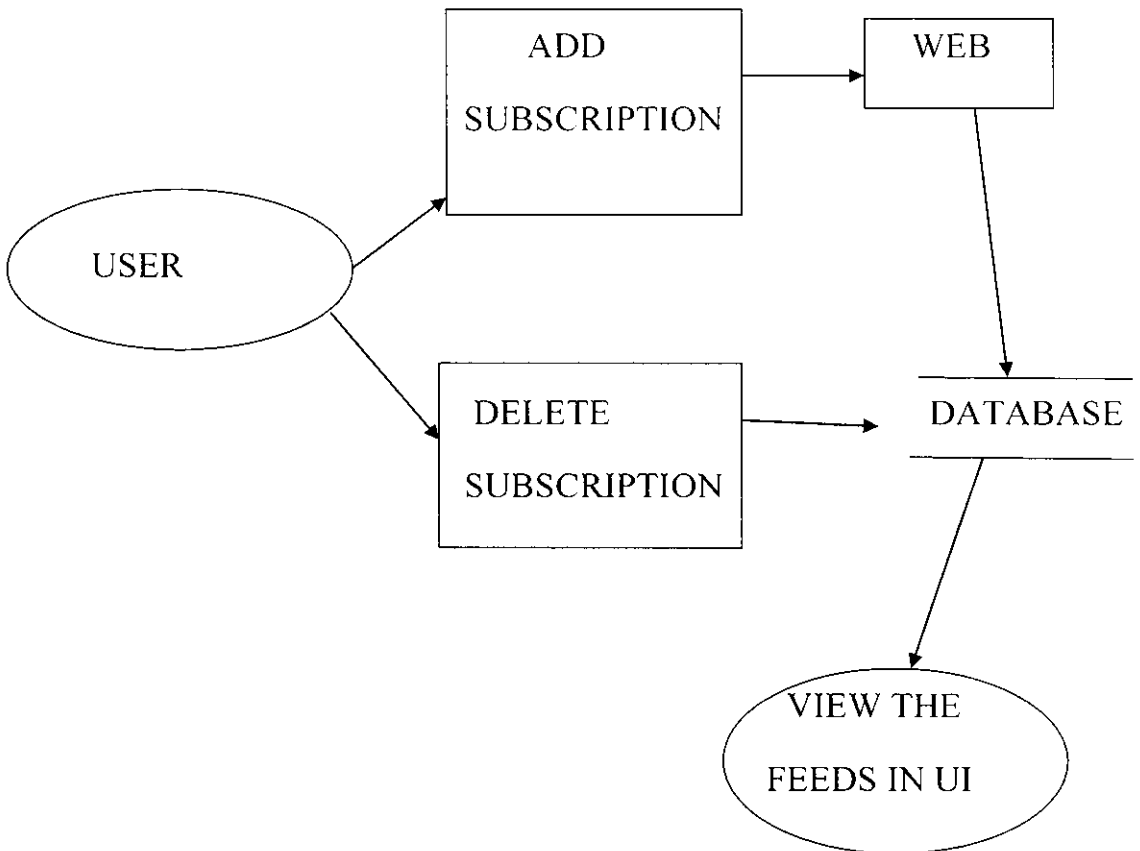


Fig no: 4.2.1 LEVEL 0 DFD



P-3/11

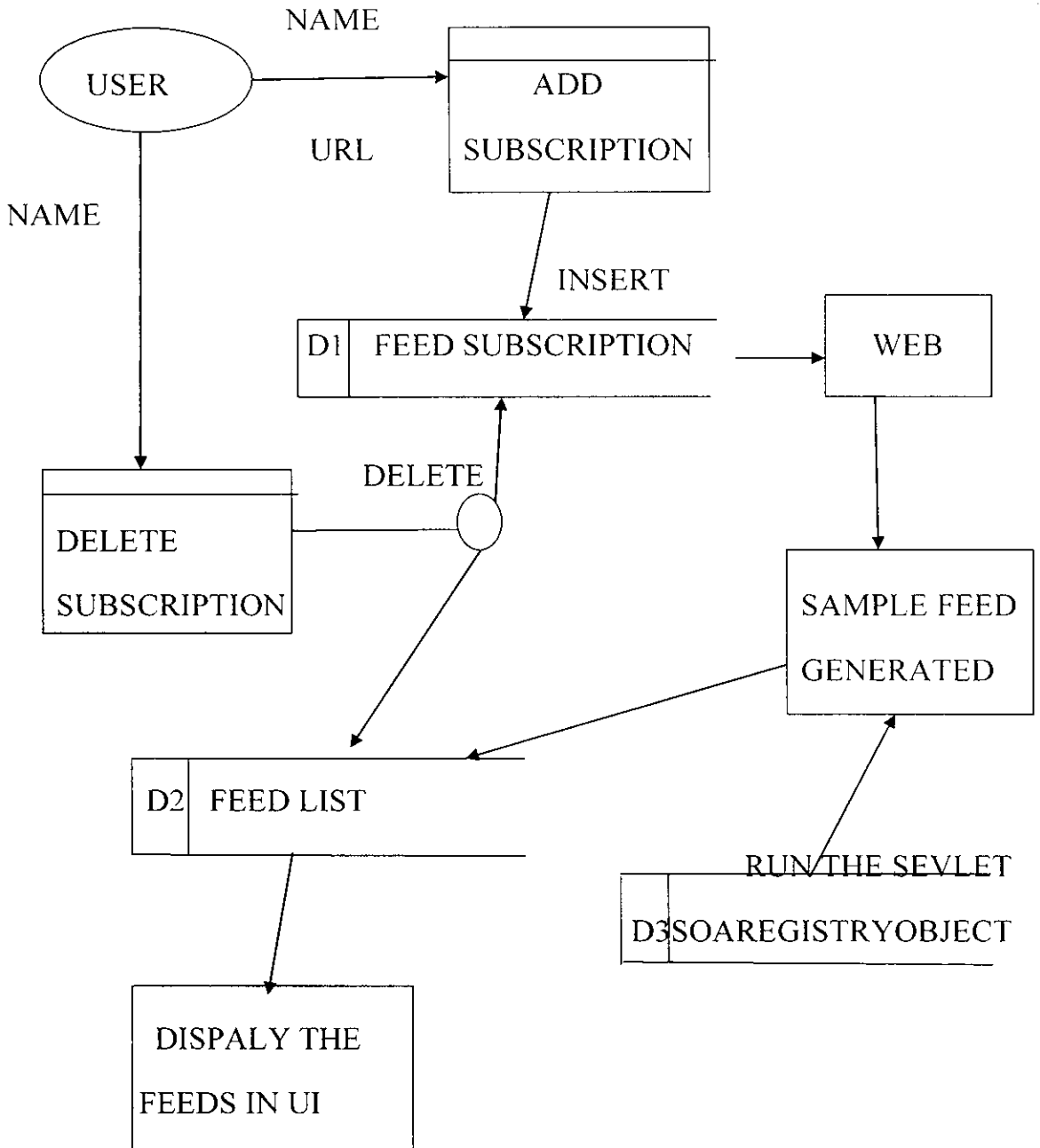


Fig no: 4.2.2 LEVEL 1 DFD

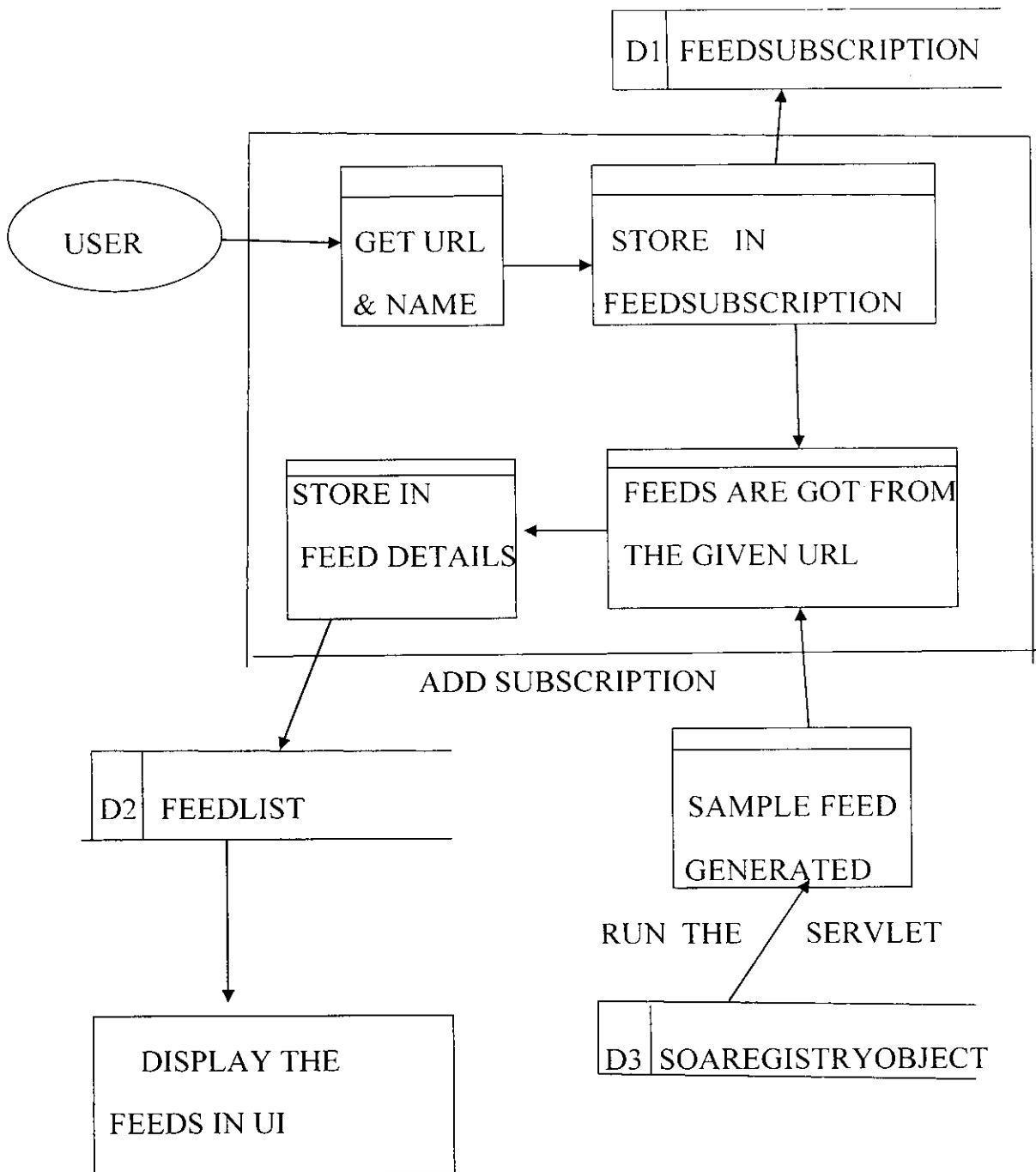


Fig no: 4.2.3 LEVEL 2 DFD ADD SUBSCRIPTION

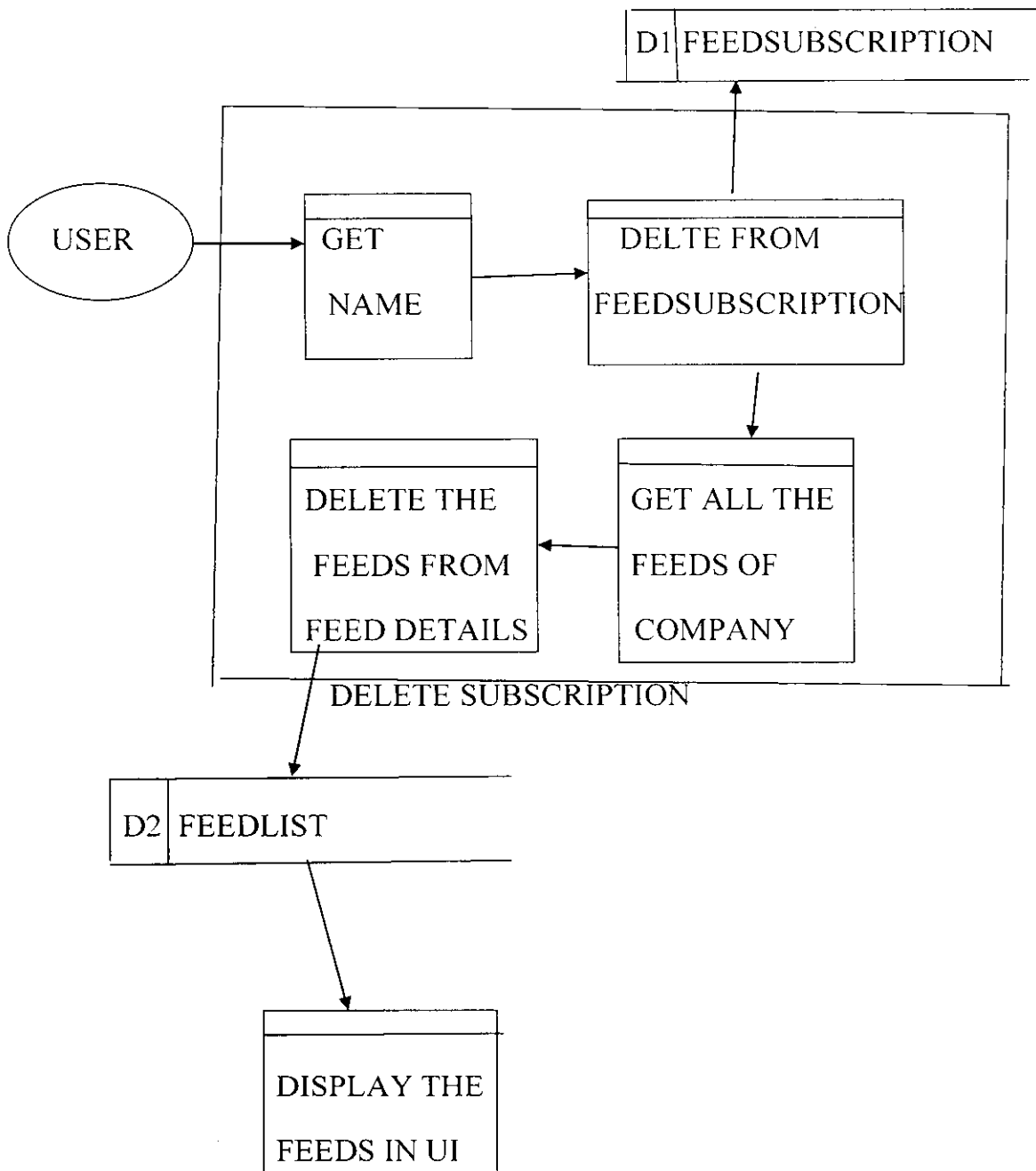


Fig no: 4.2.4 LEVEL 2 DFD DELETE

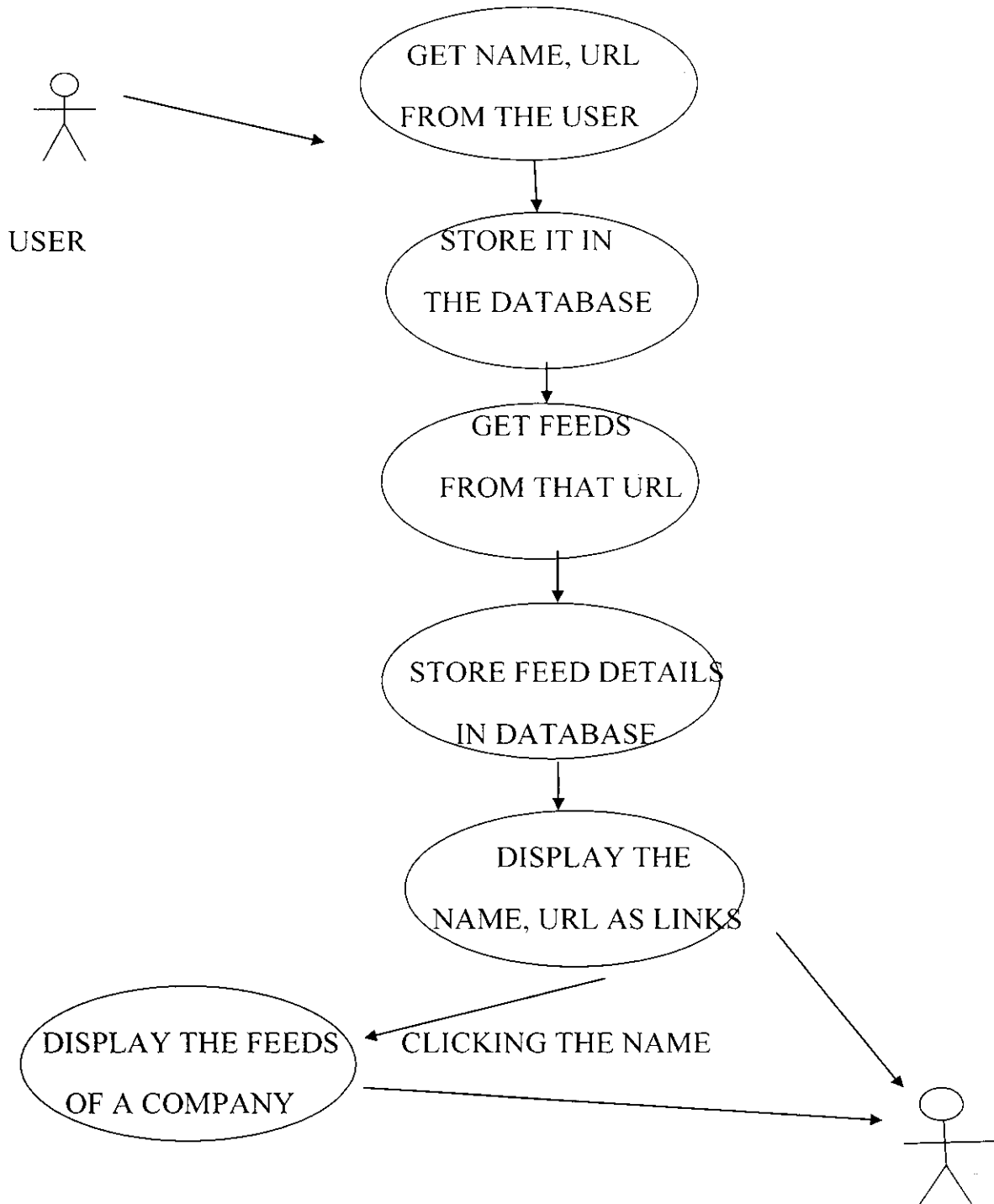


FIG 4.3.1 AddSubscription Usecase

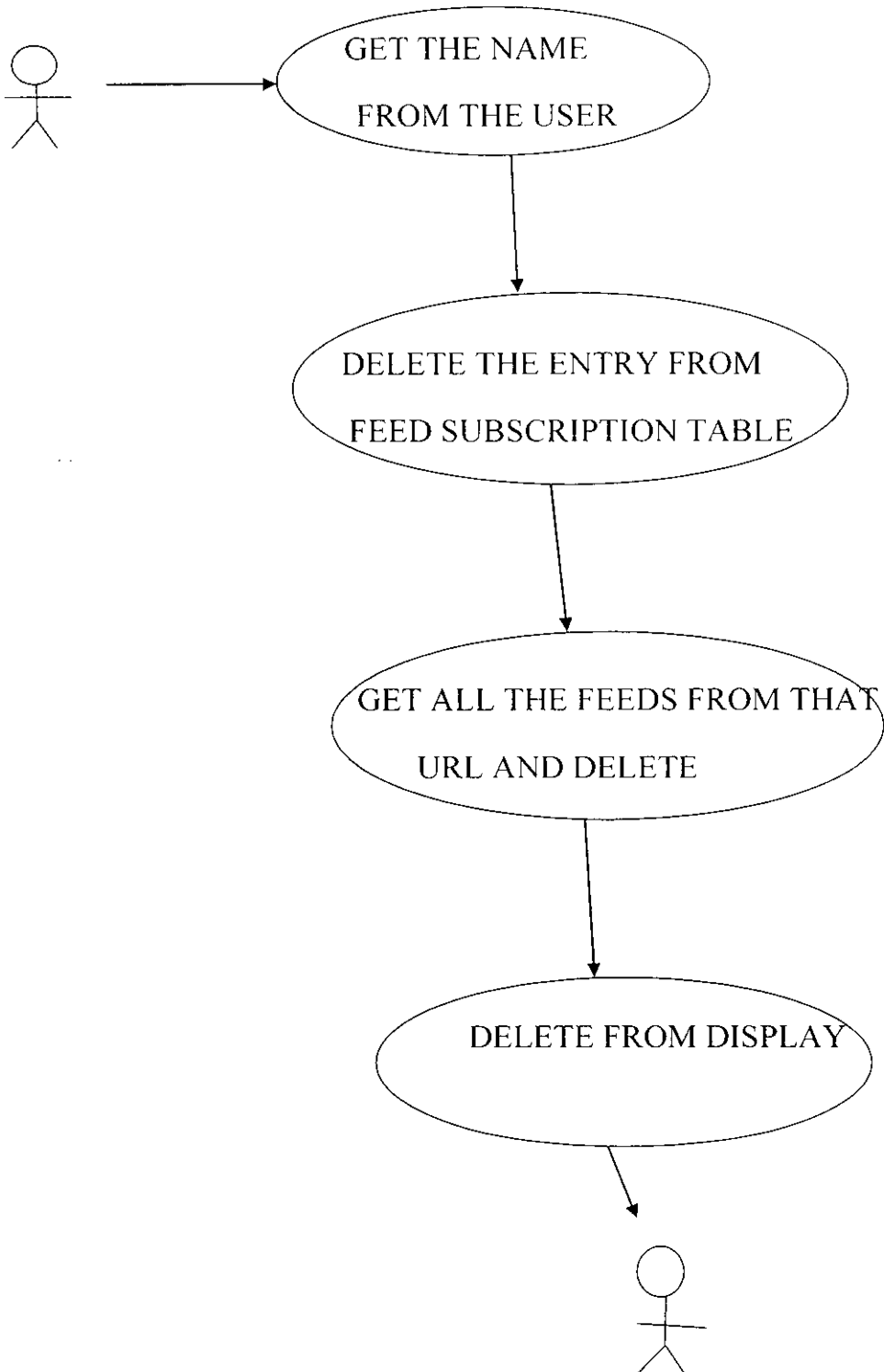


Fig no : 4.3.2 Delete Usecase

## CLASS DIAGRAM:

PACKAGE : syndication.feed.generator

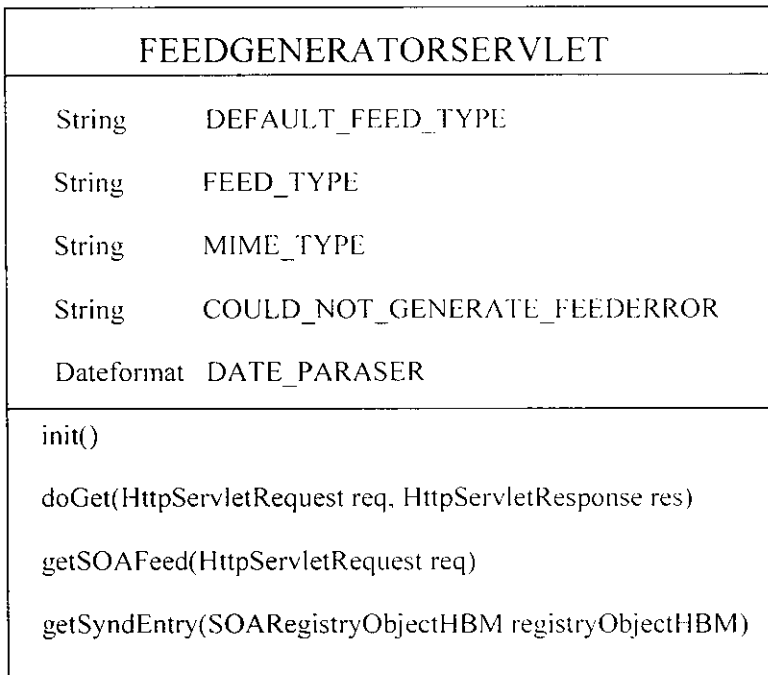


Fig no:4.4.1 FeedGenerator



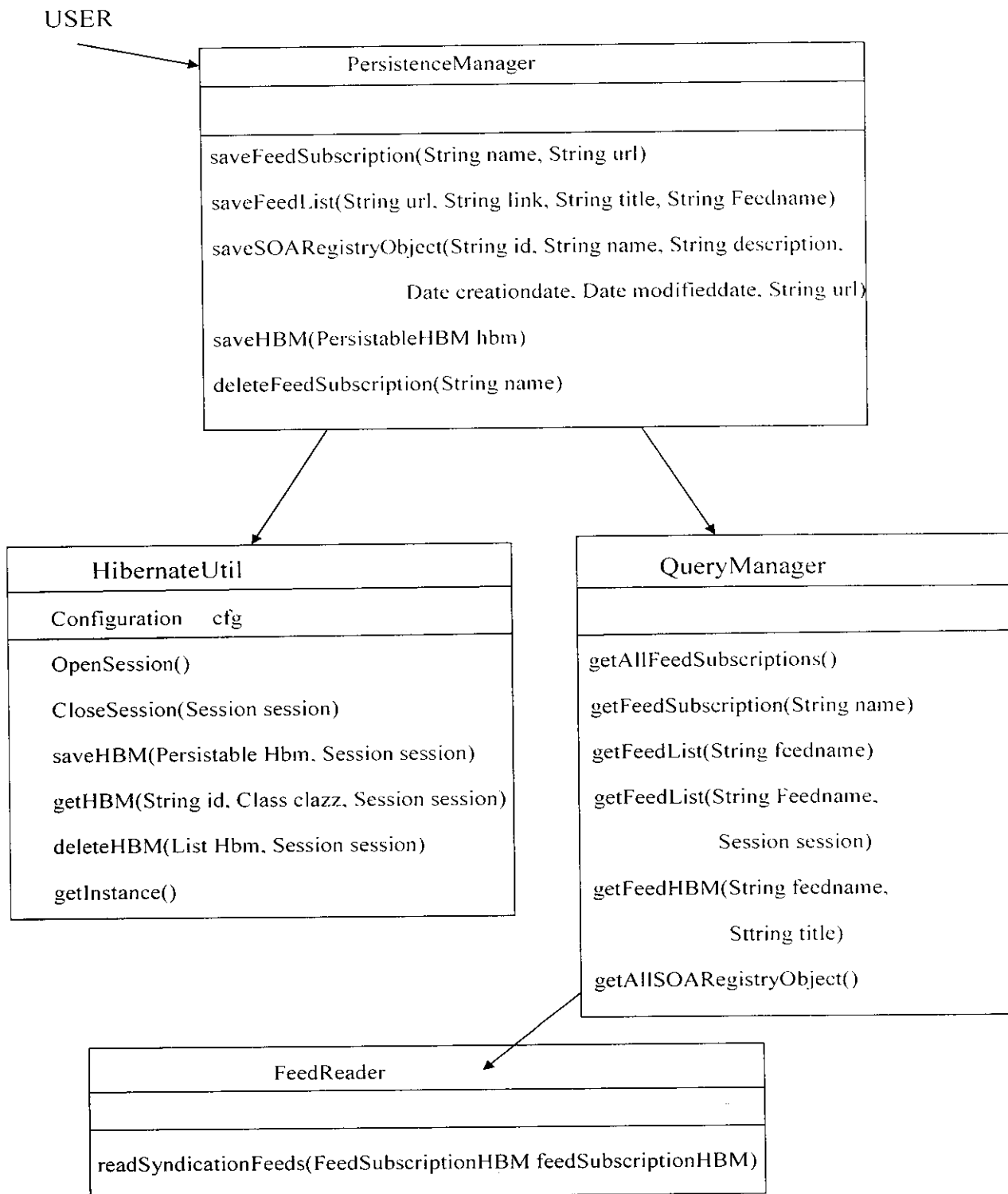


Fig no: 4.4.2 FeedReader

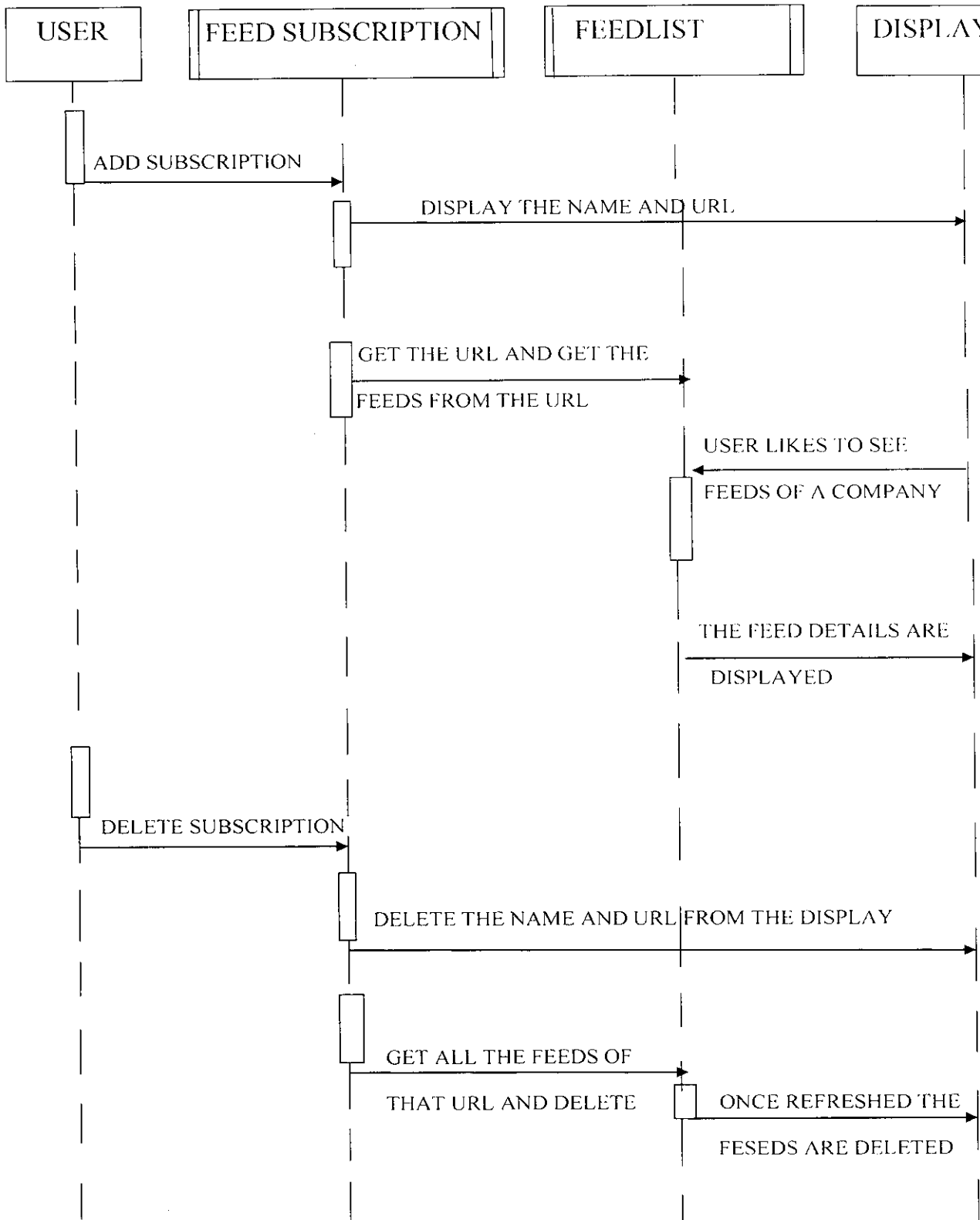


Fig no: 4.5 SEQUENCE DIAGRAM

## CHAPTER 5

### SYSTEM DESIGN

#### 5.1 MODULES

- **AddSubscription**
- **DeleteSubscription**
- **View Feeds of a particular company**
- **Sample feed generation**

#### 5.2 MODULES EXPLANATION:

##### **AddSubscription:**

If the user wish to get the updates of a new product of a new company he can provide the name of the company providing the suite and the URL of the company, which produces the feeds. The name and the URL get stored in the database table FeedSubscription. When the user comes online the client side pulls the feed details and store the details in the database table FeedList. When the user wish to view the feeds he can view the feeds in the feed reader designed.

##### **DeleteSubscription:**

If the user likes to delete a subscription he is allowed to do so by entering the name of the company and click delete. The name and URL get deleted from the FeedSubscription table and all the feeds of that

company are deleted from the FeedList table. When the UI page is refreshed the feeds of the company will not display.

### **View Feeds of a particular company:**

If the user like to see only the feeds of a particular company, he can do so by clicking the name of the company in the UI. The function in the QueryManager class is invoked and only the feeds of that company are displayed.

### **Sample feed generation:**

A sample feed is generated with the contents of the table SOARegistryObject. The feeds are generated once when the FeedGenerator servlet is run. This can be enhanced by the enterprise by making the servlet run after each update of the database or by running the servlet on a timely basis. The feeds should not irritate the customer by providing the same information again and again.

## **CHAPTER 6**

### **CODING AND TESTING**

#### **6.1 CODING**

Once the design aspect of the system is finalized, the system enters into the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required to easily screw into the system.

#### **6.2 CODING STANDARDS**

Coding standards are guidelines to programming that focus on the physical structure and appearance of the program. They make the code easier to read, understand, and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standards needed to achieve the above-mentioned objectives are as follows:

Program should be simple, clear, and easy to understand.

Naming conventions

Value conventions

Script and comment procedure

## 6.2.1 NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

### **Package names:**

The prefix of a unique package name is always written in all-lowercase ASCII letters and should be one of the top-level domain names, currently “syndication.feed”. Subsequent components of the package name vary according to the purpose of the work they do.

### **Class names:**

Class names are problem domain equivalence and begin with capital letter and have mixed cases. The HBM classes have the same name as the table name to which it is mapped to. The other classes are named using **Letter-case separated words**.

### **Methods:**

Methods should be verbs, in mixed case with the first lowercase, with the first letter of each internal word capitalized.

## **6.2.2 VALUE CONVENTIONS**

Value conventions ensure values for variable at any point of time. This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

## **6.2.3 SCRIPT WRITING AND COMMENTING STANDARD**

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

## **6.3 TESTING:**

### **INTRODUCTION:**

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding.

#### **6.3.1 UNIT TESTING**

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing. Unit testing has been performed on the modules. The syntax and logical errors have been



corrected then and there. All the syntax errors have been rectified during compilation. The output has been tested with the manual input. The data is verified whether it is stored correctly in the appropriate fields in the database.

The test case includes the following:

- The test for the database storage, retrieval using Hibernate concept.
- The test for mouse, button operations in the User interface.

Design the set of tests:

Test Set 1: For Database storage, retrieval using Hibernate

1. Are all the objects stored in the respective fields?
2. Can the required data be retrieved from the database?

Test Set 2: Mouse operations:

1. Are all functions properly addressable by the mouse?
2. Does each function perform as advertised?
3. Do multiple or incorrect mouse picks within the window cause unexpected side effects?

Test Set 1(Results):

1. Yes, all the objects are stored correctly in the database.
2. Yes, the required data can be retrieved from the database.

Test Set 2 (Results) :

1. Yes, all are addressable by the mouse.

2. Yes, function perform as advertised.
3. No, incorrect mouse picks within the window do not generate unexpected side effects.

## **6.3.2 FUNCTIONAL TESTS**

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test

### **6.3.2.1 PERFORMANCE TEST**

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

### **6.3.2.2 STRESS TEST**

Stress testing refers to the tests that put a greater emphasis on robustness, availability and error handling under a heavy load, rather than on what would be considered correct behavior under normal circumstances. The

goals of such tests may be to ensure the software does not crash in conditions of insufficient computational resources such as memory or disk space, unusually high concurrency.

The feeds were subscribed concurrently and the maximum number of subscribed feeds that the database, user interface can support are verified.

### **6.3.3 INTEGRATION TESTING**

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product

development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

The PersistenceManager and the QueryManager classes are integrated and is checked via the FeedRead program. The HibernateUtil is used to open, close, start a new session and so this class is essential for database storage, retrieval.

## **CHAPTER 7**

### **CONCLUSION AND FUTURE ENHANCEMENT**

In this project we have provided an easy way for the enterprises and its customers to interact with each other. This can be enhanced by integrating the reader with the default browser in the system so that the feeds can be viewed more conveniently. This can also be enhanced by integrating the reader with other commercial readers so that the user does not have to have separate readers for news, entertainment and for this purpose. The reader may also be included in any websites so that the online reader may always made to display the recent feeds of the company that are willing to advertise.

## CHAPTER 8

### APPENDIX

#### 8.1 SAMPLE SOURCE CODE:

PACKAGE : syndication.feed.generator

FeedGenerator servlet :

```
package syndication.feed.generator;
```

```
/**
```

```
 * This servlet is used to generate the syndication feeds
```

```
 * for the database
```

```
 *
```

```
 * @author vignesh,panneer
```

```
 *
```

```
 */
```

```
public class FeedGeneratorServlet extends HttpServlet {
```

```
    private static final String DEFAULT_FEED_TYPE = "default.feed.type";
```

```
    private static final String FEED_TYPE = "type";
```

```
private static final String MIME_TYPE = "application/xml; charset=UTF-8";
```

```
private static final String COULD_NOT_GENERATE_FEED_ERROR = "Could not generate feed";
```

```
private static final DateFormat DATE_PARSER = new SimpleDateFormat("yyyy-MM-dd");
```

```
private String _defaultFeedType;
```

```
public void init() {
```

```
    defaultFeedType =  
    getServletConfig().getInitParameter(DEFAULT_FEED_TYPE);
```

```
    _defaultFeedType = (_defaultFeedType!=null) ? _defaultFeedType :  
    "atom_0.3";
```

```
}
```

```
public void doGet(HttpServletRequest req, HttpServletResponse res)  
throws IOException {
```

```
    try {
```

```
        SyndFeed feed = getSOAFeed(req);
```

```
        String feedType = req.getParameter(FEED_TYPE);
```

```
        feedType = (feedType!=null) ? feedType : _defaultFeedType;
```

```

        feed.setFeedType(feedType);

        res.setContentType(MIME_TYPE);

        SyndFeedOutput output = new SyndFeedOutput();

        output.output(feed,res.getWriter());
    }

    catch (Exception ex) {

        String msg = COULD_NOT_GENERATE_FEED_ERROR;

        log(msg,ex);

res.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR,
msg);

    }

}

private SyndFeed getSOAFeed(HttpServletRequest req) throws
Exception {

    QueryManager queryManager = new QueryManager();

    List entries = new ArrayList();

List<SOARegistryObjectHBM> soaRegObjs =
queryManager.getAllSOARegistryObjects();

    for (SOARegistryObjectHBM registryObjectHBM : soaRegObjs) {

```



```

        SyndEntry entry = getSyndEntry(registryObjectHBM);
        entries.add(entry);
    }

    SyndFeed feed = new SyndFeedImpl();

    feed.setEntries(entries);

    return feed;
}

private SyndEntry getSyndEntry(SOARRegistryObjectHBM
registryObjectHBM) throws Exception {

    SyndEntry entry = new SyndEntryImpl();

    entry.setTitle(registryObjectHBM.getName());

    entry.setLink(registryObjectHBM.getURL());

    entry.setUri(registryObjectHBM.getId());

    entry.setPublishedDate(registryObjectHBM.getCreationDate());

    return entry;
}

```

PACKAGE : syndication.feed.persistence

CLASS : PersistenceManager

/\*\*

\* This class is used to save/update/delete HBMs

\*

\* **@author** vignesh,panneer

\*

\*/

**public class** PersistenceManager {

**public void** saveFeedSubscription(String name, String url) **throws**

Exception {

    FeedSubscriptionHBM subHBM = **new** FeedSubscriptionHBM();

    subHBM.setName(name);

    subHBM.setURL(url);

    System.out.println("Persistence Manager: " + subHBM.getName());

    saveHBM(subHBM);

        QueryManager queryManager = **new** QueryManager();

        FeedReader feedReader = **new** FeedReader();

        List<SyndEntry> entries =

        feedReader.readSyndicationFeeds(subHBM);

**for** (SyndEntry syndEntry : entries) {

            String uri = syndEntry.getUri();

            String link = syndEntry.getLink();

            String title = syndEntry.getTitle();

```
saveFeedList(uri, link, title, name, queryManager);  
    }  
}
```

```
public void saveFeedList(String uri, String link, String title, String  
feedName, QueryManager queryManager) throws Exception {
```

```
    FeedListHBM listHBM = new FeedListHBM();  
    listHBM.setURI(uri);  
    listHBM.setLink(link);  
    listHBM.setTitle(title);  
    listHBM.setFeedName(feedName);  
    saveHBM(listHBM);  
}
```

```
public void saveSOARegistryObject(String id, String name, String  
description, Date creationDate, Date modifiedDate, String url) throws  
Exception {
```

```
SOARegistryObjectHBM regObjHBM = new SOARegistryObjectHBM();  
    regObjHBM.setId(id);  
    regObjHBM.setName(name);  
    regObjHBM.setDescription(description);  
    regObjHBM.setCreationDate(creationDate);  
    regObjHBM.setModifiedDate(modifiedDate);  
    regObjHBM.setURL(url);  
    saveHBM(regObjHBM);
```

```
}
```

```
private void saveHBM(Persistable hbm) throws Exception {  
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();  
    Session session = hibernateUtil.openSession();  
    hibernateUtil.saveHBM(hbm, session);  
    hibernateUtil.closeSession(session);  
}
```

```
}
```

```
public void deleteFeedSubscription(String name) throws Exception {  
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();  
    Session session = hibernateUtil.openSession();
```

```
        FeedSubscriptionHBM subHBM = (FeedSubscriptionHBM)  
hibernateUtil.getHBM(name,FeedSubscriptionHBM.class, session);
```

```
        System.out.println("Name: " + name);
```

```
        System.out.println("subscripion: " + subHBM);
```

```
        QueryManager manager = new QueryManager();
```

```
        List feeds = manager.getFeedList(name, session);
```

```
        System.out.println("List: " + feeds);
```

```
        //adding subscription to the feeds to delete in single go.
```

```
        feeds.add(subHBM);
```

```
        //delete the hbms
```

```
        hibernateUtil.deleteHBM(feeds, session);
```

```
        hibernateUtil.closeSession(session);
    }
}
```

PACKAGE : syndication.feed.persistencemodel

FeedListHBM

**package** syndication.feed.persistence.model;

```
/**
 * Feed List HBM is used to save feed details for a particular website.
 * It collects all the feeds that are generated from the website and
 * stores it in the database
 *
 * @author vignesh,panneer
 *
 */
```

**public class** FeedListHBM **implements** Persistable {

String URI;

String link;

String title;

String feedName;

```
public String getURI() {  
    return URI;  
}
```

```
public void setURI(String uri) {  
    URI = uri;  
}
```

```
public String getLink() {  
    return link;  
}
```

```
public void setLink(String link) {  
    this.link = link;  
}
```

```
public String getTitle() {  
    return title;  
}
```

```
public void setTitle(String title) {  
    this.title = title;  
}
```

```
public String getFeedName() {  
    return feedName;  
}
```

```

    }

    public void setFeedName(String feedName) {
        this.feedName = feedName;
    }

    public String getId() {
        return getTitle();
    }
}

```

CLASS : FeedSubscriptionHBM

```

package syndication.feed.persistence.model;

```

```

/**

```

```

 * Feed Subscription HBM is used to save subscription for the feed.

```

```

 * It stores the name and URL of the website

```

```

 *

```

```

 * @author vignesh,panneer

```

```

 *

```

```

 */

```

```

public class FeedSubscriptionHBM implements Persistable {

```

```

    String name;

```

```

    String URL;

```

```
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getURL() {  
    return URL;  
}  
  
public void setURL(String url) {  
    this.URL = url;  
}  
  
public String getId() {  
    return getName();  
}  
}
```

CLASS: SoaRegistryObjectHBM

**package** syndication.feed.persistence.model;



```
import java.util.Date;
```

```
/**
```

```
* This is used to store the registry object details of a SOA Registry Object
```

```
*
```

```
* @author vignesh,panneer
```

```
*
```

```
*/
```

```
public class SOARegistryObjectHBM implements Persistable {
```

```
    private String id;
```

```
    private String name;
```

```
    private String description;
```

```
    private Date creationDate;
```

```
    private Date modifiedDate;
```

```
    private String URL;
```

```
    public String getId() {
```

```
        return id;
```

```
    }
```

```
    public void setId(String id) {
```

```
        this.id = id;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public Date getCreationDate() {
        return creationDate;
    }

    public void setCreationDate(Date creationDate) {
        this.creationDate = creationDate;
    }

    public Date getModifiedDate() {
        return modifiedDate;
    }
}
```

```

public void setModifiedDate(Date modifiedDate) {
    this.modifiedDate = modifiedDate;
}

public String getURL() {
    return URL;
}

public void setURL(String url) {
    URL = url;
}
}

```

PACKAGE : syndication.feed.reader

FeedReader

/\*\*

\* This class is used to read the Syndication feeds from the given webSite

URL

\*

\* **@author** vignesh,panneer

\*

```

*/
public class FeedReader {

    public List<SyndEntry>
readSyndicationFeeds(FeedSubscriptionHBM feedSubscriptionHBM)
throws Exception {
        String url = feedSubscriptionHBM.getURL();

        URL feedsUrl = new URL(url);
        SyndFeedInput input = new SyndFeedInput();
        SyndFeed feed = input.build(new XmlReader(feedsUrl));

        return feed.getEntries();
    }
}

```

CLASS : QueryManager

```

/**
 * This class is used to query the HBM instances
 *
 * @author vignesh,panneer
 *
 */
public class QueryManager {

```

```

    public List<FeedSubscriptionHBM> getAllFeedSubscriptions()
throws Exception {
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();
    Session session = hibernateUtil.openSession();
    String query = "from
syndication.feed.persistence.model.FeedSubscriptionHBM feed ";

    Query hbmQuery = session.createQuery(query);

    return hbmQuery.list();
}

```

```

    public FeedSubscriptionHBM getFeedSubscription(String name)
throws Exception {
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();
    Session session = null;
    try {
    session = hibernateUtil.openSession();
return (FeedSubscriptionHBM) hibernateUtil.getHBM(name,
FeedSubscriptionHBM.class, session);
    } finally {
    hibernateUtil.closeSession(session);
    }
}

```

```

public List<FeedListHBM> getFeedList(String feedName) throws
Exception {
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();
    Session session = null;
    try {
        session = hibernateUtil.openSession();

        String query = "from
syndication.feed.persistence.model.FeedListHBM feed " +
"where feed.feedName='" + feedName + "'";
        Query hbmQuery = session.createQuery(query);
        return hbmQuery.list();
    } finally {
        hibernateUtil.closeSession(session);
    }
}

```

```

public List<FeedListHBM> getFeedList(String feedName, Session
session) throws Exception {
    String query = "from
syndication.feed.persistence.model.FeedListHBM feed " +
"where feed.feedName='" + feedName + "'";

    Query hbmQuery = session.createQuery(query);

    return hbmQuery.list();
}

```

```
}
```

```
    public FeedListHBM getFeedHBM(String feedName, String title)
throws Exception {
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();
    Session session = null;
    try {
        session = hibernateUtil.openSession();

        String query = "from
syndication.feed.persistence.model.FeedListHBM feed " +
                                "where feed.feedName='" +
feedName + "' and feed.title='" + title + "'";

        Query hbmQuery = session.createQuery(query);

        List<FeedListHBM> result = hbmQuery.list();
        if (result == null) {
            return null;
        }

        return result.get(0);
    } finally {
        hibernateUtil.closeSession(session);
    }
}
```

```
public List<SOARegistryObjectHBM> getAllSOARegistryObjects()
throws Exception {
    HibernateUtil hibernateUtil = HibernateUtil.getInstance();
    Session session = hibernateUtil.openSession();

    String query = "from
syndication.feed.persistence.model.SOARegistryObjectHBM regObj ";

    Query hbmQuery = session.createQuery(query);

    return hbmQuery.list();
}
}
```





## FEED DETAILS

### *vignesh* FEED Details

URI	Title	Link
442	vignesh	<a href="http://linkedin.com/vignesh">http://linkedin.com/vignesh</a>
443	panneer	<a href="http://linkedin.com/panneer">http://linkedin.com/panneer</a>
441	karthik	<a href="http://linkedin.com/karthik">http://linkedin.com/karthik</a>

Fig no: 8.2.2 FEED DETAILS OF A PARTICULAR COMPANY

Untitled




panneer

panneer

panneer

**Fig no: 8.2.3 GENERATED FEEDS**

## DATABASE SCHEMA:


FEED_SUBSCRIPTION		Create
Table	Data	
Table	1 row	
BDIT	name	URL
	vignesh	http://localhost:5160/FeedGenerator/FeedGenerator
	ndtv	http://feeds2.feedburner.com/1ndtvNews-TopStories
	BBC	http://newsrss.bbc.co.uk/rss/newscastline_world_edition/front_page.rss.xml

rows 1 - 3 of 3

Fig no: 8.2.4 FEED\_SUBSCRIPTION TABLE

FEED_LIST						Create
Table	Data	Provider	Name	Country	Address	Phone
1	http://linkedin.com/vignesh		vignesh			440
2	http://linkedin.com/panneer		panneer			443
3	http://linkedin.com/karthik		karthik			447
4	http://feedproxy.google.com/~ndtv/feeds/TopStories/~3/UA4pPY95dwe-0m-ind-as-at-boom-pilot-tells-canadian-firms-18590.php		India's IT boom: Pilot tells Canadian firms			493
5	http://feedproxy.google.com/~ndtv/feeds/TopStories/~3/LbTqk77Sx7Q-sachin-will-always-be-my-favourite-batman		Sachin will always be my favourite batman			494

Fig no: 8.2.5 FEED\_LIST table

Create 

Key	Data	...	...	...	...	...	...
21	Indh	fstg	-	-	-	www.linkedin.com	
1	Kartik	s mca	26-MAR-10 04:22:31.463900 AM	26-MAR-10 04:22:31.463900 AM		http://linkedin.com/kartiks	
2	vignesh	s BE	26-MAR-10 04:22:33.602000 AM	26-MAR-10 04:22:33.602000 AM		http://linkedin.com/vignesh	
3	panneer	s BE Comp	26-MAR-10 04:22:33.612000 AM	26-MAR-10 04:22:33.612000 AM		http://linkedin.com/pannee	

row 1 - 4 of 4

Fig no: 8.2.6 SOA\_REGISTRY\_OBJECT TABLE

## REFERENCE :

- <http://www.allbusiness.com/sales/customer-service/1961-1.html>
- [http://www.1000advices.com/guru/customer\\_retaining.html](http://www.1000advices.com/guru/customer_retaining.html)
- Iverson, Will (December 2, 2004), *Hibernate: A J2EE Developer's Guide* (First ed.), Addison Wesley, <http://www.manning.com/bauer/>
- U.S. Patent & Trademark Office. "'RSS' Trademark Latest Status Info"
- Lakshminarayanan, Sitaraman (June 2008). Oracle Web Services Manager: securing your web services. Birmingham: Packt Publishing. p. 1. "Oracle Web Services Manager..
- Kathy, Sierra; Bert Bates & Bryan Basham. *Head First Servlets & JSP*. O'Reilly Media