FND RESULT SERVER

Software Initial Design

22.12.2010
FND Software
 Ahmet AKYOL   e1408558
Gürkan KUŞ   e1502558
Hakan EMEKCİ e1502327
Rıdvan TANIK   e1502707

Table of Content

<div align="center">

Software Initial Design Report

</div>

1. Introduction

This report is intended to describe the initial design of "FND Result Server" project of group FND. It provides clarification and design of the project in a way that not only we- the project owners and implementers- but also any IT professional can understand the road map.

1.1 Problem Definition

With the developments of technology in the web platforms many different services are offered to people to provide accessibility of information, they need, whenever and wherever they want. But of course, although the improvements of technology in this area, there are still extreme but encountered situations or needs that even this improved technology can not meet such as extremely high number of clients trying to request information from your site at the same time.

Many web solutions use general purpose databases to store information and general purpose servers to response the requests of the clients. It is very logical to use these systems in many applications but if there is possibility for your application to encounter extreme situations, then the response that these general purpose systems provides can not be offered to client at a reasonable time. Considering this, we decided to develop this project to make possible that people can get the information they need even in extreme situations.

1.2 Purpose

FND Result Server is a system that makes possible web applications

- to response to considerably more numbers of requests than general purpose systems do.
- to response to each client request faster than any general purpose system.

1.3 Scope

FND Result Server is a system that can be collaborated with any web application which uses general purpose databases such as Oracle, MySQL, MSSQL...

1.4 Definitions, Acronyms and Abbreviations

FND: Feasible Network Devolopment

http : Hyper Text Protocol

JSF : Java Server Faces

JPA : Java Persistance API

EJB : Enterprise Java Beans

NIO : New I/O

UI : User Interface

RDBMS : Relational Database Management System

1.5 References

Netbeans is the integrating development tool that we are going to use for implementing the FND Result Server : http://en.wikipedia.org/wiki/Netbeans

Scrum is an iterative, incremental methodology for project management often seen in agile software development:

http://en.wikipedia.org/wiki/Scrum_%28development%29

MySQL is a relational database management system (RDBMS)[1] that runs as a server providing multi-user access to a number of databases :

http://en.wikipedia.org/wiki/Mysql

The Netty project is an effort to provide an asynchronous event-driven network application framework and tools for rapid development of maintainable high performance & high scalability protocol servers & clients:

http://www.jboss.org/netty

Membase is a distributed key-value database management system, optimized for storing data behind interactive web applications :

http://www.membase.org/

2. System Overview

System has following components:

- Administration Tool

In order to make system aware about the database tables which will be included in the system, an administration tool is going to be created. This

administration tool which can be used by only the registered admins is going to help admins to retrieve the necessary data from database and insert it into the membase in this system. This tool is also going to make admins embed the pre-prepared fancy user interface templates into the system, so that every application which uses this system will have the freedom of using its own user interface. It means that there will be no restriction of using a standard template. In addition by using this tool, admins will be able to make configurations on the server side such as setting the number of FND Result Servers which will be used in system to balance the load of the requests, setting the ip's of these FND Result Servers.

MySQL Database

This database will have tables to hold the information about users who are admins of the system, information about the configuration of the FND Result Servers and user interface templates which will be embedded to the system in each project. Also it holds information about the projects which are created before so that by editing the properties of these projects they can be made use of in the future.

FND Result Servers

These servers will be specialized servers to meet the needs of the extreme client request cases. If the clients' requests can scale, there must be a way to scale the abilities or numbers of these servers. So, on demand their number can be increased depending on the number of requests. This property although each

FND Result Server is able to send responses to very large number of client requests , makes it possible to create multiples of the responses which each FND Result Server sends. In addition, each of these FND Result Servers is able to response to each client request. For instance, say that in our system which we created by using administration tool there are 3 different FND Result Servers. Any client request can be reponsed by any of the FND0, FND1 and FND2. Also there must be a way to stop unnecessary servers.

FND Result Server contains following three components:

> Admin Tool Client

This tool is different from administration tool. Every FND Result Server has one admin tool client. It is going to be used to monitor each of the FND Result Server. It will be possible to start, restart or stop a specific FND Result Server. Also this tool is going to give system the ability to see how much memory and resources each FND Result Server uses.

> FND Http Server

These servers will be different than general purpose servers which allow very limited number of connections to be opened. Because we are trying to optimize the system to response more client requests, these FND http servers will have the capability of opening considerably more number of connections than general purpose servers do. So, by this property each of the FND Result Servers is going to answer more numbers of client requests successfully. It means that clients will not see "Server Connection Failure" or "Connection to Server Timed Out" warnings.

> Membase DataStore

As it is known, general purpose databases save data into a file and retrieve it back from the same file. Because of these very time consuming file operations, the server can send response to client very late in terms of computing timing. In case of an extremely high number of clients trying to make queries from database, these general purpose databases will queue the queries and response each of them one by one and the last transactions will be executed very late. So either the requesting client cannot have a response or the response sent to the client is very late. By using Membase Datastore, we guarantee that the query will be returned to the client very fast because there will be no mention of file operations.

3. Design Considerations

3.1 Design Assumptions, Dependencies and Constraints

3.1.1 Portability

FND servers' admin tool will develop with java language thus the system can work on any operating system java virtual machine. There will have installation process on Linux machine because of performance issues for FND server nodes. It can be easily installed for others.

3.1.2 Reliability

When user wants to call the FND system over a given period of time, the system should correctly deliver services as expected by the user. The reliability of the system will be satisfying if it delivers services as specified.

3.1.3 Maintainability

When the FND system is in use, new requirements may emerge. When these requirements emerge, the system should be changeable to accommodate these requirements for maintaining the usefulness of the system. If the system is not maintainable, then the system cannot be modified for new requirements. In this situation, a new system should be developed to provide new requirements. The maintainability is important in order to avoid from high cost.

3.1.4 Security

The FND system should resist accidental or deliberate intrusions, when users operate on the system. If the system should not resist accidental or deliberate intrusions, then important data may be stolen by hackers. Thus, security of the system shall be low and trust of users shall be ruined. So, security of the system is very important for users.
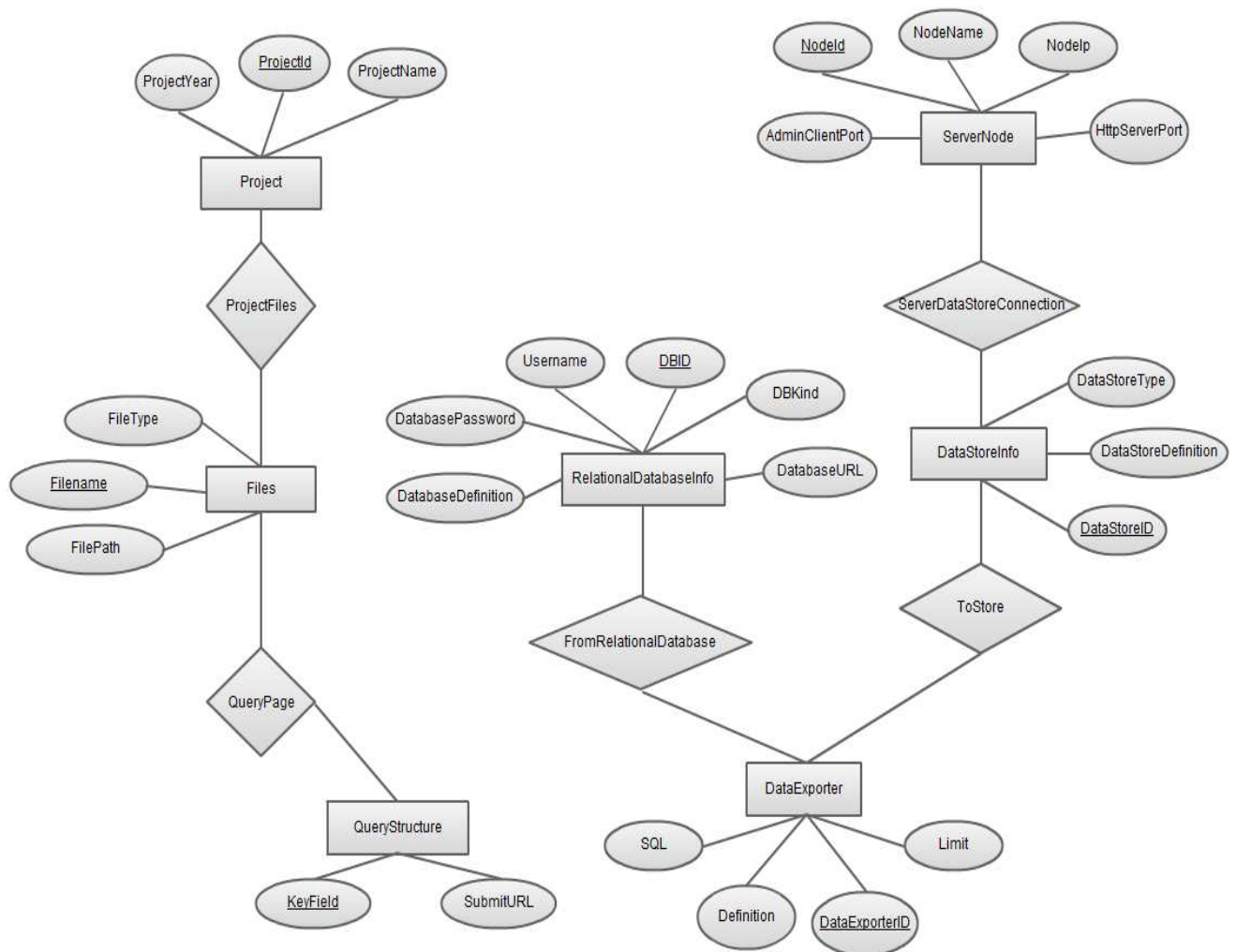
4.1.2 Availability

When the FND system has any request at any given time, system should be available, it should be up and running and able to deliver useful service at this time. The availability of the system shall be good if it delivers services when it is requested. Otherwise, if requests are not responded at any given time then it implies poor availability.

4. Data Design

4.1 Data Description and Data Flow Diagram

4.1.1 Entity Relation Diagram

Here is some important points related to above Entity Relation Diagram:

In order to use the system we need at least two databases. One of them is going to be our system's database named "FNDAdminDB" which will be used to hold information about newly created projects, user interfaces, FND server ip's, query structures and server nodes. The "FNDAdminDB" database will

consist of 7 entities and 5 relations. The names of entities and relations and their attributes are specified below. The other databases mentioned above is the databases which the system will be used to retrieve the information are the databases of the companies who wants to use our system.

- • "Project" Entity
- i) ProjectId
- ii) ProjectName
- iii) ProjectYear

"Project " entity is going to be used to hold the data information about the newly created project namely its name, its year and its Id which is the primary key of the entity.

- • "Files" Entity
- i. FileName
- ii. FileType
- iii. FilePath

"Files " entity is going to be used to hold the data information about the query pages to be used namely its name which is the primary key, its type and its path which will tell FND Result Servers where to get the query pages.

- • QueryStructure
- i. KeyField
- ii. SubmitURL

"QueryStructure " entity is going to be used to hold the data information about the query structure in the pages namely its KeyField which is the primary key and SubmitURL which is going to specify which page to be sent the form data.

- "ServerNode" Entity
i. NodeId
ii. NodeName
iii. NodeIp
iv. AdminClientPort
v. HttpServerPort

"ServerNode" entity is going to be used to hold the data information about the Nodes of FND Result Servers which are primary key NodeId, NodeName, NodeIp, AdminClientPort and HttpServerPort.

- "DataStoreInfo" Entity
i. DataStoreId
ii. DataStoreType
iii. DataStoreDefinition

"DataStoreInfo" entity is going to be used to hold the data information about the data retrieved from database and stored in the memory namely the primary key DataStoreId, DataStoreType and DataStoreDefinition.

- "DataExporter" Entity
i. DataExporterId
ii. SQL
iii. Limit
iv. Definition

"DataExporter" entity is going to be used to hold the data information about SQL which will be used to retrieve data from database, limit of the returned query rows and definition of the exporter.

- "RelationalDatabaseInfo" Entity
    i. DBID
    ii. DBKind
    iii. Username
    iv. DatabasePassword
    v. DatabaseDefinition
    vi. DatabaseURL

"RelationalDatabaseInfo" entity is going to be used to hold the data information about the database which the data is going to be retrieved. It consists of 6 attributes which are the primary key DBID, DBKind, Username, DatabasePassword, DatabaseDefinition, DatabaseURL.

- "ProjectFiles" Relation

"ProjectFiles" relation holds the relations between "Project" and "Files" entities.

- "QueryPage" Relation

"QueryPage" relation holds the relations between "QueryStructure" and "Files" entities.

- "FromRelationalDatabase" Relation

"FromRelationalDatabase" relation holds the relations between "RelationalDatabaseInfo" and "DataExporter" entities.
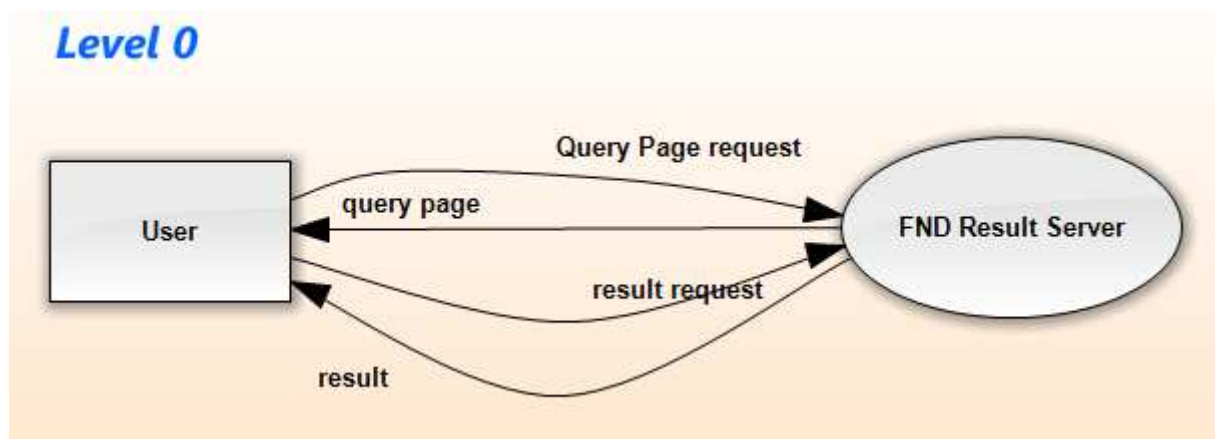
- "ServerDataStoreConnection" Relation

"ServerDataStoreConnection" relation holds the relations between "ServerNode" and "DataStoreInfo" entities.

- "ToStore" Relation

"ToStore" relation holds the relations between "DataExporter" and "DataStoreInfo" entities.

4.1.2 Data Flow Diagrams

Level 0 Data Flow Diagram represents the flow of data in the process of usage of the system by an end user. In this Data flow diagram actually there are two parts. First one is to show the user the query page, the second one is the result page.

Query Page

When user connects to the web page FND Result Servers will get the related template query page stored in the database and prepare it with the related KeyFields and SubmitURL and return it to end user as an http response so that the user can see the page on his/her browser.

Result Page

When user fills the required textfields and pushes the submit button in the query page, FND Result Servers get the information entered by the user to the known textfields, retrieves the necessary data from memory and gets the related template result page url which is stored in the result servers and prepare its fields with retrieved data and return it to user as an http response so that user can see his/her query results on the browser.
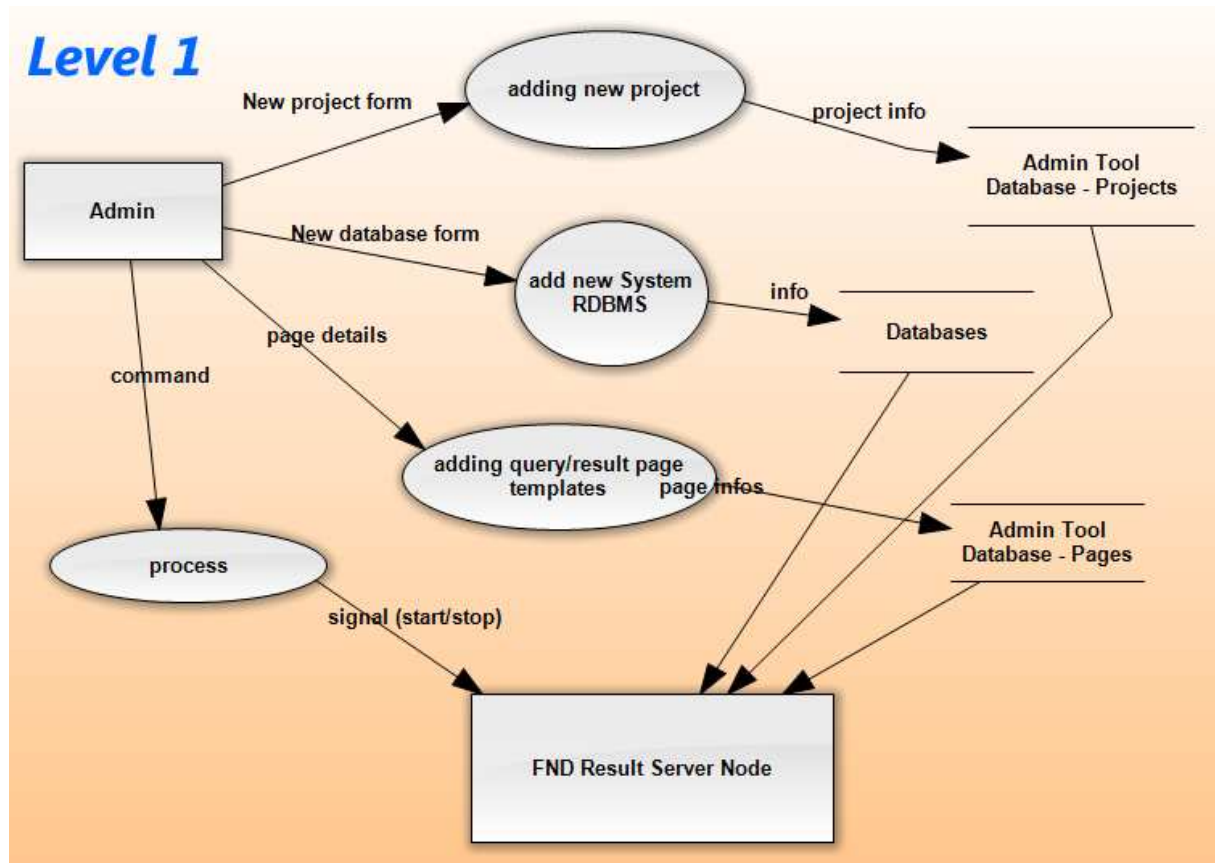
Level 1 Data Flow Diagram represents the flow of data while the admin uses the administration tool. This Data Flow Diagram consists of 4 parts.

1.New Project Form

We said that admins can create new projects. After filling the related parts in the admin tool's interface, the information will be added to   "Project" entity as a new row.

2.New Database Form

Admins can also add new database information to the system so that other databases can be used to retrieve data. This database information will be added to the database.

3.Adding Query/Result Page Templates

It will be possible for the admins to create new query/result pages and save it to the database. This speciality gives facility to change the user interface of the system without changing the actual project.

4.Process

By pressing the start button admins will be able to execute the whole system. Data will be exported to memory from specified database, and according the ip's in the database the FND Result Servers which are nodes will be ready to get requests and responses. Also the query and result pages' paths will be exported to memory so that they will not be read from database every time servers get a request. By pressing the stop button whole system will reset itself and FND Result Servers is not going to be able to response to the requests of the users.

4.2 Data Dictionary

Name	**User** Output
Input To	Admin Node
Output From	Computer of Admin
Description	Sends information to Admin Panel via Internet

Name	**Command**
Input To	Process
Output From	Admin Node
Description	Getting related command and interprets the output signal

Name	**Page Details**
Input To	Adding query/result page templates
Output From	Admin Node
Description	Sends information about templates

Name	New database form
Input To	System RDBMS
Output From	Admin Node
Description	Sends information about database such as type

Name	New project form
Input To	Adding New Project Node
Output From	Admin Node
Description	Sends information about projects

Name	Signal(start/stop)
Input To	FND result server node
Output From	Process
Description	Alarm to servers

Name	Page infos
Input To	Admin Tool(Databas-Pages)
Output From	Adding query/result page templates
Description	Sends Information about pages

Name	Database Information
Input To	Databases
Output From	System RDBMS
Description	RDBMS sends required information about Databases

Name	Project Info
Input To	Admin Tool Databases – Projects
Output From	Adding New Project Node
Description	Sends projects information.

| Name	Recreation Database
Input To	FND Result Server Node
Output From	Databases
Description	Creates new noSQL database and drag to in FND Result Server Node

Name	Recreation Pages
Input To	FND Result Server Node
Output From	Admin Tool Database-Pages
Description	Creates new quick page templates in FND result server node

5. System Architecture

5.1 Architectural Design

Major components of the project are a result server which handles "Query Page" and "Result Page" traffic which works under a specialized Linux environment and an administration tool. The administiration tool consists data translation which exports DBMS based data to a NoSQL Datastore System and templates of Result Page and Query Page.

The diagram is for better understanding of the main idea of our product. All components, interconnections and relations are analyzed in detail in the next sections of this document.
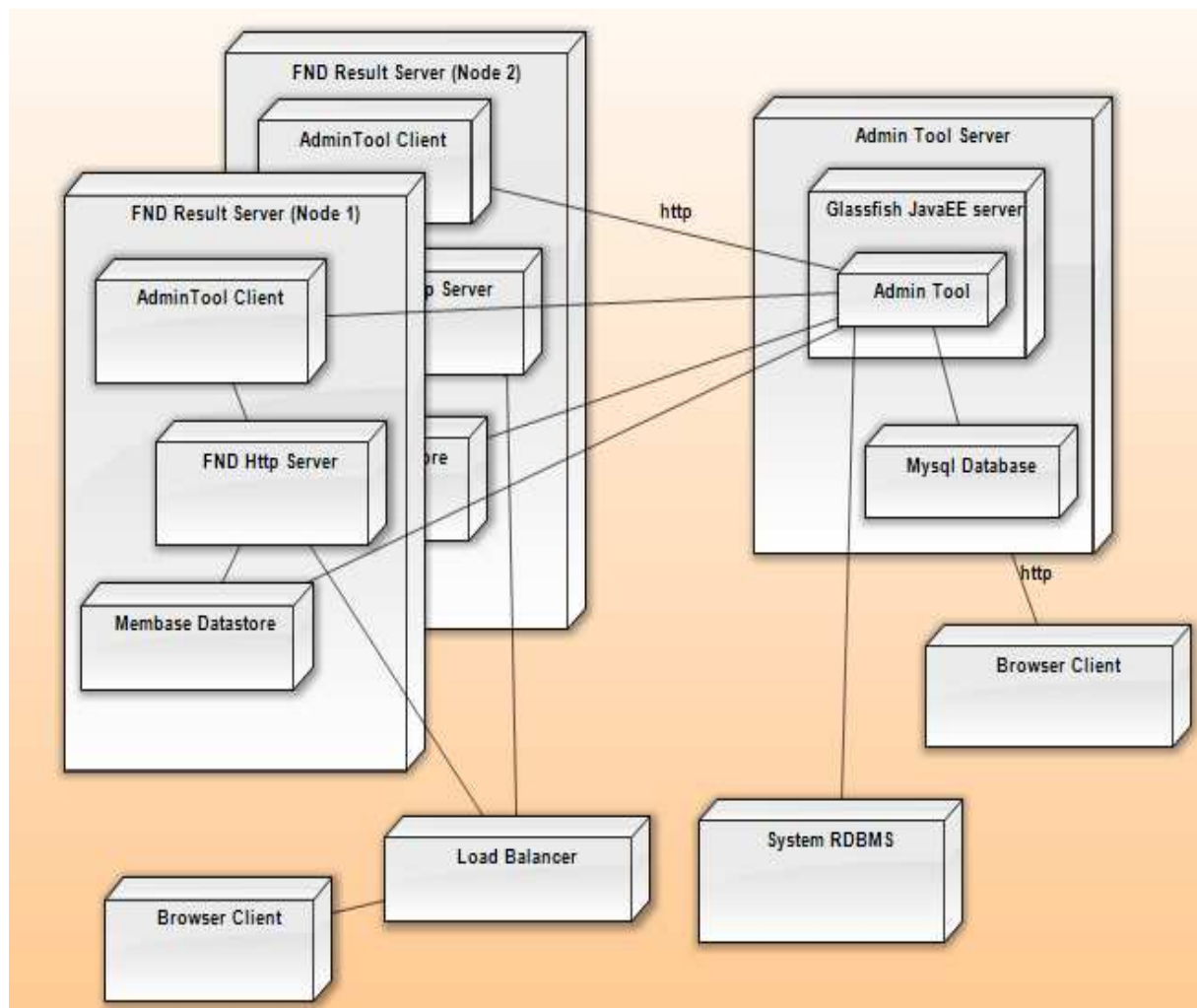
5.2 Description of Components

Admin Tool is used for controlling the whole system. It provides to admins openning/closing the FND Result Servers, transporting the data from RDBMS to NoSQL Datastores, creating result and query page templates.

FND Result Server is supposed to handle excessive requests coming from end users. It works with NoSQL Datastores such as Membase which contains data retrived from RDBMS.
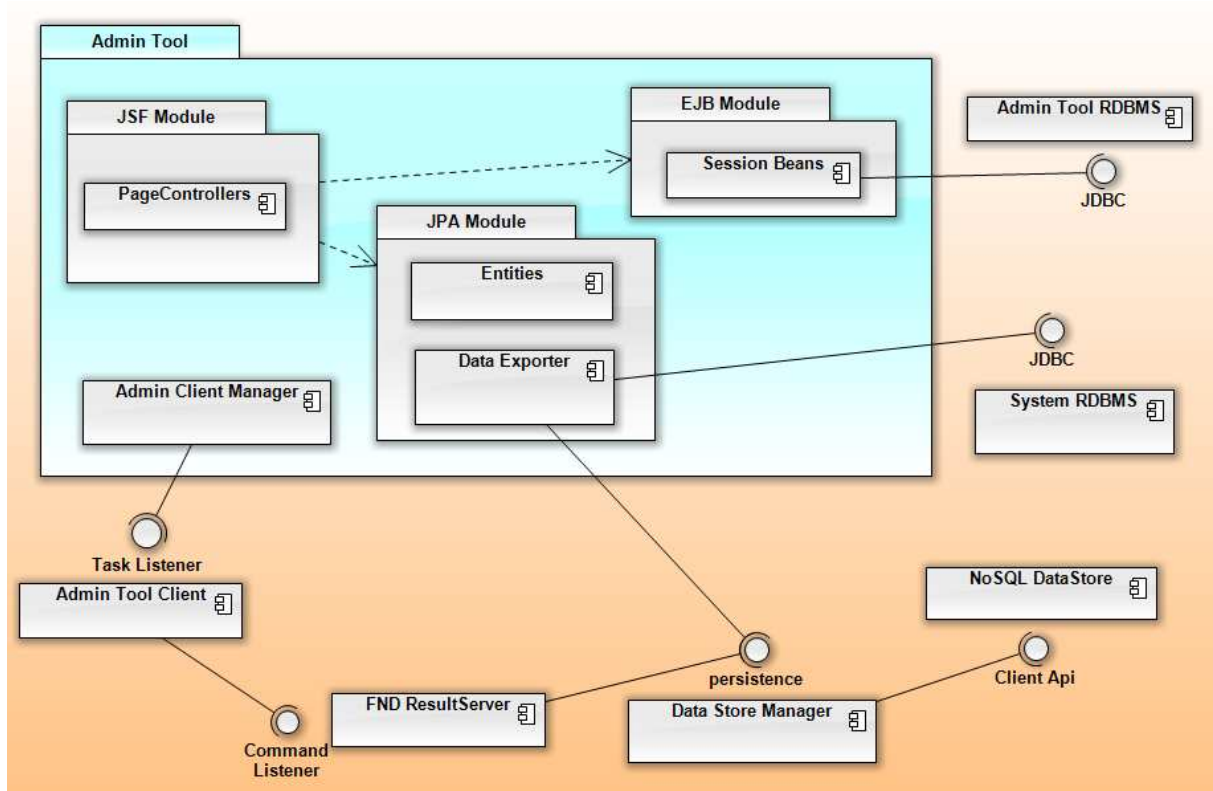
5.2.1 Component "Admin Tool"

Admin Tool is going to be implemented by JSF, JPA, EJB. It is going to be used by admins only to create project and control this project.

### *5.2.1.1 Processing narrative for component "Admin Tool"*

Admin will use this tool to create a project firstly. After that admin will choose the required data from RDBMS which is also choosen by admin to transport data to NoSQL Datastore. Admin can give command to Admin Tool Client which lies on the FND Result Server to start or stop the desired server nodes.

5.2.1.2 Component "Admin Tool" interface description

Admin Tool will has simple, user friendly interface. It is going to be implemented by JSF which is a Java-based Web application framework intended to simplify development integration of web-based user interfaces. Due to the concern of design issues we also want to use Prime Faces which is a lightweight open source component suite for Java Server Faces 2.0 featuring 100+ rich set of JSF components.

5.2.1.3 Component "Admin Tool" processing detail

As seen in the figure at the below Admin can transport data from RDBMS which represented as RelationalDBInfo to DataStoreInfo by using DataExporter. Admin can create project by using project class which is directly related with files which are used for creating query and result pages, represented as QueryStructure.
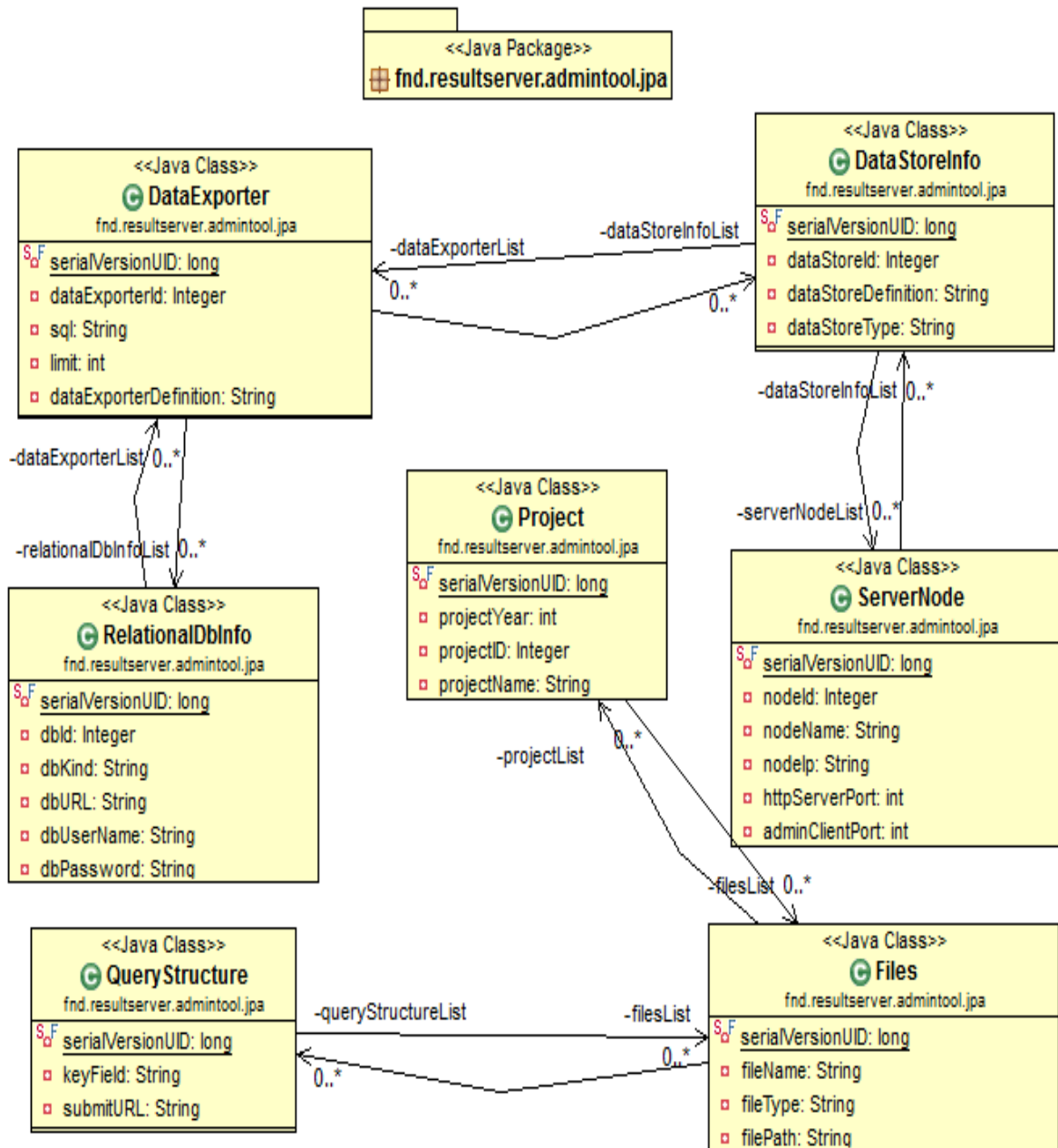
5.2.2 Component "FND Resul Server"

FND Result Server is going to be implemented by Jboss Netty. It is only going to be controlled by Admin Tool. It is responsible to answer excessive number of request coming from end users of our application.

### 5.2.2.1 Processing narrative for component "FND Resul Server"

FND Result Server's life will start with the command of Admin Tool : start. After being alive FND Result Server is ready to answer requests. It is main dusty is respond this request in reasonable time. It retrives requested data from NoSQL Datastore and send it to the demandant.

5.2.2.2 Component "FND Resul Server" interface description

Since FND Result Server is going to be controlled by Admin Tool there is no interface to the users.

5.2.2.3 Component "FND Resul Server" processing detail

As seen in the figure at the below, FND Result Server will get commands from Admin Tool by the help of CommandListener. It will retrive data from NoSQL Datastore by the help of ResultRepository. It will also servet he result in the form of a web page whose object oriented corresponds is QueryPage.
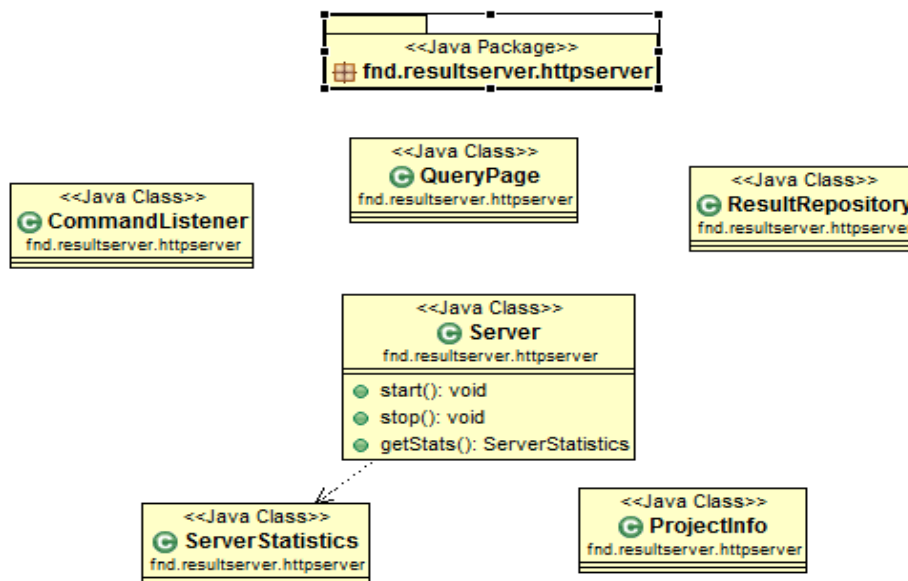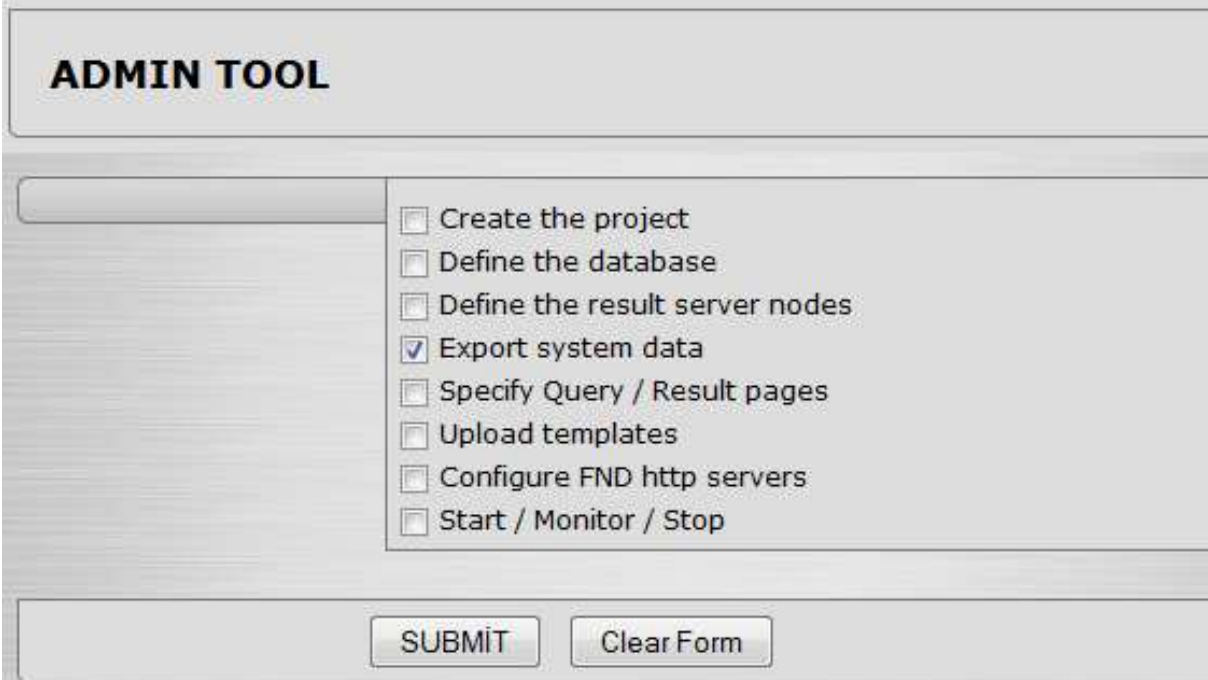
6. User Interface Design

6.1 Overview of User Interface

There are two kind of users – Admins and end users. Admin kind users will encounter Admin Tool interfaces. There are varius kind of operation that an admin can perform via Admin Tool. Therefore there are a lot of interface for admins. At below we serve some of them. On the other hand, for end users corrosponding interfaces are going to be designed by admins via Admin Tool.

6.2 Screen Images

DEFINE RELATIONAL DATABASE

URL :

KIND :

ID :

USERNAME :

PASSWORD   •••••••••

Submit

Add New Project

Name :

ID :

Year :   12 - 26 - 2010 at 21 / 03 AM

Ay   Gün   Yıl   Saat   Dakika

Submit

EDIT RELATIONAL DATABASE

URL :   www.osym.gov.tr/xsw

KIND :   ORACLE

ID :   osym

USERNAME :   root

PASSWORD   •••••••••

[EDIT] [Clear Form]

7. Libraries and Tools

7.1 JSF (Java Server Faces)

JavaServer Faces (JSF) is a Java-based Web application framework intended to simplify development integration of web-based user interfaces.

JSF is a request-driven MVC web framework based on component driven UI design model, using XML files called view templates or Facelets views. Requests are processed by the FacesServlet, which loads the appropriate view template, builds a component tree, processes events, and renders the response (typically HTML) to the client. The state of UI components (and some other objects) is saved at the end of each request (called stateSaving (note: transient true)), and restored upon next creation of that view. Several types of state-saving are available, including Client-side and Server-side state saving. Out of the box, JSF1.x uses JavaServer Pages (JSP) for its display technology, but can

also accommodate other technologies (such as XUL and Facelets). JSF2 uses Facelets by default for this purpose. Facelets is a more efficient, simple, and yet more powerful view description language (VDL).

7.2 Netbeans

NetBeans refers to both a platform framework for Java desktop applications, and an integrated development environment (IDE) for developing with Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala, Clojure, and others.The NetBeans IDE is written in Java and runs everywhere where a JVM is installed, including Windows, Mac OS, Linux, and Solaris. A JDK is required for Java development functionality, but is not required for development in other programming languages. The NetBeans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the NetBeans platform (including the NetBeans IDE) can be extended by third party developers.

7.3 Jboss Netty

Netty is a NIO client server framework which enables quick and easy development of network applications such as protocol servers and clients. It greatly simplifies and streamlines network programming such as TCP and UDP socket server.

'Quick and easy' doesn't mean that a resulting application will suffer from a maintainability or a performance issue. Netty has been designed carefully with the experiences earned from the implementation of a lot of protocols such as FTP, SMTP, HTTP, and various binary and text-based legacy protocols. As a

result, Netty has succeeded to find a way to achieve ease of development, performance, stability, and flexibility without a compromise.

7.4 Membase

Membase is a distributed key-value database management system best suited for applications that require predictable, low-latency, random access to data with high sustained throughput.

It is protocol-compatible with Memcached (both text and binary protocols), so if an application is already using Memcached, Membase can be dropped in without any change to application code or client configuration.

Membase is extremely easy to scale up, and down, making it attractive for applications in which data set size or data access volume can rapidly change. Membase scales linearly from a single server, to a cluster of hundreds of servers.

7.5 Java

Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet.

Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatilty, efficiency, and portability, Java has become invaluable to developers by enabling them to:

- Write software on one platform and run it on virtually any other platform

- Create programs to run within a Web browser and Web services

- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more

- Combine applications or services using the Java language to create highly customized applications or services

- Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat

8. Time Planning (Gannt Chart)

8.1 Term-1 Gannt Chart

AU/	Task Name	Start	Finish	Duration
1	Project Management	1/15/2010	6/24/2011	33.2w
2	Pre-proposal	1/15/2010	1/22/2010	1.2w
3	Proposal	1/15/2010	1/19/2010	1w
4	Software Requirement Specification	11/22/2010	12/3/2010	2w
5	Initial Design Report	12/6/2010	12/24/2010	3w
6	Final Design Report	12/24/2010	1/4/2011	1.6w
7	Prototype Demo	1/3/2011	1/21/2011	3w
8	Designing Menu	12/20/2010	1/21/2011	5w
9	Designing Engine Architecture	12/20/2010	1/21/2011	5w
10	Constructing Admin tool	2/21/2011	5/30/2011	14.2w
11	Final Release	5/30/2011	6/10/2011	2w
12	Testing	4/11/2011	5/12/2011	4.8w
13	User Manual	5/16/2011	5/24/2011	1.4w
14	Final Demo	6/7/2011	6/10/2011	.8w

8.2 Term-2 Gannt Chart

AU	Task Name	Start	Finish	Duration	Feb 2011			Mar 2011			Apr 2011			May 2011				Jun 2011	
					2/6	2/13	2/20	2/27	3/6	3/13	3/20	3/27	4/3	4/10	4/17	4/24	5/1	5/8	5/15	5/22	5/29	6/5	6/12	6/19
1	Project Management	1/15/2010	6/24/2011	33.2w																			
2	Pre-proposal	1/15/2010	11/12/2010	1.2w																			
3	Proposal	11/15/2010	11/19/2010	1w																			
4	Software Requirement Specification	11/22/2010	12/3/2010	2w																			
5	Initial Design Report	12/6/2010	12/24/2010	3w																			
6	Final Design Report	12/24/2010	1/4/2011	1.6w																			
7	Prototype Demo	1/3/2011	1/21/2011	3w																			
8	Designing Menu	12/20/2010	1/21/2011	5w																			
9	Designing Engine Architecture	12/20/2010	1/21/2011	5w																			
10	Constructing Admin tool	2/21/2011	5/30/2011	14.2w																			
11	Final Release	5/30/2011	6/10/2011	2w																			
12	Testing	4/11/2011	5/12/2011	4.8w																			
13	User Manual	5/16/2011	5/24/2011	1.4w																			
14	Final Demo	6/7/2011	6/10/2011	.8w																			

9. Conclusion

The Initial Design Report for "FND Result Server" gives the definition, purpose and scope of the project. It is designed in detail that the "Admin Tool" and Server Nodes. The design decision will be used during developing the project.

It includes data flow models, class diagrams, entity relationship diagrams, possible use cases. It is explained the works that we have done so far and within the schedule we give the future work to be done.