

FND RESULT SERVER

Software Requirement Specifications

12.07.2010

FND Software

Ahmet AKYOL e1408558

Gürkan KUŞ e1502558

Hakan EMEKÇİ e1502327

Rıdvan TANIK e1502707

İçindekiler

1 Introduction.....	5
1.1 Background Information.....	5
1.2 Project Title.....	6
1.3 Project Purpose.....	6
1.4 Project Scope.....	6
1.5 Team.....	10
1.5.1 Structure.....	10
1.5.2 Member Roles.....	10
1.6 Process Model.....	12
1.6.1 Meetings.....	12
2 Research.....	13
2.1 Technology Research.....	14
2.1.1 Database Management.....	14
2.1.2 Programming Languages.....	14
2.1.3 Frameworks.....	15
2.1.4 Platforms of Development.....	16
3 Overall Descriptions.....	16
3.1 Product Perspective.....	16
3.2 Design and Implementation Constraints.....	18
3.2.1 Software Constraints.....	18
3.2.2 Hardware Constraints.....	18
3.2.3 Development Environment Requirements.....	18
3.3 Product Features.....	19
4 Requirements.....	21
4.1 Functional Requirements.....	21
4.1.1 Use Case Scenarios.....	21
4.2 Performance Requirements.....	28
4.3 Design Constraints.....	28
4.4 Software System Attributes.....	29
4.4.1 Reliability.....	29
4.4.2 Availability.....	29
4.4.3 Security.....	29
4.4.4 Maintainability.....	30

4.4.5 Portability.....	30
4.5 External Interface Requirements	30
4.5.1 User Interfaces	30
4.5.2 Communications Interfaces.....	31
4.5.3 Hardware Interfaces	31
5 Data Model	32
5.1 Data Flow Diagrams	32
6 Task Assignment to Each Group Member	33
7 Project Scheduling	33
7.1 Gantt Chart.....	33
8 References	35
Appendix A: Glossary	35
FND	35
<i>Feasible Network Development</i>	35
HTTP	35
<i>Hyper Text Transfer Protocol</i>	35
JSF.....	35
Java Server Faces	35
NIO.....	35
New I/O	35

1 Introduction

This report is intended to describe the requirements of "FND Result Server" project of group FND. It provides clarification for the road map of the project to specify the requirements and detail the use cases.

1.1 Background Information

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 Project Title

The project name is "FND Result Server".

1.3 Project 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.

1.4 Project 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...

It will include 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 reponed 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.

1.5 Team

1.5.1 Structure

Our team has “Controlled Decentralized (CD) which means a defined leader who coordinates specific tasks. Problem solving is a group activity but implementation of solutions is partitioned among subgroups .Control is vertical and communication is horizontal” structure. The leader of the team is Ahmet Akyol.

1.5.2 Member Roles

We’ll use “Scrum” as software methodology(process model). Team members’ roles according to “Scrum”.

Ahmet Akyol : Product Owner

Rıdvan Tanık : Scrum Master

Hakan Emekçi : Team member

Gürkan Kuş : Team member



ScrumMaster - *The Film Director*

The ScrumMaster protects the Team from all external disturbances. He is not part of the Team and is its leader and a facilitator. He improves the productivity of the Scrum-Team and controls the "inspect and adapt" cycles of Scrum. He protects the Team and works with the Product Owner to maximize the return on investment. He makes sure that the agile ideals are understood and respected by all stakeholders. However he is not responsible for the delivery of the product.



Product Owner - *The Storywriter*

The Product Owner drives the project from the business point of view. She communicates a clear Vision of the product and she defines its main characteristics. She also accepts the product at the end of a Sprint. The Product Owner's main responsibility is to ensure that the Team only works on the most important Backlog Items needed by the organization. She has the same goals as the Team and she helps the Team to do its work during a Sprint by not disturbing its members and by giving them all necessary information promptly. The Product Owner is responsible for the return on investment.



Team - *The Actors*

The Team delivers the product and is responsible for its quality. The Team works with all the requestors - the Customers and the End Users - to create the Product Backlog. The Team analyses the Product Backlog Items so that its members will have the necessary information in order to build it. The Team creates the design of the functionality, it tests the Backlog Items and delivers the product as agreed. The Team performs its commitment voluntarily. It is accountable for its work and has to consider the nature of its organization and of the project. The Team also works continuously together with the Product Owner to define the strategic direction of the product development project.

1.6 Process Model

Scrum is an iterative, incremental methodology for project management often seen in agile software development. Although Scrum was intended for management of software development projects, it can be used to run software maintenance teams, or as a general project/program management approach.

1.6.1 Meetings

1.6.1.1 Daily Scrum Meeting

Each day, teams gather for the daily scrum meeting. Many teams new to Scrum underestimate the purpose and value of the daily scrum. There are four purposes for the daily scrum meeting:

- ✚ To synchronize effort among all team members.
- ✚ To commit to the work to be accomplished in the next day and reaffirm the team's commitment to the sprint goal.
- ✚ To identify any impediments to be addressed by the team.
- ✚ To ensure the team members are "on the same page." The full team needs to hear about the problems facing each member so that solutions can be addressed after the meeting. The daily

scrum enables them to micro-steer their progress toward their goal together.

1.6.1.2 Sprint Planning Meeting

At the start of a sprint, teams meet with the stakeholders of the project to plan the next sprint. Planning a sprint requires two meetings: the sprint prioritization meeting and the sprint planning meeting. The prioritization meeting prepares, or “grooms,” the product backlog and identifies a potential sprint goal. The sprint planning meeting creates the sprint backlog that defines the work that the team commits to completing by the next sprint review.

2 Research

We have conducted a wide research on especially on technology. We can't conduct market research because there is no similar product on this issue . However it can be said that this product satisfies the requirements of many which need huge network traffic. Technological research helped to decide on which platform, technologies, APIs and libraries will be used.

2.1 Technology Research

2.1.1 Database Management

2.1.1.1 MySQL

MySQL is a popular choice of database for use in web applications, and is a central component of the widely-used LAMP web application software stack — LAMP is an acronym for "Linux, Apache, MySQL, PHP". Its popularity is closely tied to the popularity of PHP. MySQL is used in some of the most frequently visited web sites on the Internet, including Flickr, Facebook, Wikipedia, Google — though not for searches, Nokia.com and YouTube.

2.1.1.2 Membase

Membase is a distributed key-value database management system, optimized for storing data behind interactive web applications. Membase automatically spreads data and I/O across servers. This "scale out" approach at the data layer permits virtually unlimited growth of transaction capacity, with linear increases in cost and constant per-operation performance. Membase is in production behind thousands of applications, from shared-server deployments at Heroku to dedicated Membase clusters with hundreds of servers at Zynga.

2.1.2 Programming Languages

2.1.2.1 Java

Java refers to a number of computer software products and specifications from Sun Microsystems, a subsidiary of Oracle Corporation,

that together provide a system for developing application software and deploying it in a cross-platform environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones on the low end, to enterprise servers and supercomputers on the high end. Java is used in mobile phones, Web servers and enterprise applications, and while less common on desktop computers, Java applets are often used to provide improved and secure functionalities while browsing the World Wide Web.

2.1.3 Frameworks

2.1.3.1 *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.

2.1.4 Platforms of Development

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.

3 Overall Descriptions

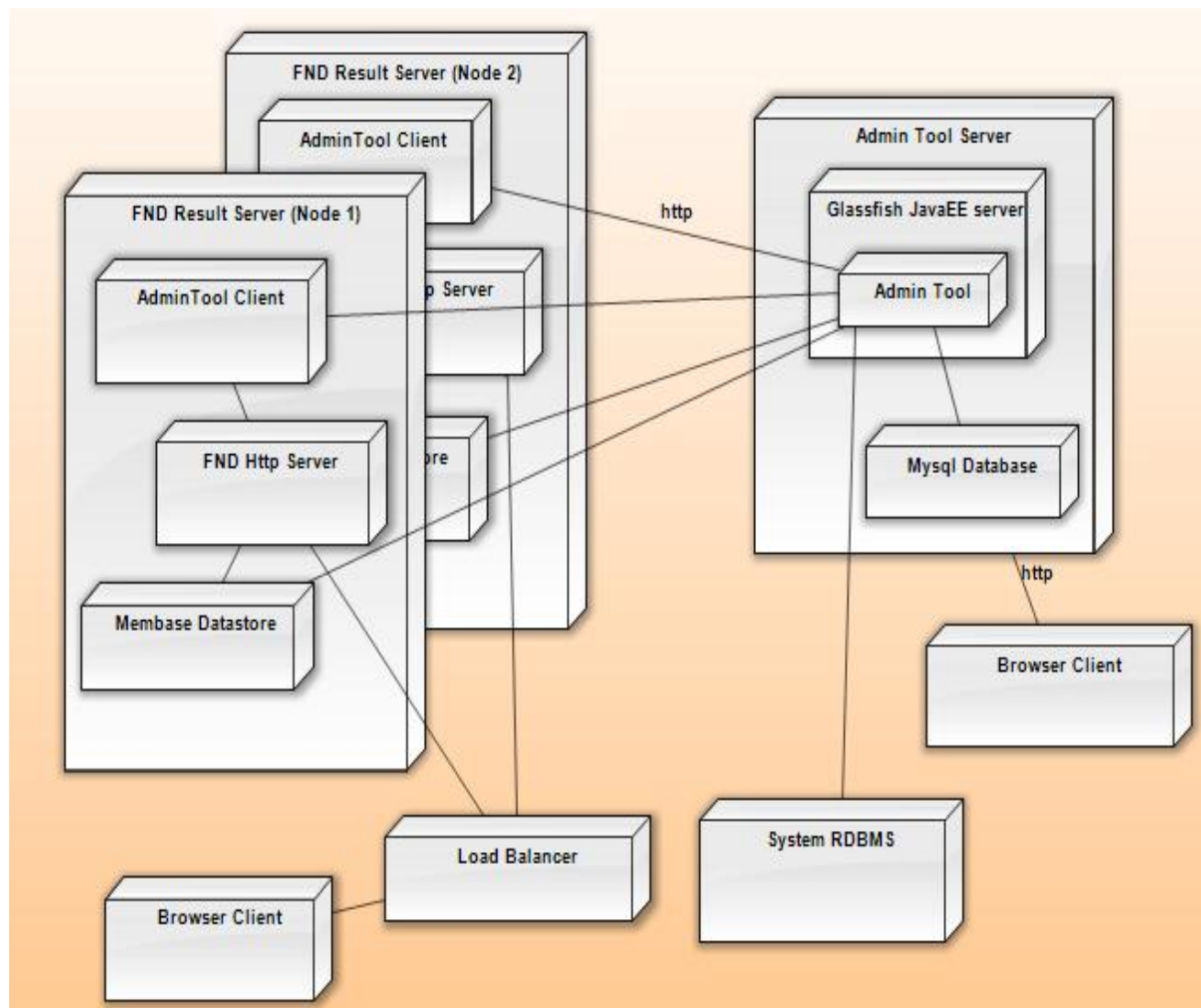
3.1 Product Perspective

Our project group intends to offer a general solution to problems under High Network Traffic such as "Service Unavailable" and/or "Service Temporarily Overloaded".

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 administration 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.



3.2 Design and Implementation Constraints

3.2.1 Software Constraints

- ✚ In order to use the system through web, internet connection should be established on the system.
- ✚ Linux operation system 64 bit
- ✚ JBoss
- ✚ MySQL server as DBMS
- ✚ Glassfish
- ✚ Mem-base
- ✚ Any browser such as Mozilla Firefox

3.2.2 Hardware Constraints

- ✚ For the server-side , we will use 3 at least 2 ghz computer with 2 gb ram.
- ✚ For the admin, 1 computer with 2 gb ram

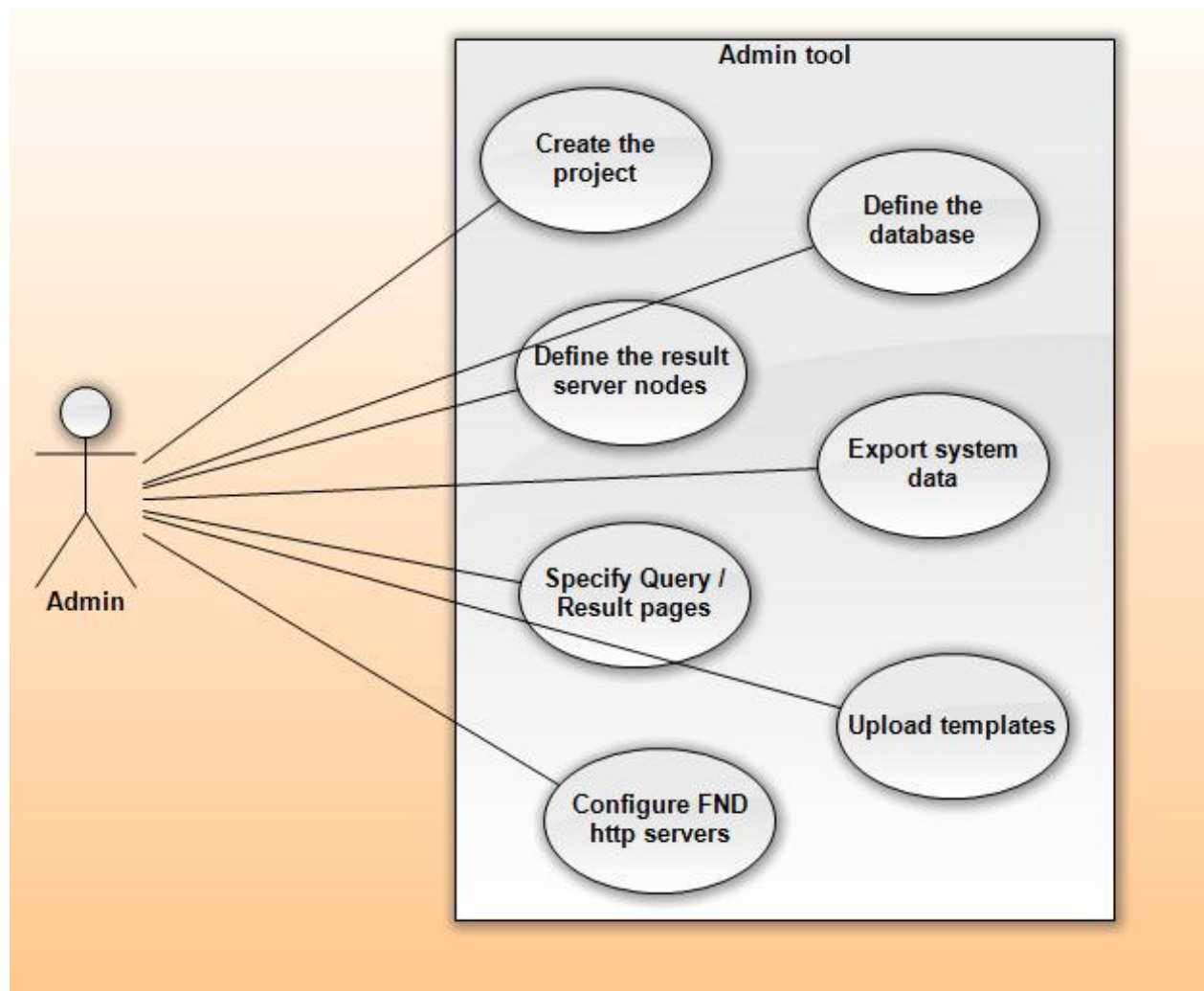
3.2.3 Development Environment Requirements

- ✚ NetBeans IDE
- ✚ MySql Administrator

3.3 Product Features

Features for admin:

- 1- Login to the system
- 2- Create the project
- 3- Define the database
- 4- Define the result server nodes
- 5- Export system data
- 6- Specify Query / Result pages
- 7- Upload templates
- 8- Configure FND http servers
- 9- Start / Monitor / Stop



These all different features are accessed by different people using different logins and passwords so that security is maintained and these logins and passwords are generated by the system itself on the administrator command.

4 Requirements

4.1 Functional Requirements

4.1.1 Use Case Scenarios

Use Case name: Login to the system

User Case description: Admin candidate user is able to enter username/password combination and login to the system successfully.

Actors: User Admin

Precondition(s): User must exist in the database.

Postcondition(s): User can login successfully.

Basic flow of events: Successful login

1. User launches the login screen.
2. User enters a combination of username and password.
3. System validates the combination and logs the user successfully.

Alternate flow A: Incorrect username/password combination

- A.3. System validation of the username/password combination fails due to incorrect entry.
- A.4. Systems asks the user to re-enter the username/password combination.
- A.5. Go back to basic flow 2.

Use Case name: Create the project

User Case description: The user is able to create project for new task.

Actors: User Admin

Precondition(s): User must have authentication rights.

Postcondition(s): User can create new project succesfully.

Basic flow of events: Successful create

1. User launches the create project screen
2. User fills the project details according to project description.
3. System creates new project.

Alternate flow A: Conflicting project names

- A1. System does not validate the project name due to existing project name.
- A2. System alerts the user to reenter another project name.
- A3. Go basic flow 2.

Use Case name: Define the database

User Case description: The user is able to define database as a source.

Actors: User Admin

Precondition(s): User must have authentication rights.

Postcondition(s): User can define the database succesfully.

Basic flow of events: Successful define database

1. User launches the “define database” screen
2. User fills the database details related to the database of the project.
3. System validates if database exists.
4. System saves the information to database of admin tool.

Use Case name: Define the result server nodes

User Case description: The user is able to define each result server nodes.

Actors: User Admin

Precondition(s): User must have authentication rights

Postcondition(s): User can define the result server nodes

Basic flow of events: Successful define the result server nodes

1. User launches the “define the result server nodes” screen
2. User fills the result server nodes’ details related to the project.
3. System validates if result server node exists.
4. System saves the information to database of admin tool.

Use Case name: Export system data.

User Case description: The user is able to export data to mem-base.

Actors: User Admin

Precondition(s): User must have authentication rights

User must have predefined database

User must have predefined result server nodes

Postcondition(s): User can export the data to mem-base

Basic flow of events: Successful define the result server nodes

1. User launches the "Export Data" screen
2. User chooses predefined database and result server nodes.
3. User creates a query which fetches the necessary fields.
4. User pushes the export button.
5. System exports data to mem- base using defined query.

Alternate flow A: Does not exist the database

- A1. System does not find the predefined database.
- A2. System alerts the user to choose again the database.
- A3. Go basic flow 2.

Alternate flow B: Does not exist the result server nodes

B1. System does not find the predefined result server nodes.

B2. System alerts the user to choose again the result server nodes.

B3. Go basic flow 2.

Use Case name: Specify Query / Result pages

User Case description: The user is able to specify names of Query / Result pages

Actors: User Admin

Precondition(s): User must have authentication rights.

Postcondition(s): User can specify names of Query / Result pages

Basic flow of events: Successful specify Query / Result pages

1. User launches the "Specify pages" screen.
2. User fills the form field name information.
3. User fills the query and result page names.
4. System saves information to database of admin tool.

Use Case name: Upload templates

User Case description: The user is able to upload created Query / Result pages

Actors: User Admin

Precondition(s): User must have authentication rights.

Postcondition(s): User can upload created Query / Result pages.

Basic flow of events: Successful upload Query / Result pages

1. User launches the "Upload templates" screen.
2. User fills the paths of template pages.
3. System uploads the template pages to the FND http server.

Alternate flow A: The paths of template pages do not exist

A1. System does not find the paths of template pages.

A2. Go basic flow 2.

Use Case name: Configure FND http servers

User Case description: The user is able to configure FND http servers by using project description.

Actors: User Admin

Precondition(s): User must have authentication rights.

User must have predefined project

User must have predefined FND server nodes

User must have predefined database

Postcondition(s): User can configure FND http servers according to project description

Basic flow of events: Successful configure FND http servers

1. User launches the "Configure FND" screen.
2. User chooses the predefined project.
3. User pushes the "Configure Now" button.
4. System configures the FND http servers.

Use Case name: Start / Monitor / Stop

User Case description: The user is able to start / monitor / stop FND http servers anytime

Actors: User Admin

Precondition(s): User must have authentication rights.

User must have predefined project

User must have predefined FND http server nodes

User must have predefined database

User must have configured FND http servers

Postcondition(s): User can configure FND http servers according to project description

Basic flow of events: Successful Start / Monitor / Stop FND http servers

1. User launches the "Start / Monitor / Stop" screen.
2. User chooses FND http server nodes.
3. User pushes the Start , monitor or stop button.
4. System Starts/Monitors/Stops chosen nodes.

4.2 Performance Requirements

Performance requirements are the most important among requirements for this project. Fundamentally, by this project it is intended that much more people get a response at the same time. It is expected that the response of every request button pressed or an option selected in the system will be in 5 seconds.

4.3 Design Constraints

While the current http servers works for 10000 – 20000 people, the FND system is designed and will be tested for 100000 users simultaneously.

4.4 Software System Attributes

4.4.1 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.

4.4.2 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.

4.4.3 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.

4.4.4 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 for provide new requirements. The maintainability is important in order to avoid from high cost.

4.4.5 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.

4.5 External Interface Requirements

4.5.1 User Interfaces

User interfaces or screens are in standart web forms . Additionally, there are some documents, buttons and text areas that are changing depending on the action taken by the users. We willl use JSF template to make screen more elegant.

4.5.2 Communications Interfaces

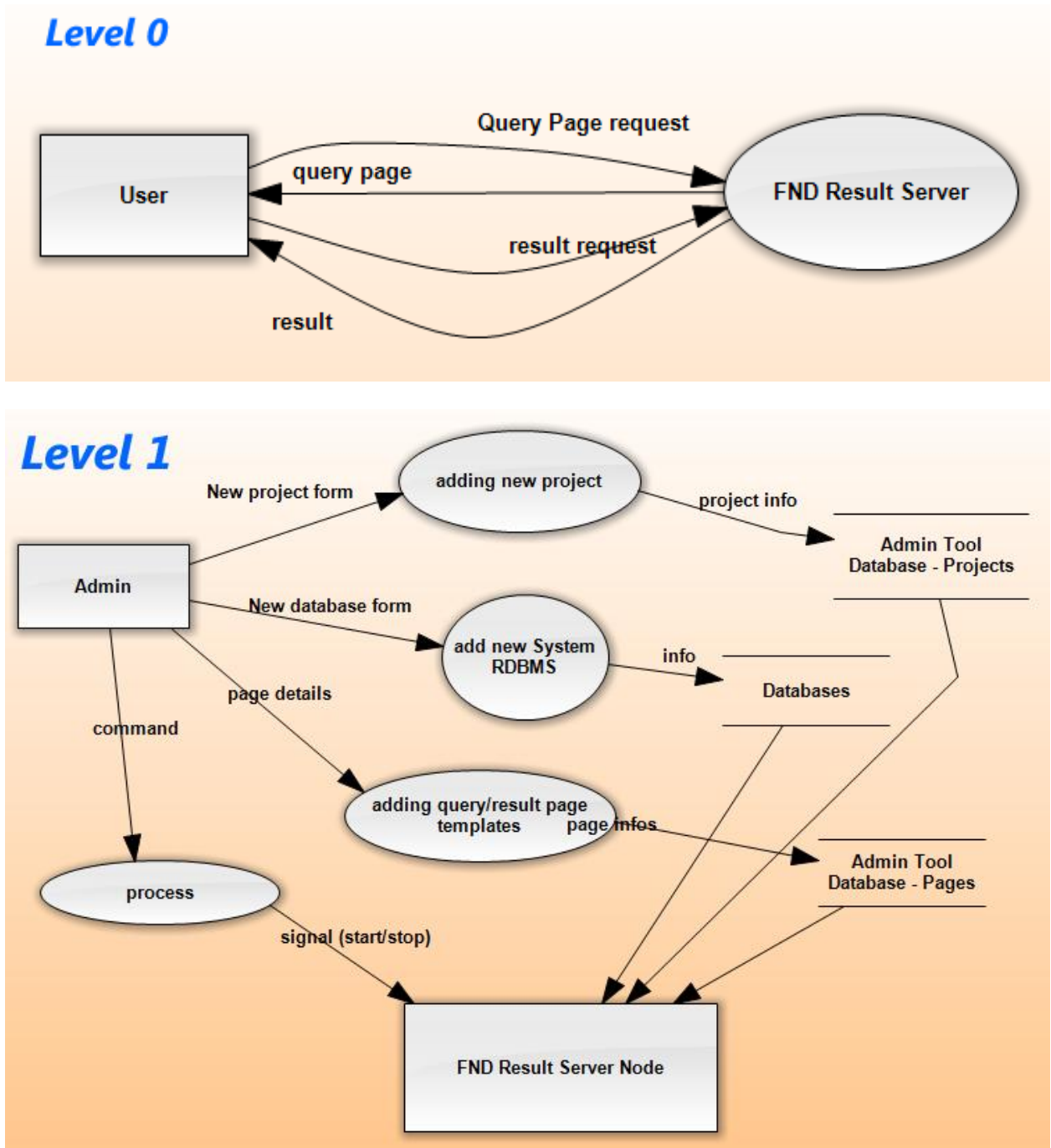
Hypertext Transfer Protocol (HTTP) will be used to provide a communication between systems, admin and users. Therefore, Internet connection and a browser are required in order for several functions to be executed.

4.5.3 Hardware Interfaces

Monitor screen – the software shall display information to the user via the monitor screen
Mouse – the software shall interact with the movement of the mouse and the mouse buttons. The mouse shall activate areas for command buttons and select options from menus.
Keyboard – the software shall interact with the keystrokes of the keyboard.

5 Data Model

5.1 Data Flow Diagrams



6 Task Assignment to Each Group Member

Each member is responsible for specific tasks as follows:

- ✚ Hakan Emekci: Admin Panel Implementations
- ✚ Ahmet Akyol: Server implementations.
- ✚ Gürkan Kuş: Testing / User interface design.
- ✚ Rıdvan Tanık: Database administrator.

Members of FND group are responsible roughly for detailing the above main tasks into minor.

7 Project Scheduling

7.1 Gantt Chart

AU	Task Name	Start	Finish	Duration	2010												2011																							
					Nov			Dec			Jan			Feb			Mar			Apr			May			Jun														
					11/7	11/14	11/21	11/28	12/5	12/12	12/19	12/26	1/2	1/9	1/16	1/23	1/30	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	11/5/2010	6/24/2011	33.2w																																				
2	Pre-proposal	11/5/2010	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																																				

8 References

<http://en.wikipedia.org/wiki/Netbeans>

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

http://en.wikipedia.org/wiki/Java_%28software_platform%29

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

<http://www.jboss.org/netty>

<http://www.membase.org/>

Appendix A: Glossary

FND	<i>Feasible Network Development</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
JSF	Java Server Faces
NIO	New I/O