## Malaria Patient Information system

**Submitted to**

Md. Nurul Ahad Tawhid

Lecturer, IIT, University of Dhaka

**Submitted by**

Abdus Satter BSSE-0401

Nadim Iqbal BSSE-0407

Md. Masudur Rahman BSSE-0413



Institute of Information Technology

University of Dhaka

Submission date:

## Chapter 1

## Introduction

This chapter describes the purpose of this Malaria Patient Information System design document. It includes the Data flow design, Architecture diagram, Component design, UI design.

### Planning Meeting

At an early stage in the project, several stakeholders and subject matter experts should be convened to discuss the project and make the product plan. Depending on the size of the project and its complexity, the meeting took several days or weeks.

* Date: 16 September, 2014

Place: IIT

Subject Matter: Data Flow Design

Group Members:

* + Abdus Satter (BSSE-0401)
  + Nadim Iqbal (BSSE-0407)
  + Md. Masudur Rahman (BSSE-0413)
* Date: 5 October, 2014

Place: IIT

Subject Matter: Architecture Design

Group Members: Above all members

* Date: 22 October, 2014

Place: IIT

Subject Matter: Component Design

Group Members: Above all members

* Date: 27 October, 2014

Place: IIT

Subject Matter: UI Design

Group Members: Above all members

### Scenario

**An MIS software for XYZ Limited**

There are 170 field stations to collect Malaria patients’ information. 170 field stations are located in 170 Upazillas and having 170 field officers to gather field level data. The data that the field officers need to collect are –

* Patient’s name
* Patient’s age
* Patient’s sex
* Patient’s address
* Duration of the disease
* Medication taken
* Current condition

The field officers will collect the data from the field operators (each of the Upazilla will have 5 operators), and send the information through a web interface to the Central Database (CDB) located at the Dhaka station. All the communications to the CDB must be authenticated.

The Project Manager (PM) will be able to query the CDB for generating reports, such as

* Age group vulnerable to malaria
* Area vulnerable to malaria
* Patient condition Vs Disease duration
* Effect of Medication in the course of time

Based on those reports, PM should be able to send Upazilla wise decisions to the field officers that need to be taken through the MIS software.

Unimportant/Generic diagrams can be avoided all together (For example, Authentication System or Collection of Data from the field operators need not be analyzed).

## Chapter 2

## Flow Oriented Model of MPI System

### 2.1 Introduction

We have almost described the Pre-admission System up to last chapter. But **Data Flow Diagram (DFD)** can play a vital role to visualize the system input and output. It is a significant modeling technique for analyzing and constructing information processes. That is why, we use DFD for the system in this chapter.

### 2.2 Data Flow Design (DFD)

The DFD takes an input-process-output view of a system. DFD literally means an illustration that explains the course or movement of information in a process. DFD illustrates this flow of information in a process based on the inputs and outputs.

In the figures, data objects are represented by arrows, external entities are boxes, transformations are circle and storages are represented by double lines.

**Level 0 DFD:**



Figure 2.1: Level-0 (Context Level) DFD of MPIS

FO = Field Officer

PM = Project Manager

MPIS = Malaria Patient Information System

MPI = Malaria Patient Information

DB = Database

**Level 1.1 DFD** (System must be authenticated)**:**



Figure 2.2: Level-1 DFD of MPIS

### 2.3 Conclusion

Data Flow Diagram helps us to visualize the MPI System as input of raw data, processing of data to transform information and output of information. The context level DFD reflects the whole system’s input-process-output. Level-1 DFD elaborates the context level more precisely and specifically.

## Chapter 3

## Software Architectural Design of MPI System

### 3.1 Introduction

The software architecture of a program or computing system is the structure of the system which comprises the software components, the externally visible properties of those components and the relationships among the components. The architectural design represents the structure of the data and program components that are required to build a computer-based system. That is why, we use the design for the MPI system in this chapter.

### 3.2 Architectural Design Process

The software architecture constitutes a relatively small, intellectually graspable model of how the system is structured and how its components work together. The basic steps of Architectural Design that we follow in our MPI System are given below.

1. Represent the System in Context
2. Define the Archetypes
3. Refine the Archetypes into Components
4. Refine the DFD Model
5. Deriving Program Architecture (Information Flow)

The whole process has been described graphically in the following pages in this chapter.

**1. Represent the System in Context**

**Super-ordinate systems**

**Peers**

Web Interface

Field Officer

I/F

Target system

MPIS

Project Manager

I/F

I/F

**Actors**

Authentication

Database

**Sub-ordinate systems**

Figure 3.1: Context Diagram of MPIS

**2. Define the Archetypes**

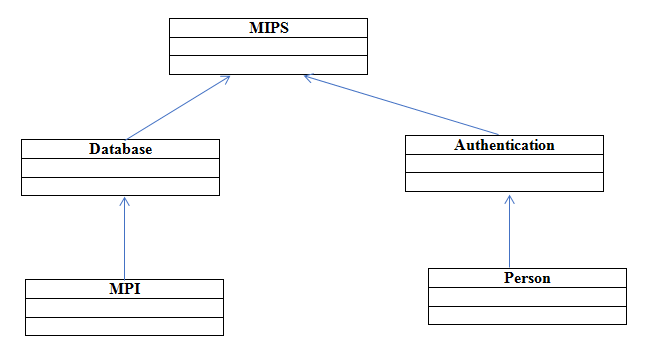


Figure 3.2: Archetypes of MPIS

**3. Refine the Archetypes into Components**

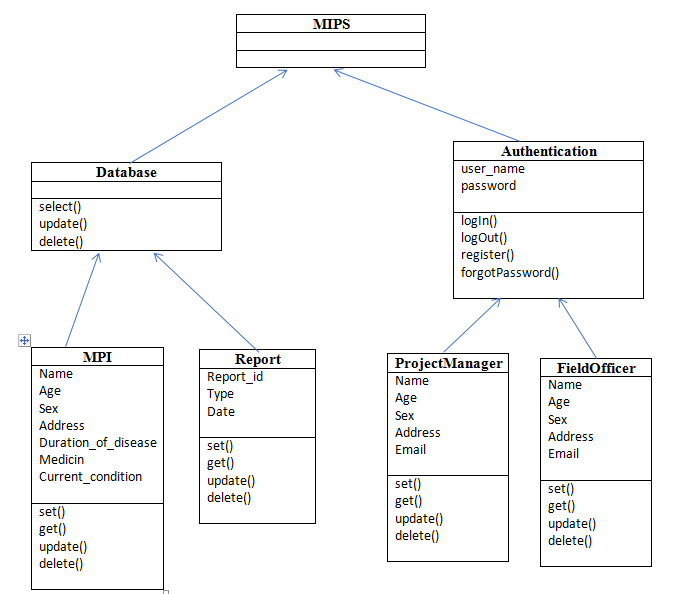


Figure 3.3: Refinery Components of MPIS

**4. Refine the DFD (Level-1) Model**



Figure 3.4: Refinery DFD (Level-1) of MPIS

**5. Deriving Program Architecture (Information Flow)**

There are 2 different types of information flow that have different treatments.

**Transform flow**: Overall data flows in sequential manner and follows one, or only a few, “straight line” paths (incoming, transform, output). Most of the cases, transform flows can be generated from the system.

**Transaction Flow**: Information flow has a single transaction node that *triggers* other data flow.



Figure 3.5: Transform Flow of Send MPI



Figure 3.6: Transform Flow of Generate Report



Figure 3.7: Transform Flow of Send Decision



Figure 3.8: Transaction Flow of Send MPIS (Whole System)

**3.3 Conclusion**

Representations of software architecture are an enabler for communication between all stakeholders interested in the development of a computer-based system. The architectural designer refines the software architecture into components to illustrate the overall structure and architectural style of the system based on the archetypes. The information flows represent the coherency of the system.

## Chapter 4

## Component Design

### 4.1 Introduction

Data, architectural, and interface design must be translated into operational software. To accomplish this, the design must be represented at a level of abstraction that is close to code. Component-level design establishes the algorithmic detail required to manipulate data structures, effect communication between software components via their interfaces, and implement the processing algorithms allocated to each component.

### 4.2 Class Elaboration

Problem Domain class Identification:

|  |
| --- |
| Project Manager |
| firstName  lastName  gender  age  mobilenumber  email  permanentaddress  presentaddress |
| Authenticate()  generateReport()  sendFeedback() |

|  |
| --- |
|  |

|  |
| --- |
| Field Officer |
| Firstname  Lastname  Gender  Age  Mobile number  Email  Permanent address  Present address  upazillaName |
| authenticate()  sendMPI()  recieveFeedback() |

|  |
| --- |
| Report |
| Type  ID  Date |
| getMPI()  createReport()  getReport() |

|  |
| --- |
| Malaria Patient Information |
| Name  gender  age  mobileNumber  permanentAddress  Duration |
| retrieveMPI()  sendMPI() |

|  |
| --- |
| Authentication |
| Username  Password  Email  Type |
| Login()  Logout()  forgotPassword()  getAuthenticationInfo()  check() |

|  |
| --- |
| Project Manager |
| firstName  lastName  gender  age  mobilenumber  email  permanentaddress  presentaddress |
| Login()  Logout()  forgotPassword()  getReport()  sendFeedback() |

|  |
| --- |
| IAuthenticate |
| Login()  Logout()  forgotPassword |

|  |
| --- |
| IGenerateReport |
| getReport() |

|  |
| --- |
| Field Officer |
| Firstname  Lastname  Gender  Age  Mobile number  Email  Permanent address  Present address  upazillaName |
| authenticate()  sendMPI()  recieveFeedback() |

|  |
| --- |
| IAuthenticate |
| Login()  Logout()  forgotPassword |

|  |
| --- |
| ISendMPI |
| sendMPI() |

|  |
| --- |
| Report |
| Type  ID  Date |
| getMPI()  createReport()  getReport() |

|  |
| --- |
| IGetMPI |
| getMPI() |

### 4.3 Collaboration Detail

MPI

[loggedIn] 2: =sendMPI (MPI)

Authentication

Field Officer

[Registered] 1: =login (userName, password)

[Registered] 1: =forgotPassword (userName, password)

[loggedIn] 2: =logout (userName, password)

Figure 4.1: Collaboration detail for Field Officer

Report

[loggedIn] 2(Report): =getReport ()

Project Manager

[Registered] 1: =login (userName, password)

[Registered] 1: =forgotPassword (userName, password)

[loggedIn] 2: =logout (userName, password)

Authentication

Figure 4.2: Collaboration detail for Project Manager

Report

: 1 = getMPI()

MPI

Figure 4.3: Collaboration detail fro Report

### 4.4 Appropriate Interface







### 4.5 Processing Flow

**Sign Up:**

Insert into Database

Available

Check User Name

Get User Information

No

Yes

Figure 4.5: Processing flow for Sign up

**Log In():**

Correct

Check

Get User name and Password

Incorrect

Figure 4.6: Processing flow for login

**Forgot Password():**

Incorrect

Get Email Id

Check

Correct

Get Email Id

Figure 4.7: Processing flow for forgot password

**generateReport():**

Store Report into DB

Create Report

Get MPI from DB

Figure 4.8: Processing flow for generate report

**UpdateReport():**

Get Report ID

Incorrect

CheckID

Correct

Retrieve Report form Database

Store Report into DB

update

Figure 4.9: Processing flow for update report

**SendFeedBack():**

Send message

Get Receiver

Create FB message

Figure 4.10: Processing flow for send Feedback

### 4.6 Elaborate behavior



Figure 4.11: elaborate behavior for PM



Figure 4.12: elaborate behavior for FO

**// I think rest of these that depicted below is not necessary, so you may ignore them if you wish….**

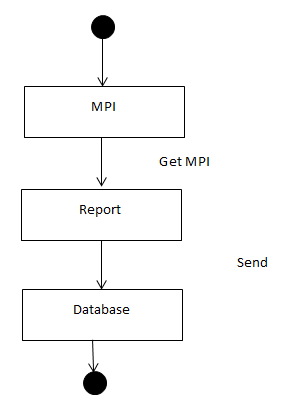


Figure 4.13: elaborate behavior for report

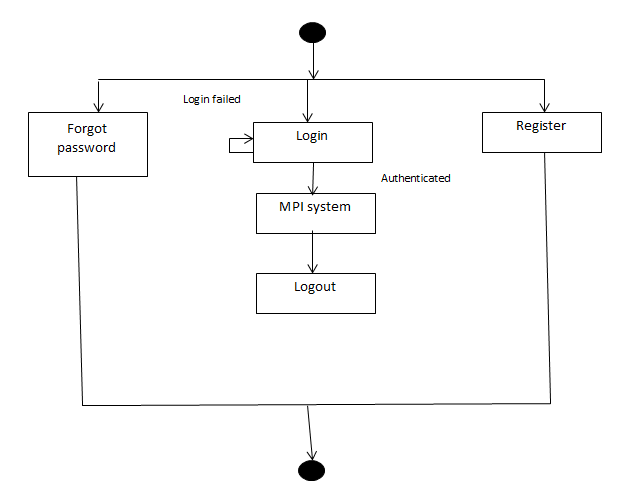


Figure 4.14: elaborate behavior for authentication

### 4.7 Deployment Model

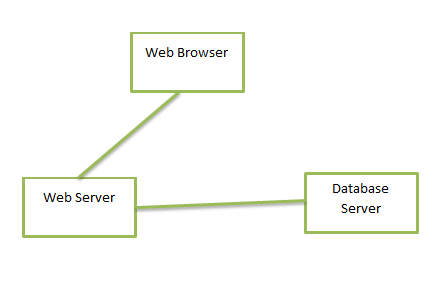


Figure 4.15: Deployment model

## Chapter 5

## User Interface Design for MPI System

### 8.1 Introduction

Requirement Analysis does take time, but solving the wrong problem takes even more time. In our MPI System, the requirements are almost done while we were doing usecases, activity diagrams and swimlanes in the SRS phase. However, we are trying to give a simple overview in this section of requirements modelling as a UI design. The UI design contains a structural element that provides an important view of content requirements for the MPI System including text, graphics and images and other data. These structural elements encompass content objects and all analysis classes - user visible entities that are created or manipulated as a user interacts with the system.

### 8.2 UI Design

There are two types of users who will interact with our application, the Field Officer and the Project Manage. Each type of users has their own task and knowledge in this application.

**The Field Officer can:**

* Fill up and store MPI
* See notification
* Change password
* Update their account information

**The Project Manager can:**

* Generate report
* Send notification
* Update report
* Change account information
* Add field officer

We assume that the users of our application are well enough to use standard browser and internet. We design our application under this assumption.

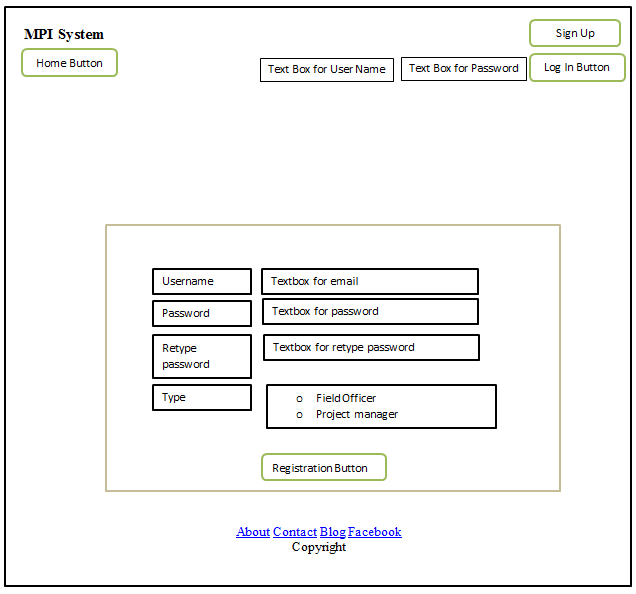


Figure 5.1: Registration Page

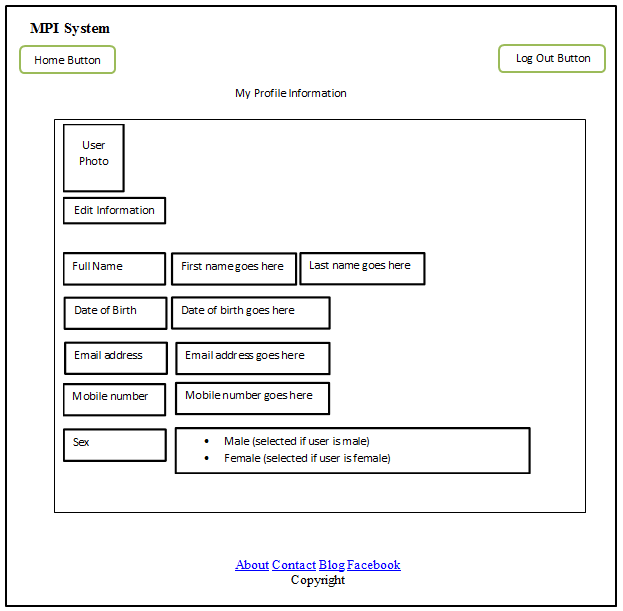


Figure 5.2: Field Officer Profile Page

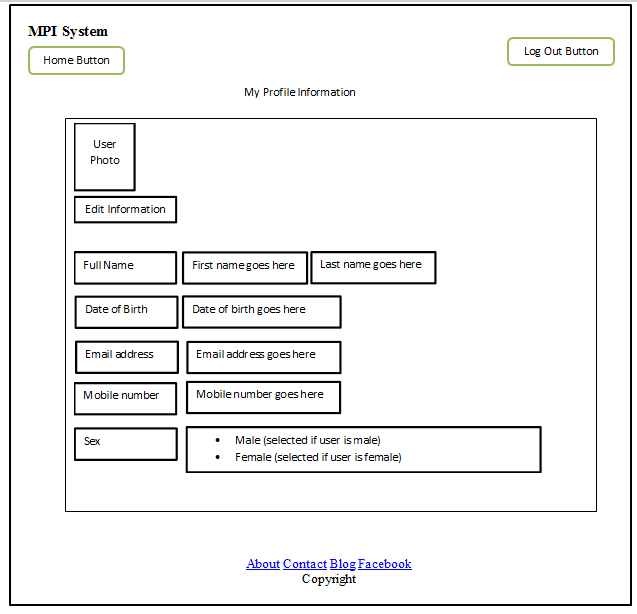


Figure 35: Project Manager Profile Page

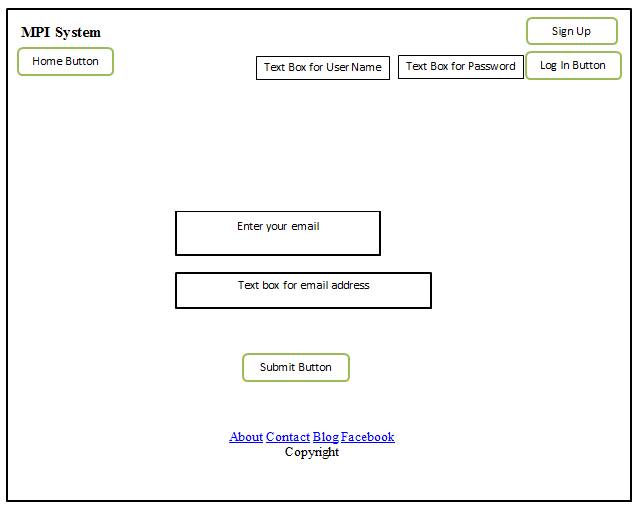


Figure 5.4: forgot password

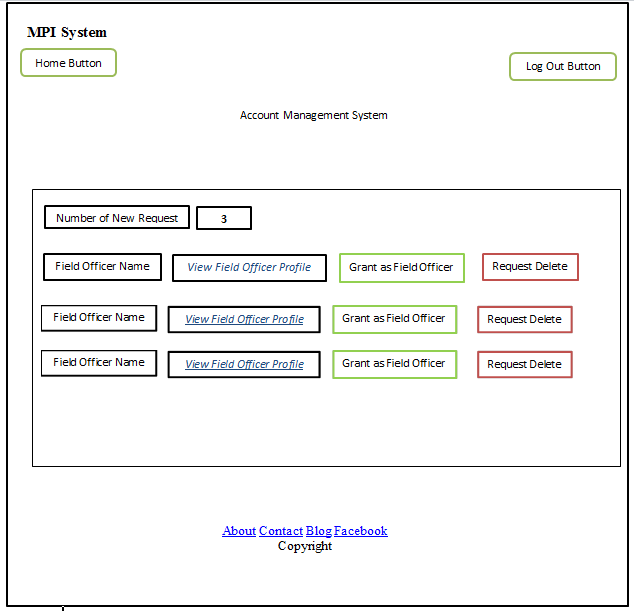


Figure 5.6: Account Management System

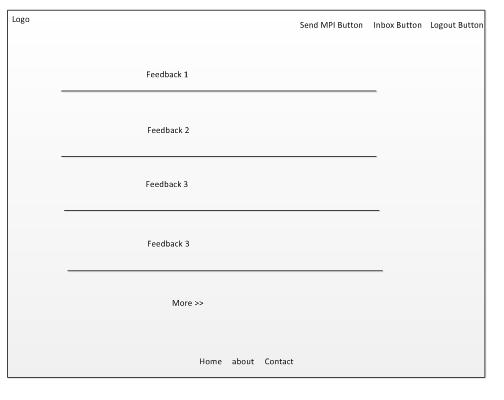


Figure 5.7: Home page for field officer

If the user clicks the “send MPI” button then he will be directed to the following page



Figure 5.8: Malaria patient information

If the user clicks the “send data” button then data will be sent to the central database.

If the user clicks “inbox” button then he will be directed to the following page where he will see all the received feedback. 

Figure 5.9: Feedback page

**UI for Project Manager**

Home Page for project manager:



Figure 5.10: Home page for PM

If the user clicks the “Generate Report” button a report will be generated based on the information provided in corresponding textboxes.

If the user clicks the “compose” button then he will see the following page to compose and send feedback

Figure 5.11: Send feedback page

User will be directed to the following page if he clicks the “Draft” button to see all the draft



Figure 5.12: Draft page

If the user clicks the “Sent” button he will be directed to following page where he will find all the sent messages



Figure 5.13: Sent messages