|  |
| --- |
| **N.I.X.Y** |
| **Project Planning Document** | |
| **Butterfly Tracking System** |
| **Version No. 2.1** |



Project Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| VersionNumber | Date | Revision Author | Description of Revision |
| 1.0 | 10/10/2015 | NIXY TEAM | Initial Creation |
| 1.1 | 11/15/2015 | NIXY TEAM | Add Estimation, Schedule, Organization |
| 1.2 | 11/22/2015 | NIXY TEAM | Finish |
| 2.0 | 11/23/2015 | Xiuyan Xin | Optimize the whole document |
| 2.1 | 11/24/2015 | Yue Dai | Optimize the whole document |

Contents

[**1. Introduction 3**](#_Toc436163548)

[**1.1 Project scope 3**](#_Toc436163549)

[**1.2 Major software functions 3**](#_Toc436163550)

[**1.3 Performance/Behavior issues 4**](#_Toc436163551)

[**1.4 Management and technical constraints 4**](#_Toc436163552)

[**2. Project Estimates 5**](#_Toc436163553)

[**2.1 Historical data used for estimates 5**](#_Toc436163554)

[**2.2 Estimation techniques applied and results 5**](#_Toc436163555)

[**2.3 Reconciled Estimate 12**](#_Toc436163556)

[**2.4 Project Resources 13**](#_Toc436163557)

[**3. Risk Management 13**](#_Toc436163558)

[**3.1 Project Risks 13**](#_Toc436163559)

[**3.2 Risk Table 14**](#_Toc436163560)

[**3.3 Overview of Risk Mitigation, Monitoring, Management 14**](#_Toc436163561)

[**4. Project Schedule** 17](#_Toc436163562)

[**4.1 Project task set** 17](#_Toc436163563)

[**4.2 Functional decomposition** 17](#_Toc436163564)

[**4.3 Task network** 18](#_Toc436163565)

[**4.4 Timeline chart** 20](#_Toc436163566)

[**5. Staff Organization** 21](#_Toc436163567)

[**5.1 Team structure** 21](#_Toc436163568)

[**5.2 Management reporting and communication** 21](#_Toc436163569)

[**6. Tracking and Control Mechanisms** 21](#_Toc436163570)

[**6.1 Quality assurance and control** 21](#_Toc436163571)

[**6.2 Change management and control** 22](#_Toc436163572)

[**7.Appendix** 22](#_Toc436163573)

[**7.1 Function points complexity matrix** 22](#_Toc436163574)

[**7.2 Activity Diagram** 23](#_Toc436163575)

[**7.3 Entity Relationship Diagram** 23](#_Toc436163576)

**1. Introduction**

**1.1 Project scope**

The purpose of Butterfly Tracking System is to monitor migration patterns of butterflies around the globe in order to preserve their habitat and analyze their behavior. When butterflies are tagged, the time and location is entered into the database. The tagged butterflies can then be tracked by reporters who will record the time and location of each sighting and update the database accordingly. The database will maintain a full record for each tagged butterfly, which contains the tag number, date tagged, and all times and locations it was sighted thereafter. The system is to use the gathered information to generate various tabular and visual reports for its users. The final product must be intuitive and easy to use, so that people with little or no computer skills could easily use it.

**1.2 Major software functions**

The Butterfly Tracking System will be implemented using a web-based interface so that it can be accessed using any modern web browser on any platform, including mobile devices, without the need of any additional software. Another benefit of such implementation is multi-user access to the system. The following is a list of core system functions:

1. Maintain and display a list of taggers/reporters
2. Maintain a list of user credentials for security and data integrity (Session Management)
3. Allow taggers/reporters to add butterfly tagging/sighting information into the database
4. Display various reports by different criteria(by tag number, tagger, reporter, location, or date)
5. Allow users to upload/download daily tracking data files based on butterfly tag (Batch file)
6. Given a coordinate and a radius (in miles) give a visual of all butterfly sightings in the region
7. Provide a user-friendly visualization of tagging and sighting
8. Implement an real-time and intuitive dashboard
9. Accept, store and display user comments and feedback

**1.3 Performance/Behavior issues**

Not Applicable.

**1.4 Management and technical constraints**

The biggest constraint for Butterfly Tracking system is time. This is mainly due to the team’s limited experience in software development. We could not find a particular software development environment that all team members were familiar with, so we expect a significant amount of time will be spent learning about a particular programming language. To meet the deadline, the team will focus on necessary functionalities of the system and once those are met additional features will be added if time permits.

**2. Project Estimates**

**2.1 Historical data used for estimates**

According to our team members’ experience before. There are some historical data that are relevant to the estimates.

1. Library Management System: we used java to develop the View and Control component, we used MySQL database Management System to maintain our data in database, and we used JDBC to connect our application to database.
   1. Original Estimates :
      1. Effort : 2 hours per day
      2. Staffing : 5 people
      3. Duration : 28 hours
   2. Actual Outcome :
      1. Effort : 4 hours per day
      2. Staffing : 5 people
      3. Duration : 120 hours
2. Hybris Yaas e-commerce website: we used AngularJS to develop the front-end application, we used Node.JS to develop the back-end application, and we use JSON to deliver data between front-end and back-end applications.
   1. Original Estimates :
      1. Effort : 8 hours per day
      2. Staffing : 4 people
      3. Duration : 80 hours
   2. Actual Outcome :
      1. Effort : 8 hours per day
      2. Staffing : 3 people
      3. Duration : 480 hours
3. Product Order System: we used the UI API that is provided by SAP to develop the fron-end application, and we used JSON file to storage data.
   1. Original Estimates :
      1. Effort : 8 hours per day
      2. Staffing : 1 people
      3. Duration : 80 hours
   2. Actual Outcome :
      1. Effort : 8 hours per day
      2. Staffing : 1 people
      3. Duration : 56 hours

**2.2 Estimation techniques applied and results**

**2.2.1 Estimation technique *m***

Tables or equations associated with estimation technique m are presented. Section 2.2.1 is repeated for each of m techniques.

2.2.1.1Estimate based on lines of codes:

According to the historical data in 2.1 we have, it illustrates the average amount of the lines of code per hour is approximately 20 LOC/hour for one person, so the estimate = total lines of code / 20 (LOC/Hour)

* + - * 1. Initial estimation

|  |  |
| --- | --- |
| **Functionality** | **Estimated Lines of Code** |
| Maintain and display a list of taggers/reporters | 500 |
| Allow taggers/reporters to add butterfly tagging/sighting into the database | 800 |
| Display various reports by different criteria(by tag number, tagger, reporter, location, or date) | 1,000 |
| Process batch files | 1,000 |
| Dashboard | 500 |
| Feedback | 800 |
| Session Management | 500 |
| Sighting by custom region | 1,000 |
| Visualization of migration route | 1,000 |
| **Estimate lines of code** | 7,100 |

1. Table 2-1 Initial estimation – based on codes

According to the formulation (estimate = total lines of code / 20 (LOC/Hour)) in 2.2.1.1 and Table 2-1 we have: total hours = 7,100(total lines of code) / 20 (LOC/hour) = 355 hours

1. Second-time estimation

|  |  |
| --- | --- |
| **Functionality** | **Estimated Lines of Code** |
| Maintain and display a list of taggers/reporters | 700 |
| Allow taggers/reporters to add butterfly tagging/sighting into the database | 1,000 |
| Display various reports by different criteria(by tag number, tagger, reporter, location, or date) | 1,400 |
| Process batch files | 1,600 |
| Dashboard | 1,000 |
| Feedback | 1,500 |
| Session Management | 1,200 |
| Sighting by custom region | 2,000 |
| Visualization of migration route | 2,000 |
| **Estimate lines of code** | 12,400 |

1. Table 2-2 Second-time estimation – based on codes

According to the formulation (estimate = total lines of code / 20 (LOC/Hour)) in 2.2.1.1 and Table 2-2 we have: total hours = 12,400 (total lines of code) / 20 (LOC/hour) = 620 hours

2.2.1.2 Estimate based pieces of functionality

1. Initial estimation

|  |  |  |
| --- | --- | --- |
| **Functionality** | | **Estimated hours per functionality** |
| Maintain and display a list of taggers/reporters | Main | 15 |
| Display | 15 |
| Allow taggers/reporters to add butterfly tagging/sighting into the database | Allow taggers to add butterfly tagging into the database | 10 |
| Allow reporters to add butterfly sighting into the database | 10 |
| Display various reports by different criteria(by tag number, tagger, reporter, location, or date) | Display reports by tag number | 6 |
| Display reports by tagger | 6 |
| Display reports by reporter | 6 |
| Display reports by location | 6 |
| Display reports by date | 6 |
| Process batch files | Loading of daily tracking data based on butterfly tag | 5 |
| Download of all/location/tag tracking data | 5 |
| Dashboard | Charts show information of butterflies | 15 |
| Charts show information of reports | 15 |
| Feedback | Feedback gathering | 10 |
| Feedback displaying | 10 |
| Session Management | Administrator | 10 |
| Normal user | 10 |
| Login Interface | 5 |
| Sighting by custom region | | 15 |
| Visualization of migration route | | 30 |
| **Total hours** | | 210 |

Table 2-3 Initial estimation – based on functionality

1. Second-time estimation

|  |  |  |
| --- | --- | --- |
| **Functionality** | | **Estimated hours per functionality** |
| Maintain and display a list of taggers/reporters | Main | 20 |
| Display | 20 |
| Allow taggers/reporters to add butterfly tagging/sighting into the database | Allow taggers to add butterfly tagging into the database | 15 |
| Allow reporters to add butterfly sighting into the database | 15 |
| Display various reports by different criteria(by tag number, tagger, reporter, location, or date) | Display reports by tag number | 8 |
| Display reports by tagger | 8 |
| Display reports by reporter | 8 |
| Display reports by location | 8 |
| Display reports by date | 8 |
| Process batch files | Loading of daily tracking data based on butterfly tag | 10 |
| Download of all/location/tag tracking data | 10 |
| Dashboard | Charts show information of butterflies | 30 |
| Charts show information of reports | 30 |
| Feedback | Feedback gathering | 20 |
| Feedback displaying | 20 |
| Session Management | Administrator | 15 |
| Normal user | 15 |
| Login Interface | 15 |
| Sighting by custom region | | 25 |
| Visualization of migration route | | 40 |
| **Total hours** | | 350 |

Table 2-4 Second-time estimation – based on functionality

* + - 1. Estimate based on function points :

According to the historical data in 2.1 we have, we can find out that the average of the amount of function point that we can produce every day is approximately 4 FP/day, which is 0.5 FP/hour. To determine the weighting factor, we referred to those five different complexity tables which are shown in the appendix.

User Input is an elementary process that processes data or control information that comes from outside the application boundary. The primary intent of a User Input is to maintain one or more ILFs and/or to alter the behavior of the system.

An external interface file (EIF) is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application. The primary intent of an EIF is to hold data referenced through one or more elementary processes within the boundary of the application counted. This means an EIF counted for an application must be in an ILF in another application.

An Internal Logical File (ILF) is a user-identifiable group of logically related data or control information maintained within the boundary of the application. The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted.

An external output (EO) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information. The processing logic must contain at least one mathematical formula or calculation, create derived data maintain one or more ILFs or alter the behavior of the system.

An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF of EIF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered.

[1]See references in 7.4 for source

1. Initial estimation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Measurement Parameter** | **Count** | **Weighting Factor** | | | **Result** |
| **Simple** | **Average** | **Complex** |
| Number of External Inputs | 39 | 3**√** | 4 | 5 | 117 |
| Number of External Outputs | 16 | 4 | 5**√** | 7 | 80 |
| Number of External Inquiries | 8 | 3**√** | 4 | 6 | 24 |
| Number of Internal Logical Files | 3 | 7 | 10**√** | 15 | 30 |
| Number of External Interfaces | 1 | 5**√** | 7 | 10 | 5 |
| **Total Count** | 256 | | | | |

Table 2-5 Initial estimation1 – based on function point

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ITEM | COMPLEXITY ADJUSTMENT QUESTIONS | SCALE | | | | | |
| No influence ---------🡪Essential | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | Does the system require reliable backup and recovery? |  |  |  | √ |  |  |
| 2 | Are data communications required? |  |  |  |  |  | √ |
| 3 | Are there distributed processing functions? |  |  | √ |  |  |  |
| 4 | Is performance critical? |  |  | √ |  |  |  |
| 5 | Will the system run in an existing, heavily utilized operational environment? |  | √ |  |  |  |  |
| 6 | Does the system require on-line data entry? |  |  |  |  | √ |  |
| 7 | Does the on-line data entry require the input transaction to be built over multiple screens or operations? | √ |  |  |  |  |  |
| 8 | Are the master files updated on-line? |  | √ |  |  |  |  |
| 9 | Are the inputs, outputs, files or inquiries complex? |  |  | √ |  |  |  |
| 10 | Is the internal processing complex? |  | √ |  |  |  |  |
| 11 | Is the code to be designed reusable? |  |  | √ |  |  |  |
| 12 | Are conversion and installation included in the design? | √ |  |  |  |  |  |
| 13 | Is the system designed for multiple installations in different organizations? | √ |  |  |  |  |  |
| 14 | Is the application designed to facilitate change and ease of use by the user? |  | √ |  |  |  |  |

Table 2-6 Initial estimation2 – based on function point

[2] See references in 7.4 for source

After going through the Table 2-5 and 2-6, we used the function points calculator and we got the result as 227.84 function points in total, which means the total hours = 227.84 (total function points) / 0.5 (function/hour) = 455.6 hours.

1. Second-time estimation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Measurement Parameter** | **Count** | **Weighting Factor** | | | **Result** |
| **Simple** | **Average** | **Complex** |
| Number of External Inputs | 45 | 3 | 4**√** | 5 | 180 |
| Number of External Outputs | 25 | 4 | 5**√** | 7 | 125 |
| Number of External Inquiries | 12 | 3 | 4**√** | 6 | 48 |
| Number of Internal Logical Files | 5 | 7**√** | 10 | 15 | 35 |
| Number of External Interfaces | 1 | 5**√** | 7 | 10 | 5 |
| **Total Count** | 393 | | | | |

Table 2-7 Second-time estimation 1– based on function points

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ITEM | COMPLEXITY ADJUSTMENT QUESTIONS | SCALE | | | | | |
| No influence ---------🡪Essential | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | Does the system require reliable backup and recovery? |  |  |  | √ |  |  |
| 2 | Are data communications required? |  |  |  |  |  | √ |
| 3 | Are there distributed processing functions? |  |  | √ |  |  |  |
| 4 | Is performance critical? |  |  | √ |  |  |  |
| 5 | Will the system run in an existing, heavily utilized operational environment? |  |  | √ |  |  |  |
| 6 | Does the system require on-line data entry? |  |  |  |  |  | √ |
| 7 | Does the on-line data entry require the input transaction to be built over multiple screens or operations? |  | √ |  |  |  |  |
| 8 | Are the master files updated on-line? |  |  | √ |  |  |  |
| 9 | Are the inputs, outputs, files or inquiries complex? |  |  |  | √ |  |  |
| 10 | Is the internal processing complex? |  |  | √ |  |  |  |
| 11 | Is the code to be designed reusable? |  |  |  | √ |  |  |
| 12 | Are conversion and installation included in the design? | √ |  |  |  |  |  |
| 13 | Is the system designed for multiple installations in different organizations? | √ |  |  |  |  |  |
| 14 | Is the application designed to facilitate change and ease of use by the user? |  |  | √ |  |  |  |

Table 2-8 Second-time estimation 2 – based on function points

[2] See references in 7.4 for source

After going through the Table 2-7 and 2-8, we used the function points calculator and we got the result as 381.21 function points in total, which means the total hours = 381.21 (total function points) / 0.5 (function/hour) = 762.4 hours.

**2.2.2 Estimate for technique *m***

Estimate generated for technique m.t

2.2.2.1. Initial estimation

* 1. Estimate based on lines of code : 355 hours
  2. Estimate based on pieces of functionality : 180 hours
  3. Estimate based on function points : 455.6 hours

2.2.2.2. Second-time estimation

* 1. Estimate based on lines of code : 620hours
  2. Estimate based on pieces of functionality : 350 hours
  3. Estimate based on function points : 762.4 hours

**2.3 Reconciled Estimate**

The final cost, effort, time (duration) estimate for the project (at this point in time) is presented here.

**2.3.1. Initial estimation**

1. Estimation of the part one

According to the first meeting we had, we planned to have 2 meetings per week, and every member will spend 8 hours per week on the part one. So the total amount of hours we estimated is

6 meetings \* 2 hours/meeting + 8 hours/week \* 2 weeks \* 4 members = 76 hours

So our first estimation of the part one is 76 hours.

1. Estimation of the whole project

According to 2.2.2, we get three totally different estimations of our whole project, since we cannot exactly tell which estimation is the best, we decided to average them. The formulation is

Reconciled estimate = (355 hours (estimate based on lines of code) + 180 hours (estimate based on pieces of functionality) + 455.6 hours (Estimate based on function points) /3 = 330.2 hours

So the 330.2 hours is our initial estimation of the whole project

* + 1. **Second-time estimation**

1. Estimation for the part one

Now the part one of the whole project is almost finished, according to the time records we have, we spent 85.5 hours on this part. And what we are going to do is to prepare for the presentation and optimize our final project planning document. There will be two more meetings, so according to the formula we made in 2.3.1.1, the estimation for the part one will be: result = 85.5 hours + 2 meetings \* 2 hours/meeting = 89.5 hours.

1. Estimation for the whole project

Since different kinds of estimation has token different time and effort, we decided to calculate the weighted average of those three different results we got. The weight of estimation based on function points is 6, the weight of estimation based on pieces of functionality is 4, and the weight of estimation based on lines of codes is 3. So the result is:

(6 \* 762.4 hours + 4 \* 350 hours + 3 \* 620 hours) / (6 + 4 + 3) = 602.6 hours

**2.4 Project Resources**

People, hardware, software, tools, and other resources required to build the software are noted here.

1. People : Nesrin Mohammed, [Issam Jassim](https://umdearborn.instructure.com/courses/401476/users/944363), [Xiuyan Xin](https://umdearborn.instructure.com/courses/401476/users/1546062), Yue Dai
2. Hardware :
   1. Intel based laptop and desktop
   2. Portable hard drive
3. Software :
   1. Operation system : Windows Server 2012, Windows 10, Mac
   2. Integrated development environment : JetBrains WebStorm 10, Eclipse Luna
   3. Documentation : Microsoft Office 2013
4. Tools :
   1. Software Configuration Management : GitHub
   2. Communication tools : Gmail, Slack
   3. Document sharing : Google Drive
   4. Others : Tiny Tools(function points calculator), Here(JavaScript API)

## 3. Risk Management

### 3.1 Project Risks

In order to quantize the risk, the number is assigned according to the possibility or seriousness of the risk. The risk analysis will be based mainly on three variables P, (Possibility of the risk), I(Impact of the risk) .

The definition of each variable mentioned above is listed as follows:

Risk Possibility (P)

H – High(8-10), M - Medium(4-7), L – Low(1-3)

Risk Impact (I)

C – Critical(9-10), H – High(7-8), M – Medium(4-6), L - Low (1-3)

The risk is listed as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Number | Risk | Risk Type | Risk Possibility/P | Risk Impact/I |
| 1 | The time that can be assigned to the project is not enough | Team risk | M4 | C9 |
| 2 | Requirement may change as the product is being produced | Requirement risk | L2 | C9 |
| 3 | Misunderstanding of the requirement | Requirement risk | L2 | H8 |
| 4 | It is possible that team member is sick or have important appointment to meet once in a while | Team risk | M5 | M5 |
| 5 | Misunderstanding or lack of cooperation between members of the team | Team risk | L3 | H8 |
| 6 | Redundancy in information that may cause delays | Control risk | M5 | L2 |
| 7 | Lack of team spirit, conflict resolution require management intervention | Team risk | L2 | M5 |
| 8 | Some of requirements may be beyond the experience of the program personnel | Team risk | M4 | L2 |
| 9 | The　metrics designed for controlling, managing and monitoring the project is not accurate | Control risk | M4 | L3 |
| 10 | The quality may decrease because of slow network speed | User risk | M4 | M5 |
| 11 | The website may not function well due to browser compatibility | User risk | M5 | M6 |
| 12 | The documentation may not be adequate to design, implement and test the system | Control risk | H9 | L2 |

Table 3-1

### 3.2 Risk Table

In order to further quantize the risk, the variable T(Total effect assessment) is added, as the basis for evaluating priority of the risks. We get T in the formula : T = P \* I. And the priority is categorized into 4 levels varying from 1 to 4. The smaller the number, the higher the priority. Besides, team members are assigned to the risks accordingly.

|  |  |  |  |
| --- | --- | --- | --- |
| Risk ID | Risk Effect | Priority | Risk Manager |
| 1 | 36 | 1 | Nesrin |
| 2 | 18 | 3 | Yue |
| 3 | 16 | 3 | Yue |
| 4 | 25 | 2 | Xiuyan |
| Risk ID | Risk Effect | Priority | Risk Manager |
| 5 | 24 | 2 | Xiuyan |
| 6 | 10 | 3 | Yue |
| 7 | 10 | 3 | Yue |
| 8 | 8 | 4 | Issam |
| 9 | 12 | 3 | Yue |
| 10 | 20 | 2 | Xiuyan |
| 11 | 30 | 1 | Nesrin |
| 12 | 18 | 3 | Yue |

Table 3-2 Risk table

### 3.3 Overview of Risk Mitigation, Monitoring, Management

1. The mitigation method is listed as follows.
2. Set minimum working hours so that enough time for the project is guaranteed.
3. Before change a plan originally decided happen, discuss the necessity, rate it between 0 to 100%. If more than 60% , execute the change.
4. Discuss over the requirement, doing related research to accurately grasp the idea of the requirement.
5. The contingency method is listed as follows.
6. If control risks take place (either inaccurate metric or inadequate document). Rate the inaccuracy 0~100%, according to assigned priority/ value to each part and the extent of completion/accuracy. If less than 50%, do part correction, else redesign the metric / document.
7. If requirement risks take place . When ambiguous, ask the TA/Professor for confirmation before move on the next step. When change, rate the effect of the change from 0~100%, according to scale, type(document change/ low priority or implementation change/ high priority). If less than 50%, assign the change to 1 group member, else all members take part.
8. If team risks take place. When lack of communication: change communication skills or compromise with product owner . When lack knowledge : learn from each other or learn online. When lack of team spirit : needs team lead and a strong team member to cooperate to change the atmosphere.
9. Discuss necessity of specific vender if Database Support goes wrong
10. When lack of time: re-prioritize the tasks.
11. The monitoring method is listed as follows.
12. Extra care on resources and control.
13. Check the project every Sunday and upload log file onto Google drive. (Log file contains the member who modifies the document/implementation, date of modification, problems that he/she has, suggestion)
14. Ask people outside our team to use our system every 2 weeks to get feedback from so that some unnoticed fault can be detected.

**4. Project Schedule**

**4.1 Project task set**

The following two tables show all tasks required to complete this project.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| TASK | Duration(days) | PRE | SUCC | DELIV | WHORESP | WHOWORK | COMP% |
| A | 1 | - | B,C | High-level estimation, additional functionality, meeting schedule, time-record plan | Sam | IY | 100% |
| B | 12 | A | D | Pressman’s project planning document, complete ER diagram | Sam | XY | 100% |
| C | 3 | A | D | Project introduction, estimation, risk management, schedule, staff organization, tracking and control, appendix | Sam | Y | 100% |
| D | 1 | B,C | E | Presentation of project 1 | Sam | IXY | N/A |
| E | 7 | D | F | Requirements | Sam | X | N/A |
| F | 7 | E | G | Build-ER diagram, design document | Sam | I | N/A |
| G | 21 | F | H,I | Butterfly tracking system | Sam | NIXY | N/A |
| H | 7 | G | J | Testing document | Sam | NX | N/A |
| I | 7 | G | J | Optimized butterfly tracking system | Sam | NI | N/A |
| J | 1 | H,I | K | Butterfly tracking system v1.0 | Sam | NIXY | N/A |
| K | 4 | J | - | New version of butterfly tracking system | Sam | NIXY | N/A |

Table 4-1 Tasks

The following table gives a description of each task ID which was used in table 4-1.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TASKID | A | B | C | D | E | F | G | H | I | J | K |
| Content | Start | Documentation | Planning | Presentation | Requirement | Design | Build | Test | Debug | Launch | Maintain |
| WHOwork | I | | X | N | Y |  |  |  |  |  |  |
| Member | Issam | | Xiuyan | Nesrin | Yue |  |  |  |  |  |  |

Table 4-2 Task ID

**4.2 Functional decomposition**

1. Start :
   1. A high-level estimation of the whole project and the part on of the project;
   2. Four additional functionalities for every team member and one additional functionality for the whole team;
   3. A meeting schedule for the future discussion;
   4. A time record plan, who will be responsible for the time recording.
2. Documentation :
   1. Write the Pressman’s project planning document which consists of Introduction, project estimates, risk management, project schedule, staff organization, tracking and control mechanisms, and appendix;
   2. By the end of the documentation, make another detailed estimate of the whole project and the part one of the project.
3. Planning :
   1. Draw a complete ER diagram of our project
   2. Discussion about the content of the Pressman’s project planning document;
   3. Prepare the presentation for the part one of the project.
4. Presentation :
   1. Give a presentation of what we have done and how did we deal with the problems that came up during the part one of the project;
   2. Submit the Pressman’s project planning document we write and the complete ER diagram we draw.
5. Requirement :
   1. Gather the requirements that are necessary and doable
   2. Analysis of the requirements we have
   3. Write the requirement document
6. Design :
   1. Design the database :
      1. The session control
      2. The butterfly table
      3. The report table
   2. Design the Browser/Server based system’s architecture
   3. Design the front-end application :
      1. The interface of the systems
      2. The web pages that both users and visitors can visit
      3. The display of the results that come from the back-end
   4. Design the back-end application;
      1. The reactions toward the requests that come from the front-end
      2. The interaction with the database
   5. Design the interaction methods between the front-end and the back-end.
7. Build :
   1. Build the database :
      1. The session control table
      2. The butterfly table
      3. The report table
   2. Build the front-end application :
      1. The interface of the systems
      2. The web pages that both users and visitors can visit
      3. The display of the results that come from the back-end
   3. Build the back-end application;
      1. The reactions toward the requests that come from the front-end
      2. The interaction with the database
   4. Build the interaction between the front-end and the back-end.
8. Test :
   1. Do the white-box testing;
   2. Do the black-box testing;
   3. Write the testing report and documentation.
9. Debug :
   1. Fix the bug that comes with the white-box testing and the black-box testing
   2. Optimize the codes and the algorithm.
10. Launch :
    1. Launch the whole butterfly tracking system;
    2. Define the system version.
11. Maintain :
    1. Make sure the system is available during the workday;
    2. Gather reports of the system, and fix the bug in time.

**4.3 Task network**

The following diagram is shows a detail network diagram of each task associated with the project. Each rectangle represents a task to be completed. The Letter in the center of the rectangle is the task ID which sits above the number representing its duration. Refer to tables 4-1 and 4-2 for more details.

The numbers on each corner of rectangles represent the following:

Top Left: Earliest start  
Top Right: Earliest finish  
Bottom Left: Latest Start  
Bottom Right: Latest Finish

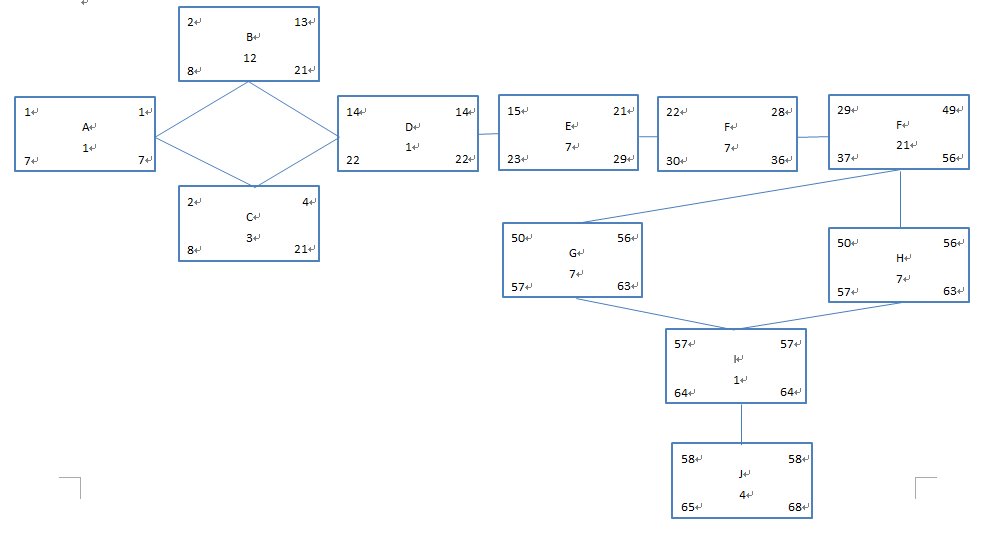


Figure 4-1 Task Diagram for BTS

**4.4 Timeline chart**

Figure 4-2 Timeline Chart

**5. Staff Organization**

**5.1 Team structure**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Role | Responsibility | Telephone | Email | Abbreviation |
| Issam | Team lead, | High-level estimation, Documentation, Design and build the back-end, Testing | (313)418-6900 | ijassim@umich.edu | I |
| Yue | front-end programmer | Building E-R diagram, Design and build the front-end ,Time-recording, design | (313)626-3185 | daiyu@umich.edu | Y |
| Xiuyan | front-end programmer | interaction, Presentation, Maintain, Design and build front-end | (313)642-3971 | xiuyanx@umich.edu | X |
| Nesrin | back-end programmer | Design and build database, testing document, activity diagram, complete ER diagram | (313)782-2814 | namohamm@umich.edu | N |

Table 5-1 Team Structure

**5.2 Management reporting and communication**

Mechanisms for progress reporting and inter/intra team communication are identified.

1. Regular meetings are held to ensure the team is of the same pace and responsibilities are equally shared.
2. To better communicate, some Internet tools are utilized. We are using Slack on mobile terminal, Google Drive for uploading and sharing files, Gmail for general low priority communication.
3. Weekly progress is reported and discussed among group members and is summarized by the team lead.
4. Team lead is responsible for the weekly report, describing where we are at, what we accomplished in the last week, and what we will do in the upcoming week, the status of our project in terms of red yellow and green, the issue we are encountering, and the solutions to deal with it. This report is to be submitted every Monday on canvas.
5. The content of each presentation is discussed in general and divided work is assigned to each member.
6. Presentation includes the past accomplishment and work in the future, the slides contains the process of the discussion by showing pictures and will present in general how the team handle the changes.

**6. Tracking and Control Mechanisms**

**6.1 Quality assurance and control**

Testing will be performed early in the development process and feedback will be given every step of the way. These tests will also focus on performance issues to ensure system resources are utilized efficiently. Our team’s quality assurance strategy is centered on continually evolving our testing process to prevent bugs from being created and ensure requirements are met as the software is developed. Extensive testing will be performed throughout the development phase to ensure that our software works as intended. The team lead, Issam Jassim, will be in charge of this process, and the following steps are in place to ensure the requirements are met:

1. The team lead will monitor the project closely and approve changes as they occur.
2. Progress status reports are presented during our regular periodic meetings as well as through our other means of communication.
3. The team lead will pay close attention to all testing results, and make changes as needed as quickly and reasonably as possible.

**6.2 Change management and control**

The team will use GitHub as a source code management system. GitHub keeps tack of all changes made by team members and maintains a change log. [3]This will ensure that all changes to the source code will be properly tracked and maintained. Using such an SCM will provide an easy way to monitor changes, as well as collaborate and download all any version of our source code if necessary. If the project requirements are changed at any time, we can back track to any version that’s more suitable to implement the new requirements.

**7.Appendix**

Supplementary information is provided here.

**7.1 Function points complexity matrix**

Data Element Type, or DET, a data element type is a unique, user recognizable, non-repeated field. This definition applies to both analyses of data functions and transactional functions.

Record Element Type, or RET, a record element type is a user recognizable subgroup of data elements within an Internal Logical File or External Interface File.

File type referenced, it can be either an Internal Logical File or an External Interface File

[1]See references in 7.4 for source

**7.1.1 Internal Logical Files**

|  |  |  |  |
| --- | --- | --- | --- |
| RETS | Data Element Types (DETs) | | |
|  | 1-19 | 20-50 | 51+ |
| 1 | 7 | 7 | 10 |
| 2 to 5 | 7 | 10 | 15 |
| 6 or more | 10 | 15 | 15 |

Table 7-1

**7.1.2 External Interface Files**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Data Element Types (DETs) | | |
| RETS | 1-19 | 20-50 | 51+ |
| 1 | 5 | 5 | 7 |
| 2 to 5 | 5 | 7 | 10 |
| 6 or more | 7 | 10 | 10 |

Table 7-2

**7.1.3 External Inputs**

|  |  |  |  |
| --- | --- | --- | --- |
| FTR's | Data Element Types (DET's) | | |
|  | 1-4 | 5-15 | 16+ |
| 0-1 | 3 | 3 | 4 |
| 2 | 3 | 4 | 6 |
| 3 or more | 4 | 6 | 6 |

Table 7-3

**7.1.4 External Outputs**

|  |  |  |  |
| --- | --- | --- | --- |
| FTR | Data Element Types (DET) | | |
|  | 1-5 | 6-19 | 20+ |
| 0-1 | 4 | 4 | 5 |
| 2-3 | 4 | 5 | 7 |
| 4 or more | 5 | 7 | 7 |

Table 7-4

**7.1.5 External Inquiries**

|  |  |  |  |
| --- | --- | --- | --- |
| FTRs | Data Element Types (DETs) | | |
|  | 1-5 | 6-19 | 20+ |
| 0-1 | 3 | 3 | 4 |
| 2-3 | 3 | 4 | 6 |
| 4 or more | 4 | 6 | 6 |

Table 7-5

**7.2 Activity Diagram**

The diagram below shows the activity diagram which shows the overall flow of control in the Butterfly Tracking System.

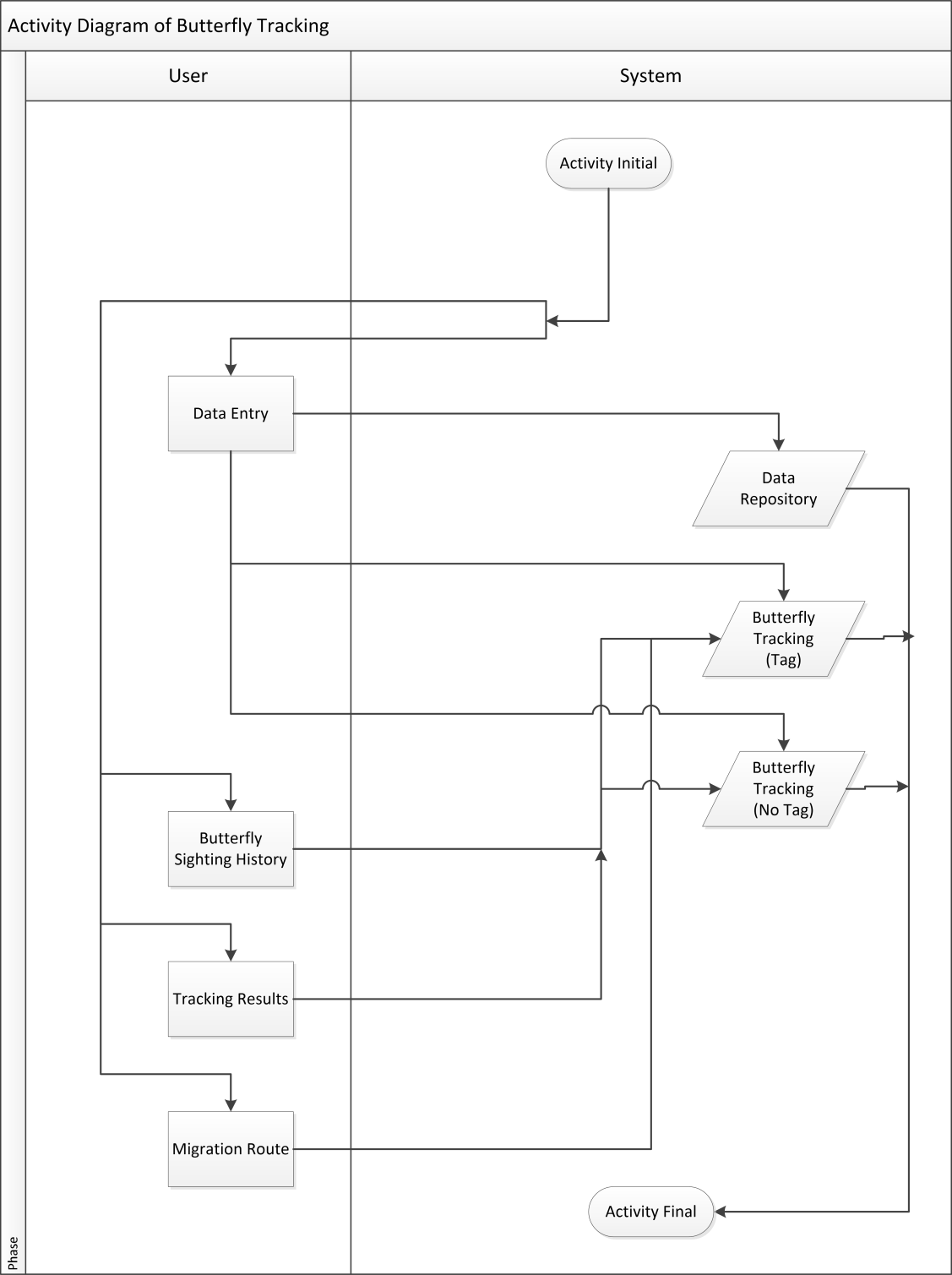


Figure 7-1 Activity Diagram for BTS

**7.3 Entity Relationship Diagram**

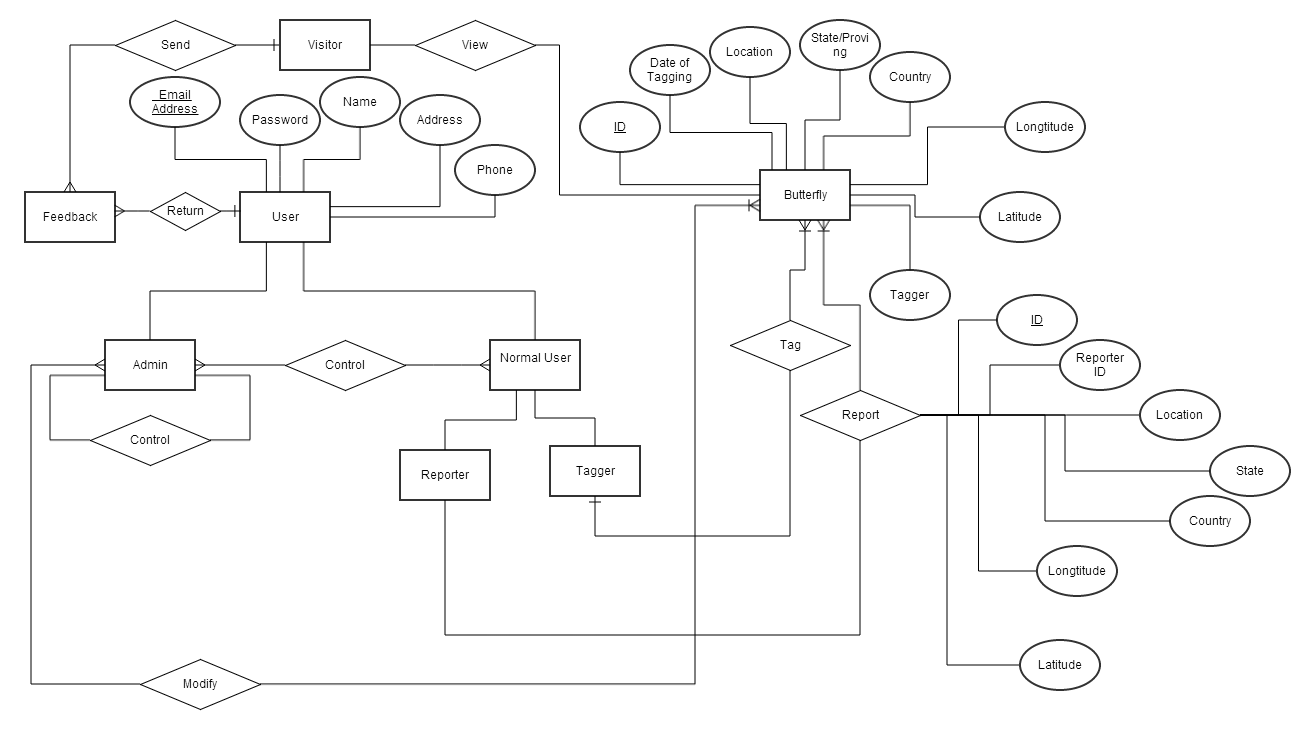


Figure 7-2 Entity Relation Diagram for BTS/ Version 1.2(11/23/2015)

**7.4 References**

[1] Alvin J. Alexander, “How to Determine Your Software Application Size Using Function Point Analysis”, <http://alvinalexander.com/FunctionPoints/FunctionPoints.shtml> [Oct.22, 2015].

[2] Software Engineering Tiny Tools, “FP CALCULATOR”, [http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/harvey/FP\_Calc.html](http://groups.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/harvey/FP_Calc.html%20) [Nov.20, 2015].

[3] Scott Chacon,  Ben Straub, “Pro Git Book”, <http://alvinalexander.com/FunctionPoints/FunctionPoints.shtml> [Oct.22, 2015].