

**CAPSTONE PROJECT II**

CMU-SE 451

**UCP CALCULATION**

Version 1.0

Date: March 26th, 2021

**FACIAL IDENTIFICATION MATCHING**

**Submitted by**

**Tran, Tran Thi Thuy**

**Han, Hoang Gia Bao**

**Trang, Tran Thi Huyen**

**Kieu, Dinh Thi Van**

**Vi, Pham Ha**

**Approved by**

**Trang, Nguyen Thi Bao M.Sc.**

Proposal Review Panel Representative:

Name Signature Date

Capstone Project 2- Mentor:

Name Signature Date

|  |  |  |  |
| --- | --- | --- | --- |
| **PROJECT INFORMATION** | | | |
| **Project acronym** | FIM System | | |
| **Project title** | Facial Identification Matching web application | | |
| **Start date** | March 1st, 2021 | **End Date** | Jun 2th, 2021 |
| **Lead institution** | International School, Duy Tan University | | |
| **Project mentor** | Trang, Nguyen Thi Bao M. Sc  Email: nguyenthibaotrang@gmail.com  Phone: 0915 774 711 | | |
| **Product owner & Contact Detail** | Trang, Nguyen Thi Bao M. Sc  Email: nguyenthibaotrang@gmail.com  Phone: 0915 774 711 | | |
| **Partner organization** | Duy Tan University | | |
| **Scrum Master** | Tran, Tran Thi Thuy | tranthuy1944@gmail.com | 090 505 7851 |
| **Team members** | Han, Hoang Gia Bao | baohanhoang2606@gmail.com | 093 563 8848 |
| Trang, Tran Thi Huyen | tranthihuyentrang7251@gmail.com | 037 495 7251 |
| Kieu, Dinh Thi Van | dinhvankieu199@gmail.com | 089 985 2540 |
| Vi, Pham Ha | Phamhavi255@gmail.com | 032 657 7627 |

|  |  |  |  |
| --- | --- | --- | --- |
| **UCP CALCULATION DOCUMENT** | | | |
| **Document Title** | UCP Calculation Document | | |
| **Author(s)** | HIT LIST Team | | |
| **Role** |  | | |
| **Date** | March 26th, 2021 | **File name:** | [FIM]UCPCalculation\_ver1.0 |
| **URL** |  | | |
| **Access** | Project and CMU Program | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **REVISION HISTORY** | | | |
| **Version** | **Person(s)** | **Date** | **Description** |
| **1.0** | All members | March 26th, 2021 | Draft for comment |

|  |  |  |  |
| --- | --- | --- | --- |
| **SIGNATURE**  **Document Approvals:** The following signatures are required for approval of this document. | | | |
| **Mentor & Product owner** | Trang, Nguyen Thi Bao M. Sc | **Signature:** |  |
| **Date:** |  |
| **Scrum master** | Tran, Tran Thi Thuy | **Signature:** |  |
| **Date:** |  |
| **Team member(s)** | Han, Hoang Gia Bao | **Signature:** |  |
| **Date:** |  |
| Trang, Tran Thi Huyen | **Signature:** |  |
| **Date:** |  |
| Kieu, Dinh Thi Van | **Signature:** |  |
| **Date:** |  |
| Vi, Pham Ha | **Signature:** |  |
| **Date:** |  |

**Contents**

[1. Purpose 6](#_Toc67822859)

[2. Scope 6](#_Toc67822860)

[3. History of UCP 6](#_Toc67822861)

[4. Purpose of UCP 6](#_Toc67822862)

[5. Components of UCP 6](#_Toc67822863)

[6. Step of UCP Calculation 9](#_Toc67822864)

[6.1. Unadjusted Use Case Weight (UUCW) 9](#_Toc67822865)

[6.2. Unadjusted Actor Weight (UAW) 9](#_Toc67822866)

[6.3. Adjusting for Technical Complexity 10](#_Toc67822867)

[6.4. Use Case Point Calculation 10](#_Toc67822868)

[7. Reference 11](#_Toc67822869)

1. **Purpose**

* This paper is aimed to introduce the Function Point Analysis (FPA), a technique for measuring size of a system from a user’s point of view.

1. **Scope**

* This document gives a concise picture of the important points of FPA (Function Point Analysis).
* In brief, this document firstly introduces FPA and then discusses the major steps in counting Function Point

1. **History of UCP**

The UCP technique was developed by Gustav Kerner in 1993 while employed at what was known at the time as Objectory Systems, which later merged into Rational Software and then IBM. The UCP method was created to estimate the software size of systems that were object-oriented. It is based on similar principles as the Function Point (FP) estimation method but was designed for the specific needs of object-oriented systems and system requirements based on use cases [1][2][3].

1. **Purpose of UCP**

Use case points (UCP or UCPs) is a software estimation technique used to forecast the software size for software development projects. UCP is used when the Unified Modeling Language (UML) and Rational Unified Process (RUP) methodologies are being used for the software design and development. The concept of UCP is based on the requirements for the system being written using use cases, which is part of the UML set of modeling techniques. The software size (UCP) is calculated based on elements of the system use cases with factoring to account for technical and environmental considerations. The UCP for a project can then be used to calculate the estimated effort for a project.

1. **Components of UCP**

* **Unadjusted Use Case Weight (UUCW)**

The UUCW is one of the factors that contribute to the size of the software being developed. It is calculated based on the number and complexity of the use cases for the system. To find the UUCW for a system, each of the use cases must be identified and classified as Simple, Average or Complex based on the number of transactions the use case contains. Each classification has a predefined weight assigned. Once all use cases have been classified as simple, average or complex, the total weight (UUCW) is determined by summing the corresponding weights for each use case. The following chart shows the different classifications of use cases based on the number of transactions and the weight value assigned for each use case within the classification.

|  |  |  |
| --- | --- | --- |
| **Use Case Classification** | **No. of Transactions** | **Weight** |
| Simple | 1 to 3 transactions | 5 |
| Average | 4 to 7 transactions | 10 |
| Complex | 8 or more transactions | 15 |

**UUCW** **= (Total No. of Simple Use Cases x 5) + (Total No. Average Use Cases x 10) + (Total No. Complex Use Cases x 15)**

* **Unadjusted Actor Weight (UAW)**

The UAW is another factor that contributes to the size of the software being developed. It is calculated based on the number and complexity of the actors for the system. Similar to finding the UUCW, each of the actors must be identified and classified as Simple, Average or Complex based on the type of actor. Each classification also has a predefined weight assigned. The UAW is the total of the weights for each of the actors. The following chart shows the different classifications of actors and the weight value assigned.

|  |  |  |
| --- | --- | --- |
| Actor Classification | Type of Actor | Weight |
| Simple | External system that must interact with the system using a well-defined API | 1 |
| Average | External system that must interact with the system using standard communication protocols (e.g., TCP/IP, FTP, HTTP, database) | 2 |
| Complex | Human actor using a GUI application interface | 3 |

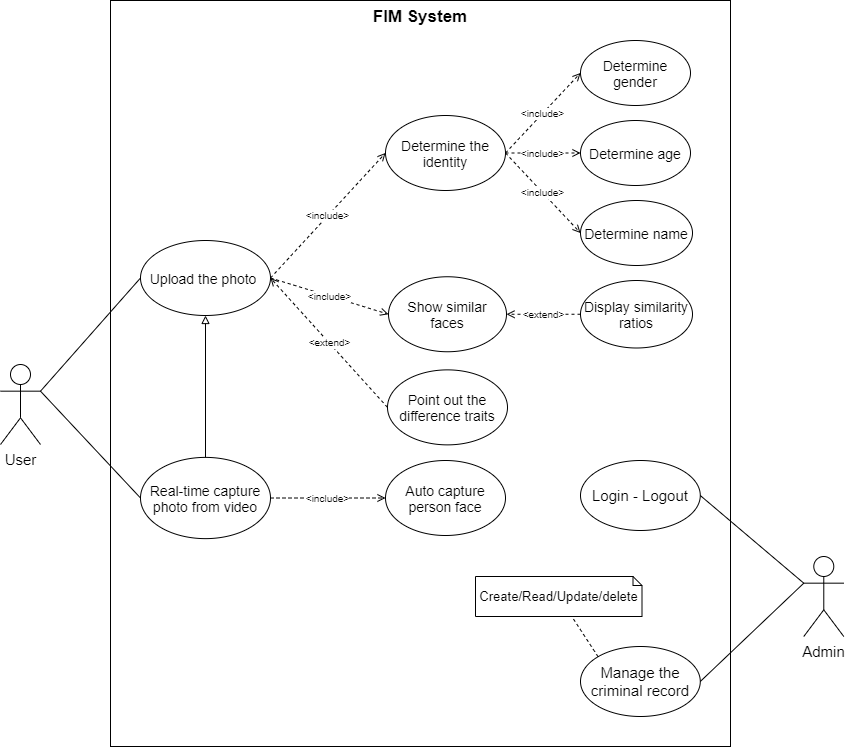
**UAW = (Total No. of Simple actors x 1) + (Total No. Average actors x 2) + (Total No. Complex actors x 3)**

* **Adjusting for Technical Complexity (VAF)**

One of the factors applied to the estimated size of the software in order to account for technical considerations of the system. It is determined by assigning a score between 0 (factor is irrelevant) and 5 (factor is essential) to each of the 13 technical factors listed in the table below. This score is then multiplied by the defined weighted value for each factor. The total of all calculated values is the technical factor (TF). The TF is then used to compute the TCF with the following formula:

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Description** | **Weight** | **Assessment** |
| F1 | Data Communications | 1 | 4 |
| F2 | Distributed Data Processing | 3 | 4 |
| F3 | Performance | 2 | 2 |
| F4 | Heavily Used Configuration | 2 | 2 |
| F5 | Transaction Rate | 1 | 4 |
| F6 | Online Data Entry | 2 | 5 |
| F7 | End-User Efficiency | 1 | 5 |
| F8 | Online Update | 1 | 0 |
| F9 | Complex Processing | 1 | 5 |
| F10 | Reusability | 1 | 5 |
| F11 | Installation Ease | 0.5 | 5 |
| F12 | Operational Ease | 0.5 | 5 |
| F13 | Multiple Sites | 1 | 2 |
| F14 | Facilitate Change | 1 | 0 |

1. **Step of UCP Calculation**
   1. **Identifying Use Case**



* 1. **Unadjusted Use Case Weight (UUCW)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case complexity** | **Weight** | **Number of Use cases** | **Product** |
| Simple | 5 | 8 | 40 |
| Average | 10 | 6 | 60 |
| Complex | 15 | 1 | 15 |
| **Total** |  | | **115** |

* 1. **Unadjusted Actor Weight (UAW)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Actor Classification** | **Weight** | **Number of Actors** | **Product** |
| Simple | 1 | 5 | 5 |
| Average | 2 | 2 | 4 |
| Complex | 3 | 2 | 6 |
| **Total** |  | | **15** |

* 1. **Adjusting for Technical Complexity (VAF)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Factor** | **Description** | **Weight** | **Assessment** | **Impact** |
| F1 | Data Communications | 1 | 2 | 2 |
| F2 | Distributed Data Processing | 3 | 2 | 6 |
| F3 | Performance | 2 | 2 | 4 |
| F4 | Heavily Used Configuration | 2 | 0 | 0 |
| F5 | Transaction Rate | 1 | 3 | 3 |
| F6 | On-line Data Entry | 2 | 1 | 2 |
| F7 | End-User Efficiency | 1 | 1 | 1 |
| F8 | On-line Update | 1 | 0 | 0 |
| F9 | Complex Processing | 1 | 3 | 3 |
| F10 | Reusability | 1 | 2 | 2 |
| F11 | Installation Ease | 0.5 | 0 | 0 |
| F12 | Operational Ease | 0.5 | 1 | 0.5 |
| F13 | Multiple Sites | 1 | 0 | 0 |
| F14 | Facilitate Change | 1 | 0 | 0 |
|  | **Total** | | | **23.5** |

**VAF = 0.65 + 0.01\*23.5 = 0.885**

* 1. **Use Case Point Calculation**

**UCP = (UUCW + UAW) \* VAF**

**= (115 + 15) \* 0.885**

**= 115.3**

* The project has **115.3** use case points:
* The team will average between **15 and 20 hours per use case point**
* A total of 5 developers (programmers, testers, DBAs, designers, etc.) will work on this project
* In this case, the complete project will take between **1726 hours** and **2301 hours** to complete (115.3\*15 = 1726 and 115.5\* 20 = 2301).
* We estimate that each developer will spend about **20 hours per week** on project tasks. The rest of their time will be sucked up by corporate overhead answering email, attending meetings, and so on.
* With 5 developers, this means the team will make **5 \* 20 = 120 hours per week.** Dividing 1726 hours by 120 hours and rounding up indicates that the overall project might take 14 (14.4) weeks.
* Dividing 2301 by 120 hours and rounding up indicates that it might take 19 (19.2) weeks.
* Our estimate is then that this project will take between **14 and 19 weeks**.

1. **Reference**

[1] Murali Chemuturi, Software Estimation Best Practices, Tools and Techniques for Software Project Estimators, J.Ross Publishing, 2009, p. 84-87

[2] Dennis, Alan R., Barbara Haley Wixom, and David Tegarden. Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach, Third Edition, John Wiley & Sons, 2009, Chapter 5 - Functional Modeling

[3] Dennis, Alan R., Barbara Haley Wixom, and David Tegarden. Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach, Fourth Edition, John Wiley & Sons, 2012, Chapter 2 - Project Management