Requirement & Function Size Report

of

EXAM PILOT

An Automated Examination Management System for NSTU

Prepared by

MIR MOHAMMAD TAHSIN | MUH2025007M

IMTIAZ CHOWDHURY | MUH2025027M

IRFANUL HAQUE | ASH1925021M

Institute of Information Technology

Noakhali Science and Technology University

Group Assignment of

SE 3204 - Software Metrics Lab

Submitted to

Md Hasan Imam

Lecturer

Institute of Information Technology

Noakhali Science and Technology University

Date of Submission: 26 December 2023

Contents

1. Project Information

Project Name	**EXAMPILOT**
Supervised By	**Md Hasan Imam** Lecturer Institute of Information Technology(IIT) Noakhali Science and Technology University
Team Members	**Mir Mohammad Tahsin** (MUH2025007M)
	Imtiaz Chowdhury (MUH2025027M)
	Irfanul Haque (ASH1925021M)
GitHub Link	**EXAMPILOT**
SRS Document	**Software Requirement & Specification Report**

2. Requirement Size

2.1 Definition: Requirement size refers to the complexity and details found in documents that outline what a project needs. These documents, known as requirements and specifications, use a mix of written text, graphs, and special mathematical diagrams and symbols. How they're presented depends on the chosen style, method, or notation.

While measuring the size of code or design is straightforward, with elements like lines, statements, bytes, classes, and methods to count, requirements documents are more varied. They can include a blend of text and diagrams. For instance, a use case analysis might involve a UML use case diagram and a set of scenarios presented as either text or UML activity diagrams. This diversity makes it challenging to come up with a single measure for the size of requirements.

2.2 Process : In different types of requirement and specification models, there are specific elements that can be counted:

- **Use case diagrams:** Counting the number of use cases, actors, and various types of relationships.
- **Use case:** Determining the number of scenarios and their size in terms of steps or activity

diagram elements.

- **Domain model (expressed as a UML class diagram):** Counting the number of classes, abstract classes, interfaces, roles, operations, and attributes.
- **UML OCL (Object Constraint Language) specifications**: Counting the number of OCL expressions and OCL clauses.
- **Alloy models:** Counting the number of alloy statements, including signatures, facts, predicates, functions, and assertions.
- **Data-flow diagrams used in structured analysis and design:** Counting processes (bubble nodes), external entities (box nodes), data-stores (line nodes), and data-flows (arcs).
- **Algebraic specifications:** Counting sorts, functions, operations, and axioms.
- **Z specifications:** Examining the various lines in the specification, whether they form part of a type declaration or a non-conjunctive predicate.

2.3 Values

Stakeholders :

Metric Type	Metric	Measurement	
Stakeholders	Number of stakeholders	**Type**	**Quantity**
		Student	30
		Exam Controller	2
		Teacher	10
		Administrator	3

Requirement Analysis :

Metric Type	Metric	Measurement
Functional & Non-Functional Requirements	Number of Functional Requirements	20
	Number of Non-Functional Requirements	25
Requirements Collection	Number of Persons Interacted	45
	Number of Interviews Taken	20
Use Case Diagram	Number of Use Cases	18
	Number of Actors	5
	Number of Relations Between Use Cases	12

	Number of Relations Between Use Case and Actors	22
	Number of Relations per Use Cases	1.83
Use Case Description	Number of Use Case Descriptions	18
	Number of Actors Per Use Case (Average)	3.6

Activity Diagram

Metric Type	Properties	Metric	
Activity diagram	Title	Number of steps	Number of branching actions
	Access Control	6	4
	Change Password	8	1
	Update Profile	6	4
	Send Request	7	4
	Receive Request	12	1
	Receive Question	9	2
	Authenticate User	7	3
	Enter Marks	6	2
	Enter Attendance	7	4
	Generate marksheet	7	4
	Generate Attendance Report	8	2
	Manage Faculty	6	1
	Manage Student	6	2
	Manage Department	6	1
	Manage Courses	6	1
	View Faculties	5	1
	Report Generate	4	2
	View Attendance Report	6	2
	Send Request for Approval	6	2
	Download Marksheet	6	2
	Approve Result Publication	7	3

Total Information

Metric Type	Metric	Measurement
Activity diagram	Total number of activity diagrams	22
	Total number of Steps	141
	Total number of branching actions	48
	Average Steps per activity diagram	6.4
	Average branching actions per activity diagram	2.18

3. Function Size

Function Points (FPs) offer a method for estimating the functional size of a system, independent of specific specification models. This technique, pioneered by Albrecht, relies on calculating a metric known as Unadjusted Function Point Count (UFC).

To compute UFC, we categorize system components into five types:

1. **External Inputs:** User-provided data elements like file names or menu selections (excluding inquiries).
2. **External Outputs:** System-generated data elements such as reports or messages.
3. **External Inquiries:** Interactive inputs requiring system responses.
4. **External Files:** Machine-readable interfaces to other systems.
5. **Internal Files:** Logical master files within the system.

Next, each item is assigned a subjective "complexity" rating on a three point ordinal scale: simple, average or complex.

3.1 Standard Functional Units with Weighted Factors

Functional Units	Weighting Factors		
	Simple	Average	Complex
External Inputs	3	4	6
External Outputs	4	5	7
External Inquiry	3	4	6
Internal Logical Files	7	10	15
External Interface Files	5	7	10

3.2 Function Point Calculation

Function Points	Quantity	Name	Complexity	Complexity Weight
External Inputs (EI)	8	User Login	Average	4
		Send Request	Complex	6
		Enter Marks	Complex	6
		Enter Attendance	Complex	6
		Download Marksheet	Average	4
		Send Question	Complex	6
External Outputs (EO)	14	Access Granted	Average	5
		Profile Updated	Average	5
		Password Changed	Average	5
		Question Received	Complex	7
		Request Received	Complex	7
		Question Moderated	Complex	7
		Marksheet Generated	Average	5
		Attendance Report Generated	Average	5
		Marksheet Downloaded	Average	5
		Questions Sent to Invigilators	Complex	7
		View Result	Average	5
		Subject Assignment Viewed	Average	5
		Notification Sent	Complex	7
		Attendance Recorded	Average	5
External Inquiry (EQ)	4	View Courses	Simple	3
		View Faculty	Simple	3
		View Attendance Report	Simple	3
		View Result	Simple	3
Internal Logical File (ELF)	6	User Database	Average	10
		Course Database	Average	10
		Faculty Database	Average	10

		Question Database	Complex	15
		Marksheet Database	Average	10
		Attendance Database	Complex	15
External Interface File (EIF)	3	Email system	Average	7
		Authentication system	Average	7
		Question moderation system	Complex	10
Unadjusted Function Point (UFC)				**214**

To complete our computation of FPs, we calculate an adjusted function point count, FP, by multiplying UFC by a technical complexity factor, TCF. Each component or sub factor is rated from0 to 5, where 0 means the sub factor is irrelevant, 3 means it is average, and 5 means it is essentialto the system being built.

3.3 Formula

$$TCF = [0.65 + 0.01 \times \sum(F_i)]$$

$$FP = UFC \times TCF$$

3.4 Technical Complexity Factor Rating (0-5) :

Rating	Description
0	No Influence
1	Incidental
2	Moderate
3	Average
4	Significant

5	Essential

3.5 Technical Complexity Factor Calculation

No.	Complexity Factors	Rating(0-5)
T1	Distributed System	4
T2	Response or Throughput Performance Objectives	3
T3	End-User Efficiency (Online)	4
T4	Complex Internal Processing	3
T5	Code Must be Re-usable	3
T6	Easy to Install	5
T7	Easy to Use	5
T8	Portable	2
T9	Easy to Change	4
T10	Concurrent	3
T11	Includes Special Security Features	5
T12	Provides Direct Access for Third Parties	1
T13	Special User Training Facilities are Required	2
T14	Scalability	3
Total		**47**

Technical Complexity Factor (TCF) = 0.65 + 0.01 * 44 = **1.12**

Function Point (FP) = Unadjusted Function Point (UFC) * Technical Complexity Factor (TCF)

= 214 * 1.12 = **239.68**

Comment :

A value of **239.68** for Function Points (FP) typically represents the size or complexity of a software project.

- **Low values (e.g., below 100):** Indicate a relatively small and less complex software project.
- **Moderate values (e.g., 100-200):** Suggest a project of medium size and complexity.
- **High values (e.g., above 200):** Indicate a large and complex software project.

A value of 239.68 FP suggests a project of substantial size and complexity. It implies that the software has a significant number of functionalities and interactions.

So The FP value of 239.68 indicates a relatively moderate level of complexity for ExamPilot project. While the system encompasses a good range of functionalities, it doesn't appear to be exceptionally large or intricate compared to other similar systems. This information can be helpful for project planning, resource allocation and effort estimation. Moreover, we can say that if we assign 1 day to implement 1 function point that the project will nearly 8 months.

3.6 Other Metrics Calculation based on FP:

From **Task : 3** (Code Structure & Code Size Measurement using Halstead approach) we get ,

Error = 1710.66

Error/FP = 1710.66/239.68 = **7.10**

From **Task : 1** (Effort & Cost Estimation) we get

Effort = 18 person-month

Cost = $18000

So,

Productivity = FP/ Effort = 239.68/18 = **13.32**

Cost Per Function = Cost / Productivity = 18000/13.32 = **$1351.35**

4. Conclusion

Key Findings:

- **Requirement Size:** The ExamPilot project exhibits a moderate level of requirement complexity, with a substantial number of stakeholders, use cases, and activity diagrams.
- **Function Points:** The system's functional size, measured as 239.68 FP, indicates a moderate level of complexity, suggesting a system with a significant number of functionalities and interactions.
- **Technical Complexity Factor:** The TCF of 1.12 reflects a moderate degree of technical complexity, with factors like distributed nature, performance objectives, end-user efficiency, and security features contributing to complexity.
- **Productivity:** The calculated productivity of 13.32 FP/person-month suggests potential areas for improvement in development processes or methodology.