PRACTICAL NO. 2

ESTIMATION OF PROJECT METRICS

AIM: In this experiment, we will learn how to estimate the cost, effort, and duration for developing a **Public Health Centre Database Management System.** Based on the analysis of estimation results, we will identify and select the most suitable development approach that aligns with the goals and constraints of the health centre.

INTRODUCTION:

Once all functional and non-functional requirements for the project are gathered, the next step is to evaluate potential solution strategies. Each strategy may vary in terms of implementation complexity, technology, and development process. Expert business analysts assess each option by examining its benefits and drawbacks, primarily through metrics such as cost, development time, and resource requirements. Accurate estimation is critical to ensure the project is delivered efficiently and supports the essential operations of the health centre.

OBJECTIVES:

After completing this experiment you will be able to:

* Categorize projects using COCOMO, and estimate effort and development time required for a project
* Estimate the program complexity and effort required to recreate it using Halstead's metrics

THEORY:

Project Estimation Techniques

A Public Health Centre Database Management System is more than just a software that stores data—it’s a comprehensive solution that manages workflows, reporting, and compliance in a healthcare setting. Such a system often comprises several modules (e.g. patient registration, diagnostics, medicine inventory, staff scheduling), and may take several months to complete depending on scope and complexity. To manage the project efficiently, it is essential to quantify and estimate the following:

* Project Size – e.g., number of modules, lines of code, database tables
* Cost – includes developer wages, tools/software licenses, and operational overhead
* Duration – time required to complete all development phases
* Effort – the total work input needed by all team members

COCOMO (Constructive Cost Model)

COCOMO is a widely used algorithmic software cost estimation model introduced by Barry Boehm. For the project, we need to categorize the system into one of the following types based on its complexity and team experience:

* Organic: If the PHC-DMS is a straightforward system with a small team and prior experience
* Semidetached: If the team has a mix of experienced and new members, and the system has moderate complexity
* Embedded: If the system integrates directly with hardware (e.g., IoT health devices) and is highly complex

This experiment focuses on two popular estimation models: COCOMO and Halstead’s metrics.

**Basic COCOMO Model**

The basic COCOMO model helps to obtain a rough estimate of the project parameters. It estimates effort and time required for development in the following way:

Effort = a \* (KDSI)b PM  
Tdev= 2.5 \* (Effort)c Months  
Person required= Effort/Tdev

where,

* KDSI is the estimated size of the software expressed in Kilo Delivered Source Instructions
* a, b, c are constants determined by the category of software project
* Effort denotes the total effort required for the software development, expressed in person months (PMs)
* Tdev denotes the estimated time required to develop the software (expressed in months)
* The value of the constants a, b, c are given below:

| **Software Project** | **a** | **b** | **c** |
| --- | --- | --- | --- |
| Organic | 2.4 | 1.05 | 0.38 |
| Semi-detached | 3.0 | 1.12 | 0.35 |
| Embedded | 3.6 | 1.20 | 0.32 |

**Intermediate COCOMO**

Takes into account additional 15 cost drivers, such as reliability, complexity, team capability, tool usage, etc. Each cost driver has a rating from “very low” to “extra high,” and their product gives the Effort Adjustment Factor (EAF):

Effort = Basic Effort × EAF  
Tdev = 2.5 × (Effort)^c

This model helps make more realistic estimates for the PHC-DMS based on the development environment.

**Complete COCOMO**

Recognizes that Public Health Centre Database Management System might have heterogeneous components (e.g., a GUI frontend, a SQL-based database, and a possible mobile app interface). Each part may fall under a different COCOMO category (e.g., GUI = Organic, DB = Semidetached, Mobile integration = Embedded). In such cases, separateestimates for each component are calculated and combined.

**Advantages of Using COCOMO for PHC-DMS**

* Easy to use for early-stage project planning
* Helps determine team size, schedule, and cost
* Provides quantitative backing to project decisions

**Drawbacks of COCOMO**

* Uses KDSI, which is hard to estimate early
* Based on the waterfall model, less suited for agile/incremental development
* Doesn’t consider reuse of existing modules (common in health IT)
* Not suitable for projects with extensive use of third-party APIs or low-code platforms

COCOMO II improves on these issues and is recommended for modern development approaches.

**Halstead’s Complexity Metrics**

This method estimates complexity and effort based on the actual code structure focusing on operators and operands used in the program.

For example, in the **PHC-DMS**, the patient registration module may contain input forms, validation logic, and database insert operations. Each of these involves operators (e.g., if, =, +) and operands (e.g., name, age, patient\_id, constants).

Given:

* n1 = number of unique operators
* n2 = number of unique operands
* N1 = total number of operators
* N2 = total number of operands

We calculate:

* Program Vocabulary: n = n1 + n2
* Program Length: N = N1 + N2
* Volume: V = N × log2(n)
* Difficulty: D = (n1 × N2) / (2 × n2)
* Effort: E = D × V
* Time to Implement: T = E / 18 (seconds)

Halstead’s metrics provide a language-independent measure of how difficult it is to build or maintain a module in PHC-DMS, useful for code review and refactoring efforts.

CASE STUDY

A Public Health Centre Database Management System

The Public Health Centreserves a large population by managing patient records, appointments, staff scheduling, inventory of medicines, and reporting of health data. As the demand and size of the health centre’s operations grow, it has become necessary to develop a comprehensive Database Management System to improve efficiency and accuracy.

The PHC-DMS will allow health centre staff—including doctors, nurses, and administrative personnel—to access and update patient information seamlessly. Patients can have their appointments scheduled or rescheduled easily, and pharmacy inventory can be tracked in real time. The system will also generate reports for government health programs and disease surveillance. The health centre’s IT management team, comprising a small group of experienced healthcare IT professionals and software engineers, has been tasked with developing the PHC-DMS. Given the team's expertise and the relatively moderate size of the project, the system can be categorized under the organic project type according to COCOMO classification.

Based on expert judgment and prior similar projects, the estimated project size is approximately 12 KLOC (thousand lines of code). This estimate will serve as the foundation for project metric estimation using the Basic COCOMO model.

For the **organic** category, the COCOMO constants are:

* a = 2.4
* b = 1.05
* c = 0.38

Effort calculation: Effort = a × (KDSI)^b = 2.4 × (12)^1.05 ≈ 33 person-months (PM)

Estimated development time: Tdev = 2.5 × (Effort)^c = 2.5 × (33)^0.38 ≈ 9.2 months

Thus, the project would ideally require around **33 person-months** and approximately **9 months** to complete.

**Complete COCOMO Consideration**

The PHC-DMS consists of multiple modules including:

* **Patient Records Module** (organic type)
* **Inventory Management Module** (semi-detached type)
* **Reporting and Analytics Module** (embedded type due to integration with external health databases)

The **Complete COCOMO model** would require estimating effort and time for each module separately and then aggregating these to get a more precise project estimate. This approach is especially important because the modules differ in complexity and resource needs.