EXPERIMENT NO. 8

Aim: Conduct Function Point Analysis (FPA) for the project.

Theory:

Function Point Analysis was initially developed by Allan J. Albrecht in 1979 at IBM and has been further modified by the International Function Point Users Group (IFPUG). The initial definition is given by Allan J. Albrecht.

Functional Point Analysis gives a dimensionless number defined in function points which we have found to be an effective relative measure of function value delivered to our customer.

Objectives of Functional Point Analysis:

- The objective of FPA is to measure the functionality that the user requests and receives.
- The objective of FPA is to measure software development and maintenance independently of the technology used for implementation.
- It should be simple enough to minimize the overhead of the measurement process.
- It should be a consistent measure among various projects and organizations.

Characteristics of Functional Point Analysis:

1. **External Input (EI):** EI processes data or control information that comes from outside the application's boundary. The EI is an elementary process.
2. **External Output (EO):** EO is an elementary process that generates data or controls information sent outside the application's boundary.
3. **External Inquiries (EQ):** EQ is an elementary process of input-output combination that results in data retrieval.
4. **Internal Logical File (ILF):** A user-identifiable group of logically related data or control information maintained within the boundary of the application.
5. **External Interface File (EIF):** A group of users recognisable logically related data allusion to the software but maintained within the boundary of another software.

Calculating the Count based on all three factors

Weighing Factor is assumed to be **Simple**

Information Domain Value	**Count**		**Weighing Factor**				**Total**
			Simple	**Average**	**Complex**		
No. of External Inputs (EIs)	**4**	*	**3**	4	6	=	***12***
No. of External Outputs (EOs)	**7**	*	**4**	5	7	=	***28***
No. of External Inquiries (EQs)	**2**	*	**3**	4	6	=	***6***
No. of Files (IFs)	**2**	*	**7**	10	15	=	***14***
No. of External Interfaces (EIFs)	**2**	*	**5**	7	10	=	***10***
					Count	=	***70***

Weighing Factor is assumed to be **Average**

Information Domain Value	**Count**		**Weighing Factor**				**Total**
			Simple	**Average**	**Complex**		
No. of External Inputs (EIs)	**4**	*	3	**4**	6	=	***16***
No. of External Outputs (EOs)	**7**	*	4	**5**	7	=	***35***
No. of External Inquiries (EQs)	**2**	*	3	**4**	6	=	***8***
No. of Files (IFs)	**2**	*	7	**10**	15	=	***20***
No. of External Interfaces (EIFs)	**2**	*	5	**7**	10	=	***14***
					Count	=	***93***

Weighing Factor is assumed to be **Complex**

Information Domain Value	**Count**		**Weighing Factor**				**Total**
			Simple	**Average**	**Complex**		
No. of External Inputs (EIs)	**4**	*	3	4	**6**	=	***24***
No. of External Outputs (EOs)	**7**	*	4	5	**7**	=	***49***
No. of External Inquiries (EQs)	**2**	*	3	4	**6**	=	***12***
No. of Files (IFs)	**2**	*	7	10	**15**	=	***30***
No. of External Interfaces (EIFs)	**2**	*	5	7	**10**	=	***20***
					Count	=	***135***

Convention for measuring Adjustment factors

0	–	No Influence
1	–	Incidental
2	–	Moderate
3	–	Average
4	–	Significant
5	–	Essential

14 Adjustment Factors:

1. ***Backup & Recovery = 5***

   Definition: Operational Ease measures the complexity of operating and managing the software. It encompasses factors like backup, recovery, and system monitoring.

   Justification: In UrbanGarden, the need for robust backup and recovery systems is critical.UrbanGarden stores sensitive customer data, order history, and product information. A high rating in this factor reflects the importance of ensuring data integrity and availability, especially in the event of system failures or data breaches.

2. ***Data Communications = 4***

   Definition: It measures the complexity of data exchange between the application and external entities.

   Justification: UrbanGarden often requires extensive data exchange with external entities, including payment gateways, shipping providers, and third-party integrations. This complexity in data communications arises from the need to securely transmit and receive data between various systems, which can affect the overall complexity of the project.

3. ***Distributed Data Processing = 4***

   Definition: It evaluates the complexity resulting from processing data across multiple locations.

   Justification: UrbanGarden involves distributed data processing to handle user requests, inventory management, and real-time order updates. Distributing these processes across multiple locations or servers can introduce complexities related to data consistency, latency, and synchronisation.

4. ***Performance Critical = 4***

   Definition: It assesses the performance requirements and constraints of the system.

   Justification: In any e-commerce platform, performance is crucial. Customers expect fast page load times and quick responses during the shopping and checkout process. Ensuring the application meets these performance criteria can require additional development effort and complexity, such as optimising code and database queries.

5. ***Existing Operating Environment = 3***

   Definition: The "Existing Operating Environment" in software development refers to the pre-existing technological infrastructure, including hardware, software systems, and network components, that serves as the foundation for a new software application.

   Justification: Applications like UrbanGarden are often built upon existing operating environments, including web servers, databases, and hosting infrastructure. While this can provide a foundation for development, it may also introduce some constraints or dependencies that need to be considered, impacting the complexity to a moderate extent.

6. ***Online Data Entry = 3***

   Definition: It evaluates the complexity of user interfaces for data entry and updates.

   Justification: Online data entry refers to user interfaces for entering data, such as customer registration forms and address input during checkout. In any e-commerce app, these forms need to be user-friendly, efficient, and capable of handling various data inputs. While they are important, their complexity is moderate compared to other aspects of the application.

7. ***Input transaction over multiple screens = 4***

   Definition: It measures the number of business transactions or operations the system must support.

   Justification: In UrbanGarden, complex transactions often involve multiple steps and screens. For example, the checkout process may include steps for selecting products, entering shipping details, choosing payment methods, and confirming orders. Handling these multi-screen transactions requires careful design and implementation, contributing to the complexity.

8. ***Master Files updated online = 3***

   Definition: It measures the complexity of real-time data updates and maintenance.

   Justification: Updating master files online relates to managing product catalogues, inventory levels, and other critical information in real-time. While important, this complexity is moderate as it involves data management and synchronisation but doesn't typically require highly complex processing.

9. ***Information domain values Complex = 2***

   Definition: "Information Domain Values (IDV) Complex" in Function Point Analysis (FPA) refers to the degree of complexity associated with intricate calculations, transformations, or business rules that the software performs on data or information within the application.

   Justification: Information domain values (IDV) complexity may be relatively lower in e-commerce applications compared to other domains. IDV complexity typically pertains to intricate calculations or data processing, which are less prominent in the core functionality of e-commerce apps.

10. ***Internal Processing Complex = 3***

    Definition: It evaluates the complexity of business logic and algorithms.

    Justification: Internal processing complexity in any e-commerce application can arise from complex business rules, pricing calculations, tax calculations, and inventory management algorithms. These complexities, while essential, are generally of moderate magnitude.

11. ***Code Designed for Reuse = 4***

    Definition: It measures the extent to which existing software components can be reused.

    Justification: Designing code for reuse in UrbanGarden can reduce development time and effort. Reusable components can include payment integrations, user authentication modules, and shopping cart functionality, contributing to the overall project complexity.

12. ***Conversion/Installation in design = 5***

    Definition: It assesses the complexity of installing the software in various environments.

    Justification: In UrbanGarden, migrating or setting up a new platform often requires data conversion, especially if you are transitioning from an old system. The complexity of handling data migration and system installation can be significant, hence the higher rating.

13. ***Multiple Installations = 4***

    Definition: It considers the complexity introduced when the system operates in multiple geographical locations.

Justification: E-commerce platforms may require installations across multiple geographical locations or for various clients. Managing and configuring these installations while ensuring consistency can add complexity to the project.

14. ***Application designed for change* = 3**

Definition: It measures the ease with which the software can be modified or adapted.

Justification: The E-commerce sector is a dynamic field with changing customer preferences and industry trends. Designing the application to accommodate future changes or enhancements is important but generally of moderate complexity. It involves maintaining a balance between flexibility and stability.

Calculating Functional Point Analysis for each of the three weighing factors

1. **Function Point (FP) for Simple:**

   (FP) = Total Count * [0.65 + 0.01 * ΣFi]
   = 70 * [0.65 + 0.01 * 51]
   = 81.2

2. **Function Point (FP) for Average:**

   (FP) = Total Count * [0.65 + 0.01 * ΣFi]
   = 93 * [0.65 + 0.01 * 51]
   = 107.88

3. **Function Point (FP) for Complex:**

   (FP) = Total Count * [0.65 + 0.01 * ΣFi]
   = 135 * [0.65 + 0.01 * 51]
   = 156.6

Calculate the FOUR parameters

(Effort, Productivity, Cost per Function Point, and Cost)

Let's calculate the four parameters (Effort, Productivity, Cost per Function Point, and Cost) based on the given formulae and the provided values:

1. **Effort**:
   - Estimated Cost = Let's assume the estimated cost is 10,000/-.
   - Labour Rate = Let's assume the labour rate is 50/- per hour.

   Effort = Estimated Cost / Labour Rate
   Effort = 10,000 / 50
   Effort = 200 hours

2. **Productivity**:
   - Function Points (FP) for each type (Simple, Average, Complex) are already calculated.
   Productivity = FP / Effort

   For Simple:
   Productivity (Simple) = 81.2 / 200
   Productivity (Simple) = 0.406 FP/hour

   For Average:
   Productivity (Average) = 107.88 / 200
   Productivity (Average) = 0.5394 FP/hour

   For Complex:
   Productivity (Complex) = 156.6 / 200
   Productivity (Complex) = 0.783 FP/hour

3. **Cost per Function Point**:
   - Cost per Function Point (for each type) can be calculated *as the reciprocal of Productivity.*

   Cost per Function Point (Simple) = 1 / Productivity (Simple)
   Cost per Function Point (Simple) = 1 / 0.406
   Cost per Function Point (Simple) ≈ 2.46/FP

   Cost per Function Point (Average) = 1 / Productivity (Average)
   Cost per Function Point (Average) = 1 / 0.5394
   Cost per Function Point (Average) ≈ 1.85/FP

   Cost per Function Point (Complex) = 1 / Productivity (Complex)

Cost per Function Point (Complex) = 1 / 0.783
Cost per Function Point (Complex) ≈ 1.28/FP

4. **Cost**:
- Function Points (FP) for each type are already calculated.

*Cost = FP * Cost per Function Point*

For Simple:
Cost (Simple) = 81.2 * 2.46/FP
Cost (Simple) ≈ 199.57

For Average:
Cost (Average) = 107.88 * 1.85/FP
Cost (Average) ≈ 199.69

For Complex:
Cost (Complex) = 156.6 * $1.28/FP
Cost (Complex) ≈ 200.45

So, based on the provided values and calculations:

1. **Effort**:
   - Simple: 200 hours
   - Average: 200 hours
   - Complex: 200 hours

2. **Productivity**:
   - Simple: 0.406 FP/hour
   - Average: 0.5394 FP/hour
   - Complex: 0.783 FP/hour

3. **Cost per Function Point**:
   - Simple: 2.46/FP
   - Average: 1.85/FP
   - Complex: 1.28/FP

4. **Cost**:
   - Simple: 199.57/-
   - Average: 199.69/-
   - Complex: 200.45/-