Department of Computer Engineering

Academic Term: First Term 2023-24

Class: T.E /Computer Sem – V / Software Engineering

Practical No:	4
Title:	Calculating function points of the Project
Date of Performance:	17/8/2023
Roll No:	9641
Team Members:	Prashant Singh(9641), Mohtashim Ali(9644), Mark Tuscano(9645), Sania Tuscano(9646)

Rubrics for Evaluation:

Sr. No	Performance Indicator	Excellent	Good	Below Average	Total Score
1	On time Completion & Submission (01)	01 (On Time)	NA	00 (Not on Time)	
2	Theory Understanding(02)	02(Correct)	NA	01 (Tried)	
3	Content Quality (03)	03(All used)	02 (Partial)	01 (rarely followed)	
4	Post Lab Questions (04)	04(done well)	3 (Partially Correct)	2(submitted)	

Signature of the Teacher:

Prashant Singh 9641

TE comps-B

Lab Experiment 4

Experiment Name: Calculating Function Points of the Project in Software Engineering

Objective: The objective of this lab experiment is to introduce students to the concept of Function Points and the Function Point Analysis (FPA) technique for measuring software size and complexity. Students will gain practical experience in calculating Function Points for a sample software project, enabling them to estimate development effort and assess project scope accurately.

Introduction: Function Points are a unit of measurement used in software engineering to quantify the functionality delivered by a software application. Function Point Analysis (FPA) is a widely used technique to assess the functional size of a software project based on its user requirements.

Lab Experiment Overview:

1. Introduction to Function Points: The lab session begins with an overview of Function Points, explaining the concept and their significance in software size measurement.
2. Defining the Sample Project: Students are provided with a sample software project along with its user requirements. The project may involve modules, functionalities, and user interactions.
3. Identifying Functionalities: Students identify and categorize the functionalities in the sample project, such as data inputs, data outputs, inquiries, external interfaces, and internal logical files.
4. Assigning Complexity Weights: For each identified functionality, students assign complexity weights based on specific criteria provided in the FPA guidelines.
5. Calculating Unadjusted Function Points: Students calculate the Unadjusted Function Points (UFP) by summing up the weighted functionalities.
6. Adjusting Function Points: Students apply adjustment factors (e.g., complexity, performance, and conversion) to the UFP to calculate the Adjusted Function Points (AFP).
7. Estimating Development Effort: Using historical data or industry benchmarks, students estimate the development effort required for the project based on the calculated AFP.

8. Conclusion and Reflection: Students discuss the significance of Function Points in software estimation and reflect on their experience in calculating Function Points for the sample project.

Learning Outcomes: By the end of this lab experiment, students are expected to:

- Understand the concept of Function Points and their role in software size measurement.
- Gain practical experience in applying the Function Point Analysis (FPA) technique to assess software functionality.
- Learn to categorize functionalities and assign complexity weights in Function Point calculations.
- Develop estimation skills to assess development effort based on calculated Function Points.
- Appreciate the importance of accurate software size measurement in project planning and resource allocation.

Pre-Lab Preparations: Before the lab session, students should familiarize themselves with the concept of Function Points and the guidelines for their calculation. They should review the categorization of functionalities and the complexity weighting factors used in Function Point Analysis.

Materials and Resources:

- Project brief and details for the sample software project
- Function Point Analysis guidelines and complexity weighting criteria
- Calculators or spreadsheet software for performing calculations

Conclusion: The lab experiment on calculating Function Points for a software project provides students with a practical approach to estimating software size and complexity. By applying the Function Point Analysis (FPA) technique, students gain insights into the importance of objective software measurement for project planning and resource allocation. The hands-on experience in identifying functionalities and assigning complexity weights enhances their estimation skills and equips them with valuable techniques for effective project management. The lab experiment encourages students to apply Function Point Analysis in real-world scenarios, promoting accuracy and efficiency in software size measurement for successful software engineering projects.

CALCULATIONS:

User Input (user login, feedback, user details (payment, address etc.)) = 3

Enquiry (product searches, product details) = 2

User Output (order confirmation, recommendations, order tracking) = 3

Interface = 3

User file (customer data storage, payment information, product dataset) = 3

FPA = UFP*CAF

FPA matrix

	COUNT	LOW	AVERAGE	HIGH
USER INPUT	3	3	4	6
ENQUIRY	2	3	4	6
OUTPUT	3	4	5	7
INTERFACE	3	5	7	10
USER FILE	3	7	10	15

CALCULATE UFP:

UFP = 3*3 + 2*3 + 3*4 + 3*5 + 3*10

=72

CALCULATE CAF:

CAF=0.65+(0.01*14*S)

=0.65+(0.01*14*3) S=3

=0.65+(0.01*42)

=1.07

FPA = UFP*CAF

= 72*1.07

=77.04

=**77**(approx.)

Postlabs

Q1. Critically evaluate the Function Point Analysis method as a technique for software sizing and estimation, discussing its strengths and weaknesses?

Function Point Analysis (FPA) is a widely used software sizing and estimation technique that has been in existence since the 1970s. It is based on the premise that the functionality delivered by a software system can be quantified in terms of function points, which can then be used to estimate effort, cost, and duration. However, like any methodology, FPA has its strengths and weaknesses, which should be carefully considered when deciding whether to use it for software sizing and estimation.

Strengths of Function Point Analysis:

- **Technology and Language Agnostic**: FPA is technology and programming language agnostic, which means it can be applied to various types of software projects regardless of the underlying technology stack. This is a significant advantage as it allows for consistent estimation across different technologies.
- **Focus on User Perspective**: FPA places a strong emphasis on capturing the functionality from a user's perspective. This user-centric approach ensures that the software's value to the end-user is central to the estimation process.
- **Objective Sizing**: FPA provides a relatively objective and standardized way to size software. Function points are determined based on a set of defined rules, reducing the subjectivity often associated with other estimation methods.
- **Historical Data**: Over time, organizations can build up historical data on function points and the effort required to deliver them. This data can be valuable for benchmarking and improving estimation accuracy.
- **Detailed Measurement**: FPA allows for a detailed measurement of software functionality, breaking it down into distinct components such as inputs, outputs, inquiries, internal logical files, and external interface files. This granularity can help in fine-tuning the estimation process.

Weaknesses of Function Point Analysis:

- **Complexity and Learning Curve**: FPA can be complex and has a steep learning curve. Estimators need to be well-trained and experienced to apply FPA effectively. This can make it less accessible for smaller organizations or those with limited resources.
- **Time-Consuming**: The process of counting function points can be time-consuming, especially for large and complex projects. This can lead to delays in the estimation process and may not be suitable for agile or fast-paced development environments.
- **Subjectivity in Counting Rules**: While FPA is considered objective, there can still be subjectivity in applying the counting rules. Different analysts may interpret the rules differently, leading to variations in estimates.
- **Limited in Early Project Phases**: FPA is most accurate when detailed requirements are available. It may not be as effective in the early stages of a project when requirements are still evolving, as it relies on a clear understanding of the system's functionality.

- **Not Suitable for All Project Types**: FPA may not be the best choice for projects with a heavy emphasis on non-functional aspects like performance, security, or usability. It primarily focuses on functional requirements and may not capture the full complexity of some software systems.
- **Influence of External Factors**: FPA does not account for external factors that can significantly impact project size and effort, such as team experience, tools, or organizational process maturity.

In conclusion, Function Point Analysis is a well-established technique for software sizing and estimation, but it is not without its limitations. Its strengths lie in its technology agnosticism, user-centric approach, and the potential for historical data reuse. However, it can be complex, time-consuming, and subject to interpretation. Organizations should carefully assess their specific needs, the project's characteristics, and the availability of skilled personnel before adopting FPA as their primary estimation method or consider complementing it with other techniques to improve accuracy.

Q3. Propose strategies to manage and mitigate uncertainties in function point estimation and how they can impact project planning and resource allocation.

Managing and mitigating uncertainties in function point estimation is crucial for effective project planning and resource allocation. Uncertainties can arise from various sources, including unclear requirements, changing scope, and human factors. Here are some strategies to address these uncertainties and understand their impact on project planning and resource allocation:

- **Requirements Analysis and Refinement**:
- **Early Requirement Clarification**: Invest time and effort in thoroughly analyzing and clarifying requirements before beginning the estimation process. Engage stakeholders to ensure a shared understanding of project goals and scope.
- **Progressive Elaboration**: Recognize that requirements may evolve over time. Use progressive elaboration to refine and update function point estimates as the project progresses and more information becomes available.
- **Historical Data and Benchmarking**:
- **Reference Historical Data**: Leverage historical data from past projects to improve estimation accuracy. Compare similar projects in terms of function points, effort, and resource allocation to identify patterns and trends.
- **Benchmarking**: Compare your initial function point estimates with industry benchmarks or standards. This can help identify discrepancies and potential sources of uncertainty.
- **Expert Involvement**:
- **Experienced Estimators**: Assign experienced estimators or subject matter experts to the estimation process. Their expertise can help in making more informed judgments and reducing uncertainty.
- **Review and Peer Evaluation**: Conduct reviews and peer evaluations of the estimation process to identify potential biases or errors in the estimation process.
- **Contingency Planning**:
- **Include Contingency in Estimates**: Add contingency buffers to estimates to account for uncertainties. These buffers should be based on historical data and expert judgment and can be adjusted as the project progresses.
- **Risk Assessment**: Perform a comprehensive risk assessment to identify potential risks and uncertainties that can affect the project. Develop mitigation plans for high-impact risks.

- **Sensitivity Analysis**:
- **Sensitivity Testing**: Conduct sensitivity analysis to assess the impact of different assumptions on the estimation. Identify the most critical variables and evaluate how changes in these variables affect project outcomes.
- **Agile and Iterative Approaches**:
- **Adopt Agile Methods**: Agile methodologies, such as Scrum or Kanban, allow for flexibility in project scope and can better accommodate changing requirements and uncertainties. This approach can mitigate the impact of uncertainties on project planning.
- **Regular Monitoring and Control**:
- **Tracking Actuals**: Continuously monitor project progress and compare actuals to estimated values. This helps in identifying discrepancies and adjusting resource allocation as needed.
- **Change Management**: Implement robust change management processes to handle scope changes and uncertainties effectively. Ensure that any changes are properly assessed and their impacts are communicated to stakeholders.
- **Documentation and Communication**:
- **Clear Documentation**: Document the assumptions, constraints, and uncertainties associated with function point estimates. Maintain clear and transparent records to facilitate communication with stakeholders.
- **Stakeholder Communication**: Keep stakeholders informed about the uncertainties and potential impacts on project planning and resource allocation. Regularly update them on changes to estimates and associated risks.

Uncertainties in function point estimation can have a significant impact on project planning and resource allocation. By employing these strategies, organizations can better manage and mitigate these uncertainties, resulting in more accurate estimates and improved project outcomes. Effective estimation and uncertainty management contribute to better resource allocation, reduced project risks, and increased project success rates.