



Developing Software Cost Estimation Model for Research and Community Service Dashboard Monitoring Applications

Thessa Karinda Araynawa^{1,*} and Renny Sari Dewi²

^{1,2} Digital Business, Universitas Negeri Surabaya, Indonesia

*) Corresponding Author (Email: thessa.22113@mhs.unesa.ac.id)

ABSTRACT

Software cost estimation is an important stage in software development planning because inaccurate estimation may lead to inefficient resource allocation and budget management. This study aims to develop a software cost estimation model by integrating the Use Case Points (UCP) and Database Complexity Index (DBCI) methods for a Research and Community Service (RCS) monitoring dashboard based on an Executive Information System (EIS). This study uses a qualitative descriptive approach through system requirement analysis, use case identification, database structure analysis, and software metric calculation. The proposed model combines functional complexity measured using UCP and database complexity measured using DBCI to produce a more comprehensive estimation result. The results show that the developed RCS dashboard has a UCP value of 55.071 and a DBCI value of 246. These values are integrated into the proposed UCPDBCI model, resulting in a value of 301.071. Based on productivity assumptions, the estimated development effort is 6,021.42 working hours. By applying pay rate calculations based on software development roles, the estimated development cost is Rp631,222,722. The proposed model provides an alternative approach for estimating software development costs by considering both functional and database complexity, particularly for dashboard-based information systems in higher education environments.

Keywords: *software cost estimation; Use Case Points; Database Complexity Index; Executive Information System; RCS dashboard*

I. INTRODUCTION

Digital transformation has encouraged organizations, including higher education institutions, to utilize information systems to support operational activities, monitoring processes, and strategic decision-making. Higher education



institutions manage various activities that generate large amounts of data, including academic activities, research, and community service. Research and Community Service (RCS) activities represent important responsibilities of universities because they contribute to scientific development and community empowerment. However, the increasing complexity of RCS activities requires an integrated information system that can provide accurate, timely, and accessible information for monitoring and evaluation processes.

Executive Information System (EIS) is one type of information system that supports decision-making by presenting summarized information through dashboards and visualization features. EIS enables decision-makers to monitor organizational performance indicators and analyze information from multiple data sources (Azima et al., 2018). In higher education environments, dashboard-based EIS can assist stakeholders in monitoring research performance, funding distribution, publication achievements, and community service activities. However, developing an EIS dashboard requires appropriate planning, particularly regarding software development effort and cost estimation.

Software cost estimation remains a challenging activity due to the complexity of software development processes and the uncertainty associated with project requirements, resources, and development environments. Recent studies have investigated various approaches to improve estimation accuracy, including systematic reviews of cost drivers, hybrid estimation models, optimization-based approaches, and data-driven methods. Ahmed et al. (2023) identified that software cost estimation models are influenced by multiple cost drivers, while Rashid et al. (2023) emphasized the importance of developing more adaptive estimation approaches to address limitations of conventional methods. Furthermore, recent studies have explored hybrid and machine learning-based models to improve prediction performance in software cost estimation (Ahmed et al., 2024a; Vanathi et al., 2024; Zhao et al., 2025).

Among software metric-based estimation approaches, Use Case Points (UCP) has been widely applied because it can estimate software size based on functional requirements during the early development phase. Previous studies have demonstrated that UCP can provide valuable estimation results by analyzing actors, use cases, and adjustment factors. Azzeh et al. (2021) reported that UCP remains one of the commonly investigated approaches for software effort estimation, while Prayitno et al. (2020) demonstrated the application of UCP combined with additional estimation techniques to improve effort prediction. Moreover, (Silhavy et al., 2023) adapted the UCP methodology for different software domains, showing its flexibility in supporting software estimation.

Database complexity is another factor that influences software development effort because database structures affect implementation difficulty, data management, and system maintenance. Database Complexity Index (DBCi) is a measurement approach that evaluates database complexity based on structural

components, including tables, attributes, keys, and relationships (Dey et al., 2014). Previous research has shown that database complexity measurement can provide additional information regarding software complexity. Juhola (2022) highlighted that data complexity measurement is important for understanding the characteristics and difficulty of managing complex data structures. However, DBCI only represents the database aspect and does not consider the functional complexity perceived by system users.

Existing studies indicate that software estimation methods commonly evaluate software complexity from a single perspective. UCP focuses on functional requirements, while DBCI focuses on database structure. However, software systems consist of interconnected functional and data components, meaning that considering only one aspect may result in incomplete estimation. Although several studies have investigated hybrid approaches for software cost estimation (Ahmed et al., 2024a; Rashid et al., 2023), research integrating functional complexity measurement using UCP and database complexity measurement using DBCI remains limited.

Therefore, this study addresses the research gap by proposing a hybrid software cost estimation model that integrates Use Case Points (UCP) and Database Complexity Index (DBCI), referred to as the UCPDBCI model. The proposed model is applied to an Executive Information System-based Research and Community Service monitoring dashboard. This study aims to develop a software cost estimation model by (1) calculating software complexity using UCP and DBCI methods, (2) integrating both estimation approaches into a proposed model, and (3) estimating software development effort and cost based on the proposed model.

II. LITERATURE REVIEW

Research and Community Service Information System

Research and Community Service (RCS) activities are fundamental components of higher education institutions that contribute to knowledge development, innovation, and social improvement. These activities involve various processes, including research proposal submission, evaluation, funding management, publication tracking, and monitoring of community service outcomes. The increasing complexity of RCS activities generates large amounts of heterogeneous data that require effective information management to support institutional decision-making. Research Information Management Systems (RIMS) and Current Research Information Systems (CRIS) have been developed to integrate various research-related information, including researcher profiles, research projects, publications, and institutional research outputs. These systems enable higher education institutions to manage research activities more effectively and support research evaluation processes (Donner, 2023; Fernandes, 2018).

Information systems have become essential tools for managing academic activities by providing integrated data processing and information delivery mechanisms. In higher education environments, information systems support operational activities as well as strategic decision-making by transforming raw data into meaningful information for stakeholders. The implementation of information management systems in universities allows institutions to improve information accessibility, optimize administrative processes, and support data-driven management practices (Triono et al., 2023). Furthermore, effective information management requires appropriate system design to ensure that institutional information can be collected, processed, and distributed according to stakeholder requirements (Fajriyah et al., 2024).

Executive Information System (EIS) is one type of information system designed to provide summarized and strategic information for decision-makers. EIS commonly utilizes dashboards and visualization techniques to present key performance indicators and assist management in monitoring organizational performance (Velcu-Laitinen & Yigitbasiglu, 2012). Dashboard-based decision support systems have been increasingly implemented in higher education institutions because they provide stakeholders with better access to critical information and support evidence-based decision-making (McCoy & Rosenbaum, 2019). Dashboards allow complex institutional data to be presented in a visual and understandable format, enabling decision-makers to identify trends, evaluate performance, and formulate strategic actions (Yoke Wah et al., 2019).

In the context of RCS management, an EIS-based dashboard can support university leaders and research management units in evaluating research productivity, funding distribution, publication performance, and community service achievements. The utilization of intelligent decision support systems and dashboard technologies in higher education demonstrates the potential of information visualization to improve managerial decision-making and institutional performance monitoring (Sharma, 2017; Ulfatriyani et al., 2020).

The development of an RCS EIS dashboard requires appropriate planning, particularly in managing the resources and costs required during the software development process. Information systems in higher education continue to face increasing complexity due to growing data volume, diverse user requirements, and the need for integrated institutional services. Therefore, proper planning and management approaches are required to ensure that information system development can be implemented effectively and efficiently (Sukorini et al., 2024).

Software Cost Estimation

Software Cost Estimation (SCE) is a fundamental activity in software engineering that aims to predict the required effort, resources, and financial cost of software development projects. Accurate estimation is essential because inaccurate predictions may result in budget overruns, schedule delays, and inefficient resource

allocation. Recent research has emphasized that software estimation remains challenging due to various influencing factors, including project characteristics, development environment, and software complexity (Ahmed et al., 2023; Haleem et al., 2025).

Various approaches have been developed to improve software cost estimation accuracy, ranging from traditional algorithmic models to optimization and machine learning-based approaches. Draz et al. (2024) proposed a predictive estimation model using ARIMA algorithms, demonstrating the potential of statistical approaches in software cost prediction. Similarly, Gandomani et al. (2025) improved analogy-based software cost estimation through optimization techniques, while Zhang et al. (2024) applied dimensionality reduction and machine learning methods to enhance estimation performance.

Although machine learning and optimization approaches have shown promising results, these approaches often require sufficient historical project datasets. In projects with limited historical data, metric-based estimation methods remain valuable because they estimate software complexity based on measurable system characteristics. Ahmed et al. (2024) proposed a hybrid estimation model to improve software cost prediction by combining multiple estimation perspectives, indicating that integrating different complexity dimensions can enhance estimation reliability.

One widely used software metric-based approach is Use Case Points (UCP), which estimates software size based on functional requirements represented through actors and use cases. However, software complexity is not only influenced by functional requirements but also by internal system structures, including database design. Therefore, integrating functional and structural complexity measurements can provide a more comprehensive estimation approach.

Use Case Points (UCP)

Use Case Points (UCP) is a software size estimation method introduced by Karner (1993) to estimate software development effort based on system functionality represented through actors and use cases. The method has received considerable attention because it can be applied during the early stages of software development when detailed implementation information is not yet available. Azzeh et al. (2021) conducted a systematic review and confirmed that UCP is widely used for software effort prediction due to its capability to represent functional requirements.

The UCP method consists of several calculation components, including Unadjusted Actor Weight (UAW), Unadjusted Use Case Weight (UUCW), Technical Complexity Factor (TCF), and Environmental Complexity Factor (ECF). UAW represents actor complexity based on the interaction level between users and the system. Meanwhile, UUCW measures use case complexity based on the number of

transactions involved in each use case. The technical and environmental factors are used to adjust the estimation based on system characteristics and development conditions (Karner, 1993).

The general calculation process begins by determining the Unadjusted Use Case Points (UUCP), which is obtained from the combination of UAW and UUCW. The UUCP value is then adjusted using TCF and ECF to produce the final Use Case Points value. The resulting UCP value can subsequently be converted into estimated effort by applying a productivity factor.

Several studies have applied and improved UCP in different software development contexts. Prayitno et al. (2020) combined UCP with the Program Evaluation and Review Technique (PERT) to estimate software effort under uncertainty conditions. Furthermore, Silhavy et al. (2023) adapted the UCP methodology for Internet of Things projects, demonstrating that UCP can be modified according to specific software characteristics.

Recent research has also focused on optimizing UCP-based estimation models. Ardiansyah et al. (2024) improved UCP estimation through metaheuristic algorithms, indicating that optimization techniques can enhance the accuracy of traditional UCP calculations.

Database Complexity Index (DBCI)

Database complexity represents an important aspect of software development because database structures influence system implementation, maintenance activities, and data processing performance. Complex data structures require additional effort in database design, integration, and management. Juhola (2022) emphasized that measuring data complexity is necessary to understand the characteristics of complex datasets and support effective information management.

Database Complexity Index (DBCI) is an approach used to measure database complexity based on structural characteristics. Unlike UCP, which focuses on functional complexity, database complexity measurement provides a structural perspective by evaluating internal system components. Therefore, integrating database complexity with functional-based estimation can provide a more comprehensive representation of software complexity. This integration approach aligns with recent trends in software estimation research that combine multiple factors to improve estimation accuracy (Ahmed et al., 2024).

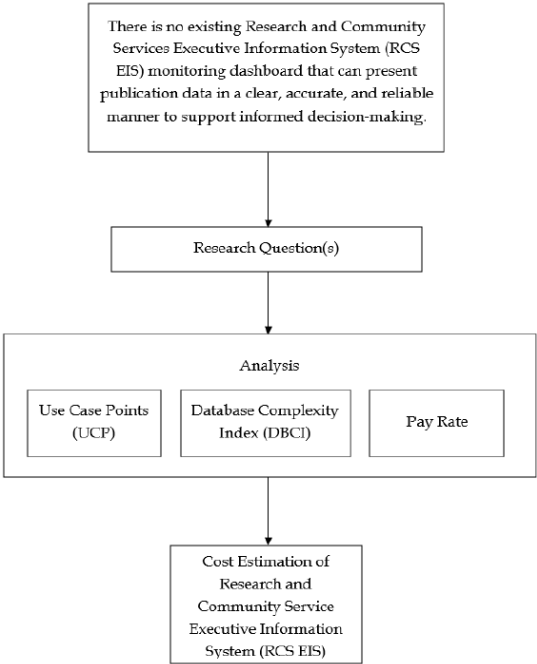
Database complexity measurement provides additional information regarding software development effort because database-related activities contribute significantly to software development processes. A system with a large number of entities and complex relationships generally requires more analysis, design, and implementation effort compared with a simpler database structure.

Although database complexity influences software development effort, previous software estimation approaches have often focused primarily on functional aspects. Therefore, integrating DBCI with UCP provides a broader estimation

perspective by combining functional complexity and database structural complexity. The integration of these two measurements is expected to improve software cost estimation accuracy, particularly for information systems with significant data management requirements, such as Research and Community Service Executive Information System dashboards.

III. METHOD

This study employs a qualitative descriptive approach combined with a software metrics-based estimation method. The qualitative approach is used to interpret system specifications and structural components of the software, while the quantitative aspect is derived from numerical estimation using software metrics.



Source: Developed by Author, 2026

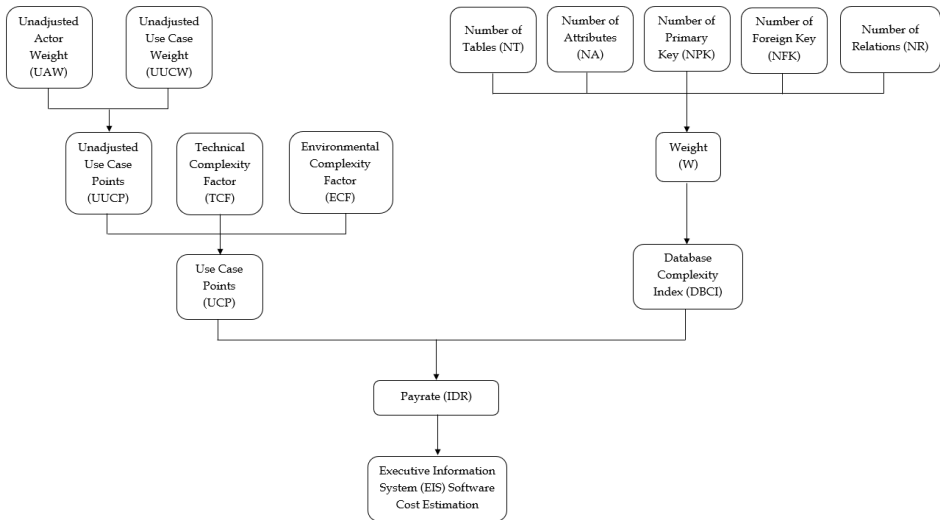
Figure 1. Research Framework for Software Cost Estimation of RCS EIS Dashboard

Figure 1 explains the research framework, consists of several stages, including problem identification, research question formulation, software complexity analysis, and cost estimation.

The estimation framework integrates two complementary methods: Use Case Points (UCP) and Database Complexity Index (DBCI). UCP is used to determine the functional size of the system based on actors and use cases, while DBCI measures the complexity of the database structure based on tables, attributes, primary keys, foreign keys, and relationships.

The integration of UCP and DBCI forms the proposed UCPDBCI model, which aims to improve the accuracy of early-stage software cost estimation, particularly for Executive Information System (EIS) development. The final output of the model is software development effort (in working hours) and cost estimation (in Indonesian Rupiah).

The proposed estimation model used in this research is illustrated in Figure 2.



Source: Developed by Author, 2026

Figure 2. Proposed Estimation Model

Figure 2 explains that obtaining the UCP, DBCI and the combination requires calculation using the following formula.

$$UCP = (UAW + UUCW) \times TCF \times ECF \tag{1}$$

$$DBCI = 5NT + 2NA + 10NPK + 5NFK + 3NR \tag{2}$$

$$UCPDBCI = UCP + DBCI \tag{3}$$

The integration of both metrics produces the UCPDBCI value, which is then converted into development effort and cost estimation using a predefined pay rate. This approach allows estimation to capture both functional system complexity (UCP) and data structure complexity (DBCI).

Then, the UCPDBCI is converted into effort using the following formula:

$$Effort = UCPDBCI \times 20 \tag{4}$$

The productivity factor of 20 person-hours was retained from Karner's original UCP method because the objective of the proposed UCPDBCI model is to enhance software size estimation by incorporating database structural complexity,

rather than to establish a new productivity model. Therefore, the integration of DBCI affects the measurement of software complexity, while the effort conversion process continues to follow the established productivity assumption proposed by Karner. Since no empirically validated productivity factor has yet been developed for the UCPDBC model, the original factor of 20 person-hours was adopted as a baseline for effort estimation. Future research should calibrate this productivity factor using actual software development projects to improve the empirical applicability of the model.

Once the effort is determined, it is converted into a pay rate to estimate the software cost.

IV. RESULTS AND DISCUSSION

Use Case Points (UCP) Calculation Results

The calculation of Use Case Points (UCP) begins with the identification of system actors and use cases based on the functional requirements of the Executive Information System (EIS) for Research and Community Service (RCS) monitoring. The identified actors and use cases are used as the basis for determining the Unadjusted Actor Weight (UAW) and Unadjusted Use Case Weight (UUCW).

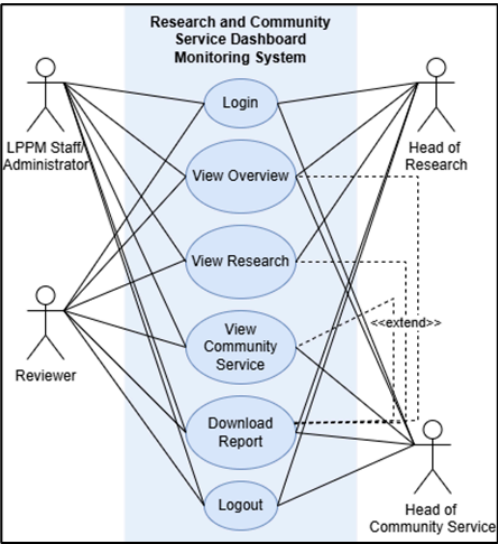


Figure 3. Identification of Actors and Use Cases for UCP Calculation

Based on the identified actors and use cases presented in Figure 3, each actor and use case is categorized according to its complexity level. The assigned weights are then used to calculate UAW and UUCW. The summary of the UCP calculation process and final results is presented in Table 1.

Table 1. UCP Calculation

Component	Calculation Basis	Value
Unadjusted Actor Weight (UAW)	Actor Categorize	12
Unadjusted Use Case Weight (UUCW)	Use Case Categorize	60
Unadjusted Use Case Points (UUCP)	UAW+UUCW	72
Technical Complexity Factor (TCF)	$0,6+(0,01 \times TF)$	1,055
Environmental Complexity Factor (ECF)	$1,4+(-0,03 \times EF)$	22,5
Final UCP Value	$UUCP \times TCF \times ECF$	55,071

Source: Developed by Author, 2026

The result indicates that the system has a moderate level of functional complexity, reflecting its role as a decision-support dashboard with multiple monitoring and reporting features.

Database Complexity Index (DBCI) Calculation Results

The DBCI calculation starts with the analysis of the database structure. The database schema is evaluated based on structural components, including tables, attributes, primary keys, foreign keys, and relationships.

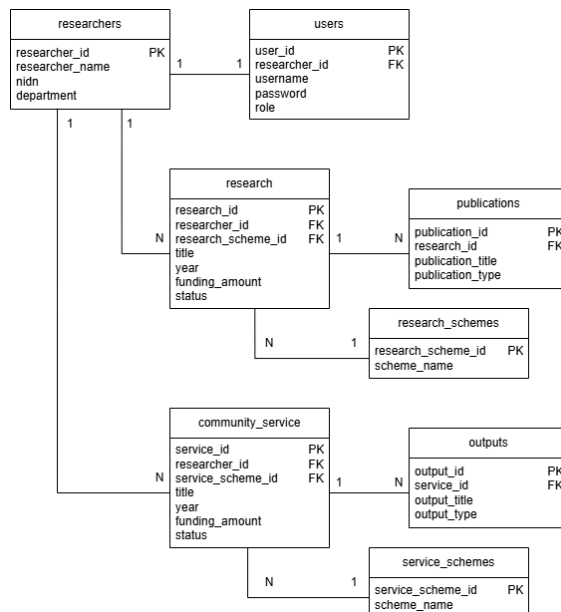


Figure 4. Conceptual Data Model for DBCI Calculation

Based on the database structure shown in Figure 4, the complexity score is calculated by assigning weights to each database component according to Equation 2. The summarized DBCI calculation results are presented in Table 2.

Table 2. DBCI Calculation

Component	Value
Number of Tables (NT)	8
Attributes (NA)	35
Primary Keys (NPK)	8
Foreign Keys (NFK)	7
Relationships (NR)	7
Final DBCI Value	246

Source: Developed by Author, 2026

The analysis shows that the system database has a moderate complexity level due to the interrelated structure between research activities, community service data, outputs, and budget information. This indicates that a relational database design is essential to ensure data consistency and integrity within the system.

Software Cost Estimation Results

After obtaining the Use Case Points (UCP) and Database Complexity Index (DBCI) values, the next stage is the software development cost estimation process using the proposed UCPDBCI model. The estimation process consists of three main stages: integrating UCP and DBCI values, converting the resulting value into development effort, and calculating the final software development cost using a weighted pay rate.

The integration of UCP and DBCI is performed to combine functional complexity and database structural complexity into a single estimation parameter. UCP represents the functional size of the system based on actors and use cases, while DBCI represents the complexity of the database structure. The combination of these two metrics produces the UCPDBCI value, which is subsequently used as the basis for effort estimation, as shown in Table 3.

Table 3. UCPDBCI Calculation

Component	Value
UCP	55,071
DBCI	246
Final UCPDBCI Value	301,071

Source: Developed by Author, 2026

The obtained UCPDBCI value is then converted into estimated development effort. The effort calculation uses a productivity rate of 20 work hours per UCPDBCI unit, which represents the estimated amount of working time required to develop one unit of software complexity. The estimated effort is calculated by multiplying the

UCPDBCI by 20, resulting in a figure of 6,021.42 hours. The value of 20 represents the productivity hours per UCP, as proposed by Karner.

After obtaining the estimated effort, the software development roles were identified. The role is determined based on four software development roles commonly used in previous studies, which are, Project Manager, System Analyst, System Engineer, and Quality Assurance. Each role is assigned a weight based on its contribution to the software development process in a form of percentage. For the Project Manager is 34% and the rest is 22%.

After obtaining the role identification, next is calculated the pay rate. The pay rate is determined based on the monthly salary benchmark obtained from the PERSOL Indonesia Salary Guide 2026. The monthly salary is converted into an hourly rate using the assumption of 22 working days per month and 8 working hours per day. The resulting hourly pay rates are then weighted according to the contribution of each software development role and used to calculate the final software development cost.

After obtaining the pay rate for each role, the next stage is to distribute the total estimated effort into each software development role based on their respective contribution percentages. The total effort of 6,021.42 hours is allocated according to the predefined role weights, where 34% is assigned to the Project Manager and 22% is assigned to each of the System Analyst, System Engineer, and Quality Assurance roles. This allocation results in the calculated effort distribution for each role, reflecting their proportional involvement in the software development process.

Subsequently, the labour cost for each role is computed by multiplying the allocated effort by the corresponding hourly pay rate. This step produces the cost contribution of each role in the overall software development process. The total software cost estimation (SCE) is obtained by summing all labor costs from the four roles, resulting in the final estimated development cost of the system. The detailed calculation of effort allocation and labor cost for each role is presented in Table 6.

Table 4. Software Cost Estimation Calculation

Role	Percentage	Effort (Hours)	Allocated Effort (Hours)	Pay Rate (IDR/hour)	Labor Cost (IDR)
Project Manager	34%	6.021,42	2.047,28	170.455	348.968.659
System Analyst	22%	6.021,42	1.324,71	62.500	82.794.525
System Engineer	22%	6.021,42	1.324,71	68.182	90.321.300
Quality Assurance	22%	6.021,42	1.324,71	82.386	109.138.238
Total Software Cost Estimation					631.222.722

Source: Developed by Author; 2026

Discussion

The proposed UCPDBCI model estimated the total software development cost for the Research and Community Service (RCS) Dashboard System at Rp631,222,722 by integrating functional complexity measured through Use Case Points (UCP) and database structural complexity measured through the Database Complexity Index (DBCI). Unlike the conventional UCP method, which estimates software size primarily based on functional requirements, actors, use cases, and adjustment factors, the proposed model additionally incorporates database structural characteristics, including entities, attributes, primary keys, foreign keys, and relationships. This integration enables software complexity to be assessed from both functional and structural perspectives within a single estimation framework.

The findings indicate that database complexity represents an important aspect of software development effort that is not explicitly captured by the conventional UCP approach. In the proposed RCS Dashboard System, the database consists of eight entities, thirty-five attributes, eight primary keys, seven foreign keys, and seven relationships, indicating substantial development activities related to database schema design, relationship modelling, data integrity implementation, and query development. These activities contribute to software development effort but are not directly represented in conventional UCP measurements. By incorporating database structural complexity through DBCI, the proposed model extends the software size measurement beyond functional requirements, providing a broader representation of software complexity, particularly for information systems with significant database dependencies.

From a practical perspective, the proposed UCPDBCI model can support software developers and project managers during the early stages of software development by assisting in budget planning, resource allocation, project feasibility analysis, and development planning. Considering both functional and database structural complexity allows the estimation process to better reflect the characteristics of data-intensive information systems, thereby providing additional information that may support project planning and decision-making.

Nevertheless, this study has several limitations. The proposed UCPDBCI model has not yet been empirically validated using actual software development projects, nor has it been comparatively evaluated against established estimation methods such as UCP or COCOMO. Therefore, this study does not claim that the proposed model provides higher estimation accuracy than existing approaches. Instead, the primary contribution of this research lies in proposing an estimation framework that integrates functional complexity and database structural complexity into a unified software size measurement. Future research is recommended to validate the proposed model using completed software projects, compare its estimation performance with existing estimation methods, and calibrate the productivity factor using empirical project datasets to further evaluate the effectiveness and applicability of the proposed model.

V. CONCLUSION AND RECOMMENDATION

This study successfully develops a software cost estimation model for the Executive Information System (EIS) Dashboard for Research and Community Service (RCS) monitoring using the proposed UCPDBCi approach. The model integrates functional complexity (Use Case Points), database structural complexity (Database Complexity Index), and productivity-based effort conversion to produce a more comprehensive estimation framework.

The results show that the system has a final UCP value of 55,071 and a DBCi value of 246, based on eight entities, thirty-five attributes, eight primary keys, seven foreign keys, and seven relationships. By combining both metrics, the estimated total software development cost is Rp631.222.722. This indicates that both functional and database complexities significantly influence the overall development effort and cost, making the proposed model more comprehensive compared to function-only estimation approaches.

However, this study has several limitations. The model is evaluated using a single case study, which limits its generalizability to other software systems. In addition, the estimated cost has not been validated through expert judgment or

comparison with actual project costs, and other influencing factors such as team experience, project risk, and technology complexity are not included. Furthermore, the system is only implemented as a prototype, so its real-world effectiveness has not been evaluated.

Future research is recommended to validate the proposed model through expert evaluation and comparison with actual development costs using accuracy metrics such as MAPE or MMRE. The model should also be applied to multiple case studies to improve its generalizability. Additionally, further improvements may include integrating other cost drivers such as risk factors and system maintainability, as well as implementing the system in real organizational environments to evaluate its practical effectiveness in supporting decision-making and RCS monitoring.

REFERENCES

- Ahmed, M., Ibrahim, N. B., Nisar, W., Ahmed, A., Junaid, M., Flores, E. S., & Anand, D. (2024a). A Hybrid Model For Improving Software Cost Estimation In Global Software Development. *Computers, Materials and Continua*, 78(1), 1399–1422. <https://doi.org/10.32604/cmc.2023.046648>
- Ahmed, M., Ibrahim, N. B., Nisar, W., Ahmed, A., Junaid, M., Flores, E. S., & Anand, D. (2024b). A Hybrid Model for Improving Software Cost Estimation in Global Software Development. *Computers, Materials and Continua*, 78(1), 1399–1422. <https://doi.org/10.32604/cmc.2023.046648>

- Ahmed, M., Ibrahim, N., Nisar, W., & Ahmed, A. (2023). Systematic Literature Review On Global Software Development Based Software Cost Estimation Models And Cost Drivers. *Indonesian Journal of Electrical Engineering and Computer Science*, 32(3), 1485–1494. <https://doi.org/10.11591/ijeecs.v32.i3.pp1485-1494>
- Ardiansyah, Indana, M., Tarmuji, A., & Hamid, F. (2024). Optimization Of Use Case Point Through The Use Of Metaheuristic Algorithm In Estimating Software Effort. *International Journal of Advances in Intelligence Informatics*, 10(1), 109–130.
- Azima, M. F., Karnila, S., & Kurniawan, H. (2018). Executive Information System (EIS) Untuk Meningkatkan Mutu Pengendalian Dan Pengawasan Terhadap Belanja Pemeliharaan Kendaraan. *Jurnal Sistem Informasi & Manajemen Basis Data (SIMADA)*, 01(02), 104–115.
- Azzeh, M., Bou, A., & Basem, I. (2021). Predicting Software Effort From Use Case Points: A Systematic Review. *Science of Computer Programming*, 204(102596), 1–26. <https://doi.org/10.1016/j.scico.2020.102596>
- Azzeh, M., Nassif, A. B., & Attili, I. B. (2021). Predicting Software Effort from Use Case Points: A Systematic Review. *Science of Computer Programming*, 204. <https://doi.org/10.1016/j.scico.2020.102596>
- Dey, P., Sinha, B. R., Romney, G., & Amin, M. (2014). Estimation of Database Complexity from Modeling Schemas. *Journal of Computing Sciences in Colleges (JCSC)*, 30.
- Donner, E. K. (2023). Research Data Management Systems And The Organization Of Universities And Research Institutes: A Systematic Literature Review. In *Journal of Librarianship and Information Science* (Vol. 55, Number 2, pp. 261–281). SAGE Publications Ltd. <https://doi.org/10.1177/09610006211070282>
- Draz, M. M., Emam, O., & Azzam, S. M. (2024). A Predictive Model For Software Cost Estimation Using ARIMA Algorithm. *(IJACSA) International Journal of Advanced Computer Science and Applications*, 15(7), 656–665.
- Fajriyah, A., Mudawamah, N. S., & Puspitadewi, G. C. (2024). Information Management Design For Higher Education. *Record and Library Journal*, 10(1), 50–62. <https://doi.org/10.20473/rlj.V10-I1.2024.50-62>
- Fernandes, S. (2018). Looking Deep At Current Research Information Systems: The Information Science Perspective In Higher Education. *Qualitative and Quantitative Methods in Libraries*, 7, 263–273.

- Gandomani, T. J., Dashti, M., & Ansaripour, S. (2025). Enhancing Analogy-Based Software Cost Estimation Using Grey Wolf Optimization Algorithm. *PeerJ Computer Science*, 1–43. <https://doi.org/10.7717/peerj-cs.2794>
- Haleem, M., Islam, M., Ahmed, M. N., Ahmed, M. M., & Khan, I. M. (2025). A Critical Review for Software Maintenance Cost Estimation Models: A Data Driven Approach. *IEEE Access*, 13(October), 184825–184849. <https://doi.org/10.1109/ACCESS.2025.3618759>
- Juhola, M. (2022). A Method To Measure Data Complexity Of A Complicated Medical Data Set. *International Journal of Imaging Systems and Technology*, 32(6), 1822–1831. <https://doi.org/10.1002/ima.22760>
- Karner, G. (1993). *Resource Estimation For Objectory Projects*.
- McCoy, C., & Rosenbaum, H. (2019). Uncovering Unintended And Shadow Practices Of Users Of Decision Support System Dashboards In Higher Education Institutions. *Journal of the Association for Information Science and Technology*, 70(4), 370–384. <https://doi.org/10.1002/asi.24131>
- Prayitno, E., Siregar, J., Dewi, Y. N., Bachri, C., Indriyani, L., & Ma'arif, S. (2020). Use Case Points (UCP) With 3 Point In Program Evaluation And Review Technique (PERT) To Estimate Effort Software. *Journal of Physics: Conference Series*, (1641), 1–6. <https://doi.org/10.1088/1742-6596/1641/1/012008>
- Rashid, C. H., Shafi, I., Ahmad, J., Thompson, E. B., Vergara, M. M., De La Torre Diez, I., & Ashraf, I. (2023). Software Cost and Effort Estimation: Current Approaches and Future Trends. *IEEE Access*, 11, 99268–99288. <https://doi.org/10.1109/ACCESS.2023.3312716>
- Sharma, H. (2017). Open Source Dashboard Technology And IEDSS For Higher Education Institutions. *International Journal Of Engineering And Computer Science*. <https://doi.org/10.18535/ijecs/v6i6.01>
- Silhavy, R., Bures, M., Alipio, M., & Silhavy, P. (2023). More Accurate Cost Estimation For Internet Of Things Projects By Adaptation Of Use Case Points Methodology. *IEEE Internet of Things Journal*, 10(21), 19312–19327. <https://doi.org/10.1109/JIOT.2023.3281614>
- Sukorini, R. S., Marini, A., & Aulia, R. N. (2024). New Era In Higher Education : Digital Trandformation And Information System Management. *Improvement: Jurnal Ilmiah Untuk Peningkatan Mutu Manajemen Pendidikan*, 11(2), 154–166. <https://doi.org/10.21009/improvement.v11i2.49441>
- Triono, W., Lesmi, K., & Khotimah, I. (2023). Optimization and Challenges of Implementing Academic Service Management Information Systems in Muhammadiyah Higher Education: A Multiple-Case Study. *Jurnal Pendidikan Islam*.

- Ulfatriyani, H., Prasetyo, E., & Kurnianti, A. (2020). Visualization Of Alumni Performance Using Dashboard To Support Higher Education Decision Making. *Emerging Information Science and Technology*, 1(3), 84–92. <https://doi.org/10.18196/eist.v1i3.13174>
- Vanathi, D., Anusha, K., Ahilan, A., & Eveline, A. S. (2024). Software Cost And Effort Estimation Using Dragonfly Whale Optimized Multilayer Perceptron Neural Network. *Alexandria Engineering Journal*, 103(July 2023), 30–37. <https://doi.org/10.1016/j.aej.2024.04.043>
- Velcu-Laitinen, O., & Yigitbasioglu, O. M. (2012). The Use Of Dashboards In Performance Management: Evidence From Sales Managers. *International Journal of Digital Accounting Research*, 12, 39–58. https://doi.org/10.4192/1577-8517-v12_2
- Yoke Wah, T., Suiying, T., & Leow Shuangjie, A. (2019). Dashboards For Decision Making In Higher Education Background Information. *ASCILITE 2019 Singapore University of Social Sciences*. https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining
- Zhang, W., Cheng, H., Zhan, S., & Luo, M. (2024). Dimensionality Reduction And Machine Learning Based Model Of Software Cost Estimation. *Frontiers in Physics*, 12(March), 1–9. <https://doi.org/10.3389/fphy.2024.1324719>
- Zhao, X., Xiong, X., Mansor, Z., Razali, R., Zakree, M., Nazri, A., & Li, L. (2025). A Data-Driven Cost Estimation Model For Agile Development Based On Kolmogorov-Arnold Networks And Adamw Optimization. *Journal of King Saud University - Computer and Information Sciences*, 37(85), 1–21. <https://doi.org/10.1007/s44443-025-00058-7>