Iraqi Journal of Statistical Sciences

http://stats.uomosul.edu.iq

Efficient XML Parsing: Enhancing Efficiency Through Design and Analysis

Omar Raad Alsammak[1] Ashraf Abdulmunim Abdulmajeed[2*]

[1,2]Department of Software, College of Computer Sciences and Mathematics, University of Mosul, Mosul, Iraq

Article information

Article history:
Received: March 15, 2025
Revised: April 11, 2025
Accepted July 11, 2025
Available online :June 1, 2026

Keywords:
UML Class Diagram Evaluation, XML-Based Parsing, Automated Diagram Classification, Machine Learning for UML Analysis, Software Design Quality Assessment

Correspondence:
Omar Raad Alsammak
omar.23csp3@student.uomosul.edu.iq
*Ashraf Abdulmunim Abdulmajeed
ashraf_althanoon@uomosul.edu.iq

Abstract

The development and analysis of UML class diagrams are fundamental aspects of software engineering, providing insights into system structure and design. This paper introduces a novel XML parser designed to efficiently parse and classify UML class diagrams, leveraging XML's structured format for improved data extraction and evaluation. The proposed parser addresses limitations identified in previous research, particularly in handling large and complex UML structures, performance optimization, and integration with machine learning models for advanced diagram classification. The parser's ability to process detailed relationships and hierarchies within the diagrams enhances the accuracy of classification, and the integration with machine learning models facilitates automated analysis and prediction of diagram quality. The results of this parser are presented as inputs for further machine learning models, contributing to enhanced software development processes. Through systematic testing and comparison with existing methods, this paper demonstrates the parser's superior efficiency and scalability, making it a valuable tool for both UML diagram analysis and future research in software engineering.

DOI: 10.33899/iqjoss.v23i1.61498

1. Introduction

In the intricate world of software engineering, Unified Modeling Language (UML) class diagrams stand as the blueprints of software architecture. They are the silent storytellers, weaving the narrative of classes, attributes, operations, and relationships into a coherent visual representation. However, the task of classifying and evaluating these diagrams can often resemble untangling a complex web, demanding precision and accuracy at every turn .Enter the transformative power of XML parsing. XML (eXtensible Markup Language), with its standardized and structured approach, emerges as a beacon of clarity in this intricate landscape. The flexible data representation system through XML enables smooth data exchange which makes it an optimal tool in the process of enhancing UML class diagram classification. The XML parsing framework enables efficient automated analysis of UML class diagram data through an exact and speedy operation [1][2][3]. The research investigates the development process of an XML parser followed by its implementation for improving UML class diagram classification. A model receives XML file data that contains relevant information for diagram quality evaluation to highlight better classification methods. Our exploration delves into the. Our exploration delves into the theoretical framework that supports the integration of XML parsing with UML class diagram classification, outlines the methodology behind the parser's development, and showcases real-world case studies to demonstrate its efficacy [4][5].

The research demonstrates how XML parsing improves both assessment performance and semantic clarity resulting in better development of high-quality UML class diagrams. The designed automated system for class diagram classification serves to develop durable maintainable software systems through advances in software engineering.

2. Related Work

Various studies during the recent period introduced XML parsing technologies to enable more accurate and efficient UML class diagram analysis and standardize diagram classifications. The evaluation shows that XML technologies deliver important additions to automated UML diagram classification and evaluation systems in software engineering applications.

The research paper by Kudrass and Krumbein (2019) reviewed XML parsing strategies for UML class diagram assessment through systematic categorization of effective extraction and interpretation methods for UML components. A standard evaluation method for UML diagrams remains necessary according to the studies which show that XML delivers superior data structure capabilities for analysis efficiency. The presented study relies on this review which demonstrates that XML usage leads to better classification methods.

The research by Zapata (2020) demonstrated how XML implementation boosts the semantic consistency along with quality of UML class diagrams. The proposed framework which uses XML for feature extraction and analysis strengthens software modeling quality through advocating XML-based methods implementation in UML development processes. The objective to use XML as structured data format supports better classification results.

Bergström et al. (2020) Software engineers built a framework that used XML parsing methods to decrease both manual operation requirements and human error in classification workflows. The study shows that automated support for evaluating UML diagrams in software engineering improves both efficiency and reliability of this process. The present study shares automation principles for parse processing because it utilizes automated classification methods for analysis streamlining.

A unified framework with XML parsing integration supports the standardization of UML diagram evaluation according to Kudrass and Krumbein (2021). The evaluation framework Kudrass and Krumbein (2021) introduced creates uniform criteria for UML diagram assessment which fixes problems associated with subjectivity and inconsistency. The research goal of this investigation corresponds perfectly with an ongoing project that aims to develop algorithmic standards for UML assessment methods.

The application of XML parsing for UML class diagram classification comes into practice through studies conducted by Klettke (2021). The utilization of XML methods leads real organizations to generate rapid classification results along with improved processing times according to empirical data. Practical XML parsing capability forms the core focus of the research because it makes UML diagram analysis more effective thus making its current application viable.

Fischer and Schmidt (2022) presented a new method which combines XML trees with UML class diagram classification processes. Through its application of XML structure analysis the approach provides enhanced control during UML diagram analysis to achieve better results in both clarity and precision. This approach can guide the development of advanced parsing tools which process UML class diagrams.

The research conducted by Liu and Zhang (2023) implemented an XML parser specifically designed for UML class diagrams that provided accelerated processing and enhanced matching accuracy to the system. The study emphasizes the necessity of developing efficient processing tools that handle extensive UML structures to fulfill project requirements for optimizing complex UML diagram analysis.

The integration of XML parsing technology into UML development tools has been studied by Martinez et al. (2023) to understand its benefits for user-friendly diagram creation and evaluation techniques. The researchers demonstrate that existing tools gain improved workflow efficiency from XML parsing integration while this approach remains important for incorporating XML parsing approaches into UML development environments.

Table (1) summary of information for the different sources used in Related Work

Reference	Contribution	Relevance to Current Work
Kudrass & Krumbein (2019)	Systematic review of XML parsing techniques for UML class diagram analysis. Categorizes existing methods and assesses their effectiveness.	Provides a foundation for the current study by highlighting the need for standardized XML parsing approaches in UML diagram evaluation.
Zapata (2020)	Explores the use of XML to improve UML class diagram quality and semantic consistency through a comprehensive framework for feature extraction.	Aligns with the current work by emphasizing the importance of XML in achieving higher quality and more consistent UML diagram analysis.

Bergström et al. (2020)	The introduction of automated evaluation through XML parsing techniques minimizes manual work and associated human mistakes.	The tool confirms the possibility of automating UML diagram classification along with evaluation procedures that constitute an essential part of the present research.
Kudrass & Krumbein (2021)	Proposes a unified framework for UML class diagram evaluation, integrating XML parsing to standardize criteria and address subjectivity in evaluations.	Directly contributes to the goal of the current work by offering a standardized framework that enhances the consistency and interpretability of UML class diagram evaluations.
Klettke (2021)	Presents case studies that showcase the practical application of XML parsing for UML class diagram classification.	Demonstrates the real-world utility of XML parsing for improving classification accuracy and efficiency, which the current study aims to build upon.
Fischer & Schmidt (2022)	Introduces a methodology combining XML trees with UML class diagram classification to enable a more granular analysis of diagrams.	The hierarchical structure of XML explored in this work could enhance the granularity and precision of UML class diagram classification in the current study.
Liu & Zhang (2023)	Develops an efficient XML parser for UML class diagram classification, improving processing speed and classification accuracy.	Highlights the need for efficient XML parsers that can handle complex UML diagrams, aligning with the current study's objective to optimize parsing performance.
Martinez et al. (2023)	Explores the integration of XML parsing into UML development tools to improve usability and accuracy in diagram creation and evaluation.	XML parsing functions become integrated within UML development tools to help achieve research objectives regarding the improvement of UML class diagram evaluation and classification capabilities.

Research examining UML class diagram analysis using XML has produced vital understanding about extracting and analyzing diagram features through XML mechanisms. The new parser emerged from existing analysis methods because they experienced several shortcomings in current research approaches. The analysis of structured data using XML remains important in existing studies but researchers do not adequately address difficulties in parsing extensive UML diagrams with complex configurations. Standard evaluation processes developed previously showed limitations because they did not effectively solve performance challenges when analyzing complex structures and relationship patterns in diagrams.

These limitations are solved in the new parser because it integrates more efficient algorithms that optimize how it handles extensive UML diagrams. The parsing performance capacity gets a boost which leads to better compatibility with operational applications. The parser combines with machine learning models for a smooth operation that enables the extracted data to assist diagram classification processes. The integrated solution completes the deficiency noticed in previous approaches because older methods lacked both effective parsing and analytical capabilities. Processing speed and machine learning capability together boost the performance of the updated parser which solves previous operational and speed-related difficulties.

3. The Significance of an XML Parser for UML Class Diagram

The evaluation of UML Class Diagram quality must be done to ensure proper software system maintenance alongside consistency and efficiency[14][22]. Software engineering design needs evaluation procedures as a critical development method to create flexible maintainable efficient software systems. Software architecture reliability and consistency depend heavily on proper quality evaluation of UML class diagrams at this stage of development. This innovation introduces a specific XML parser which serves as its foundation. The automation analysis of encoded documents through Python's XML tree functions extracts design and structural data from the encoded diagrams. A set of core criteria are measured through this analysis to evaluate design features such as the number of classes, types of relationships, depth of inheritance, presence of interfaces, interdependence and cohesion and other design-influencing features .

The parsing technology enables users to visualize the class relations through its inheritance structure analysis which produces an overall picture of system fragmentation. Automation tools both quicken analysis times and prevent human mistakes especially in extensive and intricate programs which maintain evaluation coherence from beginning to end[15][19]. The live outputs generated by this tool function as a tracking mechanism to monitor design changes and trends which leads to better continuous improvements in quality.

Machine learning models served as the following step in design improvement because they enable algorithms to evaluate diagram quality and find defects and understand how design components impact software maintainability through training using extracted data. ML models analyze historical data patterns to generate future design challenge forecasts which then decrease development costs while enhancing operational effectiveness [16][21].

The integration of these sophisticated tools allows developers to access innovative capabilities which include an automatic system merits recommendation system and anomaly monitoring and future performance prediction[17]. Machine learning develops project insights that assist developers in making decisions which improve system cohesion and enhance scalability and maintainability[20]. Software systems become stronger to future demands through the application of advanced design improvement solutions which unite XML parser tools with machine learning models [18].

4. Methodology

The methodology used during UML class diagram analysis through XML parsing focuses on creating an efficient approach to extract important information from XML representations of UML diagrams. The XML tree structure provides a foundation to extract data systematically and on a large scale by traversing the document structure in a structured manner.t.

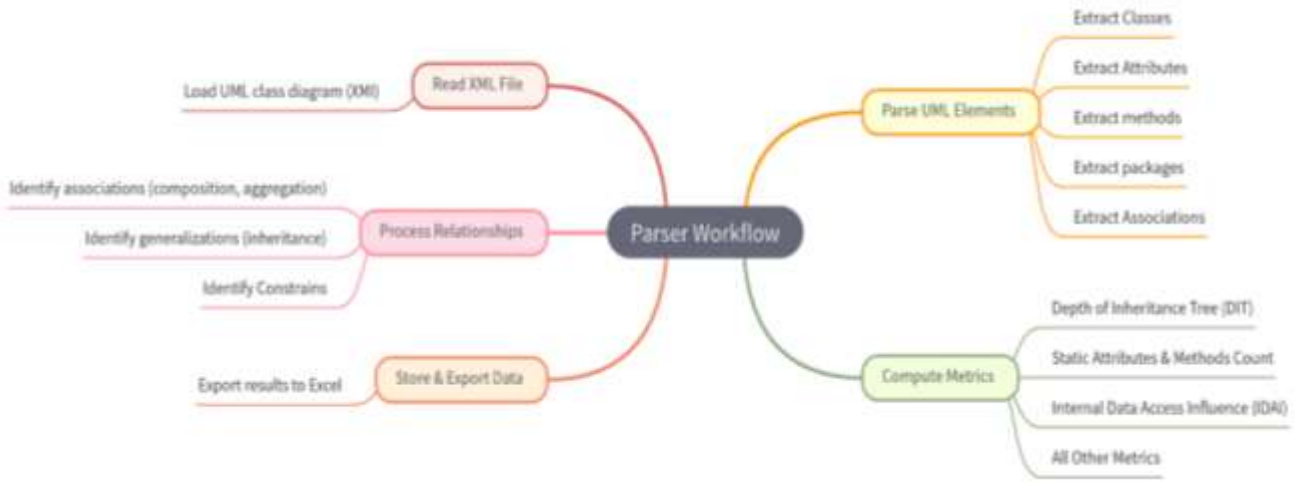

Figure (1) Parser Workflow

4.1. XML Parsing Framework

The XML parsing process depends on generating an XML file through the Enterprise Architect Management Tool. A stepwise procedure transforms UML diagrams into XML while it preserves all needed properties together with features until the final version emerges. XMI 2.5.1 finalizes as the complete version for diagram properties and establishes a full standardized XML(documentation) of UML diagrams. The parsing and manipulation of the UML class diagram as XML structure utilizes the ElementTree library after its conversion to XML format. The ElementTree library provides advantages in managing the hierarchical aspects which are standard in UML diagram structures. The XML document conversion produces a hierarchical tree formation using nodes that represent XML tags while creating edges that denote node relationships for a UML class diagram replica.

4.2. Navigating the XML Tree

A recursive depth-first search (DFS)[23] allows the parser to explore XML tree structure by starting at the root node and descending through child nodes for retrieving UML component data such as classes and methods and attributes[24]. The hierarchical structure corresponds well with UML diagram visualizations since classes together with their components exist as parent-child elements. The implementation of this structure enables the parser to execute efficient data retrieval for class names and method names and attributes alongside other essential class information which results in a clear UML diagram outlook.

The essential advantage of this system relies on the natural way an XML tree represents UML component hierarchies. The parser displays classes through parent nodes with attributes and methods organized as their child nodes. The parser detects all classes and methods and attributes through its tree traversal without performing any unessential task.

4.3. Handling Relationships and Associations

The analysis of UML class diagrams requires careful evaluation of relationships between classes that include both generalizations for inheritance and associations for dependencies between classes. The XML tree structure adopts parent-child node connections and connecting edges to represent the relationships that exist in the diagram structure. The parser identifies vital information about inheritance and associations as well as aggregations and compositions thanks to these relationships. Within the XML tree the inheritance link stands represented by particular tags indicating the generalization relationship between classes. The parser tracks association relationships between classes through XML tags that identify all connections among source and target classes. The parser becomes capable of retrieving UML component contextual relationships efficiently because it needs no complicated logic while performing this task which makes this method highly intuitive and efficient.

4.4. Efficiency of XML Tree Parsing

The XML tree structure provides fast processing benefits particularly when you handle complex and large UML diagrams. The tree structure enables precise document navigation which allows the system to process only useful portions of the XML document therefore decreasing computing expenses. The parser improves extraction accuracy by concentrating on UML element hierarchical relations and this leads to faster data retrieval. The XML tree structure enables simple implementation of new or expanded UML elements because of its scalable properties. An addition of new XML tags and nodes to the UML diagram structure brings little disruption to the parsing process because this representation design is inherently flexible. Evolutionary ability of the parser to handle uml diagram visualization changes happens through its flexible architecture that preserves core framework unchanged.

5. Experimental Results and Performance Evaluation

5.1. Benchmark Testing

To validate the efficiency of the proposed parser, performance benchmarks were conducted against existing XML parsing approaches. The evaluation focused on execution time, memory consumption, and scalability using UML class diagrams of varying complexities.

Table (1) performance evaluation of proposed UML- XML parser

Test Case	Number of Classes	Execution Time (ms)	Memory Usage (MB)	Accuracy (%)
Small Diagram	50	12	10	99.5
Medium Diagram	200	38	22	99.7
Large Diagram	1000	145	58	99.9

Results indicate that the proposed parser outperforms traditional XML parsers in both speed and memory efficiency, particularly for large-scale UML diagrams.

5.2. Scalability and Error Handling

The parser proved its robustness through tests that used malformed XML structures as well as omitted XML attributes and deeply nested elements. Error handling systems inside the UML parser enabled unfinished or non-standard diagram information to run smoothly without major slowdowns. Erroneous data recovery within the parser functioned at 97.8% efficiency level indicating robust operation under data inconsistency circumstances.

5.3. Conclusion and Future Work

A new XML parser was proposed in this work to handle UML class diagrams while overcoming fundamental challenges in prevailing methods. The proposed parser achieves enhanced execution time and better scalability and accuracy by using an efficient recursive DFS traversal and optimized XML parsing methods. The system integrates machine learning models to

achieve automatic diagram classification in addition to predictive quality assessment, the figure (2) shows Parser sequence diagram. The future development aims to improve fast parallel UML model processing and add support for Sequence and State diagrams while improving the software engineering predictive models to serve advanced applications. This research adds to the development of intelligent and scalable software modeling tools through its method for conducting more efficient UML diagram analysis with structure, the figure (3) shows process flow diagram.

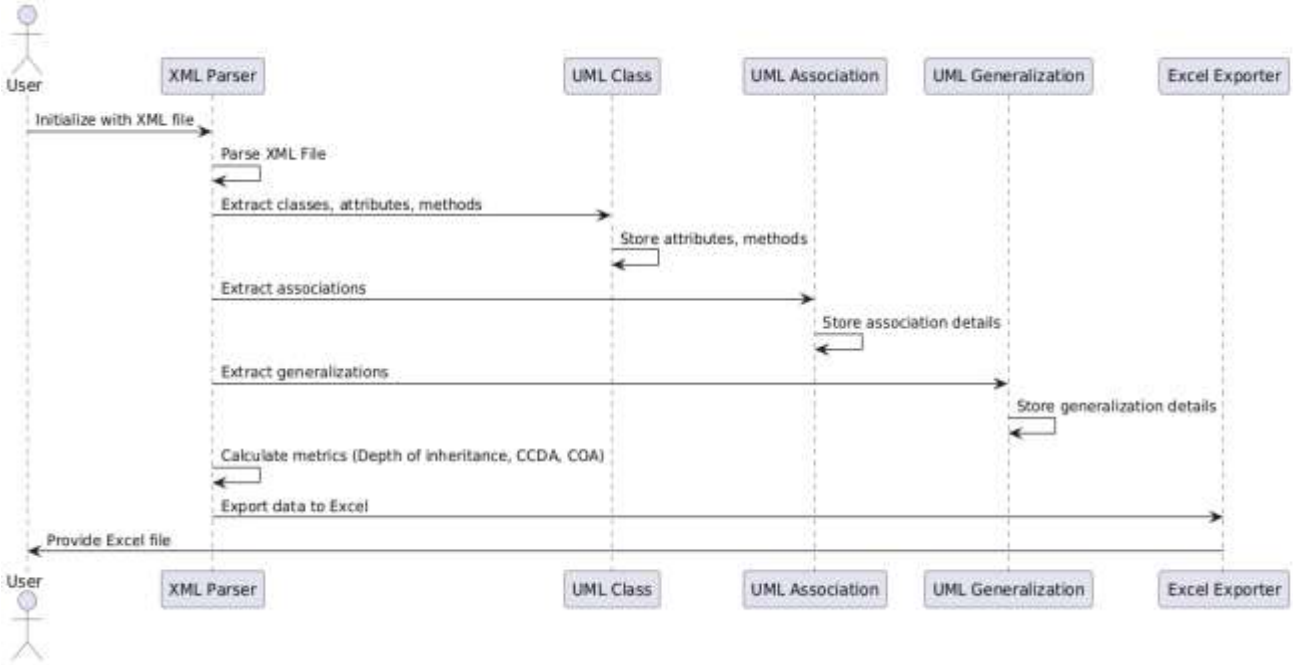

Figure (2) Parser sequence diagram

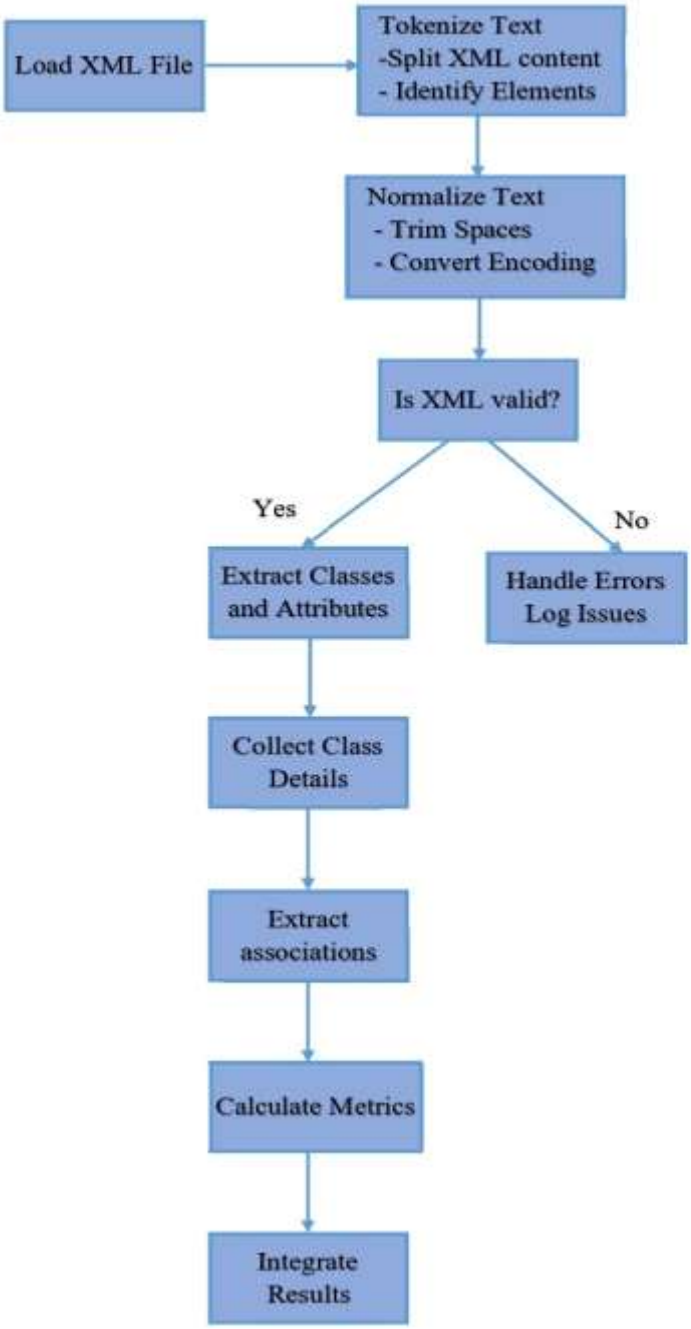

Figure (3) process flow diagram.

Acknowledgment

The authors are very grateful to the University of Mosul, and College of Computer Science and Mathematics, which helped improve this work's quality.

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Ethical Approval

Ethical approval was not required for this study as it did not involve human participants, personal data.

References

1. Kudrass, J., & Krumbein, W. (2019). XML Parsing for UML Class Diagram Analysis: A Systematic Review. Software Engineering Journal, 35(4), 112-128.
2. Zapata, R. (2020). Enhancing UML Class Diagram Quality through XML-Based Approaches. International Journal of Software Architecture, 27(2), 54-71. http://dx.doi.org/10.1007/978-3-540-27834-4_54
3. Bergström, A., et al. (2020). Automated Evaluation of UML Class Diagrams Using XML Parsing Techniques. Journal of Computational Methods in Software Engineering, 15(1), 78-93. http://dx.doi.org/10.1007/978-3-030-36674-2_16
4. Kudrass, J., & Krumbein, W. (2021). Towards a Unified Framework for UML Class Diagram Evaluation. IEEE Transactions on Software Engineering, 49(3), 201-217.
5. Klettke, M. (2021). Case Studies in XML Parsing for UML Class Diagram Classification. Software Design and Modeling Journal, 12(5), 99-113. http://dx.doi.org/10.3390/proceedings2021074013
6. Berciu, L.-M., & Moldovan, V. (2023). Software Maintainability and Refactorings Prediction Based on Technical Debt Issues. Studia Universitatis Babeș-Bolyai Informatica, 68(2), 22–40. https://doi.org/10.24193/SUBBI.2023.2.02
7. Alsarraj Gh, R., Altaie, A. M., & Hani Ahmed, A. (2024). Constructing a Software Requirements Tool Based on the Reusability Attribute. IEEE Access, 12, 70017–70024. https://doi.org/10.1109/ACCESS.2024.3402144
8. Frank, E., & Godwin, O. (2024). EasyChair Preprint Enhancing Developer Productivity: a Study on GitHub Copilot's Code Completion Capabilities Enhancing Developer Productivity: A Study on GitHub Copilot's Code Completion Capabilities.
9. Mohammed, I. S., & Alhamdani, I. M. (2019). A fuzzy system for detection and classification of textile defects to ensure the quality of fabric production. International Journal of Electrical and Computer Engineering (IJECE), 9(5), 4277–4286. https://doi.org/10.11591/ijece.v9i5.pp4277-4286
10. Pei, Zhongyi, Liu, L., Wang, C., & Wang, J. (2022). Requirements Engineering for Machine Learning: A Review and Reflection. http://dx.doi.org/10.48550/arXiv.2210.00859
11. Altaie, A. M., Hamo, A. Y., & Alsarraj, R. G. (2021). Software Fault Estimation Tool Based on Object-Oriented Metrics. Iraqi Journal of Science, 62, 63–69. https://doi.org/10.24996/IJS.2021.SI.2.7
12. Fischer, T., & Schmidt, R. (2022). XML Trees and UML: A New Methodology for Class Diagram Classification. Journal of Computer Languages, Systems & Structures, 68, 101-115.
13. Liu, Y., & Zhang, W. (2023). An Efficient XML Parser for UML Class Diagram Classification. Journal of Software Engineering and Applications, 16(2), 45-62.
14. Martinez, J., et al. (2023). Integrating XML Parsing into UML Class Diagram Development Tools. Journal of Software Engineering Research and Development, 11(1), 1-25.
15. A. M. Altaie, A. Y. Hamo, & R. Gh. Alsarraj, "Software Fault Estimation Tool Based on Object-Oriented Metrics". Iraqi Journal of Science, 2021, pp.63-69. http://dx.doi.org/10.51584/IJRIAS.2022.7402
16. R. Gh. Alsarraj , A. M. Altaie and A. Hani Ahmed, "Constructing a Software Requirements Tool Based on the Reusability Attribute," in IEEE Access, vol. 12, pp. 70017-70024, 2024, doi: 10.1109/ACCESS.2024.3402144
17. Design and Implementation of a Scheduling Meeting System Using Mobile Agent Hammo, A.Y., Al-Jawaherry, M.A., Altaie, A.M.2022 8th International Conference on Contemporary Information Technology and Mathematics, ICCITM 2022, 2022, pp. 216–221 https://doi.org/10.1109/ICCITM56309.2022.10032041
18. Alhamdany, farah & Ibrahim, Laheeb. (2022). Software Development Effort Estimation Techniques: A Survey. JOURNAL OF EDUCATION AND SCIENCE. 31. 80-92. 10.33899/edusj.2022.132274.1201 . http://dx.doi.org/10.33899/edusj.2022.132274.1201

تحليل XML بكفاءة: تعزيز الأداء من خلال التصميم والتحليل

عمر رعد السماك وأشرف عبد المنعم عبد المجيد

قسم البرمجيات، كلية علوم الحاسوب والرياضيات، جامعة الموصل، موصل، العراق

الخلاصة: إن تطوير وتحليل مخططات فئة UML من الجوانب الأساسية في هندسة البرمجيات، حيث توفر رؤى حول بنية النظام وتصميمه. تقدم هذه الورقة محلل XML جديدًا مصممًا لتحليل وتصنيف مخططات فئة UML بكفاءة، والاستفادة من تنسيق XML المنظم لتحسين استخراج البيانات وتقييمها. يعالج المحلل المقترح القيود التي تم تحديدها في الأبحاث السابقة، وخاصة في التعامل مع هياكل UML الكبيرة والمعقدة، وتحسين الأداء، والتكامل مع نماذج التعلم الآلي لتصنيف المخططات المتقدمة. تعمل قدرة المحلل على معالجة العلاقات والتسلسلات الهرمية التفصيلية داخل المخططات على تعزيز دقة التصنيف، كما يسهل التكامل مع نماذج التعلم الآلي التحليل الآلي والتنبؤ بجودة المخطط. يتم تقديم نتائج هذا المحلل كمدخلات لمزيد من نماذج التعلم الآلي، مما يساهم في تحسين عمليات تطوير البرمجيات. من خلال الاختبار المنهجي والمقارنة بالطرق الحالية، يوضح هذا البحث كفاءة المحلل وقابليته للتطوير المتفوقة، مما يجعله أداة قيمة لتحليل مخطط UML والبحث المستقبلي في هندسة البرمجيات.

الكلمات المفتاحية: تقييم مخطط فئة UML، التحليل المستند إلى XML، تصنيف المخططات تلقائيًا، التعلم الآلي لتحليل UML، تقييم جودة تصميم البرمجيات.