Administrative Service System for the Community in Tanjung Kerang Village

Adi Zamrozi, A Sanmarino
University of Indo Global Mandiri, Palembang, Indonesia

Abstract

In this study, the challenges faced by Tanjung Kerang Village in providing optimal community administration services using conventional methods are highlighted. The village, despite having a relatively large population, faces issues such as long queues, delays due to forgotten requirements, and difficulties in accessing data for administrative processes. To address these challenges, the study proposes the implementation of a community service administration information system using the Prototype Model in software development. The Prototype Model, characterized by quick iterations, user involvement, and continuous refinement, is chosen as the methodology for system development. This approach allows for the creation of a preliminary version of the software that undergoes testing and refinement in multiple cycles. The iterative nature of the model, coupled with user feedback, facilitates the gradual convergence towards a final, fully functional system. In conclusion, the Prototype Model proves instrumental in addressing the identified challenges, offering a systematic and iterative approach to enhance community administration services. The proposed information system has the potential to significantly improve service delivery, making it more effective and efficient for both administrators and the community.

Keywords: *Community administration services, Software development, Service delivery improvement*

Abstrak

Dalam studi ini diangkat tantangan yang dihadapi Desa Tanjung Kerang dalam memberikan pelayanan administrasi kemasyarakatan yang optimal dengan menggunakan metode konvensional. Meskipun memiliki jumlah penduduk yang relatif besar, desa ini menghadapi kendala seperti antrean panjang, keterlambatan karena lupa persyaratan, dan kesulitan mengakses data untuk proses administrasi. Untuk mengatasi tantangan tersebut, penelitian mengusulkan implementasi sistem informasi administrasi pelayanan masyarakat dengan menggunakan Model Prototype dalam pengembangan perangkat lunak. Model Prototipe, yang ditandai dengan iterasi cepat, keterlibatan pengguna, dan penyempurnaan berkelanjutan, dipilih sebagai metodologi untuk pengembangan sistem. Pendekatan ini memungkinkan pembuatan versi awal perangkat lunak yang menjalani pengujian dan penyempurnaan dalam beberapa siklus. Sifat model yang berulang, ditambah dengan umpan balik pengguna, memfasilitasi konvergensi bertahap menuju sistem final yang berfungsi penuh. Sistem yang diusulkan terbukti berperan penting dalam mengatasi tantangan yang teridentifikasi, menawarkan pendekatan sistematis dan berulang untuk meningkatkan layanan administrasi masyarakat.

Kata kunci: Layanan administrasi masyarakat, Pengembangan perangkat lunak, Peningkatan penyampaian layanan

1. INTRODUCTION

Service has many forms in various areas of life. Services can be categorized into three forms, namely: verbal services, written services, and deeds. Verbal services are carried out by officers in the field of public relations, information services, and other fields whose job is to provide explanations or information to anyone who needs it. The services provided by community relations officers at the village level government are administrative. Administration is a collaborative activity carried out by a group of people based on the division of labor as determined in the structure by utilizing resources to achieve goals effectively and efficiently [1], [2]. The use of a community service information system has been implemented in several villages in Musi Banyuasin Regency, such as Babat Village, Teggulang Baru Village, and Gajah Mati Village. However, several villages do not have an information system, one of which is Tanjung Kerang Village. Tanjung Kerang Village is a village located in Babat Supat District, Musi Banyuasin Regency in South Sumatra. This village, which is divided into ten hamlets, is located 32 M above sea level and is 8 km from the sub-district capital. The population in Tanjung Kerang Village consists of 1,234 families with a population of 4,696 people.

With a relatively large population, the government in Tanjung Kerang Village takes care of all Community Administration Services such as preparing marriage cover letters, SKCK cover letters, birth certificates, domicile certificates, general certificates, incapacity certificates, and death certificates. The service still uses methods that are classified as conventional even though the administrative service process uses computer equipment, but for data storage it still uses ledgers. Ledger books are used to write data on residents' administrative letters, so the services provided are not optimal. People who come every day in large numbers have to queue for quite a long time. People also sometimes forget to bring the requirements so they have to take them back quite far. Another limitation is that administrative employees on duty often have difficulty finding the data needed by the community. The best service has become an obligation for the government in every village, but using conventional methods, the community has not received optimal service. To improve the quality of service, a community service administration information system is needed in Tanjung Kerang Village so that service and data processing are more effective and efficient.

2. METHOD

The system development phase utilized in this study is the prototype model [3], [4], [5]. The prototype model is an iterative development approach where a prototype (a preliminary version of the software) is developed, tested, and refined in multiple cycles until it evolves into a final, fully-functional system. The purpose of creating a prototype is to provide a tangible representation of the software, allowing stakeholders to visualize and interact with it early in the development process. This helps in gathering feedback, refining requirements, and identifying potential issues before proceeding to the full-scale development. Key characteristics of the prototype model include:

1. Quick Iterations: The development process involves rapid iterations of building, testing, and refining the prototype.
2. User Involvement: Users and stakeholders play a crucial role in providing feedback on the prototype, ensuring that the final product aligns with their expectations.
3. Refinement: The feedback obtained from users is used to refine and improve the prototype in subsequent iterations, gradually converging towards the final system.
4. Flexible Requirements: The model is particularly useful when the requirements are not well-understood or are subject to frequent changes.
5. Risk Identification: Early identification of potential risks and issues through the prototype helps in minimizing risks during the later stages of development.

The process comprises a sequence of phases that must be finished in order, where the conclusion of each phase is reliant on the prior one. The Prototype Model in software development involves a series of iterative steps aimed at refining and improving a preliminary version of the software before full-scale development. The process begins with defining the initial requirements, followed by designing and building a prototype. Once the prototype is developed, it undergoes testing, and user feedback is actively sought. Based on the feedback received, the prototype is refined, and this cycle of building, testing, and refining is repeated until the prototype aligns with user expectations and project requirements. Throughout these iterations, the model emphasizes quick feedback loops and close collaboration with users to ensure that the evolving prototype meets their needs. Once the prototype is deemed satisfactory, the final system is developed using the lessons learned from the prototyping phase. The Prototype Model is flexible, allowing for adjustments in requirements and design as the project progresses, ultimately leading to a more robust and user-friendly final product.

3. RESULTS AND DISCUSSION

The authors commenced the design of the proposed system through the generation of a UML (Unified Modeling Language) diagram [6], [7]. The proposed system design includes Use case diagrams, Activity diagrams, Sequence diagrams, and Class diagrams.

3.1. Use Case diagram

Several things need to be described, namely actors and use cases. Actors are users who are connected to the system and can be people (indicated by their role and not their name/personnel). The actor is symbolized by the figure of a stick man with a noun at the bottom that states the role/system. Use cases are depicted with an ellipse symbol with the name of the active verb inside which states the activity from the actor's perspective [8], [9]. Figure 1 shows proposed a use-case diagram of community service administration.
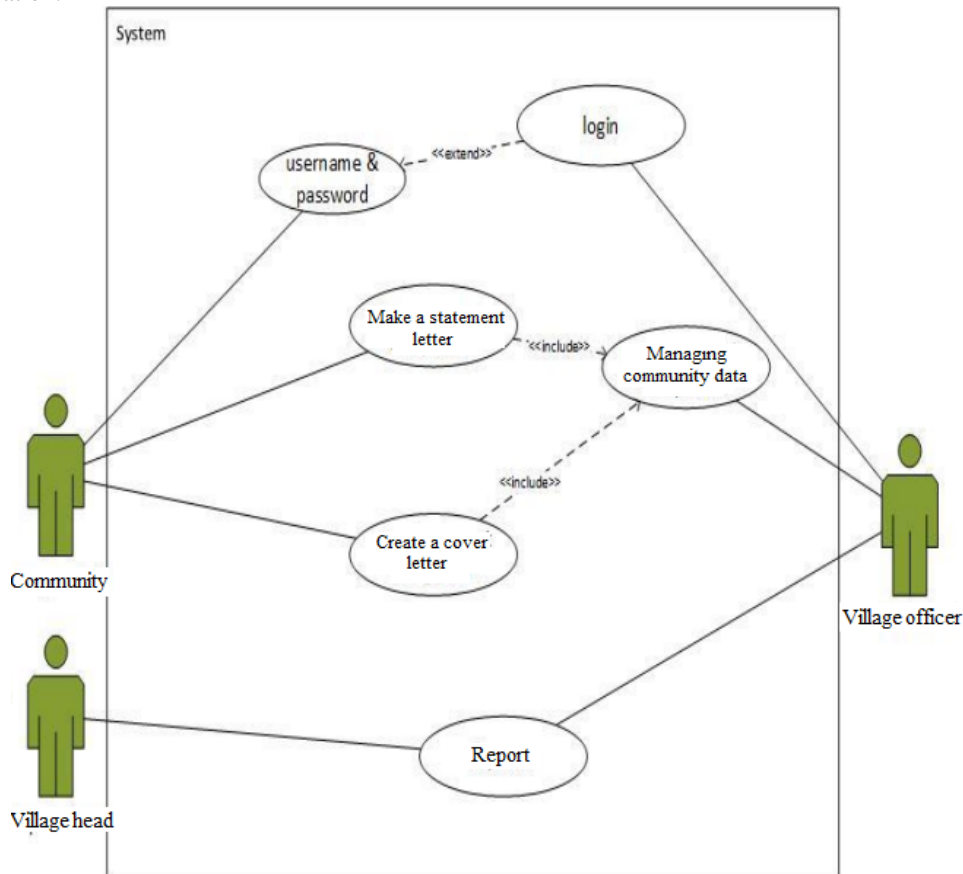
Figure 1 - Use-case diagram

3.2. Activity diagram

An activity diagram is a description of function paths in an information system [10]. In full, the activity diagram defines where the system process starts, where it stops, what activities occur during the system process, and what sequence these activities occur in.

3.3. Class diagram

Class diagrams describe the types of objects in the system and the various static relationships that exist between them [11]. Class diagrams show the properties and operations of a class and the boundaries contained in the object relationships. Figure 2 shows the class diagram of the proposed system.

masyarakat

+ No.k; int; 20
+ nama; varchar; 20
+jenis_kelamin ; varchar;20
+tempat_lahir ; varchar; 20
+tanggal_lahir ; date
+agama ; varchar;10
+status_perkawinan; varchar;20
+pekerjaan;varchar;20
+kewarganegaraan; varchar; 20
+nama_ayah; varchar; 20
+nama_ibu varchar;20

login()
submit()
edit()
keluar()

surat keterangan

+kategori surat

pilih()
kembali()

surat pengantar

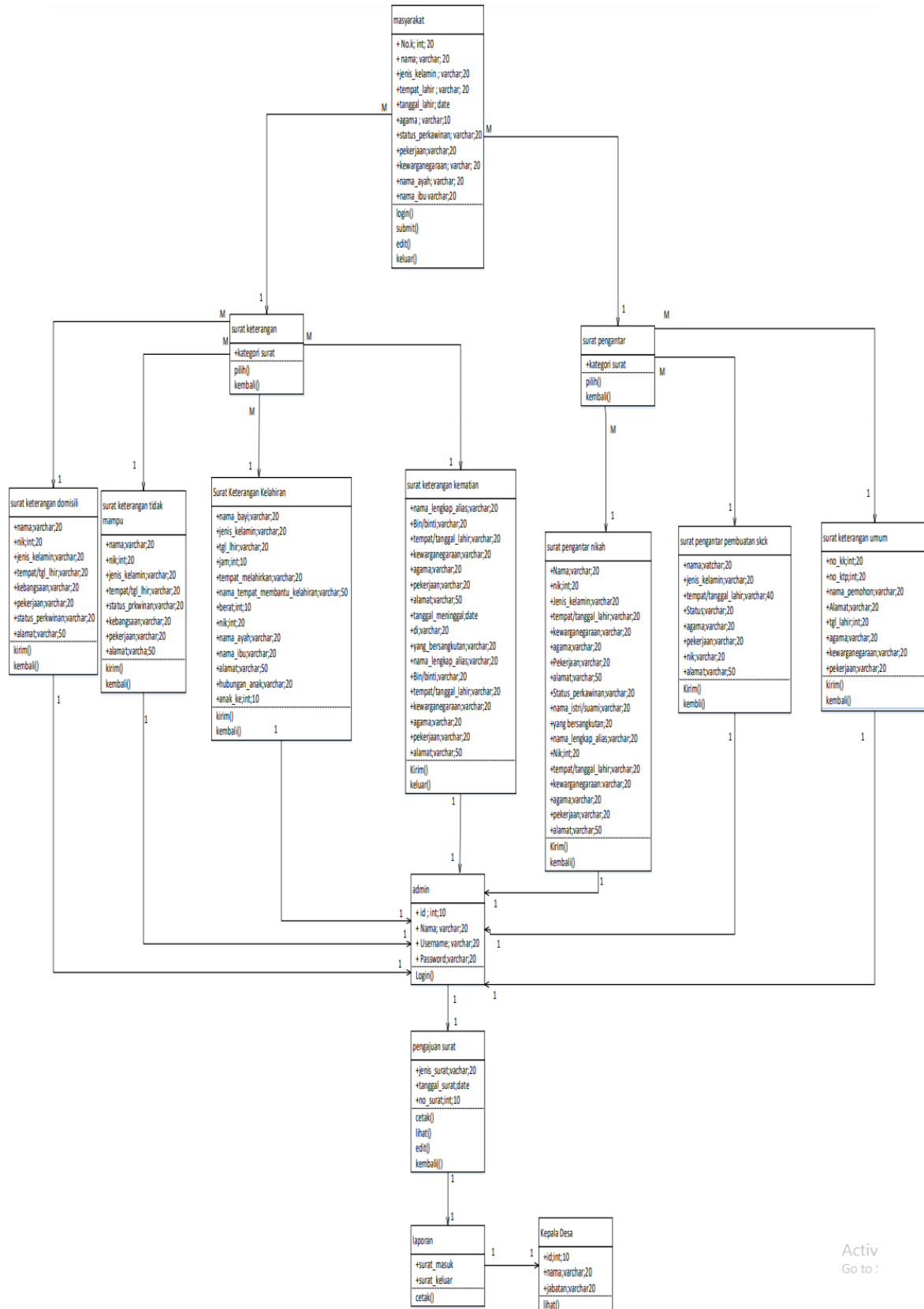+kategori surat

pilih()
kembali()

surat keterangan domisili

+nama;varchar;20
+nik;int;20
+jenis_kelamin;varchar;20
+tempat/tgl_lhir;varchar;20
+kebangsaan;varchar;20
+pekerjaan;varchar;20
+status_perkwinan;varchar;20
+alamat;varchar;50

kirim()
kembali()

surat keterangan tidak mampu

+nama;varchar;20
+nik;int;20
+jenis_kelamin;varchar;20
+tempat/tgl_lhir;varchar;20
+status_prikwinan;varchar;20
+kebangsaan;varchar;20
+pekerjaan;varchar;20
+alamat;varcha;50

kirim()
kembali()

Surat Keterangan Kelahiran

+nama_bayi;varchar;20
+jenis_kelamin;varchar;20
+tgl_lhir;varchar;20
+jam;int;10
+tempat_melahirkan;varchar;20
+nama_tempat_membantu_kelahiran;varchar;50
+berat;int;10
+nik;int;20
+nama_ayah;varchar;20
+nama_ibu;varchar;20
+alamat;varchar;50
+hubungan_anak;varchar;20
+anak_ke;int;10

kirim()
kembali()

surat keterangan kematian

+nama_lengkap_alias;varchar;20
+Bin/binti;varchar;20
+tempat/tanggal_lahir;varchar;20
+kewarganegaraan;varchar;20
+agama;varchar;20
+pekerjaan;varchar;20
+alamat;varchar;50
+tanggal_meninggal;date
+di;varchar;20
+yang_bersangkutan;varchar;20
+nama_lengkap_alias;varchar;20
+Bin/binti;varchar;20
+tempat/tanggal_lahir;varchar;20
+kewarganegaraan;varchar;20
+agama;varchar;20
+pekerjaan;varchar;20
+alamat;varchar;50

Kirim()
keluar()

surat pengantar nikah

+Nama;varchar;20
+nik;int;20
+Jenis_kelamin;varchar20
+tempat/tanggal_lahir;varchar;20
+kewarganegaraan;varchar;20
+agama;varchar;20
+Pekerjaan;varchar;20
+alamat;varchar;50
+Status_perkawinan;varchar;20
+nama_istri/suami;varchar;20
+yang bersangkutan;20
+nama_lengkap_alias;varchar;20
+Nik;int;20
+tempat/tanggal_lahir;varchar;20
+kewarganegaraan;varchar;20
+agama;varchar;20
+pekerjaan;varchar;20
+alamat;varchar;50

Kirim()
kembali()

surat pengantar pembuatan skck

+nama;varchar;20
+jenis_kelamin;varchar;20
+tempat/tanggal_lahir;varchar;40
+Status;varchar;20
+agama;varchar;20
+pekerjaan;varchar;20
+nik;varchar;20
+alamat;varchar;50

Kirim()
kembali()

surat keterangan umum

+no_kk;int;20
+no_ktp;int;20
+nama_pemohon;varchar;20
+Alamat;varchar;20
+tgl_lahir;int;20
+agama;varchar;20
+kewarganegaraan;varchar;20
+pekerjaan;varchar;20

kirim()
kembali()

admin

+ id ; int;10
+ Nama; varchar;20
+ Username;varchar;20
+ Password;varchar;20

Login()

pengajuan surat

+jenis_surat;vachar;20
+tanggal_surat;date
+no_surat;int;10

cetak()
lihat()
edit()
kembali()

laporan

+surat_masuk
+surat_keluar

cetak()

Kepala Desa

+id;int;10
+nama;varchar;20
+jabatan;varchar20

lihat()

Figure 2 – Class Diagram

3.4. System Interface

A system interface refers to the point of interaction or communication between different systems, components, or software modules within a larger system or between separate systems. It defines how

different parts of a system communicate, exchange data, or interact with each other. The admin home page is the page that will be displayed after the admin has successfully entered the system. The main admin page has several menus that make it easier for admins to interact with the database (Figure 3).
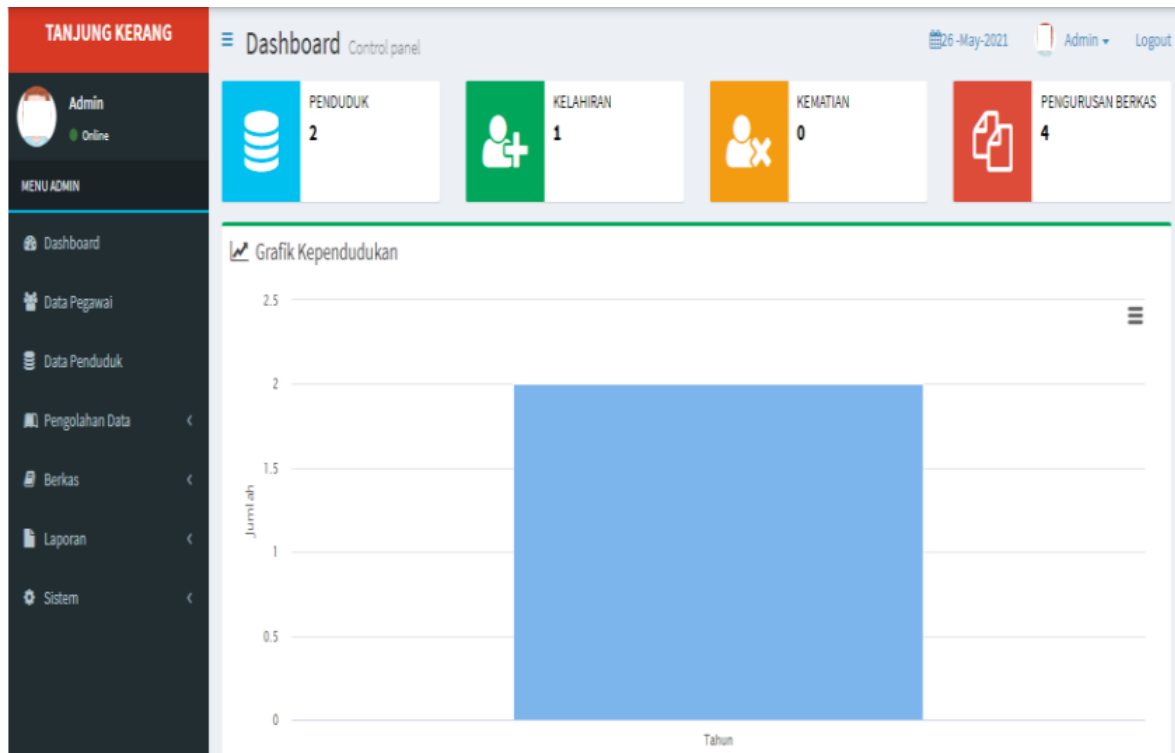

Figure 3 – The admin page

In the end, the author carries out Black Box testing of the app that has been built. Black Box testing focuses on the functional requirements of the software [12]-[15]. Thus, black box testing allows software engineers to obtain a set of input conditions that fully utilize all functional requirements for an app. Black box testing seeks to find errors in the following criteria: Incorrect or missing functions, Interface errors, Errors in data structure or database access, and Performance errors. Based on the test results, overall the app built meets all testing criteria, in line with expectations at the start of the study.

4.   CONCLUSION

This study highlights the challenges faced by Tanjung Kerang Village in providing optimal community administration services due to the use of conventional methods. The need for an information system to enhance the efficiency and effectiveness of service delivery is emphasized. The Prototype Model is then introduced as the chosen methodology for system development, characterized by quick iterations, user involvement, continuous refinement, flexibility in requirements, and early risk identification. The subsequent section outlines the results and discussions, presenting the proposed system's design through UML diagrams and emphasizing the importance of system interfaces, specifically the admin home page. The study concludes with the execution of Black Box testing, validating that the developed application meets functional requirements and aligns with the expectations outlined at the beginning of the research. Overall, the Prototype Model proves instrumental in addressing the challenges faced by Tanjung Kerang Village, offering a systematic and iterative approach to enhance community administration services.

REFERENCES

[1]   A. Adamy and H. A. Rani, "An evaluation of community satisfaction with the government's COVID-19 pandemic response in Aceh, Indonesia," *Int. J. Disaster Risk Reduct.*, vol. 69, no. 91, p. 102723, 2022, doi: 10.1016/j.ijdrr.2021.102723.
[2]   D. P. Pant, B. Acharya, and M. R. Kattel, "Association of government effectiveness, logistics performance, IT systems and income with COVID-19 mortality," *Heliyon*, vol. 9, no. 4, p. e15214, 2023, doi: 10.1016/j.heliyon.2023.e15214.
[3]   M. Riesener *et al.*, "A model for dependency-oriented prototyping in the agile development of complex technical systems,"

Procedia CIRP, vol. 84, no. March, pp. 1023–1028, 2019, doi: 10.1016/j.procir.2019.04.196.

[4] A. Rizzardi, S. Sicari, and A. Coen-Porisini, "Towards rapid modeling and prototyping of indoor and outdoor monitoring applications," *Sustain. Comput. Informatics Syst.*, vol. 41, no. December 2023, p. 100951, 2024, doi: 10.1016/j.suscom.2023.100951.

[5] C. Smyth, S. H. Xie, T. Zaborniak, M. Fellows, C. Phillips, and W. A. Kurz, "Development of a prototype modeling system to estimate the GHG mitigation potential of forest and wildfire management," *MethodsX*, vol. 10, no. October 2022, p. 101985, 2023, doi: 10.1016/j.mex.2022.101985.

[6] G. Bergström *et al.*, "Evaluating the layout quality of UML class diagrams using machine learning," *J. Syst. Softw.*, vol. 192, p. 111413, 2022, doi: 10.1016/j.jss.2022.111413.

[7] H. Wu, "QMaxUSE: A new tool for verifying UML class diagrams and OCL invariants," *Sci. Comput. Program.*, vol. 228, p. 102955, 2023, doi: 10.1016/j.scico.2023.102955.

[8] P. Danenas, T. Skersys, and R. Butleris, "Natural language processing-enhanced extraction of SBVR business vocabularies and business rules from UML use case diagrams," *Data Knowl. Eng.*, vol. 128, no. February, p. 101822, 2020, doi: 10.1016/j.datak.2020.101822.

[9] Meiliana, I. Septian, R. S. Alianto, Daniel, and F. L. Gaol, "Automated Test Case Generation from UML Activity Diagram and Sequence Diagram using Depth First Search Algorithm," *Procedia Comput. Sci.*, vol. 116, pp. 629–637, 2017, doi: 10.1016/j.procs.2017.10.029.

[10] Z. Daw and R. Cleaveland, "Comparing model checkers for timed UML activity diagrams," *Sci. Comput. Program.*, vol. 111, no. P2, pp. 277–299, 2015, doi: 10.1016/j.scico.2015.05.008.

[11] F. Chen, L. Zhang, X. Lian, and N. Niu, "Automatically recognizing the semantic elements from UML class diagram images," *J. Syst. Softw.*, vol. 193, p. 111431, 2022, doi: 10.1016/j.jss.2022.111431.

[12] D. Felicio, J. Simao, and N. Datia, "Rapitest: Continuous black-box testing of restful web apis," *Procedia Comput. Sci.*, vol. 219, no. 2022, pp. 537–545, 2023, doi: 10.1016/j.procs.2023.01.322.

[13] H. Bostani and V. Moonsamy, "EvadeDroid: A Practical Evasion Attack on Machine Learning for Black-box Android Malware Detection," *Comput. Secur.*, p. 103676, 2021, doi: 10.1016/j.cose.2023.103676.

[14] F. Pagano, A. Romdhana, D. Caputo, L. Verderame, and A. Merlo, "SEBASTiAn: A static and extensible black-box application security testing tool for iOS and Android applications," *SoftwareX*, vol. 23, p. 101448, 2023, doi: 10.1016/j.softx.2023.101448.

[15] C. Cronley *et al.*, "Designing and evaluating a smartphone app to increase underserved communities' data representation in transportation policy and planning," *Transp. Res. Interdiscip. Perspect.*, vol. 18, no. January, p. 100763, 2023, doi: 10.1016/j.trip.2023.100763.