Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024
DOI : https://doi.org/10.33395/jmp.v13i1.13623
e-ISSN : 2797-3298
p-ISSN : 2089-9424

Terbit : 02 April 2024

Kompleksitas Fungsional Perangkat Lunak Menggunakan Serangkaian Kriteria Baru dalam Unified Modeling Language (UML)

[1] Ellanda Purwawijaya
[1] Universitas Battuta, Indonesia

[1] ellanda.purwa.wijaya@gmail.com

ABSTRAK

Pengembangan teknologi yang cepat mengakibatkan kompleksitas dan ukuran desain perangkat lunak yang semakin meningkat, yang mengarah pada beban kerja administratif dan pengembangan yang tinggi. Studi ini fokus pada pengukuran kompleksitas perangkat lunak menggunakan desain database dan diagram Unified Modeling Language (UML) untuk mengambil data terkait. Model Entity Relationship (ER) yang dikembangkan oleh Peter Chen menyediakan kerangka kerja konseptual untuk mengklasifikasikan data dalam konteks relasional. Skema ORS (Object Relationship Schema) yang disempurnakan juga diperkenalkan untuk memberikan pandangan yang lebih baik mengenai persyaratan basis data. Kompleksitas sistem perangkat lunak kontemporer juga harus dinilai dalam pendidikan tinggi, yang dipengaruhi oleh faktor seperti ketergantungan fungsional, keamanan, kompleksitas komputasi, kasus penggunaan, dan struktur komponen. Penelitian ini berkontribusi pada pemahaman tentang kompleksitas perangkat lunak dan pentingnya penilaian dalam pendidikan tinggi. (Cohen & Gil, 2021)

Keyword:
Complexity index, complexity multiplier, object relationship model, unified modeling language (UML).

PENDAHULUAN

Menurut Ousterhout, menciptakan perangkat lunak komputer merupakan salah satu kegiatan paling kreatif dalam sejarah manusia. Pemrograman dianggap sebagai upaya kreatif dan artistik yang menantang. Namun, semakin meningkatnya kecanggihan perangkat lunak, proses pembuatan dan penerapan fitur dan ekstensi baru menjadi lebih sulit, berbahaya, dan memakan waktu. Hal ini menyebabkan peningkatan pengeluaran untuk pemeliharaan sistem. Untuk mengatasi kompleksitas perangkat lunak, diperlukan pemahaman yang lebih baik tentang komponen desain yang berkontribusi terhadap kerumitan perangkat lunak. Desain modular dan terorganisir telah terbukti mengurangi kompleksitas perangkat lunak dan program. Konsep objek, khususnya dalam Sistem Pemrograman Berorientasi Objek (OOPS), memberikan manfaat penting dalam mengurangi rumitnya perangkat lunak. Studi ini menekankan kesulitan dalam mendapatkan metrik yang obyektif untuk kompleksitas desain perangkat lunak pada tahap awal proses sebelum pemrograman dimulai. Pendekatan agile development berhasil menggantikan model waterfall tradisional dengan memandang desain sebagai tahap awal pengembangan perangkat lunak. Fase desain menjadi aktivitas berkelanjutan yang penting untuk memahami dan menstandarisasi kompleksitas sistem. Banyak strategi telah diusulkan untuk mengurangi kompleksitas, seperti meminimalkan kode khusus, menggunakan desain modular, dan enkapsulasi kode. Namun, penelitian ini bertujuan untuk memahami dan memperkirakan kompleksitas selama fase pengembangan proyek daripada fokus pada membatasi atau menurunkan kompleksitas.(Marmo et al., 2020)

Ada banyak ketertarikan untuk mengevaluasi kompleksitas fungsional dalam perangkat lunak sejak Albrecht mempresentasikan teknik titik fungsi pada tahun 1979. Evaluasi ulang

Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13623 p-ISSN : 2089-9424

terhadap subjek ini diperlukan karena munculnya ketergantungan fungsional dalam database dan penggunaan pendekatan berbasis skenario dalam pengembangan perangkat lunak, seperti skenario kasus penggunaan atau cerita pengguna. Kami menyediakan metode baru yang mempertimbangkan sejumlah faktor, seperti kasus penggunaan, keamanan, kompleksitas komputasi, dan ketergantungan fungsional. "Kompleksitas muncul dari akumulasi ketergantungan dan ketidakjelasan." Mengingat ketidakpastian universal dari masalah keamanan dalam model HRU, keamanan merupakan masalah lintas sektoral yang dapat berkontribusi secara signifikan terhadap kompleksitas perangkat lunak dalam sebagian besar situasi. Pemeliharaan perangkat lunak menyumbang sejumlah besar biaya pengembangan sistem komputer. Pemeliharaan perangkat lunak, yang melibatkan miliaran baris kode (LOC), merupakan tugas Sistem Informasi (IS) yang penting. Alokasi sumber daya pemeliharaan perangkat lunak perlu bergantung pada fitur perangkat lunak tertentu. Meskipun banyak teknik yang telah dikeluarkan untuk mengukur kompleksitas perangkat lunak, sebagian besar tidak memiliki proses pengukuran yang sederhana. Dua penelitian yang sering dikutip dalam bidang ini adalah metrik McCabe dan teknik Halstead. Pendekatan Halstead menetapkan hubungan matematis antara variabel dan pernyataan pemrograman, sedangkan metrik McCabe menghitung jumlah tepi kontrol dalam kode. Strategi tambahan termasuk metode berdasarkan spesifikasi, seperti metode titik fungsi, atau cakupan kode. Metrik Suite diperkenalkan oleh Chidamber dan Kemerer untuk pengkodean dan desain berorientasi objek. Selain itu, teknik-teknik seperti kompleksitas data (Metrik Chapin), kompleksitas percabangan (Metrik Sneed), kompleksitas akses data (Metrik Card), dan kompleksitas keputusan (Metrik McClure) telah diterapkan. Penelitian ini menyajikan pendekatan kuantitatif untuk mengukur kompleksitas perangkat lunak dengan menggabungkan elemen-elemen analisis kasus penggunaan, abstraksi kompleksitas pengkodean, LOC, dan metode titik fungsi. Penggunaan elemen-elemen seperti tombol dan evolusi berulang dari antarmuka pengguna grafis tidak diperhitungkan oleh metode titik fungsi yang asli. Melalui format yang rinci dan visual seperti diagram kasus penggunaan, analisis kasus penggunaan membuat hubungan antara persyaratan fungsional dan skenario kasus penggunaan.

Kemampuan kami untuk mengelola kumpulan data yang besar dan dinamis telah dimungkinkan oleh kemajuan teknologi pemodelan dan desain basis data yang terus menerus dan cepat. Pada tahun 1970, Codd mempresentasikan model data relasional. Untuk menangani fitur konseptual dan intuitif data, Chen kemudian memperkenalkan model ER. Dalam kerangka kerja artikel ini, objek terdiri dari banyak jenis data, termasuk tetapi tidak terbatas pada data yang terorganisir secara konvensional, data yang tidak terorganisir, data agregat yang ditemukan dalam formulir, kueri SQL, dan laporan, serta data yang terkait dengan gambar, audio, dan video (BLOB). Proses pengembangan sistem secara langsung dipengaruhi oleh meningkatnya kerumitan materi basis data, yang juga memengaruhi aspek-aspek lain dari infrastruktur komputasi seperti pengembangan, pemrosesan, keamanan, pemeliharaan, transmisi, tampilan, dan pengarsipan. Untuk memperkirakan kompleksitas basis data, sejumlah teknik, termasuk pendekatan berbasis spesifikasi seperti metode titik fungsi, telah dikemukakan dalam literatur. Lebih jauh lagi, Briand memperkenalkan kerangka kerja berbasis aksiomatik, sementara Zuse memperkenalkan validasi teoritis dari kerangka kerja berbasis teori pengukuran perangkat lunak. Jumlah foreign key dalam desain database merupakan ukuran yang lebih baik untuk pemahaman skema daripada kedalaman pohon referensial, menurut sebuah penelitian yang berbeda yang menggunakan analisis statistik. Dengan menggunakan indikator basis data seperti atribut, kunci, indeks, dan referensi basis data yang mudah diakses dalam Sistem Manajemen Basis Data Relasional (RDBMS), sebuah teknik yang dikenal sebagai pendekatan DC (Kompleksitas Basis Data) mengevaluasi kompleksitas ini. Pendekatan alternatif tidak hanya mempertimbangkan tabel dan hubungan yang ada dalam desain database, tetapi juga lapisan aplikasi yang mencakup formulir, laporan, dan pertanyaan. Berdasarkan penelitian sebelumnya, penelitian ini menyarankan kompleksitas database sebagai metode kuantitatif untuk menghitung kompleksitas database dengan menggunakan data dari Object

Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13623 p-ISSN : 2089-9424

Relational Schema (ORS) dan Entity Relationship Schema (ERS). Entity Relationship Diagram (ERD), yang mengarah pada skema (ERS), adalah cara konseptual dan mudah untuk membuat model basis data abstrak yang membantu pengembang dan pemrogram mencapai desain yang optimal. ERD berfungsi sebagai deskripsi hubungan di antara elemen-elemen sistem dan diimplementasikan sebagai tabel dengan kunci untuk mempermudah pemodelan relasi. Banyak aspek dari pendekatan ini yang secara rutin diperiksa untuk kemungkinan modifikasi dan perubahan, mengingat tuntutan yang semakin meningkat untuk pengembangan proyek basis data yang gesit dan kepatuhan terhadap tenggat waktu yang ketat. Sinha dkk. menyarankan untuk membuat superset dari ERS yang menguraikan hubungan antara setiap objek dalam sistem berdasarkan teknik pemodelan ERS. Sekelompok tabel yang mendefinisikan entitas dalam ERS merupakan salah satu bagian dari model ini, yang disebut sebagai objek tabel. Formulir, pertanyaan, dan laporan adalah contoh objek lain yang menjadi bagian dari penelitian ini dan dianggap sebagai kebutuhan desain. Hubungan antar objek didefinisikan dengan cara yang sama dengan ERD, sehingga menghasilkan pengembangan Object Relationship Schema (ORS), yang merupakan superset dari ERS. ORS berfungsi sebagai contoh bagaimana satu objek database bergantung pada objek lainnya, menciptakan sistem yang kohesif, tunggal, dan saling berhubungan. Penelitian ini memperkenalkan implementasi objek sebagai cluster dalam Object Relationship Diagram (ORD), yang mirip dengan cara ERS dalam mendefinisikan entitas sebagai tabel. Diklaim bahwa desain ORD baru ini akan sangat meningkatkan kegunaan dan kemampuan pemodelan RDBMS dan pembuatan basis data jika digabungkan dengan ERS saat ini. Selain itu, kombinasi ERS dan ORS memberikan pemahaman yang lebih menyeluruh tentang spesifikasi dan persyaratan database, yang meningkatkan kapasitas tim untuk menjadwalkan dan menetapkan sumber daya untuk pengembangan, pemeliharaan, dan implementasi peningkatan selama masa operasional produk. Kompleksitas database, yang dibahas pada bagian selanjutnya, terkonsentrasi pada jenis data agregat Formulir, Kueri SQL, dan Laporan dan mencakup semua Objek yang telah disebutkan sebelumnya. (Candel et al., 2022)

KOMPLEKSITAS PERANGKAT LUNAK

Penelitian ini menyajikan teknik kuantitatif untuk mengevaluasi kompleksitas perangkat lunak dengan mengintegrasikan beberapa komponen dari metode titik fungsi, baris kode (LOC), "abstraksi kompleksitas pengkodean," dan "diagram kasus penggunaan." Evolusi berulang dari GUI dan penggunaan elemen seperti tombol tidak dipertimbangkan oleh metode titik fungsi yang asli. Salah satu diagram UML yang paling banyak digunakan dalam pengembangan perangkat lunak adalah diagram use case. Use case biasanya dikembangkan sebelum tahap pengkodean, yang memungkinkan estimasi kompleksitas perangkat lunak lebih awal selama pengembangan dan membantu manajemen risiko di tahap selanjutnya. Tabel 1 mencantumkan faktor-faktor yang digunakan untuk memperkirakan Unadjusted Complexity Index (UCI) dan bobot yang telah ditetapkan yang diberikan untuk setiap kategori fungsional**.** (Solihin et al., 2020)

TINJAUAN PUSTAKA

KRITERIA UNTUK ESTIMASI (UCI)	POIN YANG DIREKOMENDASIKAN			
MASUKAN/KELUARAN	SIMPLE	AVERAGE	COMPLEX	SCORES
Jumlah Input	3	4	6	
Jumlah Keluaran	4	5	7	
KOMPONEN				
Jumlah Komponen Interaktif (5 atau kurang)	2	5	9	
Jumlah Komponen Interaktif (6 hingga 20)	10	20	30	

Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13623 p-ISSN : 2089-9424

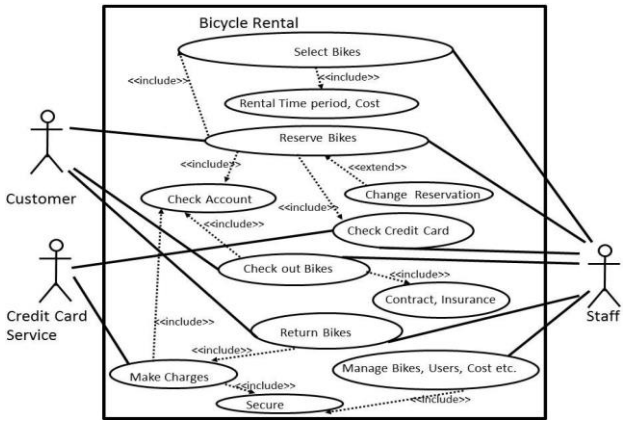
Jumlah Komponen Interaktif (21 atau lebih)	40	50	80	
DIAGRAM KASUS PENGGUNAAN				
Jumlah Kasus Penggunaan	3	5	10	
Jumlah Aktor	2	5	10	
Jumlah Total Antarmuka	5	10	20	
DIAGRAM KELAS				
Jumlah Total Kelas	2	4	8	
Jumlah Total Hubungan Antar Kelas	2	4	8	
Jumlah Total Fungsi	3	6	20	
BASIS DATA: SKEMA HUBUNGAN ENTITAS				
Jumlah Entitas	2	10	20	
Jumlah Tabel	5	10	20	
Jumlah Atribut	2	3	4	
Jumlah Kunci Asing	2	3	4	
Jumlah Tanggungan	2	4	10	
BASIS DATA: SKEMA HUBUNGAN OBJEK				
Jumlah Objek	3	5	8	
Jumlah Cluster	**5**	**10**	**20**	
TINGKAT KEAMANAN				
Keamanan Level 1 (nama pengguna, kata sandi, dll.)	**5**	**10**	**20**	
Keamanan Level 2 (tanpa kepercayaan)	**50**	**100**	**200**	
TOTAL: INDEKS KOMPLEKSITAS YANG TIDAK DISESUAIKAN (UCI)				**TOTAL**

METODE PENELITIAN

Dalam desain database relasional, hubungan antara setiap entitas dalam database ditunjukkan melalui Entity Relationship Schema (ERS). Ketergantungan fungsional adalah hal yang penting dalam database relasional. Data yang tidak akurat dapat dihasilkan dalam database jika masalah ini tidak diselesaikan pada fase awal proses desain. Normalisasi adalah proses menghilangkan ketergantungan fungsional, yang mengarah pada peningkatan jumlah tabel dalam database. Database yang dinormalisasi dengan tepat meningkatkan kompleksitas implementasi dan membutuhkan mesin database yang lebih canggih di dalam RDBMS. Seperti halnya ERS, model ER konvensional tidak dapat menangkap hubungan yang ada antara objek-objek ini dan objek-objek lain dalam sistem. Elemen desain tambahan menguraikan spesifikasi objek sistem lainnya, seperti formulir, kueri SQL, dan laporan, untuk menutup kesenjangan ini. Perspektif yang mencakup semua spesifikasi yang diperlukan diberikan oleh Object Relationship Schema (ORS), yang merepresentasikan asosiasi dan hubungan antara setiap Objek dalam sistem jaringan ini.

Skema yang lebih baik yang disebut Object Relationship Schema (ORS) dibuat dengan menggunakan teknik pemodelan ERS. Ketika dipasangkan dengan ERS, ORS memberikan pemahaman yang menyeluruh tentang persyaratan desain database. Semua objek dalam database saling terkait dan harus bekerja sama sebagai satu kesatuan yang koheren; hubungan di antara mereka ditangkap oleh ORS. Dengan memperluas atau menggantikan ERS, ORS menggunakan ide untuk mengekspresikan objek sebagai kelompok, yang sebanding dengan cara ERS dalam mendeskripsikan sesuatu sebagai tabel. Kemampuan RDBMS sangat ditingkatkan dengan integrasi model ORS baru dengan kerangka kerja ERS saat ini, yang menawarkan pemahaman yang lebih dalam tentang basis data dan persyaratan desainnya. Peningkatan ini mendukung implementasi proyek yang efektif dan mengoptimalkan proses pengembangan. Selama fase desain, kompleksitas basis data dievaluasi dengan memanfaatkan wawasan dari penelitian titik fungsi, ORS, dan masukan dari ERS. Tabel 1 memberikan deskripsi menyeluruh tentang kriteria penilaian.

Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13623 p-ISSN : 2089-9424



Gambar 1 Entity Relationship Schema (ERS)

HASIL DAN PEMBAHASAN

Tabel 2. Faktor dan Perhitungan Indeks Kompleksitas

PERTANYAAN BERBASIS FAKTOR PENGARUH	PENGARUH (0-5)
1. Apakah sistem memerlukan pencadangan dan pemulihan yang andal?	5
2. Apakah ada interaksi yang kompleks di antara komponen?	4
3. Apakah ada komponen yang diharapkan bekerja dalam mode otonom?	4
4. Apakah ada komponen yang memiliki masalah yang sulit diatasi?	0
5. Apakah ada masalah keamanan yang belum diketahui solusi algoritmanya?	4
6. Apakah entri data on-line memerlukan transaksi input yang dibuat melalui beberapa layar atau operasi?	2
7. Apakah sistem ini diperlukan untuk tindakan yang sangat penting bagi keselamatan?	1
8. Apakah input, output, file, atau pertanyaan yang rumit?	3
9. Apakah pemrosesan internalnya rumit?	2
10. Apakah kode harus dapat digunakan kembali?	3
11. Apakah sistem dirancang untuk beberapa instalasi di lokasi yang berbeda?	2
12. Apakah sistem diperlukan untuk menangani data dalam jumlah besar?	1
13. Apakah sistem dirancang untuk memfasilitasi perubahan dan kemudahan penggunaan oleh pengguna?	4
14. Faktor lainnya: (Penjelasan: Kecelakaan, Pengecualian, Pelanggaran Kontrak, dll.)	4
Jumlah faktor Pengaruh **Pengganda Kompleksitas** = 0.65 + 0.01*(Jumlah Faktor Pengaruh) **Indeks Kompleksitas (CI) =** Pengali Kompleksitas * UCI	39 1.04 **4045**

Masalah penyewaan sepeda memiliki Indeks Kompleksitas (CI) sebesar 4045, yang diperoleh dengan mengalikan Indeks Kompleksitas Tidak Disesuaikan (UCI), (Ji et al., 2022) yang, menurut Tabel 2, adalah 3889, dikalikan dengan 1,04 (pengali kompleksitas), seperti yang ditunjukkan pada Tabel 3. Khususnya, karya Albrecht menjadi inspirasi utama untuk kriteria yang disebutkan dalam Tabel 3. Faktor nomor 14 pada Tabel 3 menambahkan elemen baru yang memberikan fleksibilitas kepada para pengembang peserta dengan mengizinkan "Faktor lainnya" dengan penjelasan. (Li et al., 2021) Elemen ini memperhitungkan banyak masalah yang dihadapi pengembang, seperti pertimbangan sosial, kendala khusus proyek, persyaratan peraturan, dan masalah kepatuhan. CI (4045) dapat direvisi selama fase pengembangan berulang dan didasarkan pada pekerjaan pertama kami dari tahap awal pengembangan. (Muhammad Rhifky Wayahdi et al., 2021) Kasus penggunaan diperiksa, diagram kasus penggunaan dibuat, dan analisis persyaratan awal dilakukan sebelum nilai kompleksitas dihitung. Meskipun kasus penggunaan, juga dikenal sebagai cerita pengguna, penting

Jurnal Minfo Polgan
Volume 13, Nomor 1, April 2024 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v13i1.13623 p-ISSN : 2089-9424

dalam penelitian kompleksitas perangkat lunak kami, metode poin kasus penggunaan terlalu terbatas dan tidak memperhitungkan banyak fitur perangkat lunak yang dibahas dalam penelitian ini sebagai kriteria baru. (Mughnyanti & Ginting, 2023) Sistem penyewaan sepeda digunakan untuk mengilustrasikan bagaimana kriteria-kriteria tersebut digabungkan berdasarkan karya terobosan Albrecht karena ini adalah masalah yang relatif sederhana. Hitungan file logis Albrecht telah diganti pada Tabel 1 dan 2 dengan pengertian yang berkaitan dengan basis data modern. Cara kelemahan keamanan sistem komputer berinteraksi secara rumit dengan beberapa aspek kebutuhan program, desain, dan pengembangan merupakan faktor penting dalam kompleksitas perangkat lunak. Sebagai pendekatan yang disempurnakan, kami mengganti metode titik fungsi Albrecht, yang berguna untuk menentukan ukuran fungsional perangkat lunak pada dekade sebelumnya, dengan penekanan pada kompleksitas fungsional dan bukan pada ukuran. (Ginting & others, 2024) Analisis kasus penggunaan, ketergantungan basis data, keamanan sistem komputer, GUI, dan elemen lainnya dipertimbangkan dalam metode ini, yang melampaui pendekatan berorientasi ukuran untuk mengevaluasi kompleksitas fungsional perangkat lunak secara akurat. (Liu et al., 2021) Saran dari International Function Point Users Group (IFPUG) dipersilakan dan harus dipertimbangkan untuk berbagai sudut pandang. (M Rhifky Wayahdi et al., 2020)

KESIMPULAN

Studi ini mengusulkan pendekatan yang terinspirasi oleh metode titik fungsi untuk pengukuran kompleksitas perangkat lunak. Metode ini menetapkan dua kata dan memperkenalkan seperangkat kriteria baru: Indeks Kompleksitas (CI) dan Indeks Kompleksitas yang Tidak Disesuaikan (UCI). (Rowshan et al., 2021) Diagram kasus pengguna, gagasan logis tentang abstraksi blok kode bangunan, dan upaya GUI semuanya bekerja sama untuk membenarkan frasa-frasa ini. Selain itu, pendekatan ini dapat mengevaluasi bagaimana kompleksitas program memengaruhi biaya proyek pemeliharaan perangkat lunak dalam konteks SI. Dalam hal desain basis data, kerumitan yang timbul dari desain data melampaui penelitian sebelumnya dalam domain ini. Bergantung pada fitur-fitur desain dan pengembangan komponen basis data, pendekatan ini dapat digunakan untuk memperkirakan kompleksitas berbagai sistem. Pendekatan ini, yang memperhitungkan berbagai faktor, sangat membantu dalam pendidikan tinggi untuk mempersiapkan tenaga kerja baru yang dibutuhkan industri perangkat lunak abad ke-21. Ketika mahasiswa menghadapi bagian-bagian praktis rekayasa perangkat lunak, kami percaya bahwa insinyur perangkat lunak akan menemukan seperangkat kriteria ini membantu. Topik penelitian masa depan yang mungkin mencakup kognisi manusia, jenis ketergantungan fungsional, metadata, taktik pemahaman, pola, organisasi basis data semi-struktural dan tak terstruktur, dan strategi pengembangan perangkat lunak lainnya yang berkembang. Studi masa depan mungkin memberikan perhatian khusus pada masalah sosial yang terkait dengan pandemi COVID-19. Kemampuan untuk merencanakan dan menetapkan sumber daya yang tepat untuk pengembangan dan pemeliharaan sistem perangkat lunak sepanjang siklus operasional mereka akan ditingkatkan oleh pendekatan holistik ini, yang akan menghasilkan perkiraan yang lebih akurat dari keseluruhan persyaratan dan spesifikasi sistem. Makalah ini menyoroti pentingnya kejelasan konseptual terkait kompleksitas fungsional, memberikan dasar untuk penelitian masa depan yang dapat mengeksplorasi spesifik metodologis menggunakan kasus dari bidang yang berbeda. Kami berharap akan melihat lebih banyak perkembangan dalam area ini dalam beberapa tahun mendatang.

REFERENSI

Candel, C. J. F., Ruiz, D. S., & Garc\'\ia-Molina, J. J. (2022). A unified metamodel for NoSQL and relational databases. *Information Systems*, *104*, 101898.

Cohen, J., & Gil, J. (2021). An entity-relationship model of the flow of waste and resources in city-regions: Improving knowledge management for the circular economy. *Resources, Conservation \& Recycling Advances*, *12*, 200058.

Ginting, S. H. N., & others. (2024). Penerapan Algoritma k-means dalam Data Mining untuk Mengidentifikasi Strategi Promosi di Politeknik Ganesha Medan. *Jurnal Minfo Polgan*, *13*(1), 189–196.

Ji, H., Ye, K., Wan, Q., & Shen, L. (2022). Reasonable object detection guided by knowledge of global context and category relationship. *Expert Systems with Applications*, *209*, 118285.

Li, Z., Liu, Y., Liu, J., Yuan, Y., Raza, A., Huo, H., & Fang, T. (2021). Object relationship graph reasoning for object detection of remote sensing images. *2021 6th International Conference on Image, Vision and Computing (ICIVC)*, 43–48.

Liu, Y., Zhang, D., Zhang, Q., & Han, J. (2021). Integrating part-object relationship and contrast for camouflaged object detection. *IEEE Transactions on Information Forensics and Security*, *16*, 5154–5166.

Marmo, R., Polverino, F., Nicolella, M., & Tibaut, A. (2020). Building performance and maintenance information model based on IFC schema. *Automation in Construction*, *118*, 103275.

Mughnyanti, M., & Ginting, S. H. N. (2023). Data Mining Manhattan Distance dan Euclidean Distance Pada Algoritma X-Means Dalam Klasifikasi Minat dan Bakat Siswa. *REMIK: Riset Dan E-Jurnal Manajemen Informatika Komputer*, *7*(1), 835–842.

Rowshan, M., Burg, A., & Viterbo, E. (2021). Polarization-adjusted convolutional (PAC) codes: Sequential decoding vs list decoding. *IEEE Transactions on Vehicular Technology*, *70*(2), 1434–1447.

Solihin, W., Dimyadi, J., Lee, Y.-C., Eastman, C., & Amor, R. (2020). Simplified schema queries for supporting BIM-based rule-checking applications. *Automation in Construction*, *117*, 103248.

Wayahdi, M Rhifky, Syahputra, D., & Ginting, S. H. N. (2020). Evaluation of the K-Nearest Neighbor Model With K-Fold Cross Validation on Image Classification. *INFOKUM*, *9*(1, Desember), 1–6.

Wayahdi, Muhammad Rhifky, Ginting, S. H. N., & Syahputra, D. (2021). Greedy, A-Star, and Dijkstra's algorithms in finding shortest path. *International Journal of Advances in Data and Information Systems*, *2*(1), 45–52.