Semantic Literature Review Pada Survey Teknik Pemeliharaan Software

Symantic Literature Review on Software Maintenance Engineering Survey

Lailatul Fadilah[1], Adisa Dwi Wanti[2], Salma Zulfatul Latifah[3], Muhammad Ainul Yaqin[4,*]
[1,2,3] Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang
*e-mail: *119650032@student.uin-malang.ac.id, [2]19650037@student.uin-malang.ac.id, [3]19650038@student.uin-malang.ac.id*

Abstrak

Teknik pemeliharaan software adalah segala metode yang berhubungan dengan aktivitas untuk menjaga life-cycle sebuah software. Survey yang dilakukan terhadap teknik pemeliharaan software sudah pernah dilakukan oleh peneliti sebelumnya termasuk Lientz, Swanson, dll. Survey kali ini mengangkat latar belakang masalah yang berbeda; yaitu pesatnya perkembangan jenis teknik pemeliharaan yang tidak sebanding dengan kepastian tingkat akurasi, keefektifan penggunaan serta cost yang harus dikeluarkan. Survey yang dimaksud bertujuan untuk memberikan tinjauan komprehensif kepada pembaca tentang kegunaan, kelebihan, kekurangan teknik-teknik pemeliharaan software dalam penelitian kurun waktu 1978-2020. Metode survey yang digunakan adalah systematic literature review terhadap paper significant yang membahas isu terkait. Manfaat yang diharapkan, dapat memberikan pembaca gambaran dan pilihan penggunaan teknik pemeliharaan yang tepat guna untuk masalah pemeliharaan yang mereka hadapi. Survey menghasilkan 24 jenis teknik software. Tujuh diantaranya (reverse engineering, corrective, perfective, adaptive, restructuring, model maintenance, reengineering) masuk ke dalam klasifikasi teknik yang paling sering dibahas dalam penelitian kurun waktu 1978-2020. Dari hasil survey tersebut, didapatkan temuan berupa rekomendasi pilihan teknik pemeliharaan untuk contoh kasus Pembuatan framework maintenance, Pemeliharaan pada aplikasi berbasis web, dan Pemeliharaan Cryptologic System.

***Katakunci**: SLR, survey, teknik pemeliharaan software*

Abstrack

Software maintenance techniques are all methods related to activities to maintain the life-cycle of a software. Surveys conducted on software maintenance techniques have been carried out by previous researchers including Lientz, Swanson, etc. This survey raises a different problem background; namely the rapid development of types of maintenance techniques that are not comparable with the certainty of the level of accuracy, effectiveness of use and costs that must be incurred. The intended survey aims to provide readers with a comprehensive overview of the uses, advantages, disadvantages of software maintenance techniques in research for the 1978-2020 period. The survey method used is a systematic literature review of significant papers that discuss related issues. The expected benefits can provide readers with an overview and choice of using appropriate maintenance techniques for the maintenance problems they face. The survey resulted in 24 types of software techniques. Seven of them (reverse engineering, corrective, perfective, adaptive, restructuring, model maintenance, reengineering) fall into the classification of techniques most frequently discussed in research for the 1978-2020 period. From the results of the survey, findings were obtained in the form of recommendations for maintenance technique choices for case examples Making maintenance frameworks, Maintenance of web-based applications, and Maintenance of Cryptologic Systems.

***Keyword**: SLR, survey, software maintenance techniques*

1. PENDAHULUAN

Pemeliharaan *software* merupakan segala kegiatan atau aktivitas yang dilakukan untuk menjaga atau memelihara kinerja *software* [1]. Pemeliharaan *software* sendiri dilakukan untuk memperbaiki kesalahan pada *software* dan meningkatkan akurasi kinerja pada *software*.

History of article:
Received: November, 2022 : Accepted: November, 2022

Pemeliharaan *software* sendiri memerlukan teknik-teknik tertentu untuk pemeliharaan tersebut. Teknik-teknik pemeliharaan *software* tersebut menjalankan beberapa aktivitas, diantaranya adalah memperbaiki kualitas dan akurasi sebuah *software*, melakukan penyesuaian ulang *software* dengan lingkungan mesin yang baru, serta menyelesaikan masalah-masalah yang timbul pada *software* [2]. Masalah yang dirumuskan dalam paper *survey* ini terkait dengan perkembangan teknik pemeliharaan *software* yang selama ini memiliki kualitas yang baik, namun belum dapat memberikan kepastian tingkat akurasi atau keberhasilan di berbagai kasus pemeliharaan. Dalam kebanyakan kasus, teknik pemeliharaan tidak dapat memberikan solusi yang sebanding dengan biaya pemeliharaan yang dikeluarkan. Dengan kata lain, biaya yang dikeluarkan relatif mahal namun tingkat keberhasilannya kecil [3]. Artinya, masih banyak pemilihan penggunaan teknik pemeliharaan *software* yang kurang sesuai dengan jenis kasus pemeliharaan yang dihadapi.

Salah satu cara untuk menyelesaikan permasalahan di atas adalah dengan memberikan kajian dan daftar rekomendasi teknik-teknik pemeliharaan *software* dalam menyelesaikan berbagai kasus pemeliharaan *software*. Sehingga para *maintainers* memiliki gambaran teknik mana yang harus dipilih untuk menyelesaikan kasus pemeliharaan yang sedang dihadapi. Sejauh ini, belum ada peneliti yang secara fokus menyajikan daftar rekomendasi teknik-teknik pemeliharaan *software* tersebut. Oleh karena itu, urgensi dari *survey* yang dilakukan adalah mencari daftar teknik-teknik yang dapat direkomendasikan untuk menyelesaikan masalah-masalah terkait pemeliharaan *software* secara tepat-guna.

Survey bertujuan untuk memberikan tinjauan secara komprehensif mengenai teknik-teknik pemeliharaan *software* yang pernah digunakan atau diulas oleh para peneliti dari tahun ke tahun dalam kurun waktu tahun 1978-2020. Terutama memberikan ulasan tentang teknik yang memiliki tingkat akurasi tinggi dan tidak memakan banyak biaya dalam penggunaannya. Hal tersebut penting, karena jika teknik pemeliharaan *software* yang digunakan salah, maka akan banyak kerugian dari segi materi. Sebaliknya, jika teknik pemeliharaan *software* yang digunakan sudah tepat dan sesuai, maka proses pemeliharaan akan lebih efisien baik dari segi waktu maupun materi. *Survey* yang dilakukan menerapkan metode *systematic literature review* atau studi literatur sistematik terhadap paper-paper penelitian yang membahas mengenai teknik-teknik pemeliharaan *software*. Dimulai dari mengumpulkan data primer dan sekunder, melakukan review semua paper, menemukan jurnal dan paper yang relevan membahas isu terkait.

Survey terhadap teknik-teknik pemeliharaan *software* sudah pernah dilakukan oleh peneliti sebelumnya. Salah satu fakta terkait yang disepakati banyak peneliti bahwa teknik pemeliharaan *software* dikategorikan menjadi 4; *Corrective maintenance* [4][5][6], *Adaptive maintenance* [4][3][5], *Preventive maintenance* [7]. Dari 4 kategori tersebut mengalami percabangan dan lahirlah teknik-teknik pemeliharaan *software* yang berdiri sendiri dari segi jenis penggunaan dan area implementasi. Berdasarkan *survey* yang kami lakukan, dihasilkan 24 teknik pemeliharaan *software* antara lain teknik *Corrective Maintenance, Perfective Maintenance, Adaptive Maintenance, Reverse Engineering, Design Recovery, Data Abstraction, Decompilation, Maintenance Model, Verification and Validation, Inventory Analysis, Program and Data Restructuring, Forward Engineering, Control and Data Flow, Program Slicing, Metrics, Reengineering, Specification-based Approach to maintenance, Hill Climbing Technique, Genetic Algorithms Technique, Multi-objective evolutionary algorithms, Data Mining, Association Rules Algorithm, Classification Algorithm, Associative Classification technique*.

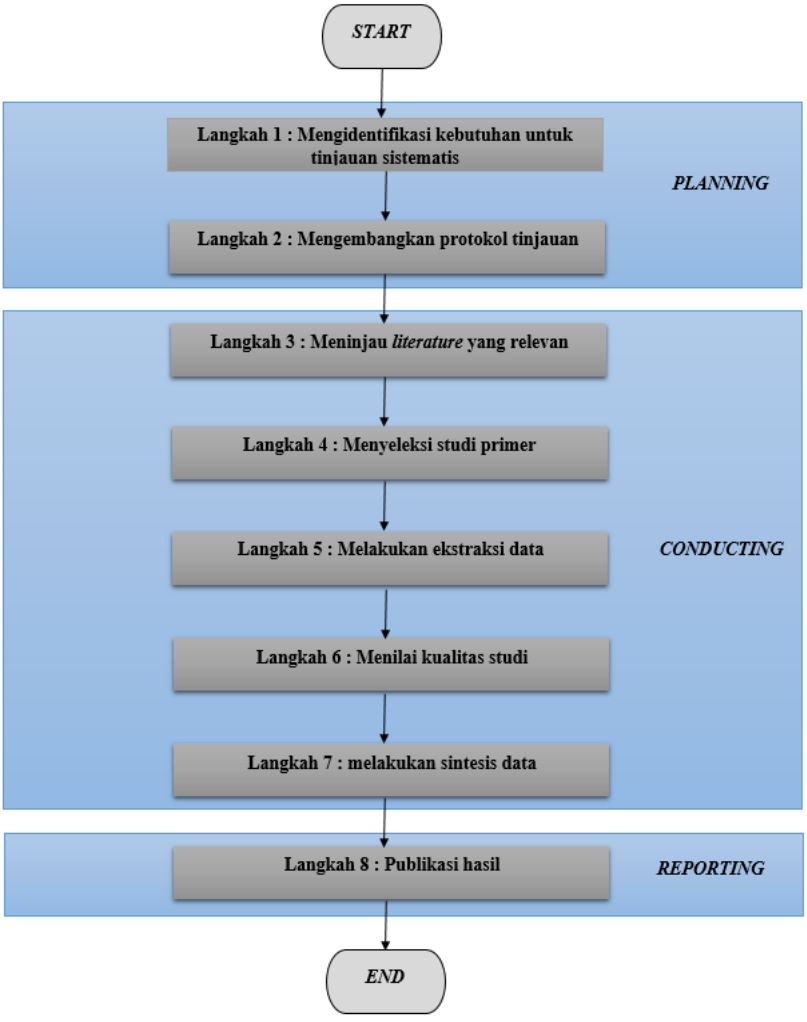
Solusi yang ditawarkan atas penyelesaian masalah adalah dengan membandingkan satu teknik dengan lainnya, ditinjau dari banyak sisi; 1) area implementasi, 2) kekurangan teknik, 3) riwayat penelitian. Juga melakukan klasifikasi teknik berdasarkan jumlah penelitian dan ketepatgunaannya dalam kasus-kasus tertentu. Kasus yang dibahas dalam paper ini antara lain pembuatan *framework maintenance*, pemeliharaan pada aplikasi berbasis web, dan Pemeliharaan *Cryptologic System.* Dari analisa perbandingan tersebut, kemudian dapat ditarik kesimpulan teknik pemeliharaan mana yang tepat untuk masing-masing kasus. *Tepat* disini berarti yang memiliki tingkat keberhasilan tinggi dan area implementasi yang sesuai dengan kasus yang dihadapi.

Hasil dari *survey* ini berupa ulasan tentang area implementasi, kekurangan, riwayat penelitian masing-masing dari 24 teknik pemeliharaan *software* yang ditemukan. *Survey* ini juga akan menghasilkan rekomendasi pilihan teknik pemeliharaan untuk 3 kasus yang dibahas di dalamnya. Juga analisa dalam bentuk tabel dan diagram yang akan memberikan informasi kepada pembaca tentang jumlah penggunaan, nama peneliti, dan tahun penelitian setiap teknik pemeliharaan. Manfaat yang diharapkan, *maintainers* dapat merefleksikan contoh kasus dalam paper ini dengan kasus pemeliharaan yang sedang mereka hadapi. Sehingga mereka dapat menemukan solusi menggunakan rekomendasi pilihan teknik yang sesuai dengan kasus mereka. Sedangkan untuk *researchers*, hasil dari *survey* ini dapat dijadikan dasar untuk menggali teknik pemeliharaan *software* lain atau yang terbaru.

2. METODE PENELITIAN

Systematic Literature Review

Systematic Literature Review (SLR) adalah metode *literature review* yang mengidentifikasi, menilai, dan menginterpretasi seluruh temuan-temuan pada suatu topik penelitian, untuk menjawab pertanyaan penelitian (*research question*) yang telah ditetapkan [8]. Metode SLR dilakukan dengan me-*review* dan mengidentifikasi jurnal secara sistematis dengan mengikuti langkah-langkah yang telah ditentukan. Tujuan penggunaan metode SLR adalah untuk menyusun kerangka pemikiran yang jelas berdasarkan rumusan masalah yang akan diteliti[9]. Gamabaran mengenai Langkah dari SLR ditunjukkan seperti gambar 1.



Gambar 1. Langkah SLR

Research Question (RQ)

Research question adalah pertanyaan mengenai penelitian yang singkat, kompleks, dan terfokus pada topik/permasalahan tertentu[10]. Adapun RQ pada penelitian ini teradapat pada table 1.

Tabel 1. Research Question

RQ	*Research Question*	Tujuan
RQ1	Jurnal apa saja yang signifikan membahas teknik pemeliharaan *software*?	Identifikasi jurnal yang signifikan dalam bidang teknik pemeliharaan *software*
RQ2	Teknik apa saja yang digunakan oleh para peneliti dalam pemeliharaan *software*?	Identifikasi teknik yang digunakan peneliti dalam pemeliharaan *software*
RQ3	Teknik apa yang paling banyak digunakan para peneliti dalam kurun waktu 1978-2021?	Identifikasi teknik pemeliharaan software yang paling banyak digunakan peneliti
RQ4	Rekomendasi teknik apa yang cocok digunakan untuk contoh studi kasus seperti Pembuatan *framework maintenance*, Pemeliharaan pada aplikasi berbasis web, dan Pemeliharaan Cryptologic System ?	Identifikasi rekomendasi teknik pemeliharaan software yang cocok digunakan untuk studi kasus seperti Pembuatan *framework maintenance*, Pemeliharaan pada aplikasi berbasis web, dan Pemeliharaan Cryptologic System

RQ 1 membantu dalam menyaring dan mengevaluasi literatur studi utama. RQ2 membantu dalam memperoleh informasi mengenai teknik pemeliharaan *software*. RQ3 memberi pemahaman dalam penggunaan teknik pemeliharaan *software* pada suatu permasalahan. RQ4 memberikan rekomendasi mengenai teknik pemeliharaan *software* yang cocok dalam beberapa kasus.

Strategi Pencarian

Proses pencarian dilakukan dengan beberapa aktivitas, seperti menentukan string pencarian yang berkaitan dengan topik permasalahan, memilih perpustakaan digital, melakukan penyaringan, dan mengambil daftar awal studi yang relevan dengan string pencarian[10]. Pencarian sumber literatur dilakukan tiga kali dalam waktu yang berbeda dengan kualifikasi jurnal atau paper atau buku yang memiliki judul yang relevan. Pencarian awal dilakukan dari 12 februari 2021 hingga 8 April 2021 pada database digital berikut: IEEE, google scholar, springer link, ACM, National Technical University of Athens dan IET. Dalam pencarian tersebut, penyurvei menemukan 20 hasil gabungan paper dan buku. Pencarian kedua dilakukan dari 18 - 19 april 2021 pada database digital IEEE dan researchgate. Penyurvei menemukan 19 paper yang memiliki judul relevan. Pencarian ketiga dilakukan pada 29 April 2021 pada database digital google books dan World Scientific Publishing menemukan 4 paper yang relevan.

Seleksi Studi

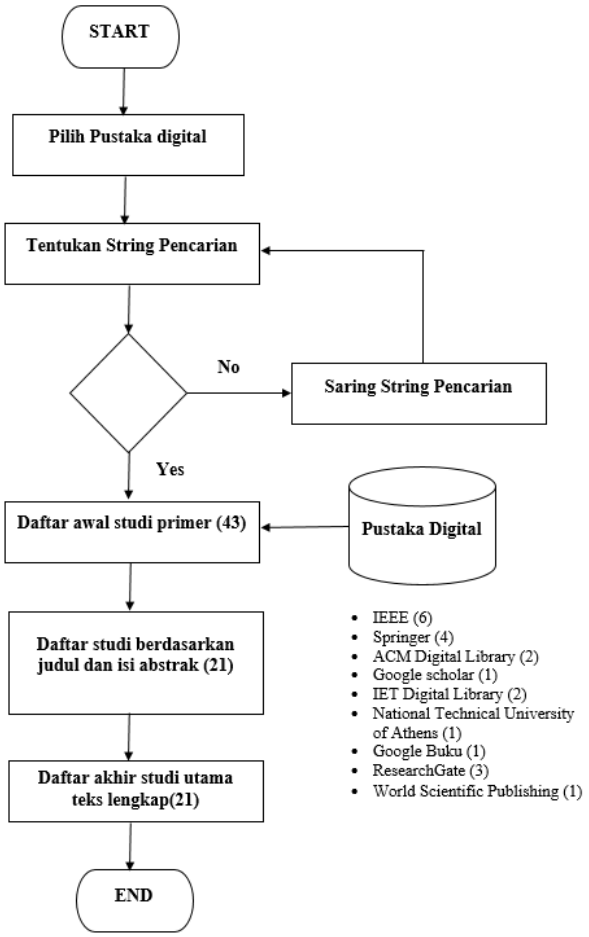
Pada tahap ini dilakukan penyaringan studi primer dengan *mereview* judul dan abstrak jurnal./ Sebelum melakukan penyaringan, menentukan kriteria *inklusi* dan *eksklusi* sehingga dapat membantu memilih studi utama yang relevan[11]. Dimana Kriteria inklusi dan eksklusi harus didasarkan pada pertanyaan penelitian [11].

Tabel 2. Kriteria Inklusi dan Eksklusi

Kriteria *Inklusi*	Kriteria *Eksklusi*
Studi berupa paper, jurnal dan buku	Studi berupa selain paper, jurnal dan buku
Studi yang membahas teknik pemeliharaan software	-

Studi yang membahas masalah pada software dan cara mengatasinya	Studi yang membahas masalah pada software tanpa membahas solusi
Studi yang membahas tingkat efektifitas dari teknik pemeliharaan software	-

Proses rinci pencarian dan jumlah studi yang diidentifikasi pada setiap tahap ditunjukkan pada gambar 2.



Gambar 1. Proses Pencarian dan Seleksi Studi

Daftar awal studi primer pada tahap pertama terdapat 43 studi (termasuk jurnal, buku dan makalah). Selain merujuk pada kriteria *inklusi* dan *eksklusi*, kualitas studi utama; tingkat relevansi dengan *research question* dan kesamaan studi juga dipertimbangkan dalam tahap ini. Studi yang relevan pada penyaringan berdasarkan judul dan isi abstrak (langkah 5) ada 21. Pada daftar akhir studi utama teks lengkap terdapat 21 jurnal.

Data Collection

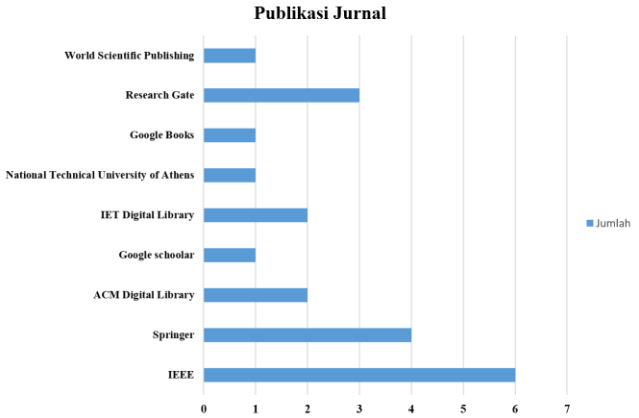
Data collection adalah proses mengumpulkan dan memastikan informasi *variable of interest* dengan cara yang sistematis [10]. Kualifikasi data primer pada penelitian ini adalah paper dengan judul dan abstrak yang membahas secara langsung mengenai teknik pemeliharaan *software*[10].

Analisis Data

Tahap ini dirancang untuk mengumpulkan data pada studi utama untuk menjawab *research question* [10].

Hasil Pencarian

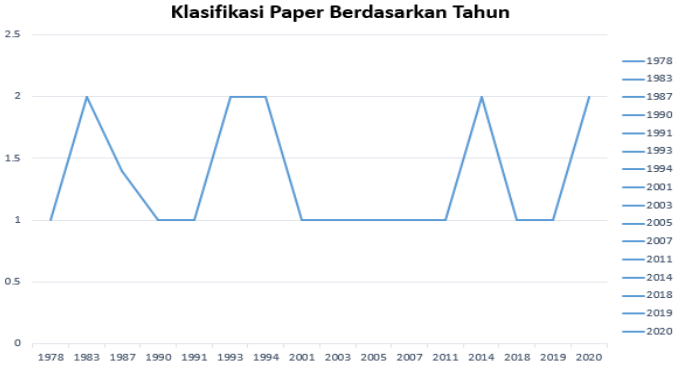
Gambar 2 menunjukkan diagram jumlah paper yang diambil untuk dataset pada perpustakaan digital.



Gambar 2. Jumlah Jurnal signifikan pada Perpustakaan Digital

Klasifikasi Jurnal

Gambar 3 di bawah ini menunjukkan diagram klasifikasi jurnal berdasarkan tahun publikasi dengan rentang waktu 1978-2020.



Gambar 4 Klasifikasi Paper Berdasarkan Tahun

Klasifikasi Paper yang Membahas Teknik Pemeliharaan *Software*

Setelah dilakukan penyaringan terhadap referensi paper yang ditemukan, terdapat 17 paper yang langsung membahas mengenai teknik-teknik pemeliharaan *software*

Kebaruan Ilmiah

Sebagai kontribusi ilmiah dan pembeda dengan hasil *survey* oleh penyurvei-penyurvei sebelumnya, paper ini memuat kebaruan ilmiah yaitu hasil analisis berupa rekomendasi teknik pemeliharaan yang dapat diimplementasikan di berbagai jenis kasus.

a. Rekomendasi teknik pemeliharan *software* dalam kasus Pembuatan *framework maintenance*.
b. Rekomendasi teknik pemeliharan *software* dalam kasus Pemeliharaan pada aplikasi berbasis web.
c. Rekomendasi teknik pemeliharan *software* dalam kasus Pemeliharaan *Cryptologic System.*

3. HASIL DAN PEMBAHASAN

Skenario Analisis terhadap Hasil *Survey*

Analisis dilakukan untuk mengklasifikasi teknik-teknik pemeliharaan *software* berdasarkan batasan-batasan tertentu. Tujuan yang akan dicapai dari analisis ini adalah menemukan variabel-

variabel untuk menjawab permasalahan yang diajukan dalam paper *survey* ini. Berikut adalah skenario analisis yang dilakukan.

a. Analisis Pertama (Jawaban RQ2) :
   Membandingkan teknik-teknik pemeliharaan *software* berdasarkan area implementasi, kekurangan, serta riwayat penelitiannya.
b. Analisis Kedua (Jawaban RQ3) :
   Mengklasifikasi teknik-teknik pemeliharaan *software* yang paling sering digunakan dalam penelitian (kurun waktu 1978-2020).
c. Analisis Ketiga (Jawaban RQ4):
   Menganalisa rekomendasi teknik-teknik pemeliharaan *software* yang cocok digunakan dalam 3 contoh studi kasus; *Pembuatan framework maintenance, Pemeliharaan pada aplikasi berbasis web, dan Pemeliharaan Cryptologic System.*

Hasil Analisis Pertama (RQ2)

Berikut adalah hasil analisis penyurvei terhadap perbandingan teknik-teknik pemeliharaan *software* yang digunakan dalam penelitian (kurun waktu 1978-2020).

Tabel 6. Perbandingan Teknik-Teknik Pemeliharaan *Software*

No	Teknik Pemeliharaan	Area Implementasi	S	Kekurangan	Riwayat Penelitian
1.	*Corrective Maintenance*	- Rekonstruksi *source code*	[5], [8]	- Tidak ada cara pasti untuk membenahi kegagalan code. - Tiap kasus penyelesaian bisa berbeda-beda.	-Lientz (1978) -Canfora (2001) -Kostas (2011) -Swebok (2014) -Levin (2019)
		- *Debugging* rutin	[4]		
		- Identifikasi kerusakan, *software (Event driven, Goal driven, Pattern driven*)	[5]		
		- Tujuan : membenahi kesalahan yang ditemukan dalam sistem.	[12]		
2.	*Perfective Maintenance*	- *Adding new functionality* - Penggabungan >1 *software* (*Software Merging*) - Transformasi *source code* (*Software Refactoring*)	[9]	- Waktu untuk proses dibutuhkan relatif lama, - Skill yang dimiliki teknisi harus sangat memadai.	-Lientz (1978) -Canfora (2001) -Kostas (2011) -Swebok (2014) -Levin (2019)
		- Tujuan : *improvement system, improvement performance*	[12], [16]		

3.	*Adaptive Maintenance*	- Mensinkronisasikan *business* process - Perpindahan ke *Network-Centric Service-Oriented environment* - Perpindahan sistem ke bahasa pemrograman lain - Tujuan : membuat sistem tetap dapat digunakan meski berganti *environment*	[5] [12]	- Waktu yang dibutuhkan lama, - Skill yang dimiliki teknisi harus sangat memadai.	-Lientz (1978) -Canfora (2001) -Kostas (2011) -Swebok (2014) -Levin (2019)
4.	*Reverse Engineering*	- Identifikasi *reusable assets* dan komponen sistem - Menemukan *objects* di program prosedural, - Menemukan *architecture software*, - Memperoleh model data konseptual, - Mendeteksi duplikasi, - Memperbarui *user interface*, - Memparalelkan sekuensial program. - *Redocumentation*	[6], [12]	- Tidak *applicable* di semua tingkat bisnis karena beberapa *prototype* biayanya relatif tinggi.	-Lientz (1978) -Chikofsky & Cross II (1990) -Bowen (1993) -Stephen, Lam & Chan (1994) -Tham (2005) -Canfora (2001) - Kusumasari (2014) - Levin (2019)
5.	*Design Recovery*	- Pemulihan abstraksi desain tingkat tinggi dari *code, dokumentasi desain, personal experience* - Restrukturisasi pada tingkat architecture system	[14], [16] [14]	- Biaya untuk alat sangat mahal.	-Chikofsky & Cross II (1990) -Tham (2005)

6.	*Data Abstraction*	- Pengubah data dari satu representasi ke representasi lain memperhatikan *semantics*	[15]	- Penggunaan alat dibutuhkan *maintainers expert* (skill tinggi) - Hasil sering tidak sejalan dengan hipotesis	-K.H.Bennet (1993)
7.	*Decompilation*	- Pembongkaran file aplikasi yang memuat informasi sistem *software*, - Memodifikasi isi file aplikasi dalam sistem *software*, - File yang dimodifikasi (file *xml*, file *gambar*)	[17]	- Pendekatan ke masalah kurang langsung atau kurang eksklusif.	-Bowen (1993)
8.	*Maintenance Model*	- Memprediksi efektivitas strategi manajemen *software* terhadap *flow pemeliharaan* yang masuk	[13]	- (cukup efektif untuk berbagai jenis kasus)	-Bowen (1993) -Kostas (2011) -Kusumasari (2014)
		- Mendapatkan opsi implementasi dan kode yang cocok untuk sistem *software*	[17]		
9.	*Specification-based Approach to Maintenance*	- Memelihara spesifikasi *software* dengan mengidentifikasi *history* pengembangannya	[17]	- Kurang akurat untuk mengatasi masalah yang berhubungan dengan sistem	-Bowen (1993)
10.	*Verification and Validation*	- Pengecekan berkala terhadap fungsi program dan *behaviour program*, - Memenuhi spesifikasi *software* yang telah disepakati di awal antar *client*	[17]	- Lebih banyak digunakan untuk masalah informal saja (ex: menjamin kepuasan *client* terkait spesifikasi *software* yang terjaga)	-Bowen (1993)
11.	*Inventory Analysis*	- Merancang rencana pengembangan, - Menyiapkan dokumen dari sistem l*egacy software* yang akan di maintain	[13]	- (Cukup efektif diimplementasikan di semua jenis kasus)	-Kusumasari (2014)

12.	*Program & Data Restructuring*	- Format ulang *code*, - Mengganti nama variabel secara konsisten di seluruh program, - Meningkatkan struktur kontrol program.	[14]	- Waktu relatif lama - Biaya mahal - Harus merombak seluruh program	-Norman (1987) -Chikofsky & Cross II (1990) -Tham (2005) -Kusumasari (2014)
13.	*Forward Engineering*	- Mengembangkan *legacy system* menjadi sistem baru	[13]	- Waktu yang dibutuhkan lama - Skill harus sangat memadai	-Chikofsky & Cross II (1990) -Kusumasari (2014)
		- Serangkaian *flow maintenance* dimulai dari pemenuhan *requirements* hingga mendesain implementasinya.	[16]		
14.	*Control and Data Flow*	- Ekstraksi *code* yang berkaitan dengan tugas tertentu dari program yang besar, - Memelihara ekstraksi tersebut sehingga tugas tertentu bisa di maintain tanpa harus mengganggu *code program* yang lain.	[14]	-	-Tham (2005)
15.	Program Slicing	- Ekstraksi *code* program yang berperan pada titik tertentu dalam program menggunakan *data flow algorithm*.	[14], [18]	-	-Gallagher (1991) -Tham (2005)
16.	*Metrics*	- Penstabilan program, *module*, dan logic *software* sebelum dilakukan *pemeliharaan*.	[1]	- Terjadinya *ripple effect** karena gagal dalam penstabilan	-Norman (1978) -Pigoski & Nelson (1994)
17.	*Reengineering*	- Identifikasi *assets*, *objects*, dan *behaviour* pada sistem target (fase *reverse engineering*), - Memodifikasi bagian-	[8], [20]	- Banyak fase yang harus dikerjakan, bahkan seringkali dalam *reengineering* aplikasi, selalu ada proses menambahkan fungsionalitas baru	-Chikofsky & Cross II (1990) -Grubb (2003) -Swebok (2014)

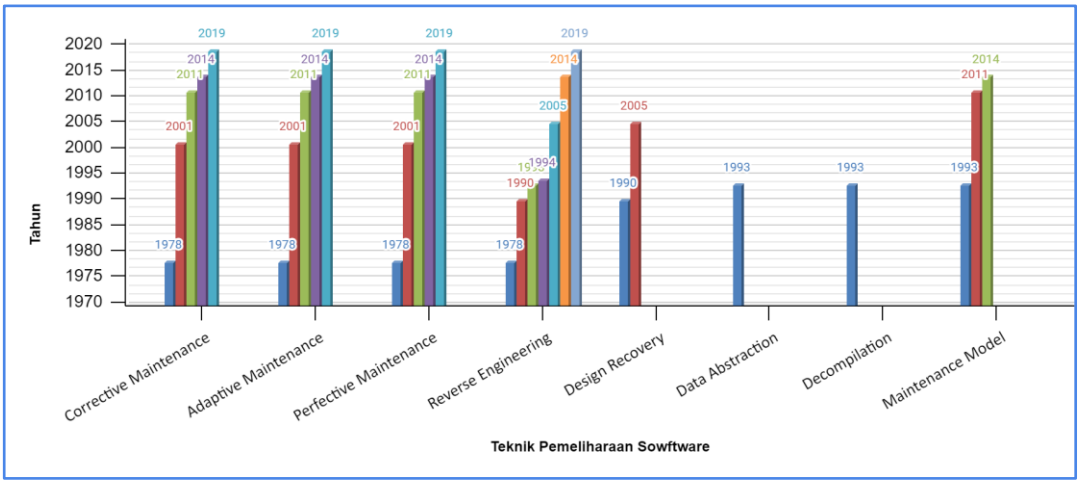
		bagian yang diinginkan ke dalam bentuk sistem baru (fase *forward engineering*)			
18.	***Search-Based Refactoring:*** *Hill Climbing Technique*	- Pencarian kerusakan lokal pada sistem software menggunakan algoritma *hill climbing*** .	[21]	- Ada kemungkinan hasil pencarian tidak menemukan solusi terbaik.	-Mohan and Greer (2018)
19.	***Search-Based Refactoring:*** *Genetic Algorithms Technique*	- Pencarian kerusakan lokal dengan menerapkan kelas algoritma evolusioner atau *Evaluation Algorithm*, yaitu peniruan proses yang digunakan di tempat lain dalam sains.	[21]	- Proses membutuhkan waktu lama - Solusi terbaik dari hasil algoritma, tergantung dari interpretasi *user* (kurang pasti)	-Mohan and Greer (2018)
20.	***Search-Based Refactoring:*** *Multi-objective evolutionary algorithms*	- Pencarian kerusakan lokal dengan menerapkan algoritma evolusioner multi objektif (MOEAs).	[21]	- Proses membutuhkan waktu lama - Solusi terbaik dari hasil algoritma, tergantung dari interpretasi *user* (kurang pasti)	-Mohan and Greer (2018)
21.	*Data Mining*	- Mengekstrak informasi dari data sistem *software* untuk meningkatkan proses pengembangan dan kualitas *software*,	[3]	-	-Shatnawi (2020)
22.	*Association Rules Algorithm*	- Menemukan pola tersembunyi dari transaksi market data.	[3]	- (cukup efektif untuk berbagai jenis kasus)	-Shatnawi (2020)
23.	*Classification*	- Menemukan kategori suatu instance data baru dengan *decision tree, naive Bayes,*	[3]	- (cukup efektif untuk berbagai jenis kasus)	-Shatnawi (2020)
24.	*Associative Classification*	- Mengekstrak pengklasifikasi kompetitif dibandingkan dengan pengklasifikasi tradisional. Contoh *CBA Algorithm,*	[3]	- (cukup efektif untuk berbagai jenis kasus)	-Shatnawi (2020)

	CMAR, CPAR, CBA2 Algorithm

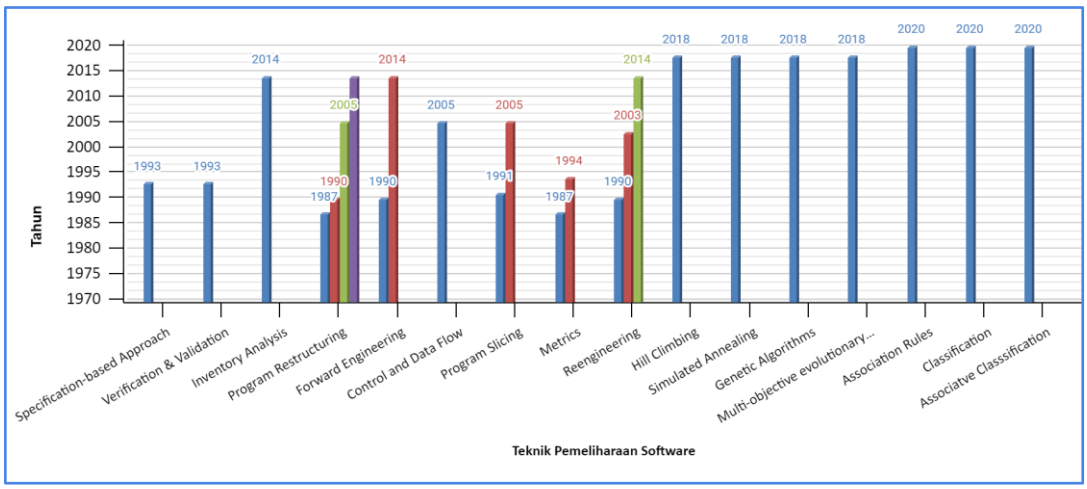
Berdasarkan tabel 6, ditemukan hasil *survey* berupa 24 teknik pemeliharaan *software* yang dianalisis area implementasi, kekurangan, serta riwayat penelitiannya kemudian dibandingkan satu sama lain.

Hasil Analisis Kedua (RQ3)

Data analisis yang didapat dari Tabel 6, diklasifikasikan lagi untuk menjawab *Research Question* yang ke-3 yaitu teknik mana yang paling banyak digunakan. Diagram yang dibuat memanfaatkan elemen tahun publikasi penelitian yang pernah dilakukan. Diagram rentang waktu penggunaan teknik pemeliharaan *software* dalam penelitian ditunjukkan oleh gambar 5 bagian (a) dan (b) di bawah ini.



Gambar 5.a. Diagram Rentang Waktu Penggunaan Teknik dalam Paper Penelitian



Gambar 5.b. Diagram Rentang Waktu Penggunaan Teknik dalam Paper Penelitian

Dari diagram pada Gambar 5.a dan 5.b adalah teknik pemeliharaan yang paling banyak digunakan dalam paper penelitian (kurun waktu 1978-2020) adalah teknik *reverse engineering* (8 publikasi)

4. KESIMPULAN

Dari hasil analisis terhadap 24 teknik tersebut, dapat ditarik kesimpulan bahwa teknik yang paling banyak digunakan dalam penelitian (kurun waktu 1978-2020) adalah teknik *reverse engineering, corrective maintenance*, *perfective maintenance, adaptive maintenance, program & data restructuring, maintenance model,* dan *reengineering.* Selain itu, proses analisis juga menghasilkan rekomendasi teknik pemeliharaan *software* yang cocok digunakan pada tiga kasus berikut ini (dan kasus lain yang serupa). Untuk kasus *maintenance pada software aplikasi berbasis web*, rekomendasi teknik yang efektif untuk digunakan adalah teknik pemeliharaan *inventory analysis, reverse engineering, data restructuring, forward engineering.* Untuk kasus *pembuatan framework software maintenance*, rekomendasi teknik yang efektif untuk digunakan adalah teknik pemeliharaan *program restructuring, reverse engineering, static program analysis, design recovery, system architecture, code generation.* Untuk kasus *pembangunan SSA untuk pemeliharaan pada Cryptologic System*, rekomendasi teknik yang efektif untuk digunakan adalah teknik pemeliharaan *corrective, perfective/adaptive, metrics program*

DAFTAR PUSTAKA

[1] N. F. Ieee, "Calhoun: The NPS Institutional Archive Faculty and Researcher Publications Faculty and Researcher Publications The state of software maintenance," *IEEE Trans. Softw. Eng. v. SE-13*, no. 3, pp. 303–310, 1987, [Online]. Available: http://hdl.handle.net/10945/40278.

[2] S. W. L. Yip, T. Lam, and S. K. M. Chan, "A software maintenance *survey*," *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, pp. 70–79, 1994, doi: 10.1109/APSEC.1994.465272.

[3] A. Al-Hawari, H. Najadat, and R. Shatnawi, "Classification of application reviews into software maintenance tasks using data mining techniques," *Softw. Qual. J.*, 2020, doi: 10.1007/s11219-020-09529-8.

[4] B. P. Lientz, E. B. Swanson, and G. E. Tompkins, "Characteristics of Application Software Maintenance," *Commun. ACM*, vol. 21, no. 6, pp. 466–471, 1978, doi: 10.1145/359511.359522.

[5] K. Kontogiannis, "Techniques for Software Maintenanc," 2011, doi: 10.1081/E-ESE-120044348.

[6] S. Levin and A. Yehudai, "Visually exploring software maintenance activities," *Proc. - 7th IEEE Work. Conf. Softw. Vis. Viss. 2019*, no. October, pp. 110–114, 2019, doi: 10.1109/VISSOFT.2019.00021.

[7] Bellin, David. *Software Maintenance : Case Studies.* Department of Computer Science, Willian Paterson College, Wayne NJ 07470. IEEE Digital Library. 1983.

[8] I. C. Society, *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*. .

[9] Kitchenham, B. A. Kitchenham, B.: Guidelines for performing Systematic Literature Reviews in software engineering. EBSE Technical Report EBSE-2007-01. EBSE Technical Report. 2007.

[10] Wahono, R. S. (2007). A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks. *Journal of Software Engineering*, *1*(1), 1–16. https://doi.org/10.3923/jse.2007.1.12

[11] Kouchaksaraei, H. R., & Karl, H. (2019). Service function chaining across openstack and kubernetes domains. *DEBS 2019 - Proceedings of the 13th ACM International Conference on Distributed and Event-Based Systems*, 240–243. https://doi.org/10.1145/3328905.3332505

[12] G. Canfora and A. Cimitile, *Handbook of Software Engineering and Knowledge*

Engineering - Vol 1: Fundamentals, no. May 2013. 2010.
[13] T. F. Kusumasari, "Analisis Proses Maintenace Aplikasi (Kasus : Aplikasi Web Emisi Gas Rumah Kaca Pada Sektor Industri Di Kementrian Perindustrian)," *J. Rekayasa Sist. Ind.*, vol. 1, no. 1, pp. 114–120, 2014.
[14] Tham, Kelly. *Toward Automating Software Maintenance.* Department of Information Systems and Computer Science (DISCS). Link Springer. 2005.
[15] K. H. Bennett, "Understanding the process of software maintenance," *IEEE Int. Conf. Progr. Compr.*, pp. 2–5, 1993, doi: 10.1109/WPC.1993.263912.
[16] I. T. Corp and G. Rekoff, "1990 - RE and Design Recovery," no. January, pp. 13–17, 1990.
[17] J. Bowen, P. Breuer, and K. Lano, "A compendium of formal techniques for software maintenance," *Softw. Eng. J.*, vol. 8, no. 5, p. 253, 1993, doi: 10.1049/sej.1993.0031.
[18] K. B. Gallagher and J. R. Lyle, "Using Program Slicing in Software Maintenance," *IEEE Trans. Softw. Eng.*, vol. 17, no. 8, pp. 751–761, 1991, doi: 10.1109/32.83912.
[19] S. Elmidaoui, L. Cheikhi, A. Idri, and A. Abran, "Machine Learning Techniques for Software Maintainability Prediction: Accuracy Analysis," *J. Comput. Sci. Technol.*, vol. 35, no. 5, pp. 1147–1174, 2020, doi: 10.1007/s11390-020-9668-1.
[20] J. Cooling, *Software maintenance concepts and practice*, vol. 20, no. 5. 1996.
[21] M. Mohan and D. Greer, "A *survey* of search-based refactoring for software maintenance," *J. Softw. Eng. Res. Dev.*, vol. 6, no. 1, 2018, doi: 10.1186/s40411-018-0046-4.
[22] T. M. Pigoski and L. E. Nelson, "Software maintenance metrics: A case study," *Proc. - 1994 Int. Conf. Softw. Maintenance, ICSM 1994*, pp. 392–401, 1994, doi: 10.1109/ICSM.1994.336755.
[23] Lientz, P. *Issues in Software Maintenance.* Computing *Survey*s, Vol. 15, No, 3. 1983