MODEL ESTIMASI HARGA PERANGKAT LUNAK MENGGUNAKAN *COSINE SIMILARITY* DAN *FUNCTION POINT*

Development of Software Cost Estimation and Resource Allocation Using Natural Language Processing, Cosine Similarity and Function Point

LUQMAN FANANI MZ
D082202001

PROGRAM STUDI MAGISTER TEKNIK INFORMATIKA
DEPARTEMEN TEKNIK IMFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
GOWA
2023

PENGAJUAN TESIS

MODEL ESTIMASI HARGA PERANGKAT LUNAK MENGGUNAKAN *COSINE SIMILARITY* DAN *FUNCTION POINT*

Tesis
Sebagai Salah Satu Syarat untuk Mencapai Gelar Magister
Program Studi Teknik Informatika

Disusun dan diajukan oleh:

LUQMAN FANANI MZ
D082202001

Kepada

FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
GOWA
2023

TESIS

MODEL ESTIMASI HARGA PERANGKAT LUNAK MENGGUNAKAN COSINE SIMILARITY DAN FUNCTION POINT

LUQMAN FANANI MZ
D082202001

Telah dipertahankan di hadapan Panitia Ujian Tesis yang dibentuk dalam rangka penyelesaian studi pada Program Magister Teknik Informatika Fakultas Teknik
Universitas Hasanuddin
Pada tanggal 06 November 2023
dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama

Dr. Eng. Zulkifli Tahir, S.T., M.Sc.
NIP. 19840403 201012 1 004

Pembimbing Pendamping

Prof. Dr. Ir. H. Ansar Suyuti, MT., IPU., ASEAN.Eng
NIP. 19671231 199202 1 001

Dekan Fakultas Teknik
Universitas Hasanuddin

Prof. Dr.Eng. Ir. Muhammad Isran Ramli, M.T. IPM., ASEAN.Eng.
NIP. 19730926 200012 1 002

Ketua Program Studi
S2 Teknik Informatika

Dr. Ir. Zahir Zainuddin, M.Sc.
NIP. 19640427 198910 1 002

PERNYATAAN KEASLIAN TESIS
DAN PELIMPAHAN HAK CIPTA

Yang bertanda tangan di bawah ini :

Nama : Luqman Fanani MZ

Nomor Mahasiswa : D082202001

Program Studi : Magister Teknik Informatika

Dengan ini menyatakan bahwa, tesis yang berjudul "Model Estimasi Harga Perangkat Lunak Menggunakan *Cosine Similarity* Dan *Function Point*" adalah benar karya saya dengan arahan dari komisi pembimbing (Dr. Eng.Zulkifli Tahir, S.T., M.Sc dan Prof.Dr.Ir. Ansar Suyuti, M.T.). Karya ilmiah ini belum diajukan dan tidak sedang diajukan dalam bentuk apapun kepada perguruan tinggi manapun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka tesis ini. Sebagian dari isi tesis ini telah dipublikasikan di Jurnal/Prosiding (*The 11th Institute of Electrical and Electronics Engineers* (IEEE) *International Internation Conference on Digital Applications, Transformation & Economy* (ICDATE) tahun 2023) sebagai artikel dengan judul "*Development of Software Cost Estimation and Resource Allocation Using Natural Language Processing, Cosine Similarity and Function Point*".

Dengan ini saya melimpahkan hak cipta dari karya tulis saya berupa tesis ini kepada Universitas Hasanuddin.

Gowa, 06 Desember 2023

Luqman Fanani MZ

KATA PENGANTAR

Alhamdulillahi rabbil'alamin, segala puji bagi Allah Subhanahu Wa Ta'ala Yang Maha Sempurna, yang telah memberikan rahmat, hidayah dan pertolongan-Nya sehingga penulis dapat menyelesaikan tesis dengan judul "**Model Estimasi Harga Perangkat Lunak Menggunakan *Cosine Similarity* Dan *Function Point***". Tak lupa pula shalawat dan salam kepada Nabi Muhammad SAW yang telah menyinari dunia ini dengan keindahan ilmu dan akhlak yang diajarkan kepada seluruh umatnya.

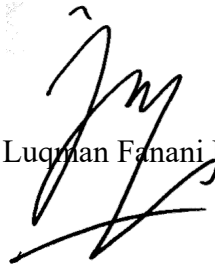
Tesis ini disusun untuk memenuhi persyaratan untuk memperoleh gelar Magister Teknik (M.Kom.) pada Program Pascasarjana Departemen Teknik Informatika. Tentunya penyelesaian tesis ini tidak terlepas dari dukungan dan bantuan dari semua pihak. Untuk itu, dengan penuh kerendahan hati penulis menyampaikan terima kasih setulus-tulusnya dan setinggi-tingginya kepada:

1. Bapak Dr. Eng.Zulkifli Tahir, S.T., M.Sc. sebagai pembimbing utama dan bapak Prof.Dr.Ir. Ansar Suyuti, M.T. selaku dosen pembimbing pendamping yang telah meluangkan waktunya kepada penulis untuk membimbing, memberikan masukan, memotivasi hingga tahap penyelesaian tesis ini.
   Bapak Prof. Dr. Ir. INDRABAYU, ST. MT. M.Bus, Sys. IPM. ASEAN. Eng dan Dr. Amil Ahmad Ilham ST. MT dan Dr. Eng. Ir Muh. Niswar, ST. M.IT. selaku penguji yang memberikan masukan dan saran yang membangun selama proses penelitian berlangsung.
2. Bapak dan Ibu dosen serta Staf Program Studi Program Studi S2 Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas bimbingan, bantuan dan arahannya selama menempuh perkuliahan.
3. Bapak Dr. Ir. Zahir Zainuddin, M.Sc. selaku Ketua Program Studi S2 Departemen Teknik Informatika, yang telah banyak mendukung dan membantu selama penulis menempuh pendidikan pascasarjana di Universitas Hasanuddin.
4. Rektor Universitas Hasanuddin dan Dekan Fakultas Teknik Universitas Hasanuddin yang telah memfasilitasi saya menempuh program Program Studi Magister.

5. Ayahanda penulis H. Moch Zachran, S.E. dan ibunda tercinta Hj. Yustina Nur. yang telah memberikan dukungan materil, doa dan motivasi yang kuat kepada penulis, sehingga penulis dapat menyelesaikan penelitian ini.
6. Istri penulis Mahrinda Pontoh, S.SI. dan anak - anak tercinta Moch. Salman Alfarisi Fanani, Silvia Alfathunisa Fanani. yang telah memberikan dukungan motivasi yang kuat kepada penulis, sehingga penulis dapat menyelesaikan penelitian ini.
7. Rekan-rekan Lab *Computer Based System* Departemen Teknik Informatika yang selalu saling mendukung dalam suka maupun duka dalam proses penyelesaian tesis ini.

Penulis menyadari bahwa tesis masih jauh dari kata sempurna dan di dalam penyelesaiannya masih menemui kesulitan dan hambatan, sehingga penulis tetap mengharapkan saran dan kritik untuk pengembangan lebih lanjut, agar dapat memberikan manfaat yang banyak bagi semua pembaca.

Gowa, 07 desember 2023



Luqman Fanani MZ

ABSTRAK

LUQMAN FANANI MZ. *Pengembangan Model Estimasi Harga Perangkat Lunak Dengan Menggunakan Metode NLP, Cosine Similarity, Dan Function Point Berdasarkan Dokumen Spesifikasi Kebutuhan*. (dibimbing oleh Zulkifli Tahir, dan Ansar Suyuti).

Estimasi biaya adalah langkah pertama pengembangan perangkat lunak yang membantu menghitung biaya dan sumber daya yang diperlukan. Proses penganggaran melibatkan analisis proyek, serta faktor-faktor seperti tidak adanya perhitungan harga yang dijadikan acuan dasar. kebanyakan masih menggunakan historical pekerjaan sebelumnya, dan alokasi tenaga ahli tidak mempunyai dasar perhitungan yang tepat untuk penetapan tenaga ahli pada suatu pekerjaan, yang berdampak pada waktu penyelesaian dan kerugian finansial akibat kesalahan perhitungan. Penelitian ini menggunakan kombinasi metode seperti text summarization word2vec untuk analisis dan pembobotan kalimat, ekstraksi katalog untuk mengidentifikasi seluruh berkas SRS yang terdeteksi sebagai fitur sistem, termasuk fitur yang memiliki sifat ambiguitas , cosine similarity untuk menentukan kedekatan bobot nilai antar kalimat yang diujikan dan metode function point sebagai penghitung hasil pengolahan nilai yang dihasilkan dari cosine similarity sehingga menghasilkan model baru dalam perhitungan dan menegaskan bahwa srs layak di diterapkan sebagai variabel perhitungan yang didasari perincian fungsionalitasnya.Hasil penelitian ini menerapkan teknik pemodelan baru yang menghasilkan sistem referensi harga dasar , penentuan jumlah tenaga ahli dalam penganggaran proyek perangkat lunak yang akurat dan efisien. Dengan demikian dapat menjadi alat bantuan perhitungan penganggaran proyek perangkat lunak di masa mendatang agar penganggaran tidak terlalu rendah atau terlalu tinggi, serta menentukan jumlah tenaga ahli yang tepat.

Kata Kunci: *Natural Language Processing, Software Cost Estimation, Software Requirement Specification (SRS), Text Summarization Word2vec, Cosine Similarity, Function Point.*

ABSTRACT

LUQMAN FANANI MZ. *Development Of Software Price Estimation Model By Using NLP, Cosine Similarity, And Function Point Methods Based On Requirement Specification Documents*. (Supervised by Zulkifli Tahir, and Ansar Suyuti).

Cost estimation is the first step of software development that calculate costs and resources required. The budgeting process involves project analysis and factors such as absence of price calculations used as a basic reference. Major rely on prior works, and allocating experts needs a proper calculation basis for assigning experts to job, which impacts completion time and financial losses due to miscalculations. This research uses combination of methods such as text summarization word2vec for sentence analysis and weighting, catalog extraction to identify all SRS files detected as system features, including features had ambiguity, and cosine similarity to determine closeness of weighted values between sentences tested and function point method as counter to processing results of values generated from cosine similarity to produce new model in calculation and confirm that SRS is feasible to be applied as a calculation variable based on its functionality details. The results of this research apply new modeling techniques to produce basic price reference system, determining number of experts in software project budgeting that is accurate and efficient. Thus, it can be a tool to calculate software project budgeting in the future so that budgeting is too low or high and determine right number of experts.

Keywords: Natural Language Processing, Software Cost Estimation, Software Requirement Specification (SRS), Text Summarization Word2vec, Cosine Similarity, Function Point

DAFTAR ISI

Halaman

HALAMAN JUDUL ..... **ii**

LEMBAR PENGESAHAAN ..... **ii**

PERNYATAAN KEASLIAN TESIS ..... **iv**

PELIMPAHAN HAK CIPTA ..... **iv**

KATA PENGANTAR ..... **v**

ABSTRAK ..... **vii**

ABSTRACT ..... **viii**

DAFTAR ISI ..... **ix**

DAFTAR TABEL ..... **xii**

DAFTAR GAMBAR ..... **xiii**

DAFTAR LAMPIRAN ..... **xv**

BAB I PENDAHULUAN **1**

1.1 Latar Belakang 1

1.2 Rumusan Masalah 2

1.3 Tujuan Penelitian 2

1.4 Manfaat Penelitian 2

1.5 Batasan Masalah 3

1.6 Sistematika Penulisan 3

BAB II TINJAUAN PUSTAKA **5**

2.1 Landasan Teori 5

2.1.1 Tinjauan jurnal penelitian 5

2.1.2 *Software reuse* 5

2.1.3 Kebutuhan perangkat lunak *(software requirements)* 8

2.1.4 Dokumen spesifikasi kebutuhan perangkat lunak 9

2.1.5 Pemrosesan bahasa natural (NLP) 10

2.1.6 Word2vec 11
2.1.7 Algoritma *cosine similarity* 13
2.1.8 Korpus 14
2.1.9 Function points 15
2.1.10 Cocomo 20
2.1.11 Unadjusted function point 22
2.1.12 Pengembangan perangkat lunak berorientasi objek 24
2.1.13 Dokumen spesifikasi kebutuhan 25
2.2 State of The Art 27
2.3 Metode Yang Diusulkan 37
2.4 Target Hasil Penelitian 40
2.5 Kerangka Pikir 41
2.6 Penelitian Terkait 45
BAB III METODE PENELITIAN 49
3.1 Rancangan Penelitian 49
3.2 Waktu Dan Lokasi Penelitian 49
3.3 Perangkat Penelitian 50
3.4 Tahapan Penelitian 51
3.5 Rancangan Sistem 53
3.6 Teknik Pengambilan Data 55
3.7 Sampel Data 55
3.8 Teknik Evaluasi Kinerja Sistem 56
BAB IV HASIL DAN PEMBAHASAN 57
4.1 Dataset Dan Data uji 57
4.2 Ekstraksi Fitur 72
4.3 Hasil Pengujian Sistem Estimasi Perangkat Lunak 75

4.4 Teknik Evaluasi Kinerja Sistem..... 87

BAB V PENUTUP..... 93

5.1 Kesimpulan 93

5.2 Saran 94

DAFTAR PUSTAKA..... 95

DAFTAR LAMPIRAN..... 97

DAFTAR TABEL

Nomor Halaman

Tabel 1. Daftar isi software requirements specification (SRS) 9
Tabel 2. Adjusted processing complexity 16
Tabel 3. *Value adjustment factor*..... 18
Tabel 4. Unadjusted function point 23
Tabel 5. State of the art penelitian..... 27
Tabel 6. Pengembangan topik penelitian..... 37
Tabel 7. Waktu dan lokasi penelitian 49
Tabel 8. IEEE SRS template 58
Tabel 9. Hasil ekstraksi fitur 72
Tabel 10. Value adjustment factor determination 81
Tabel 11. Komparasi persentase margin pada 2 metode 91

DAFTAR GAMBAR

Nomor Halaman

Gambar 1. Clone - and – own 7
Gambar 2. Arsitektur sistem pipeline untuk ekstraksi informasi berbasis dokumen teks 10
Gambar 3. Kerangka pikir penelitian 1 42
Gambar 4. Kerangka pikir 2 43
Gambar 5. Kerangka pikir penelitian 3 44
Gambar 6. Tahapan penelitian 52
Gambar 7. Blok diagram 53
Gambar 8. Detail alur sistem 54
Gambar 9. Judul cover 66
Gambar 10. Halaman penjelasan pendahuluan 67
Gambar 11. Halaman detail keseluruhan 68
Gambar 12. Halaman persyaratan antarmuka eksternal 69
Gambar 13. Fitur sistem 70
Gambar 14. Persyaratan 71
Gambar 15. Ekstrak file menjadi satuan data array 75
Gambar 16. Fungsi modifikasi kata ke kalimat 76
Gambar 17. Flowchart pemodelan estimasi pada perhitungan bobot kalimat 77
Gambar 18. Sentences weight 78
Gambar 19. Call data corpus 78
Gambar 20. Iteration data corpus dengan data uji 80
Gambar 21. Complexity 80
Gambar 22. LOC (line of code) 81
Gambar 23. Referensi nilai LOC. 82
Gambar 24. Metode COCOMO 83
Gambar 25. Implementasi Man-hours 84
Gambar 26. Distribution effort 85
Gambar 27. Hasil akhir perhitungan 86

Gambar 28. Visualisasi penentuan alokasi tenaga ahli perbandingan hasil estimasi yang dilakukan dengan menggunakan metode function point dan nilai aktual pekerjaan yang telah terimplementasi. 88
Gambar 29. Durasi visualisasi penentuan durasi pengerjaan (bulan) perbandingan hasil estimasi yang dilakukan dengan menggunakan metode function point dan nilai aktual pekerjaan yang terimplementasi. 89
Gambar 30. Grafik nilai aktual dengan metode yang diusulkan 90

DAFTAR LAMPIRAN

Nomor Halaman

Lampiran 1. Dataset..... 97
Lampiran 2. Publikasi Artikel Ilmiah..... 108

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam beberapa tahun terakhir, ukuran dan fungsionalitas perangkat lunak telah mengalami pertumbuhan besar-besaran. Bersamaan dengan ini, biaya estimasi memainkan peran utama dalam keseluruhan siklus pengembangan perangkat lunak, dan karenanya, ini adalah tugas penting yang harus dilakukan sebelum siklus pengembangan dimulai dan dapat berjalan sepanjang siklus perangkat lunak. Ini membantu dalam membuat estimasi akurat untuk setiap proyek sehingga biaya dan deliver required yang sesuai dengan apa yang dikonsepkan di awal. Ini juga membantu dalam mengidentifikasi effort hingga nominal yang diperlukan untuk mengembangkan aplikasi (M Syauqi Haris, 2020).

Siklus pengembangan perangkat lunak berbasis proyek secara keseluruhan dipengaruhi oleh prediksi akurat dari biaya pengembangan perangkat lunak. Banyak model untuk estimasi biaya perangkat. dengan melakukan perhitungan berdasarkan software requirement specification akan dilakukan ekstraksi bobot untuk mengestimasi nilai suatu perangkat lunak.

Diharapkan terciptanya sebuah anggaran yang akurat yang didasarkan pada prinsip ilmiah dan pemodelan harga yang dapat digunakan sebagai dasar estimasi yang tepat. Hal ini dilakukan dengan merujuk pada dokumen SRS (Software Requirement Specification) agar dapat menghitung secara nominal suatu sistem dengan cara mengambil nilai bobot dari setiap item yang tercantum dalam SRS. Bobot-bobot ini kemudian akan dihitung menggunakan metode Function Point. Metode perhitungan ini diharapkan dapat memberikan pendekatan yang lebih sistematis dalam menentukan besaran harga. Saat ini, tidak ada metode lain selain metode perkiraan kasar (guesstimate) yang digunakan untuk menghitung nilai suatu aplikasi. Penting juga untuk dicatat bahwa SRS merupakan dokumen internal yang biasanya tidak dapat diakses oleh konsumen, karena dokumen ini lebih ditujukan untuk digunakan oleh staf teknis.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang maka rumusan masalah pada penelitian ini adalah:

1. Bagaimana cara untuk membuktikan secara matematis anggaran yang telah selesai nilai yang ditetapkan itu benar?
2. Bagaimana mengidentifikasi suatu harga perangkat lunak yang akurat dengan hanya melihat dokumen spesifikasi kebutuhan?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah maka tujuan yang akan dicapai dari penelitian ini adalah:

1. Merancang dan mengimplementasikan model analisis sistem estimasi harga
2. Mengembangkan model estimasi harga yang dapat menentukan pemodelan dalam perencanaan pada saat dokumen spesifikasi kebutuhan telah lengkap.
3. merancang pemodelan menggunakan metode Function Point dan Distributed Effort, dengan fokus pada pengembangan sumber referensi awal dan penyesuaian satuan ukuran yang berbasis jam kerja (man-hours).

1.4 Manfaat Penelitian

Adapun manfaat yang dapat diperoleh dari penelitian ini adalah:

1. Membantu para lead engineer atau kepala proyek melakukan perhitungan berdasarkan dokumen spesifikasi kebutuhan.
2. Memberikan sumbangsih penelitian kepada para akademik atau praktisi mengenai Pemodelan estimasi perangkat lunak di Fakultas Teknik Universitas Hasanuddin.

1.5 Batasan Masalah

Adapun batasan masalah dalam penelitian ini antara lain:

1. Perhitungan estimasi harga diolah berdasarkan SRS yang lengkap.

1.6 Sistematika Penulisan

Adapun sistematika penulisan pada penelitian ini adalah:

Bab I Pendahuluan

Pada bab I ini berisi penjelasan tentang latar belakang yang menjabarkan alasan dilakukannya penelitian terkait estimasi harga perangkat lunak, berdasarkan peluang penelitian dan uraian penelitian awal tentang natural language processing dengan kombinasi beberapa seperti metode function point, cosine similarity, word2vec dan dokumen spesifikasi kebutuhan, terkait rumusan masalah, tujuan, manfaat, ruang lingkup serta sistematika penulisan penelitian dibahas pada bagian ini. .

Bab II Tinjauan Pustaka

Pada bab II ini berisi penjelasan tentang landasan teori yang digunakan dalam penelitian meliputi pemodelan estimasi harga, text mining, algoritma Word2vec,cosine similarity, dan beberapa landasan teori lainnya.

Diuraikan pula tentang tinjauan pustaka yang merupakan penjelasan tentang hasil-hasil penelitian sebelumnya yang berkaitan dengan penelitian yang dilakukan. Landasan teori merupakan suatu penjelasan tentang sumber acuan terbaru dari pustaka primer seperti buku, artikel, jurnal, prosiding dan tulisan asli lainnya untuk mengetahui perkembangan penelitian yang relevan dengan judul atau tema penelitian yang dilakukan dan juga sebagai arahan dalam memecahkan masalah yang diteliti.

Pada bab ini juga diuraikan tentang kerangka pemikiran yang merupakan penjelasan tentang kerangka berpikir untuk memecahkan masalah yang sedang diteliti, termasuk menguraikan objek penelitian serta state of the art dari beberapa penelitian terkait.

Bab III Metodologi Penelitian

Pada Bab III ini merupakan penjelasan tentang tahapan penelitian dimulai bagaimana metode penelitian, penentuan masalah, penentuan pendekatan komputasi, juga penjelasan bagaimana pengembangan dan penerapan sistem objek penelitian.

Selain itu, menjelaskan proses validasi hasil penerapan sistem, metode pengumpulan data, metode analisis data, penerapan pada masalah penelitian, konstruksi sistem serta pengujian sistem.

Bab IV Hasil dan Pembahasan

Pada bab IV berisi penjabaran hasil dan pembahasan penelitian berdasarkan teknik implikasi sistem yang digunakan serta penelitian yang telah dilakukan.

Pada bagian hasil ini membahas tentang data kuantitatif yang telah dikumpulkan berdasarkan ketetapan pada metodologi. Pembahasan menjabarkan proses pengolahan data serta interpretasinya, baik berupa deskriptif maupun penarikan inferensinya.

Implikasi penelitian merupakan uraian penjabaran mengenai keberlanjutan penelitian yang relevan dengan aspek sistem, maupun pengembangan pada aspek penelitian. Hasil analisis pemodelan estimasi harga perangkat lunak dan olah text mining yang telah dirancang dirangkum dalam bentuk tabel, dan gambar.

Bab V Kesimpulan dan Saran

Pada bab V berisi kesimpulan terhadap hasil yang didapatkan dalam penelitian ini, yang merujuk pada rumusan masalah dan saran pengembangan dari penelitian ini untuk menyempurnakan kekurangan-kekurangan yang belum tercapai pada penelitian ini, sehingga kedepannya penelitian yang dilakukan dapat dikembangkan dan bisa memperoleh hasil yang jauh lebih baik.

BAB II

TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Tinjauan jurnal penelitian

Tinjauan pustaka yang tertuang pada bab ini hasil dari studi pendahuluan yang telah dilaksanakan oleh penyusun, studi pendahuluan yang dilakukan adalah studi literatur dengan melaksanakan review terhadap jurnal internasional yang relevan dengan tema penelitian, mereview buku dan modul yang mendukung materi, melaksanakan browsing di internet dan juga menganalisis video yang relevan.

2.1.2 *Software reuse*

Rekayasa perangkat lunak berbasis reuse adalah strategi rekayasa perangkat lunak di mana proses pengembangan diarahkan untuk menggunakan kembali perangkat lunak yang ada.

Meskipun reuse telah lama diusulkan sebagai strategi pengembangan,namun baru sejak tahun 2000 an pengembangan dengan reuse telah menjadi norma pada sistem bisnis perangkat lunak. Hal ini dilakukan oleh industri untuk memenuhi tuntutan dalam industri perangkat lunak dengan biaya pengembangan dan pemeliharaan yang lebih rendah,pengiriman sistem yang lebih cepat, serta peningkatan kualitas perangkat lunak.

Keuntungan tersebut berlaku bagi pengembang perangkat lunak yang memanfaatkan reuse untuk menghasilkan produk perangkat lunak di proyek - proyek berikutnya, namun hal ini tidak berlaku bagi pengembang yang harus merancang dan membangun komponen perangkat lunak yang bisa digunakan kembali (reusable component). Biaya untuk menghasilkan reusable component bisa mencapai 25 sampai dengan 200 persen dari perangkat lunak yang ditargetkan itu sendiri (Pressman, 2010).

Konsep software reuse bisa diaplikasikan dalam beberapa level yaitu :

a) Level abstraksi

Pada level ini, konsep penggunaan kembali (reuse) tidak secara langsung terapkan pada produk atau bagian perangkat lunak yang telah jadi sebelumnya. namun lebih pada penggunaan kembali abstraksi pengetahuan dari perangkat lunak yang pernah dihasilkan baik dalam bentuk pola desain (design pattern) maupun pola arsitektur (architectural patterns).

b) Level objek

Pada level ini konsep penggunaan kembali secara langsung ditujukan pada library - library yang pernah dibangun sebelumnya, sehingga proses koding tidak perlu dilakukan kembali untuk library-library tertentu yang sudah ada. Untuk mengimplementasikan hal ini bisa dilakukan instansiasi object dan method yang tersimpan atau bahkan mungkin sudah disediakan oleh bahasa pemrograman atau IDE tool yang dipakai.

c) Level komponen

Komponen adalah gabungan dari beberapa objek dan kelas objek yang beroperasi bersama untuk menghasilkan suatu fungsi atau layanan tertentu. dalam penggunaan kembali komponen biasa dilakukan penambahan kode atau modifikasi sesuai kebutuhan. hal ini bisa dicontohkan seperti pengembangan aplikasi yang didasarkan atas suatu framework tertentu dengan tersedianya fungsionalitas umum yang bisa diimplementasikan untuk jenis aplikasi tersebut.

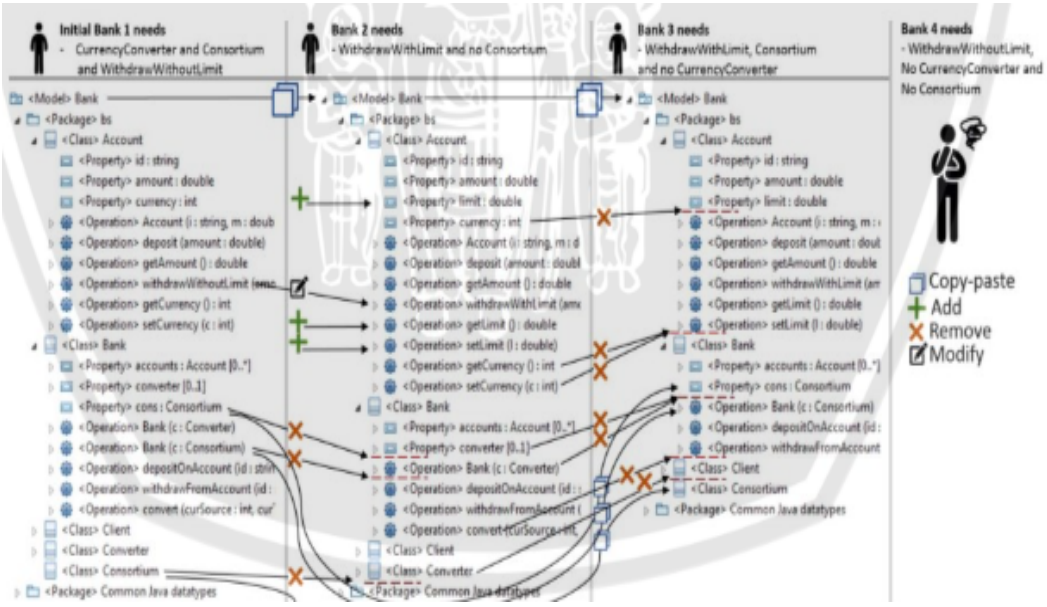
d) Level sistem

Pada level ini, penggunaan kembali sistem bersifat menyeluruh termasuk konfigurasi atas fungsionalitas sistem itu sendiri. mayoritas sistem komersial saat ini (COTS) memiliki konsep ini, dimana sistem menyediakan menu konfigurasi agar pengguna bisa secara mudah

melakukan pengaturan agar sesuai dengan kebutuhannya. software product line (SPL) adalah salah satu penggunaan kembali pada level sistem (sommerville,2015). pada SPL pengembang bisa melakukan konfigurasi untuk menghasilkan produk tertentu sesuai kebutuhan konsumen dari suatu sistem tunggal yang disebut core asset base (beuche & dalgarmo,2007). pada penelitian ini, software reuse pada level abstraksi yaitu software product line (SPL) inilah yang menjadi fokus utama.

Metode kloning dan modifikasi (clone - and - own) adalah yang paling banyak diterapkan dalam penerapan software reuse. Meskipun metode ini mampu menghemat biaya dan mempercepat proses pengembangan sistem dengan menggunakan kembali komponen yang telah dibuat sebelumnya, akan tetapi produk yang dihasilkan merupakan sistem yang terpisah antara satu dengan yang lainnya yang dihasilkan merupakan sistem yang terpisah antara satu dengan yang lainnya.

Hal inilah yang menyebabkan kesulitan pemeliharaan dan evolusi sistem di masa yang akan datang (Dubinsky et al ., 2013). Oleh karena itu, software product line dianggap sebagai solusi untuk mengatasi masalah pemeliharaan dan evolusi tersebut. Ilustrasi penggunaan metode clone - and - own dalam menghasilkan suatu produk baru dalam pengembangan perangkat lunak dapat dilihat pada Gambar 1.



Gambar 1. Clone - and – own

2.1.3 Kebutuhan perangkat lunak *(software requirements)*

Kebutuhan Perangkat Lunak (Software Requirements) adalah gambaran terhadap apa yang harus dipenuhi oleh suatu perangkat lunak. Kebutuhan tersebut mencerminkan spesifikasi kebutuhan konsumen atas suatu perangkat lunak yang memberikan layanan dengan tujuan tertentu. Adapun kemampuan yang dimiliki oleh suatu perangkat lunak tersebut haruslah memenuhi kontrak, standar, spesifikasi, atau dokumen formal lainnya (Neill & Laplante, 2003). Secara umum klasifikasi kebutuhan perangkat lunak dibagi menjadi dua yaitu functional requirements dan non-functional requirements. Functional requirements menyatakan spesifikasi layanan apa saja yang harus disediakan oleh sistem perangkat lunak, bagaimana sistem harus bereaksi atas suatu masukan tertentu dan bagaimana sistem harus bekerja dalam situasi tertentu. Sedangkan non-functional requirements adalah batasan atau tolak ukur atas layanan atau fungsionalitas sistem yang ada, sehingga sistem tersebut bisa dinilai apakah sudah bisa memenuhi kebutuhan yang ditentukan atau belum (Sommerville, 2010).

Selain klasifikasi kebutuhan secara umum di atas, Sommerville (Sommerville, 2010) juga mengkategorikan kebutuhan berdasarkan level abstraksi yang disesuaikan dengan tujuan pembacanya. Pertama yaitu User Requirements atau high-level abstract requirements yang menyatakan gambaran umum spesifikasi kebutuhan dalam bahasa natural dengan disertai gambar-gambar tentang layanan apa saja yang diharapkan dari suatu sistem perangkat lunak. Kedua yaitu System Requirements atau low-level abstract requirements yang tentu saja berisi penjelasan lebih mendetail dan aturan teknis tentang bagaimana user requirements itu berlaku.

User requirements cukup bisa dinyatakan dalam bahasa natural saja maka untuk system requirements kadang memerlukan tambahan penjelasan dalam bahasa formal agar lebih tepat dipahami dan tidak menimbulkan multi tafsir. Dalam konteks sistem komputer, spesifikasi berarti hal yang berbeda untuk orang yang berbeda. Spesifikasi bisa berupa dokumen tertulis, serangkaian model atau gambar, bahasa matematika formal, kumpulan skenario, suatu prototype, atau gabungan dari itu semua. Hal tersebut tentunya disesuaikan dengan besar kecilnya suatu proyek perangkat lunak yang dijalankan (Pressman, 2010). Oleh karena itu, diperlukan suatu standardisasi tentang bagaimana spesifikasi kebutuhan itu dituliskan.

Software Requirements Specification (SRS) adalah dokumen generik yang dijadikan acuan dalam penulisan kebutuhan suatu sistem perangkat lunak (IEEE, 1998).

2.1.4 Dokumen spesifikasi kebutuhan perangkat lunak

Dokumen spesifikasi kebutuhan perangkat lunak atau Software Requirements Specification (SRS) adalah dokumen yang dibuat dan berisi gambaran mendetail atas semua aspek dari perangkat lunak yang akan dibangun (Pressman, 2010). The U.S. Institute of Electrical and Electronic Engineer (IEEE) adalah salah satu provider standardisasi terbaik saat ini yang juga telah mengembangkan panduan struktur dokumen dalam penulisan spesifikasi kebutuhan perangkat lunak. Meskipun 20 panduan ini lebih tepat untuk sistem militer dan sistem kontrol, karena memang dikembangkan Bersama dengan U.S. Department of Defense (DoD), namun panduan tersebut bisa dijadikan sebagai dasar untuk pengembangan penulisan SRS yang tentunya disesuaikan dengan kebutuhan perusahaan atau organisasi (Sommerville, 2010) (Karl Wiegers 2013). Struktur dokumen standar dalam penulisan spesifikasi kebutuhan perangkat lunak (SRS) dapat dilihat pada Tabel 1.

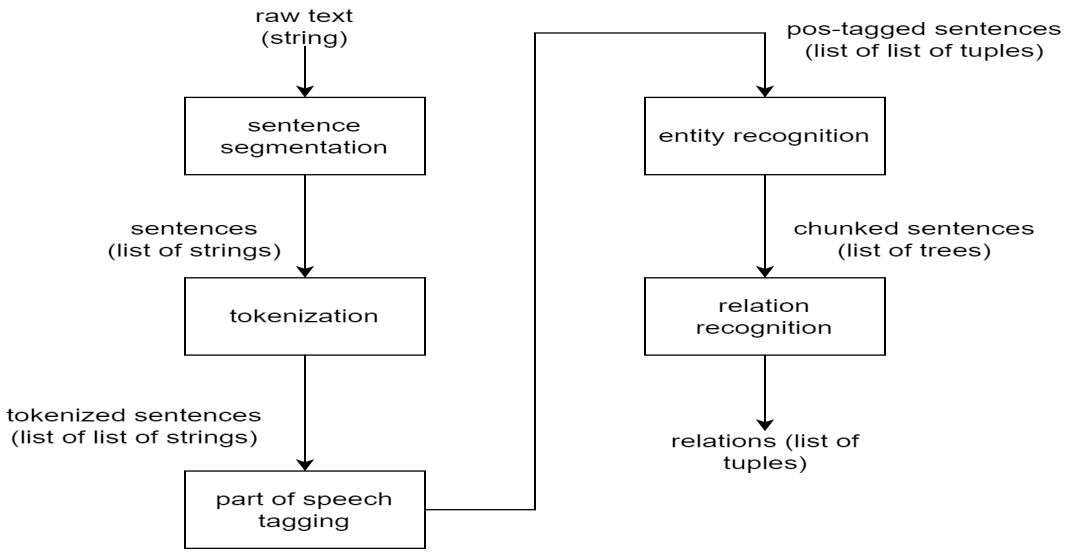
Tabel 1. Daftar isi software requirements specification (SRS)

Software requirements specification (SRS)		
1 Introduction	1.1	Purpose
	1.2	Document conventions
	1.3	Intended audience and reading suggestion
	1.4	Product scope
	1.5	References
2 Overall Description	2.1	Product perspective
	2.2	Product functions
	2.3	User classes and characteristics
	2.4	Operating environment
	2.5	Design and implementation constraints
	2.6	Assumptions and Dependencies
	2.7	User documentation
3 External interface requirements	3.1	User Interfaces
	3.2	Hardware Interfaces

Software requirements specification (SRS)		
	3.3	Software Interfaces
	3.4	Communications Interface
4 System Features	4.1	Use case diagram
5 Other Nonfunctional Requirements	5.1	Performance Requirements
	5.2	Safety Requirements
	5.3	Security Requirements
	5.4	Software Quality Attributes
	5.5	Business Rules
	5.6	User Documentation

2.1.5 Pemrosesan bahasa natural (NLP)

Bahasa natural masih menjadi pilihan utama bagi mayoritas pengembang perangkat lunak untuk mendefinisikan kebutuhan, meskipun sebenarnya telah tersedia bahasa formal beserta perangkat lunak pendukungnya yang dianggap lebih presisi dalam representasi kebutuhan dan meminimalisasi ambiguitas yang mungkin terjadi (Patterson, 2013). Oleh karena itu, pemrosesan bahasa natural (Natural Language Processing) dibutuhkan dalam setiap proses otomatisasi yang melibatkan dokumen teks seperti halnya dalam penelitian ini. Penggunaan NLP dalam suatu sistem bukan berarti kemampuan untuk memahami teks secara sepenuhnya, tetapi lebih bertujuan untuk melakukan ekstraksi konsep yang terdapat pada suatu dokumen (Lindblad et al., 2015). Contoh alur pemrosesan teks dengan NLP bisa dilihat pada Gambar 2.



Gambar 2. Arsitektur sistem pipeline untuk ekstraksi informasi berbasis dokumen teks

Dalam NLP, ambiguitas pada dokumen spesifikasi kebutuhan menjadi perhatian utama. Ambiguitas itu sendiri secara umum dibagi menjadi 2 (dua) yaitu context dependent ambiguity dan context-independent ambiguity. Ambiguitas yang context-dependent adalah ambiguitas yang muncul pada sisi rekayasa kebutuhan dan ambiguitas yang context-independent adalah ambiguitas yang muncul pada sisi bahasa atau linguistik yang digunakan (Patterson, 2013).

Sebagaimana dicontohkan dalam pernyataan kebutuhan sebagai berikut: The system shall show the weather for the next 24 hours. Arti kata the next 24 hours bisa berarti sistem menunjukkan cuaca saat ini dan terus menampilkannya selama 24 jam atau juga 23 bisa berarti sistem memperlihatkan prediksi cuaca mulai dari saat ini sampai dengan 24 jam berikutnya. Pernyataan kebutuhan tersebut adalah salah satu contoh dari ambiguitas pada sisi rekayasa kebutuhan atau context-dependent, yang hanya bisa dipahami oleh pakar yang memiliki pengetahuan terhadap domain dari permasalahan yang telah digali dan coba untuk diinterpretasikan (Kamsties et al., 2001). Dalam konteks penelitian ini, penggunaan NLP lebih ditekankan pada ekstraksi konteks suatu kalimat dalam dokumen dan suatu kata dalam kalimat.

karena fokus utamanya adalah mekanisme penggalian kalimat kebutuhan dan kata kunci yang merupakan representasi dari fitur pada suatu dokumen spesifikasi. Proses yang akan dilakukan pertama adalah teks dokumen akan diekstraksi menjadi kalimat kalimat, kemudian kalimat-kalimat tersebut akan dibagi lagi menjadi kata-kata dengan penomoran tertentu, yang disebut dengan tokenisasi. Setiap kata kemudian diberikan tanda yang menunjukkan bahwa kata tersebut adalah kata benda, kata kerja, kata sifat, atau kata depan. Pemberian tanda tersebut disebut dengan proses part of speech (POS) tagging. Kemudian setelah itu dilakukan pendefinisian relasi dari masing-masing kata yang ada di dalam suatu kalimat (word dependency parsing).

2.1.6 Word2vec

Word2vec adalah sebuah algoritma yang digunakan untuk membuat representasi vektor dari kata-kata dalam bentuk angka, yang juga disebut sebagai "word embeddings". Representasi vektor ini memungkinkan komputer untuk

memahami makna kata dan hubungannya dengan kata-kata lain dalam sebuah kalimat atau dokumen.

Algoritma word2vec menggunakan jaringan syaraf tiruan (neural network) untuk melatih model pembelajaran mesin. Dalam model ini, setiap kata dalam dataset pelatihan direpresentasikan sebagai vektor, yang kemudian diolah oleh model untuk memprediksi kata-kata yang sering muncul bersama dalam konteks tertentu.

Secara umum, word2vec tersedia dalam dua metode pembelajaran, yaitu continuous bag-of-words (CBOW) dan skip-gram. Pada metode CBOW, model word2vec mencoba memprediksi kata target dari kata-kata konteks di sekitarnya. Sementara pada metode skip-gram, model word2vec mencoba memprediksi kata-kata konteks dari sebuah kata target.

Setelah dilatih, representasi vektor kata yang dihasilkan oleh model word2vec dapat digunakan untuk berbagai tugas pemrosesan bahasa alami, seperti pemodelan bahasa, analisis sentimen, dan terjemahan mesin. Word2vec juga sangat berguna dalam memproses dataset yang memiliki banyak sekali kata-kata unik.

salah satu teknik yang digunakan untuk melakukan representasi kata dalam bentuk vektor dengan menggunakan model pembelajaran mesin (machine learning). Teknik ini memungkinkan kata-kata yang memiliki makna yang sama atau mirip, memiliki representasi yang mendekati satu sama lain di dalam ruang vektor yang dihasilkan.

Secara umum, algoritma word2vec terdiri dari dua jenis model: continuous bag-of-words (CBOW) dan skip-gram. Pada model CBOW, algoritma word2vec mencoba memprediksi sebuah kata target berdasarkan konteks yang ada di sekitarnya. Sedangkan pada model skip-gram, algoritma word2vec mencoba memprediksi konteks atau kata-kata yang ada di sekitar sebuah kata target.

Pada tahap awal, semua kata dalam korpus atau dataset pelatihan diwakili oleh sebuah one-hot encoding vektor, di mana satu nilai dalam vektor akan diisi dengan nilai 1 untuk mewakili keberadaan kata tersebut dalam dataset pelatihan, sedangkan seluruh nilai lainnya akan diisi dengan nilai 0.

Selanjutnya, algoritma word2vec menggunakan metode pembelajaran yang disebut backpropagation untuk mengoptimalkan bobot antara input dan output

layer dalam jaringan saraf tiruan. Proses pembelajaran ini berlangsung dengan cara memperbarui bobot tersebut secara iteratif pada setiap tahap pembelajaran, dengan menghitung nilai loss yang dihasilkan oleh model pada setiap iterasi.

Setelah model word2vec dilatih, vektor-vektor kata yang dihasilkan dapat digunakan untuk berbagai tugas pemrosesan bahasa alami. Salah satu contoh penerapan word2vec adalah pada tugas pengelompokkan topik (topic clustering), di mana vektor-vektor kata yang dihasilkan dapat digunakan untuk mengelompokkan dokumen-dokumen berdasarkan topik yang diwakilkan oleh kata-kata dalam dokumen tersebut. Teknik word2vec juga dapat digunakan pada tugas analisis sentimen, pemodelan bahasa alami, dan terjemahan mesin, serta aplikasi-aplikasi di bidang kecerdasan buatan lainnya

2.1.7 Algoritma *cosine similarity*

Cosine similarity berfungsi untuk membandingkan kemiripan antar dokumen, dalam hal ini yang dibandingkan adalah query dengan dokumen latih. Dalam menghitung cosine similarity, pertama yang dilakukan yaitu melakukan perkalian skalar antara query dengan dokumen kemudian dijumlahkan, setelah itu melakukan perkalian antara panjang dokumen dengan panjang query yang telah dikuadratkan, setelah itu di hitung akar pangkat dua.

Rumus dapat dilihat sebagai berikut :

$$\text{cosSim}(d_j, q_k) = \frac{\sum_{i=1}^{n}(td_{ij} \times tq_{ik})}{\sqrt{\sum_{i=1}^{n} td_{ij}^{2} \times \sum_{i=1}^{n} tq_{ik}^{2}}}$$

Keterangan:

cosSim(dj,qk)	: tingkat kesamaan dokumen dengan query tertentu
tdij	: term ke-i dalam vektor untuk dokumen ke-j
tqik	: term ke-i dalam vektor untuk query ke-k
n	: jumlah *term* yang unik dalam data set

Langkah-langkah perhitungan manual algoritma Cosine Similarity.

1. Ditentukan terlebih dahulu masing-masing query, yaitu query dari jawaban (D), query dari key jawaban (Q) dan gabungan keduanya(Queries).
2. Ketiga query tersebut dihilangkan stoplist atau simbol-simbol yang tidak mempengaruhi penilaian, seperti tanda titik, tanda koma, tanda seru, dan sebagainya
3. Ketiga query tersebut dihilangkan stopwords atau kata-kata umum yang lazim digunakan dalam suatu query, seperti "dan", "jika", "di", "namun", "tetapi", dan sebagainya.

2.1.8 Korpus

Korpus dalam bahasa Inggris corpus artinya kumpulan teks. Menurut Baker (2010:93) korpus merupakan kumpulan teks baik tulisan lisan maupun lisan yang tersimpan dalam komputer. Baker mendefinisikan korpus terdapat pada media elektronik saja. Menurut Setiawan (2017) korpus merupakan kumpulan tulisan yang ditulis oleh seseorang baik berupa hard copy dan soft copy. Korpus dalam bentuk hardcopy dapat dicontohkan seperti buku, majalah, kamus, dan koran. Contoh softcopy dapat berupa aplikasi, website, kamus online, dan lain sebagainya.

Dari pengertian tersebut dapat disimpulkan bahwa korpus merupakan kumpulan teks baik lisan maupun tulis yang ada di media cetak maupun elektronik dan dapat dijadikan sumber data. Dalam hal ini, semua jenis unit linguistik (kata, frasa, clausa, kalimat, dan wacana) sudah pasti menjadi bagian dari korpus selama terkumpul menjadi satu kesatuan bentuk. Akan tetapi belum jika tidak menjadi kesatuan (kumpulan) maka tidak dapat dikatakan menjadi korpus. Oleh karena itu, data yang disebut korpus juga identik dengan sejumlah data yang besar atau memiliki kuantitas yang mencukupi. Oleh karena itu korpus juga disebut sebagai korpora (bank bahasa).

2.1.9 Function points

Metrik Function Point dapat digunakan secara efektif untuk mengukur fungsionalitas yang dimiliki oleh sebuah sistem perangkat lunak. Dengan menggunakan data pada proyek sebelumnya, Function Point dapat digunakan untuk:

a. Memperkirakan biaya atau usaha yang diperlukan dalam tahap perancangan, coding, maupun pengujian.
b. Memperkirakan jumlah error yang ditemui pada saat pengujian
c. Memperkirakan jumlah komponen atau sumber baris pada tahap implementasi

Function Point diturunkan dari hubungan empiris antara Informasi domain perangkat lunak yang dapat diukur secara langsung dengan ukuran kualitatif kompleksitas perangkat lunak. Berikut ini adalah nilai domain Informasi yang harus didefinisikan dalam menentukan function point sebuah perangkat lunak:

a. Jumlah input eksternal (External Input – EIs). Input berasal dari luar sistem, baik dari user maupun sistem lainnya yang selanjutnya digunakan untuk mengupdate Internal Logical Files.
b. Jumlah output eksternal (External Output – EOs). Output merupakan data yang ditampilkan pada aplikasi untuk menyediakan Informasi kepada user, baik dalam bentuk laporan, tampilan di layar, pesan error, dst.
c. Jumlah inquiries eksternal (External Inquiries – EQs). Inquiries eksternal didefinisikan sebagai input online yang memicu respon dari software untuk menghasilkan output online.
d. Jumlah file logikal internal (Internal Logical Files – ILFs). File logika internal merupakan data yang dikelompokkan secara logis , disimpan secara internal dan didapat dari input eksternal.
e. Jumlah file interface eksternal (External Interface Files – EIFs). Merupakan data yang dikelompokkan secara logis namun berada diluar aplikasi yang menyediakan Informasi yang dibutuhkan aplikasi.

Penentuan kriteria kompleksitas setiap domain masih bersifat subjektif. Berikut adalah Tabel 3 contoh perhitungan count total.

Tabel 2. Adjusted processing complexity

Adjusted Processing Complexity

Information Domain Value	Count		Weighting factor: Simple	Average	Complex		
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
Count total	→						

Ket : ∑(Fi): total 14 nilai faktor penyesuaian

Prosedur penginputan nilai pada tabel Complexity dalam analisis Function Point (FP) adalah langkah-langkah yang harus diikuti untuk menentukan tingkat kompleksitas komponen perangkat lunak yang sedang dianalisis. Tabel Complexity digunakan dalam perhitungan Function Point untuk mengukur kompleksitas fungsional dari suatu perangkat lunak. Tabel Complexity biasanya memiliki beberapa tingkat kompleksitas, seperti "Low," "Average," dan "High," yang digunakan untuk menggambarkan seberapa rumit suatu komponen perangkat lunak.

Berikut adalah prosedur umum penginputan nilai pada tabel Complexity di dalam analisis Function Point:

1. Identifikasi Komponen: Tentukan komponen atau elemen perangkat lunak yang ingin Anda nilai tingkat kompleksitasnya. Komponen ini bisa berupa fungsi atau fitur tertentu dalam aplikasi, seperti tampilan layar, proses bisnis, atau modul perangkat lunak.
2. Tinjau Kriteria: Periksa kriteria yang ada dalam tabel Complexity untuk menentukan tingkat kompleksitas yang sesuai. Biasanya, kriteria-kriteria ini mencakup faktor-faktor seperti jumlah input, output, database, antarmuka pengguna, dan logika bisnis yang terlibat dalam komponen tersebut.
3. Analisis Komponen: Analisis komponen perangkat lunak dengan mempertimbangkan setiap kriteria pada tabel Complexity. Bandingkan karakteristik komponen dengan kriteria pada tabel untuk memutuskan tingkat kompleksitas yang paling sesuai.

4. Tentukan Nilai: Setelah menganalisis komponen, tentukan nilai tingkat kompleksitasnya berdasarkan kriteria yang telah Anda pertimbangkan. Nilai ini biasanya berdasarkan skala yang ada dalam tabel Complexity, seperti "Low," "Average," atau "High."

Tingkat kompleksitas yang telah ditetapkan untuk komponen tersebut. Nilai ini akan digunakan dalam perhitungan Function Point secara keseluruhan. Adapun rentang nilai sesuai prosedur yang telah ditetapkan oleh international function point users group. (Valerie Marthaler n.d.) seperti pada table di bagian 3 dimulai dari low, average, high.

Step	Action
1	Use the complexity and contribution counting rules that begin on page 6-7 to identify and count the number of RETs and DETs.
2	Rate the functional complexity using the following complexity matrix.

	1 to 19 DET	20 to 50 DET	51 or more DET
1 RET	Low	Low	Average
2 to 5 RET	Low	Average	High
6 or more RET	Average	High	High

3	Translate the ILFs and EIFs to unadjusted function points using the appropriate translation table for either ILFs or EIFs.

ILF Translation Table: Use the following table to translate the ILFs to unadjusted function points.

Functional Complexity Rating	Unadjusted Function Points
Low	7
Average	10
High	15

EIF Translation Table: Use the following table to translate the EIFs to unadjusted function points.

Functional Complexity Rating	Unadjusted Function Points
Low	5
Average	7
High	10

For example, a high complexity rating for an EIF translates to 10 unadjusted function points.

Step	Action
4	Calculate each ILF and EIF contribution to the unadjusted function point count. For example, the following table shows the calculation for one high complexity ILF, two average and one high complexity EIFs.

Function Type	Functional Complexity			Complexity Totals	Function Type Totals
ILF	0	Low	X 7 =	0	
	0	Average	X 10 =	0	
	1	High	X 15 =	15	15
EIF	0	Low	X 5 =	0	
	2	Average	X 7 =	14	
	1	High	X 10 =	10	24

Sedangkan ukuran kualitatif kompleksitas perangkat lunak ditentukan dari 14 faktor nilai penyesuaian (Value Adjustment Factor) sebagai berikut:

Tabel 3. *Value adjustment factor*

Karakteristik sistem		Keterangan
1.	Data communications	Berapa banyak fasilitas komunikasi yang digunakan untuk membantu pengiriman atau pertukaran Informasi dengan aplikasi?
2.	Distributed data processing	Bagaimana cara menangani data yang terdistribusi dan fungsi pemrosesannya?
3.	Performance	Berapa waktu tanggapan dari sistem atau keluaran yang dibutuhkan oleh user?
4.	Heavily used configuration	Seberapa berat platform hardware digunakan yang merupakan lokasi eksekusi dari aplikasi?
5.	Transaction rate	Seberapa sering transaksi dieksekusi setiap hari, minggu, bulan, dst?
6.	On-Line data entry	Berapa persen Informasi yang dimasukan secara online?

Karakteristik sistem		Keterangan
7.	End-user efficiency	Apakah aplikasi dirancang untuk efisiensi end-user?
8.	On-Line update	Berapa banyak ILF di update melalui transaksi online?
9.	Complex processing	Apakah aplikasi memiliki logika yang luas atau pemrosesan matematika?
10.	Reusability	Apakah aplikasi dikembangkan untuk memenuhi satu atau banyak kebutuhan?
11.	Installation ease	Seberapa sulit konversi dan instalasi?
12.	Operational ease	Seberapa efektif atau terotomatisasinya aplikasi pada saat start-up, back-up, dan prosedur recovery?
13.	Multiple sites	Apakah aplikasi dirancang, dikembangkan, dan didukung khusus untuk diinstal pada banyak site/perusahaan?
14.	Facilitate change	Apakah aplikasi dirancang, dikembangkan, dan didukung khusus untuk memfasilitasi perubahan?

Tabel Value Adjustment Factor (VAF) dalam analisis Function Point (FP) tidak memiliki parameter baku yang harus diisi seperti tabel Complexity atau tabel Processing Complexity. Sebaliknya, tabel VAF digunakan untuk mengakomodasi faktor-faktor yang dapat mempengaruhi kompleksitas dan kualitas perangkat lunak dalam proyek tertentu. Faktor-faktor ini adalah parameter yang perlu diidentifikasi dan dinilai oleh tim yang melakukan analisis FP.

Perhitungan akhir Function Point (FP) dilakukan dengan rumus berikut:

FP = count total x [0.65 x 0.01 $\sum$(Fi)]

- FP: Function Point yang akan dihitung
- Count total: total kelima nilai domain Informasi yang telah dilengkapi dengan nilai kompleksitas (sederhana, rata-rata, atau kompleks).

2.1.10 Cocomo

Cocomo (Constructive Cost Model) adalah model regresi berdasarkan LOC, yaitu jumlah Baris Kode . Ini adalah model estimasi biaya prosedural untuk proyek perangkat lunak dan sering digunakan sebagai proses prediksi yang andal berbagai parameter yang terkait dengan pembuatan proyek seperti ukuran, upaya, biaya, waktu, dan kualitas. Ini diusulkan oleh Barry Boehm pada tahun 1970 dan didasarkan pada studi terhadap 63 proyek, yang menjadikannya salah satu model terbaik yang terdokumentasi.

Parameter utama yang menentukan kualitas produk perangkat lunak apa pun, yang juga merupakan hasil dari Cocomo terutama adalah Upaya & Jadwal:

- Upaya: Jumlah tenaga kerja yang akan dibutuhkan untuk menyelesaikan tugas. Ini diukur dalam satuan orang-bulan.
- Jadwal: Secara sederhana berarti jumlah waktu yang diperlukan untuk penyelesaian pekerjaan, yang tentu saja sebanding dengan upaya yang dilakukan. Diukur dalam satuan waktu seperti minggu, bulan.

Model Cocomo yang berbeda telah diusulkan untuk memprediksi estimasi biaya pada tingkat yang berbeda, berdasarkan jumlah akurasi dan kebenaran yang diperlukan. Semua model ini dapat diterapkan pada berbagai proyek, yang karakteristiknya menentukan nilai konstanta untuk digunakan dalam perhitungan selanjutnya. Karakteristik ini berkaitan dengan jenis sistem yang berbeda disebutkan di bawah ini.

Definisi Boehm tentang sistem organik, semi-terpisah, dan tertanam:

1. Organik – Sebuah proyek perangkat lunak dikatakan sebagai tipe organik jika ukuran tim yang dibutuhkan cukup kecil, masalahnya dipahami dengan baik dan telah dipecahkan di masa lalu dan juga anggota tim memiliki pengalaman nominal mengenai masalah tersebut.
2. Semi-terpisah – Sebuah proyek perangkat lunak dikatakan sebagai tipe Semi-terpisah jika karakteristik vital seperti ukuran tim, pengalaman, pengetahuan tentang berbagai lingkungan pemrograman terletak di antara organik dan Tertanam. Proyek yang diklasifikasikan sebagai Semi-Terpisah relatif kurang akrab dan sulit untuk dikembangkan

dibandingkan dengan yang organik dan membutuhkan lebih banyak pengalaman serta bimbingan dan kreativitas yang lebih baik. Misalnya: Kompilator atau Sistem Tertanam yang berbeda dapat dianggap sebagai tipe Semi-Terpisah.

3. Tertanam – Sebuah proyek perangkat lunak dengan kebutuhan tingkat kompleksitas, kreativitas, dan pengalaman tertinggi termasuk dalam kategori ini. Perangkat lunak tersebut membutuhkan ukuran tim yang lebih besar daripada dua model lainnya dan juga pengembang harus cukup berpengalaman dan kreatif untuk mengembangkan model yang kompleks tersebut.

Semua jenis sistem di atas menggunakan nilai konstanta yang berbeda yang digunakan dalam Perhitungan Usaha.

Jenis Model: COCOMO terdiri dari hierarki tiga bentuk yang semakin detail dan akurat. Salah satu dari tiga bentuk dapat diadopsi sesuai dengan kebutuhan kami. Ini adalah jenis model COCOMO:

1. Model COCOMO Dasar
2. Model COCOMO Menengah
3. Model COCOMO terperinci

Tingkat pertama, Basic COCOMO dapat digunakan untuk perhitungan Biaya Perangkat Lunak yang cepat dan sedikit kasar. Keakuratannya agak terbatas karena tidak adanya pertimbangan faktor yang memadai.

COCOMO Menengah memperhitungkan Penggerak Biaya ini dan COCOMO Terperinci juga memperhitungkan pengaruh fase proyek individu, yaitu dalam hal Terperinci, memperhitungkan kedua pemicu biaya ini dan juga perhitungan dilakukan secara bertahap untuk selanjutnya menghasilkan hasil yang lebih akurat. Kedua model ini dibahas lebih lanjut di bawah ini.

Estimation of Effort: Calculations

E= a(KLOC)b

time = c (Ef fort)d

Personerquired = EfFort/time

Rumus di atas digunakan untuk estimasi biaya untuk model dasar COCOMO, dan juga digunakan pada model selanjutnya. Nilai konstanta a,b,c dan d untuk model dasar untuk berbagai kategori sistem. 3 klasifikasi project:

1. Organic
    a. Dengan pengalaman pegembang aplikasi
    b. Hanya membutuhkan tim kecil
    c. Kompleksitas pekerjaan rendah
2. Semi detached
    a. Pekerjaan dilakukan oleh gabungan senior engineering + junior engineering
    b. Pekerjaan yang dilakukan scopenya mulai dari rendah ke sedang (> dari point 1)
3. Embedded
    a. Tenaga kerja menggunakan senior engineering
    b. Membutuhkan staff divisi yang banyak
    c. Pekerjaan dengan kompleksitas tinggi

Tujuan dari model ini membantu end user / consumer membuat estimasi dari kebutuhan pembuatan sistem, divisi. Berikut adalah rumus dari perhitungan durasi dan effort sdm

E = ai(KLoC)bi(EAF).

Effort= a*(LOC)b

Duration= c*(Effort)d

Staffing = Effort/Duration

2.1.11 Unadjusted function point

Dengan cara menganalisis kebutuhan fungsional sistem yang direpresentasikan ke dalam Data Flow Diagram (DFD) untuk nantinya mempermudah analisis ke dalam function point (FP).

Tabel 4. Unadjusted function point

Komponen	Keterangan
External Input (EI)	Fungsi yang memindahkan data dari luar ke dalam aplikasi tanpa menyajikan manipulasi data.
External Output (EO)	Fungsi yang memindahkan data dari dalam ke pengguna dan menyajikan beberapa data yang telah dimanipulasi
External Inquiry (EQ)	Fungsi Yang memindahkan data dari dalam ke pengguna dan menyajikan beberapa data yang tanpa dimanipulasi.
Internal Logical File (ILF)	Logika dalam bentuk data tetap, yang dikelola oleh aplikasi melalui penggunaan masukan dari luar.
External Interface File (EIF)	Logika dalam bentuk data tetap, yang digunakan oleh aplikasi tetapi tidak berjalan dalam aplikasi tersebut

Nantinya setiap komponen Function Point diklasifikasikan tingkat kompleksitasnya. Tingkat kompleksitas menentukan kuantitas Unadjusted Function Point. Bobot kompleksitas diklasifikasikan berdasarkan DET, RET, dan FTR. Berikut ini merupakan penjabaran DET, RET, dan FTR (Longstreet, 2004).

a) (RET (Record Element Types) RET merupakan subgrup data yang dikenali oleh pengguna dalam Internal Logical File atau External Logical File. Contohnya adalah file mahasiswa yang menyimpan nomor mahasiswa, nama,udan lain seterusnya.
b) DET (Data Elements Types) DET dikenali oleh pengguna sebagai sesuatu yang unik dan tidak berulang apabila berulang maka dihitung 1 DET.
c) FTR (Files Type References) dapat berupa ILF atau EIF. Setiap ILF yang berupa EI dihitung sebagai FTR. Setiap ILF atau EIF yang direferensikan oleh EI atau EO atau EQ untuk memelihara (insert, update, delete) dianggap sebagai FTR. Untuk dapat membedakan EI atau EO atau EQ dengan EI atau EO atau EQ lainnya.

Lalu nilai UFP yang telah didapatkan diubah ke dalam Source lines of Code (SLOC). Setelah itu nilai SLOC dibagi dengan 1000 untuk mendapatkan nilai KSLOC yang bisa dimasukkan dalam persamaan usaha.

2.1.12 Pengembangan perangkat lunak berorientasi objek

Pengembangan perangkat lunak berorientasi objek, yang akan digunakan dalam pengembangan perangkat lunak dalam penelitian ini, merupakan teknik atau cara pandang terhadap sistem yang akan dikembangkan sebagai suatu kumpulan objek yang akan dikolaborasikan untuk mencapai solusi atas suatu permasalahan. Terdapat beberapa cara untuk memodelkan objek-objek yang akan dikembangkan pada suatu sistem, seperti abstraksi objek itu sendiri, kelas objek beserta propertinya, hubungan antar kelas sampai dengan abstraksi sistem secara keseluruhan.

Pengembangan perangkat lunak dengan pendekatan berorientasi objek ini dapat diterapkan melalui tahapan-tahapan yang terdiri dari analisis, perancangan, pemrograman, dan pengujian perangkat lunak. Object Oriented Analysis (OOA) merupakan metode analisis untuk menggali kebutuhan dalam sudut pandang kelas dan objek yang terdapat pada problem domain (Booch, 1980). Tahapan perancangan adalah tahapan awal yang bertujuan mendeskripsikan arsitektur perangkat lunak yang akan dibangun, model dan struktur data yang akan digunakan dalam sistem, komponen-komponen sistem dan algoritma yang akan dipakai (Sommerville, 2010).

Object Oriented Programming (OOP) merupakan metode implementasi dimana program disusun sebagai kumpulan objek yang saling berkolaborasi, adapun objek itu sendiri adalah merupakan hasil instansiasi dari kelas-kelas yang telah didefinisikan sebelumnya (Booch, 1980). Selanjutnya yaitu proses pengujian untuk memeriksa apakah program tersebut telah sesuai dengan spesifikasinya. Pengujian dapat dilakukan mulai dari bagian terkecil yaitu pengujian unit, yakni pengujian yang mengacu pada beberapa operasi/method yang terdapat pada kelas yang telah dibuat dalam proses implementasi. Kemudian proses pengujian dilanjutkan pada tahap integrasi antar 24 unit-unit sampai dengan pengujian sistem secara keseluruhan (Mili & Tchier, 2015).

Untuk mengembangkan perangkat lunak berbasis objek (object oriented), perangkat lunak terlebih dahulu dispesifikasikan dengan bahasa spesifikasi standar yaitu menggunakan Unified Modeling Language (UML).

UML merupakan sebuah bahasa pemodelan yang telah menjadi standar di berbagai industri dunia untuk merancang dan mendokumentasikan sistem perangkat lunak (Rumbaugh, Jacobson, & Booch, 2004). Pada penelitian ini, perangkat lunak yang dikembangkan sebagai proof-of-concept menggunakan pendekatan berbasis objek. Pembahasan selanjutnya akan menjelaskan beberapa diagram dan model yang digunakan dalam menspesifikasikan perangkat lunak berbasis objek.

2.1.13 Dokumen spesifikasi kebutuhan

Software Requirement Specification (SRS) atau Spesifikasi Kebutuhan Perangkat Lunak adalah dokumen formal yang berisi deskripsi tentang kebutuhan dan persyaratan untuk suatu perangkat lunak yang akan dibangun. Dokumen ini dihasilkan setelah analisis kebutuhan dilakukan dan berfungsi sebagai dasar untuk pengembangan perangkat lunak.

Dalam SRS, terdapat beberapa bagian yang biasanya harus disertakan, seperti:

1. Pendahuluan: Bagian ini menjelaskan tujuan dari dokumen SRS, deskripsi singkat dari perangkat lunak yang akan dibangun, serta informasi umum tentang proyek.
2. Deskripsi umum: Bagian ini berisi deskripsi singkat tentang perangkat lunak yang akan dibangun, termasuk fungsi utama, lingkup, dan asumsi dasar yang digunakan dalam pengembangan perangkat lunak.
3. Persyaratan fungsional: Bagian ini berisi daftar kebutuhan yang berkaitan dengan fungsionalitas perangkat lunak, termasuk fitur dan fungsi yang harus ada di dalam perangkat lunak. Persyaratan fungsional harus terkait langsung dengan tujuan dari perangkat lunak dan harus dijelaskan secara rinci dan spesifik.
4. Persyaratan non-fungsional: Bagian ini berisi daftar kebutuhan yang berkaitan dengan aspek non-fungsional perangkat lunak, seperti

kinerja, keamanan, keandalan, dan kecepatan. Persyaratan non-fungsional harus dijelaskan secara rinci dan harus dapat diukur atau diuji untuk memastikan bahwa persyaratan tersebut telah terpenuhi.

5. Kebutuhan pengguna: Bagian ini berisi daftar kebutuhan yang berkaitan dengan pengalaman pengguna dari perangkat lunak, seperti antarmuka pengguna, dokumentasi, dan kebutuhan aksesibilitas. Kebutuhan pengguna harus dijelaskan secara rinci dan harus dapat diuji untuk memastikan bahwa kebutuhan tersebut telah terpenuhi.
6. Kebutuhan lingkungan: Bagian ini berisi daftar kebutuhan yang berkaitan dengan lingkungan operasi dari perangkat lunak, seperti sistem operasi, perangkat keras, dan lingkungan jaringan. Kebutuhan lingkungan harus dijelaskan secara rinci dan harus dapat diuji untuk memastikan bahwa kebutuhan tersebut telah terpenuhi.
7. Kebutuhan bisnis: Bagian ini berisi daftar kebutuhan yang berkaitan dengan tujuan bisnis atau organisasi yang akan memanfaatkan perangkat lunak yang akan dibangun. Kebutuhan bisnis harus dijelaskan secara rinci dan harus dapat diukur atau diuji untuk memastikan bahwa kebutuhan tersebut telah terpenuhi.

2.2 State of The Art

Tabel 5. State of the art penelitian

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
1		Luqman fanani mz	Model Estimasi Harga Perangkat Lunak Menggunakan Cosine Similarity Dan Function Point	Melakukan pengembangan perhitungan harga perangkat lunak berbasis dokumen spesifikasi kebutuhan yang dimana tiap fungsionalitasnya akan dibobot dan dilakukan labeling sesuai dengan ketentuan dari teori function point dan di skoring tingkat kesulitannya yang dibagi menjadi tiga bagian seperti low 1-3, medium 4-7, high 8-10.	Menggunakan Teknik ekstraksi fitur, pembobotan dari NLP , cousine similarity untuk melihat kedekatan data uji terhadap data latih yang telah diproses dan disimpan dalam korpus.		
2	2020	M Syauqi Haris	Teknik Ekstraksi Katalog Fitur Dari Dokumen Spesifikasi Kebutuhan Sebagai Basis Pengembangan Software Product Lines	Melakukan ekstraksi fitur katalog pada SRS dengan output yang dihasilkan fiture model, dalam satu sistem mensimulasikan bisa menjadi beberapa kemungkinan bentuk.	Menggunakan teknik pemrosesan langsung menggunakan requirement boilerplate,dan POS	Bersama menggunakan SRS sebagai acuan analisis	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
3	2020	Kumar	Software Effort Estimation For COCOMO-II Projects Using Artificial Neural Network	Salah satu penyebab kegagalan untuk mengerjakan perangkat lunak tepat waktu adalah manajemen yang salah perhitungan di awal. Disebabkan requirement selalu berubah – ubah di tengah pekerjaan yang membuat *effort* yang dibutuhkan tidak bisa dihitung secara pasti. Dengan kumpulan data dari COCOMO II yang diimplementasikan dengan ANN untuk memprediksi harga.	Menggunakan COCOMO II dan artificial neural network.	Bersama menghitung Estimasi dan menetapkan sumber daya untuk semua kegiatan yang akan dikoordinasikan Implementasi ANN	=
4	2020	Azath, Mohanapriya, and Rajalakshmi	Software Effort Estimation Using Modified Fuzzy C Means Clustering And Hybrid ABC-MCS Optimization In Neural Network	Melakukan prediksi secara akurat upaya pengembangan perangkat lunak, karena hal ini penting menurut penulis. Yang akan berdampak pada buruknya fungsionalitas dan kualitas produk perangkat lunak, dan dapat merusak reputasi perusahaan penyedia itu sendiri. Dengan model expert judgement secara subjektif dianggap tidak cukup	*Modified fuzzy c means* (MFCM) untuk mengelompokkan dataset dan memodifikasi *neural network* dengan memasukkan algoritma optimasi *artificial bee colony* (ABC), *modified cuckoo search* (MCS) lalu di *hybrid* ABC-	Melakukan estimasi biaya perangkat lunak	>

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
				untuk menyelesaikan masalah ini.	MCS *algorithm*, dan hasil diperoleh tiga set model optimasi untuk proses estimasi *effort*		
5	2019	Rezende	Software Project Scheduling Problem In The Context Of Search-Based Software Engineering: A Systematic Review	Search-based optimization algorithms diterapkan pada SBSE untuk mengidentifikasi solusi paling optimal atau optimal waktu pengerjaan.	Dengan menggunakan SBSE dan fuzzy logic, melakukan tinjauan sistematis ini mengikuti pedoman yang diusulkan oleh Kitchenham yang terstruktur terbagi atas tiga fase. • Menemukan tujuan,Membuat protokol yang akan digunakan dalam penelitian • Melakukan review dan ekstraksi data • elaborasi	Menggunakan cocomo II	=
6	2019	Rizkiana Putri	Model Fuzzy Gaussian Untuk Optimasi Akurasi Estimasi Waktu	Penulis menggunakan cocomo II, juga menggunakan metode fuzzy gaussian yang digunakan dalam penelitian ini.	Menggunakan COCOMO II dan model fuzzy gaussian	Melakukan estimasi biaya dan fokus pada optimasi	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
			Proyek Perangkat Lunak	Diharapkan memberikan hasil akurasi yang lebih baik. Melakukan penurunan parameter C dan D nilainya sebanyak 0.0001 dari nilai awal karena penulis mendapatkan kesalahan akurasi sebesar dari total 83.83%			
7	2019	as'ary ramadhan	Estimasi Pada Effort Perangkat Lunak Dengan Pendekatan Feed Forward Neural Network Backpropagation (FFNN-BP)	Mendapatkan nilai estimasi biaya proyek perangkat lunak yang lebih baik dari segi akurasi dengan menerapkan algoritma genetika pada proses pelatihan feed forward neural network backpropagation (FFNN-BP) yang telah disesuaikan arsitekturnya untuk mengakomodasi COCOMO II terhadap dataset COCOMO.	Menggunakan algoritma genetika sebagai proses pelatihan FFNN-BP yang mengakomodasi formula post architecture model pada COCOMO II.	Menghitung usaha dan perhitungan post architecture model	>
8	2019	Maqdum	Implementasi Metode COCOMO II Untuk Estimasi Biaya Pengembangan	Komparasi biaya pengembangan perangkat lunak, yaitu dengan membandingkan hasil estimasi biaya yang diperoleh	Metode COCOMO II diimplementasikan kepada 2 sistem yang sudah selesai dikembangkan oleh	COCOMO II	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
			Perangkat Lunak Di CV. Profile Image Studio	menggunakan metode COCOMO II (Constructive Cost Model) dengan alokasi biaya yang dianggarkan oleh CV. Profile Image Studio. Selisih estimasi biaya untuk seluruh perangkat lunak adalah sebesar 37,24% dengan selisih waktu sebesar 52,94% dan selisih SDM sebesar 40%.	CV. Profile Image Studio.		
9	2018	Ramadhan, Sihabuddin	Estimasi Biaya Proyek Perangkat Lunak Menggunakan JST Dan Algoritma Genetika	Dengan menggunakan ANN yang mengakomodasi formula dari post architecture model (COCOMO II) dengan model yang diusulkan memberikan hasil estimasi biaya proyek perangkat lunak menjadi lebih akurat. Dalam kasus ini MMRE untuk COCOMO II adalah 73.01 FFNN-BP 39.90% dan FFNN-GA adalah 31.48%.	Algoritma genetika ANN, FFNN - BP dan GA	Menggunakan COCOMO II dan ANN	<
10	2018	Arora and Mishra	Software Cost Estimation Using Artificial Neural Network	Penggambaran prosedur yang digunakan untuk menghitung effort kerja dengan model cocomo dengan sampel 63	COCOMO dan neural network multi feed forward, yang bisa memperbaiki	Implementasi dengan Neural network	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
				perusahaan , untuk menentukan SDM engineer sesuai yang dihitung menggunakan MRE	keakuratan nilai MRE dalam perhitungan effort SDM software engineer.		
11	2017	Bilgaiyan	A Systematic Review On Software Cost Estimation In Agile Software Development	Menyajikan survei estimasi biaya yang sistematis dalam agile software development (ASD), yang akan berguna bagi pengguna untuk memahami tren terkini dalam estimasi biaya di agile software development ASD. Penelitian menunjukkan bahwa biaya yang dikeluarkan di sekitar 60%proyek berjalan di luar perkiraan awal mereka dan 15%proyek gagal karena estimasi yang sangat salah. Oleh karena itu, itu menjadi penting untuk melakukan analisis yang tepat dan akurat dari biaya yang diharapkan dan rencana yang sesuai.	Menggunakan agile software development.	Penggunaan model agile scrum bagian planning poker, menentukan tingkat kesulitan suatu pekerjaan atau module yang akan dikerjakan.	=
12	2017	Venkataiah	Application Of Ant Colony Optimization	Melakukan optimasi, penulis mengusulkan teknik ant colony untuk melakukan prediksi,	Menggunakan Metode cocomo 81 dengan teknik penyelesaian	Menghitung effort kerja Software	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
			Techniques To Predict Software Cost Estimation	menggunakan 3 dataset. Melakukan validasi silang 10 kali lipat pada perbandingan (ISBSG) dan dilakukan lagi validasi 3 kali lipat yang dilakukan di IBM data processing	ant colony optimize	engineer	
13	2018	Pospieszny, Czarnacka-Chrobot, and Kobylinski	An Effective Approach For Software Project Effort And Duration Estimation With Machine Learning Algorithms	Tujuan penelitian mempersempit gap antara penelitian terkini dalam implementasi machine learning (support vector machine, neural networks, generalized linear model). Untuk mendapatkan alat pendukung pengambil keputusan untuk perusahaan yang sedang mengembangkan atau baru mengimplementasikan sistem perangkat lunak	machine learning (SVM, ANN)	menghitung estimasi perangkat lunak	<
14	2017	Osmanbegović, Suljić, and Agić	A Review Of Estimation Of Software Products Development Costs	Penulis membuat atas dasar ketidak mampuan staff dalam penilaian perencanaan proyek perangkat lunak, ini akan berdampak pada kualitas perangkat lunak dan	review pemodelan software cost estimation COCOMO II, wideband delphi.	menggunakan COCOMO II	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
				mengurangi keuntungan perusahaan. Penulis membuat kerangka kerja software cost estimation dengan melakukan review seluruh model pada sce.			
15	2011	Sarwosri	Ekstraksi Metadata Dokumen Software Requirement Specification (Srs).	SRS diberi tag menggunakan algoritma part-of-speech tagging. Terakhir dokumen tersebut diberi bobot menggunakan algoritma TF-IDF untuk mengetahui kepentingan kalimat tersebut. Setelah semua proses selesai dan metadata telah didapatkan, sistem akan membentuk RDF dari RDF skema yang telah dibuat secara manual. Kemudian sistem akan memasukkan metadata ke dalamnya	Metode Part-of-Speech, TF-IDF, RDF	Bersama menggunakan TF-IDF	<
16	2020	Jayasiriwardene and Ganegoda	Keyword Extraction From Tweets Using NLP Tools For Collecting Relevant News.	Postingan di twitter Dalam satu thread yang hanya berupa 140 karakter terdapat kalimat yg bersifat ambiguitas seperti kalimat yang disingkat – singkat yang tidak dimengerti	Naive bayes, part-of-speech,tf-idf	Bersama menggunakan NLP	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
				oleh mesin untuk melakukan pencarian kata dasar dan bobot suatu kepentingan kata tersebut			
17	2015	Alodadi and Janeja	Similarity In Patient Support Forums: Using TF-IDF And Cosine Similarity Metrics	Melakukan pengolahan data percakapan pada forum, yang dimana memproses kedekatan dari suatu pertanyaan baru dengan pertanyaan lama, agar tidak ada thread pada postingan forum berbentuk duplikat referensi ini penulis gunakan untuk melihat cara menghitung suatu kedekatan kalimat	TF-IDF Cosine Similarity Metrics	Bersama menggunakan cosine Similarity dan pembobotan suatu kata / kalimat.	=
18	2022	Delong Han	Research On Structured Extraction Method For Function Points Based On Event Extraction	Melakukan ekstraksi fitur dokumen penawaran untuk menyelesaikan permasalahan efisiensi biaya yang biasanya digunakan secara tradisional atau guesstimate yang dihitung berdasarkan tenaga ahli, pada penelitian ini mengusulkan ekstraksi fitur berbasis function point dengan melakukan pengujian terhadap 10 kasus pada industri	Function point method	Bersama menggunakan function point untuk melakukan perhitungan	<

No.	Tahun / volume	Penulis	Judul	Deskripsi penelitian	Metode penyelesaian	Persamaan dengan usulan penelitian	Korelasi <.=,>
				mendapatkan hasil yang tervalidasi 70% meningkatkan efisiensi pada implementasi analisis			
19	2013	Ren and Yun	Research Of Software Size Estimation Method	Komparasi Estimasi ukuran perangkat lunak meletakkan dasar data untuk perencanaan proyek. Di antara semua metode estimasi perangkat lunak,metode Function point, COCOMO II, Pert, Delphi	Function point, cocomo untuk estimasi effort.	Function point	
20	2012	Bhatnagar	Comparing Soft Computing Techniques For Early Stage Software Development Effort Estimations	Melakukan penelitian perkiraan ukuran perangkat lunak, biaya, dan effort yang dibutuhkan untuk meminimalisir keterlambatan agar tidak mengalami kerugian bagi perusahaan.	Computational intelligence, Artificial neural network, jaringan saraf tiruan	Bersama menghitung ukuran perangkat lunak.	>

.

2.3 Metode Yang Diusulkan

Berdasarkan tabel State Of The Art, metode yang diusulkan adalah menggunakan Metode NLP dan function point merupakan, klasifikasi seluruh dokumen spesifikasi kebutuhan dan dilakukan labelling dan pembobotan di tiap section. Bagaimana hasil yang dikeluarkan dari beberapa parameter model software cost estimation yang diharapkan dapat mengeluarkan nilai anggaran suatu perangkat lunak tepat tidak over prices ataupun low prices.

Tingkat Keaslian (Level Orisinalitas) Topik Penelitian yang Diusulkan Pengembangan topik penelitian yang diusulkan dapat dilihat pada Tabel 7 adalah sebagai berikut:

Tabel 6. Pengembangan topik penelitian

No.Ref	Metode Penyelesaian																
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	✓														✓	✓	✓
2		✓															
3						✓	✓										
4						✓		✓									
5						✓	✓										
6						✓		✓									
7						✓	✓				✓						
8						✓											
9						✓	✓										
10						✓	✓										
11												✓					

No.Ref	Metode Penyelesaian																
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
12							✓						✓				
13														✓			
14							✓										
15	✓		✓														
16	✓	✓	✓														
17			✓														
18															✓		
19								✓									
20							✓										

Keterangan:

Nomor Referensi:

1. Model estimasi harga perangkat lunak menggunakan cosine similarity dan function point.(Topik yang diusulkan)
2. Teknik ekstraksi katalog fitur dari dokumen spesifikasi kebutuhan sebagai basis pengembangan software product lines
3. Software Effort Estimation for COCOMO-II Projects Using Artificial Neural Network
4. Software Effort Estimation Using Modified Fuzzy C Means Clustering and Hybrid ABC-MCS Optimization in Neural Network
5. Software Project Scheduling Problem in the Context of Search-Based Software Engineering: A Systematic Review
6. Model Fuzzy Gaussian Untuk Optimasi Akurasi Estimasi Waktu Proyek Perangkat Lunak

7. Estimasi Pada Effort Perangkat Lunak dengan Pendekatan Feed Forward Neural Network Backpropagation (FFNN-BP)
8. Implementasi Metode COCOMO II untuk Estimasi Biaya Pengembangan Perangkat Lunak di CV. Profile Image Studio
9. Estimasi Biaya Proyek Perangkat Lunak Menggunakan JST dan Algoritma Genetika
10. Software Cost Estimation Using Artificial Neural Network
11. A Systematic Review on Software Cost Estimation in Agile Software Development
12. Application of Ant Colony Optimization Techniques to Predict Software Cost Estimation
13. An effective approach for software project effort and duration estimation with machine learning algorithms
14. a review of estimation of software products development costs
15. Ekstraksi metadata dokumen software requirement specification (srs).
16. Keyword extraction from Tweets using NLP tools for collecting relevant news
17. Similarity in Patient Support Forums: Using TF-IDF and Cosine Similarity Metrics
18. Research on Structured Extraction Method for Function Points Based on Event Extraction
19. Research of software size estimation method
20. Comparing Soft Computing Techniques For Early Stage Software Development Effort Estimations

Metode penyelesaian :

1. Natural language processing
2. Algoritma naive bayes
3. Algoritma TF-IDF
4. Firefly Algorithm
5. Sparrow Search Algorithm

6. Neural network
7. COCOMO II / I
8. Fuzzy C Means Clustering
9. Fuzzy Gaussian
10. Feed Forward
11. Neural Network Backpropagation
12. Agile
13. Ant Colony Optimization Techniques
14. Machine learning algorithms
15. function point
16. Word2vec
17. Cosine similarity

Berdasarkan pengembangan metode software cost estimation yang dapat dilihat pada Tabel II.4 maka topik penelitian ini memenuhi level orisinalitas level 4(minimal) sesuai dengan standar tingkat keaslian yang bersumber dari pedoman penulisan proposal yang dikeluarkan oleh Program Studi Magister Teknik Informatika Universitas Hasanuddin (UNHAS) yang dapat dilihat pada Tabel II.6 berikut:

2.4 Target Hasil Penelitian

Berdasarkan tabel State of The Art penelitian yang telah dilakukan,target hasil dalam penelitian ini adalah penelitian ini mampu melakukan analisis nilai harga perangkat lunak pada dokumen spesifikasi kebutuhan.

Penelitian yang dilakukan oleh penulis termasuk dalam level enam yaitu memiliki orisinalitas.

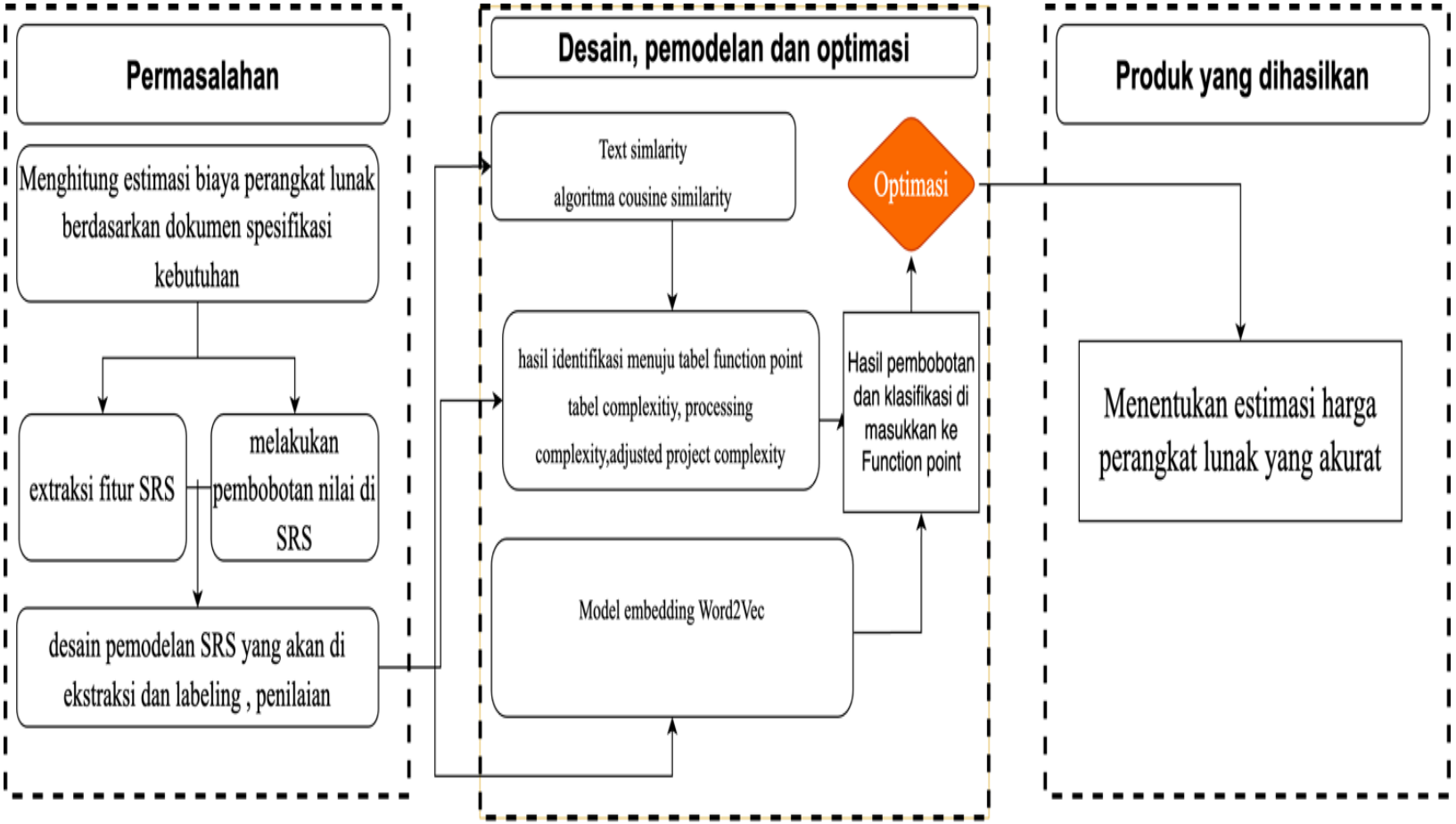
1. Objek Penelitian: (Terpakai). Ekstraksi fitur pada dokumen spesifikasi kebutuhan.
2. Jenis Permasalahan: (Baru) menghitung tiap section dokumen spesifikasi kebutuhan kedalam bobot nilai yang dihitung menggunakan function point
3. Metode Penyelesaian: (Tersedia) melakukan kombinasi algoritma atau metode dari naive bayes dengan Teknik multinomial dan algoritma TF-IDF

sehingga dapat digunakan melakukan perhitungan dan analisis berdasarkan dokumen spesifikasi kebutuhan yang telah ada.

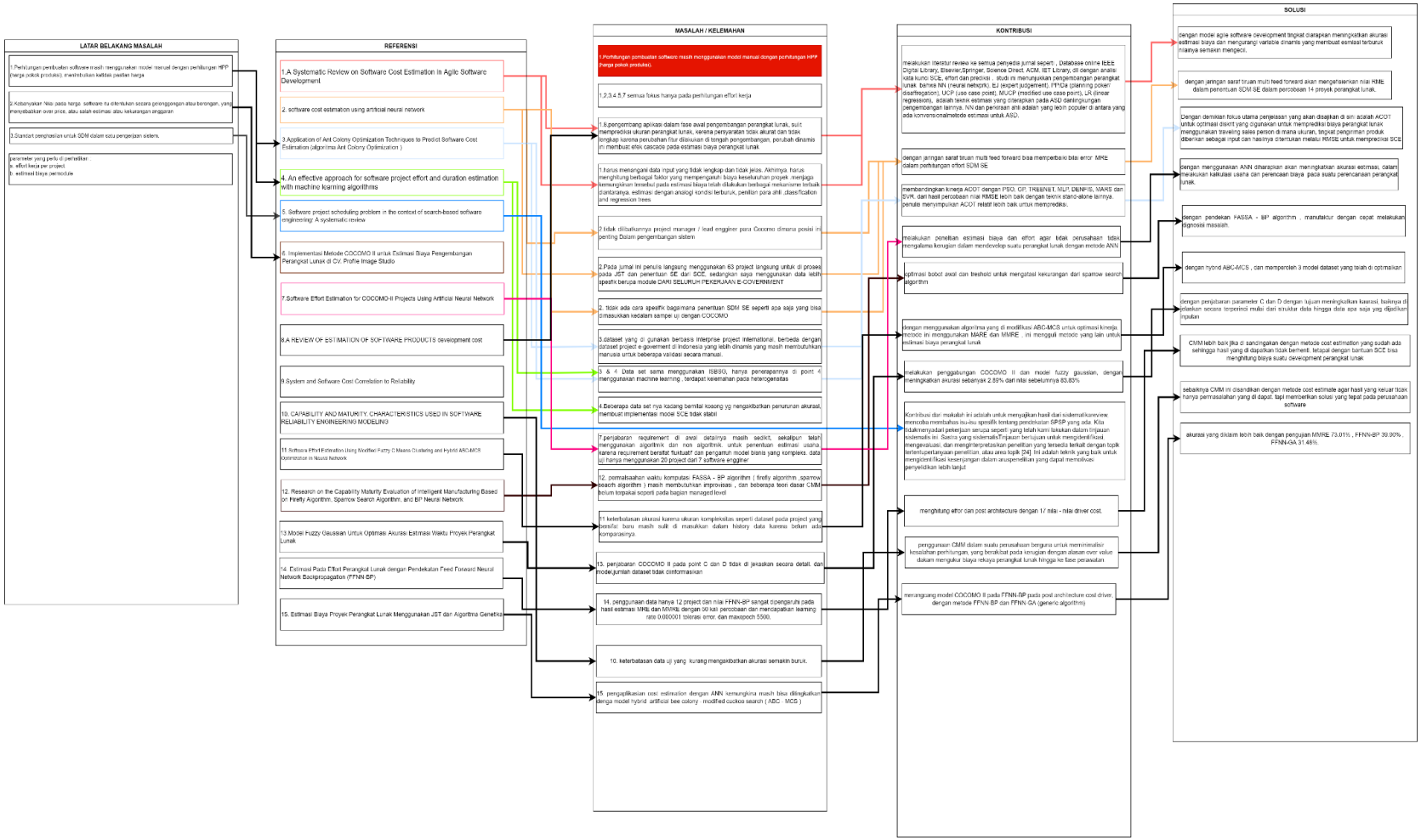
4. Status Orisinalitas: (Orisinalitas) yaitu kontribusi yang diberikan berupa Teknik analisis baru yang dapat menghitung anggaran secara akurat berdasarkan metode function point dan dokumen srs sebagai acuan utama.

2.5 Kerangka Pikir

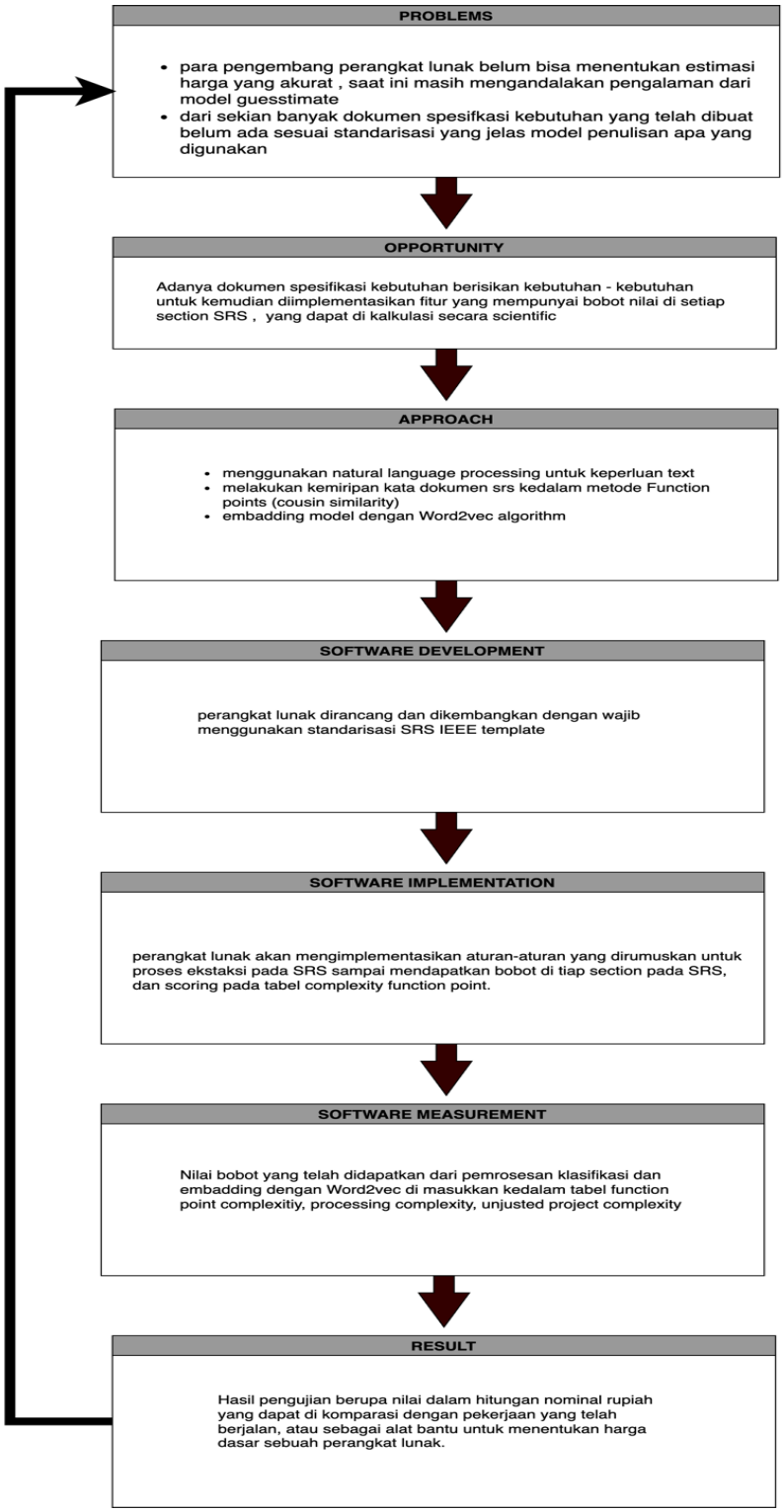
Kerangka pikir dapat dilihat pada Gambar [3,4,5], yang menjelaskan mengenai alur penelitian yang akan dilakukan.



Gambar 3. Kerangka pikir penelitian 1



Gambar 4. Kerangka pikir 2



Gambar 5. Kerangka pikir penelitian 3

2.6 Penelitian Terkait

Beberapa penelitian terkait mengenai Text mining, sistem estimasi harga perangkat lunak dengan beberapa metode berbeda seperti pada beberapa penelitian:

1. (M Syauqi Haris 2020) melakukan penelitian cost estimation merupakan bagian terpenting untuk melakukan perhitungan suatu perangkat lunak pada dokumen spesifikasi kebutuhan diantaranya, Melakukan ekstraksi fitur katalog pada SRS dengan output yang dihasilkan fiture model, dalam satu sistem mensimulasikan perangkat lunak bisa menjadi beberapa kemungkinan bentuk pada user experience hasil akhir penelitian ini menciptakan produk reuse pada perusahaan untuk mengurangi biaya pengembangan sampai 61%,mempercepat time-to-market dan meningkatkan kualitas hasil produksi yang di jelaskan.
2. (Kumar and Kumar 2020) Salah satu penyebab kegagalan untuk mengerjakan perangkat lunak tepat waktu adalah manajemen yang salah perhitungan di awal. Disebabkan requirement selalu berubah – ubah di tengah pekerjaan yang membuat effort yang dibutuhkan tidak bisa dihitung secara pasti. Dengan kumpulan data dari COCOMO II yang diimplementasikan dengan ANN untuk memprediksi harga.
3. (Dalpiaz et al. 2019) . bahwa meskipun penggunaan pemrosesan bahasa alami dapat membantu mengidentifikasi ambiguitas dan ketidak lengkapan dalam persyaratan perangkat lunak, namun inspeksi manual tetap diperlukan untuk memastikan keakuratan dan kelengkapan identifikasi. Kombinasi antara inspeksi manual dan alat bantu pemrosesan bahasa alami tampaknya merupakan pendekatan yang paling efektif dalam mengidentifikasi sinonim yang hampir sama yang dapat menyebabkan ambiguitas terminologi. Hasil eksperimen ini juga menunjukkan bahwa mengidentifikasi ambiguitas terminologi memakan waktu dan sulit untuk menentukan apakah sebuah istilah yang hampir sama dapat mempengaruhi

pengembangan sistem perangkat lunak yang tepat, sehingga para praktisi perlu memperhatikan hal ini dalam proses rekayasa persyaratan perangkat lunak.

4. (Azath, Mohanapriya, and Rajalakshmi 2020)Melakukan prediksi secara akurat upaya pengembangan perangkat lunak, karena hal ini penting menurut penulis. Yang akan berdampak pada buruknya fungsionalitas dan kualitas produk perangkat lunak, dan dapat merusak reputasi perusahaan penyedia itu sendiri. Dengan model expert judgement secara subjektif dianggap tidak cukup untuk menyelesaikan masalah ini.
5. (Rezende et al. 2019)Search-based optimization algorithms diterapkan pada SBSE untuk mengidentifikasi solusi paling optimal atau optimal waktu pengerjaan.
6. (Rizkiana Putri et al. n.d.) Penulis menggunakan cocomo II, juga menggunakan metode fuzzy gaussian yang digunakan dalam penelitian ini. Diharapkan memberikan hasil akurasi yang lebih baik. Melakukan penurunan parameter C dan D nilainya sebanyak 0.0001 dari nilai awal karena penulis mendapatkan kesalahan akurasi sebesar dari total 83.83%.
7. (as'ary ramadhan 2019) Mendapatkan nilai estimasi biaya proyek perangkat lunak yang lebih baik dari segi akurasi dengan menerapkan algoritma genetika pada proses pelatihan feed forward neural network backpropagation (FFNN-BP) yang telah disesuaikan arsitekturnya untuk mengakomodasi COCOMO II terhadap dataset COCOMO.
8. (Maqdum et al. 2019)Komparasi biaya pengembangan perangkat lunak, yaitu dengan membandingkan hasil estimasi biaya yang diperoleh menggunakan metode COCOMO II (Constructive Cost Model) dengan alokasi biaya yang dianggarkan oleh CV. Profile Image Studio.. Selisih estimasi biaya untuk seluruh perangkat lunak adalah sebesar 37,24% dengan selisih waktu sebesar 52,94% dan selisih SDM sebesar 40%.
9. (Ramadhan, Sihabuddin, and N n.d.) dengan menggunakan ANN yang mengakomodasi formula dari post architecture model (COCOMO II)

dengan model yang diusulkan memberikan hasil estimasi biaya proyek perangkat lunak menjadi lebih akurat. dalam kasus ini MMRE untuk COCOMO II adalah 73.01 .FFNN-BP 39.90% dan FFNN-GA adalah 31.48%.

10. (Arora and Mishra 2018) Penggambaran prosedur yang digunakan untuk menghitung effort kerja dengan model cocomo dengan sampel 63 perusahaan , untuk menentukan SDM engineer sesuai yang dihitung menggunakan MRE
11. (Bilgaiyan et al. 2017) menyajikan survei estimasi biaya yang sistematis dalam agile software development (ASD), yang akan berguna bagi pengguna untuk memahami tren terkini dalam estimasi biaya di agile software development ASD. Penelitian menunjukkan bahwa biaya yang dikeluarkan di sekitar 60%proyek berjalan di luar perkiraan awal mereka dan 15%proyek gagal karena estimasi yang sangat salah. Oleh karena itu, itu menjadi penting untuk melakukan analisis yang tepat dan akurat dari biaya yang diharapkan dan rencana yang sesuai.
12. (Venkataiah et al. 2017)Melakukan optimasi, penulis mengusulkan teknik ant colony untuk melakukan prediksi, menggunakan 3 dataset. Melakukan validasi silang 10 kali lipat pada perbandingan (ISBSG) dan dilakukan lagi validasi 3 kali lipat yang dilakukan di IBM data processing.
13. (Pospieszny, Czarnacka-Chrobot, and Kobylinski 2018) tujuan penelitian mempersempit gap antara penelitian terkini dalam implementasi machine learning (support vector machine, neural networks, generalized linear model). untuk mendapatkan alat pendukung pengambil keputusan untuk perusahaan yang sedang mengembangkan atau baru mengimplementasikan sistem perangkat lunak
14. (Osmanbegović, Suljić, and Agić 2017) penulis membuat atas dasar ketidak mampuan staff dalam penilaian perencanaan proyek perangkat lunak, ini akan berdampak pada kualitas perangkat lunak dan mengurangi keuntungan

perusahaan. penulis membuat kerangka kerja software cost estimation dengan melakukan review seluruh model pada SCE.

15. (Sarwosri et al. 2011) SRS diberi tag menggunakan algoritma part-of-speech tagging. Terakhir dokumen tersebut diberi bobot menggunakan algoritma TF-IDF untuk mengetahui kepentingan kalimat tersebut. Setelah semua proses selesai dan metadata telah didapatkan, sistem akan membentuk RDF dari RDF skema yang telah dibuat secara manual. Kemudian sistem akan memasukkan metadata ke dalamnya
16. (Jayasiriwardene and Ganegoda 2020). Dalam satu thread yang hanya berupa 140 karakter terdapat kalimat yg bersifat ambiguitas seperti kalimat yang disingkat – singkat yang tidak dimengerti oleh mesin untuk melakukan pencarian kata dasar dan bobot suatu kepentingan kata tersebut
17. (Alodadi and Janeja 2015) Melakukan pengolahan data percakapan pada forum , yang dimana memproses kedekatan dari suatu pertanyaan baru dengan pertanyaan lama, agar tidak ada thread pada postingan forum berbentuk duplikat referensi ini penulis gunakan untuk melihat cara menghitung suatu kedekatan kalimat
18. (Ren and Yun 2013)komparasi Estimasi ukuran perangkat lunak meletakkan dasar data untuk perencanaan proyek. Di antara semua metode estimasi perangkat lunak,metode Function point, COCOMO II, Pert, Delphi
19. (Kumar and Kumar 2020). Salah satu penyebab kegagalan untuk mengerjakan perangkat lunak tepat waktu adalah manajemen yang salah perhitungan di awal. Disebabkan requirement selalu berubah–ubah di tengah pekerjaan yang membuat effort yang dibutuhkan tidak bisa dihitung secara pasti. Dengan kumpulan data dari COCOMO II yang diimplementasikan dengan ANN untuk memprediksi harga
20. (Bhatnagar 2012)Melakukan penelitian perkiraan ukuran perangkat lunak , biaya, dan effort yang dibutuhkan untuk meminimalisir keterlambatan agar tidak mengalami kerugian bagi perusahaan.