Pengukuran Software Metric Terhadap Aplikasi Sistem Akuntansi Instansi Berbasis Akrual Untuk Pengembangan Sistem Informasi Kementerian Pemuda dan Olahraga

Julianto Rahmadi dan Karmilasari

Sistem Informasi Bisnis, Universitas Gunadarma
Jl. Margonda Raya 100, Depok
E-mail: julianto_rahmadi@yahoo.com, karmila@staff.gunadarma.ac.id

Abstrak

Tujuan penulisan ini dalah untuk melakukan pengukuran dan analisis terhadap aplikasi SAIBA dalam menemukan kekurangan atau kelemahan yang ada pada aplikasi sistem akuntansi instansi berbasis akrual yang dimiliki oleh kementerian pemuda dan olahraga dengan melakukan pengukuran terhadap aplikasi tersebut tersebut agar dapat dilakukan perbaikan sehingga proses akuntansi dapat berjalan lebih baik. Adapun metode yang dilakukan adalah sebagai berikut (1) Memahami proses bisnis instansi, dalam tahap ini adalah untuk mengetahui bagaimana proses data dapat berjalan hingga sampai pada penggunaan aplikasi SAIBA dan unit kerja apa saja yang terlibat dalam proses perjalan data hingga data dapat diolah sampai menjadi laporan keuangan (2) Review modul aplikasi SAIBA, dalam tahap ini adalah mengetahui modul apa saja yang terdapat dalam aplikasi SAIBA sehingga dapat diperoleh informasi tentang tabel proses input-output yang ada di aplikasi tersebut (3) Penyiapan Software Aplikasi, tujuan pada tahap ini adalah untuk mempersiapkan software SAIBA secara keseluruhan sebelum dilakukan pengujian dengan software metric (4) Pemilihan Software Metric, Tujuan dari tahap ini adalah untuk menentukan software metric apa yang cocok dengan aplikasi SAIBA, dalam penelitian ini digunakan software metric seperti Cppcheck dan Cppdepend (5) Pengujian Aplikasi Dengan Software Metric, Tujuan dari tahap ini adalah pengukuran terhadap aplikasi SAIBA yang sebelumnya sudah dipersiapkan dan dicari kecocokannya dengan software metric yang dapat digunakan (6) Hasil pengujian, Tujuan dari tahap ini adalah hasil yang didapat dari pengujian aplikasi SAIBA dengan software metric yang akan menampilkan kelebihan dan kekurangan dari aplikasi tersebut. Analisis yang dilakukan dengan menggunakan Cppcheck tidak menemukan kekurangan pada algoritma, container, function, iterator namun pengukuran dengan menggunakan Cppdepend ditemukan berbagai kekurangan pada cyclomatic complexity dan line of code, lalu juga ditemukan kekurangan 3 kritikal yang ada pada blok program global compile dan global match.

Kata Kunci : Pengukuran, Software Metric, Cppcheck, Cppdepend, Cyclomatic Complexity, Line Of Code, Sistem Akuntansi Instansi Berbasis Akrual.

Pendahuluan

Latar Belakang

Satuan ukuran kualitas perangkat lunak pada saat proses rekayasa dapat meliputi kompleksitas program, modularitas yang efektif dan besarnya program [5].Dalam siklus hidup pengembangan perangkat lunak ada 2 teknik yang digunakan untuk menjamin kualitas dan keandalan perangkat lunak yaitu software testing dan software metrics [6]. Software testing adalah istilah yang digunakan untuk menemukan kesalahan saat pengembangan atau pembangunan perangkat lunak. Sedangkan, software metrics digunakan untuk meningkatkan kualitas perangkat lunak dengan memantau kompleksitas perangkat lunak [6]. Kearney (1986) Mendefinisikan kompleksitas perangkat lunak sebagai sejauh mana sistem atau komponen memiliki desain atau implementasi yang sulit dimengerti dan diverifikasi [6].

Kode Sumber (source code) merupakan setiap deskripsi perintah yang dapat dieksekusi oleh perangkat lunak [6]. Selain itu, kode sumber juga diperlukan sebagai informasi untuk mengidentifikasi dan memperbaiki masalah tentang sebuah bug [6]. Kompleksitas kode sumber dapat diukur dengan beberapa metrik yaitu LOC (line of code), Mccabe's Cylomatic Complexity dan Halstead's Volume [6]. LOC adalah suatu teknik pengukuran dengan cara menghitung jumlah keseluruhan baris kode pada seluruh file kode sumber [7].

Dalam suatu Instansi Pemerintah diperlukan su-

atu aplikasi yang dapat mempermudah dan mempercepat untuk pemerosesan data yang sangat besar dan kompleks agar data-data tersebut tidak hilang atau bahkan terabaikan karena sangat banyak data yang tergolong penting dalam sebuah Instansi tersebut. Salah satu dari Instansi tersebut adalah Kementrian Pemuda Dan Olahraga (KEMENPORA) yang merupakan sebuah Instansi Pemerintah yang memerlukan Aplikasi untuk menunjang kinerja dari pegawai dalam pengolahan data dibidang akuntasi.

Sebelum tahun 2015 KEMENPORA menggunakan aplikasi yang bernama SAKPA untuk membantu pengkonversian data untuk dijadikan laporan keuangan. Namun setelah adanya Peraturan Pemerintah untuk Akuntansi Akrual maka aplikasi SAKPA tidak dapat dipergunakan lagi oleh KEMENPORA sehingga dibutuhkan aplikasi yang dapat membantu pemrosesan data yang menunjang akuntasi akrual. Sehingga Kementrian Keuangan memberikan solusi yaitu suatu aplikasi yang saling terintegrasi dengan Kementrian Keuangan dan menunjang akuntansi akrual yaitu Software Akuntansi Instansi Berbasis Akrual (SAIBA) dan digunakan untuk seluruh Instansi Pemerintahan saat ini.

Aplikasi Sistem Akuntansi Instansi Berbasis Akrual yang dimiliki Kementerian Pemuda dan Olahraga adalah sebuah Aplikasi yang digunakan untuk memproses akuntansi akrual di kementerian Pemuda dan Olahraga. Yang dialami oleh user dari kementerian pemuda dan olahraga adalah terlalu sering nya data yang dikonversi dari aplikasi sebelumnya tidak masuk kedalam aplikasi SAIBA sehingga diperlukannya proses manual untuk memasukkan data tersebut yang mengakibatkan lamanya dalam pengolahan data akuntansi pada bidang tersebut. Dibutuhkan penyelesaian masalah yang tepat sasaran untuk menyelesaikan masalah tersebut agar tidak membuat sistem informasi terkendala, Sehingga diperlukannya pengukuran agar diketahui kelemahan dari sistem aplikasi tersebut.

Dikarenakan kekurangan dari aplikasi SAIBA dalam pengkonversian data yang tidak maksimal sehingga memaksa pegawai untuk melakukan pengisian data secara manual menimbulkan keterlambatan dalam pemrosesan data akuntansi. Sehingga dilakukan pengukuran terhadap aplikasi SAIBA agar dapat diketahui kelemahan sesungguhnya dari aplikasi tersebut dengan menggunakan software metric yang dapat mengukur kinerja dari sebuah sistem atau aplikasi.

Tujuan dari penelitian ini adalah untuk melakukan analisis Software SAIBA dalam menemukan kekurangan / kelemahan yang ada pada aplikasi Sistem Akuntansi Instansi Berbasis Akrual kementerian pemuda dan olahraga dengan melakukan pengukuran terhadap aplikasi tersebut agar dapat dilakukan perbaikan sehingga proses akuntasi dapat berjalan lebih baik.

Pengukuran dan Perangkat Lunak Pengukuran

Definisi Pengukuran

Pengukuran merupakan dasar dari setiap disiplin rekayasa dan berlaku juga dalam perekayasaan perangkat lunak. Untuk mengevaluasi performa suatu system atau proses diperlukan suatu mekanisme untuk mengamati dan menentukan tingkat efisiensinya. Melalui pengukuran, maka akan diperoleh tingkat pencapaian di dalam perangkat lunak yang sedang diamati. Pengukuran menurut IEEE (1990) adalah ukuran kuantitatif dari tingkat dimana sebuah system, komponen atau proses memiliki atribut tertentu. Sedangkan mengukur (measure) adalah mengindikasikan kuantitatif dari luasan, jumlah, dimensi dan kapasitas [4].

Metrik Perangkat Lunak

Ukuran merupakan faktor utama untuk menentukan biaya, penjadwalan, dan usaha.Kegagalan dari perkiraan ukuran yang tepat akan mengakibatkan penggunaan biaya yang berlebih atau keterlambatan penyelesaian proyek. Menurut Sommerville (2003), pengukuran terbagi atas dua, yaitu pengukuran(metrik) kontrol dan pengukuran (metrik) prediktor [1]. Metrik kontrol biasanya dihubungkan dengan proses perangkat lunak, misalnya usaha dan waktu rata-rata yang dibutuhkan untuk memperbaiki cacat yang dilaporkan. Sedangkan metrik prediktor berhubungan dengan produk perangkat lunak, misalnya kompleksitas sikomatik modul, panjang rata-rata identifier pada program dan jumlah atribut dan operasi yang berhubungan dengan objek pada suatu rancangan [2].

Metrik Proses

Metrik proses digunakan untuk tujuan strategis yaitu untuk Mengukur reliabilitas proses PL secara tidak langsung yaitu dengan mengambil serangkaian metrik berdasarkan keluaran yg dapat diambil oleh proses [2]. Cara untuk meningkatkan proses perangkat lunak :

1. Mengukur atribut tertentu dari proses
2. Mengembangkan serangkaian metrik yg berarti
3. Menggunakan metrik itu untuk memberikan indikator yg akan membawa kepada sebuah strategi pengembangan.

Mengukur efektivitas proses dengan menggunakan satu set metrik berdasarkan pada hasil/keluaran dari proses seperti :

1. Kesalahan ditemukan sebelum rilis dari perangkat lunak

2. Cacat pengiriman ke dan dilaporkan oleh pengguna akhir
3. Penyebaran produk kerja
4. Usaha manusia yang dikeluarkan
5. Waktu kalender yang dikeluarkan
6. Kesesuaian dengan jadwal
7. Waktu dan usaha untuk menyelesaikan setiap kegiatan generik

Perangkat Lunak Model ISO 9126

Kualitas perangkat lunak dapat dinilai melalui ukuran-ukuran dan metode-metode tertentu, serta melalui pengujian-pengujian software. Salah satu tolak ukur kualitas perangkat lunak adalah ISO 9126, yang dibuat oleh International Organization for Standardization (ISO) dan International Electrotechnical Commission (IEC). ISO 9126 mendefinisikan kualitas produk perangkat lunak, model, karakteristik mutu, dan metrik terkait yang digunakan untuk mengevaluasi dan menetapkan kualitas sebuah produk software. Standar ISO 9126 telah dikembangkan dalam usaha untuk mengidentifikasi atribut-atribut kunci kualitas untuk perangkat lunak komputer. Faktor kualitas ISO 9126 meliputi enam karakteristik kualitas sebagai berikut:

1. Functionality (Fungsionalitas). Kemampuan perangkat lunak untuk menyediakan fungsi sesuai kebutuhan pengguna, ketika digunakan dalam kondisi tertentu.
2. Reliability (Kehandalan). Kemampuan perangkat lunak untuk mempertahankan tingkat kinerja tertentu, ketika digunakan dalam kondisi tertentu.
3. Usability (Kebergunaan). Kemampuan perangkat lunak untuk dipahami, dipelajari, digunakan, dan menarik bagi pengguna, ketika digunakan dalam kondisi tertentu.
4. Efficiency (Efisiensi). Kemampuan perangkat lunak untuk memberikan kinerja yang sesuai dan relatif terhadap jumlah sumber daya yang digunakan pada saat keadaan tersebut.
5. Maintainability (Pemeliharaan). Kemampuan perangkat lunak untuk dimodifikasi. Modifikasi meliputi koreksi, perbaikan atau adaptasi terhadap perubahan lingkungan, persyaratan, dan spesifikasi fungsional.
6. Portability (Portabilitas). Kemampuan perangkat lunak untuk ditransfer dari satu lingkungan ke lingkungan lain.

Pengukuran Yang Berhubungan Dengan Ukuran

Pegukuran yang berhubungan dengan ukuran (Metric Size Oriented) adalah pengukuran dengan normalisasi kualitas dan produktifitas atau mempertimbangkan ukuran perangkat lunak yang dihasilkan. Metric size oriented dilakukan dengan menghitung LOC (Line of Code) dari baris kode suatu perangkat lunak [3]. Pada pengembangan, pengukuran ini juga dapat mengukur:

1. Kesalahan per KLOC (Kilo Line Of Code)
2. Biaya per LOC
3. Cacat per LOC
4. Halaman dokumentasi per LOC
5. Kesalahan perorang perbulan
6. LOC perorang perbulan
7. Biaya perhalaman dokumentasi.

Line Of Code (LOC)

Menurut Pressman (2001) LOC merupakan ukuran yang kurang akurat dan merupakan sebuah topik yang menimbulkan perdebatan selama bertahun-tahun,dipandang sebagai sebuah ukuran untuk mengestimasi biaya dan waktu, tidak dapat dipastikan bahwa dua program yang mempunyai LOC sama akan membutuhkan waktu implementasi yang sama walaupun keduanya diimplementasikan dengan kondisi pemrograman yang standard. Meskipun metode ini kurang akurat dan merupakan metodologi yang belum diterima secara luas, tetapi metrik dengan orientasi ukuran ini merupakan kunci pengukuran dan banyak estimasi software yang menggunakan model ini [8].

Cyclomatic Complexity

Cyclomatic Complexity adalah sebuah *software metric* yang menyediakan ukuran kuantitatif dari kompleksitas logika dari sebuah program [10]. Dengan menggunakan hasil pengukuran atau perhitungan dari *metric cyclomatic complexity*, kita dapat menentukan apakah sebuah program merupakan program yang sederhana atau kompleks berdasarkan logika yang diterapkan pada program tersebut [10]. Apabila dikaitkan dengan pengujian perangkat lunak (*software testing*), *cyclomatic complexity* dapat digunakan untuk menentukan berapa minimal *test case* yang harus dijalankan untuk menguji sebuah program dengan menggunakan teknik *basis path testing* [10]. Pada pengujian *basis path*, aliran control logika digambarkan dengan menggunakan flow graph [9]. *Cyclomatic complexity* dari sebuah program dapat dibuat dengan menggunakan rumus dibawah ini :

$V(G) = E - N + 2$

keterangan :
V(G) : *cyclomatic complexity*
E : total jumlah *edge*
N : total jumlah *node*

Untuk menggunakan rumus-rumus ini untuk mendefinisikan *cyclomatic complexity* dari suatu modul, gambar grafik aliran kontrol G dari modul. Untuk membangun grafik kontrol aliran modul program, pisahkan modul ke dalam blok yang dibatasi oleh pernyataan yang mempengaruhi *control flow*, *likeif*, *while*, *repeat*, dan *go to* [9].

Cppdepend

CppDepend adalah alat analisis statis untuk kode C / C ++. Alat ini mendukung sejumlah besar metrik kode, memungkinkan visualisasi dependensi menggunakan grafik terarah dan matriks dependensi. Alat ini juga melakukan perbandingan snapshot basis kode, dan validasi aturan arsitektur dan kualitas. Aturan yang ditentukan pengguna dapat ditulis menggunakan kueri LINQ. Kemungkinan ini bernama CQLinq. Alat ini juga dilengkapi dengan sejumlah besar aturan kode CQLinq yang telah ditentukan.

Cppcheck

Cppcheck adalah alat analisis kode statis untuk bahasa pemrograman C dan C ++. Ini adalah alat serbaguna yang dapat memeriksa kode non-standar. Pencipta dan pengembang utama adalah Daniel Marjamäki. Cppcheck mendukung berbagai pemeriksaan statis yang mungkin tidak dicakup oleh kompiler itu sendiri. Pemeriksaan ini adalah pemeriksaan analisis statis yang dapat dilakukan pada tingkat kode sumber. Program ini diarahkan ke pemeriksaan analisis statis yang ketat, daripada heuristik. Beberapa cek yang didukung meliputi:

1. Pemeriksaan variabel otomatis
2. Batas memeriksa overruns array
3. Pemeriksaan kelas
4. Penggunaan fungsi yang ditinggalkan atau diganti menurut Open Group
5. Pengecekan keamanan pengecualian
6. Kebocoran memori
7. Kebocoran sumber daya
8. Penggunaan fungsi dan idiom Perpustakaan Template Standar tidak valid
9. Kesalahan gaya dan kinerja lainnya

Seperti halnya banyak program analisis, ada banyak kasus pemrograman yang tidak biasa yang dapat diterima dalam kasus target tertentu, atau di luar ruang lingkup programmer untuk koreksi kode sumber. Sebuah studi yang dilakukan pada Maret 2009 mengidentifikasi beberapa area di mana positif palsu ditemukan oleh Cppcheck, tetapi tidak menentukan versi program yang diperiksa. Cppcheck telah diidentifikasi untuk digunakan dalam sistem seperti paket analisis meta CERNs 4DSOFT, untuk verifikasi kode dalam perangkat pembacaan partikel energi tinggi, perangkat lunak pemantauan sistem untuk teleskop radio serta dalam analisis kesalahan besar proyek, seperti OpenOffice.org dan arsip Debian.

PEiD

PE iDentifier atau PEiD adalah sebuah aplikasi yang membaca file PE atau file executable(exe) dan dynamic link library , program PEid ini dapat mendeteksi suatu packers yang paling umum seperti cryptors dan kompiler untuk file PE. Saat ini PEiD dapat mendeteksi lebih dari 470 signatures yang berbeda dalam file PE. PEiD menjadi alat pertama yang dipakai ketika menganalisa sebuah file PE.

Perhitungan Komputasi Numeric

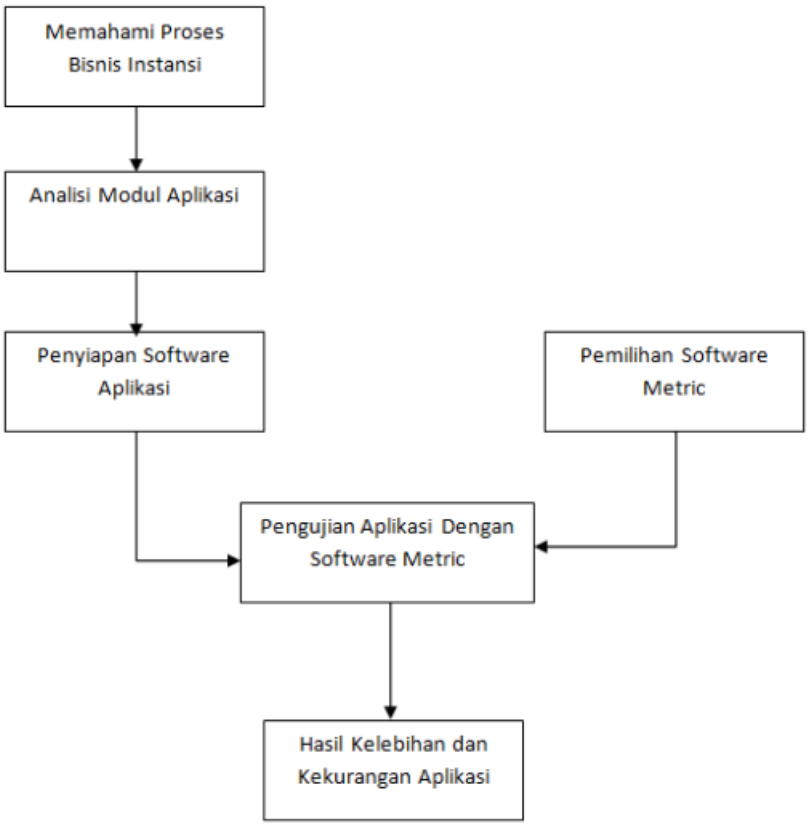
Perhitungan komputasi numeric pada metric merupakan metric peangkat lunak berorientasi fungsi ditarik berdasarkan sebuah pengukuran fungsionalitas yang disampaikan oleh aplikasi sebagai suatu nilai normalisasi. Karena fungsionalitas tidak dapat diukur secara langsung, maka fungsionalitas harus ditarik secara tidak langsung dari pengukuran langsung lainnya. Metrik berorientasi fungsi dibuat oleh Alan J. Albrecht (1979) yang disebut dengan *function point* [5]. Saat ini ada banyak variasi cara perhitungan *function point* setelah metode ini dikembangkan dan direvisi oleh International Function Point User Group (IFPUG) sejak tahun 1986.

Metodologi Penelitian

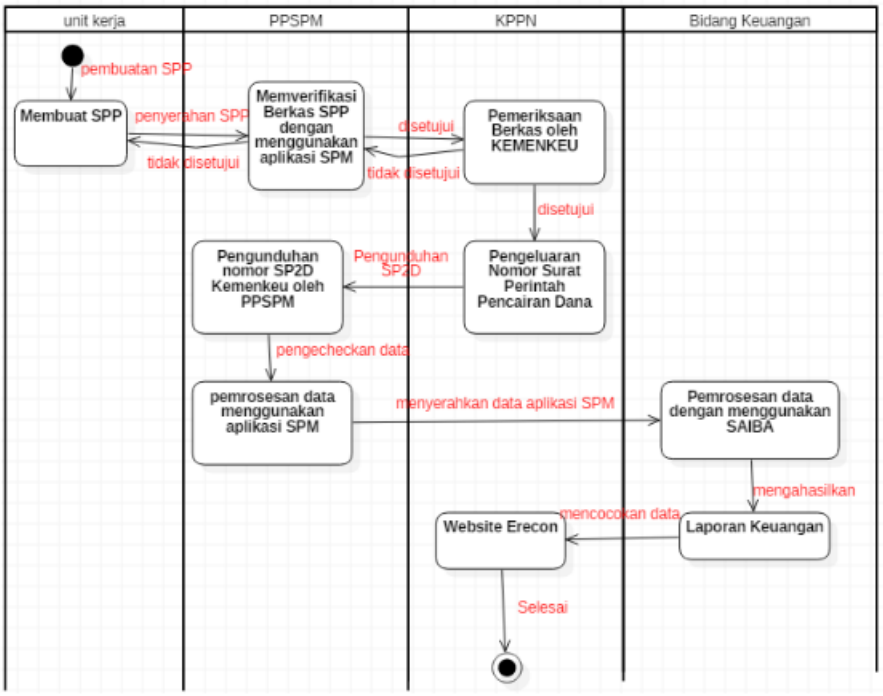
Gambar 1 menyajikan kerangka ataupun metodologi penelitian yang digunakan pada penlitian ini.

Review Proses Bisnis

Pada tahap ini dilakukan untuk mengetahui bagaimana proses data dapat berjalan hingga sampai pada penggunaan aplikasi SAIBA dan unit kerja apa saja yang terlibat dalam proses perjalan data hingga data dapat diolah sampai menjadi laporan keuangan. adapun proses bisnis yang terdapat dalam Kementerian Pemuda dan Olahraga dapat dilihat pada Gambar 2.



Gambar 1: Kerangka Umum Penelitian



Gambar 2: Proses Bisnis Pembuatan laporan keuangan.

Tahapan Proses Bisnis :

1. Setiap unit kerja akan membuat Surat Permohonan Pembayaran (SPP) untuk memenuhi kebutuhan dari setiap unit kerja tersebut agar bisa berjalan sebagai mana mestinya. Pada saat pengajuan surat permohonan pembayaran (SPP) setiap unit diperlukan melampirkan kode kegiatan dan akun setiap hal yang di lakukan agar dapat terdata

dengan baik dan benar. Surat Permohonan Pembayaran (SPP) diserahkan kepada Pejabat Penandatanganan Surat Perintah Membayar (PPSPM) agar dapat diajukan kepada Kementerian Keuangan.

2. Pejabat Penandatanganan Surat Perintah Membayar (PPSPM) akan memverifikasi berkas dari setiap unit kerja dengan menggunakan aplikasi SPM yang terdapat dalam aplikasi SAS. Apabila berkas dari unit kerja disetujui oleh PPSPM maka berkas tersebut akan dikirimkan kepada Kementerian Keuangan Namun bila dinyatakan tidak disetujui maka akan dikembalikan kepada unit kerja untuk diperbaiki. Syarat dari kelengkapan berkas tersebut adalah sebagai berikut :

a) Nomor spp
b) Kode Pembayan
c) Kode Kegiatan
d) Nomor Akun
e) Nomor Rekening Penerima
f) Dokumen Pendukung

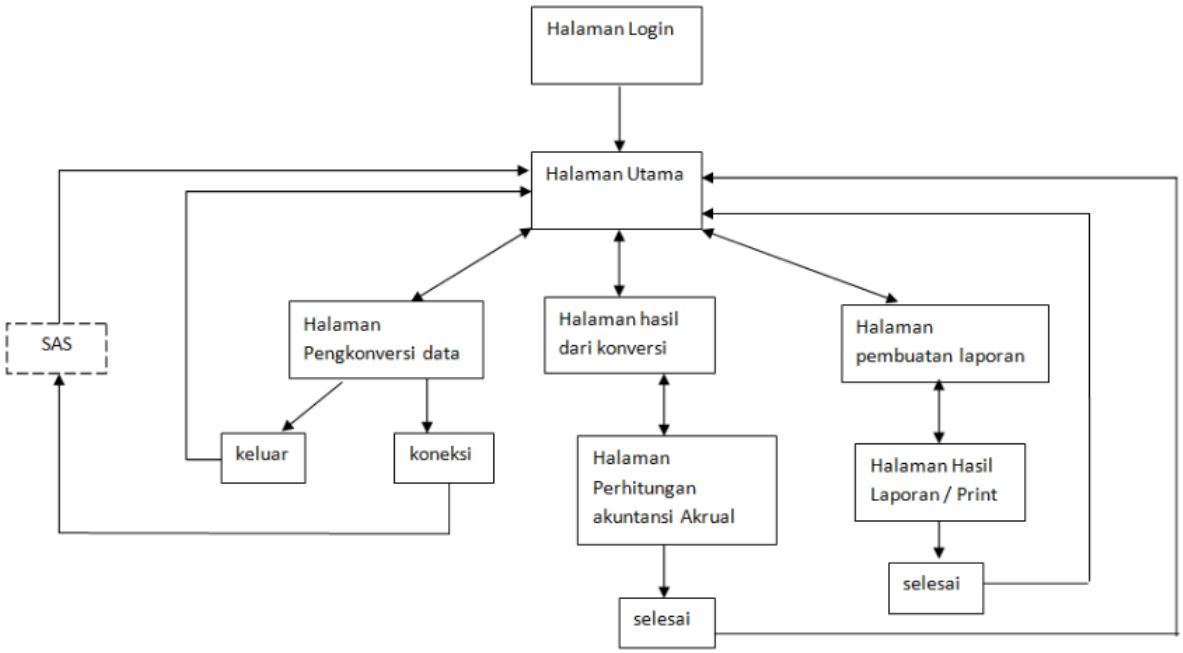
3. Setelah di verifikasi oleh PPSPM maka diajukan berkas tersebut kepada Kementerian Keuangan untuk diperiksa apakah layak unit kerja tersebut mendapatkan dana dari Kementerian Keuangan. Apabila layak maka akan dikeluarkan nomor Surat Perintah Pencairan Dana (SP2D), Namun apabila tidak disetujui maka Kementerian Keuangan akan mengembalikan berkas kepada PPSPM.

4. PPSPM akan mendownload nomor SP2D tersebut dengan menggunakan aplikasi SPM dan memprosesnya agar bisa dijadikan laporan keuangan.

5 Setelah nomor SP2D didownload dan diproses oleh PPSPM maka selanjutnya diserahkan kepada bagian keuangan untuk diproses menjadi laporan keuangan dengan menggunakan SAIBA. bagian keuangan dan akuntansi bekerja menggunakan SAIBA untuk mengolah nomor yang telah diberikan bagian PPSPM untuk menjadi laporan keuangan, SAIBA akan mengkonversi nomor yang ada pada SPM dan mencocokan dengan kode kegiatan dan nomor akun yang diberikan oleh unit kerja pada berkas kelengkapan setiap unit kerja yang mengajukan SPP. Setelah selesai menjadi laporan keuangan maka bagian keuangan akan mencocokan data laporan keuangan milik Kementerian Keuangan dengan Kementerian Pemuda dan Olahraga agar tidak terjadi selisih. Apabila terjadi selisih maka pihak bagian keuangan Kementerian Pemuda dan Olahraga akan memberikan print out dari hasil SAIBA kepada Kementerian Keuangan secara langsung.

Review Modul Aplikasi SAIBA

Pada tahap ini dilakukan untuk mengetahui modul apa saja yang terdapat dalam aplikasi SAIBA sehingga dapat diperoleh informasi tentang tabel proses input-output yang ada di aplikasi tersebut. Berikut adalah Model aplikasi SAIBA.



Gambar 3: Modul Aplikasi

Langkah-langkah yang terdapat dalam SAIBA adalah sebagai berikut :

1. Pertama-tama seorang operator dari setiap unit kerja yang telah disetujui akan mengambil nomor dengan menggunakan aplikasi SAS setelah mendapatkan nomor SP2D operator dari setiap unit kerja melakukan login kedalam aplikasi SAIBA untuk melakukan konversi data dari aplikasi SAS kedalam aplikasi SAIBA. Hal tersebut membutuhkan koneksi LAN yang diantara 2 leptop yang mana leptop yang terdapat aplikasi SPM dan leptop yang terdapat aplikasi SAIBA.
2. Lalu setelah pengkonversian data selesai maka operator akan melihat halaman hasil konversi data, bila terjadi ketidak sempurnaan dalam pengkonversian data yang mana hal tersebut terlihat dengan adanya data yang kurang dalam hal permintaan yang ada di Surat permohonan pembayaran (SPP) maka operator akan melakukan penginputan data secara manual dengan memasukkan data dari SPP secara manual kedalam SAIBA.
3. Setelah penginputan selesai dilakukan maka akan dilakukan perhitungan secara otomatis oleh SAIBA secara akrual dan langsung dapat dilihat hasil nya.
4. Halaman hasil dari perhitungan akrual diberikan kepada pihak bidang akuntansi untuk dilakukan rekonsialisiasi atau yang biasa disebut pengecekan, apakah ada terjadi selisih atau kesalahan dalam perhitungan akrual dan dilakukan mencocokan kepada website Kementerian Keuangan yaitu Omspan. bila ada kesalahan perhitungan bidang akuntansi akan melakukan pemberian data kepada pihak Kementerian Keuangan dan dilakukan pengecekkan ulang bersama.
5. Apabila perhitungan dari SAIBA telah benar maka hal selanjutnya bidang akuntansi dapat membuat laporan keuangan dengan menggabungkan 4 laporan yang telah dilakukan perhitungan oleh SAIBA yaitu :
   a. Neraca
   b. Laporan operasional (L/O)
   c. Laporan Perubahan Ekuitas (LPE)
   d. Laporan Realisasi Anggaran (LRA)
6. Laporan tersebut digabung dan menghasilkan laporan keuangan, setelah itu bidang akuntansi kembali mencocokan data kepada Kementerian Keuangan dengan website Erecon. Apabila data telah cocok maka dapat diserahkan kepada Kementerian Keuangan pada bagian Pelaporan Keuangan. Namun apabila tidak sesuai maka data dilakukan pengecekan langsung di Kementerian Keuangan yang dilakukan oleh bidang akuntansi secara manual menyerahkan kepada Kantor Kementerian Keuangan hasil dari perhitungan SAIBA.

Penyiapan Software Aplikasi

Pada tahap ini dilakukan untuk mempersiapkan software SAIBA secara keseluruhan sebelum dilakukan pengujian dengan software metric yaitu dengan mengetahui berapa jumlah LOC dan berapa jumlah tabel yang terdapat pada aplikasi SAIBA sehingga dapat di ukur menggunakan software metric dengan baik dan benar. Mencari tahu bahasa pemrograman yang digunakan oleh SAIBA yaitu C++, dengan demikian dapat diketahui software metrik apa yang cocok dengan aplikasi SAIBA. Modul apa saja yang terdapat pada SAIBA sehingga dapat dilakukan pemahaman tentang aplikasi SAIBA, Prosedur yang terdapat pada SAIBA diperlukan untuk mengetahui tahapan dalam pembuatan analisis. Database yang digunakan SAIBA adalah mysql yang merupakan multi-user yaitu hal yang diperlukan oleh SAIBA dikarenakan SAIBA digunakan oleh hampir seluruh unit kerja.

Pemilihan Software Metric

Pada tahap ini dilakukan untuk menentukan software metric apa yang cocok dengan aplikasi SAIBA sehingga tidak terjadi crash saat pengujian dikarenakan perbedaan bahasa pemrograman yang digunakan oleh software metric dengan aplikasi SAIBA, hal ini diperlukan untuk mengetahui bahasa pemprograman apa yang digunakan oleh aplikasi SAIBA setelah itu baru kita bisa memilih software metric yang sesuai dengan SAIBA.

Pada saat ini software metric yang cocok untuk aplikasi SAIBA adalah CPP Depend dan CPP Check yang menggunakan bahasa pemrograman C++ seperti aplikasi SAIBA. Karakteristik dari tools software metric CPPDepend adalah mengevaluasi secara lengkap seperti cyclomatic complexity, Line of code, coupling, dan nesting depth pada suatu aplikasi. CPP Depend juga menampilkan secara visual dependensi dari code dengan mendifinisikan aturan design, melakukan analisis dampak, dan membandingkan berbagai versi. Lalu Cpp Check yaitu untuk mengetahui beberapa type error yang ada pada aplikasi dengan menggunakan Standard Template Library (STL) yang mengandung 4 komponen utama yaitu algoritma, container, function, iterator.

Pengujian Aplikasi Dengan Software Metric

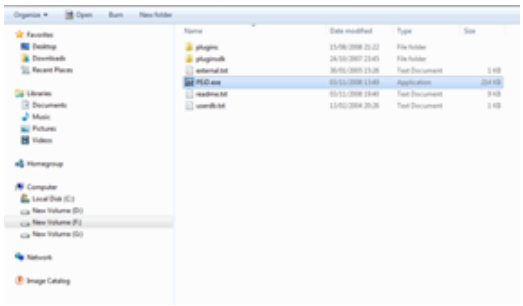
Pada tahap ini dilakukan untuk melakukan pengukuran terhadap aplikasi SAIBA yang sebelumnya sudah dipersiapkan dan dicari kecocokannya dengan software metric yang dapat digunakan. Setelah semua terpenuhi maka pengukuran dapat dilakukan untuk mengetahui kelebihan yang dimiliki oleh aplikasi SAIBA dan kekurangan yang dimiliki oleh aplikasi SAIBA yang merupakan hal yang ingin dianalisa.

Hasil pengujian

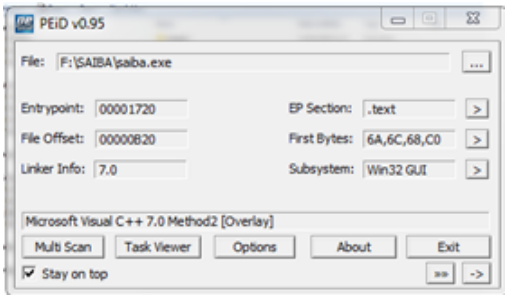
Pada tahap ini dilakukan untuk melihat hasil yang didapat dari pengujian aplikasi SAIBA dengan software metric yang akan menampilkan kelebihan dan kekurangan dari aplikasi tersebut. Dan dilakukan analisis terhadap kekurangan dari aplikasi SAIBA untuk dijadikan pertimbangan dalam proses maintenance dalam hal meninggkatkan peforma aplikasi.

Hasil dan Pembahasan

Pengujian Dengan Software PEID



Gambar 4: Halaman Exe Aplikasi PEID

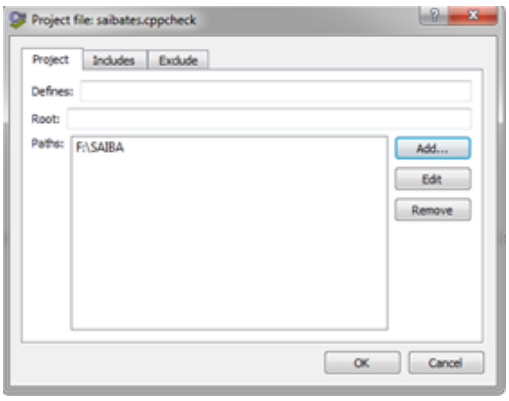


Gambar 5: Halaman Hasil Ujicoba Aplikasi PEID

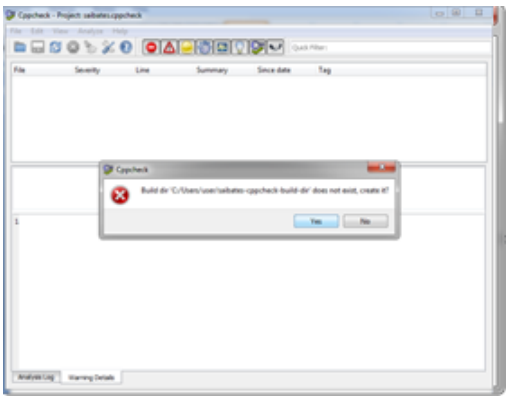
Bahasa pemrograman yang digunakan oleh aplikasi SAIBA dapat dilihat pada bagian bawah aplikasi PEID seperti gambar diatas. Sehingga dapat dipilih software metric yang sesuai untuk mengukur aplikasi SAIBA.

Uji coba dengan software metric CPPcheck

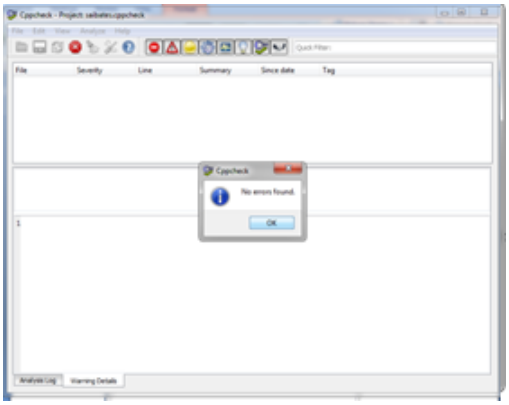
Tidak ditemukannya error oleh aplikasi CPPcheck dalam hal yaitu algoritma, container, function, iterator dari aplikasi SAIBA sehingga tidak adanya report dalam aplikasi CPPcheck ini.



Gambar 6: Halaman Hasil Penambahan Aplikasi SAIBA ke CPPcheck



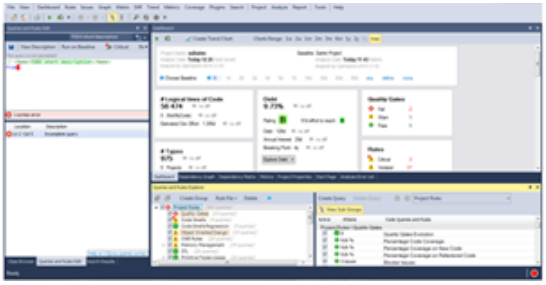
Gambar 7: Halaman Pemberitahuan Pembuatan Direktori CPPcheck



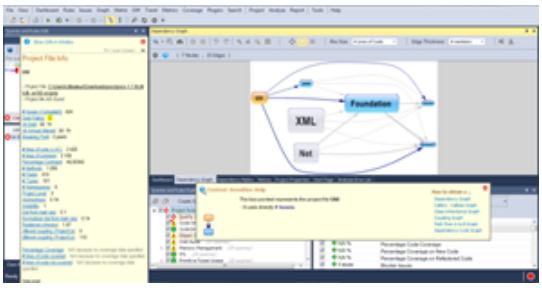
Gambar 8: Halaman Tidak Ditemukan Error CPPcheck

Uji coba dengan CPPdepend

Hasil dari Cppdepend untuk pertama kalinya adalah halaman dashboard yang merupakan tampilan keseluruhan dari pengukuran metric oleh Cppdepend yang terlihat pada Gambar dan menampilkan berbagai informasil mengenai aplikasi yang diukur yaitu SAIBA.

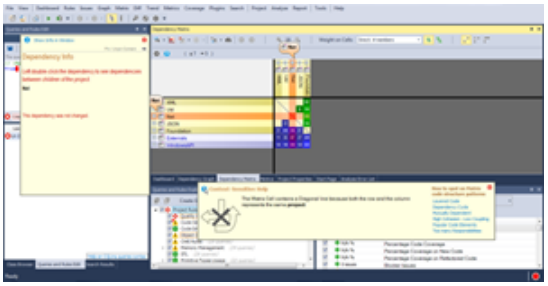


Gambar 9: Halaman Dashboard CPPdepend



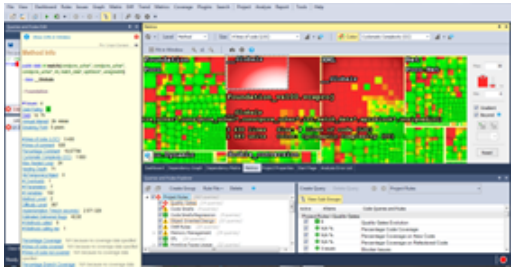
Gambar 10: Halaman Dependency Graph CPPdepend

Halaman dependency graph pada Cppdepend terdapat keterhubungan jalur antara table-table yang ada pada aplikasi SAIBA sehingga dapat dilihat bagaimana tabel-tabel tersebut saling berkaitan didalam program.



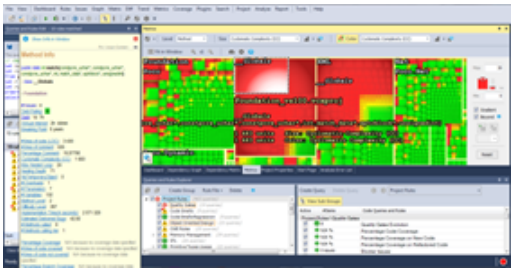
Gambar 11: Halaman Dependency Matrix CPPdepend

Halaman dependency matrix merupakan keterhubungan antara tabel-tabel yang di represintasikan menjadi bentuk matrix sehingga dapat dilihat bahwa bagaimana keterhubungan antara tabel-tabel yang berada pada program.



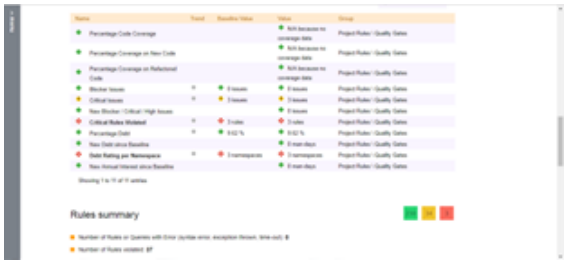
Gambar 12: Halaman Metric Metode Line Of Code CPPdepend

Halaman metric yang ditampilkan adalah berupa blok-blok dari program yang menggambarkan kekurangan dari program yang diukur. untuk blok program yang berwarna merah maka dapat diartikan bahwa program tersebut memiliki kekurangan dalam hal metode untuk line of code, Cppdepend juga memberikan informasi tentang line keberapa untuk program yang dianggap kurang baik tersebut. Untuk warna kuning maka dianggap oleh Cppdepend sebagai program yang memiliki kekurangan namun tidak terlalu merugikan didalam perjalanan program. Selanjutnya untuk yang berwarna hijau maka dianggap oleh Cppdepend program yang sudah baik dan tidak diperlukan perbaikan atau peningkatan. Sehingga untuk pemeliharaan atau maintenance sangat menghemat waktu dikarenakan dapat langsung melihat bagian yang dianggap kurang baik dan dapat langsung mencari solusinya.



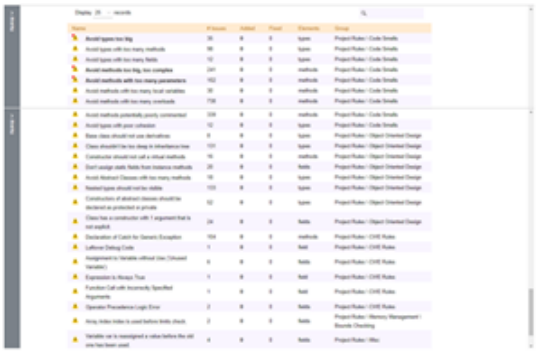
Gambar 13: Halaman Metric Metode Cyclomatic Complexity CPPdepend

Halaman metric metode cyclomatic complexity diatas memiliki keterangan yang sama dengan metode Line Of Code namun memiliki perbedaan dalam hal pengukuran bagian yaitu pada bagian Cyclomatic Complexity pada program yang diukur.



Gambar 14: Halaman Rules Summary CPPdepend

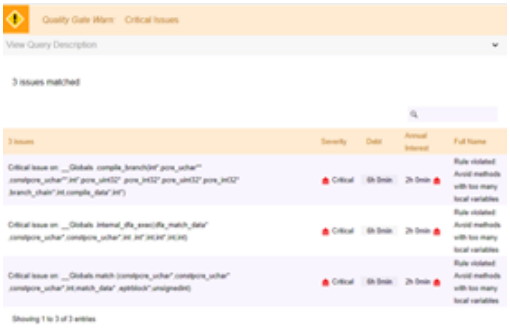
Halaman rules summary pada Cppdepend menampilkan pelanggaran pada rules yang menurut Cppdepend dapat diperbaiki untuk meningkatkan performa dari aplikasi SAIBA sehingga aplikasi bisa menjadi lebih baik.



Gambar 15: Halaman Tips Report CPPdepend

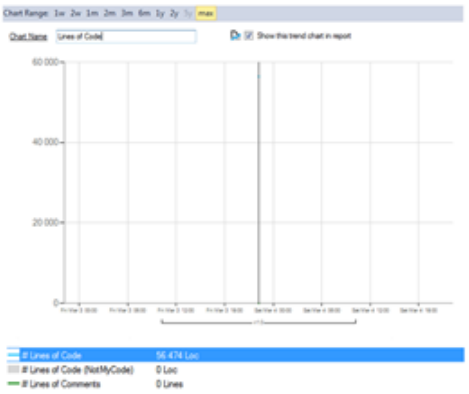
Halaman tips report Cppdepend memberikan berbagai tips-tips untuk solusi dari permasalahan yang telah ditemukan oleh Cppdepend sehingga bisa dijadikan bahan untuk pertimbangan dalam perbaikan.

Pada halaman tips report juga menampilkan jumlah kekurangan-kekurangan yang dapat diperbaiki (issues) , hal yang sudah ditambahkan (added), dan permasalahan yang telah di perbaiki (fixed) hal tersebut dapat dilihat dengan menampilkan jumlah permasalahan yang ada seperti Avoid types too big terdapat 35 issues 0 added 0 fixed, Avoid methods too big too complex terdapat 241 issues 0 added 0 fixed, dan Avoid methods with too many parameters terdapat 102 issues 0 added 0 fixed.



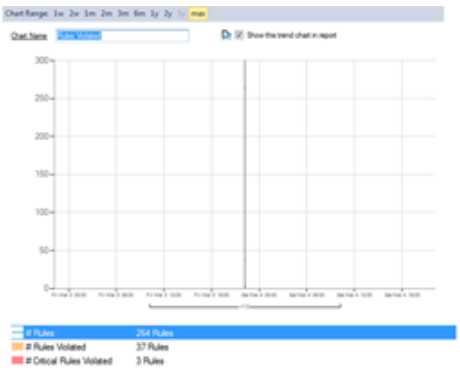
Gambar 16: Critical Issues CPPdepend

Halaman critical issues Cppdepend menampilkan 3 critical permasalahan yang dianggap perlu segera ditangani yang dikarenakan dapat menghambat berjalanannya program. Cppdepend juga memberi tahukan dimana letak permasalahan tersebut berada dengan memberikan potongan blok program yang dapat ditemukan dengan melihat kembali pada halaman matrix yang sudah ditampilkan oleh Cppdepend. Pada halaman critical issues juga menampilkan berapa lama jumlah waktu yang diperlukan untuk memproses debt dan annual interest.



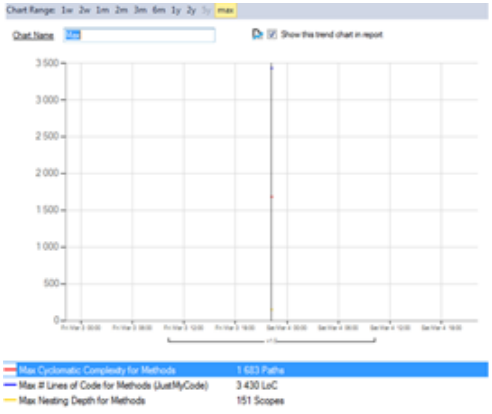
Gambar 17: Halaman Grafik Line of Code CPPdepend

Halaman grafik Line of Code Cppdepend menampilkan jumlah keseluruhan Line Of Code pada aplikasi yang diukur sehingga dapat ditentukan jumlah keseluruhan dari baris program. Pada halaman ini didapat jumlah total dari Line of Code sebesar 56.474 loc sehingga dapat diketahui bahwa terdapat 56.474 baris code yang ada pada aplikasi. Untuk Line of Code (not mycode) merupakan jumlah penambahan baris program, Lines of Comment adalah baris comentar yang ada pada aplikasi.



Gambar 18: Halaman Grafik rules violated CPPdepend

Halaman grafik rules violated pada Cppdepend dapat dilihat berapa banyak rules yang terdapat pada aplikasi, rules yang dilanggar, serta critical rules yang dilanggar. Hal ini merupakan summary dari hasil pengukuran sebelumnya sehingga grafik rules violated hanya menampilkan jumlah pada setiap bagian secara kesuluruhan. Pada halaman grafik rules violated terdapat 264 rules, 37 rules violated, dan 3 critical rules violated dari aplikasi yang diukur. Dari hasil tersebut dapat diketahui bahwa terdapat 264 aturan yang ada pada aplikasi yang diukur, 37 aturan yang kurang tepat atau dilanggar, dan 3 aturan kritis yang dilanggar.



Gambar 19: Halaman Max CPPdepend

Halaman max merupakan maximal dari penggunaan dalam setiap bagian yaitu seperti Cylomatic Complexity, Line Of Code, dan Nesting Depth dari aplikasi yang diukur. Sehingga didapatkan bahwa penggunaan maksimal setiap aplikasi dijalankan adalah sebagai berikut 1.683 jalur untuk Cyclomatic Complexity, 3.430 baris untuk Line of Code, 151 scope untuk Nesting Depth for Methods.

Gambar 20: Halaman Rata-Rata CPPdepend

Halaman Average (rata-rata) dari Cppdepend menampilkan jumlah rata-rata dari penggunaan Cyclomatic Complexity, Line Of Code, dan Nesting Depth dari aplikasi yang diukur. Sehingga didapatkan bahwa penggunaan rata-rata setiap aplikasi dijalankan adalah sebagai berikut 389 path untuk Cyclomatic Complexity, 833 line untuk Line of Code, 107 scope untuk Nesting Depth.

Analisis Keseluruhan

Proses pengukuran yang dilakukan dengan Cppcheck tidak ditemukannya kekurangan atau error yang ada dikarenakan pada bagian yang diukur oleh Cppcheck seperti algoritma, container, function, iterator tidak memiliki error sehingga Cppcheck memberikan pernyataan bahwa aplikasi tidak terdapat error dan tidak dilakukan save terhadap Cppcheck. Untuk pengukuran dengan menggunakan Cppdepend ditemukan berbagai kekurangan dalam Cyclomatic Complexity dan Line Of Code yang dapat diperbaiki dan ditingkatkan agar aplikasi SAIBA dapat menjadi lebih baik dari yang sebelumnya.

Cppdepend juga memberikan letak blok program yang perlu untuk diperbaiki sehingga untuk perbaikan dan pemeliharaan dapat dikerjakan langsung kepada pokok permasalahan, hal ini sangat membantu pihak yang ingin memperbaiki aplikasi SAIBA. Selanjutnya Cppdepend juga memberikan tips yang berfungsi sebagai bahan pertimbangan untuk maintenace sehingga pihak yang ingin melakukan maintenace dapat mempertimbangkan tips yang diberikan oleh Cppdepend untuk melakukan perbaikan. Untuk kekurangan yang kritikal Cppdepend memberikan tab khusus yang dapat dilihat pada gambar 14 pada bagian summary sehingga kekurangan ini merupakan hal yang mendesak dan merupakan hal yang patut segera untuk di perbaiki dan ditingkatkan untuk membuat aplikasi SAIBA dapat berjalan menjadi lebih baik.

Dalam hal kekurangan kritikal pada aplikasi SAIBA terdapat 3 yaitu dapat dilihat pada gambar 14 yang ada pada blok program _globals .compile_branch, _globals .internal_dfa_exec, dan _globals match menurut hasil tersebut dapat dilihat bahwa dalam blok program terdapat kekurangan dalam compile dan match yang berarti dapat disimpulkan bahwa dari ketiga kekurangan ini adalah hal yang menyebabkan kurang sempurnanya aplikasi SAIBA dalam hal konversi data yang dilakukan sehari-hari.

Penutup

Analisis yang dilakukan pada software aplikasi Sistem Akuntansi Instansi Berbasis Akrual (SAIBA) dengan tools Cppdepend dan Cppcheck telah berhasil dilakukan. Hasil dari analisis Cppcheck menampilkan bahwa tidak terdapat error atau kesalahan dalam algoritma, container, function, dan iterator sehingga dapat dikatakan bahwa aplikasi saiba memiliki pemrograman yang baik dalam 4 komponen seperti yang disebutkan sebelumnya. Hasil analisis dengan menggunakan Cppdepend terdapat beberapa kekurangan dalam hal sebagai berikut :

a. *Cyclomatic Complexity*
b. *Line Of Code*
c. *Rules*
d. *Quality gates*

Dari keempat hal diatas ditemukan paling banyak berbagai macam kekurangan dalam blok-blok program *Cyclomatic Complexity* dan *Line Of Code* yang menyebabkan kurang maksimalnya aplikasi SAIBA. Terdapat 3 critical error yang merupakan error yang harus di utamakan dalam perbaikan yang akan dilakukan agar aplikasi SAIBA

dapat menjadi lebih baik. Untuk memperbaiki hal-hal tersebut Cppdepend memberitahukan bagian blok program mana yang terdapat kekurangan sehingga dapat dilakukan perbaikan atau maintenance yang tepat sasaran sehingga dapat menghemat waktu dan biaya.

Dalam pengukuran ini hanya berfokus dengan menggunakan software metric Cppcheck dan Cppdepend yang merupakan pengukuran berfokus dalam 4 komponen utama dari Cppcheck lalu Cppdepend berfokus dalam Cyclomatic Complexity dan Line Of Code. Hal ini yang masih dapat ditingkatkan dengan mengukur komponen-komponen yang lebih detail lagi dengan menggunakan metode-metode yang berbeda. Sehingga didapatkan hasil yang sangat terperinci dan berbagai macam permasalahan yang tidak ditemukan oleh Cppdepend dan Cppcheck sehingga membuat aplikasi SAIBA menjadi aplikasi yang jauh lebih baik lagi.

Daftar Pustaka

[1] David Longstreet, "Function Point Analysis Training Course", University of Florida, 2011.

[2] Bambang Hariyanto, "Rekayasa Sistem Berorientasi Objek", Informatika, Bandung, 2004.

[3] Janner Simarmata, "Rekayasa Prangkat Lunak", ANDI, Yogyakarta, 2010.

[4] Moch Irfandi Susanto, Eko Darwiyanto & Gede Agung Ary Wisudawan, "Software Metric Measurement On Laravel Frame Work Implementation For Website Application", e-Proceeding of Engineering : Vol.2, No.3, 2015.

[5] Fathoni, "Pengukuran Kualitas Perangkat Lunak Berdasarkan Kompleksitas Menggunakan Menggunakan Metode Function Point", Jurnal Sistem Informasi (JSI). Vol 1 (2), 2009.

[6] Fredy Nendra Pranata, Fajar Pradana & Tri Astoto Kurniawan, "Pengembangan Sistem Perhitungan Kompleksitas Kode Sumber Berdasarkan Metric Halstead Dan Cylomatic Complexity", Prosiding seminar Nasional Teknologi dan Rekayasa Informasi ISSN (2540 - 9700), 2016.

[7] Kaushal Bhatt, Vinit Tarey & Pushpraj Patel, "Analysis Of Source Lines Of Code(SLOC) Metric", International Journal of Emerging Technology and Advanced Engineering, ISSN (2250-2459), Volume 2, 2012.

[8] Nahlah M.A.M.Najm, "Measuring Maintainability Index of a Software Depending on Line of Code Only", IOSR Journal of Computer Engineering, p- ISSN: 2278-8727 Volume 16, Issue 2, Ver. VII PP 64-69 www.iosrjournals.org., 2014.

[9] Charles Ikerionwu, " Cyclomatic Complexity as a Software Metric ", International Journal Of Academic Research, Vol. 2. No. 3, 2010.

[10] Meiliana, "Software Testing: Perhitungan Cyclomatic Complexity", available at : http://socs.binus.ac.id/2016/12/29/software-testing-perhitungan-cyclomatic-complexity/, 2016.