ANALISIS PROSES *MAINTENACE* APLIKASI (KASUS : APLIKASI *WEB* EMISI GAS RUMAH KACA PADA SEKTOR INDUSTRI DI KEMENTRIAN PERINDUSTRIAN)

Tien Fabrianti Kusumasari
Program Studi, Fakultas Rekayasa Industri, Telkom University
tienkusumasari@telkomuniversity.ac.id

***Abstrak—Software engineering* merupakan proses perubahan terhadap sebuah produk *software* yang bersifat *bug-fix,* penambaan fungsi baru, dan perbaikan kinerja sistem *software*, yang mana proses *maintenance* ini akan selalu diperlukan dalam semua organisasi yang didukung oleh sistem informasi. Pada penelitian ini akan membahas mengenai proses *software maintenance* dengan studi kasus pada aplikasi *web* emisi gas rumah kaca pada sektor industri di lingkungan kementrian perindustrian Indonesia. Metode yang digunakan dalam proses *maintenance* ini diadopsi dari *reengineering software* proses model dengan beberapa penyesuaian, yaitu *inventory analisis*, *reverse engineering*, data *restructuring*, dan *forward engineering*. Hasil dari penelitian ini adalah sebuah *new system* dengan penambahan fitur pelaporan secara periodik, proses verifikasi, dan penambahan *form* IPCC.**

Kata kunci—software maintenance*, *software maintenance process, web* emisi GRK, *software maintenance process model

I. PENDAHULUAN

Perubahan bisni yang cukup fluktuatif, organisasi (bisnis) memerlukan dukungan teknologi informasi untuk merespon berubahan tersebut[1].Dengan demikian permintaan perubahan terjadi pada fungsi bisnis dan teknologi informasi sebagai salah satu solusi untuk peningkatan daya saing organisasi. Kondisi ini akan menyebabkan *software* (perangkat lunak) dilakukan maintenance secara terus menerus[2]. *Software* komputer selalu berevolusi baik dari sisi ukuran maupun kompleksitas setiap saat. Perubahan tersebut terjadi ketika *error* pada *software* diperbaiki, ketika *software* beradptasi dengan lingkungan baru, ketika *customer* memerlukan beberapa perubahan dari sisi *fitur* dan fungsi, ataupun ketika *software* dilakukan *reengineering* untuk memperoleh keuntungan dalam konteks yang lebih *modern* [2].

Perubahan-perubahan pada *software* komputer tersebut disebut dengan istilah *software maintenance*. Berdasarkan kajian yang dilakukan oleh [3] menyimpulkan tentang pengertian *software maintenance* adalah perencanaan dan pelaksanaan aktivitas yang berkaitan dengan modifikasi produk *software* untuk meningkatkan kinerja, beradaptasi dengan produk atau lingkungan baru, memperbaiki kesalahan (*fault*) atau meningkatkan *maintainability* selama siklus hidup *software* (*software life cycle*). *Software maintenance* dapat dikategorikan sebagai adaptive, *corrective, dan perfective*[4].

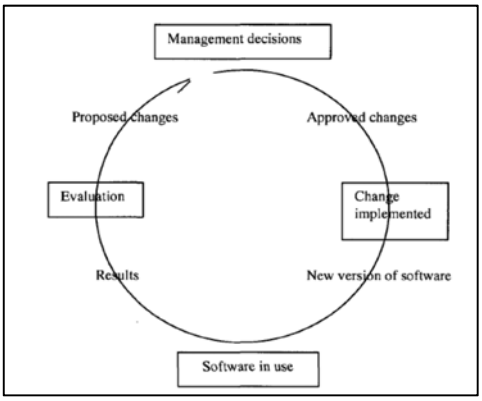
Adaptive merupakan perubahan yang terjadi akibat penyesuaian dengan kebijakan atau proses bisnis. *Corrective* merupakan perubahan *software* akibat perbaikan *defect, error*, atau meningkatkan kinerja *software*.Sedangkan *perfective* merupakan penambahan kebutuhan (*requirement*) dan fungsi baru.

Pada penelitian ini akan dibahas mengenai proses dan analisis *software maintenance* pada sebuah aplikasi *web* tentang *inventory* emisi gas rumah kaca pada sektor industri di kementrian perindustrian. Proses *maintenance* aplikasi tersebut bersifat *perfective* dan *corrective*, yaitu menambahkan beberapa fungsi baru dalam *legacy system* (aplikasi lama) serta memperbaiki *defect* yang masih belum tertangani pada fase pengembangan. Sistematika pada makalah ini adalah di awali dengan penyajian dasar teori mengenai proses-proses *software maintenance*, metode yang digunakan dalam proses *maintenance* studi kasus ini, dilanjutkan dengan analisis mengenai perubahan-perubahan yang telah dilakukan selama proses *maintenance* serta hal-hal yang harus di lakukan untuk proses *maintenance* selanjutnya.

II. *SOFTWARE MAINTENANCE PROCESS*

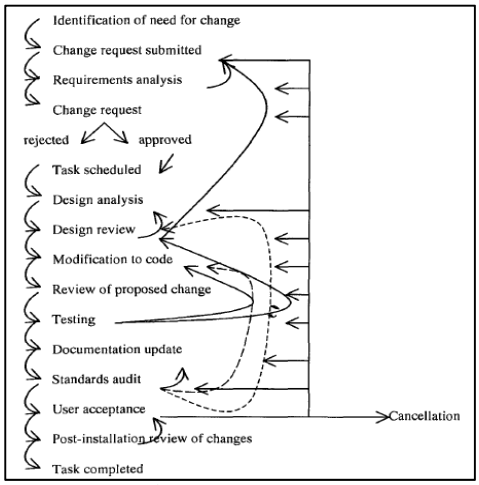
Karekteristik *software maintenance* berbeda dengan *software development* jika dilihat dari sisi usaha disetiap fase pengembangan. Berdasarkan [5]menyatakan bahwa usaha yang paling tinggi dilakukan dalam *software maintenance* terjadi pada fase analisis dan spesifikasi, sedangkan usaha tertinggi pada *software development* terjadi pada fase implementasi dan testing selama proses *software life cycle*. Proses *software maintenance* merupakan suatu hal yang kompleks, karena dilakukan pada sebuah produk perangkat lunak yang kompleks dan melibatkan kooperasi serta koordinasisehingga mempunyai porsi tertinggi dalam pembiayaan *software life cycle*[6]. [2]juga menyatakan bahwa proses *reengineering* membutuhkan waktu yang cukup besar, pembiayaan yang signifikan dan menyerap sumber daya selain menguras konsentrasi. Oleh karena itu diperlukan strategi dan metode dalam pelaksanaannya. Pada bab II ini akan diungkapkan mengenai berbagai model proses dalam *software maintenance*, yaitu Boehm's *model*, Osborn model, *iterative enhancement model*, *reuse oriented model*, *maintenance conscious life cycle model*, dan *software reengineering process model*.

Boehm's *model* (1983) merupakan model proses *software maintenance* berbasis model dan prinsip ekonomi. Boehm merepresentasikan bahwa proses *maintenance* merupakan sebuah siklus tertutup seperti pada Gambar 1 [5]. Pada model ini, keputusan manajemen menjadi pemicu proses *maintenance.* Sedangkan pengesahaan terhadap perubahan ditentukan dengan menentukan stategi dan evaluasi *cost-benefit* terhadap perubahan yang akan dilakukan. Selain hal tersebut, pengesahaan perubahan ini juga dilengkapi dengan penganggaran dan tipe pembiayaan sumber daya.



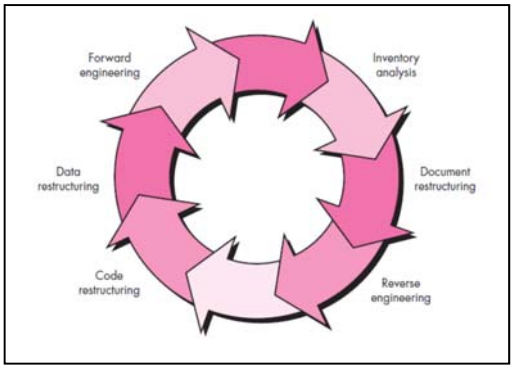
Gambar 1 *Boehm's Model*

Osborn's model [5] merupakan model *maintenance* yang dilakukan dengan iterasi secara kontinu pada siklus pengembangan perangkat lunak (seperti pada gambar), yang mensyaratkan *maintainability* di setiap fasenya. Model *maintenance* ini mengijinkan untuk membangun ulang produk *software* jika formal spesifikasi atau dokumentasi yang lengkap tidak ada. Beberapa kelemahan dari model ini adalah terdapat beberpa permasalahn teknis selama proses *maintenance* karena lemahnya manajemen komunikasi dan pengendalian. Oleh karena beberapa rekomendasi strategi yang disarankan adalah menambahkan kebutuhan maintenance dalam perubahan spesifikasi, menambahkan *software quality assurance* sebagai penjamin kualitas, memverifikasi tujuan *maintenance* yang akan dicapai, *review* kinerja sebagai *feedback* bagi manajer [5].



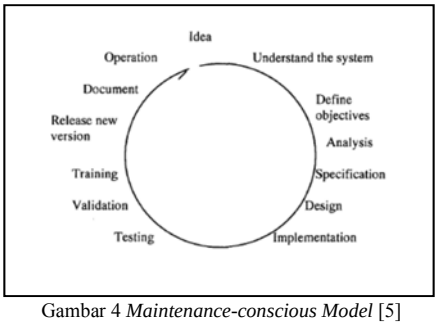
Gambar 2 *Boehm's Model*

Software reengineering process model [2], merupakan siklus aktivitas yang mana setiap aktivitas merupakan bagian dari proses sebelumnya. Model ini terdiri dari enam aktivitas seperti pada gambar 3, yaitu *inventory analysis*, *document restructuring, reverse engineering, code restructuring, data restructuring, forward engineering. Inventory analysis* merupakan analisis terhadap ketersediaan sumber daya yang ada dalam organisasi sebelum melakukan *reengineering*. *Legacy system* (sistem *software* yang ada) biasanya lemah dalam dokumentasi, oleh karena itu beberpa opsi dapat di pilih dan disesuaikan untuk menyusun dokumentasi dari *legacy system.* Fase selanjutnya adalah *reverse engineering*, yang merupakan sebuah proses analisis program dari *source code* menjadi abstraksi *high level* atau disebut dengan *design recovery*. Hasil dari *reverse engineering* tersebut adalah *design* struktur data, arsitektur, dan prosedur program dari *existing* program. Fase yang keempat adalah *restructuring code*, merupakan *refactoring source code* untuk menghilangkan pelanggaran struktur *source code* dengan menggunakan *refactoring tool*. Fase kelima adalah adalah data *restructuring*, merupakan proses memperbaiki struktur data karena arsitektur data yang baik akan mempengaruhi struktur program dan algoritma secara keseluruhan sistem. *Forward engineering*, merupakan proses pembangunan sistem *software* sesuai dengan tujuan *maintenance*.



Gambar 3 *Software* Reengineering Process Model [2]

Maintenance-conscious model merupakan *generic model* untuk *software maintenance* yang merupakan siklus tertutup. Siklus dalam model tersebut terdiri dari aktivitas sebagai berikut: pemahaman *legacy system*, menentukan tujuan *maintenance*, analisis, spesifikasi, *design*, implementasi, *testing*, validasi, *training, release* versi sitem baru, dokumentasi, operasi, ide baru.



Gambar 4 *Maintenance-conscious Model* [5]

III. METODE IMPLEMENTASI *SOFTWARE MAINTENANCE*

Metode yang digunakan dalam penelitian ini adalah dengan mengadopsi *Software Reengineering Process Model* dari [2], yang di modifikasi sesuai dengan kondisi tim serta lingkungan pengembang. Pada studi kasus penelitian ini, aplikasi *existing* telah mempunyai dokumentasi yang cukup dan dikembangkan oleh organisasi pengembang yang sama. Namun susunan tim pengembang sedikit berbeda, beberpa orang melakukan pengembangan pada *legacy system* maupun pada *new system*. Tim pengembang melakukan koordinasi dengan menggunakan *software collaboration* karena terdistribusi di berbagai lokasi, namun masih dapat melakukan koordinasi secara *face to face* dengan intensitas yang rendah.

Proses software maintenance pada penelitian ini dibagi menjadi empat aktivitas, yaitu *inventory analysis, reverse engineering, data restructuring,* dan *forward engineering*. *Inventory analysis* merupakan aktivitas dalam mempersiapkan sumber daya untuk proses *maintenance*, yaitu dengan mempersiapkan anggota tim, rencana pengembangan, serta dokumentasi *legacy system*. Aktivitas kedua adalah *reverse engineering*, merupakan aktivitas proses aktivitas pemahaman terhadap proses bisnis, arsitktur, struktur *database*, dan struktur *source code* dari *legacy system*. Aktivitas kedua ini diperlukan mengingat beberapa anggota tim tidak terlibat pada pengembangan *legacy system*.*Data restructuring* merupakan perubahan struktur data *legacy system* supaya dapat ditambahkan fungsi-fungsi baru sesuai dengan permintaan customer. Fase yang terakhir adalah *forward engineering*, yang mana merupakan proses pengembangan *legacy system* menjadi sistem baru sesuai dengan lingkup proyek *maintenance*.

Pada fase forward engineering mengadopsi *collaboration* model dari ,terdiri dari beberpa aktivitas yaitu *requirement* (identifikasi kebutuhan), *modelling* (pemodelan), construction (penulisan kode program), dan deployment (pemasangan). *Requirement* dilakukan dengan wawancara secara langsung dengan *business expert* serta dokumen busnis yang menjadi standar acuan. Fase *modelling* / pemodelan merupakan fase dimana memodelkan sistem baik berupa antar muka sistem, struktur basis data, dan arsitektur sistem yang akan dibuat melalui *collaboration tool* (*collaborative mockup design*, *email*, *online repository*, dan *chatting*). Fase *construction* meruakan fase implementasi dalam *source code* dan *testing*. *Tool* yang digunakan dalam fase *construction* tersebut adalah *software configuration management* (SCM), *email*, *chatting*, *bug trackingsoftware*, dan *online repository*.Fase terakhir adalah *deployment*, yang dilakukan dengan *online setting*.

IV. INVENTORY ANALYSIS

A. Legacy System

Aplikasi sistem informasi inventarisasi Gas Rumah Kaca (GRK) yang selanjutnya disebut dengan sistem existing merupakan sistem aplikasi yang berbasis web. Sistem aplikasi ini meberikan inforamsi mengenai data-data emisi GRK pada sektor industri di seluruh indonesia. Dalam aplikasi ini industri terdiri dari 8 jenis yaitu Industri Baja, Industri Pulp & Kertas, Industri Kimia, Industri Tekstil, Industri Pupuk, Industri Gula Rafinasi, Industri Minyak Goreng, dan Industri Keramik. Emisi GRK terdiri dari emisi gas CO2, gas CH4, dan gas N2O. Perhitungan emisi gas CO2 dilakukan dengan 2 metode, yaitu metode Neraca Massa dan Energi (NME) yang mana dalam IPCC disebut perhitungan *tier* 2 dan *tier* 3. Serta metode IPCC, yaitu metode perhitungan IPCC dengan menggunakan *tier* 1.Sedangkan perhitungan emisi gas CH4 dan gas N2O dilakukan dengan metode IPCC (*tier* 1). Masing-masing hasil perhitungan emisi gas CO2, CH4, dan N2O dikonversi menjasi ekuivalen CO2 sehingga dapat dijumlahkan yang mewakili total emisi GRK.

Data emisi GRK disajikan berdasarkan 3 klasifikasi peralatan proses pada pabrik yang paling berpotensi menghasilkan emisi GRK, yaitu peralatan penghasil energi, peralatan proses, dan peralatan pada unit pengolahan limbah. Peralatan pabrik penghasil energi terdiri dari boile, *furnace*, dan pembangkit listrik. Peralatan proses terdiri dari kiln pereduksi, kiln kalsinasi, kiln pengering, *furnace* listrik elektroda karbon, *furnace* listrik induksi, reformer, *cracking*, pirolisis, dan CO2 absorber. Sedangkan peralatan unit limbah terdiri dari incinerator, flare, landfil, dan Instalasi Pengolahan Air Limbah (IPAL).

Sistem aplikasi ini memberikan pelaporan individu (untuk masing-masing industri) serta pelaporan secara keseluruan mengenai jumlah emisi GRK yang dikeluarkan. Jumlah emisi GRK dalam sistem ini meliputi emisi gas CO2 , emisi gas CH4, dan emisi gas N2O. Masing-masing emisi gas CO2, CH4, dan N2O tersebut di hitung dengan metode IPCC (IPCC *tier* 1) maupun NME (IPCC *tier* 2 dan *tier* 3). Hasil dari perhitungan emisi masing-masing gas tersebut di konversi dalam ekuivalen CO2 yang kemudian dijumlahkan dan disebut dengan jumlah emisi GRK (dengan satuan ekuivalen CO2).

B. New System

Berdasarkan fitur yang ada dalam *legacy system* maka data yang ada dalam aplikasi tersebut bersifat sekali pakai atau sekali isi karena merupakan data inventarisasi. Sedangkan sistem informasi monitoring emisi GRK yang selanjutnya disebut dengan *new system*, mempunyai fitur yang sama persis dengan aplikasi *existing* dengan tambahan pengisian data dapat dilakukan secara periodik dan data-data lama tersimpan dan dapat digunakan sebagai information dashboard untuk program penurunan emisi GRK. Dengan adanya fitur pengisian data secara periodik, maka diperlukan sebuah proses untuk melakukan verifikasi data dari pihak kementerian yang diimplementasikan sebagai fitur verifikasi. Selain kedua fungsi tambahan tersebut, pada *maintenance* ini juga ditambahkan *form* IPCC energi, *form* IPCC *waste,* dan *form* IPCC proses untuk masing-masing *tier* (*tier* 1, *tier* 2, dan *tier* 3) sesuai dengan standar pelaporan IPCC 2006.

Secara garis besar usulan sistem dapat dilihat pada Tabel 1, pada tabel tersebut yang diberi warna merah merupakan tambahan fitur untuk sistem baru. Selain itu sistem aplikasi yang baru juga akan melakukan perbaikan *view* dan informasi mengenai pelaporan individu dan pelaporan total.

TABEL 1
FUNGSIONAL LEGACY SYSTEM DAN NEW SYSTEM

No	Menu	Fitur	Diskripsi
1	Pabrik	Master Data Pabrik	View semua data perusahaan yang telah di *input*kan
		Input Data	Meng*input*kan data-data perusahaan, peralatan pabrik,
		Update Data	Melakukan update data perusahaan dan peralatan pabrik
		Delete Data	Menghapus data pabrik
		Tambah Alat	Menambah peralatan
		Report	Pelaporan individu setiap pabrik / perusahaan
		Upload laporan	Upload file pelaporan (*.doc/ *.docx)
		Download laporan	Download file pelaporan (*.doc/ *.docx)
2	Manajemen pelaporan	Tambah Periode	Meng*input*kan periode waktu untuk pelaporan emisi (periode *input* data) dan semua data pabrik dalam satu periode tertentu
		Update periode	Update data-data dalam satu periode tertentu
		View pelaporan	Resume pelaporan laju emisi berdasarkan periode waktu
		Trend	Grafik trand perubahan laju emisi secara periodic
3	Peralatan Pabrik	*Input* data peralatan	Mengisi data-data peralatan pabrik (sesuai dengan peralatan yang diperlukan)
		Update peralatan	Mengubah data peralatan
		Delete peralatan	Menghapus peralatan pada sebuah pabrik
		View peralatan	Menampilkan peralatan pabrik
4	Approve	Pemberian status	Memberikan pengesahan terhadap data emisi yang dilaporkan (di*input*kan ke sistem) oleh industry
		Master data approve	View list pabrik , periode, dan status approve
5	Report	View	Menampilkan informasi rekapitulasi data emisi GRK secara keseluruahn
		Tranding	Grafik perubahan laju emisi secara periodic
6	User	Create user	Menambah user
		Manage user	Memberikan hak akses per jenis industry
		Manage user	Memberikan hak akses per industry (penambahan role dalam aplikasi)
		Update user	Mengubah data user
		Delete user	Menghapus user
7	Log Aktivitas	View	Informasi mengenai tanggal, waktu, dan jenis aktivitas yang telah dilakukan oleh semua user dalam aplikasi

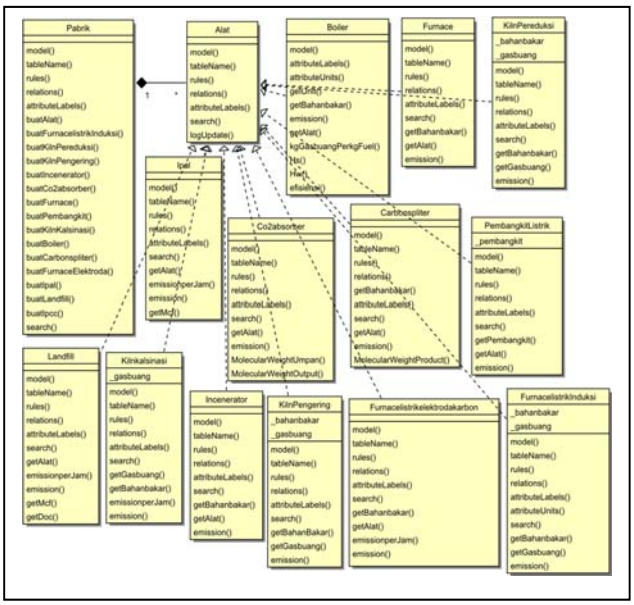
Ket : Warna → *new system*

V. REVERSE & FORWARD ENGINEERING

A. *Legacy System*

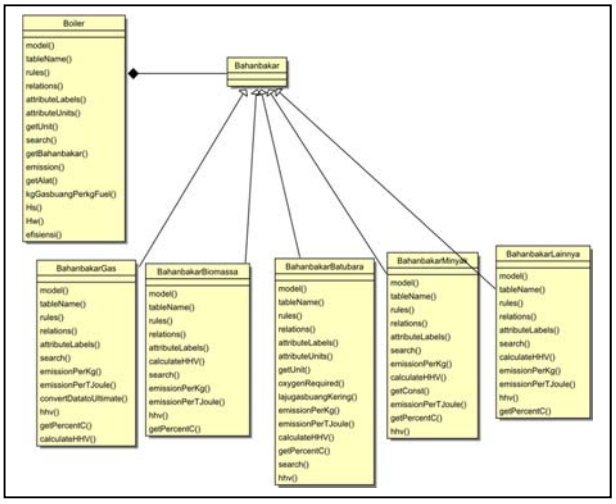
Struktur utama program dalam aplikasi *existing* terdiri dari *class* pabrik, *class* bahan bakar, dan *class* alat. *Class* pabrik memiliki relasi *one-to-many* dengan beberapa *class* Alat. Sedangkan *class Boiler* merupakan salah satu jenis (*subclass*) dari Alat, begitu pula dengan *Furnace*, Kiln Pereduksi, Co2 absorber, Carbon spliter, Pembangkit Listrik, Landfill, Kiln kalsinasi, *Incenerator*, Kiln Pengering, *Furnace* listrik elektroda karbon, dan F*urnace* listrik Induksi. Semua itu merupakan jenis-jenis alat yang dapat dimiliki oleh sebuah pabrik. Struktur class Pabrik diilustrasikan pada Gambar .

Selain informasi spesifik *boiler*, *class Boiler* juga mengandung *class* Bahan bakar. *Class* Bahan bakar dapat berupa salah satu dari class Bahan bakar Biomassa, Bahan bakar Batu bara, Bahan bakar Minyak, Bahan bakar Gas, atau Bahan bakar Lainnya. *Class* BahanBakar ini juga terdapat pada *class Furnace*, Kiln Pereduksi, Kiln kalsinasi, *Incenerator*, Kiln Pengering, *Furnace* listrik elektroda karbon, dan *Furnace* listrik Induksi. Struktur *class* Bahanbakar ini diilustrasikan pada Gambar .Pembangkit Listrik memiliki *field* berupa Jenis pembangkit. Informasi yang spesifik terhadap jenis pembangkit tersimpan dalam *class* Pltd, Pltg, atau Pltu yang dikandung oleh Pembangkit Listrik. *Class* Bahan bakar tersimpan di masing-masing *class* tersebut. Struktur class Pembangkit Listrik seperti pada Gambar 5 dan Gambar 6.

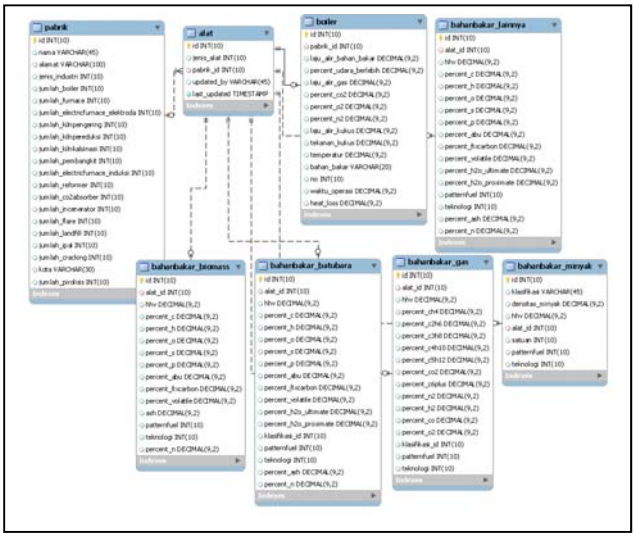


Gambar 5 *Class* Diagram Pabrik (1)

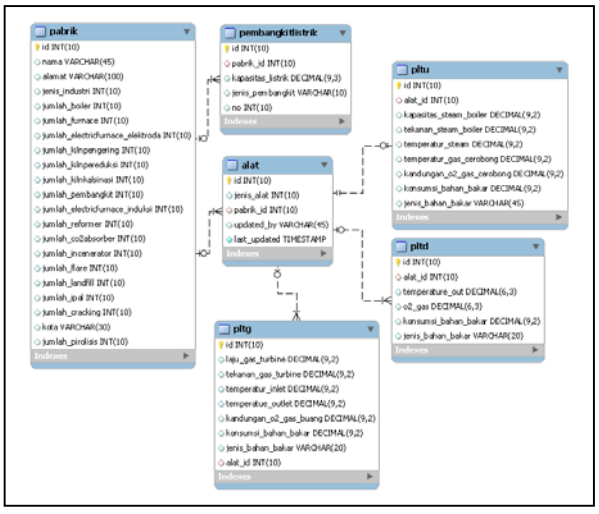
Sedangkan struktur data *legacy system* utama *legacy system* adalah tabel pabrik yang berelasi dengan table-tabel sebagai berikut: alat, pembangkit listrik, co2absorber, pirolisi, carbonfilet, *cracking*, *reformer* (Gambar 7, Gambar) ipcc, ipal, *incinerator*, kiln pengering, kiln kalsinasi, kiln pengering. Sedangkan table alat berelasi dengan table pltu, pltd, pltg, boiler, bahan bakar batubara, bahan bakar minyak, bahan bakar_gas, bahan bakar_biomassa, bahan bakar lainnya, *furnace* listrik induksi, *furnace* listrik elektroda karbon. Sedangkan tabel gas buang berelasi dengan *furnace* listrik induksi,kiln pengering, kiln pereduksi



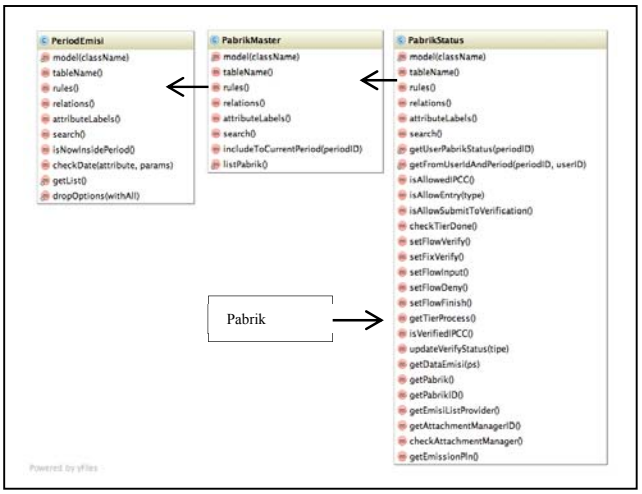
Gambar 6 *Class* Diagram Bahan Bakar (2)



Gambar 8 Struktur Data Pabrik dan Bahan Bakar



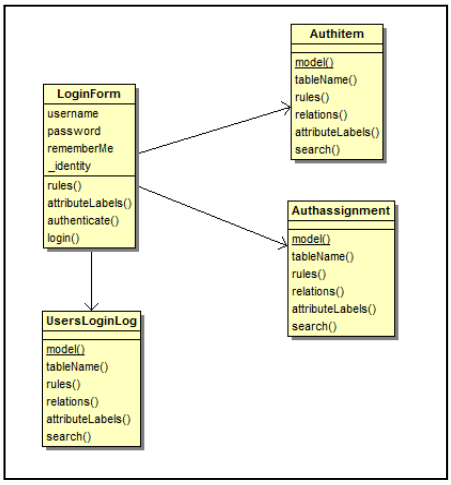
Gambar 7 Struktur Data Pabrik dan Pembangkit



Gambar 9 *Class* Diagram Periode Emisi

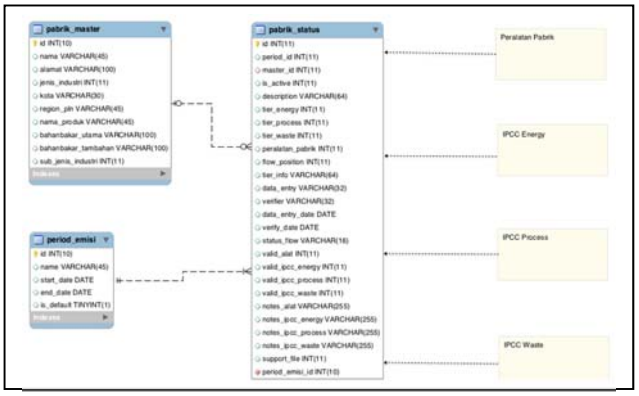
B. *New System*

Perubahan struktur pemrograman *new system* terjadi pada struktur periode pelaporan, yang ditunjukkan pada Gambar . *Class* Periode Emisi mengandung *class* Pabrik Master, sedang *class* Pabrik Master mengandung *class* Pabrik Status. Sedangkan *class* Pabrik Status mengandung class Pabrik (dari struktur *legacy system*). Proses autentifikasi dilakukan ketika user terdaftar melakukan login ke sistem. Kelas Login Form memakai kelas Authitem untuk memastikan user adalah *user* yang benar terdaftar dan memakai kelas *authassignment* untuk melihat *assignment* dari *user*. Setelah *user* diperbolehkan *login*, maka kelas *Login Form* akan memanggil kelas *User Login Log* untuk mencatat riwayat login user tersebut. Struktur autentifikasi dan *user log* ini diilustrasikan pada Gambar 9.

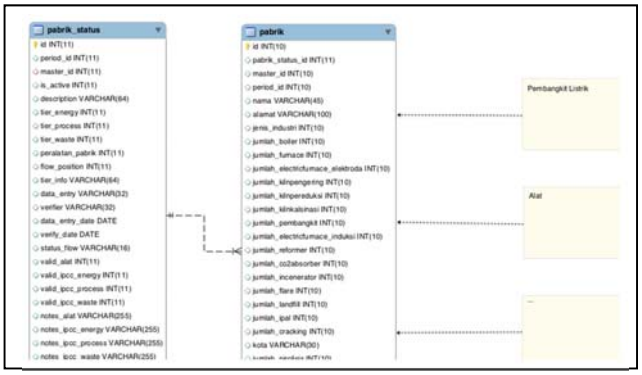


Gambar 10 *Class* Diagram Autentifikasi dan *User log*

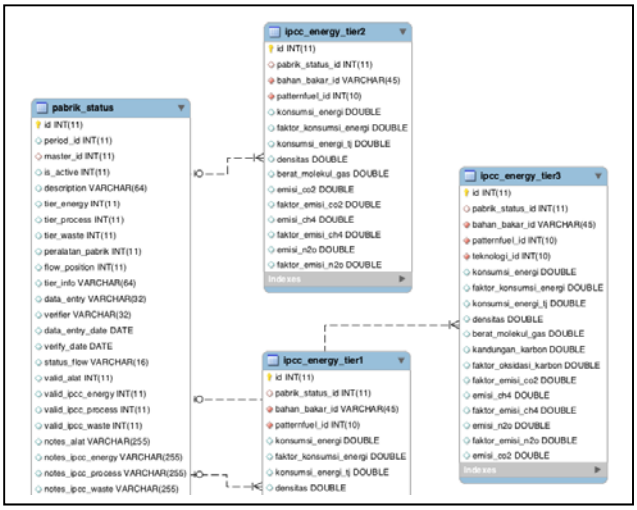
Perubahan struktur data pada *new system* adalah dengan menambahkan konsep periode pelaporan serta penambahan form pengisian data IPCC dalam 3 pilihan *tier* (energy, proses, dan limbah). Struktur data periode pelaporan dapat digambarkan dalam Entitas *Relationship* Diagram (ERD) pada Gambar dan Gambar . Tabel periode emisi berelasi dengan pabrik status, sedangkan table pabrik status berelasi dengan table pabrik *master*, table pabrik (*legacy system*), table-tabel pada IPCC energi, table-tabel IPCC *waste*, dan table-tabel IPCC proses. Untuk relasi tabel pabrik dengan tabel-tabel yang berisi peralatan dan bahan bakar menggunakan struktur data yang sama dengan *legacy system*.



Gambar 11 ERD Periode Emisi

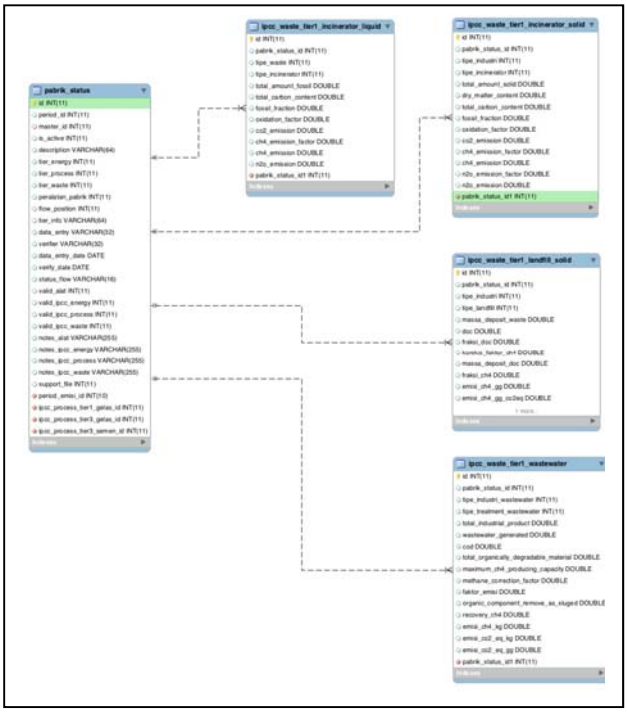


Gambar 12 ERD Pabrik Status dan Pabrik

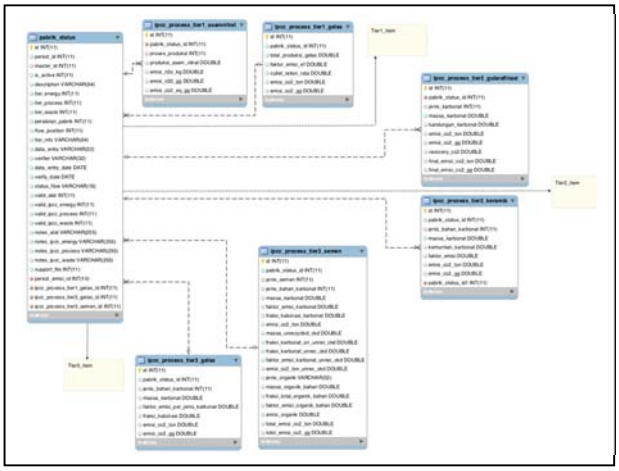


Gambar 13 ERD IPCC Energy

Sedangkan relasi antar tabel pada *form* IPCC *energy*, terjadi antara tabel pabrik status dengan ipcc *energy tier* 1,ipcc *energy tier* 2, dan ipcc *energy tier* 3 (Gambar 13). Hubungan antar tabel pada form IPCC waste diilustrasikan pada Gambar 14, yaitu tabel pabrik status berelasi dengan tabel ipcc waste *tier*1 *incinerator liquid*, tabel ipcc *waste tier* 1 *incinerator solid*, tabel ipcc waste *tier* 1 *land field solid*, dan tabel ipcc *waste tier1 wastewater*. Sedangkan Gambar 15 menunjukkan relasi antara tabel pabrik status dengna tabel-tabel jenis proses dari *tier* 1, *tier* 2, dan *tier* 3.



Gambar 14 ERD IPCC *Waste*



Gambar15 ERD IPCC Proses

VI. ANALISIS

Perubahan yang dilakukan dalam proses *maintenance* pada studi kasus ini berkaitan dengan penembahan fungsional *legacy system*. Penambahan fungsi tersebut yaitu menambahkan fitur pengisian data emisi secara periodik serta menyimpan jejak data di setiap periode yang ada, proses verifikasi data emisi yang telah di*input*kan ke dalam sistem di setiap periodenya, pembuatan form pengisian data emisi sesuai dengan pelaporan IPCC 2004. Implementasi fungsi pelaporan periodik dan penambahan form IPCC, adalah dengan menambahkan struktur data tambahan seperti pada Gambar . Pada struktur data tambahan tersebut memecah tabel pabrik pada *legacy system* menjadi tabel pabrik master dan tabel pabrik status. Kemudian selain itu juga ditambahkan tabel periode emisi untuk menyimpan data periode yang berelasi dengan tabel pabrik status. Tabel pabrik master berelasi satu ke banyak dengan tabel pabrik status, yang berfungsi untuk menyimpan data status pabrik-pabrik dalam setiap periode pelaporan.Tabel pabrik status ini berelasi berfungsi untuk menyimpan data pabrik yang ada pada periode aktif dan menyimpan status *flow* dokumen data emisi, serta semua form IPCC dan peralatan di setiap periode pelaporan.

Fungsi verifikasi data ditambahkan pada struktur program dengan dan penyimpanan status pada tabel pabrik status. Selain itu dari sisi proses bisnis fungsi verifikasi data emisi yang ditambahkan akan menambah *actor*/hak akses, yaitu verifikator yang belum ada dalam *legacy system*.

VII. KESIMPULAN DAN PENGEMBANGAN

Perubahan aplikasi pada tahap *maintenance* ini merupakan perubahan yang bersifat *corrective* dan *perfective*, karena berkaitan dengan memperbaiki *bug* pada tahap pengembangan yang masih tersisa serta penambahan fungsi monitoring pelaporan secara periodik dan penambahan *form* IPCC. Secara struktur data dan struktur program penambahan periode ini akan mengubah arsitektur *legacy system*, sehingga *new system* dapat diisikan data emisi secara periodik dan jejak setiap periode data tersebut dapat dilaporkan. Selain itu perubahan terjadi pada proses bisnis verifikasi data, yaitu penambahan fungsi verifikasi pada *legacy system*. Dan yang terakhir adalah penambahan form IPCC *energy*, IPCC *waste*, dan IPCC proses untuk *tier*1, *tier*2, dan *tier*3 sesuai dengna form standar internasional IPCC 2006. Namun *new system* belum melakukan perhitungan hasil dari pemrosesan *form-form* IPCC tersebut, hal tersebut memang keinginan dari *user* untuk memecah proses *maintenance* menjadi dua tahap.

New system dapat dikembangkan menjadi aplikasi yang sempurna dari sisi fungsi, dengan menambahkan proses perhitungan dari hasil *input form-form* IPCC. Selain itu sebaiknya *new system* tersebut diujicobakan ke industri pengguna sehingga diperoleh masukan untuk jenis-jenis *field* yang sesuai dengan kondisi lapangan di masing-masing industri.

DAFTAR PUSTAKA

[1] E. Turban, E. McLean and J. Wetherbe, Information Technology for *Management*: Transforming Organizations in the Digital Economy, Virginia: John Wiley & Sons, 2004.

[2] R. S. Pressman, *Software* engineering: a practitioner's approach, McGraw-Hill Higher Education, 2010.

[3] U. Kuhlmann, "Maintenance Activities in *Software* Process Models:Theory and Case Study Practice," University of Koblenz Landau, 2003.

[4] M. C. d. M. Filho, "Agile processes for the maintenance cycle," IBM Corporation, 2012.

[5] P. Grubb and A. A. Takang, *Software* Maintenance: Concepts and Practice, second ed, Singapore: World Scientific, 2003.

[6] I. Podnar and B. Mikac, "*Software* Maintenance Process Analysis Using Discrete-Event Simulation," in *Software* Maintenance and Reengineering, 2001. Fifth European Conference on , Lisbon, 2001.