

**PERANCANGAN MIDDLEWARE UNTUK MENGHUBUNGKAN
SISTEM INFORMASI DAGANG DENGAN APLIKASI
E-FAKTUR DARI DIREKTORAT JENDERAL PAJAK**
**Designing Middleware to Connect Trade Information System with e-Faktur
Application from Indonesia Taxation Authority**

Tony Wijaya¹

Jurusan Sistem Informasi STMIK Pontianak
Jl. Merdeka No. 372 Pontianak, Kalimantan Barat
E-mail: mail.tonywijaya@gmail.com

Diterima: 14 Desember 2016/ Disetujui : 27 Desember 2016

ABSTRACT

Since July 1st, 2016, every entrepreneur in Indonesia must implement e-Faktur application from Indonesia Taxation Authority. The objective is to ease entrepreneurs to report the company's tax and provide more detailed information about product and service offered by entrepreneurs. For most of companies that already have information system, implementation of e-Faktur application means they have to do double work to input operational data into two different applications. The objective of this research is to design a middleware that can connect trade information system so it can export data to e-Faktur application. This middleware is designed using Agile Methodology with Extreme Programming approach which value working software feature. Tested with black box methodology to ensure exported data is correct. It has feature to adjust decimal so it will be very accurate. It also very easy to operate, save operational time, increase tax accuracy dan ease entrepreneur to report tax to government.

Keywords: *Middleware, Trade Information System, e-Faktur.*

ABSTRAK

Penerapan aplikasi e-Faktur dari Direktorat Jenderal Pajak sudah diwajibkan kepada setiap pengusaha di Indonesia secara nasional sejak 1 Juli 2016. Tujuannya adalah untuk mempermudah pengusaha melaporkan pajak perusahaannya dan memberikan informasi lebih detil tentang barang atau jasa yang dijual. Bagi sebagian besar perusahaan yang sudah memiliki sistem informasi, penerapan aplikasi e-Faktur menyebabkan pengusaha harus melakukan entri data operasional yang sama ke dua aplikasi yang berbeda. Tujuan penelitian ini menghasilkan middleware untuk menjembatani sistem informasi dagang sehingga dapat melakukan ekspor data dari sistem berjalan ke aplikasi e-Faktur. Pendekatan dalam perancangan menggunakan metode Agile dengan pendekatan Extreme Programming yang lebih mengedepankan tercapainya fitur yang akan dibangun. Pengujian dilakukan dengan metode black box untuk memastikan data hasil ekspor sudah sesuai. Memiliki fitur penyesuaian desimal sehingga tidak akan terjadi selisih nilai pembulatan. Middleware ini memberikan kemudahan dalam pengoperasian, menghemat waktu operasional, meningkatkan akurasi nilai pelaporan pajak dan mempermudah pengusaha untuk melaporkan pajak kepada pemerintah.

Kata kunci : *Middleware, Sistem Informasi Dagang, e-Faktur.*

1. PENDAHULUAN

[1] E-Faktur atau Faktur Pajak yang berbentuk elektronik merupakan faktur pajak yang dibuat melalui aplikasi atau sistem elektronik yang ditentukan dan/atau disediakan oleh Direktorat Jenderal Pajak (DJP). Untuk menerapkan pembuatan e-Faktur ini, DJP telah menyediakan aplikasi yang dapat

diinstall di perangkat komputer Pengusaha Kena Pajak (PKP) dan e-Faktur ini otomatis terhubung ke program e-SPT, sehingga akan memudahkan PKP dalam membuat SPT Masa Pajak Pertambahan Nilai (PPN) secara elektronik menggunakan program e-SPT. Penerapan ini secara nasional dilakukan mulai tanggal 1 Juli 2016.

Latar belakang DJP membuat aplikasi ini adalah karena memperhatikan masih terdapat penyalahgunaan faktur pajak, diantaranya wajib pajak non PKP yang menerbitkan faktur pajak padahal tidak berhak menerbitkan faktur pajak, faktur pajak yang terlambat diterbitkan, faktur pajak fiktif, atau faktur pajak ganda. Selain itu karena beban administrasi yang begitu besar bagi pihak DJP sehingga suatu sistem elektronik untuk faktur pajak dipandang sangat memberikan efisiensi bagi DJP maupun PKP itu sendiri. Jika berbicara mengenai manfaat, dalam dunia modern tentu semua aplikasi berbentuk elektronik sangat memberikan efisiensi bagi penggunaannya. Pengguna akan merasa nyaman baik dalam hal proses pekerjaan maupun penyimpanan hasil pekerjaan. Penerbitan Faktur Pajak tidak lagi membutuhkan tanda tangan basah karena Faktur pajak elektronik ini menggunakan tanda tangan digital (digital signature) berbentuk QR code, kemudian tidak ada kewajiban untuk mencetak faktur pajak, serta aplikasi ini merupakan satu kesatuan dengan pelaporan Surat Pemberitahuan (SPT) yang selama ini dilaporkan melalui e-SPT.

Sedangkan bagi DJP melalui aplikasi e-Faktur ini kita makin mudah melakukan pengawasan dengan adanya proses validasi Pajak Keluaran - Pajak Masukan (PK-PM), adanya data lengkap dari setiap faktur pajak serta meminimalisir proses penyimpanan dokumen. E-Faktur mempermudah pelayanan karena akan mempercepat proses pemeriksaan, pelaporan, dan pemberian nomor seri faktur pajak. Selain itu juga sistem berbasis elektronik ini akan meminimalkan penyalahgunaan penggunaan faktur pajak oleh perusahaan fiktif atau pihak yang tidak bertanggung jawab sehingga potensi pajak yang hilang menjadi sangat kecil. Melalui e-Faktur DJP berharap dapat mengatasi permasalahan dalam administrasi PPN sehingga penerimaan pajak dari sektor PPN dapat semakin optimal, selain bagi PKP dapat menjalankan usahanya menjadi jauh lebih baik. Sinergi antara DJP dan Wajib Pajak diharapkan mampu membangun sistem perpajakan yang lebih baik di masa depan.

Bagi sebagian besar perusahaan yang telah memiliki sistem informasi, penerapan aplikasi e-Faktur memberikan kerja ekstra pada bagian operasional. Data yang harus diinput ke aplikasi e-Faktur sama dengan data operasional pada sistem informasi yang sedang berjalan. Hal ini berarti pihak perusahaan mengerjakan pekerjaan yang sama sebanyak dua kali. Di samping itu, untuk dapat menggunakan aplikasi e-Faktur dengan baik tentunya membutuhkan pelatihan pula. Hal ini bisa menyebabkan dua kemungkinan pada perusahaan. Kemungkinan yang pertama adalah beban kerja karyawan bertambah dengan mempelajari aplikasi e-Faktur. Sedangkan kemungkinan kedua adalah bertambahnya karyawan baru yang dikhususkan untuk melakukan pengolahan data pada aplikasi e-Faktur. Baik kemungkinan pertama ataupun kedua, tidak ada efek positif bagi perusahaan. Maka dari itu, penulis mencoba membuat sebuah perangkat lunak yang bertindak sebagai “jembatan” antara sistem informasi dagang yang telah dipakai oleh perusahaan dengan aplikasi e-Faktur dari DJP. Perangkat lunak ini dinamakan middleware.

Middleware merupakan sebuah perangkat lunak yang memberikan sebuah layanan antar-aplikasi di luar layanan yang dapat diberikan oleh sebuah sistem operasi. Tujuan middleware adalah untuk memudahkan baik pengembang aplikasi maupun pemakai untuk berkomunikasi antara dua atau lebih aplikasi yang berbeda. Komunikasi yang terjalin antara dua atau lebih aplikasi melalui middleware menyebabkan kompleksitas antara aplikasi-aplikasi tersebut berkurang.

Penelitian yang terkait dengan middleware adalah sebuah jurnal internasional dengan judul “A Comprehensive Study of Advancement of Electrical Power Grid and Middleware Based Smart Grid Communication Platform”. [2] Dalam penelitian tersebut, disebutkan bahwa middleware adalah sebuah software yang dapat mengatasi heterogenitas dan kompleksitas dalam sistem-sistem yang terdistribusi. Penelitian ini membahas pentingnya pembacaan data dari mesin pembangkit listrik secara digital. Pembacaan data ini melibatkan sebuah *smart grid* yang menggantikan peralatan konvensional yang non-digital yang mempunyai komponen yang sangat banyak dengan fungsi yang berbeda pula. Jadi alat ini memiliki kompleksitas dan heterogenitas yang sangat tinggi. Penulis tersebut membuat sebuah *smart grid* yang merupakan middleware yang dapat dapat menghilangkan heterogenitas dan kompleksitas dari peralatan konvensional yang sedang dipakai saat itu. Dengan demikian, middleware ini berfungsi sebagai “jembatan” antara mesin pembangkit listrik dengan pemakai. [3] Penelitian lainnya yang terkait middleware yaitu “An Approach of Software Engineering

through Middleware". Penelitian ini membahas secara mendalam mengenai pendekatan-pendekatan yang diperlukan untuk membangun sebuah perangkat lunak middleware; penerapan ke beberapa sistem operasi yang sudah dikenal seperti Windows (COM) dan Linux (CORBA); solusi-solusi middleware yang biasa digunakan seperti *Transaction Middleware*, *Message-Oriented Middleware*, *Procedural Middleware*, dan *Component Middleware*.

Dengan merujuk kepada penelitian sebelumnya, pada penelitian ini penulis membuat sebuah middleware untuk menghubungkan dua aplikasi yang berbeda yaitu: sistem informasi dagang dan aplikasi e-Faktur.

2. METODE PENELITIAN

Dalam penelitian ini menggunakan metode perancangan perangkat lunak Agile (*Agile Software Development*) yang merupakan kerangka konseptual untuk mengembangkan perangkat lunak yang memperkenalkan beberapa tahapan perulangan (iterasi) selama siklus hidup sebuah proyek [4]. Karakteristik dari metode Agile adalah ringan, skala kecil hingga menengah, kebutuhan (*requirement*) yang samar dan/atau sering berubah, perancangan yang sederhana, hingga sistem yang minimal untuk menyelesaikan proyek. Nilai yang dijunjung oleh Agile adalah:

- Mengedepankan individu dan interaksi daripada proses dan alat
- Mengedepankan perangkat lunak yang berfungsi sesuai kebutuhan daripada dokumentasi yang lengkap
- Mengedepankan kolaborasi dengan pelanggan daripada negosiasi kontrak
- Mengedepankan respon terhadap perubahan daripada mengikuti rancangan

Extreme Programming (XP) merupakan pendekatan dari Agile yang paling luas digunakan di seluruh dunia. Tahapan pada proses XP yaitu [5]:

1. Planning, merupakan proses di mana individu atau tim pengembang perangkat lunak mendengarkan kebutuhan-kebutuhan (*requirements*) dari pemakai tentang proses bisnis berjalan dan mengetahui output apa yang dibutuhkan serta fitur dan fungsionalitas umum yang akan direalisasikan.
2. Design, XP melakukan tahapan rancangan dengan cara KIS (*Keep It Simple*) atau dengan kata lain dengan cara yang sesederhana mungkin. Rancangan tersebut berisi panduan implementasi sebuah proyek. Tidak lebih dan tidak juga kurang.
3. Coding, XP merekomendasikan *pair programming* yaitu 2 orang yang berada pada 1 komputer untuk menyelesaikan permasalahan pemrograman secara real-time. Individu pertama akan fokus pada detail teknis pemrograman, sementara individu lainnya fokus pada logika dan standar yang harus dipenuhi supaya tetap berada pada jalur rancangan sehingga hasilnya sesuai dengan yang diharapkan.
4. Testing, dalam tahapan ini penulis menggunakan pengujian blackbox untuk memastikan fitur yang akan dirancang dapat berjalan dengan baik. Dalam hal ini yaitu menghubungkan data dari sistem informasi penjualan ke aplikasi e-Faktur dari DJP.

Unified Modeling Language (UML) merupakan sebuah standar bahasa pemodelan yang digunakan untuk menggambarkan sebuah sistem informasi yang akan dibangun [6]. UML dapat bersifat *platform-independent* ataupun *platform-specific* tergantung pada pilihan perancang sistem. Diagram yang dipakai oleh penulis untuk menggambarkan middleware pada penelitian ini adalah *Use Case*, *Activity*, *Sequence*, dan *Class Diagram*.

Microsoft Visual Studio 2015 merupakan sebuah rangkaian tool untuk melakukan rekayasa perangkat lunak, dari fase perancangan melalui tampilan visual, kode program, pengujian, perbaikan error, analisis kualitas kode program dan performa, implementasi ke pelanggan, serta pengumpulan data pemakaian program [7]. Aplikasi middleware ini berbasis desktop yang dirancang, dikode, dan diuji menggunakan Visual Studio 2015. Bahasa pemrograman C# dan dikompilasi untuk .NET Framework versi 4.0.

Middleware ini tidak memiliki basis data tersendiri karena hanya bersifat sebagai "jembatan" antara 2 aplikasi. Aplikasi yang dihubungkan oleh middleware ini adalah sistem informasi dagang PT. Cemerlang Andalan Nusantara dengan aplikasi e-Faktur dari DJP. Basis data yang digunakan oleh

sistem informasi dagang PT. Cemerlang Andalan Nusantara adalah MySQL, sedangkan basis data aplikasi e-Faktur adalah basis data embedded. Adapun aspek penelitian meliputi fitur ekspor data penjualan, retur-penjualan, pembelian, retur-pembelian, dan penyesuaian desimal.

HASIL DAN PEMBAHASAN

Perancangan middleware antara sistem informasi dagang dengan aplikasi e-Faktur ini diawali dengan menganalisa format file impor yang dibutuhkan oleh aplikasi e-Faktur dari DJP. Aplikasi ini membutuhkan file impor dengan ekstensi *Comma Separated Value (*.csv)*. Apabila dibuka dengan aplikasi *Microsoft Excel 2013* maka tampilannya adalah seperti gambar di bawah ini:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|----|----|------------|--------------|--------------|------------|-------------|----------------|-----------|------------------------|----------------|------------|------------|---------------|------------------------|--------------|---------------|---------------|-----------------|-----------|
| 1 | FK | KD_JENIS_T | FG_PENGGANTI | NOMOR_FAKTUR | MASA_PAJAK | TAHUN_PAJAK | TANGGAL_FAKTUR | NPWP | NAMA | ALAMAT LENGKAP | JUMLAH_DPP | JUMLAH_PPN | JUMLAH_PPNBM | ID_KETERANGAN TAMBAHAN | FG_UANG_MUKA | UANG_MUKA_DPP | UANG_MUKA_PPN | UANG_MUKA_PPNBM | REFERENSI |
| 2 | LT | NPWP | NAMA | JALAN | BLOK | NOMOR RT | RW | KECAMATAN | KELURAHAN | KABUPATEN | PROVINSI | KODE_POS | NOMOR TELEPON | | | | | | |
| 3 | OF | KODE_OBJEK | NAMA | HARGA_SATU | JUMLAH_B4 | HARGA_TO | DISKON | DPP | PPN | TARIF_PPN | PPNBM | | | | | | | | |
| 4 | FK | 1 | 0 | 3.32E+11 | 8 | 2016 | 29/08/2016 | 0 | CIRCUIT MC JL RAYA SO | 492273 | 492273 | 0 | 492273 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | OF | H33 | 12359 | H33 | 12359 | 51818 | 10 | 466364 | 25909 | 492273 | 492273 | 0 | 492273 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | FK | 1 | 0 | 3.32E+11 | 8 | 2016 | 29/08/2016 | 0 | BERJAYA VA JL SIAM NO | 254546 | 254546 | 0 | 254546 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | OF | H4FIT | 1256 | H4FIT | 1256 | 25456 | 10 | 254546 | 0 | 254546 | 25454.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | FK | 1 | 0 | 3.32E+11 | 8 | 2016 | 30/08/2016 | 0 | MAYANG S/ JL TRANS KJ | 672728 | 672728 | 0 | 672728 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | OF | H4FIT | 2456 | H4FIT | 2456 | 33637 | 20 | 672728 | 0 | 672728 | 67272.7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | FK | 1 | 0 | 3.32E+11 | 8 | 2016 | 31/08/2016 | 0 | SINAR JAYA JL TRANS KJ | 2472728 | 2472728 | 0 | 2472728 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | OF | H1 | 12258 | P H1 | 12258 | P | 21820 | 10 | 130911 | 43636 | 174547 | 17454.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | OF | H1 | 13258 | 2 H1 | 13258 | 2 | 33636 | 10 | 201818 | 67273 | 269091 | 26909.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | OF | H3 | 12356 | P H3 | 12356 | P | 16364 | 10 | 98182 | 32727 | 130909 | 13090.9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | OF | H3FIT | 1245 | H3FIT | 1245 | 25455 | 10 | 152727 | 50909 | 203636 | 20363.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | OF | H3 | 13628 | R H3 | 13628 | R | 29091 | 10 | 174545 | 58182 | 232727 | 23272.7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 16 | OF | H4FIT | 1256 | H4FIT | 1256 | 25455 | 10 | 152727 | 50909 | 203636 | 20363.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | OF | H4FIT | 2456 | H4FIT | 2456 | 33636 | 10 | 201818 | 67273 | 269091 | 26909.1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | OF | H4FIT | 1245 | H4FIT | 1245 | 30000 | 10 | 180000 | 60000 | 240000 | 24000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | OF | H4 | 12342 | P H4 | 12342 | P | 24545 | 10 | 147273 | 49091 | 196364 | 19636.4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | OF | H11 | 12362 | H11 | 12362 | 69091 | 10 | 414545 | 138182 | 552727 | 55272.7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

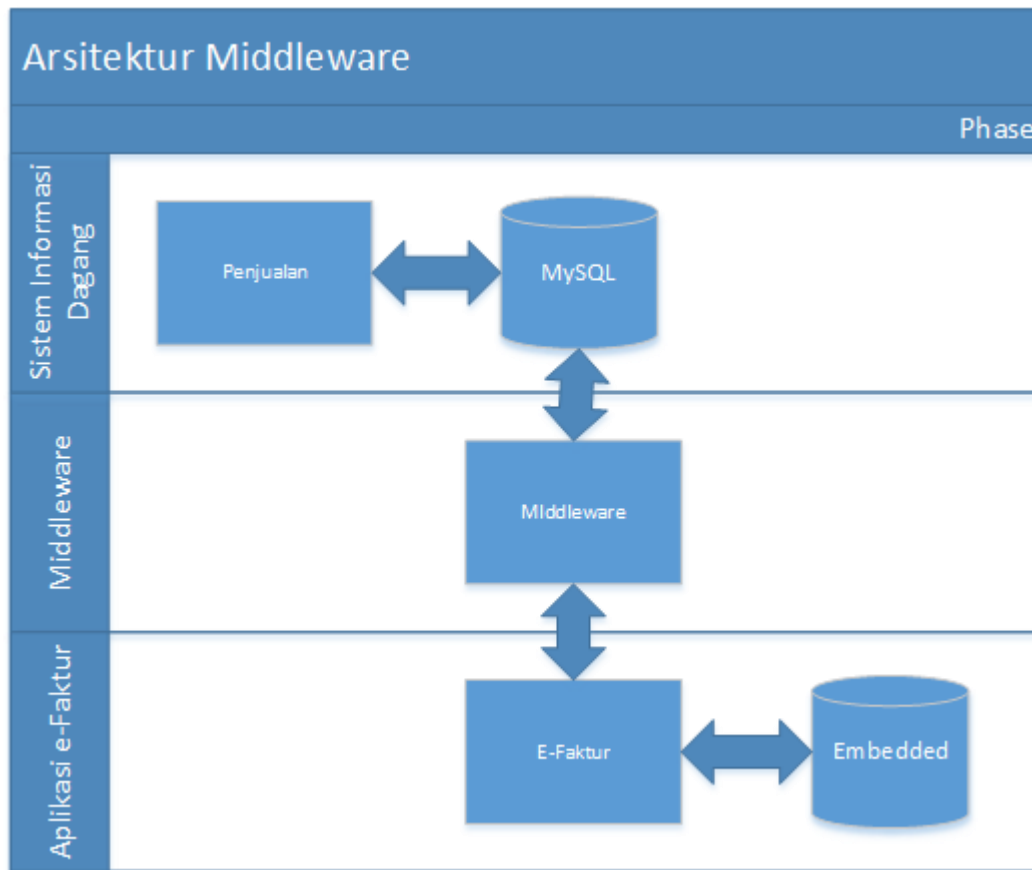
Gambar 1. Tampilan file impor (*.csv) pada Microsoft Excel 2013.

Pada gambar di atas, dapat dilihat bahwa file impor ini memiliki 3 buah header, yaitu:

- Header Faktur Pajak**, merupakan header yang berisi informasi lengkap mengenai faktur pajak yang diterbitkan. Kolom-kolomnya antara lain:
 - FK, yang berarti jenis faktur pajak. Biasanya kolom ini tetap bernilai FK pula.
 - KD_JENIS_TRANSAKSI, biasanya selalu bernilai 1.
 - FG_PENGGANTI, biasanya selalu bernilai 0.
 - NOMOR_FAKTUR, berisi nomor faktur pajak.
 - MASA_PAJAK, merupakan periode faktur pajak. Berisi nilai bulan dalam angka (1 – 12).
 - TAHUN_PAJAK, merupakan tahun faktur pajak. Berisi angka tahun.
 - TANGGAL_FAKTUR, merupakan tanggal faktur pajak diterbitkan. Tidak seperti aplikasi e-SPT yang mewajibkan pemakai menggunakan format tanggal Indonesia (dd/MM/yyyy), aplikasi e-Faktur bisa menyesuaikan dengan apapun format yang dipakai oleh komputer anda.
 - NPWP, merupakan Nomor Pengusaha Wajib Pajak dari lawan transaksi perusahaan. Diisi dengan format NPWP yang benar. Namun apabila lawan transaksi tidak memiliki NPWP, maka diisi dengan nilai 0.
 - NAMA, yaitu nama perusahaan yang menjadi lawan transaksi. Apabila memiliki NPWP, maka nama ini harus sesuai dengan nama yang tertera pada NPWP.
 - ALAMAT LENGKAP, yaitu alamat perusahaan yang menjadi lawan transaksi. Apabila memiliki NPWP, maka alamat ini harus sesuai dengan alamat yang tertera pada NPWP.
 - JUMLAH_DPP, merupakan total dari Dasar Pengenaan Pajak pada faktur pajak. Jumlah ini merupakan total dari semua barang / jasa setelah dikurangi diskon.
 - JUMLAH_PPN, merupakan hasil perhitungan 10 % dari JUMLAH_DPP. Nilai inilah yang wajib dilaporkan oleh pengusaha.
 - JUMLAH_PPNBM, biasanya tetap bernilai 0.
 - ID_KETERANGAN TAMBAHAN, biasanya dibiarkan dalam keadaan kosong.
 - FG_UANG_MUKA, biasanya tetap bernilai 0.
 - UANG_MUKA_DPP, biasanya tetap bernilai 0.
 - UANG_MUKA_PPN, biasanya tetap bernilai 0.

- UANG_MUKA_PPNBM, biasanya tetap bernilai 0.
 - REFERENSI, biasanya dibiarkan dalam keadaan kosong.
2. **Header Pengusaha Kena Pajak (lawan transaksi)**, berisi informasi lengkap mengenai lawan transaksi. Header ini digunakan untuk merekam informasi lawan transaksi ke dalam tabel referensi “lawan transaksi” apabila lawan transaksi tersebut belum pernah direkam sebelumnya. Perlu diperhatikan bahwa apabila lawan transaksi memiliki NPWP, maka informasi di bawah ini harus diisi sesuai dengan informasi yang tertera pada kartu NPWP. Kolom-kolomnya antara lain:
- NAMA.
 - JALAN.
 - BLOK
 - NOMOR
 - RT
 - RW
 - KECAMATAN
 - KELURAHAN
 - KABUPATEN
 - PROPINSI
 - KODE_POS
 - NOMOR_TELEPON
3. **Header Detil Transaksi (barang / jasa)**, berisi informasi mengenai barang / jasa yang dijual pada faktur pajak. Kolom-kolomnya antara lain:
- NAMA, yaitu nam barang / jasa yang dijual.
 - HARGA_SATUAN, yaitu harga satuan sebelum dikenai PPN.
 - JUMLAH_BARANG, yaitu jumlah barang / jasa yang dijual.
 - HARGA_TOTAL, yaitu hasil perhitungan HARGA_SATUAN dikalikan dengan JUMLAH_BARANG.
 - DISKON, yaitu diskon untuk barang / jasa yang dijual. Merupakan hasil perhitungan dari diskon per barang /jasa dikalikan dengan JUMLAH_BARANG.
 - DPP, yaitu Dasar Pengenaan Pajak. Merupakan hasil perhitungan dari HARGA_TOTAL dikurangi dengan DISKON.
 - PPN, yaitu jumlah PPN yang wajib dilaporkan oleh pengusaha. Merupakan hasil perhitungan dari DPP dikalikan dengan 10 %.
 - TARIF_PPNBM, biasanya tetap bernilai 0.
 - PPNBM, biasanya tetap bernilai 0.

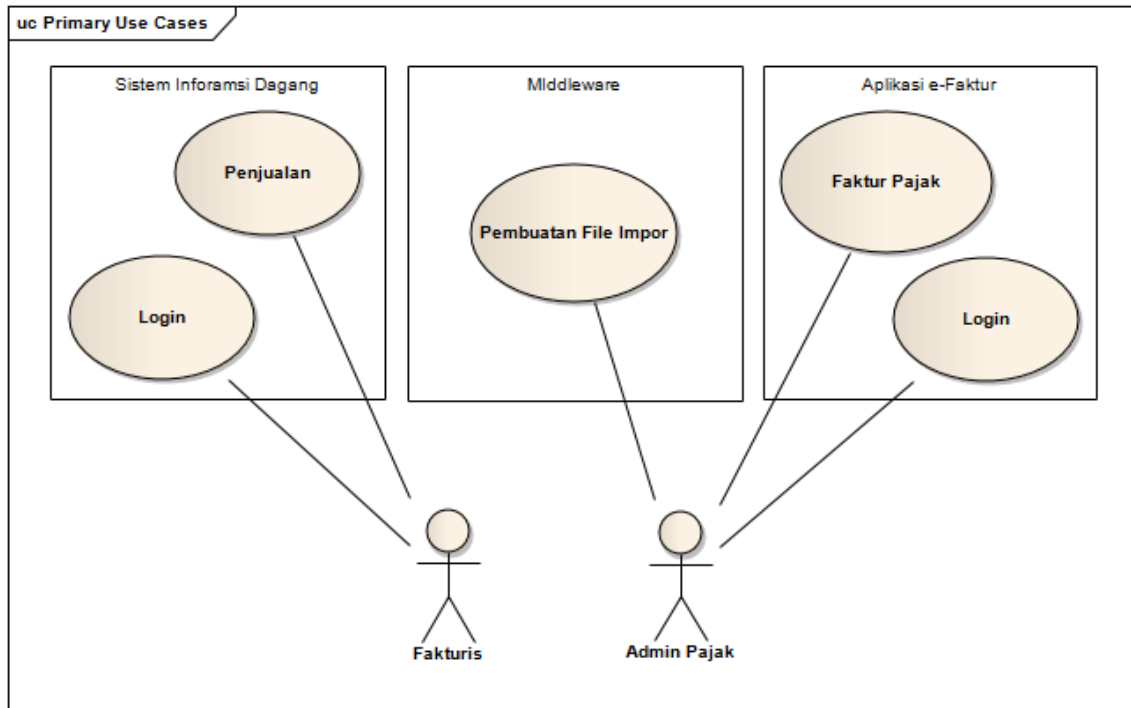
Setelah mengetahui format dari file impor yang dibutuhkan oleh aplikasi e-Faktur, tahap selanjutnya adalah merancang arsitekturmiddleware yang dapat menghasilkan file impor tersebut dari sistem informasi dagang. Arsitektur middleware bertujuan untuk mendiskripsikan posisi sistem informasi dagang, middleware dan aplikasi e-Faktur serta interaksi yang terjadi antara ketiga komponen tersebut. Berikut ini adalah arsitektur middleware yang diusulkan:



Gambar 2. *Arsitektur Middleware antara Sistem Informasi Dagang dengan Aplikasi e-Faktur.*

Gambar di atas menunjukkan bahwa middleware tidak masuk ke dalam salah satu sistem yang ada. Middleware berdiri sendiri dan bertugas menghubungkan sistem informasi dagang dengan aplikasi e-Faktur. Middleware dapat mengakses basis data sistem informasi dagang untuk memperoleh data yang diperlukan. Dari data tersebut middleware mengkonversinya menjadi data yang dapat dimengerti oleh aplikasi e-Faktur, yaitu sebuah file impor faktur pajak keluaran. Middleware tidak dapat mengakses basis data embedded dari aplikasi e-Faktur.

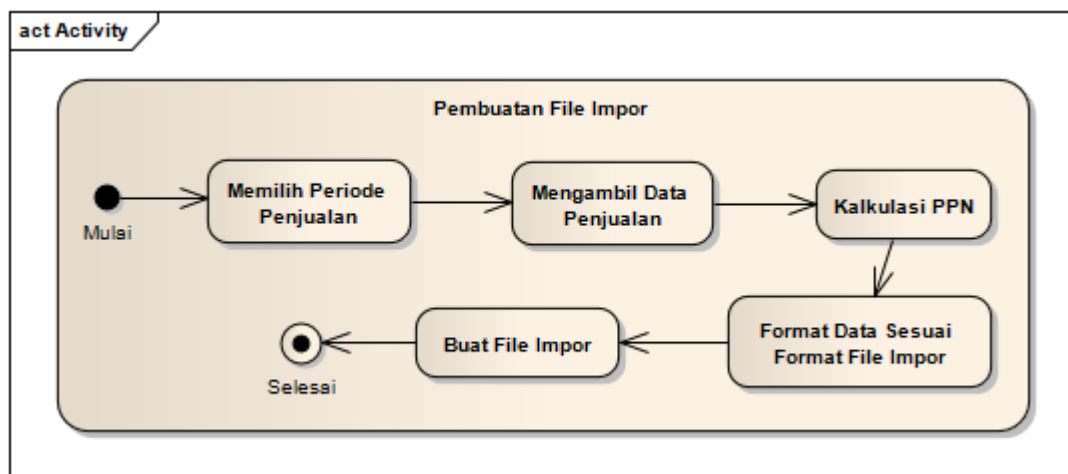
Untuk mengetahui wewenang pemakai terhadap aplikasi yang ada, dapat diketahui melalui salah satu diagram UML yaitu Use Case. Diagram Use Case untuk penelitian ini dapat dilihat pada gambar di bawah ini:



Gambar 3. Use Case Middelware antara Sistem Informasi Dagang dan Aplikasi e-Faktur.

Pada gambar 3 dapat dilihat bahwa terdapat 2 aktor yang berperan dalam prosedur penjualan hingga data masuk ke dalam aplikasi e-Faktur. Aktor pertama adalah Fakturis yang bertugas membuka faktur penjualan untuk operasional perusahaan. Data yang diinput fakturis akan masuk ke dalam basis data sistem informasi dagang. Setelah data terekam ke dalam basis data sistem informasi dagang, maka Admin Pajak dapat mengambilnya menggunakan middleware yang dirancang oleh penulis. Middleware menarik data dari basis data sistem informasi dagang kemudian mengkonversinya menjadi sebuah file impor faktur pajak keluaran. Dari file impor tersebut, admin pajak dapat mengimpornya masuk ke dalam basis data embedded aplikasi e-Faktur.

Tahapan aktivitas yang dilakukan dapat dilihat pada UML yaitu Activity Diagram. Activity Diagram pada saat membuat file impor dari middleware dapat dilihat pada gambar di bawah ini:



Gambar 4. Activity Diagram Pembuatan File Impor.

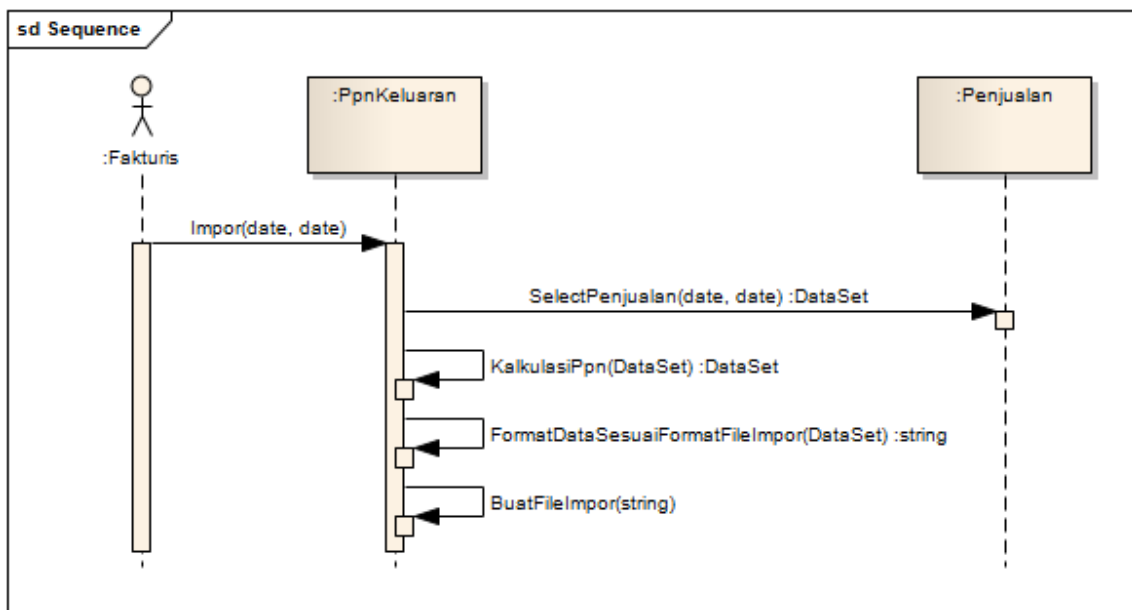
Tahapan pertama dari Activity Diagram di atas adalah memasukkan periode penjualan. Admin pajak memasukkan periode penjualan yang diinginkan yaitu tanggal awal dan tanggal akhir periode penjualan. Selanjutnya middleware akan menarik data penjualan berdasarkan periode yang sudah dimasukkan.

Dari data penjualan tersebut middleware mulai menghitung setiap record secara rinci, yaitu per barang / jasa yang dijual. Menghitung Dasar Pengenaan Pajak (DPP) adalah dengan rumus $(\text{Harga Satuan} - \text{Diskon}) * \text{Qty}$. Perlu diperhatikan bahwa DPP dihitung setelah dikurangi diskon. Setelah mendapat nilai DPP, maka

middleware akan memvalidasi nilai DPP tersebut. Aturan dari aplikasi e-Faktur adalah nilai DPP harus bulat. Maka dari itu, apabila hasil perhitungan DPP adalah desimal. Perlu dilakukan penyesuaian supaya mendapat nilai bulat. Dari nilai DPP yang sudah divalidasi tersebut, kemudian akan dikalikan 10% untuk mendapatkan nilai PPN per barang / jasa. Proses ini dilakukan kembali untuk barang / jasa yang lain dalam 1 faktur penjualan.

Setelah selesai mengulang proses dalam 1 faktur penjualan, maka dilakukan akumulasi terhadap DPP dan PPN. Middleware selanjutnya akan memvalidasi apakah nilai akumulasi DPP dan PPN sudah sesuai. Apabila belum, maka dilakukan penyesuaian sampai nilainya pas. Yang artinya $DPP * 10\%$ harus bernilai setara dengan nilai PPN yang sudah diakumulasi. Proses ini diulang pula untuk setiap faktur penjualan dalam periode penjualan yang dipilih oleh admin pajak.

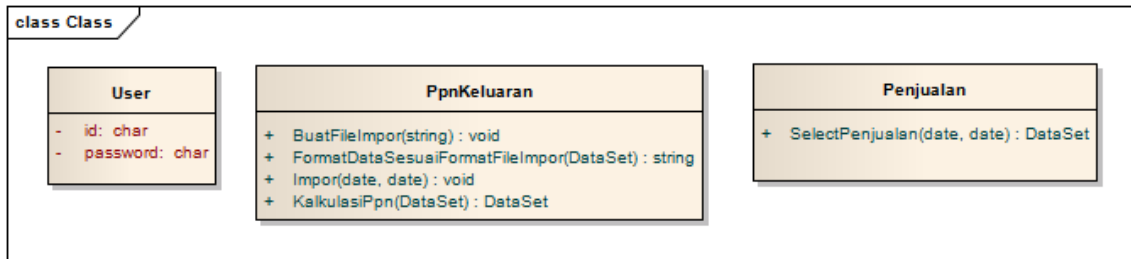
Setelah selesai menghitung DPP dan PPN untuk setiap faktur penjualan, maka middleware akan memformat data yang dikumpulkan menjadi format file impor yang sudah diterangkan pada bagian sebelumnya. File yang sudah diformat akan ditulis ke dalam file *Comma Separated Value* (*.csv). File ini akan dapat dibaca oleh aplikasi e-Faktur untuk diimpor ke dalam basis data embedded.



Gambar 5. Sequence Diagram Pembuatan File Impor.

Pada gambar 5, penulis menyajikan Sequence Diagram yang menunjukkan interaksi antara fakturis dengan 2 class yaitu PpnKeluaran dan class Penjualan. Urutan prosedur juga dapat dilihat pada diagram ini. Yang pertama yaitu fakturis memanggil fungsi Import() dengan 2 parameter yaitu tanggal1 dan tanggal2 dengan tipe data *date*. Kedua parameter ini merupakan periode penjualan yang ingin dibuat file impor-nya. Tahap selanjutnya, class PpnKeluaran akan meminta data penjualan kepada class Penjualan melalui fungsi SelectPenjualan() yang mengembalikan nilai (*return value*) berupa *DataSet*. *DataSet* merupakan kumpulan record dalam .NET Framework yang berisi record-record dari basis data. *DataSet* tersebut kemudian diproses oleh class PpnKeluaran melalui fungsi KalkulasiPpn(). Rincian proses pada fungsi ini sudah dijelaskan pada diagram Activity. Fungsi ini juga mengembalikan nilai dalam bentuk *DataSet*. Fungsi FormatDataSesuaiFormatFileImpor() akan memproses nilai kembalian ini menjadi tipe data *string* yang berisi format file impor yang sudah dapat dibaca oleh aplikasi e-Faktur. Format file impor yang sudah tersedia kemudian diubah oleh fungsi BuatFileImpor() menjadi file CSV.

Class diagram digunakan untuk menampilkan beberapa kelas yang ada dalam middleware. *Class diagram* mendeskripsikan jenis-jenis objek dalam sistem. *Class diagram* menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. *Class diagram* memiliki 3 bagian utama yaitu attribute, operation, dan name. (Gambar 6).



Gambar 6. Class Diagram Middleware antara Sistem Informasi Dagang dengan Aplikasi e-Faktur.

Pada gambar 6, dapat dilihat semua class beserta atribut dan fungsi yang digunakan pada gambar 5 (Sequence Diagram). Nilai kembali juga dijabarkan secara jelas sehingga koneksi antara gambar 5 dan gambar 6 menjadi nyata.

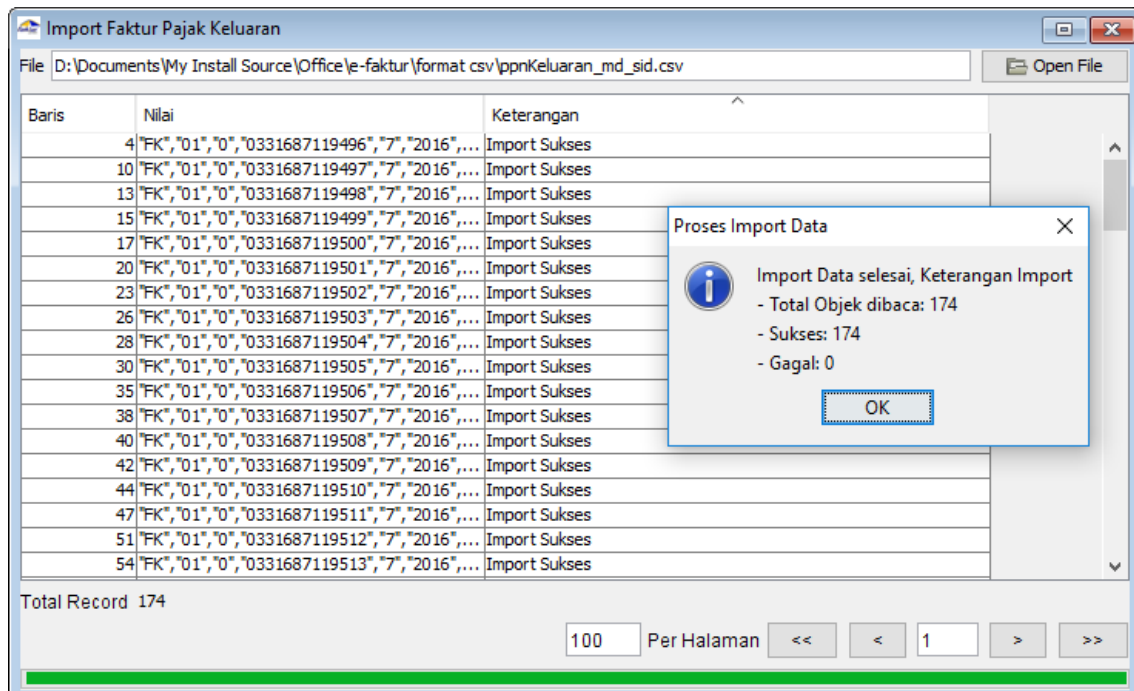
Middleware antara sistem informasi dagang dan aplikasi e-Faktur merupakan program berbasis desktop yang dirancang dengan menggunakan bahasa pemrograman C# dari .NET Framework 4.0. Tools yang digunakan adalah Visual Studio 2015. Berikut merupakan hasil rancangan form pembuatan file impor (Gambar 7).

Gambar 7. Hasil Rancangan Form Pembuatan File Impor

Gambar 7 menunjukkan tampilan yang sangat sederhana dari middleware. Sesuai dengan tujuan dari middleware yaitu middleware adalah sebuah software yang dapat mengatasi heterogenitas dan kompleksitas sebuah sistem yang terdistribusi [2], maka rancangan ini benar-benar merefleksikan tujuan tersebut. Pada formulir ini seorang fakturis cukup memilih file tujuan dan periode penjualan. Setelah klik tombol Ekspor, maka middleware akan melakukan proses yang sudah dibahas pada bagian sebelumnya dan menghasilkan sebuah file *Comma Separated Value* (*.csv) yang dapat dibaca oleh aplikasi e-Faktur.

Pengujian sistem dilakukan penulis untuk menguji 2 hal. Yang pertama adalah apakah middleware yang dibangun sudah sesuai dengan kebutuhan fungsional atau belum. Pengujian kedua adalah seberapa besar dampak yang dirasakan oleh kehadiran middleware ini dibandingkan tanpa adanya middleware.

Pengujian pertama dilakukan dengan metode blackbox testing, yaitu menguji apakah fitur yang menjadi kebutuhan sistem sudah dapat dipenuhi oleh middleware yang dihasilkan. Berikut penulis sajikan capture hasil impor pada aplikasi e-Faktur menggunakan file *.csv yang dihasilkan oleh middleware (Gambar 8).



Gambar 8.Hasil Impor pada Aplikasi e-Faktur dari DJP.

Pada gambar 8, dapat dilihat bahwa penulis menggunakan aplikasi e-Faktur untuk melakukan proses impor dan berjalan dengan sukses. Total record yang dibaca adalah 174 record, yang berarti ada 174 faktur penjualan. Record yang sukses diimpor adalah 174 record, yang artinya berhasil 100%. Dengan demikian, pengujian pertama telah berhasil.

Pengujian kedua dilakukan dengan cara membandingkan peningkatan efisiensi sebelum dan sesudah penerapan middleware. Waktu yang diperlukan untuk menginput data berupa 176 faktur penjualan ke dalam aplikasi e-Faktur secara manual dibandingkan dengan waktu yang diperlukan untuk membuat file impor dengan middleware dan kemudian mengimpornya ke dalam aplikasi e-Faktur. Untuk input manual ke dalam aplikasi e-Faktur, seorang admin pajak membutuhkan waktu paling cepat 2 menit untuk setiap faktur penjualan. Apabila dikalikan dengan 176 faktur penjualan, maka waktu yang diperlukan adalah 352 menit (5 jam 52 menit) dengan asumsi admin pajak tersebut tidak beristirahat atau jeda sama sekali. Sedangkan waktu yang diperlukan oleh middleware untuk membuat sebuah file impor yang berisi 176 faktur penjualan adalah antara 1 – 2 detik saja. Ditambah dengan waktu yang diperlukan oleh admin pajak untuk melakukan impor ke dalam aplikasi e-Faktur yaitu paling cepat 1 menit. Jadi total waktu yang dibutuhkan hanya 1 menit 2 detik saja. Untuk aktivitas input secara manual ke dalam aplikasi e-Faktur, admin pajak perlu memeriksa kembali hasil input untuk memastikan tidak ada kesalahan. Waktu yang diperlukan untuk memeriksa kembali sebanyak 176 faktur penjualan adalah sekitar 30 – 40 menit. Dan tingkat kesalahan yang biasa terjadi dalam menginput 176 faktur penjualan adalah 5 hingga 10 faktur penjualan. Tabel hasil perbandingan sistem lama dibandingkan dengan sistem baru sesudah implementasi middleware dapat dilihat pada Tabel 1:

Tabel 1.Perbandingan Pengujian Sistem

| Pengujian | Manual | Middleware |
|---|----------------|-----------------|
| Memindahkan 176 faktur penjualan ke aplikasi e-Faktur | 5 jam 52 menit | 1 menit 2 detik |
| Memeriksa kembali faktur hasil input | 30 – 40 menit | 0 detik |
| Tingkat kesalahan input | 5 – 10 faktur | 0 faktur |

KESIMPULAN

Berdasarkan hasil dan pembahasan, maka dapat diambil kesimpulan bahwa implementasi middleware berhasil menghubungkan sistem informasi dagang dengan aplikasi e-Faktur. Hal ini berarti middleware mampu meningkatkan efisiensi kerja yang sangat besar karena perusahaan tidak perlu melakukan input data operasional dua kali seperti sebelum dilakukan implementasi. Selain itu, hasil implementasi sistem mengurangi tingkat kesalahan input data sebanyak 100%. Berdasarkan pengujian sistem yang dilakukan, penerapan middleware ini dapat dikatakan telah menghilangkan kompleksitas terhadap heterogenitas sistem yang dialami oleh PT. Cemerlang Andalan Nusantara.

SARAN

Penulis menyadari bahwa middleware ini masih bisa dikembangkan lagi untuk fitur PPN masukan, Retur PPN Penjualan, dan Retur PPN Pembelian. Maka harapan penulis untuk penelitian selanjutnya adalah supaya dapat dikembangkan menjadi lebih lengkap dan semakin meningkatkan efisiensi kerja serta membantu pengusaha dalam melaporkan pajak yang benar dan akurat.

DAFTAR PUSTAKA

- [1] Direktorat Jenderal Pajak Kementerian Keuangan, 2015, DJP dan PKP siap melaksanakan E-Faktur (Online), (<http://www.pajak.go.id/content/article/djp-dan-pkp-siap-melaksanakan-e-faktur>, diakses 22 September 2016)
- [2] Ali, Hassan., Mamun, Abdullah Al., Anwar, Sultan., [A Comprehensive Study of Advancement of Electrical Power Grid and Middleware Based Smart Grid Communication Platform](#) (Online), *International Journal of Advancements in Technology*, ISSN (Online):0976-4860, Vol 7, No. 2, 2016, Hal 4.
- [3] Wedpathak, Ganesh S., [An Approach of Software Engineering through Middleware](#) (Online), *International Journal of Innovative Research in Computer and Communication Engineering*, ISSN (Online): 2320-9801, Vol. 3, No. 2, 2015, Hal 1-8.
- [4] N., Dr. Rajkumar, J., Selvakumar, B., Ranjith Babu, [Extended Scrum Method of Agile Practice for Small Scale Project Development](#) (Online), *International Journal of Innovative Research in Science, Engineering and Technology*, ISSN (Online): 2319-8753, Vol 3, Special Issue 1, 2014, Hal 374.
- [5] Pressman, R., *Software Engineering : a Practitioner's Approach*. Sixth Edition. Mcgraw-Hill., Singapore. 2005.
- [6] Larman, C. (2012). *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development*. Pearson Education India.
- [7] Microsoft Developer Network (MSDN), Visual Studio IDE (Visual Studio 2015) (Online), (<https://msdn.microsoft.com/en-us/library/dn762121.aspx>, diakses 23 September 2016).