Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.2024117365
Vol. 11, No. 6, Desember 2024, hlm. 1213-1222 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

MIGRASI DATA HASIL OPTIMALISASI PEMUATAN KONTAINER PADA *LEGACY SYSTEM* KE *SYSTEM APPLICATIONS AND PRODUCTS* DENGAN *BATCH DATA COMMUNICATION*

Rehulina Tarigan*[1], Ely Nuryani[2], Susy Katarina Sianturi[3], Nur Hidayanti[4], Widyawati[5]

[1,2,4,5]Universitas Banten Jaya, Serang, [3]Sekolah Tinggi Teknologi Ilmu Komputer Insan Unggul, Cilegon
Email: [1]rtarigan@unbaja.ac.id, [2]elynuryani@unbaja.ac.id, [3]susykatarina@insanunggul.ac.id,
[4]nurhidayanti@unbaja.ac.id, [5]widyawati@unbaja.ac.id
[*]Penulis Korespondensi

(Naskah masuk: 11 Juni 2023, diterima untuk diterbitkan: 19 November 2024)

Abstrak

Modul SAP yang diimplementasi di PT. Indah Kiat Pulp & Paper, tidak mempunyai sistem untuk optimalisasi pemuatan *order* ke kontainer. Sebelum implementasi SAP, proses memasukkan data *order*, optimalisasi pemuatan data *order* ke kontainer dan pembuatan surat jalan dilakukan semuanya pada sistem CSO (*Container Stuffing Optimization)*, sehingga data *order* pasti sinkron dengan data surat jalan atau sesuai dengan aktual pengiriman. Setelah implementasi SAP, data *order* untuk optimalisasi pemuatan kontainer dimasukkan pada CSO (*legacy system*) sedangkan proses pembuatan surat jalan dilakukan pada SAP. Dua proses yang berbeda dilakukan pada dua sistem dengan *platform* berbeda. Penggunaan dua sistem yang berbeda mengakibatkan aktual pengiriman pada dokumen surat jalan tidak sesuai dengan data *order* dari pelanggan yang mengakibatkan komplain pelanggan. Solusi untuk mengatasi permasalahan ini adalah dengan melakukan migrasi data *order* hasil optimalisasi pemuatan pada CSO ke SAP sehingga data *order* terintegrasi dengan data surat jalan. Metode BDC dipergunakan untuk melakukan migrasi ke SAP yaitu dengan mentransfer data *order* hasil optimalisasi pemuatan dari CSO ke SAP serta mengotomatisasi proses *picking* dan *packing* di SAP untuk menghasilkan surat jalan. Hasil dari penelitian berupa sinkronisasi data *order* dan data surat jalan serta percepatan proses pembuatan surat jalan menjadi 3 kali lebih cepat dari semula sehingga dapat mengurangi komplain pelanggan dan mempercepat proses pengiriman. Hasil pengujian *black box* memperlihatkan fungsional sistem berjalan dengan baik.

Kata kunci: BDC, migrasi data, otomatisasi, pengujian *black box,* SAP

DATA MIGRATION OF CONTAINER LOADING OPTIMIZATION ON LEGACY SYSTEM TO THE SYSTEM APPLICATIONS AND PRODUCTS USING BATCH DATA COMMUNICATION

Abstract

SAP module implemented at PT. Indah Kiat Pulp & Paper does not have a system for optimizing the loading of orders into containers. Before implementing SAP, the process of entering order data, optimizing loading of order data into containers and creating delivery note was all done in the CSO (Container Stuffing Optimization) system, so that order data was definitely in sync with delivery note data or in accordance with actual delivery. After implementing SAP, order data for optimizing container loading is entered into CSO (legacy system) while the delivery note creation process is carried out in SAP. Two different processes are carried out on two systems with different platforms. The use of two different systems resulted in the actual delivery of the delivery note document not matching the order data from the customer which resulted in customer complaints. The solution to overcome this problem is to migrate the order data resulting from loading optimization in CSO to SAP so that the order data is integrated with the delivery note data. The BDC method is used to migrate to SAP, namely by transferring order data resulting from loading optimization from CSO to SAP and automating the picking and packing process in SAP to produce delivery note. The results of the research are in the form of synchronizing order data and delivery note data as well as accelerating the process of making delivery note documents to 3,775 times faster than before, thereby reducing customer complaints and speeding up the delivery process. The black box testing results show that the system functionality is running well.

Keywords: *automation, BDC, black box testing, data migration, SAP*

1. PENDAHULUAN

SAP (*Systems, Applications and Products*) merupakan salah satu merek spesifik dari perangkat lunak ERP (*Enterprise Resource Planning*) yang dikembangkan oleh perusahaan perangkat lunak Jerman SAP SE. Ada berbagai modul SAP yang dirancang untuk berbagai keperluan perusahaan mulai dari keuangan hingga manajemen rantai pasokan. SAP menjangkau jauh dan luas, termasuk perusahaan lokal dan perusahaan global (Red Global, 2022). Sistem SAP mengintegrasikan semua proses fungsional pada perusahaan mulai dari penjualan produk atau jasa sampai menghasilkan uang (*order to cash*). Pada kenyataannya, implementasi SAP dapat membutuhkan waktu yang lama (satu atau dua tahun) karena harus dilakukan beberapa penyesuaian dengan proses bisnis di organisasi atau perusahaan. Ada kalanya ditemukan kasus yaitu proses bisnis yang masih dibutuhkan perusahaan dalam menjalankan bisnisnya, tidak ada pada modul SAP. Modul yang tidak ada di SAP itu merupakan bagian *legacy system* (*sub legacy system*) yang masih harus dipertahankan perusahaan. Terhadap *legacy system* itu sendiri, akan dilakukan transformasi/modernisasi ke sistem SAP. Menurut (R. Khadka, 2016), bahwa perusahaan dengan *legacy system* (sistem warisan) dihadapkan pada dilema. Meskipun *legacy system* sudah tidak sesuai dengan perkembangan teknologi di era digital bahkan membutuhkan biaya pemeliharaan yang besar, perusahaan tidak dapat begitu saja menyingkirkan sistem perangkat lunak lama karena perangkat lunak ini merupakan bagian dari sistem inti yang dipergunakan untuk menjalankan bisnis sehari-hari. Oleh karena itu, perlu dilakukan modernisasi perangkat lunak dengan tetap menggunakan proses bisnis dari perangkat lunak lama tetapi dalam lingkungan teknologi baru.

Beberapa organisasi yang memiliki *legacy system*, tetap menggunakan beberapa bagian darinya karena mendukung pengiriman layanan kepada pelanggan. Sistem ini penting bagi organisasi karena telah dioperasikan selama bertahun-tahun dan memiliki nilai bisnis yang tinggi (Abu Bakar, Humairath KM., Razali, Rozilawati., Jambari, Dian Indrayani., 2020). Data pada *legacy system* dan SAP perlu diintegrasikan dengan melakukan migrasi data dari *legacy system* ke SAP. Hal yang sama dialami oleh PT. Indah Kiat Pulp & Paper pada saat implementasi SAP yaitu ada sistem yang dipergunakan untuk melakukan optimalisasi pemuatan kontainer tetapi sistem ini tidak ada di SAP. Sistem aplikasi optimalisasi pemuatan kontainer yang biasa disebut *Container Stuffing Optimization* (CSO) merupakan sistem yang berfungsi untuk memaksimalkan pemuatan *paper sheet* atau *paper roll* ke dalam kontainer berdasarkan data *order* yang tercantum pada dokumen *Sales Order* (SO). Pemuatan optimal ini untuk mengurangi biaya pengiriman ke pelanggan, terlebih pengiriman ke luar negeri (*export*). Hasil dari sistem ini berupa pemuatan optimal yang memberikan data berupa jumlah kontainer minimal yang diperlukan, jumlah dan spesifikasi barang serta susunan barang di dalam setiap kontainer Sebelum dimuat ke dalam kontainer, dilakukan *scan label* yang ada pada setiap barang pada sistem CSO. Hasil *scan* label akan digunakan untuk mencetak DN pada sistem CSO, sehingga dapat dikatakan bahwa data ini adalah aktual pengiriman ke pelanggan yang sesuai dengan data order pada dokumen SO.

Pada saat SAP sudah diimplementasikan, maka proses cetak surat jalan (DN) dilakukan di SAP. Sebelum mencetak DN, data hasil *scan* label pada sistem CSO dimasukkan ke sistem SAP melalui proses *picking* dan *packing* pada *transaction code* VL02N. Hasil *scan* label sebagai *output* sistem CSO merupakan *input* bagi sistem SAP. Berdasarkan uraian di atas, menjadi jelas bahwa ada dua sistem dengan *platform* (*database* dan sistem aplikasi) berbeda yang dipergunakan untuk menghasilkan surat jalan atau DN yang sesuai dengan data *order*. Keterlibatan dua sistem dengan *platform* berbeda ini menyebabkan beberapa hal:

a. Berpotensi menimbulkan ketidaksinkronan antara data order dengan aktual pengiriman karena terjadi *human error* saat input data hasil sistem CSO ke SAP. Hal ini menambah biaya pengiriman karena salah kirim spesifikasi barang.
b. Pembuatan DN pada sistem SAP melalui banyak langkah (*step*) pada proses *picking* dan *packing* di *transaction code* (tcode) VL02N. *Transaction code* pada SAP digunakan untuk mendapatkan akses mudah ke aplikasi khusus atau untuk memanggil proses tertentu yang direpresentasikan dengan *screen* berupa *user interface* tertentu. Berbagai kategori tcode ditentukan menurut area aplikasi dan modul pada SAP seperti tcode SE11-*Dictionary defenition*, SHDB – *Batch input recorder* atau VL02N – *Logistcs Execution Shipping* (Tutorialspoint.com). Untuk membuat surat jalan satu kontainer menghabiskan waktu 15 menit. Satu SO terdiri dari 20 sampai 25 kontainer yang diharapkan berjalan ke pelabuhan secara bersamaan sehingga semua kontainer dapat dimuat ke satu kapal untuk dikirim ke pelanggan. Proses *picking* dan *packing* yang cukup lama menyebabkan pencetakan DN lama, sehingga pengiriman berpotensi terlambat.

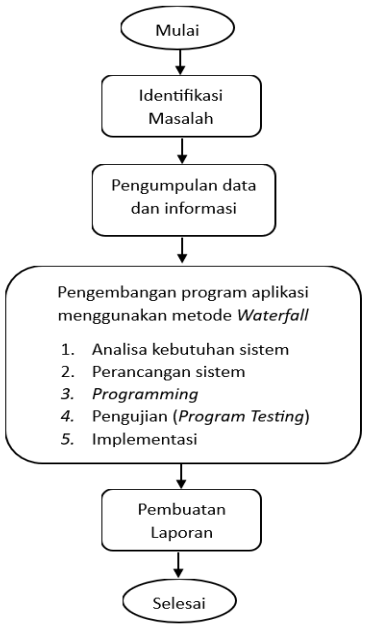
Untuk mengatasi permasalahan tersebut, perlu dibuat program aplikasi untuk mengintegrasikan data sistem CSO dan SAP serta mengotomatisasikan proses *picking* dan *packing* pada SAP sehingga:

a. Data order yang ada pada *Sales Order* sinkron atau sesuai dengan data aktual pengiriman pada dokumen *Delivery Note* (DN).
b. Proses pembuatan *Delivery Note* (DN) lebih cepat karena program aplikasi dapat mengotomasikan proses *picking* dan *packing* dengan cara mengurangi *step* atau langkah proses yang ada.

Nama program aplikasi untuk mengintegrasikan data CSO ke SAP adalah *Data Migration Interface*.

2. METODE PENELITIAN

Secara umum, beberapa tahap proses yang dilakukan pada penelitian, dapat digambarkan pada diagram berikut:



Gambar 1. Tahap proses penelitian

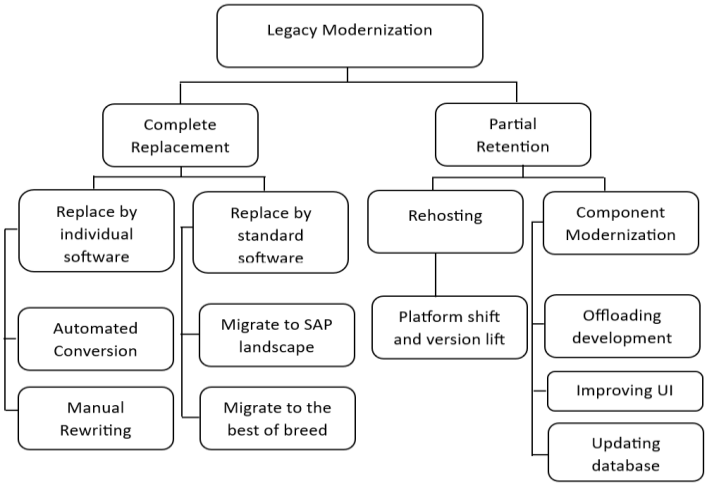
Identifikasi masalah berupa komplain pelanggan karena sering terjadi ketidaksesuaian data *order* dan aktual pengiriman pada surat jalan. Pengiriman ke pelanggan lambat karena pembuatan surat jalan di SAP untuk satu kontainer membutuhkan waktu lama yaitu melalui banyak *step* (26 langkah) pada proses *picking* dan *packing*. Pengumpulan data dan informasi dilakukan dengan observasi secara langsung ke *user* untuk mengetahui secara detail proses optimalisasi pemuatan ke kontainer pada sistem CSO dan melakukan proses *picking* dan *packing* di SAP untuk membuat surat jalan. Metode pengembangan program aplikasi menggunakan *waterfall*. Model *waterfall* membagi siklus hidup pengembangan perangkat lunak menjadi satu set fase yang dapat dimulai setelah fase sebelumnya selesai dikerjakan. Dengan demikian proses pengembangan dilakukan secara berurutan (*sequence linier*) (GeeksforGeeks, 2022). Model *waterfall* secara umum terdiri dari (Sommerville Ian, 2011):

1) Analisa kebutuhan sistem
   Analisa kebutuhan sistem dilakukan berdasarkan data yang diperoleh pada saat observasi dan wawancara serta diskusi dengan pengguna. Berdasarkan pengamatan yang dilakukan di lapangan, diperlukan sistem untuk dapat mengintegrasikan data dari CSO ke SAP.
2) Perancangan sistem
   Perancangan *input* berupa *user interface* yaitu *screen* untuk mengumpulkan data dari sistem CSO sebagai hasil optimalisasi pemuatan data *order* ke kontainer (Gambar 7). Data tersebut disimpan dalam format *flat file* (file.txt) dan disimpan ke *local disc* untuk selanjutnya ditransfer ke SAP. Migrasi data dari CSO ke SAP dilakukan melalui *screen* seperti pada Gambar 8. Proses yang ada pada *screen* ini akan mentransfer data *order* di sistem CSO untuk menghasilkan surat jalan di SAP melalui proses *picking* dan *packing*. Perancangan proses meliputi proses pembuatan *flat file*, proses perekaman (*recording*) untuk *picking* dan *packing* di SAP dengan Tcode SHDB serta perancangan blok program otomatisasi proses pembuatan surat jalan (Gambar 6.). Program atau sistem yang dibangun tidak menghasilkan *output* dalam bentuk laporan (*report*), melainkan menghasilkan sinkronisasi data aktual pengiriman di SAP dengan hasil optimalisasi pemuatan data *order* ke kontainer di CSO.
3) *Programming*
   Tahap pembuatan kode untuk dijalankan sehingga menghasilkan program aplikasi sesuai dengan analisa dan perancangan. Program dibuat dengan menggunakan bahasa khusus untuk SAP yaitu ABAP. Program dibuat sesuai dengan blok program pada Gambar 6.
4) Pengujian (*integration testing*)
   Pengujian terintegrasi meliputi pengumpulan data dari CSO, disimpan ke lokasi tertentu pada *local disc*. Ada kalanya terjadi *error* apabila jenis dan tipe data yang akan ditransfer dari CSO tidak sesuai dengan yang dibutuhkan di SAP. Pengujian berikutnya dilakukan pada proses otomatisasi pembuatan surat jalan di SAP. Proses ini memastikan bahwa program sudah dibuat sesuai dengan hasil perekaman terhadap semua langkah sebanyak 26 *step* yang dilakukan untuk *picking* dan *packing* pada Tcode VL02N.
5) Implementasi
   Sistem *go-live* dan siap dipakai oleh pengguna untuk memasukkan data sebenarnya.

Penelitian menggunakan metode *waterfall* untuk program aplikasi karena sistem yang akan dikerjakan mempunyai ruang lingkup yang sudah jelas dan tidak terlalu luas serta tidak perlu ada banyak perubahan selama proses pembuatannya. Pada prinsipnya *waterfall model* cocok dipergunakan jika persyaratan perangkat lunak yang dibutuhkan sudah dapat dipahami dengan baik dan tidak ada perubahan secara radikal selama proses pengembangan sistem (Sommerville Ian, 2011). Hasil penelitian dilaporkan dalam bentuk karya ilmiah yang dipublikasikan pada jurnal nasional terakreditasi.

2.1. Modernisasi *Legacy System*

Organisasi yang masih menggunakan *legacy system* harus memeriksa kemampuan sistem yang ada untuk mendukung bisnis organisasi.



Gambar 2. Diagram solusi modernisasi sistem

Modernisasi *legacy system* perlu dilakukan untuk memastikan sistem dapat memenuhi harapan yang dibutuhkan. Menurut (Lizbeth Wilkins, 2018), solusi untuk modernisasi *legacy system* dapat dikelompokkan menjadi: *Complete Replacement* dan *Partial Retention* seperti Gambar 2.

a. *Complete Replacement*
   Penggantian menyeluruh sistem lama dapat dilakukan dengan perangkat lunak standar atau perangkat lunak yang dikembangkan sendiri dengan menulis ulang kode program pada sistem lama baik secara manual maupun otomatis.
b. *Partial Retention*
   Perusahaan tetap mempertahankan sistem lama tapi melakukan modernisasi dengan mengganti arsitektur *platform*. Modernisasi dilakukan pada komponen sistem seperti *user interface* atau melakukan *upgrade database*.

Sedangkan menurut (Shannon J., Barnes, 2022), bahwa pendekatan untuk melakukan modernisasi *legacy system* terdiri dari:

a. *Replace legacy system*
   Cara ini melibatkan penggantian sistem lama dengan baru, misalnya ke SAP
b. *Rebuild legacy system*
   Membangun kembali sistem lama dengan menggunakan teknologi terbaru dan menambah fitur baru sesuai dengan kebutuhan terkini dari organisasi.
c. *Rehost legacy system*
   Memindahkan sistem lama ke lingkungan *hosting* baru (*cloud*) sehingga dapat diakses dari mana saja.
d. *Re-platform legacy system*
   Modernisasi terkait dengan pemindahan *platform* sistem lama ke *platform* baru, seperti dari Unix ke Windows.
e. *Refactor and re-architect legacy system*
   Opsi hibrid merupakan teknik yang melibatkan penambahan teknologi baru pada sistem lama untuk meningkatkan kinerja sistem pada level *security*, *safety* dan *compliance*.
f. *Retain or retair legacy system*
   Pendekatan ini memungkinkan organisasi untuk dapat mempertahankan sistem lama atau menghentikan sepenuhnya. Jika organisasi mempertahankan sistem lama maka harus ada rencana untuk mengoptimalkan fungsi paling penting dari sistem sehingga dapat bekerja lebih efisien. Jika menghentikan sepenuhnya sistem lama, berarti organisasi mentransmisikan semua *user* ke sistem lain yang sudah ada.

Beberapa faktor yang dapat mempengaruhi penggantian atau modernisasi *legacy system*, seperti yang dikemukakan oleh (E. Langbauer, 2019), di antaranya adalah:

a. Keinginan untuk modernisasi secara cepat. Hal ini biasa terjadi pada manajemen tingkat atas yang ingin mengganti sistem lama secara cepat.
b. Ada tekanan untuk penggantian perangkat lunak misalnya, sistem lama sudah tidak mendukung bisnis yang ada, resiko sistem keamanan atau perubahan undang-undang dan peraturan.
c. Pengalaman dari tim proyek yang mendorong untuk melakukan modernisasi sistem
d. Biaya perawatan yang tinggi pada sistem lama
e. Tujuan perusahaan juga dapat mempengaruhi modernisasi sistem. Misalnya jika standarisasi adalah tujuan bisnis, maka perangkat lunak standar merupakan pilihan yang tepat.

2.2. Migrasi Data ke SAP

Michael & Johann (dalam Syaiful Bakhri, 2019) mengatakan migrasi data adalah proses transfer data bisnis dari sistem non-SAP ke SAP. Migrasi data terkait dengan *legacy system* dan *legacy data*. *Legacy system* adalah sistem aplikasi yang mempunyai data yang ditransfer ke SAP. Data ini disebut *legacy data*. Migrasi data dalam sistem SAP dilakukan dengan pemrograman untuk mengubah data yang heterogen dari sistem lain ke SAP. Berbagai jenis proyek migrasi data dapat dijelaskan dari berbagai sudut pandang seperti berikut ini (Densborn, F et al., 2016):

a. *Initial Goal (Green Field)*
   Migrasi data dilakukan ke dalam sistem SAP yang baru diimplementasikan. Data berasal dari sistem sumber yang berbeda.
b. *Upgrading an Existing System (Brown Field)*
   Dengan memutakhirkan sistem yang ada, data baru atau proses bisnis baru di SAP diperluas dengan cara melakukan migrasi data ke SAP dari sumber lain.
c. *Phased Roll-Out*
   Tahap ini merupakan kombinasi *initial load* dan *upgrading an existing system*. Proyek migrasi data dilakukan jika terjadi restrukturisasi lanskap teknologi informasi. Jenis ini biasanya dilakukan jika ada merger dan akuisisi anak perusahaan lokal atau internasional sehingga dilakukan pemusatan sistem di SAP.
d. *Big Bang*
   Implementasi *big bang* adalah peluncuran bertahap dalam satu fase tunggal. Hal ini biasa dilakukan jika terdapat sejumlah anak perusahaan menggunakan proses bisnis yang sama tanpa atau sedikit melakukan perubahan pada proses perusahaan induk. Resiko kegagalan implementasi *big bang* lebih tinggi dibandingkan dengan bertahap (*phased roll-out*).
e. *System Optimization*
   Jika sistem produksi SAP harus dipindahkan sepenuhnya ke sistem SAP yang lain, harus digunakan teknik khusus yang disebut *system optimizations* (*system landscap optimizations*). Hal yang paling umum dilakukan untuk pengoptimalan sistem adalah menjual anak perusahaan atau mengakuisisi perusahaan lain.
f. *Cloud Migration*
   Merupakan proses migrasi data dari sistem perusahaan ke *cloud* publik sebagai *Software as a Service* (SaaS). Untuk solusi *cloud*, SAP menawarkan *tools* untuk memigrasikan data perusahaan, sehingga data dari file Excel atau XML dapat dimigrasikan ke sistem *cloud*.

Menurut (Kellton Tech, 2022), *tools* ETL (*Extract-Transform-Load*) merupakan alat yang menyediakan langkah-langkah praktis untuk migrasi data ke SAP, yaitu:

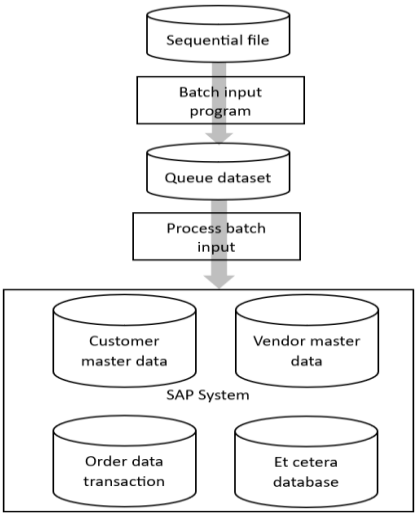
a. *Extract*
   Ini adalah langkah awal dalam menyediakan data yang valid dari sistem sumber. Penggalian data bisnis dilakukan, diidentifikasi, diekstrak dan dibersihkan sehingga proses transfer data ke SAP berjalan dengan baik. Pada penelitian ini, data yang sudah valid dari sistem CSO disediakan dalam bentuk *flat file* yang akan ditransfer ke sistem SAP.
b. *Transform*
   Langkah ini memfasilitasi harmonisasi antara *legacy system* dan SAP. Selama proses ini, kita dapat mengolah data menjadi format SAP melalui pembuatan sejumlah *function* atau *procedure* untuk mentransfer data ke SAP. Pada penelitian ini proses transfer data dari CSO ke SAP dilakukan dengan membuat program ABAP untuk mengambil data dari *flat file* yang disimpan di *local disk* untuk selanjutnya ditransfer ke sistem SAP.
c. *Load*
   Memuat data yang sepenuhnya sudah diproses dan divalidasi ke modul SAP yang sesuai. Pada penelitian ini, data CSO ditransfer ke modul *Sales & Distribution* menggunakan *transaction code* (tcode) VL02N dan melakukan proses *picking* dan *packing* untuk setiap kontainer secara otomatis.

2.3. Metode Migrasi *Batch Input Session*

SAP menyediakan beberapa metode untuk melakukan migrasi data dari sistem non-SAP (*legacy system*) ke sistem SAP (S. Markandeya, 2017):

a. *Direct Input*
   Metode ini memerlukan modul untuk melakukan validasi terhadap data yang diinput dari sistem non-SAP. Migrasi data dengan metode *direct input* dapat dikelompokkan ke metode *Legacy System Migration Workbench* (LSMW) yang terdiri dari:
   1) *Standard batch/direct input*
   2) *Batch input recording*
   3) *Business object method* (BAPI)
   4) IDoc (*intermediate document*)
b. *Batch Input Session* atau *BDC*
   Aktivitas diawali dengan melakukan perekaman (*recording*) traksaksi, misalnya untuk membuat data pelanggan menggunakan *transaction code* (tcode) XD01. Setelah selesai melakukan perekaman, dibuat program ABAP berdasarkan hasil perekaman tersebut. Ketika program ABAP dieksekusi untuk melakukan migrasi, data tidak secara langsung dimasukkan ke table modul fungsional SAP tetapi data dimasukkan terlebih dahulu ke dalam antrian atau sesi tabel basisdata (*session database table*). Selanjutnya *session* dapat dijalankan secara *background* dengan menggunakan sebuah tcode untuk mentransfer data dari *session* ke tabel basisdata fungsional pada SAP.
c. *Call Transaction*
   Program ABAP yang sama seperti pada metode BDC dapat dipergunakan pada metode *call transaction*. Perbedaannya adalah bahwa pada metode *call transaction*, data secara langsung ditransfer ke dalam tabel basis data modul fungsional SAP.

Metode *batch input session* akan mengeksekusi sebuah *session*, tidak dengan cara interaktif satu per satu, tetapi transfer data ke SAP dilakukan sekaligus dalam jumlah besar. Gambar 3. menjelaskan alur proses migrasi data ke SAP (SAP Documentation, 2015).



Gambar 3. Proses *batch input session*

Sequential file adalah *flat file* berupa file.txt. *Batch input program* merupakan program untuk mengambil data dari *flat file* yang dibuat dengan menggunakan ABAP. *Queue dataset* berupa data yang disimpan sementara pada tabel BDCDATA yang ada dan sudah didefinisikan pada *batch input program*. Selain data dari *flat file*, BDCDATA juga berisi sejumlah kode instruksi yang diperoleh dari hasil perekaman tcode tertentu. Proses *batch input* adalah program pemanggilan *transaction code* dengan sintaks: CALL TRANSACTION 'VL02N' USING BDCDATA MODE 'E' UPDATE 'S' sehingga semua data yang ada pada BDCDATA dapat ditransfer ke sistem SAP. Struktur tabel BDCDATA terdiri dari nama kolom: PROGRAM, DYNPRO, DYNBEGIN, FNAM, FVAL seperti pada Tabel 1. (TCodeSearch.com):

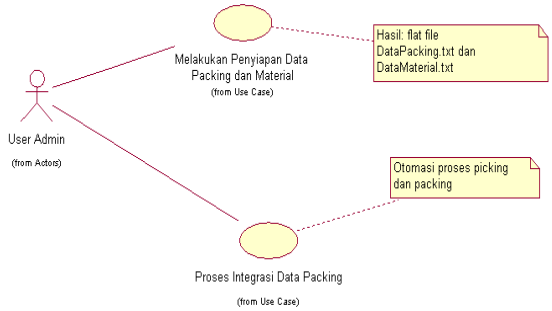
Tabel 1. Struktur tabel BDCDATA

Column Name	Description	Data Element	Data Type	Length
PROGRAM	BDC module	BDC_PROG	CHAR	40
DYNORO	Dynpro value	BDC_DYNR	NUMC	4
DYNBEGIN	BDC Dynpro Start	BDC_START	CHAR	4
FNAM	Field Name	FNAM	CHAR	132
FVAL	Field Value	BDC_VAL	CHAR	132

3. HASIL DAN PEMBAHASAN

3.1. Pemodelan Proses

Keseluruhan proses dalam perancangan program *Data Migration Interface* digambarkan dalam bentuk pemodelan visual dengan *usecase diagram*. *User admin* mempersiapkan data *flat file* pada aplikasi CSO melalui proses *scan* label. *Usecase* Proses Integrasi Data Packing dilakukan pada program SAP. Sebelum menjalankan program *Data Migration Interface*, admin mengetikkan tcode untuk program *Data Migration Interface* yaitu: YSRQ301T. Program akan melakukan proses *picking* dan *packing* secara otomatis (*background*).



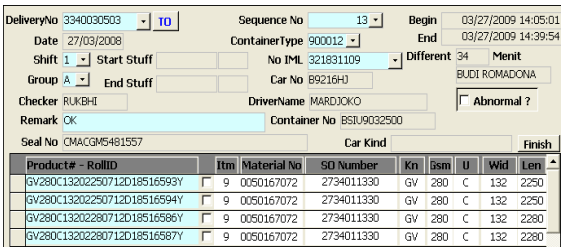
Gambar 4. Diagram *usecase*

3.2. Program *Data Migration Interface*

Proses pembuatan program *Data Migration Interface* pada SAP mempunyai beberapa tahap seperti yang dijelaskan pada penjelasan sub bab berikutnya.

3.2.1. Proses Pembuatan *Flat File*

Data *flat file* berasal dari hasil *scan label* pada aplikasi CSO (*Container Stuffing Optimization*) sebagai *legacy system* (Gambar 5.)



Gambar 5. *Screen* input kontainer dan *scan label*

Hasil *scan label* disimpan ke *local disc* sebagai *flat file* yang dibagi menjadi dua *file* dengan nama DataPacking.txt dan DataMaterial.txt. Setiap *file* mempunyai struktur seperti pada Tabel 2. dan Tabel 3. Struktur tabel ini akan di-*copy* ke struktur *temporary table* yang ada di program *Data Migration Interface* untuk menampung data dari *flat file*.

Tabel 2. Struktur tabel DataPacking

No	Nama Field	Tipe Data	Keterangan
1	DlvNo	Text	Nomor order
2	ContrNo	Text	Nomor kontainer
3	ContrType	Text	Tipe kontainer
4	SeqNo	Text	Nomor urut kontainer
5	CarNo	Text	Nomor mobil
6	ContrWeight	Text	Berat maksimum kontainer
7	Driver	Text	Nama supir
8	SealNo	Text	Nomor seal

Tabel 3. Struktur tabel DataMaterial

No	Nama Field	Tipe Data	Keterangan
1	Item	Text	Nomor item
2	Matnr	Text	Nomor material
3	Qty	Text	Jumlah roll
4	Weight	Text	Berat roll
5	HUMaterial	Text	Handling unit material

3.2.2 Proses Perekaman (*Recording*) dengan Tcode SHDB

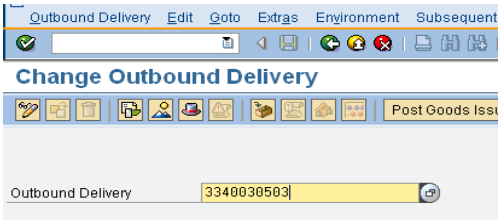
Perekaman transaksi proses *picking* dan *packing* dilakukan dengan tcode SHDB. Pada saat melakukan proses perekaman, hal yang paling perlu diperhatikan adalah bahwa proses tidak boleh mengalami kesalahan. Jika pada saat perekaman, ada sedikit saja proses yang mengalami kesalahan, maka perekaman harus diulang dari awal. Supaya diperoleh hasil perekaman yang valid, harus dilakukan simulasi beberapa kali sampai diperoleh hasil perekaman (*recording*) tanpa kesalahan. Semua hasil perekaman disimpan sementara pada tabel BDCDATA yang didefinisikan pada program *Data Migration Interface*. Semua hasil perekaman transaksi proses *picking* dan *packing* pada VL02N akan dieksekusi secara *background*. Tabel 4. menjelaskan langkah-langkah (*step*) proses *picking* dan *packing* pada tcode VL02N yang dilakukan secara manual yang terdiri dari 26 langkah.

Tabel 5 merupakan contoh data yang dimasukkan ke tabel BDCDATA pada program ABAP untuk transaksi mengganti satuan TO (ton) ke KG (kilogram). Data itu diperoleh dari hasil perekaman transaksi VL02N.

3.2.3. Pembuatan *Batch Input Program* (*Data Migration Interface*)

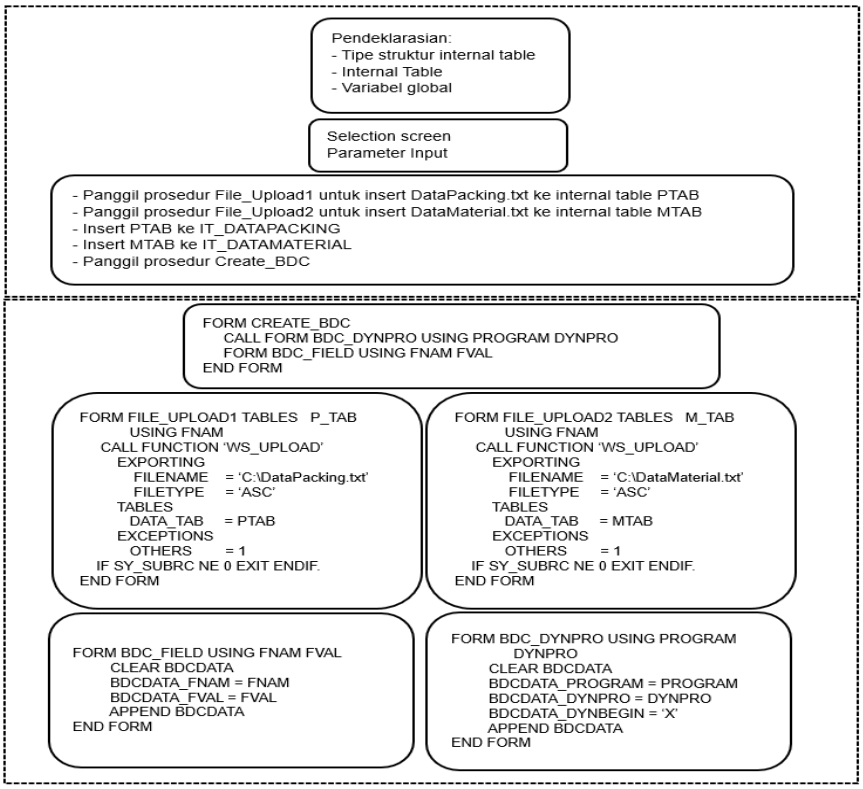
Gambar 6. menjelaskan tentang struktur program *Data Migration Interface*. Terdapat dua blok besar. Blok pertama merupakan blok untuk program utama. Blok ke dua berisi sejumlah *sub procedure* yang dipanggil pada program utama. *Sub procedure* FILE_UPLOAD1 berfungsi untuk melakukan *upload* data DataPacking.txt dari *local disk* dan melakukan *insert* ke *internal table* PTAB. Sedangkan *sub procedure* FILE_UPLOAD2 berfungsi untuk melakukan *upload* data DataMaterial.txt dan melakukan *insert* ke *internal table* MTAB. *Sub procedure* BDC_DYNPRO dengan parameter PROGRAM dan DYNPRO merupakan prosedur untuk melakukan *insert* nomor dan nama *screen* berdasarkan hasil rekaman ke dalam *table* BDCDATA. *Sub procedure* BDC_FIELD dengan parameter FNAM dan FVAL merupakan prosedur untuk melakukan *insert* nama *field* dan nilai *field* berdasarkan hasil rekaman ke dalam *table* BDCDATA. Kedua sub prosedur ini dipanggil dari sub prosedur CREATE_BDC. Jadi pengisian *table* BDCDATA berdasarkan hasil rekaman dilakukan pada *sub procedure* CREATE_BDC. Setelah dilakukan *insert* ke BDCDATA terhadap semua hasil rekaman, maka dipanggil transaksi VL02N untuk menjalankan proses secara *background.*

Tabel 4. Urutan *step* (langkah) perekaman proses *picking* dan *packing* pada tcode VL02N

Step	Keterangan	Step	Keterangan
1	Masukkan nomor order misalnya 3340030503 	14	Klik tombol PackHus 
2	Tekan Enter	15	Masukkan item yang di-*pack* ke dalam kontainer dengan cara pilih kontainer beserta dengan semua *handling unit* yang dibuat
3	Cari nomor material (item) yang sesuai pada report Packing List	16	Klik *icon pack*
4	Ganti satuan berat dari TO menjadi KG untuk setiap item	17	*Double click* nomor kontainer dan masuk *screen* baru
5	Tekan Enter setiap selesai mengganti TO menjadi KG. step 3 dan 4 diulang untuk setiap item	18	Cari nomor kontainer yang sedang diproses
6	Pilih menu Edit → Pack. Klik tombol Pack Material	19	Klik tab Status
7	Pada *sub screen* All existing Hus available for packing isi kolom Packing Materials dengan *dummy* 900000	20	Klik User Status
8	Tekan Enter	21	Pilih no. 03
9	Cari material yang bersesuaian untuk *dummy* tersebut pada *sub screen* Material to be Packed	22	Isi Contents dengan nomor seal
10	Isi total berat pada kolom Partial Qty	22	Klik tab Means of transp. Data
11	Pilih *dummy* pada *sub screen* All existing Hus (available for packing) bersamaan dengan material yang sesuai pada *sub screen* Material to be Packed	24	Isi Driver dengan nama supir
12	Klik *pack icon*, ulangi mulai step 7 untuk semua item	25	Isi Alt.Driver dengan nomor mobil
13	Create kontainer misalnya BSIU9032500 pada *sub screen* All existing Hus (available for packing)	26	Klik *icon save* untuk menyimpan hasil transaksi

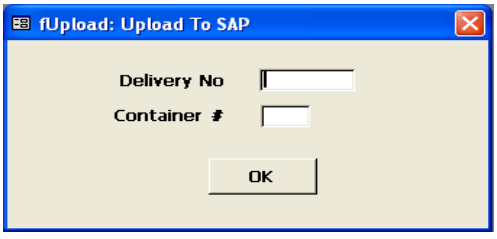
Tabel 5. Contoh isi tabel BDCDATA berdasarkan perekaman proses ganti TO ke KG

1	PROGRAM	DYNPRO	DYNBEGIN	FNAM	FVAL	KETERANGAN
1	SAPMV50A	4004	X			
2				BDC_OKCODE	/00	Tekan enter
3				LIKP-VBELN	3340030503	Entry nomor order
4	SAPMV50A	1000	X			
5				BDC_OKCODE	/00	Tekan enter
6				LIPS-VRKME(09)	KG	Ganti TO ke KG pada baris 9
7	SAPMV50A	1000	X			
8				BDC_OKCODE	/00	Tekan enter

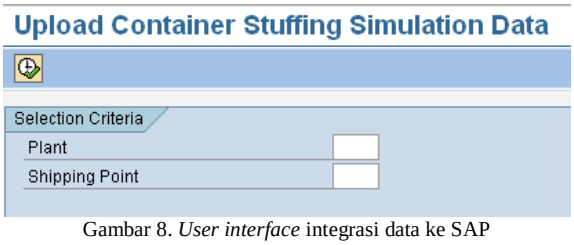


Gambar 6 *Block diagram* program *Data Migration Interface*

Terdapat dua *user interface* yaitu *user interface* pada program CSO dan *user interface* pada program *Data Migration Interface* di SAP. *User interface* pada program CSO untuk proses penyiapan data *packing* material atau pembuatan *flat file* yang berisi data yang akan diintegrasikan ke SAP. Pada *interface* ini, *user* cukup memasukkan nomor order lalu klik OK, maka akan dibentuk *flat file* untuk menampung semua data yang akan diintegrasikan ke SAP (Gambar 7)



Gambar 7. *User interface* pada program CSO



Gambar 8. *User interface* integrasi data ke SAP

User interface program *Data Migration Interface* (Gambar 8.) yang ada pada sistem SAP dipergunakan untuk proses integrasi atau migrasi data dari program CSO ke SAP.

3.3. Pengujian Sistem

3.3.1 Pengujian *Black Box*

Pengujian *black box* hanya dilakukan dengan mengeksekusi unit atau modul pada program, lalu diamati apakah hasil dari modul itu sesuai dengan proses bisnis yang diharapkan. Pengujian dilakukan untuk memastikan bahwa suatu *event* atau modul akan menjalankan proses yang tepat dan menghasilkan *output* sesuai dengan rancangan.

Tabel 8. Hasil *pre test* kuesioner (sebelum implementasi)

No.	Nama	Pertanyaan						Total
		1	2	3	4	5	6	Waktu/kontainer (menit)
1	Hendri Wijokongko (A)	3	6	5	1	1	15 menit	(3x6) x 15 = 270
2	Gozali Tamam (B)	4	5	5	5	2	20 menit	(4x5) x 20 = 400
3	Agus Dwi (C)	4	6	4	4	1	20 menit	(4x6) x 20 = 480
4	Effendi (D)	3	6	5	4	2	20 menit	(3x6) x 20 = 360
	Rata-rata							**18.875**

Tabel 9. Hasil *post test* kuesioner (sesudah implementasi)

No.	Nama	Pertanyaan						Total
		1	2	3	4	5	6	Waktu/kontainer (menit)
1	Hendri Wijokongko (A)	3	6	5	0	0	5 menit	(3x6) x 5 = 90
2	Gozali Tamam (B)	4	5	5	0	0	5 menit	(4x5) x 5 = 100
3	Agus Dwi (C)	4	6	4	0	0	5 menit	(4x6) x 5 = 120
4	Effendi (D)	3	6	5	0	0	5 menit	(3x6) x 5 = 90
	Rata-rata							**5**

Tabel 6. menjelaskan pengujian *black box* yang dilakukan terhadap modul pembentukan *flat file* dan modul pada program *Data Migration Interface*. Pada sistem CSO tersedia tombol OK pada *user interface*. Jika tombol itu diklik, maka sistem akan menjalankan proses pembentukan *flat file* yang berisi data dari CSO yang akan ditransfer atau dimigrasikan ke sistem SAP. Pada *user interface* di sistem SAP terdapat tombol yang berupa *icon* jam. Jika *icon* itu diklik, maka sistem SAP akan menjalankan tcode YSRQ301T untuk melakukan otomatisasi proses *picking* dan *packing* untuk menghasilkan surat jalan sesuai dengan data yang dimigrasikan dari sistem CSO.

Tabel 6. Hasil pengujian *black box*

Input	Proses	Sistem	Output	Hasil
Penekanan tombol OK	Private Sub cmdOK_Click ()	CSO	Menghasilkan *flat file*	Sesuai
Penekanan tombol 	Eksekusi program YSRQ301T	SAP	Otomatisasi proses *picking* dan *packing*	Sesuai

3.3.2 Acceptance Testing

Untuk mengetahui perbandingan kinerja sistem sebelum dan sesudah implementasi khususnya dalam hal kecepatan proses *picking* dan *packing* untuk mencetak surat jalan, maka diberikan kuesioner kepada *user*. Kuesioner cukup diisi oleh empat orang admin yang mewakili setiap kelompok *shift* dan grup (A, B, C dan D) pada bagian pengiriman barang. Daftar pertanyaan berupa waktu proses dan tingkat kesalahan input data yang diberikan kepada responden pada waktu sebelum dan sesudah implementasi program seperti yang ada pada Tabel 7.

Jawaban terhadap pertanyaan tersebut akan dianalisa sehingga diperoleh angka berapa lama rata-rata user melakukan proses picking dan packing untuk pembuatan satu surat jalan untuk setiap kontainer. Tabel 8. pada kolom 6, menampilkan angka waktu yang dibutuhkan untuk pembuatan surat jalan per kontainer.

Tabel 7. Kuesioner kinerja sistem

No.	Pertanyaan
1	Berapa rata-rata order yang harus ditangani setiap *shift*?
2	Berapa rata-rata kontainer yang harus dimuat untuk setiap order?
3	Berapa rata-rata item yang harus dimuat untuk setiap kontainer?
4	Berapa rata-rata dalam satu bulan terjadi kesalahan input jumlah dan spesifikasi barang yang dimuat ke dalam kontainer
5	Berapa rata-rata dalam satu bulan terjadi komplain dari pelanggan karena salah dalam pengiriman barang?
6	Berapa rata-rata waktu yang diperlukan untuk membuat satu surat jalan?

Jika pada grup A ada 3 order dan setiap order rata-rata terdiri dari 6 kontainer (6 surat jalan) serta pembuatan surat jalan untuk 1 kontainer rata-rata 15 menit, maka total waktu yang diperlukan adalah (3x6) x 15 menit = 270 menit. Rata-rata waktu yang diperlukan untuk membuat satu surat jalan setiap hari adalah 18.875 menit.

Tabel 9. merupakan hasil jawaban kuesioner setelah implementasi program. Pada kolom 4 dan 5 tentang rata-rata terjadi kesalahan input dan komplain pelanggan sudah tidak ada. Rata-rata waktu yang diperlukan untuk membuat satu surat jalan adalah 5 menit termasuk proses cetak surat jalan.

Berdasarkan waktu rata-rata untuk membuat satu surat jalan sebelum dan sesudah implementasi, maka terdapat peningkatan kecepatan (*speed up*) sebesar 18.875/5 = 3.775 kali. Terjadi efisiensi waktu sebesar 3 kali setelah implementasi, juga tidak terjadi lagi kesalahan dalam memasukkan data ke sistem, karena sudah terjadi sinkronisasi data *order* dari sistem CSO ke SAP.

KESIMPULAN

Migrasi data *order* hasil pemuatan kontainer pada sistem CSO (*legacy system*) ke SAP dapat melakukan sinkronisasi data *order* pada *Sales Order* (SO) dengan aktual pengiriman pada surat jalan atau *Delivery Note (DN)*. Hal ini akan meningkatkan

keakuratan dan konsistensi data. Migrasi data mengurangi jumlah komplain pelanggan karena data pada SO sudah sesuai dengan DN. Otomatisasi proses *picking* dan *packing* pada SAP untuk mempercepat proses pembuatan surat jalan karena program dapat mengurangi langkah proses yang semula 26 langkah (Tabel 4) menjadi hanya 3 langkah dengan memasukkan Plant dan Shipping Point pada program di SAP. Terjadi efisiensi waktu atau peningkatan kecepatan pembuatan surat jalan sebesar 3 kali lebih cepat setelah implementasi. Proses migrasi dan otomatisasi ini efektif untuk mendukung *On Time Delivery* ke pelanggan.

DAFTAR PUSTAKA

HUSSEIN, A. A. (2021). Data Migration Need, Strategy, Challenges, Methodology, Categories, Risks, Uses with Cloud Computing, and Improvements in Its Using with Cloud Using Suggested Proposed Model (DMig 1). *Journal of Information Security*, 79-103.

ABU BAKAR, HUMAIRATH KM., RAZALI, ROZILAWATI., JAMBARI, DIAN INDRAYANI., 2020. A Guidance to Legacy Systems Modernization. International Journal on Advanced Science Engineering Information Technology, 3(10), p.1042-1050

SOMMERVILE, IAN, 2011. Software Engineering. 9th ed. Boston: Pearson Education, Inc.

MUTIA DWI RACHMANINGTYAS, RIFKY SYARIPUDIN, & AJI ANGGONO. (2021). Analisis Migrasi Data dalam Implementasi SAP Financial and Controlling pada PT POS Indonesia. *Syntax Literate: Jurnal Ilmiah Indonesia*, *6*(8), 3941-3952.

GEEKSFORGEEKS, 2022. Software Engineering| Classical Waterfall Model, [online] Tersedia di https://www.geeksforgeeks.org/software-engineering-classical-waterfall-model/ [Diakses 15 Maret 2023]

SYAIFUL BAKHRI, 2019. Evaluasi Faktor-Faktor yang Mempengaruhi Performa Implementasi SAP ERP di Industri Retail. Walisongo Journal of Invormation Technology, 1(2), p.111-124.

KELLTON TECH, 2022. How Data Migration in SAP Drives Maximum Transformation Value.

RED GLOBAL, 2022. What is SAP Software Used For, [online] Tersedia di: https://www.redglobal.com/news-blog/what-is-sap-software-used-for [Diakses 20 Maret 2023]

DENSBORN, F., FINKBOHNER, F., GRADL, J., ROTH, M. & WILLINGER, M., 2016. Data Migration with SAP. 3rd ed. Boston: Rheinwerk Publishing Inc.

R. KHADKA, 2016. Revisiting Legacy Software System Modernization. Utrecht University.

LIZBETH WILKINS, 2018. White Paper Legacy Modernization: Application Management & Modernization, [online]. Tersedia di: https://silo.tips/download/white-paper-legacy-modernization-product-solution-application-management-moderni [Diakses 28 Maret 2023]

SHANNON J., BARNES, 2022. How to Choose the Right Legacy System Modernization Strategy for Your Business, [online] Tersedia di: https://www.orientsoftware.com/blog/legacy-system-modernization/ [Diakses 28 Maret 2023]

E. LANGBAUER, 2019. Factors Influencing the Legacy System Modernization Strategy. Johannes Kepler Unicersity Linz

S. MARKANDEYA, 2017. Pro SAP Scripts, Smartforms and Data Migration: ABAP Programming Simplified. New York: Springer Science + Business Media Finance Inc.

SAP DOCUMENTATION, 2015. Batch Input: Concepts, [online] Tersedia di: https://help.sap.com/doc/saphelp_snc70/7.0/en-US/69/c250344ba111d189750000e8322d00/frameset.htm [Diakses 30 Maret 2023]

TCODESEARCH.COM. BDCDATA Table in SAP: Batch Input: New table field structure [online] Tersedia di: https://www.tcodesearch.com/sap-tables/BDCDATA [Diakses 30 Maret 2023]

TUTORIALSPOINT.COM. SAP – Transaction Codes [online] Tersedia di: https://www.tutorialspoint.com/sap/sap_transaction_codes.htm [Diakses 30 Maret 2023]

UCHE NNENE. (2020). Legacy Data Migration. In *Practical Guide to Data Migration with SAP S/4 HANA Migration Cockpit* (pp. 43-49). Gleichen, Germany: Espresso Tutorials.

ADAM MENTSIEV, TIMUR AYGUMOV, & ELMIRA AMIROVA. (2023). Data-Driven Digital Transformation: Challenges and Strategies for Effective Big Data Management. *RT&A*, *18*(5), 526-531.

VINDHA NOVRIANI TANJUNG, MUHARDI SAPUTRA, & WARIH PUSPITASARI. (2019). Analysis of Assessment Cycle of Migration Data from OROS to SAP Hana using Activity based Costing Method in Telecommunication Industry. *Conference: International Conferences on Information System and Technology*, (pp. 253-260).