Jurnal ICT : Information Communication & Technology
Vol. 24, N0.2, Desember 2024, pp. 146-151
p-ISSN: 2302-0261, e-ISSN: 2303-3363

Design Middleware untuk Koneksi dan Interoperability Aplikasi dan Database di Bank

David Jefri Aruan[1*], Jan Everhard Riwurohi[2], Bagus T Prabawa[3]
[1]Program Studi Magister Ilmu Komputer, Universitas Budi Luhur, Indonesia
[2]Program Studi Teknik Informatika, Universitas Budi Luhur, Indonesia
[3]PT Collega Intri Pratama, Jakarta - Indonesia
Email: [1]2411600675@student.budiluhur.ac.id, [2]yan.everhard@budiluhur.ac.id, [3]bagustp@gmail.com

INFORMASI ARTIKEL

Histori artikel:

Naskah masuk, 20 November 2024
Direvisi, 21 Desember 2024
Diiterima, 30 Desember 2024

Kata Kunci:

Digital bank,
ISO 8583,
JSON,
WSDL,
Middleware

ABSTRAK

Abstract- *In the digital banking industry, banking activities ranging from opening an account, transfers, deposits, to closing an account no longer have to go directly through a bank office or via an ATM (Automatic Teller Machine) but can be done via a smartphone/electronic device. In fact, the trend of digital transactions is increasing, this is confirmed by ASPI in its statistical report. The communication message formats commonly used in banking today are ISO 8583, JSON and WSDL. Indonesia developed BIFAST for payment system infrastructure. Middleware was developed to help solve the interconnection of several applications and interoperability problems. Middleware is needed to migrate from mainframe applications to client/server applications and also to provide communication between different platforms.*

Abstrak- Dalam industri perbankan digital aktivitas perbankan mulai dari pembukaan akun atau rekening, transfer, deposito, hingga penutupan akun tidak lagi harus melalui langsung via kantor bank atau via ATM (Anjungan Tunai Mandiri atau *Automatic Teller Machine*) namun sudah dapat melalui *smartphone*/perangkat elektronik. Bahkan perkembangan transaksi via digital trendnya semakin meningkat hal ini terkonfirmasi oleh ASPI dalam laporan statistiknya. Format pesan komunikasi yang umum dipakai di perbankan saat ini adalah ISO 8583, JSON dan WSDL. Indonesia mengembangkan BIFAST untuk infrastruktur sistem pembayaran. *Middleware* dikembangkan untuk membantu memecahkan interkoneksi beberapa aplikasi dan masalah interoperabilitas. *Middleware* sangat dibutuhkan untuk bermigrasi dari aplikasi mainframe ke aplikasi *client/server* dan juga untuk menyediakan komunikasi antar *platform* yang berbeda.

Penulis Korespondensi*:*

Bagus T Prabawa
PT Collega Inti Pratama
Talavera Office Park, Jl. TB Simatupang, Cilandak, Kota Jakarta Selatan, Jakarta 12430
Email: bagustp@gmail.com

1. Pendahuluan

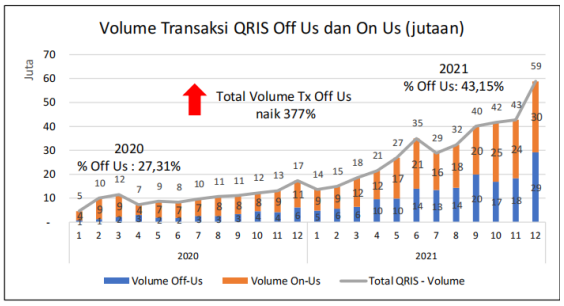
Perkembangan teknologi *Internet of Things* (IoT) dan kecerdasan buatan semakin krusial semenjak pertengahan tahun 2010-an, teknologi ini mengakselerasi perubahan atau perkembangan industri secara luas. Perubahan ini merupakan hasil dari proses evolusi teknologi secara bertahap dan berkesinambungan, dan Revolusi Industri 4.0 dapat dianggap sebagai suatu periode transisi yang sedang berlangsung hingga saat ini.

Dalam industri perbankan, tuntutan digitalisasi perbankan menuntut bank konvensional menyediakan layanan digital seperti m*obile banking* dan *internet banking*, serta makin tumbuhnya Bank digital dimana bank digital dapat melakukan semua aktivitas perbankan mulai dari

pembukaan akun atau rekening, transfer, deposito, hingga penutupan akun melalui smartphone/perangkat elektronik tanpa harus datang langsung ke kantor bank. Bahkan sesuai Peraturan Otoritas Jasa Keuangan No. 12 Tahun 2021, Bank Digital hanya punya kantor pusat tanpa kantor cabang secara fisik.

Bank konvensional biasanya atau umumnya untuk bertransaksi melalui transaksi langsung via kantor bank atau via ATM (Anjungan Tunai Mandiri atau *Automatic Teller Machine*). Secara sistem bank menerapkan sistem *core banking* untuk mencatat semua data nasabah, rekening dan transaksi keuangan, sistem *switching* untuk transaksi via mesin ATM atau EDC (*Electronic Data Capture*).

Perkembangan transaksi digital semakin berkembang, ini dapat dilihat dari data berita statistik sistem pembayaran Indonesia (ASPI) pada triwulan 1 tahun 2022 [16]. Sesuai diagram di bawah ini, menunjukan bahwa menunjukkan trend kenaikan yang sangat signifikan. Secara total, volume transaksi QRIS mencatat pertumbuhan sebesar 201,90% dibanding tahun 2020. Total volume transaksi QRIS pada tahun 2021 mencapai 374,69 juta transaksi. Fenomena yang sama juga tampak pada total nominal transaksi QRIS yang pada tahun 2021 mencatat pertumbuhan sebesar 236,72% dibandingkan tahun 2020. Pada tahun 2021, total nominal transaksi QRIS mencapai IDR 27,63 T



Gambar 1: Volume transaksi QRIS

Kebutuhan untuk interkoneksi dan *interoperability* antar aplikasi dan database menjadi suatu keniscayaan. Pengembangan *middleware* yang diharapkan dapat memenuhi kebutuhan tersebut.

2. Tinjauan pustaka

2.1. ISO 8583

ISO 8583 ini banyak digunakan untuk transaksi financial terutama di dunia perbankan. ISO 8583 dapat dimanfaatkan untuk transaksi financial non bank. Implementasi ISO 8583 pada mesin ATM dan EDC pada saat transaksi, mesin ATM atau EDC akan mengirimkan *request* dalam bentuk ISO 8583 ke server untuk diproses.

Pesan berformat ISO 8583 adalah sebuah rangkaian data *alphanumeric* yang tersusun menurut aturan tertentu dan dibaca menurut aturan tertentu sesuai kaidah dari ISO 8583. Jika sebuah aplikasi ingin membungkus informasi ke dalam rangkaian data berformat ISO 8583 maka disebut melakukan *generate* (*packaging*) selanjutnya data ISO 8583 dapat dikirimkan ke host lawannya.

Sedangkan jika aplikasi menerima rangkaian data berformat ISO 8583 dan ingin membaca informasi yang terkandung didalamnya maka disebut melakukan *parsing* selanjutnya informasi tersebut diproses oleh aplikasi yang bersangkutan.

STRUKTUR FORMAT ISO 8385		
MTI	BITMAP	Data Elements
0210	F23A4401AAA09300 0000000010004000	165049481200006371010000000010000000102206431801809213431810221023601102103111031112450494812000063 71

Gambar 2: Struktur Format ISO 8583

Komponen format

- *MessageType Indicator* (MTI)

MTI memuat informasi mengenai versi ISO-8583, *message class, message function, dan message origin*. Lebar data dari MTI adalah 4 bit.

- *Bitmap*

Bitmap memuat informasi mengenai pesan yang menunjukkan data elements yang digunakan. Lebar data dari *bitmap* adalah 8 *byte*.

- *Data Elements*

Data elements memuat informasi transaksi. Lebar data dari *data elements* adalah 16 *byte*.

Berbagai jenis transaksi seperti melakukan cek informasi saldo, *authorization*, verifikasi, pembayaran melalui *point of sale*, penarikan tunai, transfer, dan pengembalian memiliki formatnya masing-masing, bergantung pada data element nomor 24 dan MTI. MTI untuk pembayaran melalui *point of sale*, penarikan tunai , pembayaran, transfer, dan pengembalian adalah 200 (*request*) dan 210 (*response*).

Format ISO 8583 secara luas digunakan oleh pihak institusi keuangan dan non perbankan misalnya pendidikan untuk transaksi sistem pembayaran. Format ISO 8583 diimplementasikan dalam pengembangan *host to host* aplikasi perbankan ATM, *billing online* dan *payment gateway* [2][3] [4][5]

2.2. JSON

JSON (*JavaScript Object Notation)* adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena

menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

JSON merupakan format pertukaran data ringan yang mudah untuk dibaca dan ditulis oleh manusia, serta mudah di terjemahkan (*parse*) dan dibuat (*generate*) oleh komputer. Penulisan format JSON tidak bergantung pada salah satu bahasa pemrograman, sehingga JSON bisa digunakan sebagai bahasa pertukaran data antar bahasa pemrograman. Format JSON telah diimplementasikan dalam aplikasi untuk pertukaran data dalam aplikasi berbasis web dan aplikasi berbasis *mobile* [6][7][8][11].

Struktur data universal digunakan pada JSON, meliputi kumpulan pasangan nilai atau objek (*object*) dan daftar nilai terurutkan atau larik (*array*). Objek merupakan nilai yang tidak terurutkan. Penulisan objek dimulai dengan simbol { dan diakhiri dengan simbol }. Setiap nama diikuti dengan titik dua (:) dan koma (,) untuk memisahkan setiap pasangan nilai.

2.3. WSDL

Web Services Description Language (WSDL) adalah file berbasis XML yang pada dasarnya memberi tahu aplikasi klien apa yang dilakukan oleh layanan web. File WSDL digunakan untuk menjelaskan secara singkat apa yang dilakukan layanan web dan memberi klien semua informasi yang diperlukan untuk dapat terhubung ke layanan web dan menggunakan semua fungsionalitas yang telah disediakan.

Dokumen WSDL digunakan untuk menggambarkan layanan web. Deskripsi ini diperlukan agar aplikasi klien dapat memahami apa yang sebenarnya dilakukan oleh layanan web. File WSDL itu sendiri dapat terlihat sangat kompleks bagi pengguna, tetapi juga berisikan semua informasi yang diperlukan aplikasi klien untuk menggunakan layanan web yang relevan.

Menurut [9] WSDL merupakan sebuah bahasa berbasis XML yang digunakan untuk mendefinisikan *web-service* dan menggambarkan bagaimana cara untuk mengakses *web-service* tersebut.

Deskripsi WSDL mendefinisikan sebuah service sebagai kumpulan dari port dimana tiap-tiap port didefinisikan secara abstrak sebagai *portType* yang mendukung sekumpulan operasi-operasi. Tiap-tiap operasi memproses sekumpulan pesan tertentu.

2.4. BI-Fast

BI-FAST adalah infrastruktur sistem pembayaran yang disediakan Bank Indonesia yang dapat diakses melalui aplikasi yang disediakan industri sistem pembayaran dalam memfasilitasi transaksi pembayaran ritel bagi masyarakat. Implementasi BI-FAST oleh bank kepada nasabahnya akan dilakukan secara bertahap sesuai dengan rencana bank dalam mempersiapkan kanal pembayaran bagi nasabahnya masing-masing

Pengembangan BI-FAST adalah tonggak penting reformasi digitalisasi sistem pembayaran nasional sebagai implementasi BSPI 2025 bersama QRIS, SNAP, dan reformasi regulasi sistem pembayaran. BI-FAST merupakan inisiatif nasional (*national driven*) untuk menciptakan infrastruktur SP ritel yang lebih efisien, untuk memenuhi kebutuhan masyarakat dalam bertransaksi ekonomi dan keuangan yang cepat, mudah, murah, aman, dan andal, yang memperkuat konsolidasi industri SP nasional dan membangun ekonomi-keuangan digital yang integrated, interoperable & interconnected, dan membentuk unicorn-unicorn nasional yang tangguh.

BI-FAST akan menjadi backbone infrastruktur sistem pembayaran ritel masa depan, yang mengakselerasi pembayaran menggunakan berbagai instrumen dan kanal secara real time, aman, mudah, dan beroperasi 24/7. Implementasi BI-FAST bertujuan mewujudkan terciptanya layanan sistem pembayaran yang CEMUMUAH (Cepat, Mudah, Murah, Aman, Andal), untuk mengakselerasi pemulihan ekonomi dan mendorong pertumbuhan, serta inklusi ekonomi dan keuangan. Bank Indonesia terus memperkuat sinergi kebijakan dan implementasi BI-FAST dengan pelaku industri, dalam rangka mengintegrasikan Ekonomi Keuangan Digital (EKD) nasional [17].

2.5. *Advanced Encryption tandard* (AES)

Advanced Encryption Standart (AES) merupakan kriptografi algoritma kunci simetrimetris dengan panjang kunci dari AES bisa 16, 24, 32 *bytes* (128, 192 atau 256 bits).

Perbedaan panjang kunci akan mempengaruhi jumlah round yang akan diimplementasikan pada algoritma AES. Algoritma enkripsi AES terdiri dari operasi *Sub-Bytes, Shift-Rows, MIX-Columns,* dan *Add-key*. Diputaran terakhir operasi *Mix-Columns* ditiadakan [10]

2.6. *Keyed-Hash Message Authentication Code* (HMAC)

MAC atau *Message Authentication Code* adalah satu teknik autentikasi yang melibatkan penggunaan kunci untuk menghasilkan blok data berukuran tetap (*fixed-size block*). Sedangkan HMAC adalah teknik dari MAC dengan memanfaatkan fungsi hash terhadap pesan dan kemudian mengenkripsi pesan tersebut dengan sebuah kunci yang hanya diketahui oleh pengirim dan penerima [10]. Secara umum, pesan yang dikirimkan oleh pengirim (*sender*) dapat divalidasi

sebagai otentik (tidak dimodifikasi oleh pihak lain) oleh pihak penerima (*receiver*).

2.7. *Middleware*

Menurut Oracle, *middleware* adalah lapisan perangkat lunak yang terletak di antara sistem operasi dan aplikasi di setiap sisi jaringan komputer terdistribusi [1], sedangkan menurut technopedia middleware mirip dengan sistem operasi karena dapat mendukung program aplikasi lain, menyediakan interaksi terkontrol, mencegah gangguan antara komputasi dan memfasilitasi interaksi antara komputasi pada komputer yang berbeda melalui jaringan Layanan Komunikasi [12].

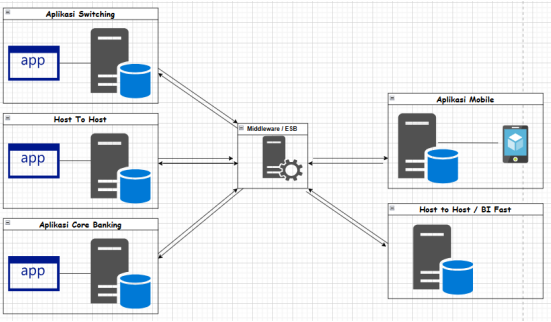
Middleware sebagai sebuah *software* yang terletak di antara aplikasi dan sistem operasi, jaringan atau database. *Middleware* memudahkan para pengguna dengan cara menyembunyikan detail database dan jaringan dari aplikasi yang terdistribusi. Dengan berkembangnya sistem komputasi dengan teknologi terbaru, middleware menjadi fasilitas penghubung dengan cara melakukan "*virtual homogenity*" di dalam sistem tersebut [19]. Saat ini, *middleware* juga dikembangkan dalam bentuk *portable* untuk berbagai macam platform sehingga *middleware* dapat berperan untuk menyamakan tampilan dan layanan dimanapun aplikasi tersebut dieksekusi [18].

Terdapat dua bentuk dukungan komunikasi di dalam *middleware* yaitu *Remote Procedure Call* (RPC) yang mencakup data transfer, network programming, dan failures. Sedangkan bentuk komunikasi yang kedua adalah dalam bentuk messaging. Layanan yang diberikan oleh middleware mencakup 8 layanan adalah: 1) *directory services 2) transaction services 3) security services 4) management services 5) event services 6) persistence services 7) load balancing 8) configuration services* [13].

Middleware dikembangkan untuk membantu memecahkan interkoneksi beberapa aplikasi dan masalah interoperabilitas. *Middleware* sangat dibutuhkan untuk bermigrasi dari aplikasi mainframe ke aplikasi *client/server* dan juga untuk menyediakan komunikasi antar *platform* yang berbeda [14][15].

3. Metode penelitian

Dalam penelitian ini akan merancang middleware yang digunakan untuk mengkoneksikan dan interoperabilitas database dari aplikasi yang terdapat dalam operasional harian bank. Terdapat database *Core Banking System*, database *Switching System*, database *Host to Host*, database *Mobile Applications*, dan database *Host to Host BI-Fast*.



Gambar 3: Flow interoperabilitas database aplikasi

Proses atau tahapan *Middleware* dalam proses interoperability data sebagai berikut:

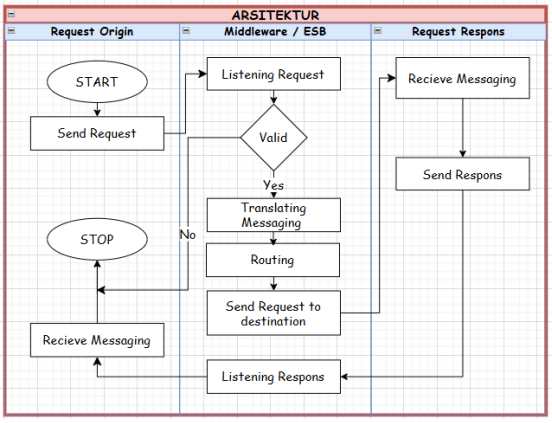
- Status *listening* untuk menangkap *request message*
- Menterjemahkan *request message* sesuai dengan format komunikasi data yang digunakan
- Melakukan *mapping* sesuai dengan format komunikasi dari request message asal ke *request message* tujuan
- Mengirimkan *request message* ke database tujuan.
- Menerima *response* dari *request* yang dikirimkan.

Tabel 1: Tabel routing message

Aplikasi	IP	Port	Status	Format
CBS	xxx.xxx.	5555	Client	JSON
Switching	xxx.xxx.	1234	Server	ISO8583
Mobile Banking	xxx.xxx.	4444	Server	JSON
Host to Host	xxx.xxx.	8888	Server	JSON
BI-Fast	xxx.xxx.	8889	Client	ISO20022

4. Hasil dan Pembahasan

Arsitektur *middleware* yang diusulkan dalam penelitian dapat dilihat dalam diagram di bawah ini:



Gambar 4: Arsitektur Middleware

Aplikasi dan database yang akan diintegrasikan didaftarkan *IP* dan *port* dalam parameter *middleware*. Aplikasi *switching*, aplikasi mobile banking, aplikasi *Core Banking System*, *Payment System* dan aplikasi *Corporate & Government System* didaftarkan masing-masing IP dan portnya.

- *Listening*

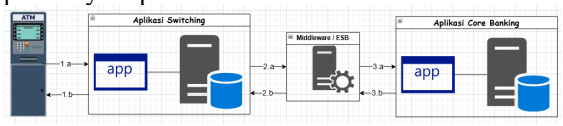
  Middleware mendengarkan setiap request yang muncul dari *IP* dan *Port* yang sudah didaftarkan. Untuk memastikan *request* tercatat maka dibuat dalam antrian / *queue*.
- *Validasi*

  Reguest dilakukan pengecekan apakah *request* tersebut valid atau tidak. Dilakukan pengujian autentikasinya. Jika *request invalid* maka akan direspon bahwa *request invalid*, namun jika valid maka *request* akan dilanjutkan ke tahap berikutnya.
- Translating

  Dilakukan proses *parsing* terhadap *request*, informasi diproses sesuai dengan format komunikasi tujuan. Dilakukan proses *generate* sesuai dengan format komunikasi tujuan. Seperti transaksi di ATM untuk tarik tunai maka akan ditranslate format ISO 8583 ke format JSON.
- Routing

  Request yang sudah *ditranslate* sesuai dengan format tujuan akan dikirimkan ke tujuan.

Dalam implementasi *Middleware* dapat diberikan ilustrasi transaksi Cek Saldo via ATM dimana alur prosesnya seperti di bawah ini:



Gambar 5: Flow request dari ATM

Kegiatan dari masing-masing tahapan:

- Tahap 1.a: *Message* dalam format NDC / bahasa mesin ATM dikirimkan ke aplikasi switching, melalui NDC *adapter message* dikonversi menjadi message dengan format ISO 8583.
- Tahap 2.a: *Request* dikirimkan ke *Middleware* dalam format ISO 8583. *Middleware* melakukan *parsing*, karena tujuannya ke aplikasi *core banking* dengan format JSON dilakukan *generate message* dengan format JSON.


```
0200F23A4401AAA0930000000000100040081650494812000063713000000000000000010220206030
17960090603102210236011021031110311124504948120000637 1=
30072210000000094640000000001ATM CABANG UTAMA JKT
IDN000A9E5F074481FF84F3068202740095058000040000 5F2A0203605F3401019A032410229C01409F
02060000000000009F0306000000000000 9F101C0101A000000000FA2A65CD000000000000000000000000
00000000000009F1A0203609F2608F397E631D0FC96259F3602071E9F370410AFAD5A9F2701809F3403
0200009F33036040209F3501145A0850494812000063714F07A0000006021010840 7A0000006021010
1808F397E631D0FC9625031110771<FS>000000000000<FS><FS>0<FS><FS>0<FS>
U32660C548139E140B4033FD3ED2E6C46<FS>1<FS>123456<FS>90091003862<FS>001
```


Gambar 6: *Request* dalam format ISO 8583

- Tahap 3.a: Mengirimkan *request* dengan message sudah dalam format JSON ke *core banking*.


```
{
        "acqId": "111",
        "arpc":
"820274009505800004000 05F2A0203605F3401019A032410229C01409F020600000000000009F030600
00000000009F101C0101A000000000FA2A65CD000000000000000000000000000000009F1A0203609
F2608F397E631D0FC96259F3602071E9F370410AFAD5A9F2701809F3403020000 9F33036040209F3501
145A0850494812000063714F07A0000006021010840 7A0000006021010",
        "cardNumber": "5049481200006371",
        "clientId": "collega-id",
        "fwdId": "111",
        "merchantType": "6011",
        "pinblock": "",
        "posEntry": "100",
        "referenceNumber": "000000009464",
        "settlementDate": "1023",
        "terminalId": "00000001",
        "terminalLocation": "ATM CABANG UTAMA                JKT IDN",
        "traceNumber": "017960",
        "transmissionDateTime": "2024-10-22T09:06:03+07:00"
}
```


Gambar 7: *Request* dalam format JSON

- Tahap 3.b: *Core banking* mengirim *response* dalam format JSON. *Middleware* melakukan parsing dan melakukan *generate message* dalam format ISO 8583.


```
{
        "transmissionDateTime": "2024-10-22T09:06:03+07:00",
        "settlementDate": "1023",
        "merchantType": "6011",
        "posEntry": "100",
        "traceNumber": "017960",
        "referenceNumber": "000000009464",
        "cardNumber": "5049481200006371",
        "pinblock": "",
        "acqId": "111",
        "fwdId": "111",
        "clientId": "collega-id",
        "accountNumber": null,
        "terminalId": "00000001",
        "terminalLocation": "ATM CABANG UTAMA                JKT IDN",
        "arpc":
"820274009505800004000 05F2A0203605F3401019A032410229C01409F020600000000000009F030600
00000000009F101C0101A000000000FA2A65CD000000000000000000000000000000009F1A0203609
F2608F397E631D0FC96259F3602071E9F370410AFAD5A9F2701809F3403020000 9F33036040209F3501
145A0850494812000063714F07A0000006021010840 7A0000006021010",
        "arqc": null,
        "responseCode": "05",
        "responseDescription[0].idn": "General Error",
        "responseDescription[0].eng": "General Error",
        "transactionAmount": null
}
```


Gambar 8: *Response* dalam format JSON

- Tahap 2.b: Mengirimkan *response* dalam format ISO 8583 ke aplikasi *switching* dan switching mengkonversi ke format NDC / bahasa mesin ATM dan dikirimkan ke mesin ATM yang melakukan *request*.
- Tahap 1.b: Menerima *response* dari *Switching* dalam format NDC / bahasa mesin ATM.

5. Kesimpulan

Dengan semakin banyak aplikasi dan database dengan format komunikasi yang berbeda-beda kebutuhan akan interkoneksi dan interoperabiliti menjadi sangat penting. Pengembangan *middleware* dapat menjawab kebutuhan dimaksud. Aplikasi dan *database* yang akan diintegrasikan, dicatatkan IP dan portnya dalam tabel parameter. Dibuatkan *routing* dan *translate* untuk masing-masing format komunikasi.

Daftar Pustaka

[1] Oracle. Overview of Oracle Fusion Middleware. 2019. Diambil kembali dari Oracle Help Center: https://docs.oracle.com/cd/E28280_01/core.1111/e10103/intro.htm

[2] Kuntoro Nanang, Ladjamuddin Siti Madinah. Host To Host Simulation Application Design On Atm Transactions With ISO 8583. Incomtech; 8 (1). 2019.
[3] Yunandar, Rahmat T. "Implementasi Iso 8583 Untuk Host to Host Mahasiswabina Sarana Informatika Melalui Channelbiller PT. Finnet Indonesia Berbasis Aix." Jurnal Khatulistiwa Informatika, vol. 4, no. 2, 2016, pp. 97-113, doi:10.31294/swabumi.v4i2.1019.
[4] Mohammad Ridwan, Djalal Er Riyanto, Eko Adi Sarwoko, "Implementasi ISO 8583 untuk billing online mahasiswa universitas Diponegoro melalui channel SMS Banking Bank Rakyat Indoensia berbasis AIX. 2014
[5] Anggraeni, W. (1). Aplikasi ISO 8583 Bridging pada Billing untuk Layanan Online Payment. Dinamik, 10(3). https://doi.org/10.35315/dinamik.v10i3.27. 2005
[6] Cokro, Robby. Web Services Menggunakan Format JSON. Respati. 14. 10.35842/jtir.v14i2.282. .2019
[7] Peng, Dunlu & Cao, Lidong & Xu, Wenjie. Using JSON for Data Exchanging in Web Service Applications. 7. 2011.
[8] Budi Warsito, Ary. Ananda, Ajeng & Triyanjaya, Dian. Penerapan Data JSON Untuk Mendukung Pengembangan Aplikasi Pada Perguruan Tinggi Dengan Teknik Restfull Dan Web Service. Technomedia Journal. vol.2.no.1 .2017.
[9] Shohoud, Y., "Introduction to WSDL", DevXpert Corporation. 2001,
[10] Stallings, W., Cryptography and Network Security: Principles and Practice, 5th edition. Cryptography and Network Security: Principles and Practice, 5th edition, 2010.
[11] Surya Guna, Nandana, Rahmatullah, Alam & Nur Rachman, Andi. IMPLEMENTASI JSON PARSING UNTUK PERTUKARAN DATA PADA APLIKASI VPN CLIENT BERBASIS MOBILE. Jurnal Nero.vol.8.no.1. 2023
[12] techopedia. Techopedia. 2019. Diambil kembali dari Middleware: https://www.techopedia.com/definition/450/middleware
[13] Vinoski, Steve. An Overview of Middleware. Springer-Verlag Berlin Heidelberg, pp. 35–51, 2004
[14] Suprihadi, S., Wijono, S., & Hartomo, K. D. . Analysis service architecture utilizing middleware for information services management systems. International Journal of Applied Science and Engineering Analysis Analysis, 21(2), 1–15. 2024. https://doi.org/doi.org/10.6703/IJASE.202406_21(2).001
[15] Widiyandari, Y. B. . Analisis Dan Perancangan Sistem Integrasi Middleware Pada Internet Banking. Dspace.Uii.Ac.Id, 1-57. 2022 https://www.aspi-indonesia.or.id/statistik-qris/. Diakses, 14 Oktober 2024.
[16] https://www.bi.go.id/ . Diakses, 14 Oktober 2024.
[17] Wisswani, N. W., & Wijaya, I. W. K. . Message oriented middleware for library's metadata exchange. Telkomnika (Telecommunication Computing Electronics and Control), 16(6), 2756–2762. 2018. https://doi.org/10.12928/TELKOMNIKA.v16i6.9475
[18] Mesmoudi, Y., Lamnaour, M., El Khamlichi, Y., Tahiri, A., Touhafi, A., & Braeken, A. A Middleware based on Service Oriented Architecture for Heterogeneity Issues within the Internet of Things (MSOAH-IoT). Journal of King Saud University - Computer and Information Sciences, 32(10), 1108–1116. 2020. https://doi.org/10.1016/j.jksuci.2018.11.011