Jurnal Informatika Ekonomi Bisnis

http://www.infeb.org

2023 Vol. 5 No. 1 Hal: 39- 47 e-ISSN: 2714-8491

Microservices Design and Development for the Basic Logistic Functionalities in the Unified Process

Benny Susanto[1]✉, Fitrah Rumaisa[2]

[1,2]Fakultas Teknik, Universitas Widyatama

mideel_neutral@yahoo.com

Abstract

The software development process has gone through significant change in the recent years. Now delivering a system that merely satisfies functional requirements is no longer enough. Companies want their systems to be easily maintained, scalable, fault tolerant, available, integrated, and other attributes. The microservice architecture was designed to create such a robust system where a big monolith system is broken down into multiple small services that communicate each other to fulfill the requirements. One area of functional requirements that exist and needed in almost all kind of companies is the logistic part. This paper will discuss how microservices architecture can be designed and implemented to support logistic functionality that can be reused in a lot of companies as the base system.

Keywords: Microservices, Web Services, Logistic, Enterprise Resource Planning, The Unified Process.

Abstrak

Proses pengembangan perangkat lunak telah mengalami perubahan signifikan dalam beberapa tahun terakhir. Sekarang memberikan sistem yang hanya memenuhi persyaratan fungsional tidak lagi cukup. Perusahaan ingin sistem mereka mudah dipelihara, dapat diskalakan, toleran terhadap kesalahan, tersedia, terintegrasi, dan atribut lainnya. Arsitektur layanan mikro dirancang untuk membuat sistem yang kuat di mana sistem monolit besar dipecah menjadi beberapa layanan kecil yang saling berkomunikasi untuk memenuhi persyaratan. Salah satu bidang persyaratan fungsional yang ada dan dibutuhkan di hampir semua jenis perusahaan adalah bagian logistik. Makalah ini akan membahas bagaimana arsitektur microservices dapat dirancang dan diimplementasikan untuk mendukung fungsionalitas logistik yang dapat digunakan kembali di banyak perusahaan sebagai sistem dasar.

Kata kunci: Layanan Mikro, Layanan Web, Logistik, Perencanaan Sumber Daya Perusahaan, Proses Terpadu.

1. Pendahuluan

Sebagian besar bidang bisnis memiliki berbagai kegiatan dasar yang serupa, apa pun bidangnya. Kegiatan ini terkait dengan fungsi logistik yang umumnya dibutuhkan di bidang tersebut. Logistik adalah proses perencanaan, pengelolaan dan pelaksanaan aliran dan penyimpanan barang dan informasi terkait yang efektif, kuat dari titik asal ke titik konsumsi, yang bertujuan untuk memenuhi kebutuhan pelanggan [1]. Logistik merupakan proses perencanaan, penerapan dan pemantauan efisiensi dan efektivitas aliran langsung dan balik serta penyimpanan bahan baku, bahan dalam proses, produk dan jasa, dan informasi terkait antara titik asal dan titik konsumsi untuk memenuhi persyaratan pelanggan [2]. Logistik sebagai pengelolaan semua kegiatan yang memfasilitasi pergerakan dan koordinasi penawaran dan permintaan dalam menciptakan manfaat waktu dan tempat [3].

Kegiatan ini meliputi pesanan pembelian dan penjualan, pembelian dan penjualan, pembayaran pembelian dan penjualan, pesanan penerimaan dan pengiriman barang [4]. Untuk memenuhi kebutuhan dalam memelihara informasi dan mendukung kegiatan dasar tersebut *Enterprise Resource Planning* (ERP) menjadi salah satu solusi bagi perusahaan tersebut dimana ERP mengintegrasikan berbagai area dan sub-sistem dalam sebuah perusahaan menjadi satu kesatuan *system* [5]. Dalam beberapa tahun terakhir, solusi ERP hadir, baik versi *free*, *open source*, maupun versi berbayar karena permintaan dari banyak perusahaan untuk memenuhi kebutuhan mereka [6].

Sistem ERP sebagai perangkat lunak yang cocok untuk fungsi manajemen bisnis yang mencakup pembelian, penjualan, manajemen stok, sumber daya manusia, sumber daya keuangan, dan layanan logistik yang menyediakan basis data terpadu bagi organisasi untuk diintegrasikan dan dikorelasikan dengan organisasi lain oleh serangkaian prosedur bisnis [7]. Integrasi dan korelasi ini membantu menciptakan lingkungan informasi yang transparan yang memungkinkan manajemen senior untuk membangun harapan berdasarkan informasi *real-time* yang menggambarkan proses organisasi dan keadaan ekonomi [8]. Fungsi Logistik Umum yang disediakan oleh perangkat lunak ERP disajikan pada Tabel 1.

Diterima: 09-01-2023 | Revisi: 25-01-2023 | Diterbitkan: 31-03-2023 | doi: 10.37034/infeb.v5i1.185

Tabel 1. Fungsi Logistik Umum yang disediakan oleh Perangkat Lunak ERP

Activity	Distribution	Manufacture	Hospital	Workshop	Restaurant	Online Shop
Purchase Order	✓	✓	✓	✓	✓	✓
Vendor Invoice	✓	✓	×	×	×	×
Vendor Payment	✓	✓	×	×	×	×
Purchase Return	✓	✓	✓	✓	✓	✓
Vendor	✓	✓	✓	✓	✓	✓
Sales Order	✓	✓	×	✓	✓	✓
Sales Invoice	✓	✓	✓	✓	✓	✓
Sales Payment	✓	✓	✓	✓	×	✓
SP Confirmation	✓	✓	✓	✓	×	✓
Billing	✓	✓	×	×	×	×
Stock Reservation	✓	✓	×	×	×	×
Delivery Order	✓	✓	×	✓	✓	✓
DO Confirmation	✓	✓	×	✓	✓	✓
Sales Return	✓	✓	✓	✓	✓	✓
Customer	✓	✓	✓	✓	✓	✓
Stock Adjustment	✓	✓	✓	✓	✓	✓
Transfer Stock	✓	✓	✓	✓	✓	✓
Incoming Stock	✓	✓	✓	✓	✓	✓
Outcoming Stock	✓	✓	✓	✓	✓	✓
Item	✓	✓	✓	✓	✓	✓
Service	×	×	✓	✓	×	×
Warehouse	✓	✓	✓	✓	✓	✓
Employee	✓	✓	✓	✓	✓	✓
User	✓	✓	✓	✓	✓	✓

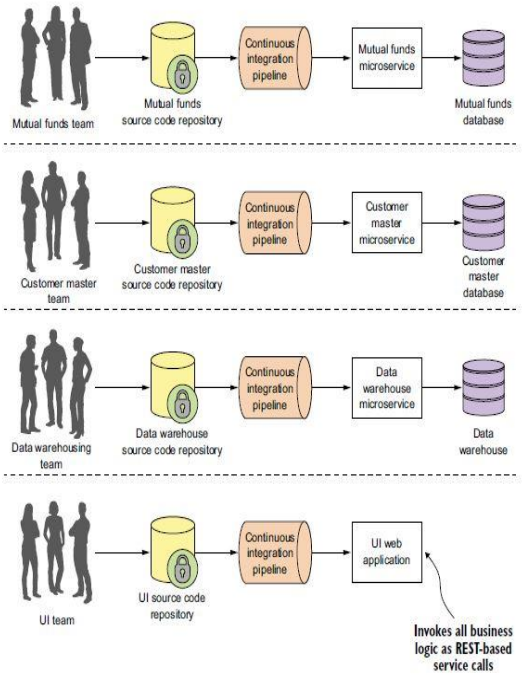
Sistem ERP sebagai seperangkat perangkat lunak yang menyediakan proses transfer data terintegrasi antara area fungsional yang berbeda untuk menyatukannya dalam satu lingkungan sistem, mengoordinasikan proses bisnis, meningkatkan struktur administrasi antar area fungsional untuk memastikan sistem informasi terintegrasi yang mencakup proses sistem dalam organisasi termasuk database yang terorganisir untuk meningkatkan stabilitas organisasi, menyediakan penyimpanan data yang mudah diakses dan digunakan dalam proses pengendalian stok, pembelian, penjualan, penagihan, dan akuntansi pinjaman pribadi [9].

Setelah penggunaan besar-besaran arsitektur yang menggunakan teknologi seperti layanan web, terutama sistem perangkat lunak dapat dibagi menjadi 2 bagian, yaitu *backend* dan *frontend* [10]. Di masa lalu, bagian *backend* dari keseluruhan sistem yang bertanggung jawab atas pemrosesan logika bisnis digunakan sebagai satu kesatuan dan monolit [11]. Artinya bagian backend hanya terdiri dari satu software utama, dan juga di-deploy ke satu server [12]. Server ini mencakup semua fungsi yang diperlukan dalam sistem itu, sistem yang satu ini juga melayani semua permintaan dari semua klien [13].

Namun arsitektur monolit ini memiliki beberapa kelemahan yaitu: Satu titik kegagalan; Sulitnya melakukan horizontal scalling; Kesulitan dalam mengintegrasikan kerja banyak tim; Umumnya semua tim dibatasi untuk menggunakan teknologi yang sama.

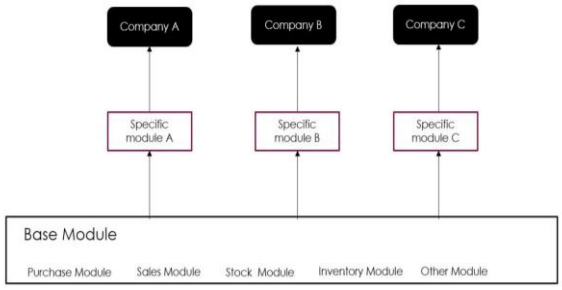
Setelah pengembang melewati semua masalah yang disebutkan di atas, arsitektur layanan mikro dikembangkan [14]. Arsitektur ini memiliki beberapa atribut yang berlawanan dengan sistem monolit. Bagian *backend* tidak digunakan sebagai satu perangkat lunak besar ke satu server, tetapi sistem dipecah menjadi beberapa layanan yang lebih kecil [15]. Ada satu layanan untuk menangani persyaratan untuk mengelola suku cadang pesanan pembelian dan layanan lain untuk menangani persyaratan untuk mengatur bagian pesanan pengiriman [16]. Dengan arsitektur ini, beberapa masalah yang disebutkan di atas dapat diselesaikan dengan *trade off* menambah kerumitan untuk merancang, mengimplementasikan, dan mengelola semua layanan yang lebih kecil ini dibandingkan dengan solusi monolit yang sederhana [17]. Pengertian *microservices* adalah Layanan mikro adalah komponen otonom yang mengisolasi kemampuan bisnis berbutir halus [18]. Selain itu, layanan mikro biasanya berjalan pada prosesnya sendiri dan berkomunikasi menggunakan antarmuka standar dan protokol ringan [19].

Layanan mikro didistribusikan, layanan perangkat lunak yang digabungkan secara longgar yang menjalankan sejumlah kecil tugas yang terdefinisi dengan baik [20]. Berdasarkan definisi di atas, dapat disimpulkan bahwa *microservices* adalah arsitektur sistem di mana bagian *backend* (yang terdiri dari logika bisnis dan bertanggung jawab untuk menangani permintaan klien) yang merupakan sistem monolit besar di pasar (tradisional) yang terdiri dari beberapa cakupan dan area fungsionalitas, dipecah menjadi beberapa layanan yang lebih kecil dan dipisahkan satu sama lain dalam hal penerapan [21]. Layanan ini berinteraksi satu sama lain untuk memenuhi fungsionalitas yang diperlukan dan mereka adalah satu kesatuan sistem. Kolaborasi beberapa tim pengembangan saat mengembangkan sistem layanan mikro akan ditampilkan pada Gambar 1.



Gambar 1. Kolaborasi beberapa tim pengembangan saat mengembangkan sistem layanan mikro.

Makalah ini membahas tentang desain awal arsitektur *microservices* untuk sistem logistik dasar yang dibutuhkan di sebagian besar bidang bisnis. Tujuan dari penelitian ini adalah untuk membuat rancangan sistem *backend microservices*. Sama seperti sistem monolit, sistem backend dasar harus dapat digunakan kembali dan dapat dengan mudah digunakan kembali untuk banyak sistem di berbagai bidang bisnis. Desain Sistem Inti Backend ditampilkan pada Gambar 2.



Gambar 2. Desain Sistem Inti Backend

Berdasarkan Gambar 2 dapat dilihat bahwa sistem dibagi menjadi 3 lapisan utama yaitu: Domain layer, domain layer merupakan high level layer dalam sistem dimana kelas-kelasnya terinspirasi/berbasis dari Domain Model; Lapisan teknis, adalah lapisan tempat logika umum berada dan digunakan oleh lapisan atas seperti Domain Layerr; Foundation layer, merupakan layer yang mengandung logika level rendah seperti akses databse dan utility library.

Setiap lapisan atas memiliki ketergantungan terhadap lapisan bawah, baik secara langsung maupun tidak langsung.

2. Metode Penelitian

Dalam penelitian/pengembangan ini, penulis menggunakan Unified Process sebagai kerangka proses. Pengembangan dilakukan sebanyak 1 iterasi pada fase inception dan 3 iterasi pada fase elaborasi. Setiap iterasi memiliki rentang waktu 3 minggu dengan tambahan interval 1 minggu antar iterasi. Unified Process adalah kerangka kerja proses untuk mengembangkan suatu sistem (atau tujuan pengembangan lainnya). Unified Process juga merupakan metodologi pengembangan yang bersifat iteratif dan bersifat fleksibel serta mengadopsi aktivitas/praktik lain dari metode agile lainnya seperti Scrum, XP, dll.

Umumnya setiap iterasi di UP memiliki rentang waktu 2 – 4 minggu. Dengan sifatnya yang iteratif, UP dirancang sedemikian rupa sehingga dapat beradaptasi dengan perubahan kebutuhan dari para pemangku kepentingan. Perubahan ini dapat disebabkan karena berbagai alasan, termasuk kesalahpahaman, temuan baru, peluang baru, atau perubahan tak terduga lainnya dari kebutuhan bisnis. Setiap iterasi dapat memberikan umpan balik dan masukan yang berarti dari semua pemangku kepentingan sehingga arah pembangunan dapat berada di jalur yang tepat untuk memenuhi tujuan utama. Ada 4 fase utama dalam UP, yaitu: Inception, ini adalah periode awal ketika tim pengembangan dan pemangku kepentingan melakukan tinjauan umum untuk memutuskan apakah pengembangan layak dilanjutkan atau tidak; Elaborasi, ini adalah fase di mana sebagian besar persyaratan dikumpulkan dan dipahami dengan baik secara iteratif melalui berbagai umpan balik dan masukan dalam berbagai iterasi. Fitur yang paling krusial dan penting juga dibuat di sini meskipun beberapa perubahan dan revisi masih dapat dilakukan; Konstruksi, ini adalah fase di mana sebagian besar fitur dikembangkan dan beberapa perubahan persyaratan tambahan juga terintegrasi; Transisi, ini adalah fase akhir dimana tim pengembangan fokus untuk menyempurnakan dan memberikan produk akhir kepada pengguna.

UP juga memiliki disiplin ilmu yang merupakan kategori kegiatan yang dilakukan pengembangan selama proyek berlangsung. Adapun disiplin ilmu yang dilakukan selama penelitian ini adalah Pemodelan bisnis, yaitu terkait dengan pemodelan bisnis; Requirements, untuk mengidentifikasi kebutuhan system; Analysis and Design, untuk membuat model desain yang nantinya akan diimplementasikan pada disiplin Implementasi. Perbedaan dari kedua model di atas adalah Model Analisis lebih pada desain konseptual dibandingkan dengan Model Desain yang merupakan desain sebenarnya; Implementasi, ini adalah kegiatan implementasi (coding secara umum); Test, hal ini berkaitan dengan pengujian sistem yang sedang dikembangkan.

Berbagai bidang bisnis dan perusahaan memiliki fungsi dan aktivitas dasar yang berkaitan dengan logistik dan banyak dari aktivitas ini mirip dengan sedikit perubahan kecil seperti pesanan pembelian dan

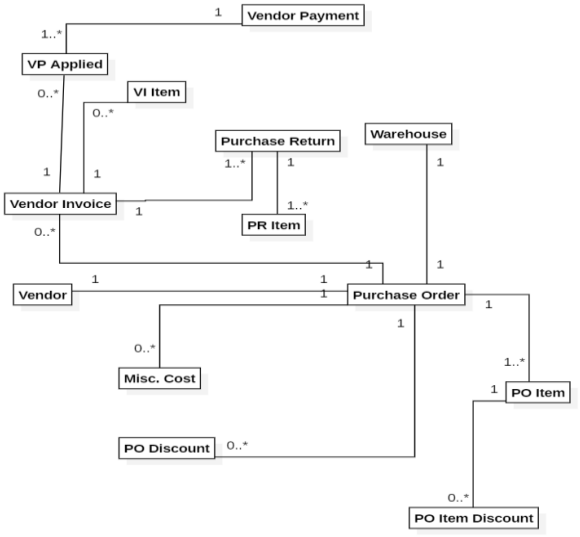
penjualan, pembayaran vendor dan penjualan, pesanan penerimaan dan pengiriman barang dan lain-lain. Fungsi umum ini dapat didukung oleh aplikasi seperti aplikasi ERP. Fungsionalitas dasar ini umumnya terkait dengan logistik karena sebagian besar bidang bisnis dan perusahaan memiliki area logistik. Penelitian ini diambil berdasarkan beberapa proyek yang telah diselesaikan yang melibatkan perusahaan lain. Perusahaan-perusahaan ini termasuk: Perusahaan penyalur; Perusahaan manufaktur; Rumah Sakit; Bengkel; Restoran; Toko online.

Dalam Terminologi RUP, Model Domain berguna untuk memahami objek-objek yang ada di dunia nyata (bukan objek desain perangkat lunak) yang terkait dengan sistem yang sedang dikembangkan. Model Domain mengilhami penciptaan Model Desain yang akan diimplementasikan oleh Pemrograman Berorientasi Objek (OOP) dalam bahasa pemrograman yang dipilih. Model Domain dibuat dalam disiplin Pemodelan Bisnis. Sub bab ini akan membahas beberapa artefak utama dan penting yang termasuk dalam Model Domain. Model Domain yang dibuat akan menginspirasi pembuatan Model Desain pada bab 3. Diagram tersebut sengaja hanya mengidentifikasi entitas tanpa menunjukkan atribut masing-masing yang akan dimasukkan ke dalam Model Desain. Paket-paket di Lapisan Domain mengelompokkan komponen sistem berdasarkan fungsionalitas umum yang ada di sebagian besar bidang bisnis sebagaimana yang disajikan pada Tabel 2.

Tabel 2. Kelompok Komponen Sistem Berdasarkan Fungsionalitas Umum

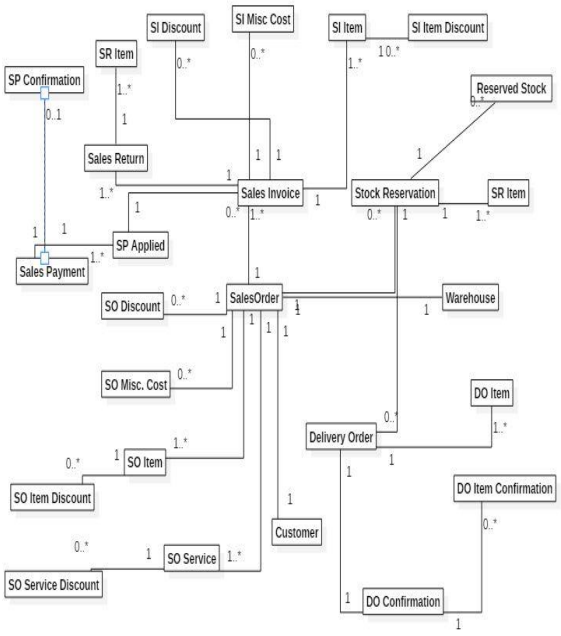
Discipline	Activities Summary	Artifact Type
Project Management	Reviewing the whole proporsal and making sure that the proporsal worths to be continued after considering the benefits and the trade offs	Software Development Plan (SDP)
Business Modelling	Understanding the main business process and the activities in a lot of the business fields that in general include the activities which are supported by common logistic functionalities of ERP softwares.	Business Use Case/Business Use Case Model Domain Model
Requirement	Understanding the overall requirements to fulfill the needs of the main business processes/activities in the business fields which are generally supported by common logistic functionalities of ERP softwarees	Vision Software Requirement Specification (SRS) Use Case/Use Case Model
Analysis and Design	Creating rough overall design to estimating the efforts in building the proporsed system.	Software Architecture Document (SAD)

Kelas utama dari gambar 3 adalah Purchase Order PO yang dapat memiliki beberapa vendor invoice. Diagram Kelas Pembelian akan ditampilkan pada Gambar 3.



Gambar 3. Diagram Kelas Pembelian

Secara umum, pelanggan dapat memulai pesanannya dengan pesanan penjualan dilanjutkan ke faktur penjualan. Domain Model Bagian Penjualan akan ditampilkan pada Gambar 4.



Gambar 4. Domain Model Bagian Penjualan

Pada Gambar 4 class diagram area penjualan. Secara umum, pelanggan dapat memulai pesanannya dengan pesanan penjualan dilanjutkan ke faktur penjualan.

3. Hasil dan Pembahasan

Sub bagian selanjutnya akan membahas kegiatan dan berbagai artefak dari proses yang dilakukan. Proses ini dipecah menjadi 4 bagian utama yaitu

a. Inception, ini adalah titik awal dimana tim menentukan persyaratan utama dan kasus

penggunaan untuk memutuskan apakah proyek harus dilanjutkan atau tidak;

b. Elaborasi – Iterasi 1, ini adalah Fase Elaborasi pertama dimana desain awal dan penting dilakukan;

c. Elaborasi – Iterasi 2, pada iterasi ini dibuat sebagian Model Implementasi sebagai proof of concept;

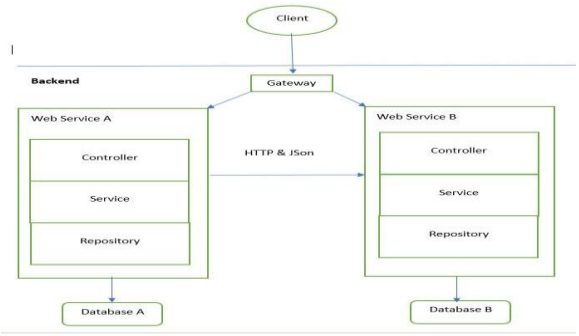
d. Elaborasi – Iterasi 3, pada iterasi ini dilakukan lebih banyak pengembangan implementasi sas, dilakukan juga sebagian pengujian.

Tahap inception bertujuan untuk melakukan pemeriksaan awal di tingkat atas dan beberapa bagian penting/kritis dari sistem yang diusulkan dan untuk memutuskan apakah proyek yang diusulkan harus dilanjutkan atau tidak. Beberapa disiplin ilmu yang dilakukan pada fase ini adalah Manajemen Proyek, Pemodelan Bisnis dan Kebutuhan. Disiplin Analisis dan Desain juga dapat dilakukan untuk memberikan gambaran awal tentang keseluruhan arsitektur yang akan dibangun. Elaborasi adalah fase di mana sebagian besar persyaratan utama diidentifikasi, direvisi, dan diselesaikan. Persyaratan pada tahap ini mengalami beberapa perubahan berdasarkan produk inkremental yang disampaikan dan setelah dievaluasi dengan pemangku kepentingan terkait. Tim pengembangan menerima umpan balik dari pemangku kepentingan bisnis untuk memastikan pengembangan berjalan sesuai rencana.

Disiplin Lingkungan dalam RUP meliputi kegiatan membangun lingkungan perlu melakukan pengembangan dan memutuskan teknologi yang dibutuhkan untuk melakukan implementasi. Hasil ini akan digunakan sebagai kendala untuk Disiplin Pelaksanaan. Sub bab ini akan membahas tentang alat dan teknologi yang dipilih dalam Disiplin Implementasi untuk mewujudkan tujuan yang telah diputuskan sebelumnya setelah mempertimbangkan berbagai trade off dan persyaratan dan setelah mengumpulkan berbagai informasi yaitu Pendapat para programmer dan penggunaan standar di industry; Informasi di internet; Pengalaman penulis; Tinjauan literatur. Pertimbangan tambahan juga mencakup adalah Masyarakat yang menggunakan teknologi; Kemudahan dalam pengembangan; Kemudahan dalam perawatan; Tingkat penguasaan penulis atas teknologi yang dipilih; Dokumentasi dan tutorial teknologi.

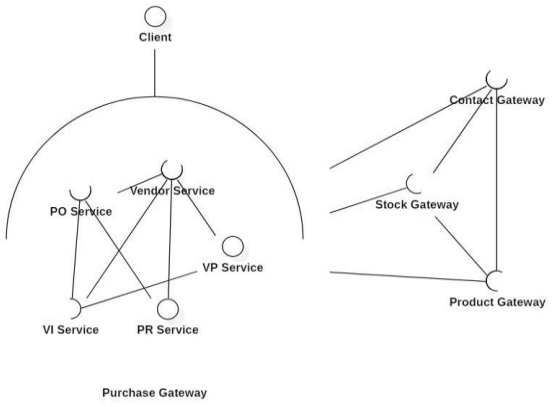
Maka, penulis memutuskan untuk menggunakan beberapa teknologi seperti yang tercantum di bawah ini, teknologi tersebut akan digunakan selama tahap implementasi adalah Java sebagai bahasa pemrograman utama; Pegas sebagai kerangka kerja backend; Spring Boot adalah sub proyek dari Spring Framework dan membuat pengembangan lebih mudah dengan fitur konfigurasi otomatisnya. Spring Cloud adalah proyek lain berdasarkan Spring Framework yang dibuat khusus untuk mengimplementasikan layanan mikro. PostgreSQL sebagai DBMS; HTTP sebagai protokol komunikasi antar layanan; JSON sebagai format data untuk berkomunikasi antar layanan; Maven sebagai alat untuk membangun proyek.

Selanjutnya membahas tentang perancangan layanan secara keseluruhan termasuk ketergantungan antar layanan. Seperti yang telah dibahas sebelumnya, untuk mengakses suatu sumber daya dalam suatu layanan, klien (termasuk layanan lainnya) menghubungi gateway dan permintaan dikirim secara tidak langsung ke gateway yang akan dikirimkan ke layanan tujuan. Gambar 5 mengilustrasikan arsitektur Controller-Service_Repository yang merupakan arsitektur standar yang digunakan oleh sistem yang dibangun menggunakan Spring Framework.



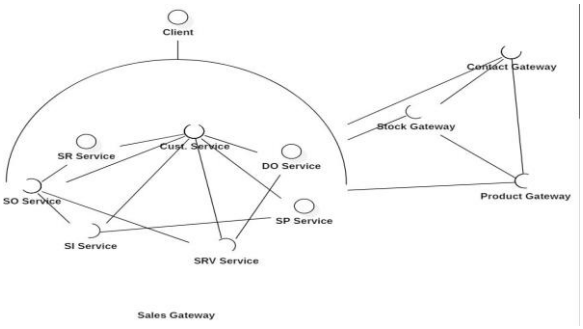
Gambar 5. Arsitektur Standar menggunakan Spring Framework

Pada Gambar 5 menjelaskan tentang Arsitektur Sistem dari Sudut Pandang Lapisan. Selanjutnya pada Gambar 6 mengilustrasikan desain layanan pada sub sistem pembelian secara keseluruhan. Layanan tersebut meliputi layanan untuk mengelola *Purchase Order* (PO), *Vendor Invoice* (VI), *Vendor Payment* (VP), *Purchase Return* (PR) dan Vendor.



Gambar 6. Ikhtisar Layanan Pembelian

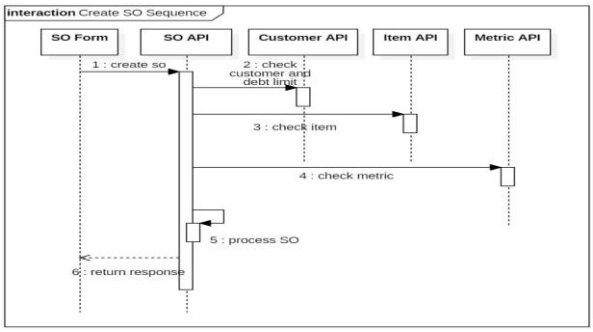
Gambar 6 menggambarkan rancangan layanan pada sub sistem penjualan secara keseluruhan. Layanan tersebut meliputi layanan untuk mengelola Sales Order (PO), Sales Invoice (VI), Sales Payment (VP), Stock Reservation (SR), Sales Return (PR) dan Customer akan ditampilkan pada Gambar 7.



Gambar 7. Ikhtisar Layanan Penjualan

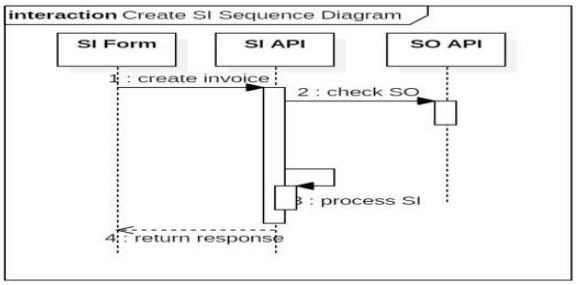
Dalam terminologi UP/RUP, Model Analisis adalah artefak yang digunakan sebagai desain konseptual tingkat tinggi dan juga bertindak sebagai realisasi use case. Model Analisis dapat berkembang menjadi Model Desain dan dibuang setelah itu dan juga dapat disimpan dan berdiri bersama dengan Model Desain.

Perlu diketahui bahwa pada bagian ini, inisiasi dilakukan oleh klien secara manual (dan admin menggunakan formulir/kondole), namun untuk tujuan kemudahan. Dalam sistem nyata, semua tindakan dapat dilakukan baik secara manual maupun otomatis setelah klien (pelanggan) mengirimkan permintaan pesanan penjualan. Selanjutnya akan ditampilkan hasil data Model Analisis Layanan Pesanan Penjualan pada Gambar 8.



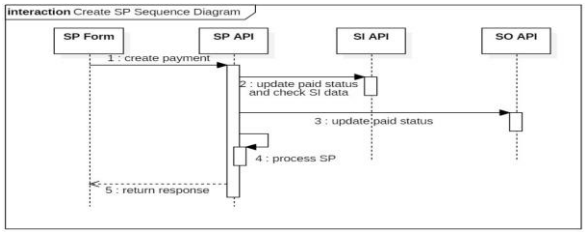
Gambar 8. Model Analisis Layanan Pesanan Penjualan

Gambar 8 mengilustrasikan interaksi yang terjadi untuk membuat Sales Order. Dimulai dari klien (Formulir SP) yang mengirimkan permintaan pembuatan pesanan penjualan, kemudian memverifikasi informasi pelanggan termasuk hutang, item, dan metrik dan kemudian mengembalikan respons. Selanjutnya Model Analisis Layanan Faktur Penjualan akan ditampilkan pada Gambar 9.



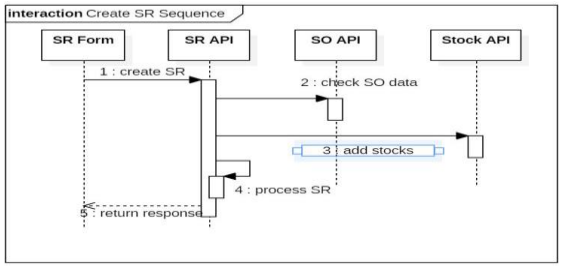
Gambar 9. Model Analisis Layanan Faktur Penjualan

Pada Gambar 9 mengilustrasikan interaksi yang terjadi untuk membuat faktur penjualan. Dimulai dari client (Formulir SI) yang mengirimkan request ke Sales Invoice API yang memverifikasi data sales order terkait melalui Sales Order API. Selanjutnya Model Analisis Layanan Pembayaran Penjualan akan ditampilakn pada Gambar 10.



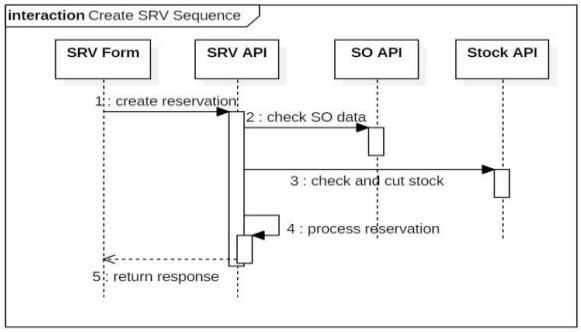
Gambar 10. Model Analisis Layanan Pembayaran Penjualan

Pada Gambar 10 mengilustrasikan interaksi yang terjadi antara layanan terkait saat membuat pembayaran penjualan. Klien (Formulir SP) mengirimkan permintaan untuk mengubah data pembayaran ke Sales Payment API yang akan memverifikasi data Faktur Penjualan dan melakukan pembaruan status pembayaran dan kemudian memverifikasi data Pesanan Penjualan terkait. Selanjutnya Model Analisis Layanan Retur Penjualan pada Gambar 11.



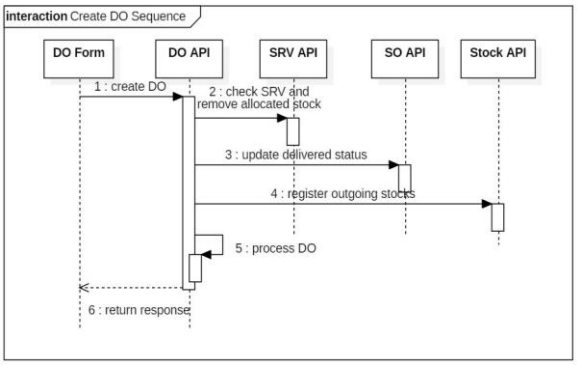
Gambar 11. Model Analisis Layanan Retur Penjualan

Pada Gambar 11 mengilustrasikan interaksi antar layanan yang terjadi saat membuat data retur penjualan. Diprakarsai oleh klien (Formulir SR) yang mengirimkan permintaan ke Sales Return API untuk membuat data retur penjualan yang kemudian akan memverifikasi data pesanan penjualan terkait dan mengurangi stok melalui Stock API. Selanjutnya Model Analisis Layanan Pemesanan Stok ditampilkan pada Gambar 12.



Gambar 12. Model Analisis Layanan Pemesanan Stok

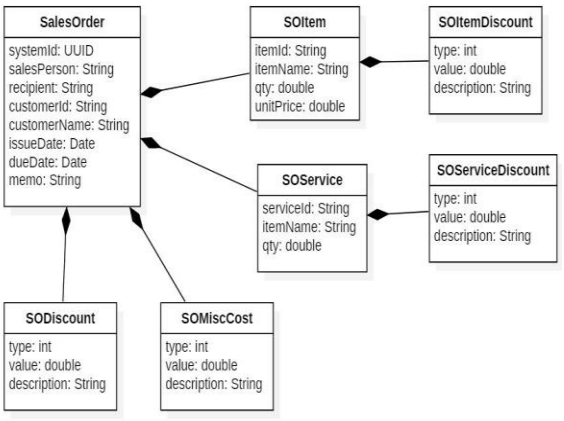
Pada Gambar 12 mengilustrasikan interaksi yang terjadi antar layanan saat membuat data stok rrservasi. Diawali dari klien (Formulir SRV) yang mengirimkan permintaan untuk membuat data pemesanan stok termasuk alokasi stok ke SRV API yang akan memverifikasi data pesanan penjualan terkait dan selanjutnya akan mengurangi stok. Selanjutnya Model Analisis Layanan Pesan Antar ditampilkan pada Gambar 13.



Gambar 13. Model Analisis Layanan Pesan Antar

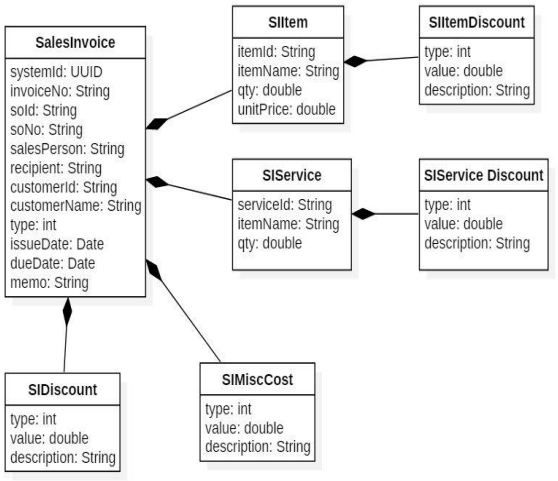
Pada Gambar 13 mengilustrasikan interaksi yang terjadi antar layanan untuk membuat data delivery order. Diprakarsai oleh klien (Do Form) yang akan mengirimkan permintaan untuk membuat data pesanan pengiriman ke Delivery Order API yang akan mengurangi stok yang dialokasikan melalui Stock Reservation API yang akan memperbarui status pesanan penjualan terkait melalui Sales order API , itu juga akan mendaftarkan stok keluar melalui Stock API.

Dalam terminologi RUP, seperti halnya Analysis Model, Design Model juga berperan sebagai realisasi use case dan merupakan abstraksi dari Implementasi Model. Berbeda dengan Model Analisis yang berfokus pada desain konseptual dan kasar, Model Desain fokus pada desain yang akan menjadi kendala untuk Implementasi Model. Design Model pada umumnya terinspirasi dari Domain Model jika dibuat sebelumnya. Sub bab ini akan membahas tentang desain kelas/entitas pada masing-masing layanan. Selanjutkan Diagram Kelas Layanan Pesanan Penjualan ditampilkan pada Gambar 14.



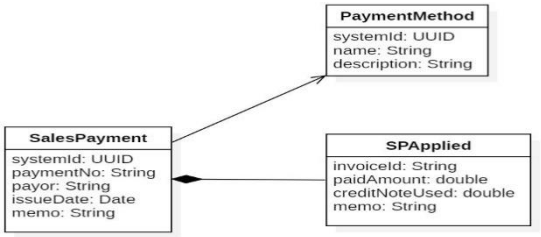
Gambar 14. Diagram Kelas Layanan Pesanan Penjualan

Pada Gambar 14 menjelaskan class diagram untuk Sales Order Service. Entitas utama dalam diagram kelas ini adalah Kelas Pesanan Penjualan yang mencakup beberapa item, layanan, dan diskon. Item dan layanan itu sendiri memiliki diskonnya sendiri. Selanjutkan Diagram Kelas Layanan Faktur Penjualan ditampilkan pada Gambar 15.



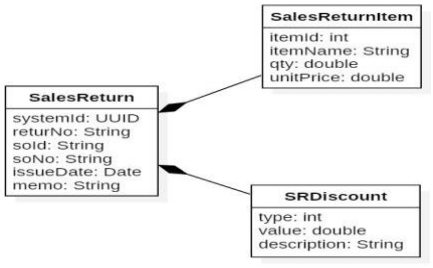
Gambar 15. Diagram Kelas Layanan Faktur Penjualan

Pada Gambar 15 menjelaskan diagram kelas untuk Layanan Faktur Penjualan. Entitas utama dalam diagram ini adalah Kelas Faktur Penjualan yang mencakup beberapa item, layanan, dan diskon. Item dan layanan itu sendiri memiliki diskonnya sendiri. Selanjutnya Diagram Kelas Layanan Pembayaran Penjualan pada Gambar 16.



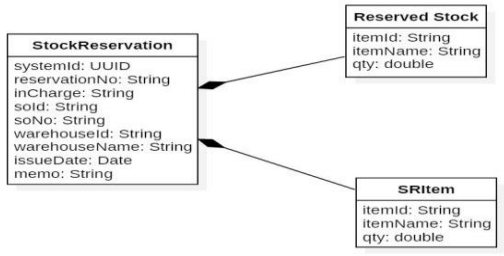
Gambar 16. Diagram Kelas Layanan Pembayaran Penjualan

Pada Gambar 16 menjelaskan class diagram untuk Sales Payment Service. Entitas utama dalam diagram ini adalah Kelas Pembayaran Penjualan yang mencakup beberapa pembayaran dan memiliki metode pembayaran. Selanjutnya Diagram Kelas Layanan Retur Penjualan pada Gambar 17.



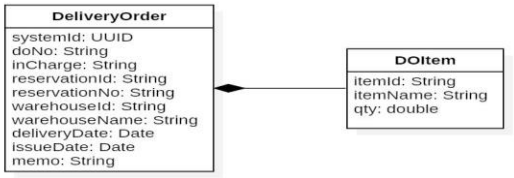
Gambar 17. Diagram Kelas Layanan Retur Penjualan

Pada Gambar 17 menjelaskan Gambar 3.13 adalah class diagram untuk Layanan Retur Penjualan. Entitas utama dalam diagram ini adalah Kelas Retur Penjualan yang mencakup beberapa item. Selanjutnya Diagram Kelas Reservasi Stok pada Gambar 18.



Gambar 18. Diagram Kelas Reservasi

Pada Gambar 18 menjelaskan class diagram untuk Stock Reservation Service. Entitas utama dalam diagram ini adalah Kelas Reservasi Stok yang mencakup stok dan barang yang dipesan. Selanjutnya Diagram Kelas Layanan Pesan Antar pada Gambar 19.



Gambar 19. Diagram Kelas Layanan

Pada Gambar 19 Menjelaskan class diagram untuk Layanan *Delivery Order*. Entitas utama dalam diagram ini adalah *Delivery Order Class* yang mencakup beberapa item.

4. Kesimpulan

Berdasarkan penelitian dan pengembangan yang dilakukan, diperoleh kesimpulan adalah arsitektur microservoces memiliki kompleksitas yang tinggi dibandingkan monolit karena ada beberapa tantangan tambahan yang perlu kita tangani saat menerapkannya antara lain: memilih ukuran yang tepat, transaksi non atomik, performace, masalah ketersediaan, dan beberapa masalah lain yang tidak ada di monolit.

Dengan menggunakan kerangka kerja khusus seperti Spring Cloud, pengembang dapat fokus pada logika bisnis daripada infrastruktur layanan seperti:

a) Pemisahan layanan;

b) Interaksi di antara mereka;

c) titik akhir API yang mereka miliki;

d) Desain entitas di setiap layanan;

e) Keseluruhan struktur layanan (tampilan penyebaran pada RUP);

f) Struktur di setiap layanan dari berbagai perspektif (proses, logis, dan implementasi).

Membuat aplikasi layanan mikro memerlukan sumber daya tambahan termasuk tenaga dan waktu karena kerumitannya. Namun, arsitektur layanan mikro mendukung pemisahan bagian-bagian sistem sehingga pengembang mendapat manfaat dari kepemilikan kode independen yang dapat membuat proses pengembangan menjadi lebih lancar. Unified Process adalah kerangka kerja proses yang telah terbukti menjadi salah satu standar industri saat membangun sistem yang kompleks dalam sebuah proyek. Dengan menggunakan framework proses seperti UP, pengembangan dapat lebih terstruktur, dan tingkat keberhasilan meningkat.

Daftar Rujukan

[1] Aljawarneh, N., & Alomari, Z. S. (2018). The Role of Enterprise Resource Planning Systems ERP in Improving Customer Relationship Management CRM: An Empirical Study of Safeway Company of Jordan. *International Journal of Business and Management*, *13*(8), 86. DOI: https://doi.org/10.5539/ijbm.v13n8p86 .

[2] Waseem, M., Liang, P., Shahin, M., Di Salle, A., & Márquez, G. (2021). Design, monitoring, and testing of microservices systems: The practitioners' perspective. *Journal of Systems and Software*, 182. DOI: https://doi.org/10.1016/j.jss.2021.111061 .

[3] Zhou, X., Peng, X., Xie, T., Sun, J., Ji, C., Li, W., & Ding, D. (2021). Fault Analysis and Debugging of Microservice Systems: Industrial Survey, Benchmark System, and Empirical Study. *IEEE Transactions on Software Engineering*, *47*(2), 243–260. DOI: https://doi.org/10.1109/TSE.2018.2887384 .

[4] Wang, Y., Kadiyala, H., & Rubin, J. (2021). Promises and challenges of microservices: an exploratory study. *Empirical Software Engineering*, *26*(4). DOI: https://doi.org/10.1007/s10664-020-09910-y .

[5] Colanzi, T., Amaral, A., Assunção, W., Zavadski, A., Tanno, D., Garcia, A., & Lucena, C. (2021). Are we speaking the industry language? The practice and literature of modernizing legacy systems with microservices. In ACM International Conference Proceeding Series (pp. 61–70). *Association for Computing Machinery*. DOI: https://doi.org/10.1145/3483899.3483904 .

[6] Di Francesco, P., Lago, P., & Malavolta, I. (2018). Migrating Towards Microservice Architectures: An Industrial Survey. *In Proceedings - 2018 IEEE 15th International Conference on Software Architecture, ICSA* 2018 (pp. 29–38). Institute of Electrical and Electronics Engineers Inc. DOI: https://doi.org/10.1109/ICSA.2018.00012 .

[7] Schermann, G., Cito, J., Leitner, P., Zdun, U., & Gall, H. C. (2018). We're doing it live: A multi-method empirical study on continuous experimentation. *Information and Software Technology*, *99*, 41–57. DOI: https://doi.org/10.1016/j.infsof.2018.02.010 .

[8] Carvalho, L., Garcia, A., Assunção, W. K. G., Bonifácio, R., Tizzei, L. P., & Colanzi, T. E. (2019). Extraction of configurable and reusable microservices from legacy systems: An exploratory study. *In ACM International Conference Proceeding Series* (Vol. A). Association for Computing Machinery. DOI: https://doi.org/10.1145/3336294.3336319 .

[9] Bilgin, B., Ünlu, H., & Demirörs, O. (2020). Analysis and Design of Microservices: Results from Turkey. In 2020 Turkish National Software Engineering Symposium, UYMS 2020 - *Proceedings. Institute of Electrical and Electronics Engineers Inc*. DOI: https://doi.org/10.1109/UYMS50627.2020.9247022 .

[10] Jonák, R., Smutný, Z., Šimůnek, M., & Doležel, M. (2020). Route and travel time optimization for delivery and utility services: An industrial viewpoint. *Acta Informatica Pragensia*, *9*(2), 200–209. DOI: https://doi.org/10.18267/J.AIP.133 .

[11] Gayialis, S. P., Kechagias, E. P., & Konstantakopoulos, G. D. (2022). A city logistics system for freight transportation: integrating information technology and operational research. *Operational Research*, *22*(5), 5953–5982. DOI: https://doi.org/10.1007/s12351-022-00695-0 .

[12]Álvarez López, Y., Franssen, J., Álvarez Narciandi, G., Pagnozzi, J., González-Pinto Arrillaga, I., & Las-Heras Andrés, F. (2018). *RFID technology for management and tracking: E-health applications. Sensors (Switzerland), 18*(8). DOI: https://doi.org/10.3390/s18082663 .

[13]Fernandez, P. (2016). "Through the looking glass: envisioning new library technologies" how artificial intelligence will impact libraries. *Library Hi Tech News*, *33*(5), 5–8. DOI: https://doi.org/10.1108/LHTN-05-2016-0024 .

[14]Stoyanov, T., Vaskevicius, N., Mueller, C. A., Fromm, T., Krug, R., Tincani, V., … Echelmeyer, W. (2016). No More Heavy Lifting: Robotic Solutions to the Container Unloading Problem. *IEEE Robotics and Automation Magazine*, *23*(4), 94–106. DOI: https://doi.org/10.1109/MRA.2016.2535098 .

[15]Miasnikova, O. V., & Tabolich, T. G. (2020). Development of Approaches to an Organizational and Functional Structure Creating of the Eurasian Economic Union Digital Transport Corridors Ecosystem. *Digital Transformation*, (1), 23–35. DOI: https://doi.org/10.38086/2522-9613-2020-1-23-35 .

[16]Scherbakov, V., & Silkina, G. (2019). Logistics of smart supply chains. *Atlantis Press*. DOI: https://doi.org/10.2991/icdtli-19.2019.15 .

[17]R, S. … D, S.-K. (2021). Formation Of The Integration Platform of Information Supply Of The "Wheeled Vehicles - Infrastructure" System. *The National Transport University Bulletin*, *1*(50), 221–232. DOI: https://doi.org/10.33744/2308-6645-2021-3-50-221-232 .

[18]Ilie Zudor, E., & Holmstrom, J. (2005). Solution framework proposal: Taking effective control over the project delivery chain with automatic identification and agent-based solutions. *Assembly Automation*, *25*(1), 59–65. DOI: https://doi.org/10.1108/01445150510579012 .

[19]Cantillo, V., Macea, L. F., & Jaller, M. (2019). Assessing Vulnerability of Transportation Networks for Disaster Response Operations. *Networks and Spatial Economics*, *19*(1), 243–273. DOI: https://doi.org/10.1007/s11067-017-9382-x .

[20]Hjollund, N. H. I., Larsen, L. P., Biering, K., Johnsen, S. P., Riiskjær, E., & Schougaard, L. M. (2014). Use of Patient-Reported Outcome (PRO) Measures at Group and Patient Levels: Experiences From the Generic Integrated PRO System, WestChronic. *Interactive Journal of Medical Research*, *3*(1), e5. DOI: https://doi.org/10.2196/ijmr.2885 .

[21]Figurski, J., & Niepsuj, J. (2021). Reliability of functioning of logistic processes. *Systemy Logistyczne Wojsk*, *54*(1), 125–134. DOI: https://doi.org/10.37055/slw/140410 .