PERANCANGAN ARSITEKTUR SISTEM TIKET ELEKTRONIK KERETA API MENGGUNAKAN KERANGKA *SERVICE ORIENTED ENTERPRISE ARCHITECTURE* (Studi Kasus : PT Railink)

Irpan Budiana, Irma Wibiyanti
Program Studi Teknik Informatika ST INTEN
Email : irpan.budiana@gmail.com, irma26wibiyanti@gmail.com

ABSTRAK

Penelitian bertujuan untuk memberikan *guideline* perancangan arsitektur *enterprise* sistem *e-ticketing* kereta api di Indonesia berorientasi *service* dengan menggunakan kerangka *Service Oriented Enterprise Architecture (SOE*A). Metodologi analis dan desain berorientasi objek (*OOAD*) digunakan dalam menggambarkan struktur perancangan sistem *e-ticketing* kereta api. Sedangkan kerangka perancangan arsitektur *enterprise e-ticketing* kereta api berorientasi *service* terdiri dari analisis arsitektur *as-is* yang menganalisis arsitektur yang sedang berjalan, dilanjutkan dengan perancangan arsitektur *to-be* untuk merancang arsitektur yang akan dikembangkan baik dari segi proses bisnis, aplikasi, dan infrastruktur. Perancangan arsitektur *enterprise e-ticketing* kereta api berbasis *service* juga menggunakan pola aristektur berbasis *microservice* dalam rangka memenuhi kebutuhan sistem yang *resilient, scalable, interoperable,* dan mampu diintegrasikan dengan berbagai standar *platform.*

Kata Kunci : Perancangan, arsitektur *enterprise*, *e-ticketing*, *service oriented enterprise architecture, service oriented architecture, microservice, object-oriented analysis and design*

1. PENDAHULUAN

PT Railink sebagai sebuah perusahaan swasta yang merupakan anak perusahaan dari PT Kereta Api Indonesia (Persero) dengan PT Angkasa Pura II (Persero) bertindak selaku operator dari Kereta Api (KA) Bandara. Dalam mengoperasikan KA Bandara dan menunjang bisnis usaha terutama dalam angkutan penumpang, PT Railink melakukan bisnis penjualan tiket yang diterapkan di stasiun-stasiun pemberangkatan maupun channel-channel pemesanan dan pembayaran tiket secara elektronik melalui *e-ticketing*. Dalam implementasi sistem *e-ticketing* tersebut dibutuhkan sebuah teknologi yang memungkinkan bentuk akhir dari sebuah program atau aplikasi komputer berupa sebuah *service* atau fungsi yang melakukan sebuah tugas atau proses spesifik yang dikenal dengan istilah *web service*. Dimana dengan adanya teknologi *web service*, memungkinkan perpaduan fungsi-fungsi dalam membangun sebuah program aplikasi

tanpa bergantung lagi pada sistem operasi maupun bahasa pemrograman yang digunakan.

Modul-modul *e-ticketing* yang dibangun dan diintegrasikan diantaranya modul pemesanan, pembayaran (antar bank dan *payment gateway*), *inventory* tempat duduk, *customer* relationship management, voucher, dan lain-lain. Dengan adanya kebutuhan integrasi modul, sistem *e-ticketing* yang dibangun memerlukan ketersediaan yang tinggi (*availability*), dapat memenuhi aspek resiliensi supaya dapat beradaptasi serta bertahan terhadap perubahan maupun kesalahan maupun kerusakan infrastruktur serta memiliki kemampuan menyesuaikan kebutuhan baru yang belum eksplisit tersedia sebelumnya (*scalability).*

Sehingga untuk pengembangan perangkat lunak sistem *e-ticketing* memerlukan sistem yang mendukung integrasi bisnis berbagai layanan (*service)* dari berbagai sumber informasi dan transaksi, interoperabilitas antar layanan, ketersediaan tinggi, memiliki aspek resiliensi dan skalabilitas, serta mampu mengakomodasi adanya perubahan untuk kemudian dilakukan adaptasi terhadap perubahan tersebut dengan cepat dan tepat, maka diperlukan sebuah *pattern* arsitektur pengembangan perangkat lunak yang mampu mengakomodasi kebutuhan dan faktor kualitas dari perangkat lunak tersebut dengan arsitektur sistem berbasis *service* (*service-oriented*). Arsitektur ini lebih dikenal dengan nama *microservice.*

2. LANDASAN TEORI

2.1 Arsitektur *Enterprise*

Arsitektur *enterprise* merupakan pendekatan logis, komprehensif, dan holistik untuk merancang dan meng-implementasikan sistem dan komponen sistem yang bersama (Parizeau, 2002). Arsitektur *Enterprise* merupakan *Blueprint* pemetaan hubungan antar komponen dan semua orang yang bekerja di dalam perusahaan secara konsisten untuk meningkatkan kerja sama/kolaborasi, serta koordinasi diantaranya (Ward, John and Peppard, Joe, 2002). Arsitektur *Enterprise* adalah deskripsi dari misi *stakeholder* yang di dalamnya termasuk informasi, fungsionalitas/kegunaan, lokasi organisasi dan parameter kinerja. EA mengambarkan rencana untuk mengembangkan sebuah sistem atau sekumpulan sistem (Osvalds, 2001).

2.2 *Service Oriented Architecture* (SOA)

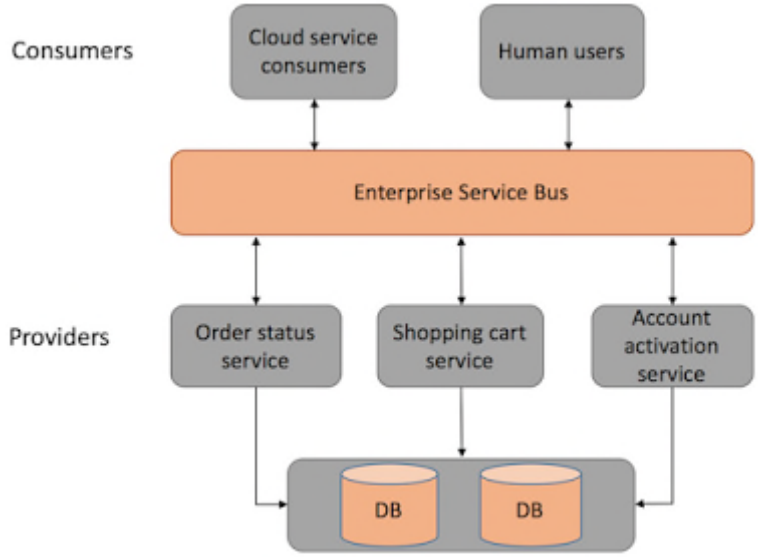
Service Oriented Architecture (SOA) adalah bagian utama dari *service computing platform* yang membawa konsep, teknologi, dan tantangan baru. da tiga hal penting yang menjadikan sebuah infrastruktur dapat disebut sebagai *service oriented architecture*, yaitu logika bisnis yang dienkapsulasi sebagai *service*, dan proses komunikasi antar *service* dengan menggunakan message. Dalam hal ini, *service* layer yang akan menjembatani hubungan antara business logic dan application logic (Erl, 2005). *Service Oriented Architecture* adalah sebuah kumpulan yang terdiri atas tools, teknologi, framework, dan *best practice* yang memudahkan implementasi sebuah *service* secara cepat. Proses dalam mengimplementasi SOA menggunakan metodologi yang mengidentifikasikan *service* yang dapat dipergunakan kembali (*reusable)* dalam aplikasi dan organisasi suatu perusahaan (Mittal, 2007)

2.3 Arsitektur *Microservice*

Microservice is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies. (Lewis and Fowler:2014)

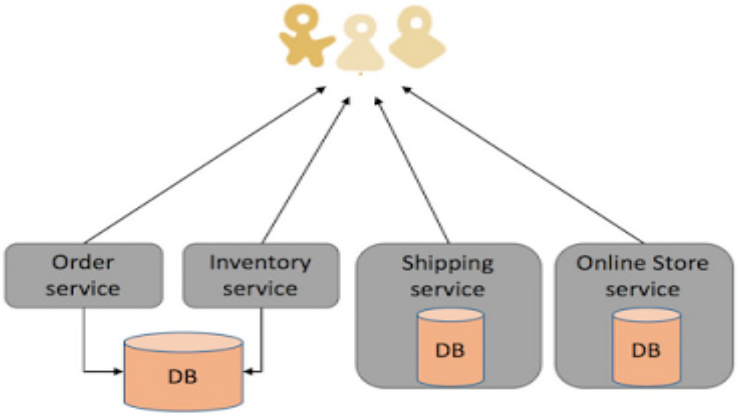
Arsitektur *microservice* memecah aplikasi yang kompleks ke dalam *service-service* otonom yang kecil. Proses-proses independen saling berkomunikasi melalui berbagai bahasa *Application Programming Interface* (API) yang juga independen. *Service-service* tersebut saling terpisah dan fokus pada tugas atau pekerjaan yang kecil, sehingga hal tersebut menjadikannya sebagai sebuah pendekatan modular dalam membangun sebuah sistem. *Service-service* tersebut mudah diganti, dirancang untuk memenuhi kebutuhan bisnis yang spesifik, dan dapat diimplementasikan menggunakan teknologi yang berbeda-beda. Selain itu, karena *microservice* bersifat otonom, *service* tersebut dibangun dan didistribusikan menggunakan metodologi pengembangan *software Continuous Delivery* (CD).

Microservice dan SOA



Gambar 1. Arsitektur SOA

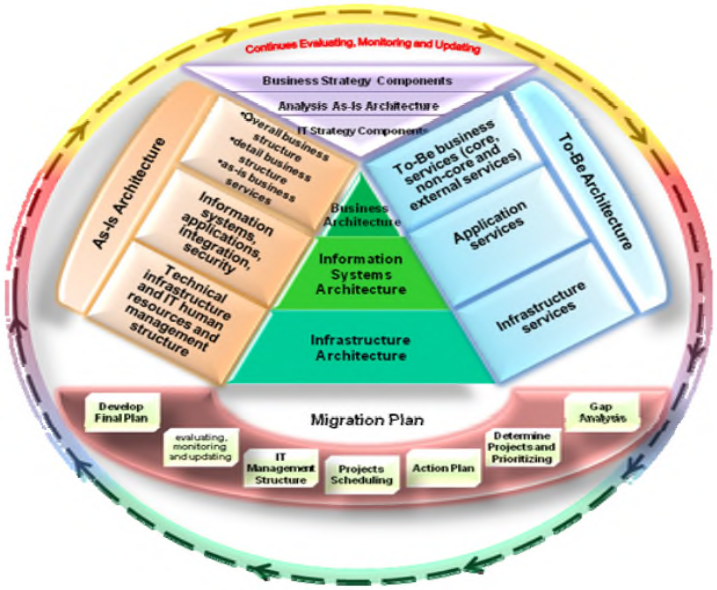
Di dalam *microservice, service-service* dapat beroperasi dan didisribusikan secara independen dari *service* lainnya tidak seperti SOA, sehingga memudahkan dalam pendistribusian *service* versi baru serta perubahan spesifikasi *service* secara mandiri. Di dalam SOA, ESB menjadi sebuah titik tunggak kegagalan yang akan berdampak pada keseluruhan aplikasi dikarenakan semua *service* berkomunikasi melalui ESB, jika salah satu *service* lambat, dapat mengakibatkan ESB harus ditutup untuk *request* dari *service* tersebut. Dari hal inilah, *microservice* jauh lebih baik dalam hal *fault tolerance*. Sebagai contoh, jika terdapat sebuah kobocoran memori dalam sebuah *microservice,* maka hanya *microservice* itu saja yang terkena dampak. *Microservice* lainnya akan tetap lanjut menangani request-*request*. Di dalam *SOA, service-service* berbagi penyimpanan data (Gambar 2) sementara dalam *microservice*, setiap *service* dapat mempunyai penyimpanan data tersendiri.



Gambar 2. Arsitektur *Microservice*

Baik SOA dan *microservice*, pengembangan harus mempertimbangkan kompleksitas arsitektur dan sistem terdistribusi. Pengembang harus mengimplementasikan mekanisme komunikasi antar-*service* di antara *microservice* atau di dalam ESB. Perbedaan utama antara SOA dan *microservice* terletak pada ukuran dan lingkupnya. Ukuran *microservice* secara signifikan lebih kecil dari pada kecenderungan ukuran SOA. Di sisi lain, sebuah SOA dapat berupa sebuah arsitektur monolitik atau dapat juga terdiri dari beberapa *microservice*.

2.4 Kerangka *Service Oriented Enterprise Architecture (SOEA)*



Gambar 3. Kerangka *Service Oriented Enterprise Architecture* (SOEA)

Kombinasi EA dan SOA menghasilkan kerangka untuk mendokumentasikan aspek bisnis dan IT dengan mengutilisasi pendekatan berbasis *service*.

Akar *framework* SOEA berasal dari Federal *Enterprise* Architecture *Framework* (FEAF) dan *Project Management Body of Knowledge* (PMBOK). *Layer* arsitektur yang terdapat dalam SOEA adalah:

1. Layer Arsitektur

   Dalam *framework* SOEA, terdapat tiga *layer* arsitektural:

   a. Arsitektur Bisnis: Arsitektur Bisnis merupakan bagian penting dan *layer* pertama dalam SOEA yang mempengaruhi layer arsitektural lainnya. Di dalam *layer* ini, dikombinasikan pendekatan *Business Process Management* (BPM) dengan *service orientation*.

   b. Aristektur Aplikasi : Di dalam arsitektur aplikasi didefinisikan suatu pendekatan komposit aplikasi yang mirip dengan *Component Based Software* (CBS) yang fokus pada pembangunan sistem *software* yang besar dengan mengintegrasikan komponen *software* yang sudah ada.

   c. Arsitektur Infrastruktur

      Setelah menentukan aplikasi komposit, di dalam *layer* infrastruktur, diidentifikasi lingkungan di mana *service* aplikasi mendukung semua *service* bisnis.

2. Fase Proyek

   Tabel 1 menggambarkan fase, langkah, aktifitas, dan output yang diharapkan dari sebuah proyek SOEA.

Tabel 1. Fase, Langkah, dan output aktivitas dari *framework* SOEA

Phase	Step	Activity	Deliverable
Phase 0: Project Planning		▪ Organizing the project team ▪ Finalizing the scope of the project ▪ Developing project management plan	▪ Project organizational structure ▪ Scope statement ▪ Project Management Plan (PMP)
Phase 1: As-Is Architecture	Business As-Is Architecture	▪ Identifying the overall business structure ▪ Extracting organization's business strategy components ▪ Identifying detail business structure ▪ Extracting as-is business services	▪ Business conceptual model ▪ IT and business strategy components ▪ Business function model, process model and data model ▪ As-Is business services
	IT As-Is Architecture	▪ Identifying technical IT resources ▪ Identifying non-technical IT resources	▪ Information systems, applications, integration, security and infrastructure ▪ IT human resources and management structure
Phase 2: Analysis of As-Is Architecture			▪ analysis of Business and IT As-Is Architecture
Phase 3: To-Be Architecture	Business To-Be Architecture IT To-Be Architecture		▪ To-Be business services (core, non-core and external services) ▪ Application services ▪ Infrastructure services
Phase 4: Migration Plan		▪ Gap analysis ▪ Identifying projects and prioritizing ▪ Developing evaluation and updating plan ▪ Proposing IT management structure	▪ Gap analysis report ▪ Proposed projects ▪ Evaluation and updating plan ▪ IT management structure

2.5 *Unified Modelling Language (UML)*

Unified modelling language (UML) merupakan seperangkat model kontruksi dan notasi yang dibentuk dalam pengembangan sistem berorientasi pada objek (Satzinger, 2012). Diagram-diagram bagian dari UML yang dapat dijelaskan sebagai berikut:

2.5.1 *Diagram Activity*

Diagram *Activity* merupakan diagram yang menunjukkan alur kerja atau aktivitas *user* secara berurutan.

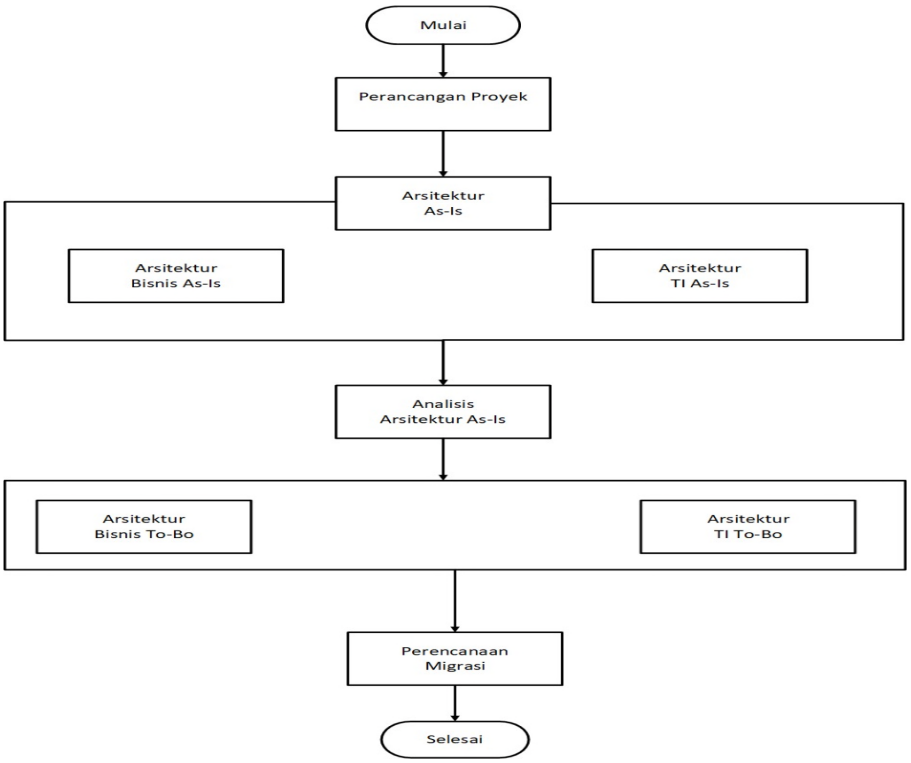
2.5.2 *Diagram Use Case*

Diagram *Use Case* adalah diagram yang menunjukkan aktivitas yang dilakukan oleh sistem berupa respon terhadap permintaan pengguna serta hubungan antara aktor–aktor pengguna tersebut di dalam sistem.

2.5 *E-ticketing*

Electronic Ticketing (*e-ticketing*) adalah sebuah dokumen elektronik yang banyak digunakan sebagai tiket penumpang moda transportasi. Di sisi lain menyebutkan *e-ticketing* merupakan suatu cara untuk mendokumentasikan proses penjualan dari aktifitas perjalanan pelanggan tanpa harus mengeluarkan dokumen secara fisik (Grace, 2009). Sehingga dapat diartikan bahwa *e-ticketing* merupakan sebuah teknologi yang berguna untuk menggantikan pengolahan dan penggunaan tiket tradisional (*paper ticket*) (Iwuagwu, 2006). Melihat dari sisi perkeretaapian, sistem *e-ticketing* berpeluang untuk meminimalkan biaya dan mengoptimalkan kenyamanan penumpang. *E-ticketing* mengurangi biaya proses tiket, menghilangkan formulir kertas dan meningkatkan fleksibilitas penumpang dan agen perjalanan. Tujuannya adalah untuk membuat pembelian atau pemesanan tiket lebih mudah (Karami, 2006).

3. METODOLOGI



Gambar 4. Metodologi Penelitian

Gambar 4 menunjukkan alur metodologi penelitian ini. Pertama-tama dilakukan analisis terhadap arsitektur bisnis dan TI *as-is* untuk mendapatkan gap analysis yang selanjutnya akan dibuatkan perancangan arsitektur bisnis dan TI *to-be*.

3.1 Analisis Arsitektur Bisnis *As-is*

Analisis fokus pada pengumpulan informasi pada posisi saat ini dari bidang-bidang kegiatan penjualan tiket dan membuat gambaran yang jelas tentang gap capability yang bisa menjadi hambatan dalam mencapai potensi pendapatan secara keseluruhan.

Analisis arsitektur bisnis tiket memetakan gambaran awal tujuan, sasaran, dan capaian bisnis tiket PT Railink secara arsitektural. Untuk mengetahui arsitektur bisnis *as-is* tiket PT Railink, maka perlu mengetahui perkembangan penggunaan dari *e-ticketing as-is* di unit komersial di PT Railink. Arsitektur bisnis *as-is* diketahui berdasarkan dokumen *blueprint* yang dimiliki oleh PT Railink.

3.2 Analisis Arsitektur TI *As–Is*

Analisis arsitektur TI yang mencakup arsitektur TI dari sistem penjualan tiket saat ini di PT Railink, melakukan *review* terhadap infrastruktur teknologi informasi yang digunakan dan melakukan pengamatan peran sumber daya dan sistem informasi di dalam proses bisnis penjualan tiket.

3.3 Perancangan Arsitektur Bisnis *To-be*

Perancangan arsitektur bisnis *to-be* meliputi tiga *service* bisnis berikut:

1. *Service* bisnis *core*

   Service bisnis *core* terkait dengan proses bisnis utama sistem tiket PT Railink dimulai dari transaksi *reservasi (booking),* pembayaran, *gate-in* dan *gate-out* serta pembatalan tiket.

2. *Service* bisnis *non-core*

   Service bisnis *non-core* terkait dengan proses bisnis pendukung. *Service* bisnis non-core meliputi *service* tiket manual dan pengiriman *e-mail*.

3. *Service* eksternal

   Service eksternal yang diharapkan terdiri dari *service dashboard* laporan eksekutif dan rekonsiliasi pendapatan tiket *payment gateway*.

3.4 Perancangan Arsitektur TI *To-be*

Perancangan arsitektur TaI *to-be* akan meliputi *service-service* sebagai berikut:

1. *Service* Aplikasi

   Service aplikasi dibangun berdasarkan *service* bisnisnya. Suatu *service* bisnis dapat memiliki satu atau lebih *service* aplikasi disertai dengan penamaan *service* dan alamat URL *Service* dapat diakses sesuai kesepakatan standar protokol komunikasi yang menjembatani aplikasi-aplikasi *front-end* dengan *service*nya.

2. *Service* Infrastruktur

   Service infrastruktur dirancang untuk mendukung *service* aplikasi baik digunakan secara fisik maupun menggunakan virtualisasi melalui sekumpulan sumber daya infrastruktur seperti *server web, server aplikasi, server database, server* lain, dan media penyimpanan (*storage*).

4. HASIL DAN PEMBAHASAN

4.1 Analisis Arsitektur *As-is*

Analisis kondisi arsitektur saat ini (*as-is*) dilakukan berdasarkan kondisi bisnis serta kondisi TI perusahaan. Analisis tersebut ditujukan terhadap informasi posisi saat ini dan dalam rangka membuat gambaran proses bisnis yang sedang berlangsung disertai *gap capability* yang menjadi hambatan di dalam perancangan sistem *e-ticketing* kereta api sebagai upaya perusahaan dalam hal ini PT Railink mendorong pendapatan dari bisnis transportasi angkutan penumpang kereta api.

4.1.1 Analisis Arsitektur Bisnis *As-is*

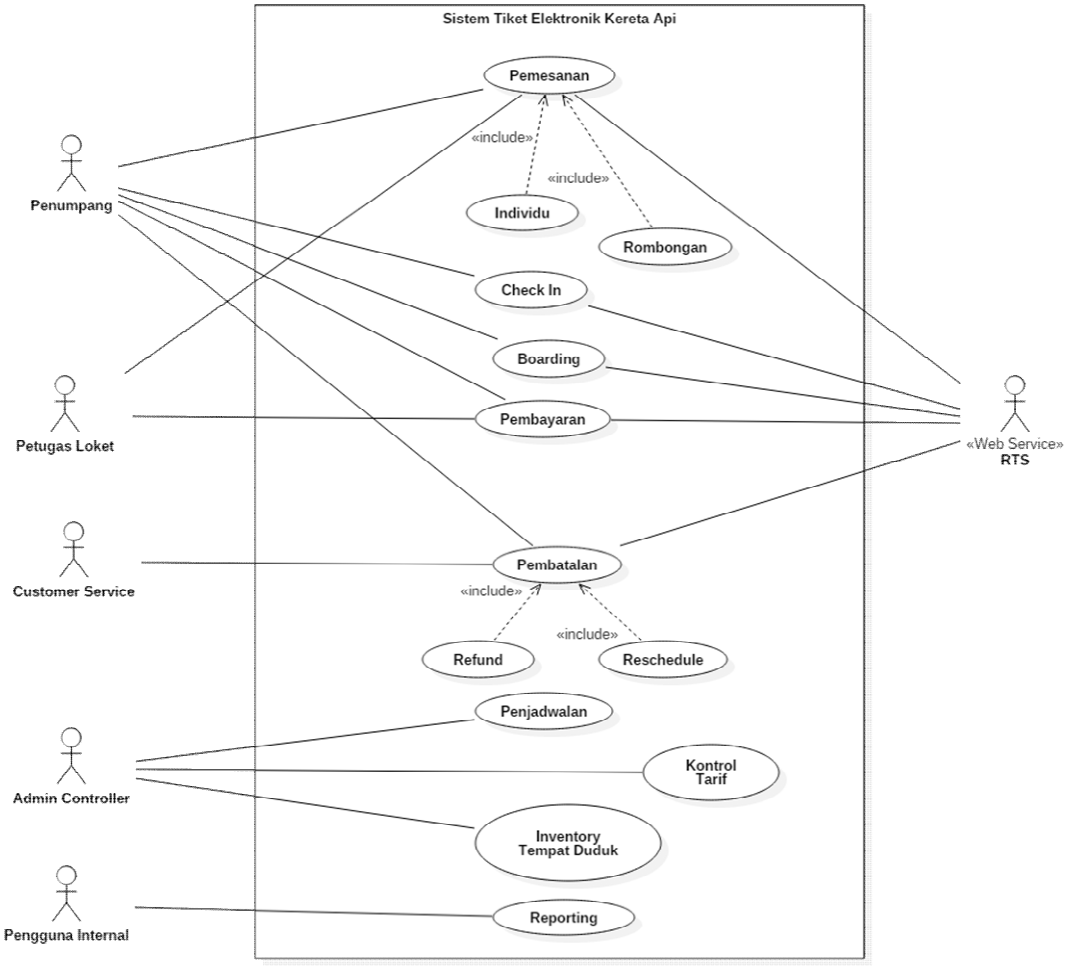
Dalam analisis arsitektur bisnis *as-is*, didapat dari hasil analisis struktur bisnis perusahaan dan relasinya dengan *stakeholders*. Analisis bisnis *as-is* memetakan gambaran awal tujuan, sasaran, dan capaian bisnis.

Gambar 5. Konteks Bisnis Perusahaan

Bisnis utama PT Railink terdiri dari usaha bisnis angkutan penumpang, komersialisasi aset, dan layanan pendukung bisnis lainnya. Faktor kunci keberhasilan untuk mencapai keunggulan operasional yaitu dengan upaya-upaya: penerapan *safety* dengan teknologi yang tepat, peningkatan kualitan SDM, dan peningkatan akses pengguna. Siklus dalam mencapai keunggulan operasional tersebut tidak terlepas dari kegiatan-kegiatan: pengetahuan tentang pasar, adopsi teknologi yang tepat, dan pemanfaatan lahan secara efektif. Sistem *e-ticketing* kereta api merupakan pilihan utama yang menunjang pendapatan perusahaan dalam bisnis angkutan penumpang.

4.1.1.1 *Service Bisnis As-is*

Diagram *Use Case* dari *Service* Bisnis *As-is* dapat digambarkan pada gambar 7 sebagai berikut:



Gambar 6. Diagram use Case Proses Bisnis *As-is*

Proses bisnis *as-is* secara global untuk penjualan tiket seperti pada Gambar memiliki 3 (tiga) kelompok proses bisnis utama sebagai berikut:

a. Pemesanan dan Penjualan Tiket, serta Pembayaran untuk penumpang individu dan rombongan;
b. Pembatalan dan Pengembalian Bea (*Refund*)
c. *Check In* dan *boarding* untuk penumpang individu maupun rombongan;

Proses bisnis *as-is* untuk pengguna internal dan administrator memiliki 4 (empat) proses bisnis sebagai berikut:

a. Proses penjadwalan (*scheduling*)
b. Proses pentarifan
c. Pengendalian *inventory* tempat duduk
d. Pembuatan Laporan-laporan

4.1.1.2 *Gap Analysis*

Dari hasil identifikasi terhadap proses bisnis *as-is*, maka didapat beberapa gap kebutuhan bisnis pada sistem yang akan diimplementasikan sebagaimana dijelaskan pada tabel 3 berikut:

Tabel 3. Identifikasi *Gap* Kebutuhan Proses Bisnis

No.	Masalah	Kondisi *As-is*	Kebutuhan TI	*Gap*
1	Pemesanan dan pembayaran dilayani di Loket	Pemesanan dan Pembayaran yang dilakukan di stasiun yang masih dilayani di loket	Pelayanan pemesanan dan pembayaran yang cepat cukup dilakukan melalui *Vending Machine*.	Waktu pelayanan yang tidak cepat menimbulkan antrian calon penumpang pada jam-jam sibuk. Peniadaan petugas loket
2	Pembayaran tunai pada loket di stasiun	Pembayaran menggunakan uang tunai	Pembayaran dilakukan secara non-tunai untuk mempercepat proses transaksi pembayaran dan minimal kesalahan perhitungan manual uang tunai.	Mesin EDC tunggal untuk akomodasi transaksi non-tunai berbagai bank nasional.
3	Proses Check-in dan Boarding	Penumpang yang sudah memiliki tiket harus check-in dan boarding	Proses gate-in dijadikan satu proses baik check-in dan boarding untuk meringkas proses antrian penumpang masuk peron dan penumpang tidak menumpuk di stasiun.	Proses *gate-in* dan *gate-out*

4.1.2 Analisis Arsitektur *TI As-is*

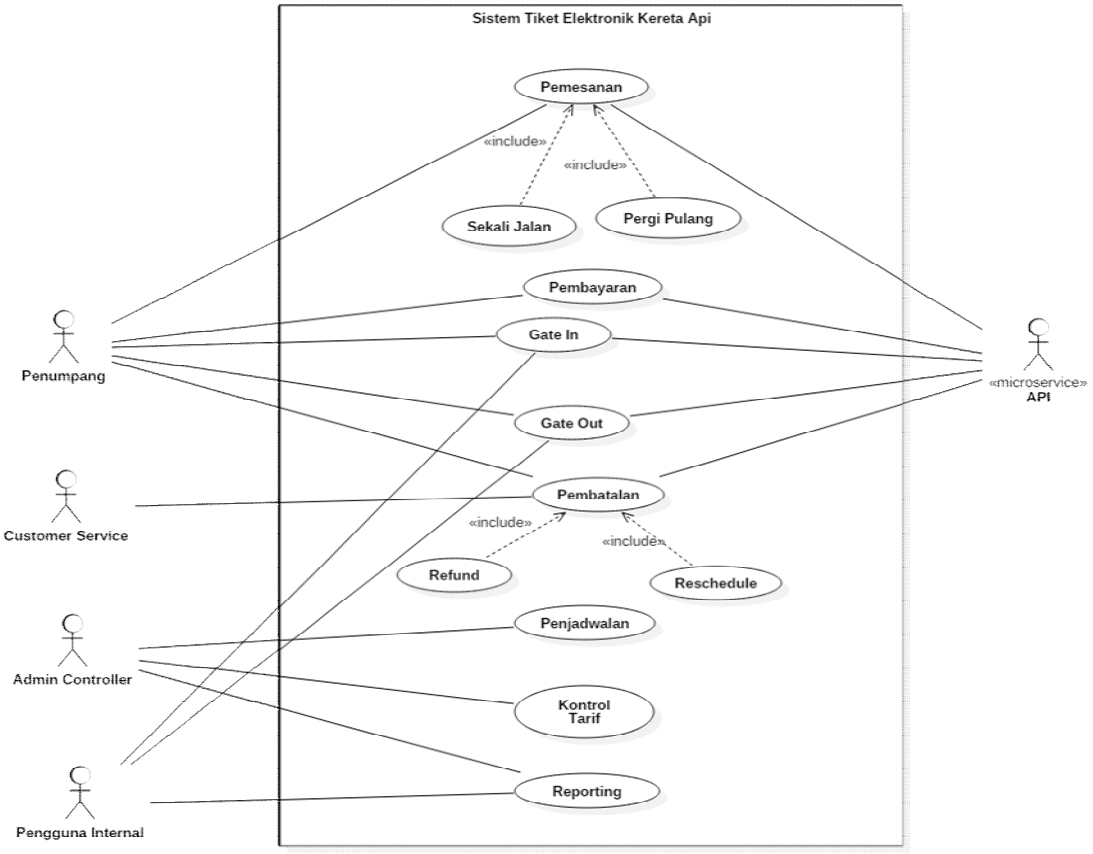
Analisis arsitektur TI *As-is* terdiri dari Analisis Arsitektur TI dengan identifkasi sumber daya TI teknis dan non-teknis. Analisis arstiektur TI *As-is* mencakup analisis arsitektur dari sisi sistem informasi dan infrastruktur.

a. Perancangan Arsitektur *To-be*

Perancangan Arsitektur *to-be* dari sistem tiket elektronik kereta api pada PT Railink terdiri dari perancangan Arsitektur bisnis *to-be* dan perancangan Arsitektur TI *to-be*.

b. *Perancangan* Arsitektur *Bisnis To-be*

Perancangan Arsitektur Bisnis *to-be* dapat dijelaskan pada diagram *use case* bisnis sebagai berikut:



Gambar 7. Diagram *Use Case* Bisnis *To-be*

4.2.1.2 Diagram *Activity*

Dari fungsi-fungsi bisnis yang terdapat dalam Diagram *Use Case To-be* Sistem *E-ticketing* Kereta Api, maka terdapat aliran aktivitas pada Diagram *Activity* sebagai berikut:

1. Diagram *Activity* Proses Pemesanan
2. Diagram *Activity* Proses Pembayaran
3. Diagram *Activity* Proses *Gate-in*
4. Diagram *Activity* Proses *Gate-out*
5. Diagram *Activity* Proses Pembatalan
6. Diagram *Activity* Proses Penjadwalan

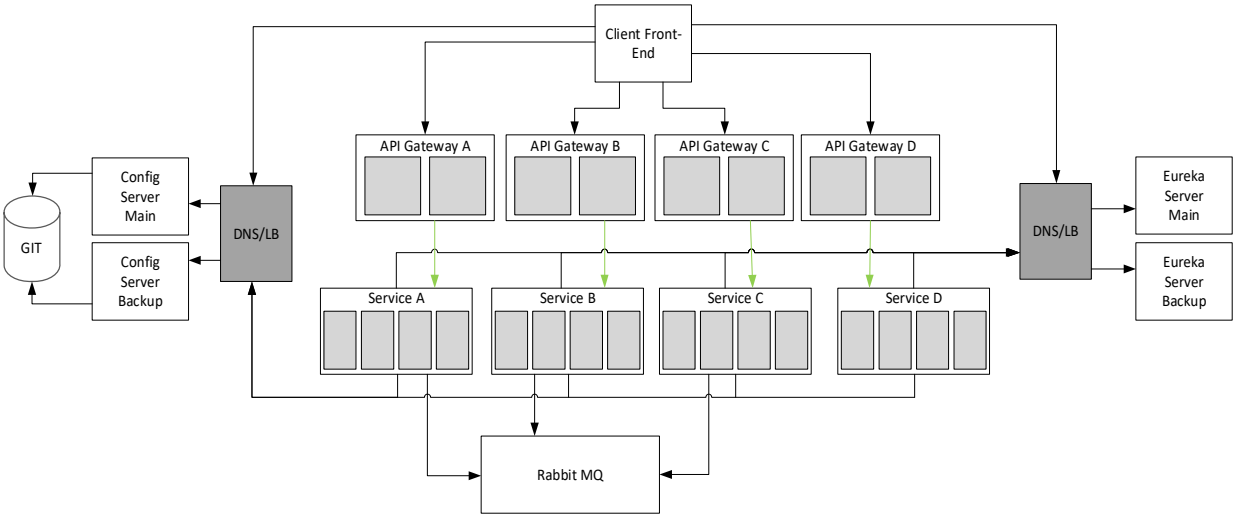
7. Diagram *Activity Setting* Tarif
8. Diagram *Activity Web* Reporting

4.2.1.3 *Service* Bisnis *To-be*

Berdasarkan definisi *use case*, kebutuhan *service-service* dikekompokkan ke dalam kategori *service core*, *service non-core*, dan *service* eksternal. *Service* core merupakan *service* utama atau penting dari sebuah *service* bisnis, dimana bisnis mengembangkan atau mengoperasikan aktivitas utamanya. *Service* non-core merupakan kumpulan dari *service* penunjang *service core*. Dan kelompok *service* eksternal kaitanya dengan kebutuhan tambahan dan hubungan *service* dengan lingkungan eksternal. *Service core* yang dikembangkan pada sistem *e-ticketing* kereta api *to-be* yaitu: *Service Schedule, Service Transaction, Service Payment, dan Service Gate*. *Service* non-core terdiri dari *Service* Info dan *Service* E-mail. Sedangkan *service eksternal* yaitu *service payment-gateway*.

4.2.2 Perancangan Arsitektur TI *To-be*

Sistem *E-ticketing* kereta api dikembangkan menggunakan *Java Spring Boot*, dengan arsitektur dapat dilihat pada Gambar 9 yang menggambarkan implementasi Arsitektur *Microservice*. Gambar 9 juga menunjukkan letak dari *service-service* yang berfungsi untuk menangani fungsi bisnis aplikasi, yaitu pada *microservice* node yang terbungkus oleh Zuul proxy sebagai gateway sistem *load balancer*, dengan Eureka server dan Config server untuk memungkinkan manajemen sistem yang terpusat dan dinamis.



Gambar 8. Arsitektur *Microservice* Sistem *E-ticketing* Kereta Api

Keterangan :

1. Spring Cloud Config

   Spring Cloud Config menyediakan dukungan server dan client-side untuk konfigurasi externalized dalam sistem terdistribusi. Dengan "Config Server" sistem ini memiliki tempat sentral untuk mengelola properti eksternal untuk aplikasi di semua lingkungan. Implementasi standar dari backend server penyimpanan menggunakan "git".

2. Server Eureka.

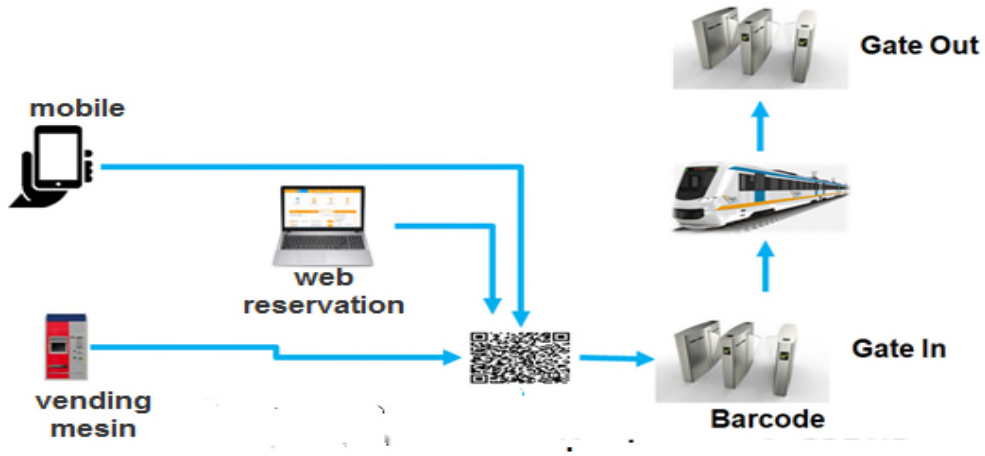
   Server Eureka untuk *Service* registry: Sebuah registri *service* menyediakan lingkungan runtime untuk *service* untuk secara otomatis mempublikasikan ketersediaannya pada saat runtime. Sebuah registri akan menjadi sumber informasi yang baik untuk memahami topologi *service* di setiap titik.

3. Zuul Proxy

   Zuul Proxy secara internal menggunakan server Eureka untuk *service* discovery, dan *Ribbon* untuk load balancing antara *service*. Proxy Zuul juga mampu melakukan routing, monitoring, mengelola ketahanan, keamanan, dan sebagainya. Secara sederhana, dapat dipertimbangkan untuk menggunakan Zuul *service* reverse proxy. Dengan Zuul, bahkan dapat diubah perilaku dasar dari *service* tersebut pada lapisan API.

4.2.2.3 Perancangan Arsitektur *Infrastruktur To-be*

Blok diagram aliran secara global operasional darii Sistem *E-ticketing* Kereta Api *to-be* dapat diseperti pada gambar 9.



Gambar 9. Diagram aliran global operasional sistem *e-ticketing* kereta api *to-be*

Beberapa tahapan dalam mendapatkan tiket kereta api ada beberapa cara yaitu:

1. Pemesanan melalui aplikasi mobile

   Customer dapat melakukan pemesanan melalui handphone dengan aplikasi yang berbasis android atau IOS dan langsung mendapatkan nomor tiket sesuai dengan tanggal pemesanan setelah melakukan pembayaran sehingga bisa memudahkan *customer* tanpa harus datang langsung ke stasiun. Pada saat *customer* sudah ke stasuin *customer* cukup menempelkan *barcode* pada mesin gate yang ada di stasiun pada saat pemberangkatan kereta maupun keluar dari stasiun kereta api
2. Pemesanan atau pembelian tiket langsung kereta api dilakukan dengan cara mendatangi langsung ke stasiun dan dilakukan melalui vending machine
3. Pemesanan melalui *web* reservasi, *customer* bisa langsung memesan melalui alamat *web* reservasi kereta api dan langsung membayar melalui e-commerce maupun transfer sehingga ketika datang ke stasiun cukup menunjukan barcode tiket yang sudah dibeli pada mesin gate kereta api.

5. KESIMPULAN

Perancangan Arsitektur bisnis sistem *e-ticketing* dimulai dengan melakukan analisis terhadap proses bisnis *as-is* dengan mendefiniskan fungsi-fungsi bisnis pada sistem *e-ticketing* yang sedang berjalan. Definisi fungsi-fungsi bisnis tersebut dianalisis

gap yang menjadi kendala dalam pengembangan untuk menentukan perubahan fungsi bisnis dalam perancangan arsitektur bisnis *to-be*. Hasil *gap analysis* dari perancangan arsitektur bisnis tersebut adalah:

a. Sistem pembayaran tiket elektronik secara non-tunai.
b. Pelayanan penjualan tiket elektronik yang tidak menggunakan loket sebagai *channel* penjualannya.
c. Perbaikan proses pada proses validasi penumpang kereta api yang akan naik kereta api dengan melakukan proses *gate-in* dan *gate-out*.

Perancangan Arsitektur teknologi informasi sistem *e-ticketing* kereta api *to-be* berbasis *service* menerapkan perancangan Arsitektur aplikasi menggunakan kerangka *Service Oriented Enterprise Architecture* (SOEA) berbasis *microservice* dimana fungsi bisnis dari perancangan aristektur bisnis, masing-masing dijadikan sebagai *service-service* kecil yang independen, resilien, dan dapat berintegrasi dengan platform yang berbeda dengan standar protokol API. Arsitektur sistem *e-ticketing* kereta api didukung dengan topologi infrastruktur jaringan yang memiliki kapabilitas tinggi untuk memenuhi kebutuhan sistem *e-ticketing* yang dapat beroperasi selama 24 jam non-stop dan *downtime server* yang minimal.

6. DAFTAR PUSTAKA

Assauri, Sofyan., Manajemen Pemasaran Dasar, Konsep, dan Strategi. PT. Jakarta: Grafindopersada.2011

David, Fred. R. and Forest R., Strategic Management Concept and Cases 14th Edition. Pearson Education. New Jersey. USA. 2012.

Engels, G., Assman, M., *Service*-Oriented *Enterprise* Architectures: Evolution of Concepts and Methods. Department of Computer Science University of Paderborn. München, Germany. 2008.

Erl, Thomas, *Service*-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall PTR. New Jersey. USA. 2005.

Haki, M. Kazem, Forte, M. Wentland. *Service* Oriented *Enterprise* Architecture Framework, University of Lausanne, Switzerland. 2010

Heydt-Benjamin, T.S., Chae, H., Defend, B., Fu, K,. Privacy for Public Transportation, University of Massachusetts, Amherst, MA. USA. 2006

Karami, Mitra. Factor Influencing Adopting Online Ticketing. Lu lea University of Technology. Swedia. 2006