STUDI KASUS IMPLEMENTASI SERVICE ORIENTED ARCHITECTURE (SOA) DI CREDIT SUISSE

Sartika Kurniali

Information Systems Department, School of Information Systems, Binus University
Jl. K.H. Syahdan No. 9, Palmerah, Jakarta Barat 11480
SartikaKurniali@binus.edu

ABSTRACT

Credit Suisse Group (CSG) is a leading company engaged in global financial services. At the end of 90s IT infrastructure complexity of Credit Suisse has reached a critical situation. The existing IT infrastructure is no longer able to support the required business functionality. This leads to the introduction of an integrated architecture based on Service-Oriented Architecture (SOA). The purpose of this study is to learn how Credit Suisse successfully implements SOA both on a technical and organizational level. Analyses were performed by processing factual and reference data acquired.
From the research, the successful implementation is due to clarity of interface, clarity of process, management commitment, and solid technology. Their success does not come easily since they also face obstacles and conflicts on the implementation.

Keywords: *service-oriented architecture (SOA), service, reuse*

ABSTRAK

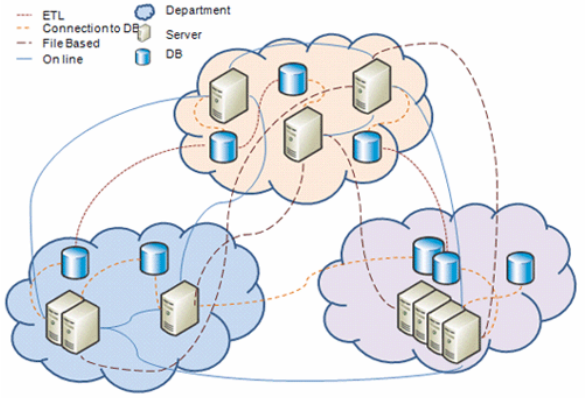
Credit Suisse Group (CSG) adalah perusahaan terkemuka yang bergerak pada jasa finansial secara global. Pada akhir tahun 90-an kompleksitas infrastruktur IT Credit Suisse telah mencapai situasi kritis. Infrastruktur IT yang sudah ada tidak lagi dapat mendukung fungsionalitas bisnis yang diperlukan. Hal ini membuat dikenalkannya sebuah arsitektur yang terintegrasi berdasarkan Service-Oriented Architecture (SOA). Tujuan penelitian ini adalah mempelajari bagaimana Credit Suisse sukses melakukan implementasi SOA baik pada level organisasi maupun teknis. Analisis dilakukan dengan mengolah data faktual yang didapatkan dan referensi. Dari hasil penelitian, kesuksesan implementasi karena kejelasan interface, kejelasan proses, komitmen manajemen, dan teknologi yang solid. Kesuksesan juga tidak didapat dengan mudah karena implementasi juga menghadapi kendala dan konflik.

Kata kunci: *service-oriented architecture (SOA), service, penggunaan kembali*

PENDAHULUAN

Credit Suisse Group (CSG) adalah perusahaan terkemuka yang bergerak pada jasa finansial secara global yang berpusat di Zurich. Unit bisnis pelayanan finansial Credit Suisse menangani klien pribadi dan perusahaan kecil dan menengah dengan pelayanan bank pribadi dan konsultasi finansial, produk bank, dan pensiun dan asuransi dari Winterthur. Bisnis unit Credit Suisse First Boston, sebuah bank investasi melayani institusi global, perusahaan, dan klien individu dan berperan juga sebagai perantara finansial. Saham Credit Suisse Group didaftarkan di Switzerland dan New York. Grup ini memperkerjakan sekitar 60.000 karyawan dan beroperasi di lebih dari 50 negara. Pada 31 Maret 2004, asetnya telah mencapai CHF 1.241,3 milliar.

Namun Credit Suisse mengalami masalah klasik karena *middleware overload* yang disebabkan terlalu banyaknya produk dan koneksi *point-to-point*. Skenario yang akhirnya menciptakan situasi "*integration spaghetti*" (Istilah yang dipakai Gartner) (Herr, Bath, & Koschel, 2004), membuat mereka melakukan perubahan di satu sistem yang bisa memiliki efek berkelanjutan, yang hasilnya tidak bisa diperkirakan karena setiap departemen memiliki sistem sendiri seperti pada Gambar 1.



Gambar 1 Tipikal *integration spaghetti* di Perusahaan

Credit Suisse tidak puas dengan model integrasi *point-to-point* yang hanya menyelesaikan masalah per proyek. Hal ini telah membuat pengembangan aplikasi baru sangat kompleks dan terkadang tidak mungkin dilakukan.

Pada akhir tahun 90-an, kompleksitas infrastruktur IT telah mencapai situasi kritis. Bahkan tidak lagi dapat mendukung fungsionalitas bisnis Credit Suisse. Solusi ad hoc untuk mengintegrasikan sistem IT setelah merger dan akuisisi pun tidak sukses.

Pertumbuhan fungsionalitas bisnis Credit Suisse yang tidak bisa lagi didukung oleh infrastruktur IT telah mendorong dikenalkannya sebuah arsitektur yang terintegrasi berdasarkan *Service-Oriented Architecture (SOA)* ke dalam perusahaan. Pindah dari sistem mainframe ke arsitektur berorientasi layanan (SOA) yang menggunakan layanan web menarik dilakukan, tetapi menakutkan (Rodriguez et al., 2013). Implementasi SOA pun diterapkan pada level organisasi dan teknis. Dengan pendekatan implementasi proyek pilot, Credit Suisse telah berhasil mengimplementasikan tiga *service bus* yang berbeda untuk bisa memenuhi kebutuhan berbeda dari komunikasi *synchronous*, komunikasi *asynchronous*, dan *bulk data transfer*.

Keuntungan implementasi antara lain penggunaan kembali *service*, pengembangan aplikasi yang lebih efisien, dan peningkatan kolaborasi. Hal ini termasuk pengambilan keputusan yang lebih

sistematis mengenai penggunaan kembali dan target yang lebih spesifik untuk pengembang *service*. Tujuan penelitian ini untuk melihat apa saja yang harus perlu dilakukan untuk memperkenalkan SOA di perusahaan.

METODE

Penelitian ini menggunakan metode penelitian kualitatif berupa studi kasus. Bahan penelitian didapatkan lewat studi literatur dari berbagai sumber.

HASIL DAN PEMBAHASAN

Solusi Teknologi Informasi

Tantangan dalam industri perbankan memaksa bank untuk merenovasi sistem inti mereka untuk bertahan hidup persaingan sengit (Rong et al., 2013). CIO Credit Suisse membuat keputusan untuk mengatasi masalah di atas dengan mengenalkan sebuah arsitektur yang terintegrasi berdasarkan *service-oriented architecture (SOA)*. Pendekatan yang dilakukan ternyata berhasil. Mereka mendefinisikan dan mengimplementasikan bus integrasi menurut kebutuhan yang paling mendesak dan langsung mendapatkan manfaat darinya.

Setelah sukses mengimplementasi bus informasi yang menyediakan komunikasi *synchronous*, Credit Suisse menambahkan *bus event* yang digunakan untuk integrasi *backend to backend* yang menyediakan komunikasi *asynchronous*. Bus kedua ini menggunakan prinsip yang mirip, tetapi menggunakan kriteria teknis yang sedikit berbeda dan oleh karenanya menyediakan keuntungan tambahan. Bahkan tipe ketiga bus integrasi beroperasi menggunakan transfer *file* untuk komunikasi.

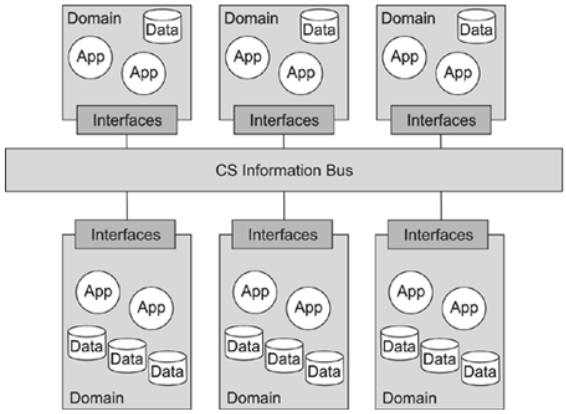
Arsitektur

Arsitektur yang diimplementasikan menggabungkan tiga paradigma. CSIB yang digunakan untuk komunikasi *synchronous* yang digunakan untuk akses *front-end* ke aplikasi di dalam, *event bus infrastructure* (EBI) yang menggunakan komunikasi *asynchronous* untuk mengintegrasikan aplikasi baru yang tidak berada di dalam perusahaan, dan terakhir *bulk integration infrastructure* yang menggunakan transfer file untuk komunikasi.

Integrasi Synchronous dengan CSIB

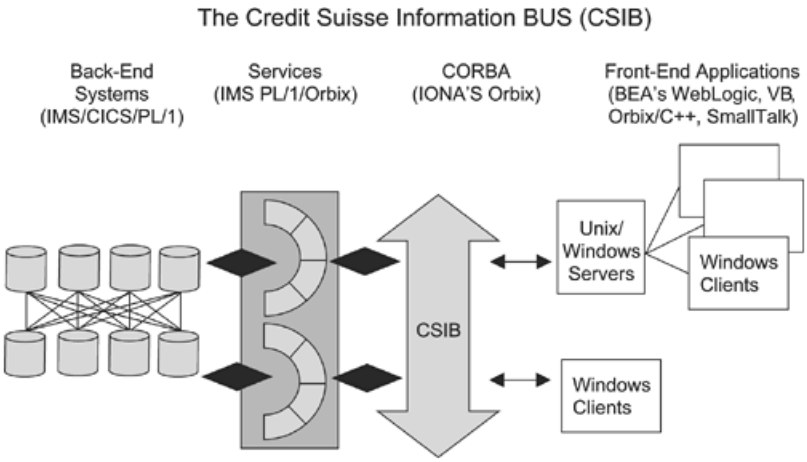
Implementasi bus informasi dimulai dengan membuat struktur dari aplikasi yang ada menjadi sekitar 20 domain. Sebuah domain aplikasi menggabungkan semua data dan aplikasi yang dimiliki oleh suatu area bisnis. Gambar 2 menunjukkan bagaimana aplikasi dienkapsulasi ke dalam domain yang memiliki coupling ketat di dalam domain, tetapi longgar ketika berpindah domain karena menggunakan bus informasi.

Namun, CSG memutuskan tidak melakukan usaha untuk melakukan faktor ulang logika dalam domain atau antara domain, ketika komunikasi tidak melewati bus informasi. CSG memutuskan perubahan mendatang dari aplikasi yang ada akan merubah komunikasi dalam domain yang akan membuat hilangnya *coupling* infrastruktur tanpa memberikan dampak pada service yang sudah ada.



Gambar 2 Pembagian Aplikasi ke dalam Domain yang *loosely coupled*

Pertama kali CSIB diimplemntasikan menggunakan teknologi CORBA. Gambar 3 menyediakan gambaran detail bagaimana arsitektur awal dan teknologi pada layer lainnya.



Gambar 3 Implementasi awal Credit Suisse information bus

Selain CORBA, DCE (Distributed Computing Environment) dan DCOM juga dievaluasi, tetapi tidak dipakai. Integrasi antara CORBA and EJB (Enterprise Java Beans) digunakan untuk implementasi aplikasi *frontend*. Karena abstraksi yang kuat pada teknologi yang sudah dipakai, CORBA sebenarnya bisa digantikan oleh teknologi lain, seperti Web service. Namun, sejauh ini CORBA bekerja dengan baik dan masih digunakan untuk implementasi *service* baru. Teknologi *remote procedure call* (RPC) seperti CORBA dan DCOM memungkinkan koneksi jarak jauh antara komponen yang berada di *workstation* dan *server* (Erl, 2005).

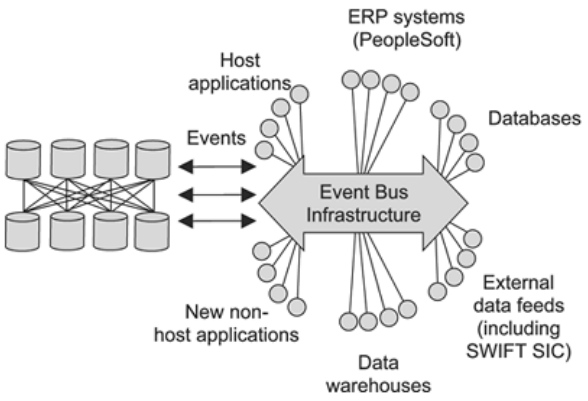
Integrasi Asynchronous dengan EBI

Di tahun 2000, ketika bus informasi bus telah sukses diimplementasikan, Credit Suisse memutuskan menambahkan *platform* integrasi kedua. Ide awalnya adalah untuk menangani integrasi aplikasi *backend-to-backend* (dalam satu domain atau antar domain) dengan konsep yang sama yang telah terbukti sukses ketika mengenalkan SOA dan bus informasi.

Credit Suisse menyebutnya penambahan *bus event* sebuah generalisasi SOA pada arsitektur yang berbasis komponen. Istilah "*service*" digunakan pada bus informasi. Namun, pendekatan pada *bus event* lebih umum dan juga termasuk komunikasi *asynchronous*.

EBI menggunakan kembali semua konsep yang semua digunakan di CSIB Furthermore, CSIB dan EBI berbagi implementasi *service* yang sama, yang memfasilitsi penggunaan kembali logika bisnis dan *live data sharing* di luar lingkup sebuah infrastruktur tunggal yang membuktikan bahwa SOA di CSG benar-benar terlepas dari teknologi yang digunakan.

Infrastruktur *bus event* saat ini berupa solusi integrasi berbasis pesan yang mendukung *routing* berbasis topik dan transformasi seperti pada gambar 4.



Gambar 4 Integrasi Arsitektur CSG termasuk bus event

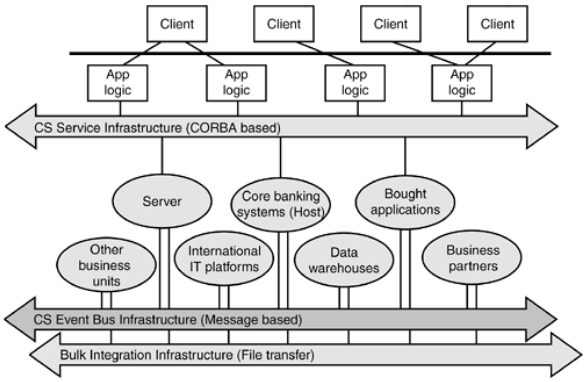
Secara teknis EBI berbasiskan antrian pesan. Implementasi saat ini bergantung pada produk IBM, WebSphere suite. WebSphere MQ (MQ Series) menyediakan dasar yang bisa diandalkan untuk penyimpanan dan transportasi pesan, sementara WebSphere Business Integration Message Broker menyediakan fasilitas untuk trasnformasi pesan dan *publish-and-subscribe*. Credit Suisse secara khusus menekankan pentingnya manajemen interface dan *contract*. Keduanya ini harus dipenuhi untuk memisahkan ketergantungan aplikasi.

Pada kenyataannya, dasar dan proses pada *bus event* sama persis dengan di bus informasi. Ini tidak terbatas pada sifat abstrak dan desain dan proses pengembangan, tetapi juga detail teknis. Oleh karenanya, interface *service* untuk *bus event* dispesifikasikan dengan IDL (Interface Definition Language) dan event dan pesan menggunakan tipe informasi yang sama seperti bus informasi menggunakan kembali struktur data yang telah dimodelkan untuk *service* tersebut.

Bulk Integration Infrastructure

Bulk Integration Infrastructure adalah tipe bus *software* ketiga yang dikembangkan di CSG. Gambar 5 mengilustrasikan hubungan antara ketiga bus. Bus ketiga akan bertanggung jawab utuk manajemen konsisten untuk pertukaran data berbasiskan file.

Bus ketiga ini juga akan menggunakan kembali teknologi yang sama. Saat ini masih dalam pengembangan dan menggunakan transfer file *point-to-point* konvensional. Tujuannya menjadi perantara untuk kontrol tersentralisasi dari beragam transfer data dengan file.



Gambar 5 Hubungan Antara Bus Informasi dan Bus Event

Repositori, Service Interface, dan Contract

Repositori dikembangkan oleh CSG dan berisi informasi tentang desain dan manajemen *service* dan *events*. Untuk setiap *service*, disediakan informasi berikut: (1) *service interface*, yaitu spesifikasi IDL; (2) properti, yaitu yang dimiliki oleh *service*, seperti efek pada data (misalnya, baca, tulis), ketersediaan (misalnya, publik, *private*), dan status saat ini (misalnya, tes, produksi); (3) perencanaan, yaitu informasi mengenai tanggal tes direncanakan atau dibuat dan tanggal *service* dioperasikan; (4) informasi kontak, yaitu informasi orang yang bisa dihubungi, misalnya pengembang, pemilik, desainer, dan atau pengguna; (5) detail implementasi, yaitu informasi tentang implementasi *service* seperti modul dan *database* yang digunakan; (6) kondisi *service*, yaitu deskripsi teks kondisi sebelum dan kondisi sesudahnya yang dijamin oleh *service*; (7) pengecualian, yaitu daftar kode yang tidak dipakai dengan sedikit penjelasan; (8) level *service*, yaitu informasi tentang waktu respon dan perkiraan permintaan per jam, hari, dan bulan (rata-rata dan maksimum); (9) deskripsi parameter, yaitu detail tentang argumen yang digunakan dan dihasilkan *service*.

Choreography, Keamanan, dan Manajemen

Implementasi SOA di Credit Suisse tidak termasuk solusi kompleks SOA untuk *choreography*, keamanan, atau manajemen.

Komponen workflow hanya digunakan untuk *workflow* yang melibatkan pengguna orang. Namun, pada level *service* tidak ada model proses eksplisit yang menyebabkan operasi *service orchestrated* ke dalam workflow yang kompleks. SOA di CSG bisa diklasifikasikan sebagai *networked* pada tahap ekspansi yang membedakan tahap kedewasaan SOA. Credit Suisse terus memonitor fungsionalitas yang terus meningkat yang disediakan dan mempertimbangkan penggabungan komposisi *service* dan model *workflow* untuk perkembangan SOA nya.

Solusi keamanan yang digunakan di Credit Suisse berdasarkan PKI (Public Key Infrastructure). Solusi PKI digunakan untuk integrasi internal saja pada saat ini. Integrasi Eksternal menggunakan koneksi terdedikasi.

Untuk mengatur distribusi *service* dan infrastruktur yang sudah ada beragam alat bantu digunakan. Alat bantu khusus CORBA mendukung manajemen sistem umum. Sebagai tambahan, Credit Suisse membuat alat bantu yang mengakses dan memproses data di sebuah komponen *logging* utama. Komponen *logging* berisi informasi detail tentang *service* yang dijalankan, termasuk informasi input data dan hasilnya ketika *service* dijalankan.

Pilot Project SOA

Walaupun tidak ada rencana bisnis yang dibuat sebelum dikenalkannya SOA, keputusan implementasi ini dibuat langsung oleh CIO Credit Suisse. Implementasi juga berkaitan dengan dua proyek besar di Credit Suisse, yaitu rekonstruksi *data center*, yang diperlukan sejumlah besar merger dan akuisisi, dan membersihkan *data warehouse*. Pengenalan SOA dimulai dengan sebuah proyek pilot skala kecil di tahun 1997, tetapi telah ditetapkan dari awal bahwa akan digunakan untuk keseluruhan infrastruktur IT Credit Suisse.

Lingkup Proyek

Sitem pembukuan sentral di Credit Suisse menangani sekitar lima juta rekening dengan sekitar 218 juta pergerakan rekening per tahun. Arsitektur service Credit Suisse mencakup bisnis perbankan dan juga mencakup Winterthur, yang dimiliki oleh CSG yang juga memiliki infrastruktur IT sendiri berbasiskan SOA.

Credit Suisse memiliki infrastruktur IT tipikal seperti perusahaan finansial besar lainnya dan memiliki sekitar 600 aplikasi, sekitar 12 juta baris kode (untuk sistem inti) dan *platform* beragam (IBM *mainframe*, Unix, Windows). Awal mula sistem IT di tahun 70-an berupa aplikasi berbasis terminal. Pada akhir 80-an dan awal 90-an, aplikasi *client/server* ditambahkan, berdasarkan generator 4GL dan teknologi beriorientasi objek menggunakan Smalltalk. Dengan maraknya internet dan intranet, arsitektur *multi-tier* dibuat dan aplikasi baru kebanyakkan dibangun dengan Java. Namun, aplikasi *mainframe* tetap didukung dan diperbaharui dan tetap dipilih untuk aplikasi berorientasi transaksi. Lingkungan teknologi informasi yang berfokus pada *mainframe* merupakan arsitektur umum di banyak organisasi, terutama di industri finansial seperti bank.

Dampak Bisnis

Dari sudut pandang bisnis infrastruktur SOA harusnya menjadi dasar untuk: (1) kegiatan perbankan *multi-channel*; (2) *online trading*; (3) konsolidasi dengan portfolio aplikasi bisnis inti.

Sebagai fondasi SOA, Credit Suisse mendesain sebuah infrastruktur bisnis penting yang menyediakan: (1) manajemen dan administrasi tersentralisasi; (2) operasional 24 jam dan tujuh hari penuh; (3) mendukung terjadi beberapa ribu pengguna *concurrent*; (4) *throughput* yang tinggi; (5) waktu respon *sub-second* (di bawah satu detik)

Aplikasi yang dibangun pada infrastruktur baru diharapkan menyediakan akses kepada pelanggan lewat internet dan karyawan lewat intranet. Terakhir *gateway* tambahan dibangun untuk merealisasikan integrasi B2B dengan rekan bisnis lewat internet.

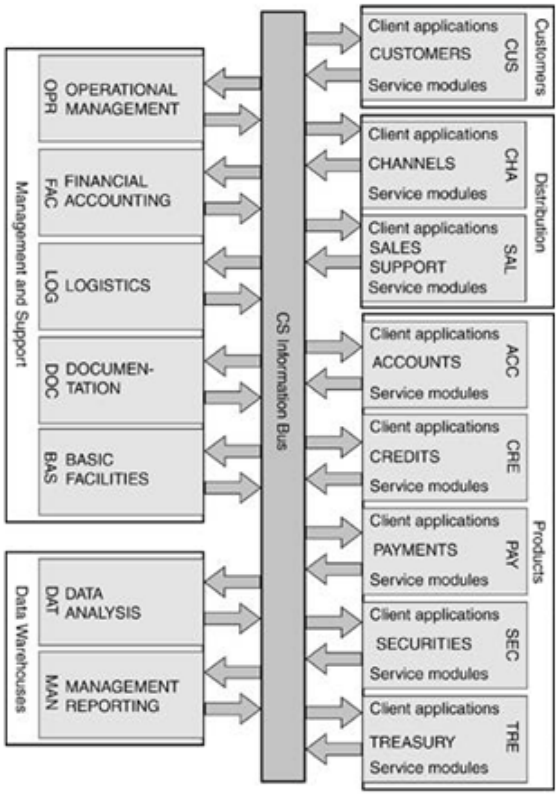
Dampak Teknologi

CSG memiliki lima tujuan utama ketika memperkenalkan arsitektur integrasi berbasis SOA: (1) integrasi teknis, yaitu pengaturan ketergantungan antara *platforms* secara teknis; (2) integrasi logis, yaitu pengaturan ketergantungan antara aplikasi dan komponen pada level semantik bisnis; (3) integrasi proses dan *desktop*, yaitu integrasi aplikasi yang beragam menurut proses bisnis dan *workflow* pengguna; (4) integrasi *software* yang dibeli, dalam hal ini perlunya metode dan alat bantu untuk mengintegrasikan *software* eksternal seefisien mungkin; (5) integrasi B2B, yaitu integrasi dengan rekan bisnis, *supplier*, dan pelanggan.

CSG mencapai kelima tujuan di atas dengan tiga tipe berbeda infrastruktur terintegrasi yang saling melengkapi disertai sebuah infrastruktur *workflow* untuk integrasi proses. Credit Suisse

information bus yang mendukung komunikasi *synchronous, event bus infrastructure* yang mendukung komunikasi *asynchronous*, dan *bulk integration infrastructure* yang menggunakan transfer *file* sebagai basis untuk komunikasi. Bersamaan ketiga infrastruktur ini membentuk fondasi IT Credit Suisse, yang bertujuan menghubungkan aplikasi bisnis berdasarkan *contract* yang telah didefinisikan dengan jelas.

Gambar 6 menunjukkan pendekatan integrasi berdasarkan domain yang menjadi dasar Information Bus. Komunikasi melewati domain yang berbeda, seperti bagian penjualan, logistik, atau analisis data dicapai melalui *information bus* dan memastikan *loose coupling* antara domain tersebut.



Gambar 6 Bus informasi menggabungkan domain yang berbeda

Seperti definisi dari komunikasi *asynchronous*, kriteria teknis utama menjamin sekali dan hanya sekali pengiriman pesan.

Service pada pengguna bisa meminta pesan dan implementasi *service* bisa merespon suatu saat kemudian. *Service* pada pengguna juga akan mengambilnya pada saat respon tersedia. Tidak ada penghalangan pada jalur informasi dan ada lebih banyak hal bisa dikerjakan paralel

Kebutuhan arsitektur utama untuk interaksi berdasarkan pesan adalah: (1) konektivitas *asynchronous* dan transformasi pesan; (2) diseminasi *real-time* data penting; (3) *routing* statis dan berbasiskan *content*; (4) *publish* dan *subscribe* berbasiskan topik; (5) *point-to-point messaging*; (6) peningkatan konsistensi data melewati aplikasi yang berbeda; (7) integrasi standar *software.*

Implementasi SOA

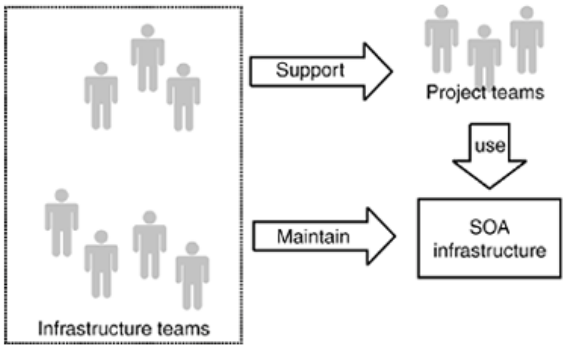
Walau dengan dukungan dari CIO beberapa ketidakpastian tetap terjadi, sebagian besar karena masalah teknis. Problem utama adalah kompleksitasnya dan biaya dan lamanya waktu yang

ditimbulkan. Khususnya untuk memastikan penggunaan ulang *service* yang dianggap sebagai faktor utama yang menyebabkan peningkatan biaya pada proses mendesain. Sebagai tambahan keberatan teknis menggunakan CORBA (*common object request broker architecture*) yang merupakan dasar dari *service bus* CSG meningkat. Pada CORBA, sebagai ganti menyediakan server, yang mengekspos banyak fungsi yang bisa diakses secara *remote*, fungsionalitas dipecah sehingga bisa secara unik diidentifikasi, objek yang secara *remote* bisa mengatur *state* mereka sendiri. Para pengembang percaya bahwa metode ini terlalu banyak menghabiskan sumber daya, dan pengembang dengan Java berargumentasi bahwa pendekatan murni dengan J2EE akan lebih baik dan lebih disukai..

Namun, Credit Suisse menanggapi perlawanan ini dengan serius. Arsitektur ini diberikan sebuah posisi yang utama dan penting dalam departemen IT dan menegaskan bawah pengecualian dan penyimpangan dari pendekatan yang dipilih tidak akan ditoleransi. Penggunaan kembali diprioritaskan, didokumentasikan, dan secara agresif disosialisasikan, sepertinya juga sistem yang *decoupling*. Perjuangan ketat untuk menuju SOA dan aplikasi yang akurat membantu kesuksesan implementasi dan kebanyakkan personel yang keberatan menjadi yakin akan keuntungan SOA.

Proses dan Struktur

Credit Suisse membentuk dua tim infrastruktur untuk menerapkan arsitektur berbasiskan SOA. Tim pertama bertanggung jawab untuk membuat *middleware* yang diperlukan dan yang lain mendukung pengembang menggunakan teknologi. Tim-tim ini bertanggung jawab satu sama lain melalui tugas yang berkaitan dan didukung oleh arsitek integrasi dari departemen arsitektur utama (lihat Gambar 7).



Gambar 7 Tim-tim yang mendukung proyek dan menjaga infrastruktur SOA

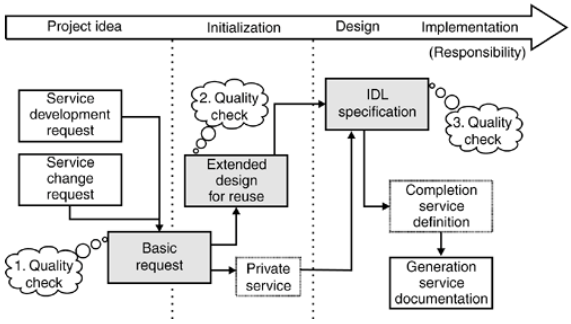
Pertama, tim-tim ini harus membuat detail proses dan struktur yang akan menyertai pengenalan SOA. Detail ini juga akan menyangkut persetujuan untuk *service contract* dan *interface*, sebagai tambahan selain definisi yang jelas untuk desain dan proses pengembangan untuk *service*.
Kedua, tim harus mensosialisasikan konsep ini dan mendukung pengembang dari unit bisnis yang berbeda dengan baik. Tantangannya adalah bagaimana untuk menjual konsep penggunaan kembali, yang pada awalnya hanya menimbulkan perspektif hanya akan menimbulkan lebih banyak biaya dan waktu.

Akhirnya tim-tim bertanggung jawab untuk meninjau ulang definisi *service* dan bertindak sebagai kendali mutu. Sekali lagi penggunaan kembali adalah konsep kunci di sini dan faktor yang membedakannya dengan manajemen proyek “tradisional”. Tim tersebut harus memastikan bahwa pengembangan *service* harus diikuti dengan detail proses dan dipenuhinya kriteria yang diperlukan untuk integrasi arsitektur. Khususnya mereka harus memastikan bahwa unit bisnis tidak mengabaikan SOA dan mendapatkan ijin dari pihak manajemen untuk membuat “pengecualian”.

Salah satu aspek penting dari SOA adalah membuat definisi proses yang jelas untuk pembuatan *service*. Proses ini dimulai dengan komunikasi antara konsumen service dan menghasilkan spesifikasi yang sangat detail. Berdasarkan spesifikasi, sebuah keputusan diambil untuk membangun *service* baru atau menggunakan *service* yang sudah ada, yang mungkin dimodifikasi atau dikembangkan untuk memenuhi kebutuhan aplikasi baru. Peninjau arsitektur meninjau ulang keputusan ini sebelum desain dan implementasi service bisa mulai dilakukan. Peninjau arstitektur sendiri terdiri dari designer berpengalaman yang dari grup utama arsitektur dan dari tim pendukung.

Pengembangan service berlanjut dengan pendekatan *bottom-up* yang ketat sebelum perencanaan dasar *service* dimulai. Sebaliknya sebuah *service* akan didefinisikan dan diimplementasikan ketika aplikasi tertentu membutuhkannya, seperti pada Gambar 8. Hal ini memungkinkan konstruksi arsitektur service dalam proses *incremental* dan efisien.

Pendekatan *top-down* berdasarkan definisi formal *interface* seperti IDL atau WSDL. Pendekatan *bottom-up* menganalisa kode dari implementasi *service*.



Gambar 8 Desain proses berdasarkan SOA di CSG: Permintaan dibuat secara *bottom-up*, dan kualitas dipastikan secara *top-down*

Repository Service

Credit Suisse menggunakan sebuah pusat repositori untuk mempublikasi semua informasi relevan tentang sebuah *service*. Repositori ini berisi deskripsi *service* pada level bisnis dan spesifikasi teknis *interface* berdasarkan IDL (Interface Definition Language CORBA).

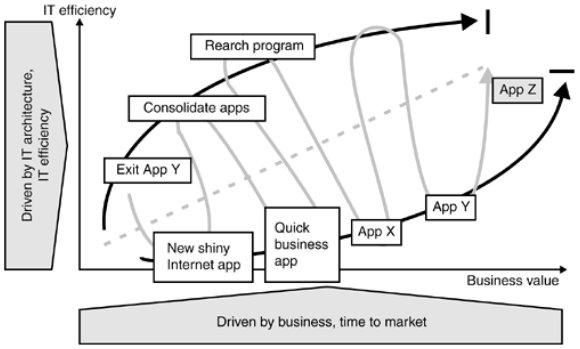
Biasanya, grup pengguna yang berbeda memerlukan keduanya, tetapi di beberapa kasus, pertama pengembang mengakses deskripsi level bisnis untuk menghasilkan spesifikasi detail lalu deskripsi IDL pada saat fase implementasi yang sebenarnya.

Repositori juga berisi informasi mengenai pembaharuan *service*, yang khususnya penting untuk mengembangkan *service* Credit Suisse.

Manajemen Proyek

Seperti penjelasan sebelumnya, kompleksitas infrasttruktur IT CSG mencapai situasi kritis di akhir 90-an karena implementasi ulang dari fungsionalitas yang ada secara ekonomis dan teknis tidak mungkin. Pendekatan *Managed Evolution* diterapkan, yaitu dengan transformasi tahap demi tahap sitem yang ada ke sistem yang baru. Setiap proyek dalam tahapan ini diharapkan bisa mengkontribusikan kepada optimasi sturuktural sistem IT.

Gambar 9 menggambarkan prinsip *Managed Evolution* yang berisi dua dimensi dasar dari infrastruktur IT, efisiensi IT dan nilai bisnis. Efisiensi IT menangkap seberapa cepat dan tidak mahalnya sistem bisa diubah. Nilai bisnis memberikan manfaat dari infrastruktur untuk pengguna, yaitu unit bisnis di perusahaan. Setiap proyek idealnya harus mengkontibusikan dua dimensi ini.



Gambar 9 Prinsip *managed evolution*

Untuk menjamin kontribusi pada level ini kepada keseluruhan arsitektur, kejelasan panduan, proses, dan struktur harus dibangun untuk membantu membedakan "kesempatan baik dari yang buruk". Credit Suisse memiliki tim khusus yang bertanggung jawab untuk panduan tersebut dan memonitor aplikasi mereka langsung di bawah CIO.

Metodologi yang digunakan di Credit Suisse untuk mengatur proyek bisnis kebanyakan berdasarkan model *waterfall*. Namun, faktor penting kesuksesan proyek tersebut adalah kendali atas penyerahan proyek, seperti spesifikasi tampilan dan koordinasi dengan tim infrastruktur yang sesuai.

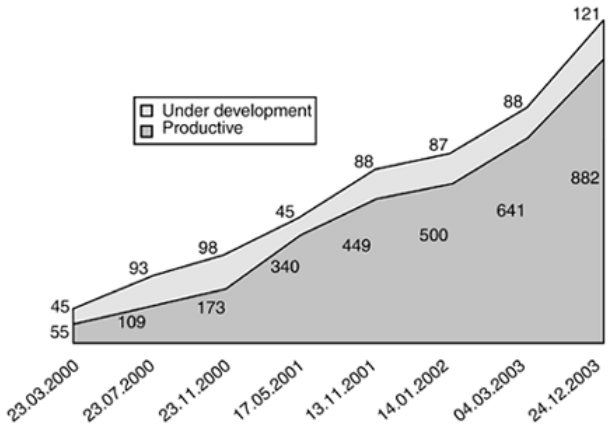
Hasil dan Keuntungan Implementasi SOA

Aplikasi *backend* inti untuk integrasi sebagian besar berjalan di IMS (80%) dan CICS (20%) pada *mainframe* S/390 besar. Aplikasi ditempatkan pada *server* Unix dan Windows. Aplikasi *frontend*, yang sedang digunakan dan sebagian dalam pengembangan menggunakan J2EE, C++, Smalltalk, HTML, COM, dan Visual Basic.

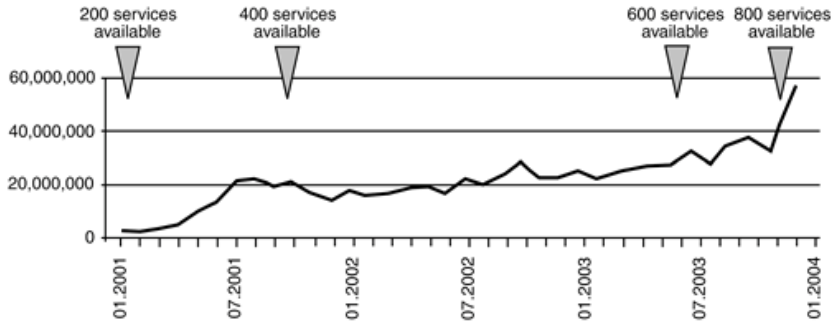
Ketika Information Bus dioperasikan tahun 1999, lima aplikasi *frontend* ditempatkan, menyediakan 35 *service* ke sekitar 800 pengguna. Satu tahun kemudian, aplikasi yang didukung meningkat menjadi 21 dengan 173 *service* dan 9.000 pengguna. Angka ini meningkat drastis menjadi 500 *service* di tahun 2000, digunakan lebih dari 50 aplikasi *frontend* dan lebih dari 100.000 pengguna (termasuk baik pelanggan *Internet banking* dan staff internal).

Tahun 2004 bus informasi menawarkan lebih dari 700 *service* dan menangani antara 1,5 and 2 juta permintaan, dengan total 10 juta permintaan setiap minggu atau 50 juta per bulannya. Event Bus yang menangani sekitar 20 aplikasi memproses 500.000 pesan lainnya per hari.

Gambar 10 menunjukkan pengembangan *service* yang tersedia di Credit Suisse dari tahun 2000 sampai 2003. Gambar 11 menunjukkan perkembangan penggunaan *service* tersebut. Stagnasi yang terjadi pada tahun 2002 dan 2002 dikarenakan keadaan bisnis yang sulit.



Gambar 10 Pengembangan *service* yang tersedia di CSG dari Tahun 2000 Sampai 2003



Gambar 11 Permintaan *service* di CSG dari tahun 2000 – 2003

Ada beberapa keuntungan utama yang dialami oleh CSG.

Pertama, penggunaan kembali *service*. Service bisnis digunakan di berbagai aplikasi. Walau tingkat rata penggunakan kembali hanya 1,6 ketika menghitung untuk keseluruhan *service*, tetapi beberapa *service* bisnis bisa digunakan kembali sampai 12 aplikasi. Rendahnya nilai rata-rata disebabkan fakta bahwa banya *service* saat ini sedang digunakan oleh satu aplikasi saja. Karena service pada mulanya dibangun ketika ada seorang pengguna yang memintanya, akan diperlukan waktu sebelum pengguna lain ingin menggunakannya. Penggunaan kembali didorong oleh repositori tersentralisasi yang berisi *service interface* dan dokumentasi yang detail.

Keuntungan kedua adalah pengembangan aplikasi yang lebih efisien. Karena banyak penggunaan kembali *service*, pengembangan aplikasi telah dipercepat dengan signifikan. Biasanya, ketika sebuah aplikasi dikembangkan, 75 sampai 80 persen dari service yang diperlukan telah ada di *repository*. Ini meningkatkan waktu yang diperlukan untuk mengimplementasikan solusi baru secara dramatis dan juga menawarkan penghematan biaya.

Keuntungan lainnya adalah peningkatan kolaborasi antara pengembang unit bisnis dengan programmer yang mengimplementasikan aplikasi bisnis inti. Juga dapat diamati, bahwa programmer PL/1 yang berpengalaman yang telah menjadi tidak termotivasi beberapa tahun ini menjadi aktif berpartisipasi dalam proses pengembangan.

PENUTUP

Keuntungan yang didapat dari implementasi SOA tidak dicapai tanpa kerja keras. CSG mengalami adanya ketidakpastian secara kontinu mengenai pendekatan yang digunakan untuk mengintegrasikan arsitektur, misalnya keluhan bahwa CORBA terlalu menghabiskan banyak sumber daya, terlalu kompleks, dan terlalu lambat. Keberatan ini tidak disampaikan oleh semua pihak, tetapi dukungan konsisten dari pihak manajemen atas mengatasi hal ini.

Ada juga tahap kritis ketika Information Bus terancam gagal karena kesuksesannya. Semakin banyak pengguna mengakses aplikasi yang dibangun di atas CSIB, kinerja, keandalan, dan ketersediaaan menjadi harus selalu ada. Sekali lagi memiliki pihak manajemen yang mendukung dan anggaran yang mencukupi mengatasi masalah pada tahap kritis ini.

Juga diketahui bahwa *decoupling* yang belum dicapai untuk integrasi internal tidak harus dipenuhi untuk integrasi eksternal, yang menuntut persyaratan lebih.

Pada akhirnya, pendekatan *bottom-up* yang ketat diterapkan ke seluruh pengembangan SOA mungkin akan dilengkapi dengan pertimbangan mungkin akan dilengkapi dengan pertimbangan *top-down* di masa depan. Hal ini termasuk pengambilan keputusan yang lebih sistematis mengenai penggunaan kembali dan target yang lebih spesifik untuk pengembang *service*. Salah satu idenya adalah mengurangi biaya untuk mengembangkan *service* yang mungkin tidak akan digunakan. Aspek lain adalah indentifikasi *service* yang hilang, ketika mereka tidak secara langsung sedang dibutuhkan oleh sebuah aplikasi.

Credit Suisse menekankan empat aspek utama, yaitu *interface,* proses, komitmen manajemen, dan teknologi yang solid. Bukti dari kesuksesan integrasi arsitektur berbasis SOA adalah didasarkan fakta bahwa konsep dan metodologi yang pada awalnya dibangun untuk bus informasi *synchronous* bisa digunakan kembali satu per satu ketika memperkenalkan bus event. Lebih jauh lagi, implementasi integrasi struktur *bulk* juga didasari pada dasar yang sama. Hal ini mendemonstrasikan bahwa kedua konsep dan metodologi benar-benar menghasilkan hasil yang diinginkan dan bahwa ketiganya tidak terkait dengan teknologi yang dipakai.

DAFTAR PUSTAKA

Erl, T. (2005). *Service-Oriented Architecture: Concepts, Technology, and Design.* New Jersey: Prentice Hall.

Herr, M., Bath, U., & Koschel, A. (2004). Implementation of a service oriented architecture at deutsche post MAIL. *Lecture Notes in Computer Science, 3250*, 227-238.

Rodriguez, Juan M., Crasso, M., Mateos, C., Zunino, A., Campo, M. (2013). Bottom-up and top-down cobol system migration to web services. *Internet Computing, IEEE, 17* (2), 44 – 51.

Rong Liu, Wu, F., Patnaik, Y., Kumaran, S. (2009). Business entities: an SOA approach to progressive core banking renovation. *Services Computing, 2009. SCC '09. IEEE International Conference,* 466 - 473, 21-25 Sept.