

# Bab 1

## SISTEM INFORMASI

---

### 1.1 Istilah Sistem Informasi

Sistem informasi, secara umum didefinisikan oleh H. M. Jogianto (2006) sebagai berikut:

*“Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan sasaran yang tertentu”*

Sedangkan, Kenneth Laudon C (2012) mendefinisikan sistem informasi sebagai berikut:

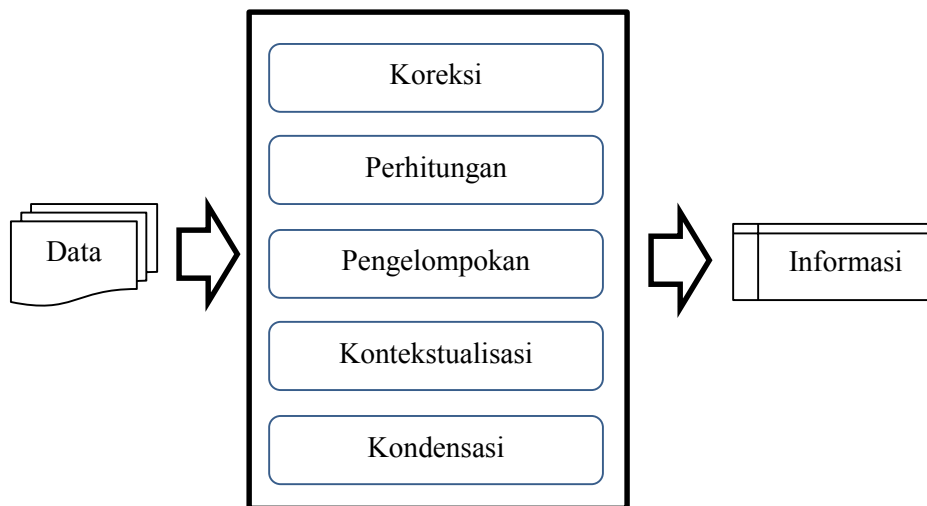
*“a set of interrelated components that collect (or retrieve), process, store, and distribute information to support decision making and control in an organization. In addition to supporting decision making, coordination and control, information systems may also help managers and workers analyze problems, visualize complex subjects, and create new products.”*

Dengan demikian, sistem informasi bisa didefinisikan dari sisi prosedur maupun dari sisi komponen pendukungnya. Intinya adalah sebuah sistem yang memanfaatkan data dengan peralatan teknologi yang memadai sehingga bisa menghasilkan informasi yang dapat digunakan dalam menunjang berbagai proses bisnis mulai dari level operasional sampai dengan level pengambilan keputusan tertinggi.

Pada era digital seperti saat ini sistem informasi tidak terlepas dari bantuan perangkat komputer dan jaringan internet/intranet, sehingga sering disebut juga dengan sebutan lengkap *computer based information system*. Saat ini, apabila disebut dengan sistem informasi artinya sudah pasti mengacu pada suatu sistem informasi yang menggunakan komputer sebagai alat bantu, dan harus bisa diakses melalui internet. Seiring dengan perkembangan teknologi informasi terutama perangkat *mobile* seperti *tablet*, *smartphone*,

*laptop*, maka sistem informasi saat ini dituntut untuk bisa diakses dengan cepat dari dan kapan saja.

Sistem informasi memiliki beberapa ciri atau karakteristik yaitu komponen sistem (*componenents*), batasan sistem (*boundary*), lingkungan sistem (*environments*), penghubung sistem (*interface*), Masukan sistem (*input*), pengolahan (*processing*), keluaran sistem (*output*), sasaran (*objective*) dan tujuan (*goal*)



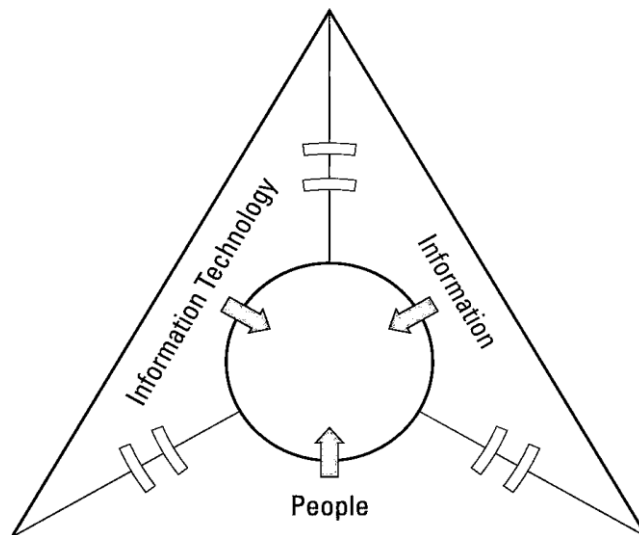
Gambar 1.1 Hubungan Data dan Informasi<sup>[11]</sup>

Sebagaimana yang ditunjukkan pada gambar 1.1, sebuah sistem informasi akan memasukkan data, kemudian memprosesnya sedemikian rupa sesuai dengan jenis atau karakter masing-masing sistem, sehingga akhirnya dihasilkan informasi yang berkualitas yaitu valid bebas dari kesalahan, tepat waktu, dan relevan dengan kebutuhan proses bisnis.

## 1.2 Sumber Daya Teknologi Informasi

Sebagaimana sumber daya lain, sumber daya teknologi informasi memiliki peranan yang sangat penting dalam menunjang kesuksesan penerapan sistem informasi di dalam suatu *enterprise*. Pemanfaatan teknologi informasi secara tepat dapat mendukung sepenuhnya keseluruhan rangkaian proses bisnis dalam sebuah *enterprise*. Sumber daya teknologi informasi pada

dasarnya terdiri dari tiga komponen utama sebagaimana dijelaskan pada gambar 1.2.



Gambar 1.2 Sumber Daya Teknologi Informasi<sup>[15]</sup>

1. *Information Technology*

Teknologi informasi adalah salah satu pendukung utama suatu *enterprise*. Infrastruktur teknologi informasi yang dibutuhkan untuk mendukung kelancaran proses bisnis secara menyeluruh. Infrastruruktur ini mencakup *software*, *hardware* dan *communication technology*.

2. *Information*

Sumber pengambilan kebijakan dan keputusan oleh suatu *enterprise* yang bersasa dari data-data yang terkumpul dari dalam dan luar *enterprise*.

3. *People*

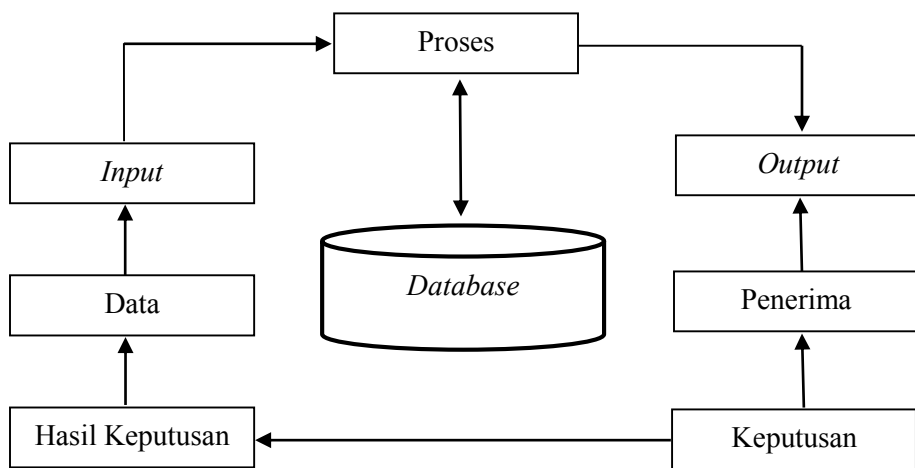
Yang menjadi kunci utama sebagai pelaku sistem yang telah disediakan. karena pada dasarnya sebuah sistem bukanlah sebuah robot pintar yang bisa berjalan sendiri dan mengambil keputusan sendiri.

Keterlibatan orang dalam menjalankan sistem berbasis komputer sangat dominan, dan sangat menentukan kualitas dan keberlangsungan sumber daya lainnya. Oleh karena itu, dalam pengembangan arsitektur *enterprise*, pembahasan tentang *who* artinya siapa yang terlibat

menjadi perhatian khusus sehingga dimasukkan ke dalam pembahasan *framework*.

### 1.3 Teknologi Pendukung

Secara umum komponen utama yang menyusun sebuah sistem informasi berbasis komputer dan jaringan komputer dapat dijelaskan melalui siklus sistem informasi seperti pada gambar 1.3.



Gambar 1.3 Siklus Sistem Informasi<sup>[11]</sup>

Teknologi *database* (penyimpanan data) merupakan *core technology* dari suatu sistem informasi karena data dan informasi akan disimpan dan dikelola sedemikian rupa secara efisien dan efektif di dalamnya, sehingga akan menjadi sumber daya yang sangat penting bagi berbagai macam sistem informasi yang terkait. Oleh karena itu, dalam pembuatan sistem informasi harus dipilih teknologi *database* yang *reliable* disesuaikan dengan kapasitas data yang harus disediakan, terutama untuk *enterprise* yang menjalankan bisnis dengan menggunakan data-data sangat penting berbagai pihak seperti bisnis perbankan.

Komponen pendukung sebuah sistem informasi berbasis komputer adalah sebagai berikut.

1. *Software*

Program aplikasi komputer yang secara langsung mendukung jalannya proses bisnis, dimana dalam pembuatannya memerlukan *programming language*, *database management system* (DBMS), *operating system*, sistem sekuriti, dll.

2. *Hardware*

Perangkat keras merupakan perangkat dasar berupa barang elektronik yang diperlukan untuk menjalankan berbagai *software* pendukung sebuah sistem informasi, seperti diantaranya komputer, *harddisk*, *flashdisk*, printer, *scanner*, dll.

3. *Communication*

Teknologi komunikasi merupakan teknologi infrastruktur untuk mengfungsikan dan mengoptimalkan *software* dan *hardware*. Teknologi ini mencakup berbagai alat untuk membangun suatu jaringan komputer baik intranet atau internet. Teknologi yang bisa digunakan saat ini ada banyak alternatifnya, tidak hanya teknologi jaringan berbasis kabel, termasuk teknologi *wireless* yang sangat memudahkan proses komunikasi antar berbagai alat komputer seperti *Wifi*, jaringan GSM, dll.

*Open source* adalah salah satu alternatif solusi pemanfaatan *software* secara legal dan terpercaya dimana *source code* disediakan secara bebas dan digunakan secara proposional disesuaikan dengan kebutuhannya masing-masing dalam suatu pengembangan aplikasi sistem informasi. Kehandalan *software* yang bersifat *open source* tidak jauh berbeda dengan yang bersifat *proprietary*, bahkan beberapa lebih unggul karena dikembangkan bersama-sama oleh suatu tim, dan tujuannya untuk kepentingan bersama. Beberapa *software open source* yang populer dan standar digunakan dalam pengembangan sistem informasi adalah sebagai berikut.

1. *Programming Language*

Bahasa pemrograman untuk kodifikasi program aplikasi sistem informasi, diantaranya adalah Java, PHP, Python, dll.

2. *Database Management System*

*Software* yang dikhususkan untuk pengelolaan data dalam skala besar, diantaranya adalah PostgreSQL, MySQL, FireBird, dll.

3. *Integrated Development Environment (IDE) Tool*  
Software pendukung untuk memberikan kemudahan bagi pengembang dalam mengembangkan aplikasi sistem informasi, seperti Eclipse, NetBeans, dll.
4. *Framework*  
Software pendukung untuk dalam pengembangan aplikasi sehingga aplikasi terbagi menjadi beberapa bagian aplikasi yang bisa dikembangkan secara terpisah, seperti CodeIgniter, Spring, Struts, Maven, dll.
5. *Operating System*  
Sistem *software* yang merupakan *core software* sebagai *platform* aplikasi di komputer. Sistem operasi yang berbasis unix seperti Linux, FreeBSD, OpenBSD. Selain itu ada juga *operating system* lain yang tidak berbasis unix seperti ReactOS.
6. *Virtual Private Network*  
Software yang digunakan untuk membangun jaringan lokal secara virtual, dengan memanfaatkan jaringan publik seperti internet. Sebagai contoh OpenVPN.

## 1.4 Tipe Sistem Informasi

Dalam suatu *enterprise* terdapat banyak pihak-pihak yang akan memanfaatkan teknologi informasi untuk menjalankan proses bisnisnya masing-masing. Berbagai jenis sistem informasi dibutuhkan disesuaikan dengan tingkat manajerial atau spesifikasi bisnis yang dilakukan. Dengan demikian, sistem informasi bisa dikelompokkan berdasarkan kebutuhan fungsi manajerial *enterprise*.

Sistem informasi yang dibutuhkan untuk mendukung proses bisnis *enterprise* secara umum bisa dikelompokkan ke dalam empat kategori sebagai berikut:

1. TPS (*Transaction Processing System*)  
Sistem yang sifatnya transaksional dan operasional, dan merupakan jenis sistem informasi yang akan memberikan masukan data (penyedia *raw data*) bagi sistem informasi yang lainnya. Sistem ini pada dasarnya hanya mendukung kegiatan operasional bisnis yang menyangkut berbagai macam transaksi. Oleh karena itu, kedudukan sistem ini sangat penting karena sebagai fondasi sistem yang menjadi sumber data

bagi sistem informasi yang lain di atasnya, dan akan sangat menentukan kualitas informasi yang dihasilkan oleh sistem informasi MIS, DSS dan EIS. Dengan demikian, proses validasi data merupakan hal yang mutlak dilakukan dalam pembuatan sistem ini, agar dijamin tidak terjadi kesalahan pada proses pemasukan data.

2. MIS (*Management Information System*)

Sistem informasi yang berada satu level di atas TPS. Sistem ini yang mendukung kegiatan bisnis pada level manajerial dengan memanfaatkan data-data yang telah diproses melalui TPS. Data yang dikumpulkan oleh TPS, selanjutnya akan diolah lebih lanjut, sehingga menghasilkan suatu informasi dalam mengontrol pelaksanaan kegiatan bisnis. Untuk mengolah data menjadi informasi dalam sistem informasi ini, biasanya cukup menggunakan fasilitas pengolahan *query* dengan bantuan *Structure Query Language* (SQL). *Query* yang dibangun disesuaikan dengan kebutuhan bisnis setiap sistem sebagaimana karakteristik proses bisnisnya masing-masing.

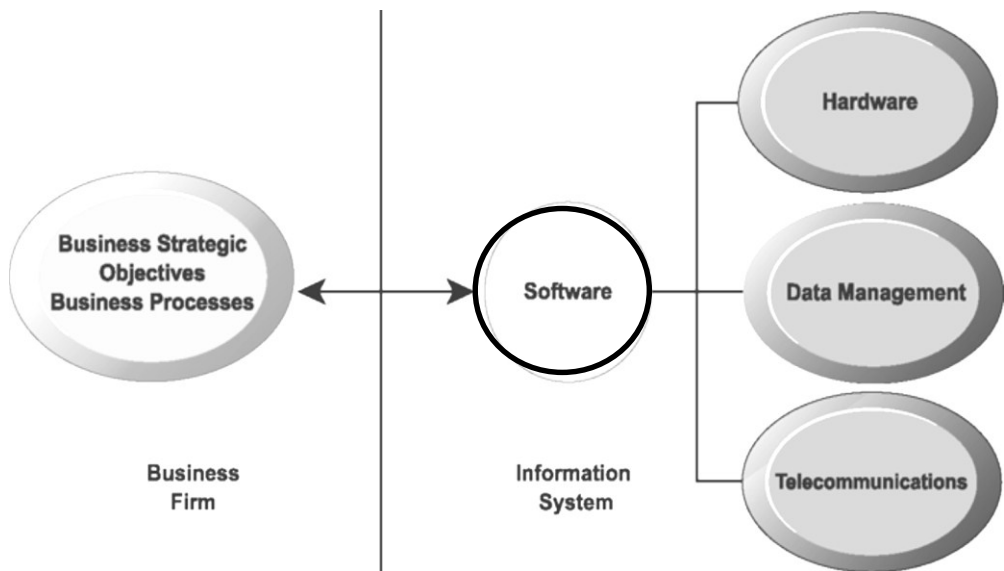
3. DSS (*Decision Support System*)

Sistem ini mendukung kegiatan yang bersifat analitis dengan memanfaatkan bongkahan data berukuran besar untuk diproses lebih lanjut, sehingga menghasilkan pengetahuan tentang apa sebenarnya yang terkandung dalam bongkahan data tersebut. Untuk pengolahan datanya bisa digunakan teknologi *Data Mining*, sehingga bisa menggali informasi yang penting dari suatu bongkahan data yang sangat penting dalam proses pengambilan keputusan. Bongkahan data merupakan kumpulan data dalam kurun waktu yang cukup lama, perlu dilakukan proses denormalisasi terhadap struktur datanya agar pemrosesan lebih cepat. Dalam hal ini istilah *Data Warehouse* merupakan salah satu pekerjaan yang dilakukan dalam membangun DSS. Dengan demikian *Data Warehouse* dan *Data Mining* merupakan dua komponen utama yang saling melengkapi dalam membentuk sistem informasi jenis ini.

4. EIS (*Executive Information System*)

Sistem yang digunakan untuk kepentingan para manajer puncak dalam suatu *enterprise* agar bisa memonitor kinerja bisnis secara keseluruhan dan *up-to-date* hanya dengan melihat informasi yang ditampilkan dalam bentuk grafik. Berkaitan dengan sistem informasi jenis ini, dikenal juga istilah *War Room* yaitu mengacu pada suatu ruangan khusus yang dipergunakan untuk pengolahan data *enterprise* secara menyeluruh baik data dari dalam maupun dari luar *enterprise*, yang ditampilkan secara visual dalam berbagai bentuk grafik, agar informasi yang ditampilkan benar-benar dapat dipahami secara mudah dan cepat

oleh para manajer puncak dalam membantu pengambilan keputusan, kebijakan, strategi bisnis.



Gambar 1.4 Hubungan Sistem Informasi Dan Proses Bisnis<sup>[15]</sup>

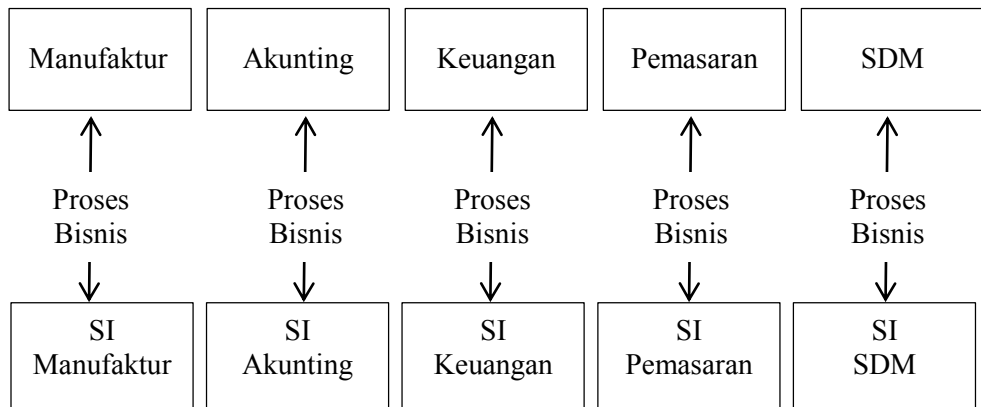
Gambar 1.4 menunjukkan secara umum bahwa suatu *enterprise* sangat membutuhkan keberadaan sistem informasi berbasis komputer yang mencakup berbagai proses bisnis yang dijalankan sehingga bisa mendukung kelancaran proses bisnisnya dan meningkatkan kinerja *enterprise* secara keseluruhan. Idealnya tidak ada satu proses bisnis pun yang tidak didukung oleh *software* sistem informasi.

Sebagaimana yang ditunjukkan pada gambar 1.5, bahwa konsep sistem informasi yang bersifat konvensional merupakan sistem yang terkotak-kotak satu sama lain, masing-masing sistem mandiri walaupun secara kebutuhan sistem informasi yang tersedia lengkap dan berada dalam satu lingkungan *enterprise*<sup>[3]</sup>. Dengan demikian, hal ini menyebabkan tidak terjadinya *data sharing* di antara sistem yang ada, dan akan mengakibatkan berbagai masalah diantaranya sebagai berikut:

1. Duplikasi data di dalam sistem yang berbeda.
2. Bisa saja sistem informasi yang dibutuhkan lengkap tetapi tidak bisa diintegrasikan.
3. Manajer puncak akan kesulitan untuk mendapatkan informasi yang menyeluruh secara *enterprise*.

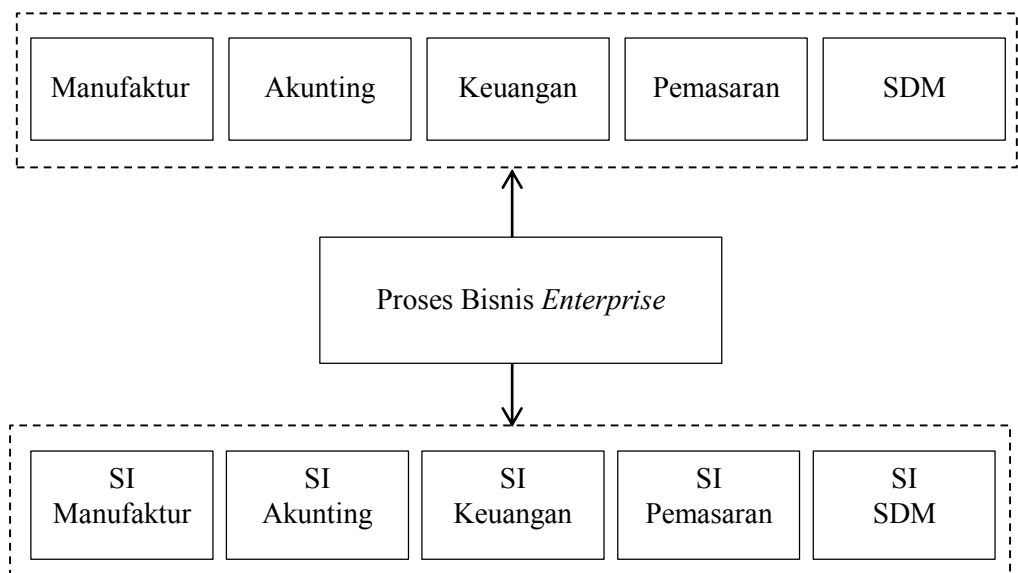


#### 4. Pemeliharaan sistem lebih kompleks.



Gambar 1.5 Sistem Informasi Konvensional

Sedangkan dalam sistem informasi *enterprise*, seluruh sistem informasi yang diperlukan secara keseluruhan harus lengkap dan diintegrasikan menjadi satu kesatuan sistem yang saling terkait satu sama lain, baik dari aspek data, aplikasi maupun infrastrukturnya, sehingga semua permasalahan dalam sistem informasi konvensional bisa dipecahkan, seperti yang dijelaskan secara konsep pada gambar 1.6.



Gambar 1.6 Sistem Informasi *Enterprise*

Gambar 1.6 menjelaskan bahwa seluruh proses bisnis *enterprise* didukung oleh sistem informasi terkait yang merupakan satu kesatuan sistem yang saling terhubung satu sama lain (tidak terpisah seperti pada gambar 1.5). Dari sisi data, setiap sistem bisa menggunakan data yang dibuat oleh sistem yang lain, dan tidak boleh terjadi duplikasi data yang akan menghasilkan informasi yang tidak valid dan tidak berkualitas. Sedangkan, dari sisi aplikasi setiap sistem memiliki aplikasi yang berbeda, tetapi bisa memiliki komponen sistem yang dipakai bersama, untuk memudahkan proses pemeliharaan sistem secara keseluruhan. Dengan kemajuan teknologi *software* saat ini, keseluruhan sistem yang diperlukan tidak harus berada dalam suatu *server* yang sama secara fisik, tetapi sistem tersebar di *server* yang berbeda dengan tetap terjaga menjadi suatu sistem informasi *enterprise* yang terpadu.

*Enterprise Resource Planning* (ERP) merupakan salah satu contoh kongkrit implementasi dari *integrated enterprise information system*. ERP saat ini lebih merupakan suatu paket aplikasi sistem informasi *enterprise* yang mendukung keseluruhan proses bisnis *enterprise* terutama *enterprise* dengan kategori manufaktur. Berbeda dengan jenis *software* lainnya, ERP merupakan paket *software* yang bisa diubah sesuai dengan kebutuhan masing-masing *enterprise* baik ERP yang bersifat *open source* maupun yang bersifat *proprietary*. Karena proses bisnis yang didukung oleh ERP bersifat umum, maka banyak *enterprise* yang lebih menyesuaikan proses bisnisnya dengan aplikasi yang tersedia di ERP.

ERP saat ini tidak hanya untuk kepentingan *enterprise* yang berbentuk manufaktur, tetapi sudah berkembang menjadi beberapa jenis ERP, yang bisa mendukung jenis *enterprise* lainnya seperti rumah sakit, perguruan tinggi, sekolah, lembaga pemerintah, dll. ERP *open source* seperti Compiere, OpenBravo, OpenERP, dll. Sedangkan ERP *proprietary* seperti SAP, Web Dynamic, dll.

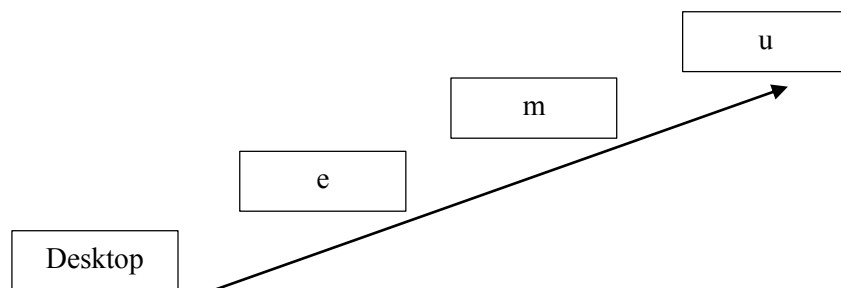
Untuk menentukan ERP mana yang lebih baik digunakan, tentunya harus dilakukan kajian berupa perencanaan arsitektur *enterprise* terlebih dahulu untuk mengetahui karakter proses bisnisnya, keperluan data, aplikasi dan infrastruktur teknologi. Dengan demikian, berdasarkan kajian tersebut bisa diambil keputusan paket ERP mana yang sebaiknya diimplementasikan, disesuaikan dengan ketersediaan dana dan kebutuhan fungsinya.

ERP yang dikhususkan untuk mendukung kelancaran proses bisnis di lingkungan pemerintahan lebih dikenal dengan istilah e-Governance. Dalam rangka mewujudkan tata-kelola pemerintahan yang lebih bersih dan transparan, penerapan e-Governance mulai dijalankan di berbagai level pemerintahan, walaupun belum mencakup keseluruhan proses bisnis. Sebagai contoh adalah Sistem Informasi Rumah Sakit (SIRS) berbasis *open source*

yang disediakan oleh Kementerian Kesehatan untuk pengelolaan rumah sakit khususnya rumah sakit milik pemerintah.

## 1.5 Evolusi Sistem Informasi

Seiring dengan perkembangan teknologi informasi, istilah sistem informasi mengalami evolusi seperti pada gambar 1.7 yaitu dari sistem yang berbasis desktop lalu menjadi segala sesuatu sistem pengolahan data berbasis jaringan komputer yang penamaannya dimulai dengan awalan “e”. Pada awalnya muncul istilah *email* sebagai bentuk digital *mail* menggantikan *mail* konvensional. Kemudian, berkembanglah berbagai jenis sistem informasi yang masing-masing sehingga muncul istilah-istilah seperti e-Learning, e-Library, e-Commerce, e-KTP dll. Dengan demikian, muncullah secara umumnya istilah e-Business yang mengacu kepada sistem informasi berbasis web supaya sistem bisa diakses dari mana saja dan kapan saja.



Gambar 1.7 Evolusi Sistem Informasi

Huruf “e” sendiri sebenarnya berasal dari kata Bahasa Inggris “electronic”. Secara harfiah berarti segala sesuatu sistem informasi yang menggunakan perangkat elektronik. Kenyataannya huruf “e” bermakna bukan hanya sekedar elektronik, tetapi lebih luas yaitu mengandung pengertian perangkat komputer yang terkoneksi melalui jaringan internet. Dengan demikian huruf “e” bermakna segala sesuatu sistem informasi yang memanfaatkan komputer dan jaringan internet dalam proses pengolahan datanya. Oleh karena itu, saat ini penamaan sebuah sistem informasi lebih populer menggunakan awalan “e” dari pada hanya sekedar nama sistem informasi.

Kecenderungan penggunaan *gadget* berupa perangkat *smartphone* atau *tablet* didukung dengan tersedianya berbagai *software* yang bersifat *open source*, memungkinkan akses ke sistem informasi melalui internet semakin bervariasi, sehingga penggunaan sistem informasi bisa dilakukan dari mana saja dan kapan saja. Oleh sebab itu, muncullah istilah *mobile computing*. Namun demikian dalam perkembangannya istilah sistem informasi dengan diawali huruf “m” sebagai pengganti dari “e” kurang populer.

Selain istilah *mobile* muncul juga istilah *ubiquitous* terkait perkembangan peralatan apa saja yang bisa dipakai dan tidak harus selalu berbentuk komputer (*wearable computing*) pada umumnya, tetapi bisa berbentuk jam tangan, kacamata, dll. Sehingga muncul istilah aplikasi yang bisa akses ke suatu sistem informasi melalui perangkat seperti ini yang disebut dengan istilah yang dimulai dengan huruf “u”.

Apapun istilahnya dan apapun perangkat penghubungnya, bahwa pesatnya perkembangan teknologi informasi ditandai dengan semakin beragam dan semakin praktis perangkat pengakses sistem informasi sehingga bisa digunakan oleh siapa saja yang ingin mengakses ke sistem informasi, dan dimana serta kapan saja.

# Bab 2

## ARSITEKTUR *ENTERPRISE*

---

### 2.1 Pengertian Arsitektur

Salah satu institusi terkenal di bidang teknologi dan sistem informasi bernama *Chief Information Officer Council* menyatakan bahwa arsitektur (*architecture*) dalam pembahasan tentang sistem informasi berskala *enterprise* didefinisikan sebagai berikut<sup>[7]</sup>:

*“The structure of components, their interrelationships, and the principles and guidelines governing their design and evolution over time.”*

Pengertian tersebut memiliki makna bahwa yang dimaksud dengan arsitektur pada dasarnya adalah berupa gambaran sebuah struktur perusahaan/lembaga/institusi yang terdiri dari berbagai macam komponen pendukung dan relasi antar komponen tersebut. Disamping itu, juga mengandung prinsip dan petunjuk yang bisa digunakan untuk bagaimana mengevolusikan baik merubah atau memperbaiki struktur yang ada menjadi struktur yang baru yang lebih terpadu dalam kurun waktu yang ditentukan.

Pengertian mengenai arsitektur dalam sistem informasi biasanya terdiri dari:

1. Gambaran kondisi saat ini
2. *Blueprint* atau visi ke depan
3. *Roadmap* perencanaan untuk pencapaian visi

### 2.2 Pengertian *Enterprise*

Institusi *Chief Information Officer Council* memberikan batasan terkait dengan pengertian kata *enterprise*, yaitu dengan mendefinisikannya sebagai berikut:

*“An organization supporting a defined business scope and mission. An enterprise includes interdependent resources (people, organization, technology) who must coordinate their functions and share information in support of a common mission.”*

Kata *enterprise* sebenarnya memiliki makna yang tidak bisa diwakilkan oleh kata-kata lainnya seperti perusahaan atau organisasi sehingga lebih tepat institusi sebagai padanannya. Tetapi *enterprise* digunakan untuk menggambarkan bagaimana situasi bisnis secara umum dalam suatu entitas institusi, dalam berbagai ukuran, mulai dari ukuran kecil, menengah sampai besar. Disamping itu *enterprise* juga menunjukkan kepada institusi yang telah memanfaatkan teknologi khususnya teknologi informasi dalam menjalankan roda bisnisnya.

Secara konseptual *enterprise* dapat digambarkan sebagai sekumpulan orang-orang yang menetapkan suatu tujuan tertentu, dan memiliki sumber daya (*resource*) tertentu untuk dikelola agar tujuan tersebut dapat tercapai dalam suatu kurun waktu.

Dengan demikian, *enterprise* bisa dikatakan sebagai suatu institusi yang memiliki ruang lingkup bisnis tertentu, memiliki misi yang telah ditetapkan, dan memiliki berbagai sumber daya potensial seperti manusia, teknologi maupun organisasi itu sendiri. Suatu organisasi harus bisa mengkoordinir fungsinya dan menyebarkan informasi ke seluruh bagian organisasi dalam rangka mendukung tercapainya visi dan misi organisasi itu sendiri.



Gambar 2.1: *Enterprise*<sup>[15]</sup>

Suatu organisasi bisa juga mengandung arti korporasi secara keseluruhan, bisa merupakan divisi dari suatu korporasi, bisa berbentuk perusahaan yang berorientasi profit (PT, CV, PD, dll), dan bisa juga berupa institusi pemerintahan (pemerintah pusat, pemerintah daerah provinsi maupun kota, kementerian, rumah sakit, dll), atau bisa juga berupa jaringan organisasi yang terpisah secara geografis tetapi memiliki tujuan tertentu yang sama.

Sebagaimana dijelaskan pada gambar 2.1, pada dasarnya organisasi apapun bisa dipandang sebagai suatu *enterprise* selama memiliki struktur organisasi, ruang lingkup bisnis dan tujuan berupa misi bersama yang telah disepakati. *Enterprise* tidak memandang besar kecilnya organisasi, tetapi atau lebih melihat pada bagaimana proses bisnis organisasi tersebut dijalankan. Sebab, bisa saja unit organisasinya kecil tetapi menjalankan proses bisnis yang kompleks, sebaliknya unit organisasinya besar tetapi proses bisnis yang dijalankannya sederhana.

Selain itu, pengertian *enterprise* bisa memiliki makna yang cukup luas, tidak hanya sekedar berupa organisasi bisnis atau institusi yang cenderung pada orientasi keuntungan secara finansial (*profit oriented*), tetapi bisa juga dalam bentuk institusi pendidikan atau badan amal berupa yayasan atau koperasi yang tidak secara langsung berorientasi mencari keuntungan finansial tetapi lebih pada pelayanan publik.

Disamping itu, mengacu kepada dokumen *The Open Group Architecture Framework* atau TOGAF, bahwa pengertian *enterprise* tidak selalu harus mengacu pada keseluruhan organisasi, tetapi bisa juga difokuskan pada satuan yang lebih kecil seperti divisi, departemen, atau kantor cabang.

## **2.3 Pengertian Arsitektur *Enterprise***

Arsitektur *enterprise* merupakan salah satu cabang disiplin bidang sistem informasi, yang menurut *Chief Information Officer Council* dapat didefinisikan sebagai berikut:

*“A strategic information asset base, which defines the mission, the information necessary to perform the mission, the technologies necessary to the mission, and the transitional processes for implementing new technologies in response to the changing mission needs. An enterprise architecture includes a baseline architecture, target architecture, and a sequencing plan.”*

Sedangkan menurut Zachman definisi Arsitektur *Enterprise* dijelaskan sebagai berikut:

*"that set of descriptive representations (i.e. 'models') that are relevant for describing an Enterprise such that it can be produced to management's requirements (quality) and maintained over the period of its useful life (change)."*

Dengan kata lain, dapat disimpulkan bahwa Arsitektur *Enterprise* merupakan sebuah metode untuk menyusun elemen-elemen *enterprise* bisa berupa sekumpulan model dan hubungan antar elemen *enterprise* yang digunakan dalam merencanakan, mendesain dan merealisasikan suatu struktur *enterprise*, proses bisnis, sistem informasi dan infrastruktur yang terkait di dalamnya. Arsitektur *enterprise* juga bisa menggambarkan tentang bagaimana perencanaan jangka panjang pembangunan sebuah sistem atau sekumpulan sistem.

Pada dasarnya, pengembangan arsitektur *enterprise* memiliki fungsi sebagai berikut

1. Memberi gambaran umum bagaimana hubungan antara tujuan organisasi dan sistem informasi.
2. Bisa menjadi bahan bagi para manajer organisasi untuk mendukung proses pengambilan keputusan investasi dalam penerapan teknologi informasi dan komunikasi.
3. Memanfaatkan teknologi informasi dan komunikasi untuk mendukung aktivitas proses bisnis organisasi sekaligus di berbagai tingkatan manajer, sekaligus dalam rangka penghematan biaya.
4. Meningkatkan kemampuan integrasi data antar bagian atau divisi dalam suatu organisasi seperti pengembangan standar-standar dalam sistem informasi dan mengurangi jumlah antarmuka antar aplikasi.

## **2.4 Peranan Arsitektur *Enterprise***

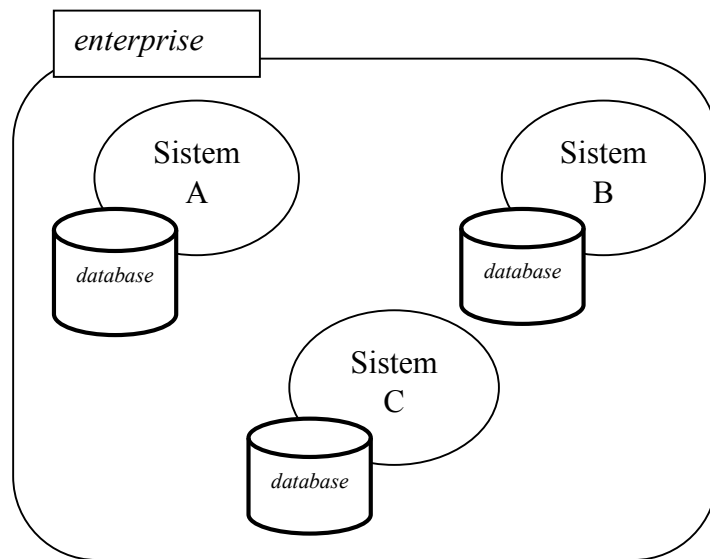
Pengembangan Arsitektur *Enterprise* sangat penting dilakukan oleh setiap *enterprise* baik perusahaan maupun institusi pemerintahan dengan tujuan utamanya adalah dalam rangka bagaimana mewujudkan keselarasan antara penerapan kemajuan teknologi informasi (*software* maupun *hardware*) dan kebutuhan *enterprise* yang sebenarnya dalam mendorong kelancaran proses bisnis yang dilakukannya<sup>[32, 33, 36]</sup>.

Dengan demikian, diharapkan antara kedua unsur pokok ini (teknologi informasi dan bisnis) tidak terjadi gap atau ketimpangan yang besar.



Sebaliknya justru dapat dibangun keselarasan yang saling menunjang secara positif dalam rangka lebih meningkatkan kinerja *enterprise* baik untuk tujuan yang bersifat *profit* maupun *non-profit*.

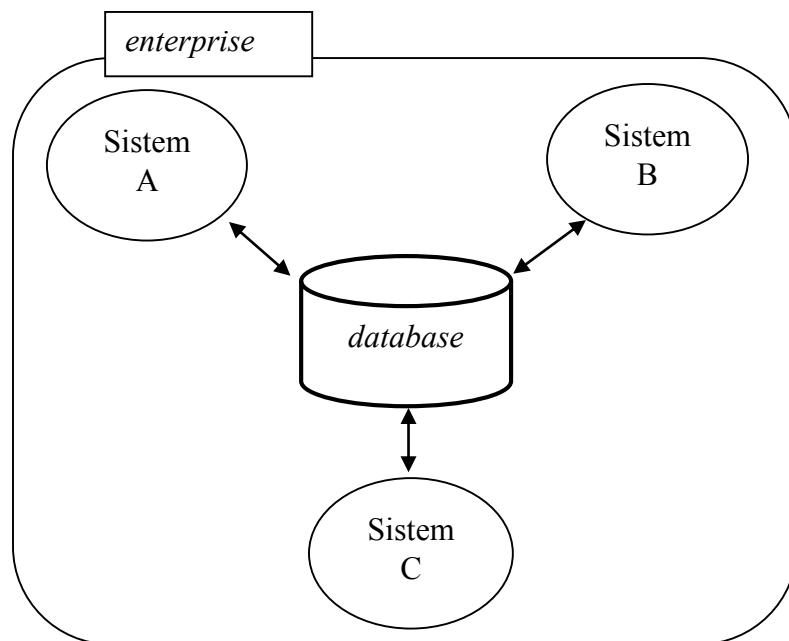
Terutama berkaitan dengan unsur teknologi informasi yang sifatnya selalu mengalami perubahan dengan cepat, sehingga perlu diupayakan supaya perubahan tersebut tidak mengganggu stabilitas kelancaran proses bisnis yang sedang berlangsung, sebaliknya bagaimana harus bisa dimanfaatkan secara tepat dan efisien dalam mendukung kinerja dari bisnis *enterprise*.



Gambar 2.2 Multi Sistem Yang Tidak Terintegrasi

*Output* atau produk yang diharapkan dari pengembangan suatu arsitektur *enterprise*, pada prinsipnya harus bisa menghasilkan *blueprint* (istilah lainnya *masterplan* atau rencana induk) yang akan menjadi panduan atau pedoman yang bermanfaat bagi para manajer dalam merencanakan, mengukur, dan memantau pemanfaatan teknologi informasi dalam mendukung keseluruhan proses bisnis *enterprise*. Disamping itu, *blueprint* juga dapat dijadikan pedoman bagi para pengembang (*developer*) aplikasi sistem informasi untuk membangun berbagai kebutuhan program aplikasi komputer yang sesuai dengan kebutuhan bisnis *enterprise* secara menyeluruh dan terintegrasi. *Blueprint* juga memberi pedoman mana saja sistem yang memiliki skala prioritas untuk dikembangkan lebih awal.

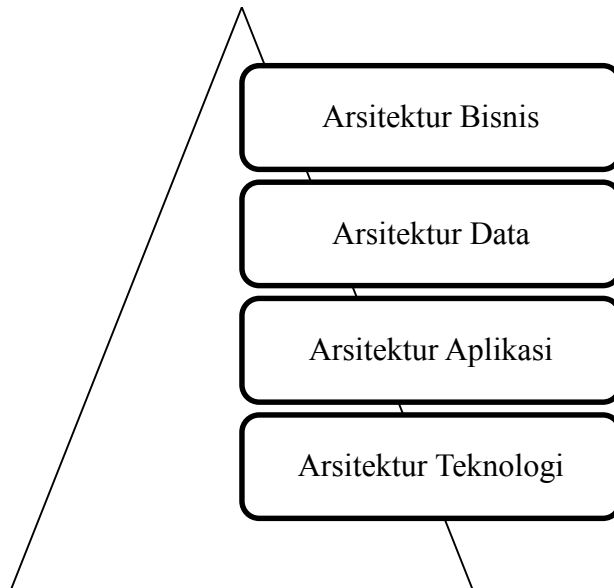
Dengan adanya pedoman berupa *blueprint* sistem informasi *enterprise* diharapkan pengembangan sistem informasi tidak bersifat sporadis sehingga bisa terjadinya kotak-kotak aplikasi sistem informasi yang terpisah satu sama lain seperti pada gambar 2.2, walaupun sistem-sistem aplikasi yang telah dibuat tersebut lengkap. Karena tidak terjadinya *information sharing* antar sistem yang ada, maka akan mengakibatkan terjadinya duplikasi data dan informasi yang diolah tidak terintegrasi, disamping akan terjadinya pembengkakan biaya pengembangan yang tidak efisien. Oleh karena itu, *blueprint* sistem informasi *enterprise* merupakan suatu keharusan dan memiliki posisi sangat penting dalam mengarahkan kegiatan pengembangan sistem informasi yang dibutuhkan oleh suatu *enterprise* dengan memegang prinsip *comprehensive* (menyeluruh) dan *integrated* (terpadu), sehingga hasil dari pengembangan arsitektur *enterprise* idealnya sebagaimana yang ditunjukkan seperti pada gambar 2.3.



Gambar 2.3 Sistem Informasi *Enterprise* Terintegrasi

Sistem informasi *enterprise* harus mencakup kebutuhan data secara keseluruhan sistem dalam suatu *enterprise*, sehingga secara logis seluruh data harus disimpan dalam suatu database secara transparan dan bisa diakses atau digunakan oleh bermacam-macam sistem informasi yang memerlukan data tersebut.

Produk yang dihasilkan dari penyusunan arsitektur *enterprise* dengan menggunakan metodologi apapun, pada dasarnya minimal harus bisa menghasilkan empat buah komponen dasar arsitektur *enterprise* seperti yang ditunjukkan oleh gambar 2.4, yaitu sebagai berikut.



Gambar 2.4 Komponen Utama Arsitektur *Enterprise*<sup>[35]</sup>

Komponen bisnis arsitektur *enterprise* (visi & misi, fungsi bisnis, alur informasi, lingkungan sistem)

1. Arsitektur Bisnis

Arsitektur yang paling utama dan menjadi sumber rujukan utama dalam penyusunan komponen arsitektur *enterprise* lainnya. Pemahaman yang benar tentang arsitektur bisnis menjadi kunci sukses pengembangan arsitektur *enterprise*. Arsitektur ini menggambarkan bagaimana kondisi kegiatan bisnis yang sedang berjalan dan mungkin akan dilakukan perbaikan (*business process reengineering*) untuk waktu yang akan datang.

Sedangkan, komponen teknis arsitektur *enterprise* (*software, hardware, communication*) terdiri dari:

2. **Arsitektur Data**  
Menggambarkan kebutuhan data secara global yang meliputi seluruh bagian di dalam *enterprise* yang harus disediakan secara terintegrasi, dalam rangka mendukung proses bisnis yang dijalankan.
3. **Arsitektur Aplikasi**  
Menunjukkan gambaran kebutuhan berbagai aplikasi utama apa saja yang diperlukan dalam rangka memfasilitasi kegiatan bisnis *enterprise* dalam mengolah data secara menyeluruh dan terintegrasi sehingga memberikan manfaat bagi kegiatan bisnis *enterprise* secara keseluruhan.
4. **Arsitektur Teknologi**  
Memberikan gambaran bagaimana pemanfaatan kemajuan teknologi informasi khususnya yang berkaitan dengan infrastruktur jaringan komputer dalam mendukung ketiga arsitektur yang sudah didefinisikan sebelumnya. Kebutuhan terhadap teknologi jaringan komputer merupakan hal pokok karena akan menjadi *backbone* atau tulang punggung integrasi keseluruhan sistem informasi yang mendukung *enterprise*.

Produk-produk arsitektur *enterprise* yang dihasilkan oleh setiap penerapan metodologi bisa jadi berbeda. Hal ini sangat tergantung pada tingkat rincian tahapan yang dilakukan dalam suatu metodologi dilihat dari berbagai aspek dan fungsi, sehingga menyebabkan perbedaan kelengkapan jenis produk arsitektur *enterprise*.

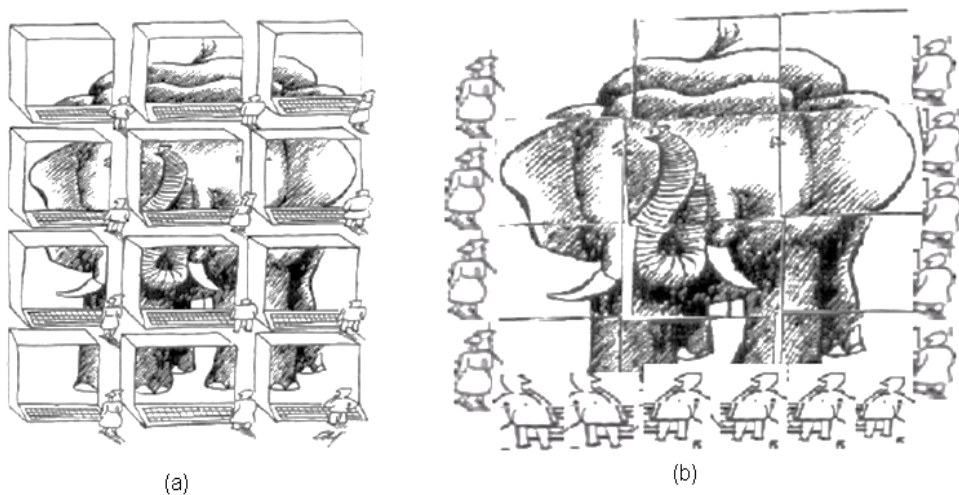
## **2.5 Sistem Terpadu**

Dalam suatu sistem informasi berskala *enterprise*, berbagai sistem informasi yang dibutuhkan memiliki keterkaitan yang kuat sama lain. Dengan demikian, masalah integrasi sistem harus dilakukan supaya data secara keseluruhan dapat diolah untuk menghasilkan suatu informasi yang lengkap, dan tidak terjadi duplikasi dan berkualitas yang sangat diperlukan oleh para pemegang kebijakan dalam suatu *enterprise* dalam proses pengambilan keputusan.

Sedangkan dari aspek aplikasi sistem yang terintegrasi dibutuhkan dalam membangun aplikasi yang ringan dan lebih mudah dalam proses pemeliharaan sistem, karena aplikasi sistem akan disusun dari berbagai modul yang mana

ada beberapa modul yang sifatnya umum akan digunakan oleh aplikasi sistem yang berbeda.

Dengan demikian, sesuai dengan konsep pengembangan sistem berbasis objek dimana modul yang berupa komponen program sedapat mungkin harus bersifat *reusable* sehingga memungkinkan bahwa suatu aplikasi sistem bisa dibangun dengan lebih cepat dan lebih *reliable* dengan menggunakan komponen-komponen yang tersedia dan sudah teruji pada pengembangan aplikasi sistem sebelumnya. Manfaat lain dari penerapan *reusable* adalah untuk memudahkan proses pemeliharaan *software* apabila diperlukan perubahan-perubahan yang diperlukan.



Gambar 2.5 Analogi Sistem Terpadu<sup>[16]</sup>

Gambaran kebutuhan sistem informasi yang terintegrasi seperti yang ditunjukkan pada gambar 2.5. Seperti pada prinsip *puzzle*, bahwa sistem *enterprise* harus dibagi-bagi menjadi bagian-bagian sistem yang lebih kecil (gambar 2.5.a) bersifat mandiri, tetapi apabila digabungkan bagian-bagian sistem tersebut maka akan dapat menghasilkan suatu sistem yang besar berskala *enterprise* (gambar 2.5.b) dimana antara satu sistem dengan sistem yang lainnya sifatnya saling mendukung. Analoginya bahwa bagaimana gambar seekor gajah (sebagai sistem informasi *enterprise*) dipecah-pecah menjadi beberapa potongan gambar kecil (*puzzle*) yang apabila digabungkan kembali, pada akhirnya akan membentuk gambar seekor gajah dengan ukuran yang benar dan utuh. Apabila ada bagian *puzzle* yang hilang maka timbul masalah yang berakibat tidak menampilkan seekor gajah utuh. Artinya sistem

informasi *enterprise* untuk mendukung proses bisnis yang kompleks harus dianalisis dan dibagi-bagi menjadi beberapa sistem informasi yang tetap harus terintegrasi satu sama lain sehingga bisa menghasilkan informasi yang benar dan utuh. Tidak ada satu bagian sistem informasi pun yang terpisah atau yang hanya memiliki data yang terpisah, atau ada bagian-bagian (beberapa sistem informasi) yang duplikasi, sehingga pada akhirnya bisa mengakibatkan tidak membentuk informasi yang sesungguhnya.

Permasalahannya adalah bagaimana memilah-milah sistem informasi *enterprise* yang harus mendukung proses bisnis yang kompleks menjadi beberapa sistem informasi berskala kecil tetapi harus tetap bisa diintegrasikan? Oleh karena itulah, maka diperlukan pengembangan arsitektur *enterprise* agar diketahui mana saja sistem informasi yang benar-benar diperlukan dan bagaimana caranya supaya dapat dikembangkan secara bertahap berdasarkan skala prioritas tertentu.

## 2.6 Faktor Keberhasilan

Beberapa prinsip dasar yang perlu diperhatikan dalam mensukseskan dalam pengembangan suatu arsitektur *enterprise* yaitu sebagai berikut<sup>[31]</sup>.

1. Prinsip *Enterprise*

Pengembangan arsitektur *enterprise* yang dilakukan diharapkan mendukung seluruh bagian organisasi dan mencakup seluruh unit-unit organisasi di seluruh level manajerial dalam menjalankan rangkaian proses bisnis secara menyeluruh dari proses hulu sampai dengan proses hilir.

2. Prinsip Teknologi Informasi

Mengarahkan secara konsisten pemanfaatan teknologi informasi pada seluruh bagian organisasi, termasuk mencakup unit-unit organisasi yang akan menggunakannya.

3. Prinsip Arsitektur

Merancang arsitektur sistem informasi berdasarkan kebutuhan proses bisnis yang dijalankan oleh suatu *enterprise* yang meliputi seluruh area bisnisnya, dan bagaimana mengimplementasikan sampai sistem informasi yang dihasilkan benar-benar mendukung kegiatan *enterprise*.

## 2.7 Keuntungan Pengembangan Arsitektur *Enterprise*

Pelaksanaan pengembangan arsitektur *enterprise* akan memberikan beberapa keuntungan dalam mengelola *enterprise* itu sendiri<sup>[31, 32]</sup>, diantaranya sebagai berikut

1. Pandangan arsitektural *enterprise*  
Alat bantu untuk menggambarkan kompleksitas dari suatu sistem informasi yang mencakup keseluruhan proses bisnis yang dijalankan oleh suatu *enterprise*.
2. Fokus pada penggunaan strategi pemanfaatan teknologi informasi  
Bagaimana memanfaatkan kemajuan teknologi informasi secara optimal dan menyelaraskannya dengan kebutuhan bisnis dalam mendukung kinerja *enterprise*.
3. Sarana komunikasi antar unit teknologi informasi dan unit lain  
Mendukung keselarasan antara unit teknologi informasi sebagai pengelola teknologi informasi dan unit-unit lain organisasi, sehingga penerapan teknologi informasi dapat dioptimalkan sepenuhnya dalam proses pengambilan keputusan.
4. Panduan bagi *enterprise* dalam pengelolaan investasi teknologi informasi  
Meningkatkan efisiensi penggunaan teknologi informasi yang membutuhkan dana investasi yang besar, bisa secara maksimal digunakan untuk mendukung peningkatan kinerja proses bisnis secara *enterprise*.

Melihat keuntungan yang bisa diperoleh dari pengembangan arsitektur *enterprise* tersebut, maka sudah sewajarnya setiap organisasi untuk menghasilkan pedoman yang jelas tentang rencana pengembangan sistem informasi di masa datang. Terutama di era persaingan global dimana persaingan akan dimenangkan oleh organisasi yang didukung oleh berbagai sistem informasi yang sesuai.

## 2.8 Pengembangan Arsitektur *Enterprise*

Pada prinsipnya untuk mengembangkan suatu arsitektur *enterprise* dalam suatu organisasi dibagi menjadi tiga tahapan utama seperti dijelaskan pada gambar 2.6.

1. Inisialisasi

Tahap inisiasi untuk melakukan berbagai persiapan seperti menentukan tim pengembang, menentukan visi dan misi, mengalokasikan biaya dan waktu, serta yang penting adalah mendapatkan komitmen dari pimpinan puncak untuk mengembangkan sampai dengan mengimplementasikan sistem informasi *enterprise* secara terencana.

2. Pengembangan *Roadmap*

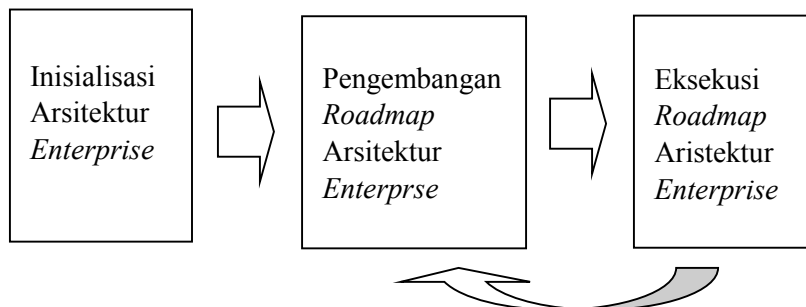
Mengembangkan berbagai arsitektur *enterprise* terutama menganalisis arsitektur bisnis dari proses bisnis yang dijalankan oleh organisasinya, dilanjutkan dengan pembuatan empat arsitektur utama yaitu arsitektur data, arsitektur aplikasi, dan arsitektur teknologi.

3. Eksekusi *Roadmap*

Pemilihan metodologi tertentu dan mengembangkan secara bertahap berdasarkan skala prioritas tertentu seluruh aplikasi dari sistem informasi yang telah teridentifikasi sampai bisa diimplementasikan.

Ketiga tahapan dasar pengembangan arsitektur *enterprise*, dikembangkan dengan pendekatan yang berbeda sehingga menghasilkan berbagai metodologi yang berbeda dan menghasilkan dokumen-dokumen yang berbeda pula. Setiap metodologi memiliki keunikan tersendiri untuk digunakan pada objek *enterprise* yang sesuai.

Pemilihan metodologi pengembangan arsitektur *enterprise* mana yang layak digunakan, hal ini sangat tergantung pada target dari arsitektur *enterprise* yang diinginkan, dimana arsitektur *enterprise* bisa dibuat yang bersifat konseptual, atau bersifat implementatif<sup>[8, 9]</sup>.



Gambar 2.6 Siklus Hidup Arsitektur *Enterprise*

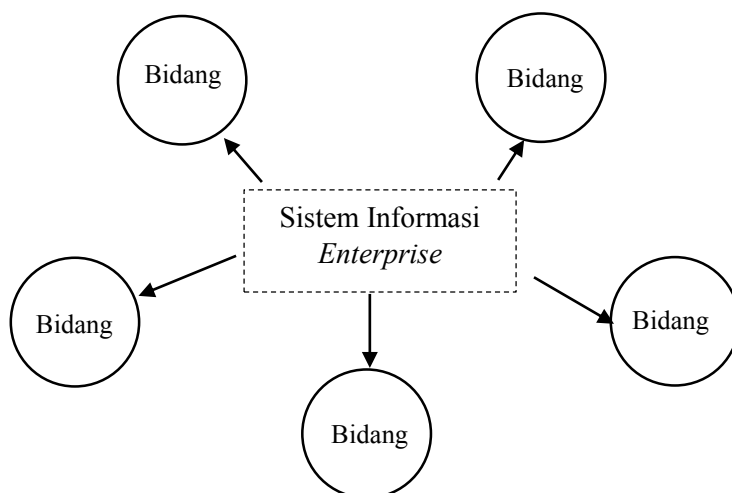


## 2.9 Pendekatan Implementasi

Produk *software* sistem informasi *enterprise* yang dihasilkan sebagai hasil dari pengembangan arsitektur *enterprise*, selanjutnya bisa diimplementasikan dengan menggunakan beberapa pendekatan. Pendekatan yang dipakai, haruslah disesuaikan terhadap kondisi masing-masing *enterprise* yang akan menjalankannya. Kondisi *enterprise* yang dimaksud tidak hanya dilihat dari ukuran atau kompleksitas *enterprise*, tetapi juga dari kondisi sumber dayanya maupun sumber daya manusianya. Pendekatan apapun yang dijalankan terlebih dahulu harus diawali dengan pelatihan penggunaan sistem baik aspek teknis maupun non-teknis<sup>[14]</sup>.

### 1. *Big Bang*

Cara penerapan sistem informasi *enterprise* secara serentak dan menyeluruh di setiap bidang yang ada dalam suatu *enterprise*. Cara seperti ini hanya cocok dan bisa diterapkan pada lingkungan *enterprise* yang sudah benar-benar siap dari sumber daya yang tersedia seperti infrastruktur, manusia yang cukup ilmu dan motivasi tinggi, kebijakan pendukung, dll. Jika sumber daya nya diperkirakan dalam kondisi tidak memungkinkan atau masih kurang memadai untuk mengoperasikan sistem informasi, maka cara seperti ini sebaiknya tidak dilakukan, karena memiliki risiko tinggi yaitu kegagalan yang paling besar jika dibandingkan dengan cara penerapan yang lainnya. Ilustrasi dari penerapan *Big Bang* dijelaskan seperti pada gambar 2.7.

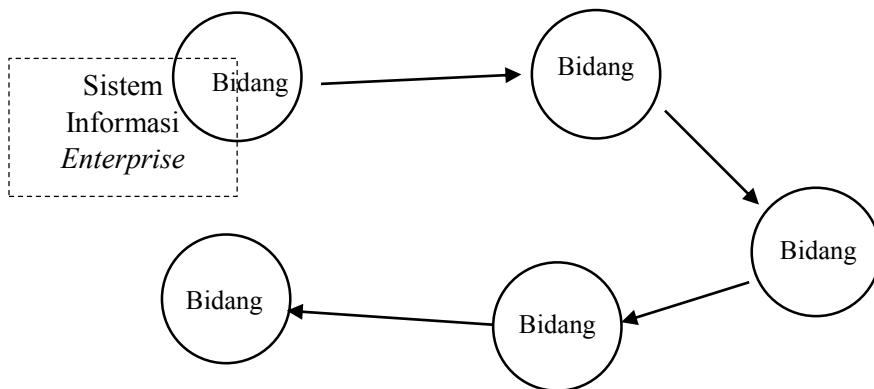


Gambar 2.7 *Big Bang*

## 2. *Small Bang (Pilot Approach)*

Cara penerapan sistem informasi *enterprise* dengan terlebih dahulu analisis kondisi *enterprise*, kemudian menentukan skala prioritas dari keseluruhan *enterprise*, kemudian menentukan salah satu bagian dari *enterprise* yang sumber dayanya diperkirakan sudah siap menerapkan aplikasi sistem informasi sebagai *pilot project*.

Dengan demikian, model percontohan seperti ini salah satu bidang bisa dijadikan sebagai *role model* sehingga diketahui *know-how* nya berdasarkan pengalaman dari penerapan sistem informasi *enterprise* di bidang yang terpilih.



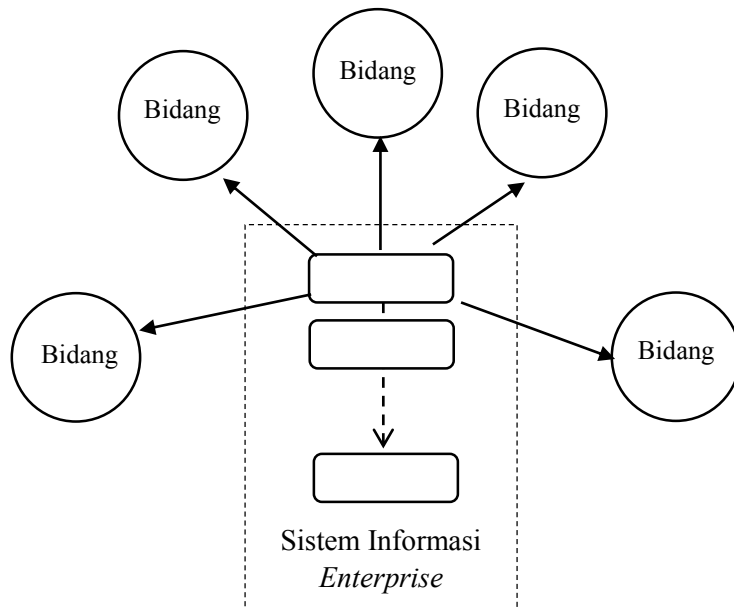
Gambar 2.8 *Small Bang*

Dengan demikian, dalam cara seperti ini risiko kegagalan penerapan sistem informasi *enterprise* bisa diantisipasi, dan pengalaman pengelolaan di *role model* bisa jadi pengetahuan dasar dan diharapkan akan mempermudah penerapannya di bidang-bidang lainnya. Ilustrasi dari penerapan *small bang* dijelaskan seperti pada gambar 2.8.

## 3. *Step-by-step (Phased Approach)*

Cara Penerapan ini dilakukan dengan menyiapkan hanya beberapa aplikasi sistem informasi tertentu saja di seluruh atau sebagian dari *enterprise*. Kemudian, dalam jangka waktu tertentu tergantung kondisi *enterprise* masing-masing, dilakukan evaluasi, dan bisa dilanjutkan secara bertahap menggunakan sistem informasi lainnya yang terkait.

Pendekatan rasional untuk organisasi yang tidak ingin mengambil risiko tinggi terhadap kegagalan penerapan sistem informasinya. Pendekatan seperti ini memerlukan waktu yang relatif lebih lama dibandingkan dengan pendekatan lainnya. Ilustrasi dari penerapan *step-by-step* dijelaskan seperti pada gambar 2.9.



Gambar 2.9 *Step-by-step*

# Bab 3

## ***ENTERPRISE ARCHITECTURE PLANNING (EAP)***

---

### **3.1 Istilah EAP**

Steven H. Spewak mendefinisikan istilah EAP dalam pengembangan arsitektur *enterprise* sebagai berikut:

*“Enterprise Architecture Planning is the process of defining architectures for the use of information in support of the business and the plan for implementng those architectures.”*

Berdasarkan pengertian tersebut, EAP merupakan usaha-usaha untuk menentukan kebutuhan akan berbagai arsitektur (data, aplikasi dan teknologi) yang dapat mendukung pelaksanaan bisnis suatu *enterprise*, serta menentukan perencanaan dari implementasi arsitektur *enterprise*.

Perencanaan arsitektur *enterprise* merupakan kebutuhan yang paling dasar dari suatu *enterprise* dalam menjalankan roda bisnisnya dengan berbasis komputer. Penyusunan EAP yang baik merupakan pokok keberhasilan pengembangan sistem informasi *enterprise*, karena setiap data, aplikasi dan teknologi yang dibutuhkan sudah direncanakan dan dipersiapkan implementasinya, sehingga seluruh kebutuhan sistem informasi bisa dikembangkan secara terpadu (*integrated information system*).

### **3.2 Masterplan Arsitektur Enterprise**

*Masterplan* sering disebut juga dengan *blueprint*. Kedua istilah ini merupakan istilah umum yang terkait dalam pengembangan arsitektur *enterprise*. *Masterplan* atau *blueprint* merupakan dokumen yang sangat penting sebelum memulai pengembangan aplikasi sistem informasi yang diperlukan oleh *enterprise*.

*Masterplan* memiliki fungsi sebagai pedoman dalam melakukan berbagai pengembangan sistem informasi yang diperlukan oleh suatu *enterprise* dalam menjalankan proses bisnisnya dengan memanfaatkan teknologi informasi dari mulai tahapan analisis, perancangan sampai dengan implementasi dan bahkan pemeliharaan sistem secara berkesinambungan.

*Masterplan* juga merupakan dokumen induk yang berisi berbagai arahan yang menjelaskan tentang apa saja dan bagaimana sistem informasi *enterprise* harus dibangun secara sistematis, berdasarkan pada metodologi yang digunakan dalam pembuatan *masterplan*-nya. Oleh sebab itu, suatu *masterplan* harus jelas, terarah dan terukur pencapaiannya.

### 3.3 Metodologi EAP

EAP merupakan metodologi untuk membuat perencanaan arsitektur *enterprise* yang pertama kali diperkenalkan oleh Steven H. Spewak pada sekitar tahun 1990-an. EAP fokus pada kata *planning* dari suatu arsitektur *enterprise*, hasil dari penggunaan metodologi ini jika dipetakan ke dalam *framework* Zachman hanya akan mencakup enam sel (seperti pada tabel 3.1), dengan kata lain hanya mencakup dua perspektif dan tiga kolom saja dari matriks Zachman.

EAP hanya fokus pada pembuatan empat komponen utama arsitektur *enterprise* yang meliputi perspektif *planner* dan *owner*, serta meliputi kolom data, fungsi, dan teknologi. Sebenarnya kalau dianalisis lebih lanjut khusus untuk arsitektur aplikasi sebenarnya bisa mencakup perspektif *designer*.

Tabel 3.1 *Framework* EAP<sup>[20, 32]</sup>

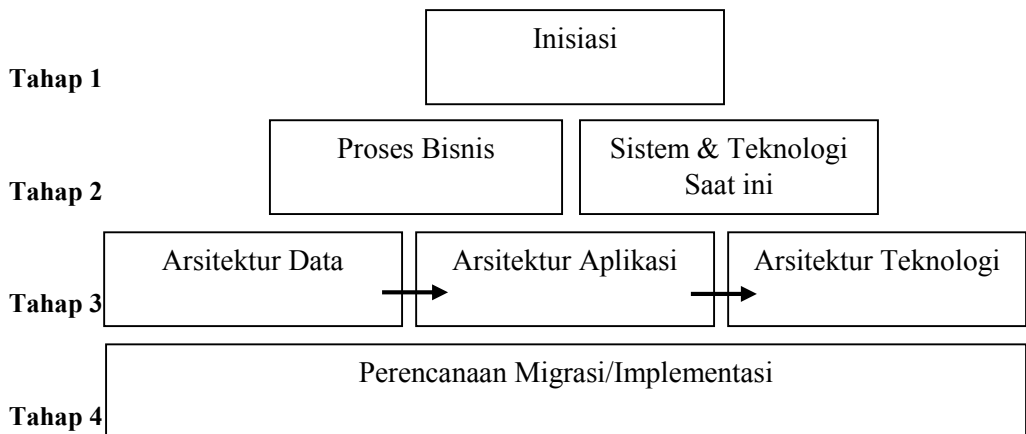
	<b><i>Data (what)</i></b>	<b><i>Function (how)</i></b>	<b><i>Network (where)</i></b>
<b><i>Planner</i></b>	Daftar entitas penting dalam <i>enterprise</i>	Daftar fungsi bisnis penting	Daftar lokasi operasional
<b><i>Owner</i></b>	Hubungan antar entitas bisnis (menggunakan <i>Entity Relationship Diagram</i> )	Dekomposisi fungsi dan proses bisnis menggunakan DFD, BPMN, dll	Jaringan logistik (node & link)  Komunikasi antar lokasi bisnis

Pada prinsipnya penyusunan arsitektur *enterprise* dengan menggunakan EAP sangat menekankan hal-hal sebagai berikut.

1. Pendefinisian model bisnis yang stabil
2. Penentuan data *dependency* sebelum mengimplementasikan sistemnya
3. Urutan implementasi sistem berdasarkan pada data *dependency*

Seperti yang ditunjukkan pada gambar 3.1, metodologi EAP memiliki tahapan yang sederhana dan bersifat linier karena pada prinsipnya hanya membahas tentang masalah pokok empat jenis komponen arsitektur *enterprise*, dengan mempertimbangkan kondisi pada sistem atau teknologi informasi yang sudah atau sedang digunakan.

Sesuai dengan namanya EAP ditujukan untuk tahapan awal pengembangan *masterplan* arsitektur *enterprise* yaitu perencanaan empat arsitektur dasar dari suatu sistem informasi *enterprise*. Dengan demikian semua artifak (dokumen, gambar, diagram, keterangan, dll) yang dihasilkan oleh EAP ini pada umumnya masih berupa konsep yang akan dijadikan dasar untuk pengembangan sistem informasi *enterprise*.



Gambar 3.1 Tahapan EAP<sup>[31]</sup>

EAP secara umum dibagi ke dalam empat tahapan atau *level* yaitu sebagai berikut.

## **Tahap 1: *Project initiation***

Tahapan awal yang merupakan landasan bagi pengerjaan berikutnya dari pengembangan arsitektur *enterprise*. Pada tahapan ini merencanakan beberapa persiapan proyek seperti penentuan ruang lingkup, pembentukan tim proyek, kebutuhan sumber daya manusia, penentuan biaya awal, penentuan alat bantu, penyusunan jadwal, penentuan metode yang akan digunakan dalam tiap tahapan, penetapan alat bantu *software*, dan analisis penting lainnya yang diperlukan untuk kelancaran pekerjaan.

Sebelum memulai pelaksanaan pengembangan arsitektur *enterprise*, beberapa langkah penting yang harus dipersiapkan adalah sebagai berikut:

1. Menentukan ruang lingkup dan objek
2. Membuat visi pengembangan arsitektur *enterprise*
3. Menentukan metode perencanaan
4. Menganalisis sumber daya teknologi informasi
5. Menyusun tim pengembang
6. Mempersiapkan perencanaan
7. Mendapatkan konfirmasi komitmen dan pendanaan

## **Tahap 2: *Knowledge base of the business processes and required information***

Tahapan untuk menganalisis kondisi saat ini yang sedang berlangsung pada objek yang akan diamati. Analisis fokus pada bagaimana memahami proses bisnis yang sedang berjalan, kemudian divisualisasikan dengan menggunakan diagram proses bisnis supaya tingkat kompleksitas proses bisnis yang berlangsung dalam suatu *enterprise* bisa tergambarkan secara jelas. Penggambaran proses bisnis merupakan hal yang paling pokok dan benar-benar yang valid dan disepakati oleh anggota pengembang, karena kesalahan pemahaman dalam proses bisnis akan berdampak terhadap komponen arsitektur *enterprise* lainnya. Perubahan proses bisnis bisa terjadi kapan saja, dan hal ini harus bisa diatasi dalam penggambaran proses bisnis yang benar, agar tidak muncul kerancuan dalam pembuatan komponen arsitektur lainnya yang terkait.

Disamping mengidentifikasi proses bisnis, perlu juga melakukan proses inventarisir berbagai peralatan yang berkaitan dengan pemanfaatan aplikasi sistem dan teknologi informasi yang sedang digunakan, dilihat dari sisi *software* dan *hardware*. Perlu juga untuk

membuat inventaris sistem dalam bentuk *Information Resource Catalog (IRC)* yang juga disebut ensiklopedia sistem atau *inventory* sistem. Tidak ada format khusus yang ditetapkan untuk pembuatan IRC, tetapi pada intinya dokumen yang menjelaskan tentang penggunaan sistem informasi dan *resource* yang terkait dengannya.

Antara analisis proses bisnis dan inventarisir sistem dan teknologi saat ini, tidak ada ketentuan mana yang harus diprioritaskan. Dengan demikian sifatnya fleksibel disesuaikan dengan kondisi dari objek pengembangan masing-masing, bisa dilakukan yang mana dulu atau jika memungkinkan bisa juga dilakukan secara bersamaan oleh dua tim yang berbeda.

### **Tahap 3: *The Future architecture***

Kegiatan utama dari penyusunan arsitektur *enterprise* yang ditargetkan untuk dicapai di masa yang akan datang. Pada tahapan ini, EAP sangat menekankan bahwa penyusunan arsitektur *enterprise* harus memiliki skala prioritas tertentu, artinya harus dimulai dari perancangan arsitektur data, perancangan arsitektur aplikasi, dan perancangan arsitektur teknologi secara berturut-turut. Namun demikian sebenarnya antara penyusunan arsitektur data dan arsitektur aplikasi masih bisa diperdebatkan mana yang harus didahulukan berdasarkan referensi metodologi yang lain seperti TOGAF-ADM.

Penyusunan arsitektur data dan aplikasi pada tahapan ini, sebenarnya bisa saja dilakukan oleh tim yang berbeda secara bersamaan, karena dalam perencanaan arsitektur *enterprise* antara data dan aplikasi masih dalam tataran konseptual.

Supaya lebih sistematis pembuatan komponen arsitektur pada tahapan ini bisa dilakukan dengan melalui dua tahapan berdasarkan dua perspektif berbeda yang dimulai dari perspektif *planner* dan dilanjutkan dengan perspektif *owner*.

### **Tahap 4: *Implementation***

Mendefinisikan rencana kegiatan implementasi sistem-sistem yang berhasil diidentifikasi dan dibutuhkan oleh *enterprise*, serta bagaimana rencana migrasi dari sistem lama ke sistem baru. Dalam tahapan ini



termasuk juga perkiraan dari segi waktu, sumber daya, biaya yang dibutuhkan.

Pada tahapan ini sebaiknya juga direncanakan hal-hal lain yang penting dalam pengembangan arsitektur *enterprise* yaitu:

1. *Software development life cycle* / SDLC  
Setiap sistem informasi yang teridentifikasi akan memiliki karakteristik yang berbeda, sehingga perlu dipertimbangkan langkah-langkah implementasinya dengan menggunakan SDLC yang mana yang tepat supaya efektif dan efisien dalam pembuatannya.
2. Alat bantu (*Tool*)  
*Tool* berupa aplikasi *software* yang membantu membuat tingkat efektifitas dan efisiensi pengembangan sistem informasi menjadi lebih baik.
3. Sumber daya manusia  
Tim pengembang tersusun dari berbagai keahlian dan jumlahnya yang diperlukan sangat tergantung pada ketersediaan dana dan waktu yang dialokasikan.

Langkah-langkah yang perlu dipertimbangkan dalam menyusun perencanaan migrasi atau implementasi.

1. Menetapkan urutan atau skala prioritas dari setiap aplikasi sistem informasi yang telah didefinisikan dan akan dibuat. Untuk menunjukkan skala prioritas aplikasi yang akan dibangun, bisa menggunakan portofolio aplikasi yaitu berupa tabel kategori aplikasi sistem informasi yang bersifat strategis, berpotensi tinggi, operasional kunci, dan pendukung.
2. Mengukur usaha, kemampuan sumber daya (*resource*) di bidang teknologi informasi yang dimiliki, dan merancang jadwal implementasi sistem.
3. Menentukan faktor-faktor kesuksesan, dan menghasilkan rekomendasi tentang segala hal yang berkaitan dengan pengembangan sistem informasi *enterprise*.

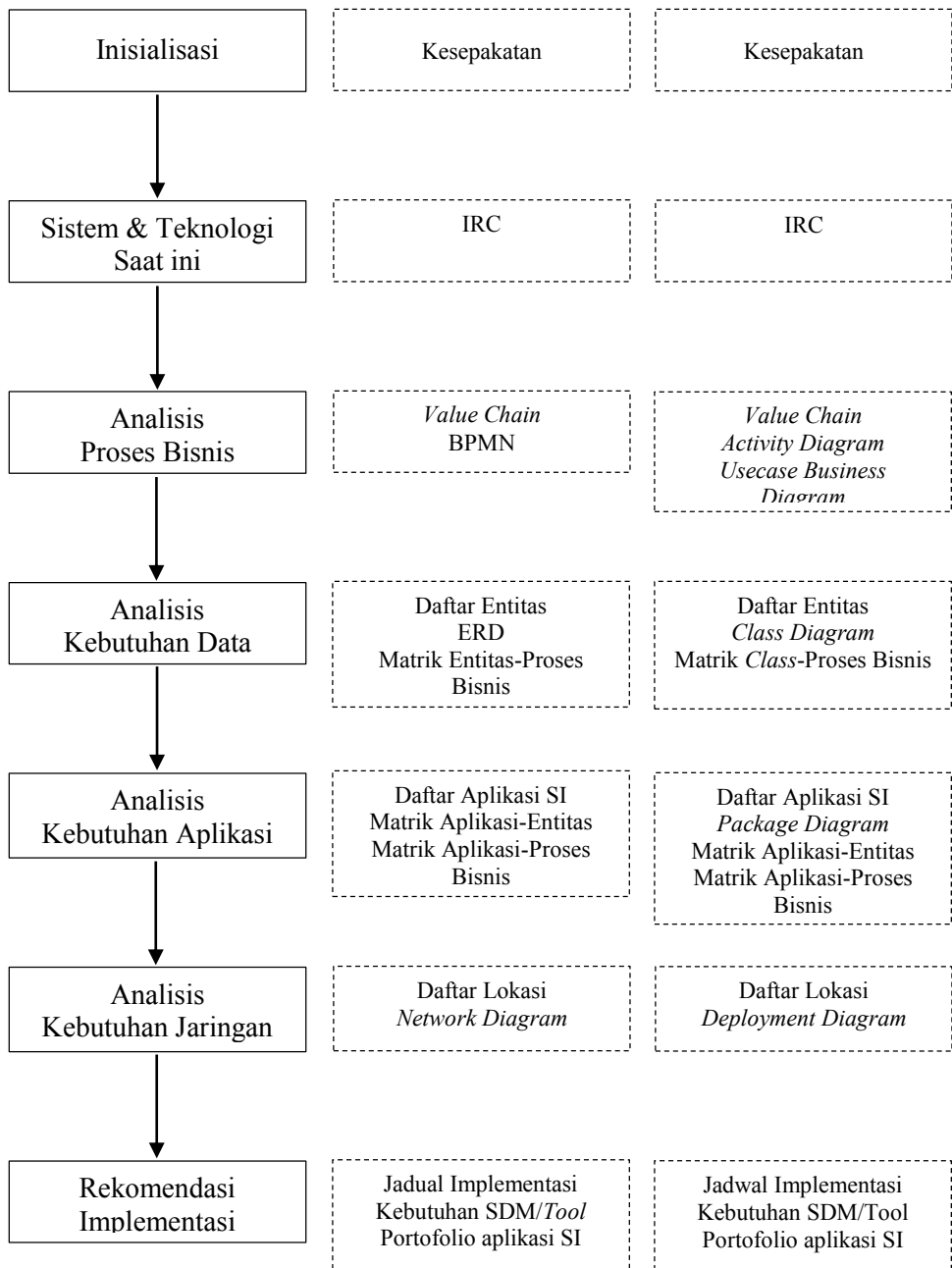
EAP pada awalnya merupakan metodologi yang menggunakan pendekatan terstruktur sehingga artifak yang dihasilkanpun adalah *Data Flow Diagram* (DFD), *Entity Relationship Diagram* (ERD), *flow chart*, dll. Dengan demikian seolah-olah merupakan metodologi yang sifatnya konvensional, sehingga kelihatannya kurang layak diterapkan untuk pengembangan arsitektur *enterprise* saat ini.

Seiring dengan perkembangan teknologi pengembangan sistem termasuk *software*, pendekatan objek mulai banyak digunakan dalam pengembangan *software*. Hal ini berdampak pula terhadap pengembangan arsitektur *enterprise*, dengan munculnya metodologi lain yang mengadopsi pendekatan objek diantaranya *The Open Group Architecture Framework* (TOGAF)-ADM dan *Enterprise Unified Process* (EUP).

Tabel 3.2 *Framework* EAP dengan Pendekaan Objek

	<b><i>Data (what)</i></b>	<b><i>Function (how)</i></b>	<b><i>Network (where)</i></b>
<b><i>Planner</i></b>	Daftar kandidat <i>class</i> bisnis <i>enterprise</i>	Daftar <i>Usecase</i> bisnis dalam menjalankan <i>enterprise</i>	Daftar lokasi operasional
<b><i>Owner</i></b>	Hubungan antar <i>class</i> bisnis (menggunakan <i>Class Diagram</i> )	Dekomposisi fungsi dan proses bisnis menggunakan BPMN, <i>Activity Diagram</i>	Jaringan logistik (node & link) menggunakan <i>Deployment Diagram</i>

Oleh sebab itu, dalam berbagai penelitian arsitektur *enterprise* EAP mengalami perubahan sesuai dengan perkembangan teknologi informasi, yaitu pengembangan arsitektur *enterprise* dengan metodologi EAP mulai menggunakan pendekatan objek, sehingga artifak-artifak yang dihasilkannya juga menyesuaikan sebagaimana yang biasa dihasilkan melalui pendekatan objek seperti yang ditunjukkan pada tabel 3.2.



Gambar 3.2 EAP dengan Pendekatan Terstruktur dan Objek

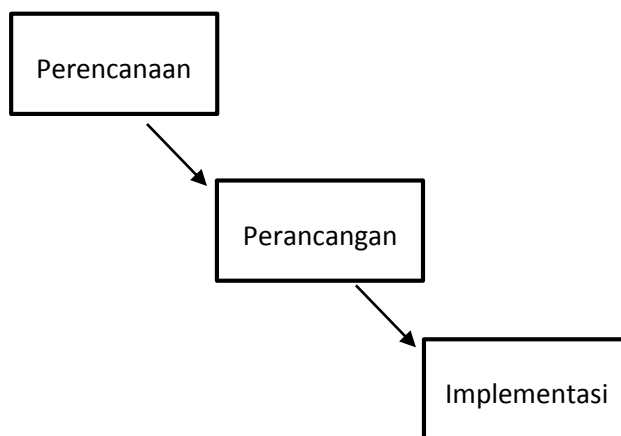
Adapun artifak-artifak yang dihasilkan umumnya menggunakan alat bantu *Unified Modeling Language (UML)* yang khusus digunakan dalam pendekatan

objek, seperti *class diagram* digunakan untuk menggantikan ERD, *package diagram* untuk penjelasan kebutuhan aplikasinya. Sedangkan untuk penjelasan proses bisnis selain *activity diagram*, bisa juga digunakan diagram lain yang lebih lengkap seperti *Business Process Modeling Notation* (BPMN). BPMN bukan merupakan bagian dari UML tetapi bisa digunakan untuk menunjang UML.

Pengembangan arsitektur *enterprise* menggunakan metodologi EAP dan dengan dua pendekatan yang berbeda (terstruktur dan objek), perbedaannya bisa dilihat seperti pada gambar 3.2. Beberapa artifak yang dihasilkan ada yang sama dan ada juga yang berbeda.

### 3.4 Tipe Pengembangan Arsitektur

Perencanaan arsitektur *enterprise* merupakan langkah awal dan terpenting dalam pengembangan keseluruhan dari suatu arsitektur *enterprise* dalam mewujudkan suatu sistem informasi *enterprise* yang terintegrasi. Oleh sebab itu, perlu dilakukan tahapan-tahapan selanjutnya dalam rangka mendefinisikan dan menyusun komponen-komponen lainnya dari suatu arsitektur yang lebih lengkap dan lebih rinci, sehingga akhirnya dihasilkan artifak/dokumen yang bisa dijadikan pedoman dalam implementasi sistem informasi *enterprise*.



Gambar 3.3 Tahapan Pengembangan Arsitektur *Enterprise*

Tahapan keseluruhan dari suatu pengembangan arsitektur *enterprise* ditunjukkan seperti pada gambar 3.3. Tahapan sebelumnya merupakan masukan bagi tahapan berikutnya. Dengan demikian tahapan berikutnya

merupakan keterangan yang lebih rinci atau lebih implementatif dari tahapan sebelumnya.

Adapun pendekatan yang bisa digunakan dalam pengembangan arsitektur *enterprise* bisa secara terstruktur maupun secara objek. Tetapi berdasarkan perkembangan teknologi informasi khususnya teknologi pengembangan software, pendekatan objek merupakan pendekatan yang lebih efektif terutama apabila sampai dengan pembuatan aplikasinya. Walaupun kalau dilihat dari sisi penerapan *database*, pembuatan arsitektur data berbasis objek masih memiliki banyak kendala karena *software object oriented database management system* yang tersedia masih sangat terbatas dan memiliki banyak kekurangan dibandingkan dengan yang relasional. Oleh karena itu, dalam kenyataannya implementasi arsitektur data masih lebih layak menggunakan *Object Relational Database Management System (ORDBMS)*.

Metodologi EAP yang ditunjukkan pada gambar 3.1 memang digunakan untuk perencanaan, tetapi kalau dipertahikan lebih lanjut terhadap proses-proses yang ada pada tahap 2 (arsitektur data, arsitektur aplikasi, dan arsitektur teknologi), sebenarnya masih bisa diperdalam tidak hanya sekedar konsep arsitektur saja untuk kepentingan analisis, tetapi bisa dilanjutkan dengan perancangan terhadap tiga komponen utama dari suatu arsitektur *enterprise*. Untuk membantu sejauh mana perancangan harus dihasilkan, sebagai acuan bisa digunakan *framework Zachman* (dibahas dalam bab tersendiri), agar artifak-artifak yang dihasilkan dalam perancangan bisa terukur.

Hasil perancangan arsitektur *enterprise* dilanjutkan dengan implementasi terhadap masing-masing sistem informasi, disesuaikan dengan rencana implementasi yang telah ditetapkan pada tahap perencanaan.

### 3.5 Metodologi Alternatif

EAP merupakan salah satu metodologi yang paling pertama dan banyak digunakan dalam perencanaan pengembangan arsitektur *enterprise*, baik dari sistem yang tidak ada sama sekali, maupun pengembangan sistem yang telah ada. Seringin dengan perkembangan teknologi informasi dan kompleksitas proses bisnis, diperkenalkan beberapa metodologi baru yang tidak hanya fokus terhadap tahapan pengembangan saja, tetapi fokus juga pada artifak-artifak lain yang bisa menunjang komponen utama arsitektur *enterprise* (proses bisnis, data, aplikasi, dan teknologi). Selain itu metodologi yang baru lebih menggunakan pendekatan objek daripada pendekatan terstruktur.

Metodologi lainnya sebagai alternatif selain dari EAP diantaranya adalah sebagai berikut:

1. TOGAF-ADM

Metodologi ini dikembangkan oleh The Open Group, dan merupakan metodologi yang bersifat iteratif disesuaikan dengan tujuan dari pengembangan arsitektur *enterprise*. Bisa diterapkan untuk tujuan perencanaan, perancangan bahkan sampai dengan implementasi<sup>[7]</sup>.

2. OADP

Metodologi ini dikembangkan oleh salah satu vendor *database* yaitu Oracle. Karena vendor ini telah banyak mengembangkan produk *Enterprise Resource Planning* (ERP), maka vendor ini mengeluarkan salah satu alternatif metodologi untuk pengembangan arsitektur *enterprise* yaitu Oracle *Architecture Development Process* (OADP). Sama seperti TOGAF-ADM metodologi ini juga bersifat iteratif untuk menghasilkan arsitektur yang lengkap[].

3. EUP

Metodologi ini merupakan ekstensi dari SDLC yang bernama *Rational Unified Process* (RUP) yang sudah banyak digunakan dalam pengembangan suatu *software* berorientasi objek. EUP dikembangkan untuk kepentingan pengembangan arsitektur *enterprise* yang cakupannya luas dan berguna dalam menyusun strategi pengembangan sistem informasi *enterprise*. Pada dasarnya EUP menambahkan tahapan *enterprise dicipline* yang merupakan penyusunan awal arsitektur *enterprise* sebelum pengembangan setiap sistemnya dengan menggunakan RUP<sup>[1, 6]</sup>.

Ketiga metodologi tersebut secara *default* menggunakan pendekatan objek. Sebagai konsekwensinya, proses pengembangan yang dilakukan tidak bersifat linier seperti pada EAP tetapi lebih bersifat iteratif, artinya pengembangan dilakukan secara *step-by-step* dari mulai ruang lingkup yang kecil secara sedikit demi sedikit. Dalam pelaksanaanya juga lebih banyak melibatkan *stakeholder* dalam memenuhi kelengkapan kebutuhan sistem informasi yang sesuai secara maksimal.

# Bab 4

## ARSITEKTUR BISNIS

---

### 4.1 Definisi Proses Bisnis

Laudon<sup>[15]</sup> menjelaskan tentang istilah proses bisnis dengan menggunakan definisi sebagai berikut.

*“Business process refer to the set of logically related tasks and behaviors that organizations develop over time to produce specific business results and the unique manner in which these activities are organized and coordinated.”*

Proses bisnis bisa dikatakan sebagai sekelompok tugas dan aktivitas yang saling berhubungan secara logis, serta diatur dan dikoordinasikan supaya bisa menghasilkan suatu bisnis yang spesifik dan unik. Dengan kata lain, dalam pemodelan proses bisnis akan dilakukan pengujian struktur organisasi, mengetahui peran-peran yang dijalankan dalam organisasi, dan menunjukkan keterkaitan antara satu sama lainnya. Pemodelan proses bisnis juga tidak hanya untuk memahami apa yang terjadi di dalam organisasi, tetapi juga untuk pemahaman yang di luar organisasi.

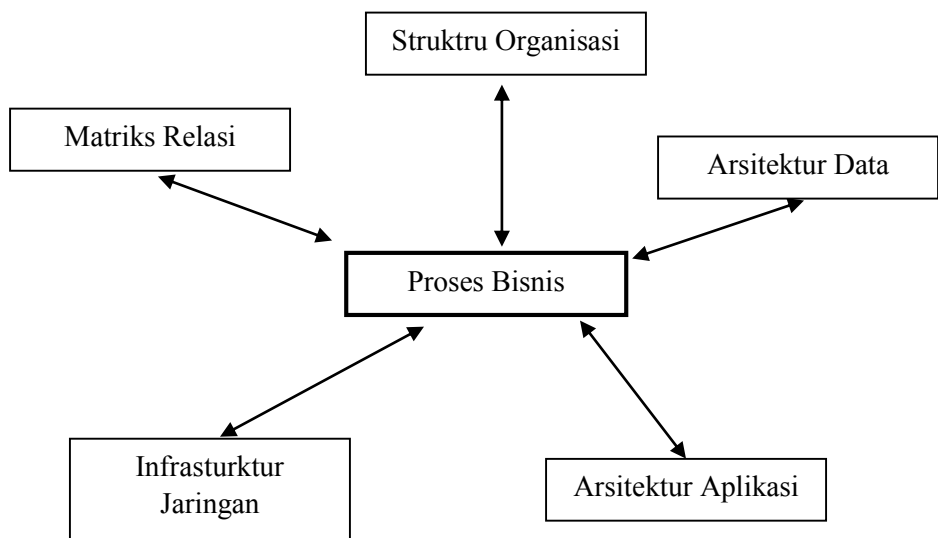
Tujuan memodelkan proses bisnis adalah sebagai berikut:

1. Mendapatkan pemahaman yang benar tentang visi organisasi dan perangkat pendukungnya, serta mendokumentasikan apa saja yang dilakukan oleh organisasi. Bisa juga untuk menjelaskan siapa saja yang terlibat dalam proses bisnis, dan menggunakan entitas bisnis apa saja.
2. Bisa menjadi sarana untuk melakukan rekayasa ulang proses bisnis yang terjadi. Dengan demikian bisa dianalisis efektifitas proses bisnis dan permasalahannya untuk dilakukan perbaikan.

Semua proses bisnis pada intinya harus bisa menjalankan kebutuhan sebagai berikut:

1. Mendukung secara optimal proses pengelolaan sumber daya (*resource enterprise*).
2. Menghasilkan nilai tambah terhadap hasil bisnis yang dilakukan pada proses bisnis sebelumnya.

Penyusunan arsitektur bisnis merupakan kegiatan yang paling utama dan memegang peranan penting dalam pengembangan arsitektur *enterprise*, karena dari hasil analisis proses bisnis *enterprise* inilah, maka bisa dianalisis lebih lanjut tentang kebutuhan komponen arsitektur *enterprise* yang lainnya, seperti yang ditunjukkan pada gambar 4.1. Dari gambaran proses bisnis, bisa diketahui bagaimana gambaran data yang diperlukan, siapa saja melakukan apa saja dalam menjalankan proses bisnisnya, gambaran aplikasi apa saja yang diperlukan oleh *enterprise*, gambaran infrastruktur jaringan komputer seperti apa yang diperlukan, dan seterusnya.



Gambar 4.1 Hubungan Arsitektur Bisnis dan Komponen Arsitektur *Enterprise*

Dengan demikian, kesalahan dalam menyusun arsitektur bisnis akan berakibat fatal terhadap keseluruhan tahapan pengembangan arsitektur *enterprise*. Hal ini dikarenakan, proses bisnis merupakan *core architecture* yang menyebabkan berakibat langsung terhadap penyusunan komponen-komponen arsitektur lainnya.



Oleh karena ini, visualisasi proses bisnis menjadi hal yang sangat penting. Visualisasi proses bisnis sangat bermanfaat untuk memberikan gambaran bagaimana proses bisnis yang sedang dilakukan *enterprise* saat ini dengan menggunakan simbol-simbol tertentu, sehingga bisa memberikan gambaran yang jelas bagaimana tingkat kerumitan proses bisnisnya.

Adanya sebuah model proses bisnis selain memperjelas proses, juga memungkinkan dilakukannya *Business Process Reengineering* (BPR) dalam rangka efisiensi proses bisnis yang bisa dilakukan oleh *enterprise* di masa yang akan datang, dimana pada akhirnya bisa lebih meningkatkan kinerja *enterprise* itu sendiri.

Langkah-langkah yang umum dilakukan dalam mendokumentasikan keseluruhan proses bisnis adalah sebagai berikut:

1. Dokumentasi struktur organisasi *enterprise* yang menjadi objek pengembangan. Struktur organisasi sangat terkait dengan proses bisnis yang dilakukan karena menunjukkan gambaran siapa melakukan apa dalam keseluruhan proses bisnis.
2. Mengidentifikasi dan menentukan keseluruhan proses bisnis yang berjalan. Pengelompokan proses bisnis berdasarkan pada area fungsi bisnis.
3. Mendistribusikan hasil pembuatan arsitektur bisnis kepada pihak-pihak terkait.

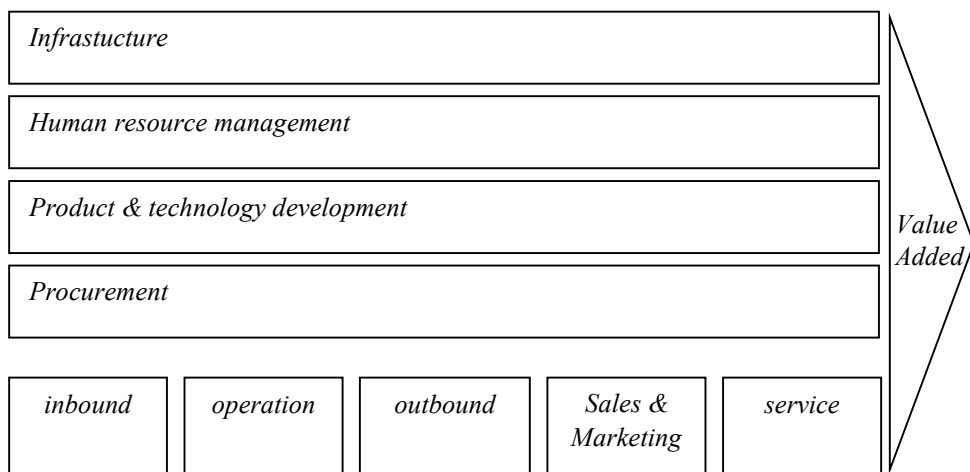
Sebelum melakukan analisis proses bisnis bisa juga dilakukan analisis terhadap sistem dan teknologi yang ada, sejauh mana digunakan untuk kepentingan proses bisnis. Langkah-langkah dasar dalam melakukan analisis sistem teknologi saat ini adalah sebagai berikut:

1. Menentukan ruang lingkup, objek, dan perencanaan kerja (*workplan*) dari IRC.
2. Mempersiapkan koleksi data
3. Koleksi data
4. Data Entry
5. Validasi IRC
6. Menggambar skema
7. Distribusi IRC
8. Mengatur dan memelihara hasil dari IRC

## 4.2 Analisis *Value Chain*

*Value chain* merupakan *global description of enterprise business process*, dimana bermanfaat untuk menggambarkan proses bisnis secara menyeluruh yang berjalan dalam suatu *enterprise*, dan bagaimana keterkaitan antara seluruh proses bisnis tersebut dalam rangka menghasilkan nilai tambah suatu *enterprise*. Nilai tambah (*value added*) bisa disebut juga dengan *margin* atau visi/misi dari suatu *enterprise*. *Value chain* sangat praktis untuk menjelaskan proses bisnis dari mulai hulu sampai hilir dimana terjadi nilai tambah akibat dari beberapa proses terkait dijalankan. *Value chain* pertama kali dikemukakan oleh Michael Porter pada tahun 1985 (seperti pada gambar 4.2).

*Value Chain* pada awalnya dimaksudkan untuk menjelaskan gambar umum proses bisnis di lingkungan *enterprise* dalam dunia industri manufaktur. Tetapi dalam perkembangannya, banyak juga digunakan untuk penggambaran global proses bisnis *enterprise* secara umum.



Gambar 4.2 *Value Chain*<sup>[31]</sup>

Teori ini pada prinsipnya memperkenalkan suatu kerangka visual yang sederhana untuk membantu menjelaskan bagaimana proses bisnis *enterprise* yang ada secara menyeluruh (dari hulu sampai hilir) dengan cara yang sederhana yaitu dengan mengidentifikasi dan menginventarisasi proses bisnis keseluruhan ke dalam area-area proses bisnis sesuai dengan fungsinya yang berjalan dalam suatu *enterprise*.

*Value Chain* membagi proses bisnis *enterprise* menjadi dua kelompok besar fungsi bisnis yaitu:

1. *Primary activities*

Bagian menggambarkan urutan kegiatan utama yang dilakukan oleh *enterprise (core business enterprise)*.

2. *Support activities*

Bagian yang menggambarkan kegiatan-kegiatan yang sifatnya membantu *core business*.

1. *Primary Activities* (Aktivitas utama)

Merupakan area-area fungsi bisnis utama yang merupakan *core activities* dari suatu *enterprise*, yang saling berkaitan untuk mengumpulkan masukan, menerapkan transformasi pertambahan nilai dari satu area fungsi ke area fungsi lainnya selanjutnya sampai menjadi keluaran dari *enterprise*.

Aktivitas utama bisa dibagi menjadi beberapa area fungsi bisnis yang meliputi fungsi logistik masukan (*inbound logistics*), operasi (*operations*), logistik keluaran (*outbound logistics*), pemasaran dan penjualan (*marketing and sales*), dan layanan (*service*). Aktivitas utama tidak akan bisa berjalan dengan sebagaimana mestinya jika tidak ditunjang oleh berbagai aktivitas pendukung yang diperlukan.

2. *Support Activities* (Aktivitas pendukung)

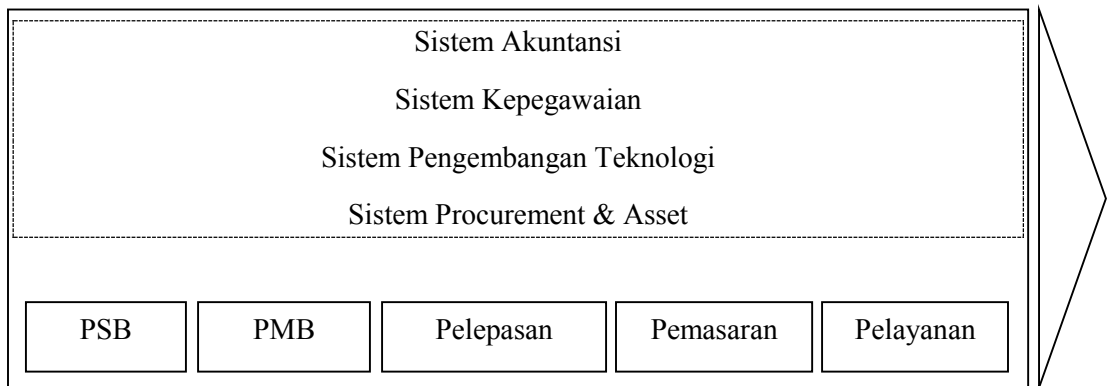
Kegiatan yang meliputi area-area fungsi bisnis yang mengelola sumber daya *enterprise* dan sifatnya memberikan dukungan terhadap hampir di setiap tahapan pertambahan nilai atau area fungsi pada aktivitas utama.

Aktivitas pendukung terdiri dari beberapa area fungsi bisnis yaitu fungsi infrastruktur perusahaan (*firm infrastructure*), manajemen sumber daya manusia (*human resource management*), pengembangan teknologi (*technology development*), dan pengadaan (*procurement*).

Setiap area fungsi bisnis baik di aktivitas utama maupun di aktivitas pendukung, dapat lebih diperjelas lagi dengan menggambarkan proses bisnis yang berjalan secara detail yaitu menggunakan *tool* lain seperti BPMN (*Business Process Modeling Notation*), *Use Case Business Model*, dll yang lebih tepat dalam penjabaran dari deskripsi masing-masing area fungsi bisnis tersebut.

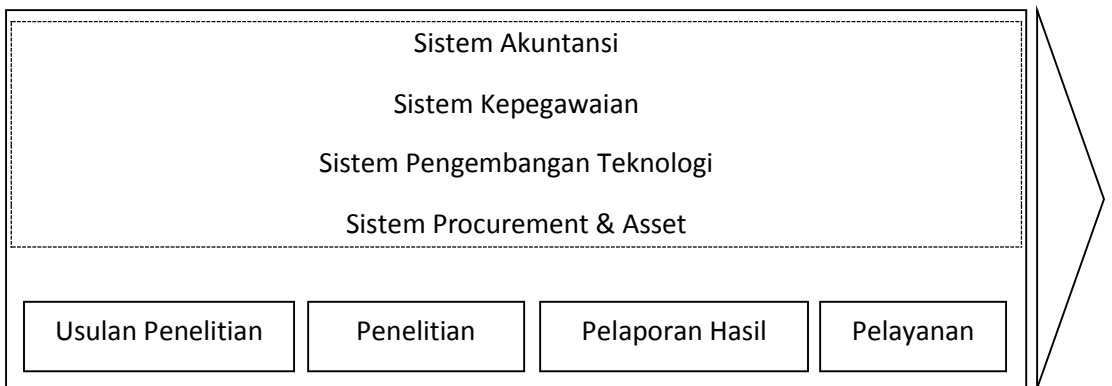
Pemodelan dengan *value chain* dengan modifikasi yang tepat bisa juga diterapkan pada jenis *enterprise* lainnya selain manufaktur. Sebagai contoh kasus, dalam suatu *enterprise* berbentuk institusi pendidikan seperti SMA, SMP, dll<sup>[21, 26, 27, 29, 30]</sup>. Gambaran umum proses bisnis yang sedang berlangsung di dalamnya bisa dijelaskan dengan menggunakan *value chain*

seperti yang ditunjukkan pada gambar 4.3. Proses bisnis utama dimulai dari Penerimaan Siswa Baru (PSB), Proses Belajar Mengajar (PMB), Pelepasan, Pemasaran, dan Pelayanan. Aktivitas utama ini didukung oleh sistem kepegawaian, sistem akutansi, sistem pengembangan teknologi, dan sistem *procurement & aset*.



Gambar 4.3 *Value Chain* Institusi Pendidikan

Seperti pada gambar 4.4, dalam suatu institusi penelitian kegiatan utamanya adalah penelitian itu sendiri yang dimulai dari pengajuan proposal berupa usulan penelitian oleh setiap peneliti<sup>[8, 9]</sup>. Penelitian biasanya didanai dari berbagai sumber yang berasal dari dana pemerintah. Berdasarkan usulan penelitian maka akan dilaksanakan penelitian itu sendiri untuk jangka waktu tertentu. Selama penelitian dilakukan berbagai kegiatan diantaranya mengikuti seminar, progress report berkala, pembuatan *prototype* dsb.



#### Gambar 4.4 *Value Chain* Institusi Penelitian

Dalam tahap akhir penelitian dilakukan pembuatan laporan hasil akhir penelitian sebagai bentuk pertanggungjawaban penggunaan anggaran pemerintah. Selanjutnya sebagai wujud tanggung jawab peneliti sebagai layanan kepada publik memberikan penyebaran hasil penelitian dalam berbagai bentuk kegiatan sosialisasi agar publik mengetahui dan bisa mengambil manfaat dari hasil penelitian yang telah dilakukan. Untuk menjalankan penelitian tentunya perlu didukung kegiatan pendukung lainnya yang dilakukan oleh berbagai pihak yang bersifat administratif seperti kegiatan pengadaan barang, pencatatan keuangan, koordinasi pegawai dsb yang secara tidak langsung akan menentukan kesuksesan kegiatan utama.

### **4.3 *Business Prosess Modeling Notation (BPMN)***

BPMN adalah alat bantu untuk menggambarkan alur proses bisnis dengan menggunakan notasi-notasi tertentu<sup>[10]</sup>. Dilihat dari fungsinya BPMN merupakan pengembangan dari alat bantu sebelumnya seperti diagram *flowchart*, diagram aktivitas, dan diagram alir data. Dengan demikian, BPMN memiliki fitur-fitur yang merupakan gabungan dari ketiga diagram tersebut, ditambah dengan fitur-fitur lainnya untuk lebih menyempurnakan penggambaran proses bisnis.


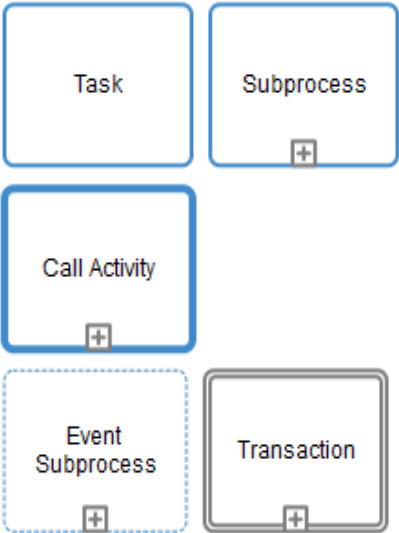

Melalui visualisasi proses bisnis yang tepat dengan menggunakan BPMN ini, diharapkan tingkat kerumitan suatu proses bisnis bisa dipahami dengan mudah oleh siapa saja yang berkepentingan. Dengan demikian BPMN membantu menyamakan persepsi terhadap suatu proses bisnis yang dianalisis, dan menghindari kesalahpengertian diantara pihak-pihak terkait tentang proses bisnis yang akan didukung oleh sistem informasi nantinya. Apabila diperlukan bisa dilakukan perbaikan proses bisnis yang ada (*business process reengineering*) untuk penyempurnaan terhadap proses bisnis *enterprise* secara keseluruhan.






























































BPMN secara umum tersusun dari empat komponen utama yaitu sebagai berikut:

#### ***FLOW OBJECTS***

Notasi yang digunakan untuk menunjukkan alur objek dalam suatu proses bisnis, sebagaimana yang ditunjukkan pada tabel 4.1 dan gambar 4.5.

Tabel 4.1 *Flow Objects*

<b>Object</b>	<b>Keterangan</b>	<b>Simbol</b>
<i>Event</i>	Menunjukkan kejadian dari suatu kegiatan dimulai dari <i>Start</i> , <i>Intermediate</i> , <i>End</i> . (selengkapnya di gambar 4.5)	
<i>Activity</i>	Menunjukkan aktivitas yang dilakukan oleh suatu proses bisnis.	
<i>Gateway</i>	Fungsinya untuk menunjukkan alur percabangan atau pilihan seperti <i>Forking</i> , <i>Merging</i> , <i>Joining</i> .	


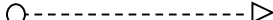

Type	Start			Intermediate				End
	Normal	Event Subprocess	Event Subprocess non-interrupt	catch	boundary	boundary non-interrupt	throw	
None								
Message								
Timer								
Conditional								
Link								
Signal								
Error								
Escalation								
Termination								
Compensation								
Cancel								
Multiple								
Multiple Parallel								

Gambar 4.5 Jenis Event

## CONNECTING OBJECTS

Notasi (tabel 4.2) untuk menunjukkan hubungan antar objek dalam proses bisnis tergantung pada kedekatan objeknya.


Tabel 4.2 *Connecting Objects*

<b>Object</b>	<b>Keterangan</b>	<b>Simbol</b>
<i>Sequence flow</i>	Menunjukkan urutan aktivitas yang dilakukan antar proses bisnis dalam suatu <i>pool</i> .	
<i>Message flow</i>	Menunjukkan alur pesan yang terjadi antara dua proses yang tidak berada dalam suatu <i>pool</i> yang sama.	
<i>Association</i>	Menunjukkan data berupa <i>input</i> atau <i>output</i> yang diberkaitan dengan suatu proses	

## **SWIMLANES**

Seperti pada tabel 4.3, *swimlane* untuk mengelompokkan aktivitas ke dalam partisipan proses yang menjalankan fungsi atau tanggungjawab secara visual.

Tabel 4.3 *Swimlane*

<b>Object</b>	<b>Keterangan</b>	<b>Simbol</b>
<i>Pool</i>	Merepresentasikan pelaku suatu aktivitas. Sehingga bisa diketahui siapa melakukan apa dalam suatu sistem.	
<i>Lane</i>	Pengelompokan pelaku yang lebih detail.	

Menggunakan *pool* atau tidak, itu sangat tergantung kebutuhan dalam menganalisis proses bisnis. Tanpa menggunakan *pool* proses bisnis dapat digambarkan secara umum saja tanpa perlu mengetahui siapa-siapa yang



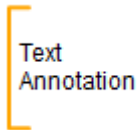


terlibat. Sedangkan dengan menggunakan *pool*, informasi lebih rinci bisa lebih digambarkan.

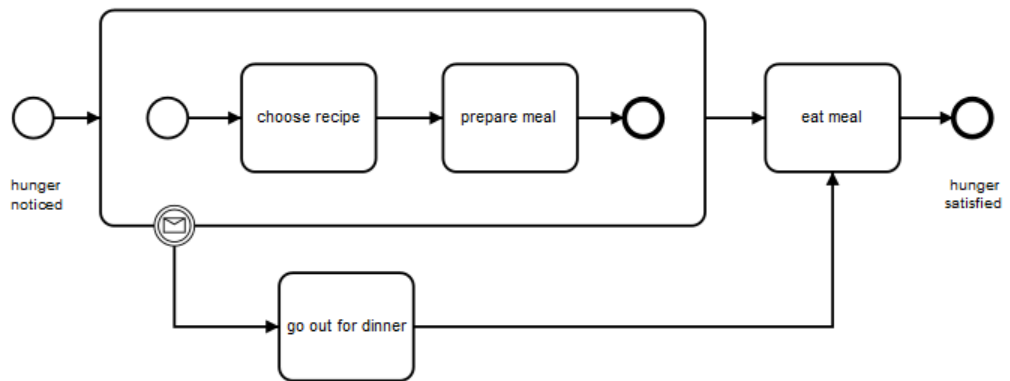
## ARTIFACTS

Komponen untuk menjelaskan segala macam bentuk dokumen data yang terkait dalam suatu aktivitas tertentu dalam proses bisnis (tabel 4.4).

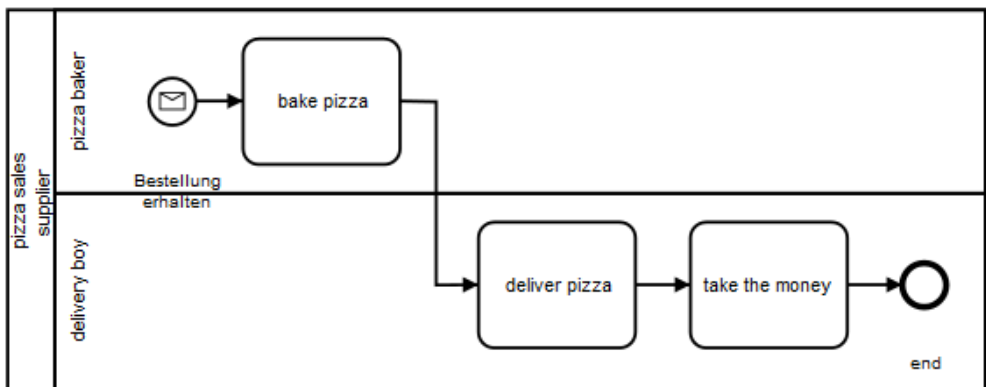
Tabel 4.4 *Artifact*

<b>Object</b>	<b>Keterangan</b>	<b>Simbol</b>
<i>Data object</i>	Menunjukkan data apa yang dibutuhkan oleh suatu aktivitas, atau data apa yang dihasilkan oleh suatu aktivitas.	 <p>The image shows two BPMN symbols: a 'Data Object' represented by a rectangle with a folded top-right corner, and a 'Data Store' represented by a cylinder.</p>
<i>Group</i>	Tujuannya untuk lebih memudahkan dalam menganalisis proses dan pendokumentasian. Tidak berpengaruh apa-apa terhadap aktivitas sama sekali.	 <p>The image shows a BPMN Group symbol, which is a rounded rectangle with a dashed border.</p>
<i>Annotation</i>	Merupakan komentar atau deskripsi tambahan agar bisa lebih memperjelas diagram. Gunakan komentas secara proposional.	 <p>The image shows a BPMN Text Annotation symbol, which is a bracket-shaped rectangle.</p>

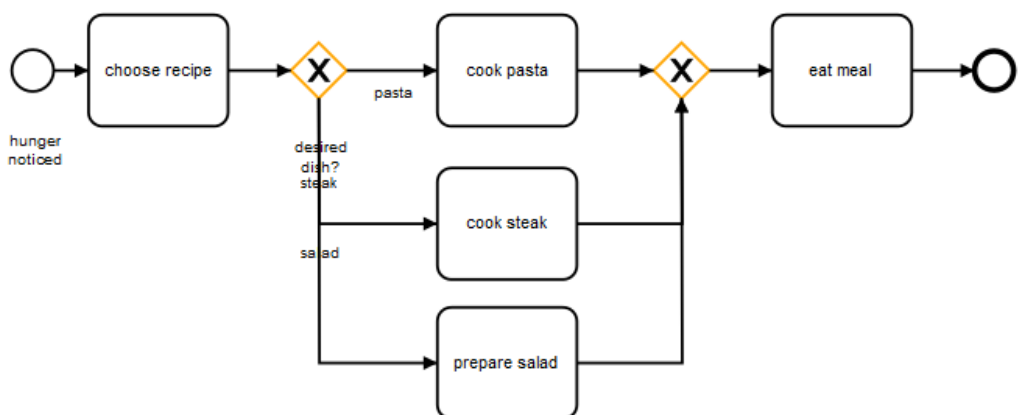
Ada beberapa *software* di pasaran yang bisa digunakan sebagai alat bantu untuk menggambarkan BPMN adalah Microsoft Visio, YaoQiang BPMN Editor, Bizagi *Process Modeller* (standar kit nya bisa diunduh secara gratis), baik yang bersifat *open source* maupun yang bersifat *proprietary*. Contoh visualisasi proses bisnis dengan BPMN ditunjukkan pada gambar 4.6 s/d gambar 4.12.



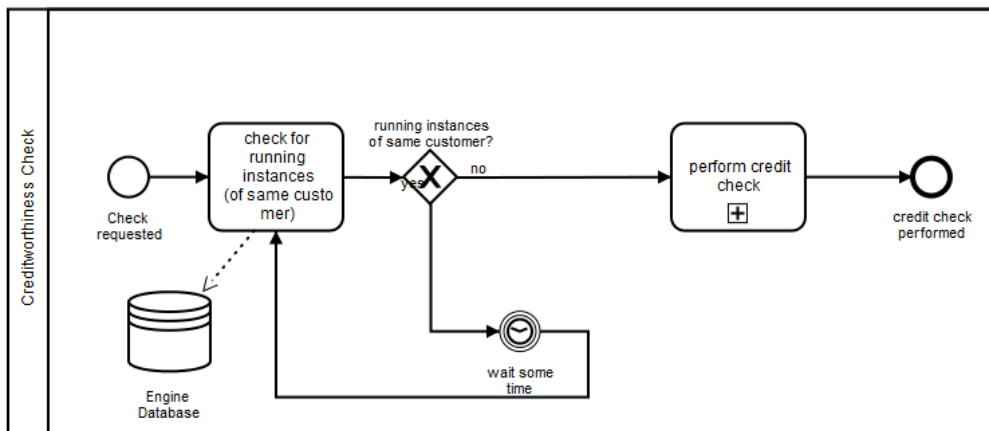
Gambar 4.6 *Looping* dengan Pembatasan *Event Email*



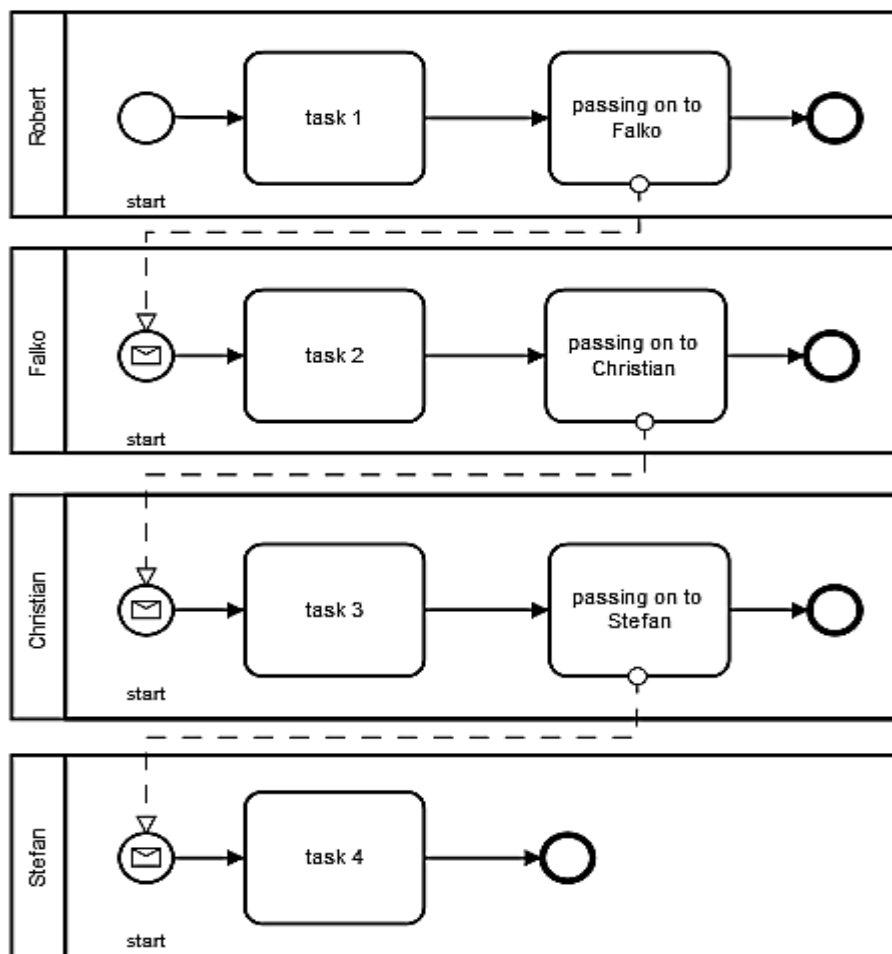
Gambar 4.7 Penggunaan *Pool* dan *Swimlane*



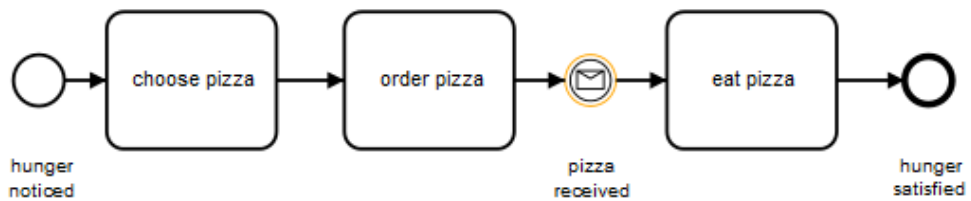
Gambar 4.8 Penggunaan *Gateway*



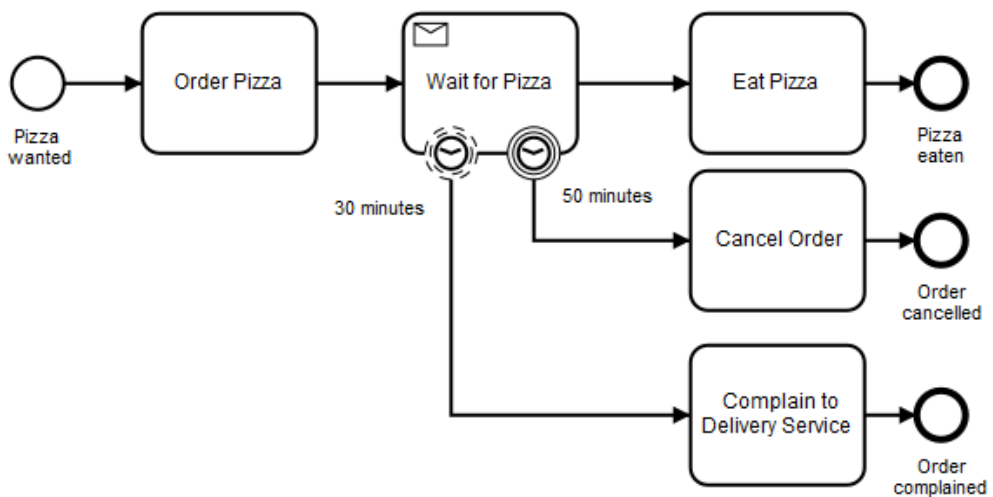
Gambar 4.9 Penggunaan *Artifact*



Gambar 4.10 Contoh Penggunaan *Pool*



Gambar 4.11 Penggunaan *Event Email*



Gambar 4.12 Penggunaan *Interruption*

#### 4.4 Diagram *Use Case* Bisnis

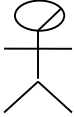
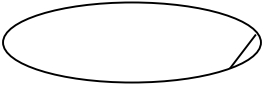

Setelah proses bisnis divisualisasikan dengan menggunakan BPMN, untuk lebih memperjelas lagi bagaimana proses bisnis yang terjadi, maka sebagai langkah berikutnya adalah dengan membuat rangkuman proses bisnis. Rangkuman proses bisnis tersebut merupakan kumpulan layanan (*service*) apa saja yang harus disediakan oleh sistem informasi dalam bentuk *use case* yang menunjukkan dukungan terhadap proses bisnis. Sebuah *use case* bisa saja mencakup beberapa proses bisnis sekaligus.


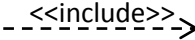
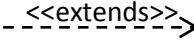
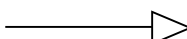
Dalam kaitan penyusunan proses bisnis biasanya disebut juga dengan *use case business*. Diagram *use case* merupakan salah satu diagram UML (*Unified Modeling Language*) yang sering digunakan dalam penyusunan sistem

informasi karena sifatnya yang praktis dalam menggambarkan secara global tentang layanan berupa aplikasi sistem apa saja yang diperlukan untuk menunjang proses bisnis yang diinginkan. Dengan demikian, diagram ini membantu dalam menentukan sistem informasi apa yang diperlukan untuk melayani kepentingan proses bisnis apa yang sesuai<sup>[2,4]</sup>.

Diagram *use case* bisnis menggunakan beberapa notasi sederhana seperti dijelaskan pada tabel 4.5. Sedangkan Gambar 4.13 menunjukkan contoh penggunaan *use case* bisnis.

Tabel 4.5 Notasi *Use Case* Bisnis

<b>Notasi</b>	<b>Keterangan</b>
<p><i>Actor</i> Bisnis</p> 	<p>Menunjukkan seseorang atau sesuatu yang berperan atau berinteraksi dalam proses bisnis. Walaupun digambarkan dengan simbol orang, <i>actor</i> tidak hanya orang tetapi bisa juga berupa kelompok atau sistem.</p>
<p><i>Use Case</i> Bisnis</p> 	<p>Menggambarkan proses bisnis apa yang dilakukan. Dengan kata lain menjelaskan aktivitas bisnis utama apa saja yang bisa dilakukan dalam sebuah <i>enterprise</i>.</p> <p><i>Use case</i> bisnis menggunakan format penulisan nama berupa kata kerja atau frase kata kerja karena menunjukkan proses. Setiap <i>use case</i> bisnis diberi penjelasan tambahan yang lebih rinci tentang apa saja yang terjadi di dalamnya.</p> <p>Aliran kerja dalam suatu <i>use case</i> bisnis dapat didokumentasikan dengan menggunakan nomor urut yang menunjukkan tahapan demi tahapan aktivitas dalam sebuah proses bisnis.</p>
<p><i>Boundary</i></p> 	<p>Menunjukkan batasan-batasan sistem, karena suatu sistem yang kompleks bisa saja dibagi menjadi beberapa sub-sistem yang lebih sederhana untuk memudahkan pemahaman sistem secara keseluruhan.</p>

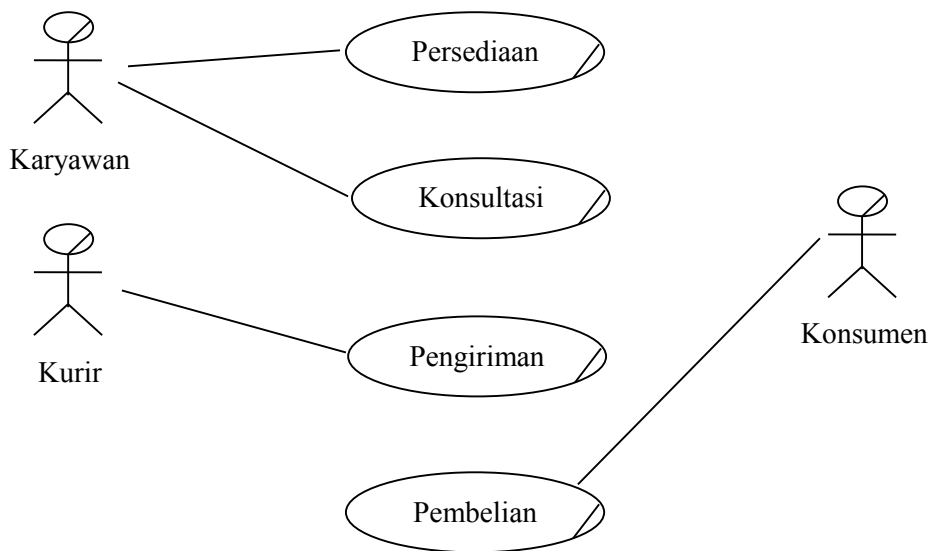
<i>Assosiation</i> 	Menyatakan interaksi antara <i>actor</i> bisnis dan <i>use case</i> bisnis, baik hubungan dari <i>actor</i> bisnis ke <i>use case</i> bisnis maupun dari <i>use case</i> bisnis ke <i>actor</i> bisnis.
<<include>> 	Menunjukkan hubungan yang sangat erat antara dua buah <i>use case</i> sehingga akibatnya <i>use case</i> yang dituju harus selalu diikutsertakan dalam setiap proses di <i>use case</i> yang menggunakannya.
<<extends>> 	Menunjukkan perluasan proses dari <i>use case</i> yang sudah ada.  Karena penerapannya sangat mirip penggunaan generalisasi, maka terkadang <<extends>> digunakan untuk <i>use case</i> yang bersifat <i>optional</i> (tidak selalu) artinya hanya terjadi pada waktu-waktu tertentu saja pada saat <i>runtime</i> atau sistem berjalan.
Generalisasi ( <i>Generalization</i> ) 	Menunjukkan hubungan umum-khusus antara dua buah <i>use case</i> .  Arah panah harus menghadap ke <i>use case</i> yang memiliki sifat umumnya.

Setiap *use case* bisnis yang telah teridentifikasi sebaiknya dilengkapi dengan keterangan yang berisi penjelasan singkat maupun penjelasan detail, supaya memperjelas keterkaitan antara *use case* dengan proses bisnis yang didukungnya. Sebagai contoh format penjelasan *use case* bisnis seperti ditunjukkan pada tabel 4.6.

Dari hasil penggambaran rangkuman proses bisnis dengan menggunakan diagram *use case* ini, akan memudahkan dalam penentuan kandidat aplikasi sistem yang dibutuhkan. Setiap *use case* minimal menunjukkan ke satu aplikasi sistem, atau bisa jadi dua *use case* atau lebih bisa ditangani oleh satu aplikasi sistem.

Tabel 4.6 Format Penjelasan *Use Case* Bisnis

Item	Keterangan
Nama <i>Use Case</i> Bisnis	Memberi nama yang unik sebuah <i>use case</i> bisnis sebagai identitasnya.
<i>Actor</i> Bisnis	Menjelaskan siapa/apa yang berperan dalam menjalankan <i>use case</i> bisnis.
<i>Description</i>	Keterangan singkat untuk menjelaskan fungsi dari <i>use case</i> bisnis.
<i>Main/Basic flow</i>	Urutan aktivitas normal yang dijalankan per tahapan.
<i>Alternative/exception flow</i>	Aktivitas yang dilakukan apabila ada kejadian tertentu.



Gambar 4.13 Contoh Diagram *Use Case* Bisnis

Tabel 4.7 menerangkan lebih rinci tentang prose bisnis yang ada di dalam use case bisnis Pembelian.

Tabel 4.7 Penjelasan *Use Case* Bisnis Pembelian

Pembelian	Memberi nama yang unik sebuah <i>use case</i> bisnis sebagai identitasnya.
<i>Actor</i> Bisnis	Konsumen
<i>Description</i>	Menjelaskan bagaimana konsumen harus melakukan langkah-langkah dalam melakukan pembelian suatu barang.
<i>Main/Basic flow</i>	Secara umum tahapan pembelian barang sebagai berikut: <ol style="list-style-type: none"> <li>1. Pembeli memeriksa katalog untuk menentukan barang yang dibeli.</li> <li>2. Pembeli menentukan barang dan jumlah yang akan dibeli</li> <li>3. Pembeli menentukan cara bayar dan cara pengiriman barang.</li> </ol>
<i>Alternative/exception flow</i>	Jika jumlah barang yang dimaksud tidak tersedia di gudang, maka pembeli bisa membatalkan pembelian atau melakukan pemesanan barang.

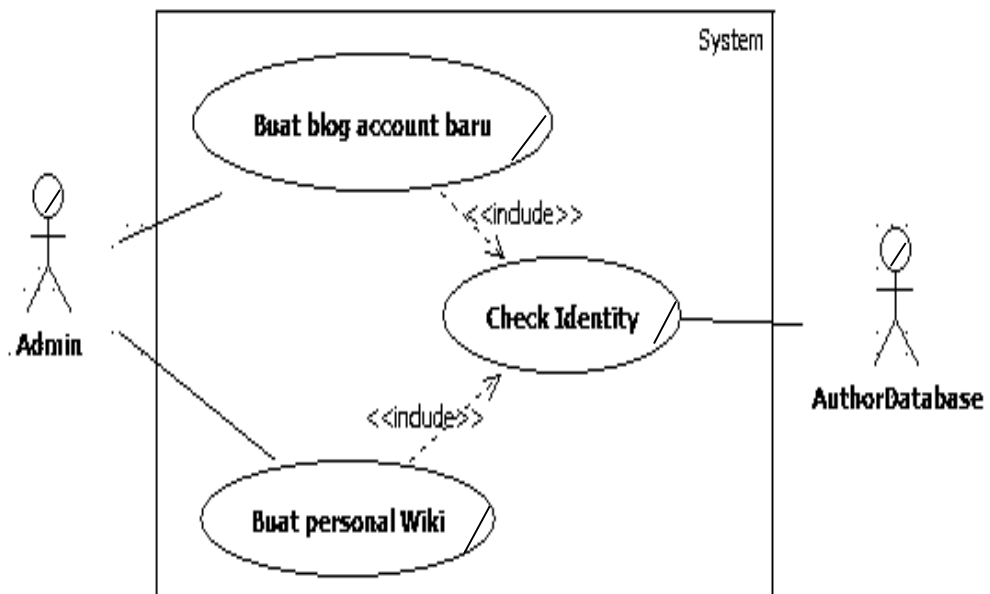
Sebenarnya dalam perencanaan arsitektur *enterprise* setiap *use case* bisnis tidak perlu diberi penjelasan terlalu rinci dan cukup dengan adanya keterangan singkat dari setiap *use case* bisnis saja. Karena dalam perencanaan fungsi use case untuk mencari kandidat aplikasi sistem yang diperlukan. *Use case* bisnis lebih berfungsi sebagai alat bantu untuk menghubungkan antara proses bisnis dan aplikasi sistem yang diperlukan.

Pada prinsipnya penggunaan *stereotype* <<include>>, <<extends>>, dan generalisasi bukanlah merupakan suatu keharusan tetapi tujuannya lebih bersifat memperjelas sebuah *use case*. Oleh karena itu, apabila penggunaan ketiga asosiasi tersebut malah memperumit fungsionalitas sistem sebaiknya dihindari dan jangan memaksakan untuk menggunakannya. Karena kadang-kadang hanya karena perbedaan sudut pandang saja sebenarnya antara *stereotype* <<include>> dan <<extends>> bisa menggambarkan kondisi sistem yang sama. Contoh penggunaan <<include>> seperti pada gambar 4.14, sedangkan penggunaan <<extends>> pada gambar 4.15.

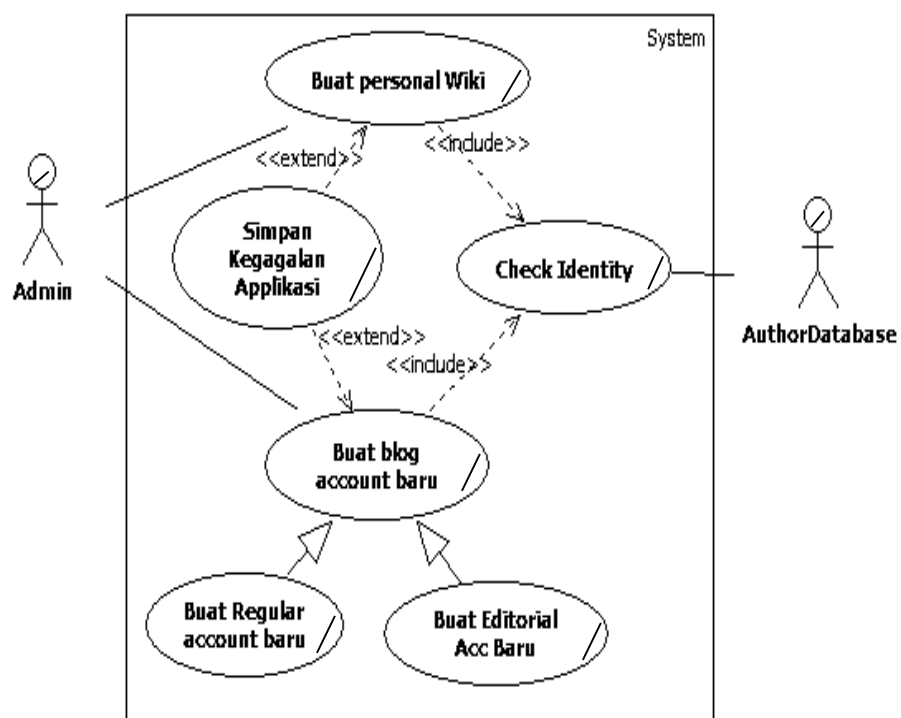


Di gambar 4.14 ini ada tiga buah *use case* bisnis yang bisa dijalankan oleh dua *actor* bisnis. Dalam contoh di gambar 4.14 *use case* bisnis “Check Identity” selalu dilakukan apabila *actor* bisnis “Admin” akan menjalankan proses bisnis “Buat Blog Account Baru” dan proses bisnis “Buat Personal Wiki”. Sementara *actor* bisnis “Author Database” bisa melakukan satu aktivitas saja yaitu proses bisnis “Check Identity”.

Pada gambar 4.15 *use case* bisnis “Simpan Kegagalan Aplikasi” hanya akan terjadi pada saat tertentu saja misalkan pada saat *actor* “Admin” melakukan pelanggaran prosedur yang semestinya harus dilakukan ketika menjalankan proses bisnis “Buat Personal Wiki” dan proses bisnis “Buat Blog Baru”. *Use case business* “Buat Blog Account Baru” sebenarnya terdiri dari dua tipe *account*, yang bisa dipilih oleh *actor* Admin.



Gambar 4.14 Penggunaan *Include*



Gambar 4.15 Penggunaan *Extends*

# Bab 5

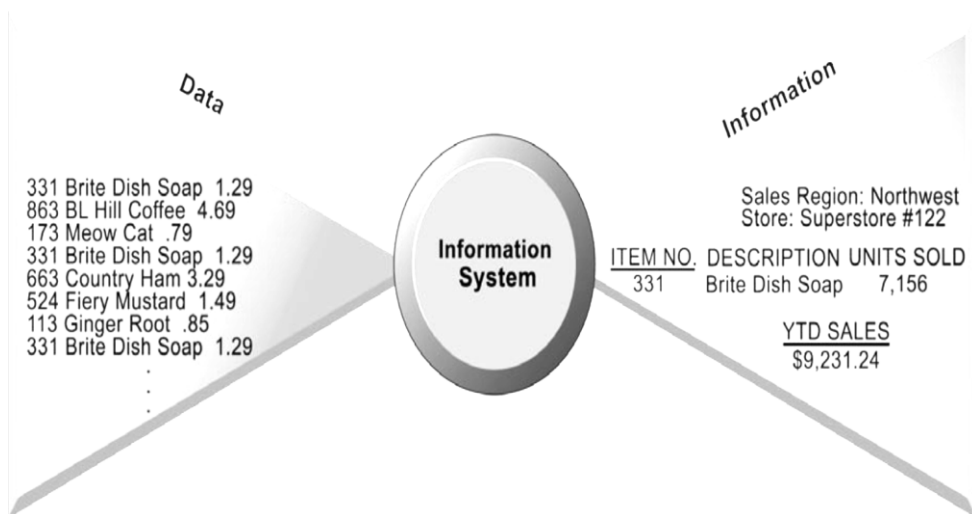
## ARSITEKTUR DATA

---

### 5.1 Pengertian Data dan Informasi

Data pada dasarnya berupa satuan fakta atau kejadian yang independen atau berdiri sendiri, dimana jika dikumpulkan maka akan menjadi sumber utama pembentukan suatu informasi.

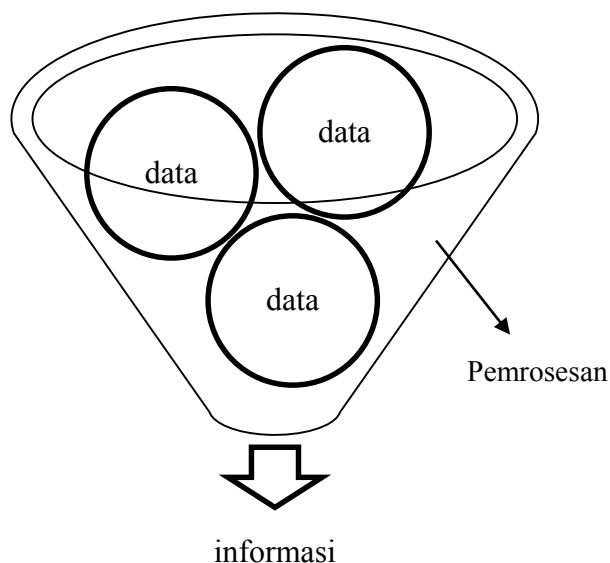
Data berupa serpihan fakta yang masih terpisah-pisah, sehingga data tersebut bisa dikatakan masih/sangat kurang nilainya dan tidak akan banyak diambil manfaatnya untuk kegiatan operasional bisnis apalagi untuk pengambilan keputusan kalau tidak dikumpulkan, disimpan dan diolah terlebih dahulu agar menjadi suatu informasi yang penting dalam proses pengambilan keputusan.



Gambar 5.1 Data dan Informasi<sup>[15]</sup>

Sebagaimana yang ditunjukkan pada gambar 5.1, bahwa dalam suatu toserba, contoh data adalah dimana seseorang membeli beberapa barang kebutuhan sehari-harinya. Setelah data dari beberapa orang dalam kurun waktu tertentu dikumpulkan dan disimpan, maka akan diperoleh beberapa informasi tentang berapa jumlah pembeli, barang-barang apa saja yang dibutuhkan pembeli, waktu kapan saja yang banyak kunjungan pembeli ke toserba, dll. Sehingga informasi tersebut bisa dimanfaatkan oleh manajer dalam proses pengambilan keputusan yang diperlukan dalam menjalankan kegiatan bisnisnya.

Dengan demikian, seperti dijelaskan pada gambar 5.2, informasi merupakan hasil dari proses pengolahan beberapa data sehingga mendapatkan nilai pengetahuan yang bermanfaat. Nilai manfaat informasi akan sangat tergantung pada bagaimana proses pengolahannya dan bagaimana informasi itu disajikan<sup>[11, 31]</sup>.



Gambar 5.2 Data dan Informasi

Informasi yang harus disediakan oleh suatu sistem informasi harus memenuhi kriteria tertentu agar memiliki kualitas yang layak digunakan dalam proses operasional, manajerial atau pengambilan keputusan dalam suatu *enterprise*.

Kriteria bahwa suatu informasi bisa dikatakan memiliki kualitas adalah sebagai berikut.

1. *Accuracy*

Terhindar dari kesalahan (sesuai dengan fakta apa adanya, tidak direkayasa), tidak mengandung arti/makna yang bias (*ambiguous*) atau memiliki kejelasan maksud, dan memenuhi prinsip-prinsip *completeness*, *correctness* dan *security*. Oleh karena itu, diperlukan proses validasi data yang teliti agar data yang dimasukkan benar-benar sesuai dengan fakta. Data yang tidak akurat menyebabkan informasi yang dihasilkannya juga tidak akurat (mengandung keraguan) sehingga tidak bisa digunakan untuk pengambilan keputusan.

2. *Up to date*

Sesuatu akan bermakna jika tepat waktunya pada saat benar-benar diperlukan. Informasi harus merupakan hasil pengolahan dari data yang terbaru dan bukan berdasarkan pada data yang sudah kadaluarsa. Informasi yang *up to date* sangat penting terutama untuk sistem informasi yang berbasis web dan *online*, karena tidak selalu yang online itu menyajikan informasi terkini kalau tidak dilakukan *up to date* secara periodik. Dengan demikian, untuk sistem informasi yang berbasis web dituntut untuk menyajikan informasi secara *online* dan *realtime*.

3. *Relevancy*

Sesuai dengan kebutuhan atau manfaat yang diharapkan oleh setiap pengguna informasi terutama bagi pihak-pihak (para manajer) di berbagai bidang dan level *enterprise* dalam proses pengambilan keputusan.

Ketiga kriteria tersebut harus bisa dipenuhi dalam suatu sistem pemrosesan data menjadi informasi, karena salah poin saja tidak dipenuhi, maka akan menyebabkan informasi tersebut tidak bermanfaat, artinya telah terjadi kegagalan dalam pemrosesan.

Langkah-langkah yang diperlukan dalam proses penyusunan arsitektur data adalah sebagai berikut:

1. Membuat daftar kandidat entitas data yang diperlukan dalam suatu proses bisnis, dan yang dihasilkannya
2. Menentukan pilihan entitas data, melengkapinya dengan atribut dan menentukan bagaimana hubungan antar entitasnya

3. Membuat hubungan antara entitas data dan fungsi bisnis
4. Menyebarkan arsitektur data

## 5.2 Penyusunan Data

Beberapa langkah yang diperlukan untuk melakukan penyusunan data yang diperlukan oleh keseluruhan proses bisnis *enterprise*, adalah sebagai berikut:

1. Pengumpulan entitas data
2. Hubungan antar entitas
3. Normalisasi data

Penyusunan arsitektur data pada dasarnya adalah bagaimana mendefinisikan jenis data utama yang dibutuhkan untuk mendukung proses bisnis *enterprise*. Arsitektur data terdiri dari sekumpulan entitas dimana setiap entitas memiliki beberapa atribut yang menunjukkan karakteristik dari entitas tersebut dan memiliki relasi/hubungan terhadap entitas yang lain.

Beberapa langkah dasar untuk mendefinisikan arsitektur data adalah sebagai berikut.

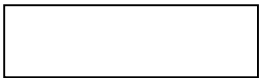
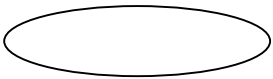


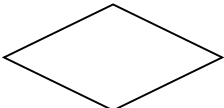

1. Menganalisis entitas yang bisa dilakukan dengan proses bisnisnya (BPMN) dimana biasanya disebutkan beberapa entitas awal. Dilanjutkan dengan menelusuri *use case* bisnisnya, maka akan didapatkan beberapa entitas tambahannya.
2. Daftarkan calon entitas data dengan meninjau model bisnis dan deskripsi sistem dan teknologi yang dipakai.
3. Tetapkan entitas yang akan dipakai.
4. Definisikan setiap entitas tersebut dan mendokumentasikannya (Diagram ER).
5. Hubungkan entitas data dengan fungsi bisnis detail.

## 5.3 Diagram *Entity Relationship* (ER)

Diagram ini untuk menggambarkan kebutuhan data dari suatu sistem informasi dengan menunjukkan entitas-entitas, dan keterkaitan antar entitas, serta dilengkapi dengan atribut-atribut dari setiap entitasnya seperti pada gambar 5.3<sup>[23, 28]</sup>. Pada dasarnya diagram ER merupakan model dasar untuk

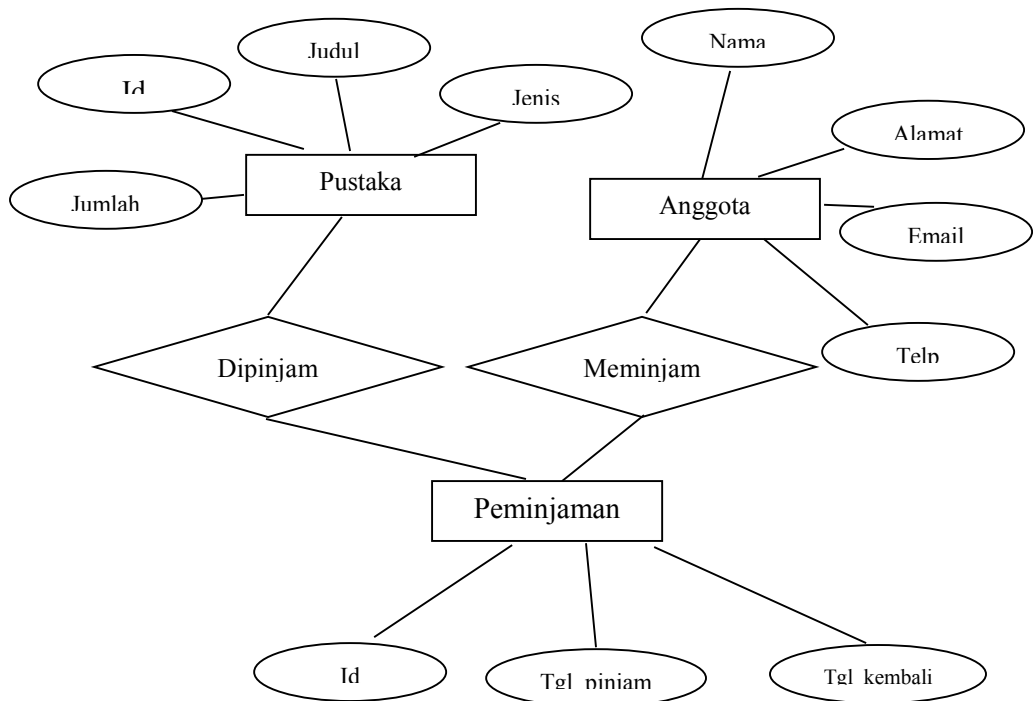
perancangan dan pembuatan struktur *database* pada sistem *relational database*. Notasi dalam diagram ER seperti pada tabel 5.1.

Tabel 5.1 Notasi ERD

Notasi	Keterangan
<p><i>Entity</i> (entitas)</p> 	Entitas adalah kandidat tabel dalam suatu basisdata yang akan menyimpan satuan data yang diperlukan sistem.
<p><i>Attribute</i> (atribut)</p> 	Bisa dikatakan juga sebagai kolom atau field dari suatu tabel basisdata.
<p><i>Primary Key Attribute</i></p> 	Atribut yang menunjukkan kunci utama dari suatu entitas. Atribut ini harus bersifat unik, artinya bahwa diantara setiap baris data yang tersimpan dalam suatu entitas tidak boleh menyimpan data yang sama.
<p><i>Multivalued Attribute</i></p> 	Atribut data berupa <i>field</i> /kolom khusus memiliki nilai lebih dari satu.
<p><i>Relation</i> (relasi)</p> 	Menunjukkan hubungan antar entitas. Penamaanya umumnya dimulai dengan kata kerja.
<p><i>Association</i> (asosiasi)</p> 	Penghubung antara relasi dan entitas di mana di kedua ujungnya dituliskan <i>multiplicity</i> kemungkinan jumlah pemakaian.

Langkah-langkah yang harus dilakukan dalam penyusunan diagram ER adalah sebagai berikut:

1. Membuat daftar entitas berdasarkan pengamatan proses bisnis saat ini.
2. Membuat keterkaitan antara satu entitas dengan entitas lainnya.
3. Melakukan proses normalisasi terhadap entitas yang berhubungan sehingga diperoleh hubungan antar entitas dalam kondisi normal.



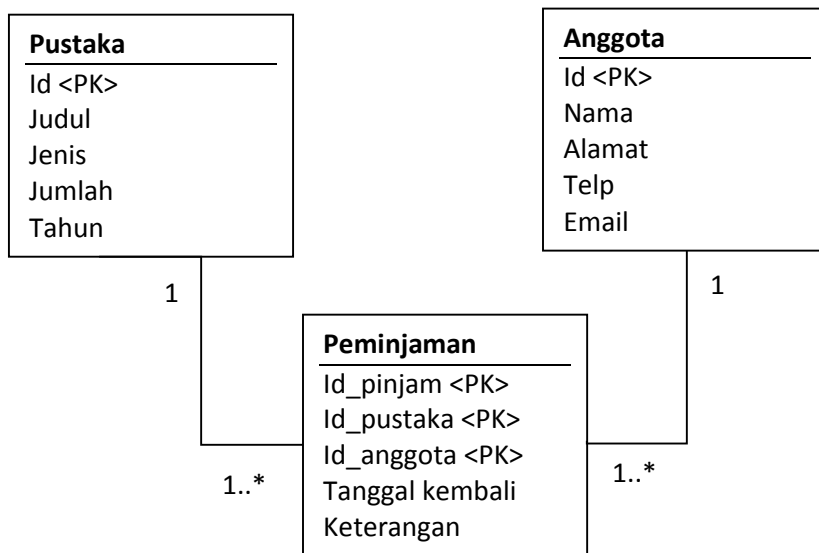
Gambar 5.3 Contoh ERD

### **Conceptual Data Model (CDM)**

CDM merupakan salah satu model konsep data yang berkaitan dengan sudut pandang user terhadap data yang disimpan di dalam *database*. CDM biasanya dibuat dalam bentuk beberapa tabel yang dilengkapi dengan beberapa atribut data berupa *field*/kolom tanpa disertai dengan tipe datanya, dan dilengkapi dengan garis yang menunjukkan bagaimana gambaran relasi antar beberapa tabel tersebut.



Gambar 5.4 menunjukkan contoh CDM dimana semua entitas lengkap dengan nama-nama atribut saja yang diperlukan, dan dilengkapi dengan relasi antara ketiga entitas tersebut.

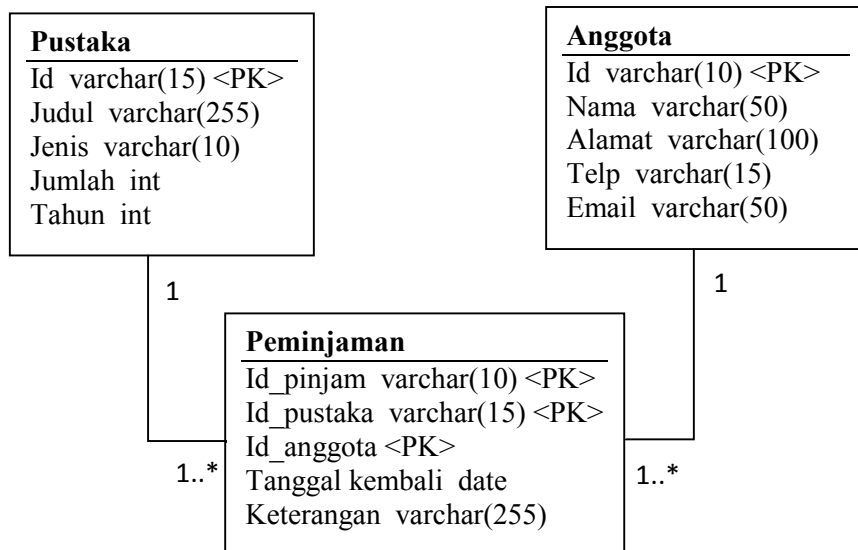


Gambar 5.4 Contoh CDM

### ***Physical Data Model (PDM)***

PDM menggunakan sejumlah tabel untuk menggambarkan data serta hubungan antar data tersebut. Setiap tabel mempunyai sejumlah *field*/kolom di mana setiap kolomnya dilengkapi dengan nama yang bersifat unik dan juga dilengkapi dengan tipe datanya masing-masing. Dengan demikian, PDM merupakan konsep data yang menerangkan secara rinci bagaimana struktur suatu data dalam bentuk tabel untuk disimpan di dalam *database*. Gambar 5.5 menampilkan contoh struktur PDM.

Adapun dalam implementasi *database*, akan terjadi perbedaan tipe atribut antara satu *Relational Database Management System* (RDBMS) dengan RDBMS yang lainnya. Perbedaan tersebut sebenarnya biasanya hanya penamaannya saja. Oleh karena itu, harus dilakukan penyesuaian tipe atribut berdasarkan pada ketentuan masing-masing RDBMS yang akan digunakan.



Gambar 5.5 Contoh PDM

## 5.4 Perancangan Database

Berdasarkan pada rancangan diagram ER selanjutnya akan dibuatkan struktur *database* dalam suatu server *database* dengan menggunakan aplikasi RDBMS tertentu, bisa yang bersifat *proprietary* (berbayar untuk mendapatkan lisensinya) ataupun yang bersifat *open source* (gratis tanpa harus mengeluarkan biaya untuk mendapatkan lisensinya). Prinsipnya semua RDBMS secara umum menggunakan bahasa pengolahan data standar yaitu *Structured Query Language* (SQL) yang memungkinkan pemrosesan data menjadi lebih sederhana dengan hanya memperhatikan apa yang akan diolah dan bagaimana hubungan antar tabel yang diolahnya<sup>[15]</sup>.

Untuk meningkatkan kinerja RDBMS dalam mengelola data salah satu caranya adalah dengan menggunakan *clustering*, fungsinya sebagai berikut:

1. Kecepatan pengaksesan data dipengaruhi oleh cara data tersebut disimpan secara fisik di dalam *database*, serta bagaimana cara diperolehnya dari *database*.
2. Menempatkan *record-record* yang memiliki kesamaan data pada posisi yang berdekatan secara fisik di dalam *harddisk*. Cara seperti ini akan

menyebabkan proses *sequence* ketika terjadi permintaan data bisa mengakibatkan proses lebih cepat.

*Clustering* sendiri terdiri dari dua tipe yaitu

1. *Intra file clustering*

*Record* data yang tersimpan di dalam satu tabel diusahakan disimpan juga pada posisi yang berdekatan secara fisik di dalam *harddisk*. Karena proses permintaan data biasanya banyak ditujukan dari satu tabel.

2. *Inter file clustering*

Menempatkan *record-record* yang ada kesamaannya dari beberapa tabel dalam suatu tempat tertentu di *harddisk*, sehingga memudahkan proses pencarian data dan berlangsung lebih cepat. Hal ini perlu dilakukan terutama untuk tabel-tabel yang terkait erat pada saat proses normalisasi.

## 5.5 Normalisasi

Dalam perencanaan arsitektur *enterprise* sebenarnya normalisasi tidak perlu dilakukan. Normalisasi perlu dilakukan ketika proses perancangan sistem informasi. Normalisasi sifatnya sangat teknis dan tujuannya adalah untuk meningkatkan kinerja penyimpanan data yang lebih efisien, dengan mengurangi sebanyak mungkin duplikasi/pengulangan yang tidak perlu dalam penyimpanan data<sup>[19, 28]</sup>.

Normalisasi perlu dilakukan untuk sistem yang sifatnya transaksional dan data yang diolahnya juga masih aktif artinya masih mengalami banyak perubahan. Efek negatif dari normalisasi adalah proses pengolahan datanya lebih kompleks karena akan lebih banyak melibatkan entitas terkait. Demikian juga waktu pemrosesan data menjadi relatif lebih lama karena adanya proses tambahan akibat dari adanya pembuatan index pada saat normalisasi dilakukan.

Proses untuk menghindari terjadinya duplikasi data dasar penyimpanan data bisa dioptimalkan. Normalisasi dilakukan ketika merancang *database*. Yang harus diperhatikan terutama terhadap entitas-entitas yang memiliki hubungan banyak ke banyak.

Normalisasi pada dasarnya adalah memisahkan kolom tabel yang harus ditulis berulang, dengan dilengkapi tabel tambahan. Artinya selama adanya

pengulangan data maka normalisasi bisa dilakukan. Tetapi, secara umum proses normalisasi terdiri dari tiga tahapan sebagai berikut:

1. Normalisasi pertama / 1-NF

Setiap data dibentuk dalam *flat file* dan tidak ada set atribut yang berulang-ulang.

Persyaratan normal pertama adalah:

- a. Pada setiap data dibentuk dalam sebuah *flat file*, data dibentuk dalam satu demi satu *record*, nilai dari *field* itu berupa "*atomic value*".
- b. Tidak ada atribut yang diset berulang (duplikat).
- c. Ditentukannya *primary key* untuk tabel / relasi.
- d. Setiap atribut hanya memiliki satu pengertian.

2. Normalisasi kedua / 2-NF

Mencari hubungan tidak langsung antar atribut. Artinya ketergantungan antar atribut dihilangkan.

Persyaratan normal kedua adalah:

- a. Bentuk datanya telah memenuhi kriteria normal pertama.
- b. Atribut bukan *primary key* harus memiliki ketergantungan fungsional penuh terhadap *primary key*.

3. Normalisasi ketiga / 3-NF

Melengkapi proses sebelum sehingga benar-benar tidak ada pengulangan dan tidak ada ketergantungan antar sesama atribut.

Persyaratan normal ketiga adalah:

- a. Bentuk datanya telah memenuhi kriteria bentuk normal kedua
- b. Atribut yang bukan *primary key* harus memiliki ketergantungan pada transitif, yaitu suatu atribut bukan *primary key* tidak boleh memiliki ketergantungan fungsional pada atribut yang bukan *primary key* lain, seluruh atribut yang bukan *primary key* pada suatu relasi hanya memiliki ketergantungan fungsional terhadap *primary key* yang direlasikan saja.

Pada dasarnya proses normalisasi terus dilakukan selama masih adanya pengulangan penyimpanan data yang tidak efisien dalam suatu tabel. Dengan demikian, semakin dilakukannya normalisasi maka akan semakin menambah

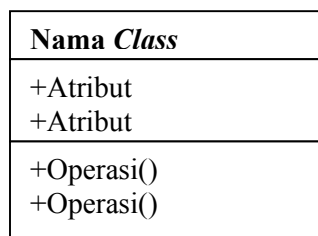
jumlah tabel yang dibutuhkan, semakin banyak keterkaitan antar tabel yang harus diperhatikan dalam pengolahan data. Tanpa disadari normalisasi merupakan proses alami yang pada saat pembuatan arsitektur data tanpa disadari sudah memperhatikan konsep normalisasi.

Normalisasi tidak selamanya diperlukan untuk setiap pengembangan suatu sistem informasi. Hal ini sangat tergantung pada jenis sistem informasinya. Sistem informasi yang mengolah data yang statis atau data yang sudah tidak terjadi perubahan lagi seperti sistem *Data Warehouse*, maka normalisasi tidak dilakukan, melainkan denormalisasi yang merupakan proses kebalikan dari normalisasi. Denormalisasi mendorong penggabungan beberapa tabel agar pemrosesan lebih sederhana dan kinerja yang dihasilkan lebih baik.

## 5.6 Diagram Class

Arsitektur data dengan menggunakan pendekatan objek (*object oriented*) dapat digambarkan dengan menggunakan diagram *class*<sup>[19, 24]</sup>. Diagram ini adalah salah satu diagram penting dalam UML untuk menggambarkan bagaimana struktur suatu sistem dilihat dari sisi hubungan antar data dalam bentuk hubungan antar *class*, walaupun diagram ini bisa juga digunakan untuk menggambarkan kebutuhan modul aplikasi.

Berbeda dengan diagram ER di dalam diagram *class* tidak dikenal istilah normalisasi walaupun hanya menggambarkan kebutuhan datanya. Simbol suatu *class* seperti pada gambar 5.6. Atribut melambangkan data yang diolah, sedangkan operasi artinya fungsi yang terkait dengan pengolahan data di dalam *class*.



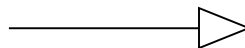
Gambar 5.6 Simbol *Class*

Penamaan suatu *class* pada dasarnya menggunakan kata benda yang mewakili berbagai objek yang sejenis, seperti *class* mobil mewakili kendaraan roda empat honda, toyota, suzuki.

Hubungan antar *class* dalam suatu sistem secara umum terdiri dari tiga jenis yaitu sebagai berikut:

1. *Inheritance*

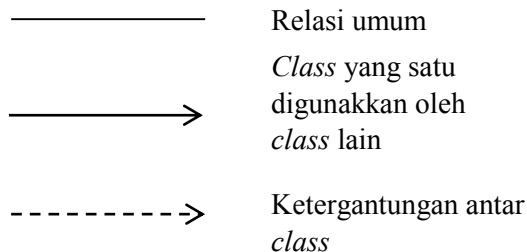
Gambar 5.7 merupakan simbol *inheritance* yaitu hubungan perwarisan antar *class*, dimana *class* mewarisi (*subclass*) akan mendapatkan atribut yang sama dengan *class* induknya (*superclass*), ditambah atribut lainnya yang diperlukan.



Gambar 5.7 *Inheritance*

2. *Association*

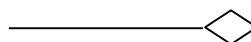
Gambar 5.8 merupakan simbol *association* yaitu hubungan yang paling dasar antar *class*, biasanya memiliki beberapa multiplicity seperti pada diagram ER.



Gambar 5.8 *Association*

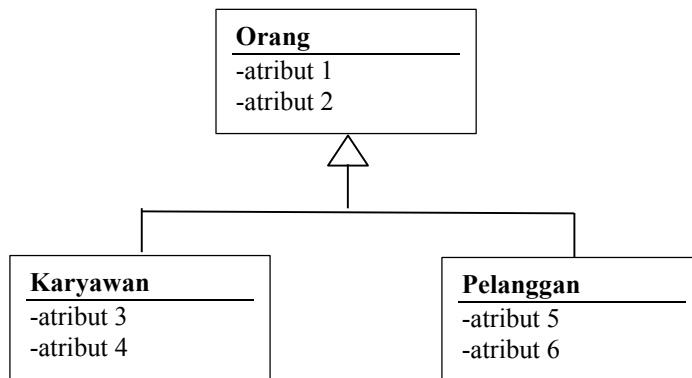
3. *Aggregation*

Gambar 5.9 merupakan simbol *aggregation* yaitu hubungan yang menunjukkan bahwa suatu *class* terdiri dari gabungan beberapa *class* lainnya.



Gambar 5.9 *Aggregation*

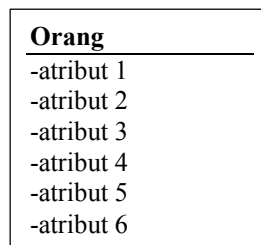
Kebutuhan data yang digambarkan dengan pendekatan objek berupa *class diagram* apabila akan diimplementasikan ke dalam struktur *database*, maka diperlukan perubahan menjadi diagram ER (*mapping*) terlebih dahulu. Permasalahan yang utama untuk perubahan ini adalah masalah *inheritance* seperti pada kasus di gambar 5.10. Proses perubahan ini bisa dilakukan dengan salah satu dari tiga cara berikut ini.



Gambar 5.10 *Inheritance Antar Class*

#### 1. *Filtered mapping*

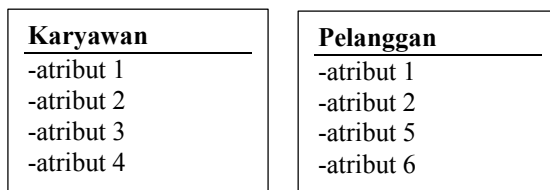
Cara ini adalah teknik yang paling sederhana, karena semua *class-class* yang terkait dalam pewarisan dibuat menjadi satu tabel, seperti contoh pada gambar 5.11.



Gambar 5.11 *Filtered Mapping*

#### 2. *Horizontal mapping*

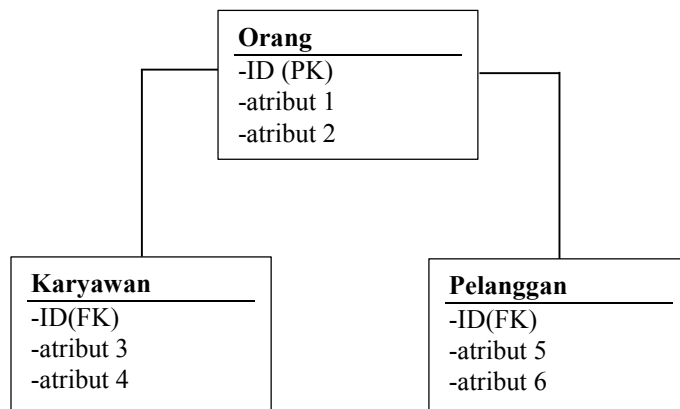
Caranya dengan membuat setiap satu turunan *class* dan *superclass* menjadi masing-masing satu tabel. Contoh penerapannya seperti pada gambar 5.12, dengan dibuat dua buah tabel dimana seluruh atribut dari tabel superclass dibuat di tabel turunannya.



Gambar 5.12 *Horizontal Mapping*

### 3. *Vertical mapping*

Caranya membuat setiap *class* yang terkait diubah ke dalam tabel masing-masing, lalu hubungan pewarisannya diganti dengan membuat hubungan relasi satu-ke-satu dari tabel pengganti *superclass* dengan setiap tabel pengganti *class* turunan. Seperti contoh pada gambar 5.13, relasi dibuat dengan memasang *primary key* (PK) pada tabel *superclass*, dan memasang *foreign key* (FK) di masing-masing *tabel class* turunan. Isi dari *field* FK harus sama (mengacu) kepada nilai dari *field* PK.



Gambar 5.13 *Vertical Mapping*

Perbandingan diantara ketiga cara tersebut dapat dilihat seperti pada tabel 5.2. Masing-masing metode memiliki kelebihan dan kekurangan, sehingga



dalam merancang dan mengimplementasikannya ke dalam *database* harus dipertimbangan *trade-off* antara efisiensi ruangan penyimpanan dan waktu *query* yang diharapkan.

Tabel 5.2 Perbandingan Antar Metode *Mapping*

Metode	Kelebihan	Kekurangan
<i>Filtered mapping</i>	<ol style="list-style-type: none"> <li>1. Kinerja query menjadi lebih cepat karena tidak memerlukan proses join antar tabel</li> <li>2. Implementasi sederhana</li> </ol>	Terdapat banyak ruang penyimpanan yang disia-siakan ( <i>field</i> yang kosong), terutama jika kelas banyak turunannya.
<i>Horizontal mapping</i>	<ol style="list-style-type: none"> <li>1. Kinerja query lebih cepat karena tidak membutuhkan join</li> <li>2. Penggunaan ruang penyimpanan data lebih efisien karena tidak ada field dibiarkan kosong</li> </ol>	Implementasi query lebih kompleks, terlebih jika terdapat perubahan rancangan (seperti penambahan atau pengurangan field)
<i>Vertical Mapping</i>	Penggunaan ruang penyimpanan lebih efisien, tidak ada field dibiarkan kosong tanpa data	Kinerja query akan turun karena harus terlebih dahulu melakukan join beberapa tabel terkait dengan tabel induknya.

## 5.7 Jenis DBMS

Dalam merencanakan implementasi pengelolaan *database* ada banyak alternatif bisa dilakukan. Sistem informasi saat ini secara standar dibuat dengan menggunakan konsep *client/server*, sehingga setiap sistem informasi memerlukan DBMS untuk pengelolaan *database* yang efektif dan efisien. Karena DBMS pada umumnya berbasis relational maka disebut pula RDBMS. RDBMS dapat menggunakan berbagai tipe *software* baik yang bersifat *open source* maupun yang *proprietary*. Secara teknis perbedaan antara RDBMS yang *open source* dan *proprietary* tidak signifikan. Seiring dengan perkembangan teknologi *software* biasa masing-masing produk selalu di-*update* dengan teknologi yang terbaru.

Beberapa alternatif RDBMS yang bisa digunakan adalah seperti yang ditunjukkan di tabel 5.3.

Tabel 5.3 Jenis RDBMS

<b>Nama DBMS</b>	<b>Tipe</b>	<b>Keterangan</b>
MySQL	<i>Open source</i>	RDBMS yang paling populer digunakan dalam pengembangan berbagai sistem informasi.
PostgreSQL	<i>Open source</i>	RDBMS yang sangat powerful sekelas Oracle, memiliki sistem sekuriti yang handal
Oracle	<i>Propietary</i>	RDBMS yang paling populer karena kehandalannya untuk melakukan pengolahan data dalam ukuran besar ( <i>big data</i> ), selain itu ditunjang dengan perlengkapan pengolahan data yang lengkap dan kinerja terjamin.
SQLServer	<i>Propietary</i>	Produk RDBMS keluaran Micorsoft, pada dasarnya kehandalannya tidak perlu dipertanyakan, merupakan saingan untuk Oracle.
Sybase	<i>Propietary</i>	RDMBS yang layak dipertimbangkan dalam pengembangan sistem informasi <i>enterprise</i> , karena kelengkapan data dan fungsi yang diperlukan dalam pemrosesan <i>database</i> .

## 5.8 Tipe Data

Untuk keperluan implementasi database dalam sebuah RDBMS, yang perlu diperhatikan adalah tipe data dari masing-masing field yang ditetapkan dalam suatu tabel. Setiap vendor database memiliki definisi masing-masing tentang tipe data. Secara umum tipe data memiliki kesamaan walaupun penamaannya sedikit terdapat perbedaan satu sama lain.

Tipe data yang biasa digunakan dalam perancangan *database* sangat tergantung pada RDBMS yang akan digunakan, beberapa tipe data diantaranya adalah seperti pada tabel 5.4

Tabel 5.4 Tipe data *database*

<b>Tipe</b>	<b>Keterangan</b>
<i>Char(n)</i>	Untuk menyimpan data dalam bentuk karakter dengan panjang karakter sebanyak n huruf. Jika tidak terisi semuanya artinya data tersebut tetap di simpan sebanyak n karakter.
<i>Varchar(n)</i>	Untuk penyimpanan data seperti char, tetapi penyimpanannya disesuaikan dengan jumlah karakternya, maksimum n karakter. Pemakainnya bisa lebih efisien diripa tipe <i>Char</i> . Di beberapa vendor yang berbeda disebut juga dengan tipe data <i>Text</i> atau BLOB
<i>Number(n)</i>	Untuk penyimpanan data dalam bentuk bukan huruf tapi dianggap sebagai bilangan yang dapat diproses secara matematika. Bilangannya bisa juga berbentuk desimal. Beberapa vendor mendefinisikannya dengan tipe data INT, <i>Numeric</i> .
<i>Date</i>	Untuk penyimpanan data tanggal yang banyak digunakan untuk proses pencatatan transaksi.
<i>Boolean</i>	Untuk penyimpanan data yang bernilai <i>logic</i> yaitu <i>true</i> atau <i>false</i> .

Dalam RDBMS modern, disediakan juga tipe data yang bisa digunakan untuk kepentingan penyimpanan data yang kompleks, seperti untuk menyimpan data jenis gambar, video, suara, dll.

## 5.9 *Constraint*

*Constraint* adalah ketentuan tertentu yang diberlakukan kepada sebuah *field*, sehingga *field* tersebut memiliki persyaratan tertentu yang harus dipenuhi agar proses penyimpanan datanya berhasil. Beberapa *constraint* yang umum berlaku di dalam RDBMS diantaranya adalah sepeti pada tabel 5.5.

Penggunaan *constraint* memang bersifat pilihan. Untuk menjang kinerja agar bisa bekerja secara optimal, maka dalam impelentasi *database* secara optimal banyak menggunakan *constraint*.

Tabel 5.5 Bererapa Jenis *Constraint*

<b>Tipe</b>	<b>Keterangan</b>
<i>Primary Key</i>	Untuk menyatakan bahwa sebuah <i>field</i> harus terisi dan tidak boleh isinya sama di seluruh tabel. Tipe ini bisa berlaku untuk satu <i>field</i> atau beberapa <i>field</i> . Setiap field yang diset primary key maka otomatis <i>not null</i> dan <i>unique</i> .
<i>Foreign Key</i>	Untuk menunjukkan relasi antar tabel dimana suatu <i>field</i> isinya harus mengacu kepada <i>field</i> di tabel induk dan memiliki nilai yang sama.
<i>Unique</i>	Untuk membatasi sebuah <i>field</i> bahwa ketika diisi datanya tidak boleh ada yang sama dalam sebuah tabel. Tetapi field masih diperbolehkan dalam keadaan kosong.
<i>Not Null</i>	Untuk pembatasan sebuah <i>field</i> supaya tidak dibiarkan kosong tanpa data.

### 5.10 Peningkatan Kinerja *Database*

Selain beberapa teknik yang telah dibahas sebelumnya, ada beberapa teknik lain yang bisa digunakan agar *database* bisa bekerja dengan lebih optimal. Teknik tersebut diantaranya adalah seperti yang dijelaskan pada tabel 5.6.

Tabel 5.6 Teknik Peningkatan Kinerja *Database*

Teknik	Keterangan
<i>Indexing</i>	<p>Pada dasarnya sebuah tabel dibaca secara <i>sequence</i> sesuai dengan urutan address penyimpanannya. Cara seperti ini tidak masalah ketika jumlah datanya masih relatif kecil. Semakin banyak data akan semakin turun kinerja pembacaan datanya. Oleh karena itu, perlu dibantu dengan index supaya pembacaan data tidak dilakukan secara <i>sequence</i>, tetapi dibantu dengan pointer yang langsung bisa menuju pembacaan yang dimaksud. Pointer-pointer ini bisa dibuat melalui index.</p> <p>Efek samping dari <i>index</i> adalah menambah proses pengolahan ketika dilakukan penambahan, perubahan dan penghapusan data, karena yang diproses selain tabel juga <i>indexnya</i>. Oleh karena itu, hanya field yang datanya sering dicari saja sebaiknya dibuat <i>index</i>.</p> <p>Setiap <i>field</i> yang diset sebagai <i>primary key</i>, secara otomatis akan dibuat <i>indexnya</i>.</p>
<i>Clustering</i>	<p>Salah satu cara untuk meningkatkan kinerja pencarian data terutama waktu dilakukan <i>query</i>. Cara ini pada dasarnya akan menyimpan data secara fisik di tempat penyimpanannya pada posisi yang berdekatan, sehingga bisa mengurangi waktu pencarian secara <i>sequence</i>.</p> <p>Ada dua cara yang bisa dilakukan secara clustering, yaitu sebagai berikut:</p> <ol style="list-style-type: none"> <li>1. <i>Intra-file clustering</i> Setiap baris data di dalam tabel yang mirip isinya disimpan berdekatan secara fisik di tempat penyimpanan.</li> <li>2. <i>Inter-file clustering</i> Menggabungkan dua buah tabel yang berrelasi menjadi sebuah tabel dan setiap baris data yang mirip isinya disimpan berdekatan.</li> </ol>

# Bab 6

## ARSITEKTUR APLIKASI

---

### 6.1 Pengertian Arsitektur Aplikasi

Arsitektur aplikasi pada dasarnya menggambarkan daftar kandidat aplikasi apa saja yang berdasarkan survey memang diperlukan oleh *enterprise*. Aplikasi dibutuhkan untuk mendukung proses bisnis yaitu untuk melaksanakan pemrosesan data yang telah teridentifikasi pada arsitektur data. Dengan demikian arsitektur aplikasi merupakan penghubung antara arsitektur bisnis dan arsitektur data<sup>[5, 13, 25]</sup>.

Disamping itu juga, arsitektur aplikasi harus menggambarkan bagaimana hubungan antara kandidat aplikasi yang ada sehingga membentuk satu kesatuan sistem yang terintegrasi dengan menjauhi permasalahan duplikasi aplikasi, hal ini akan mengakibatkan semakin sulitnya proses pemeliharaan aplikasi jika terjadi perubahan, karena aplikasi yang sama yang berada di sistem yang berbeda harus dikelola berulang-ulang.

Untuk mengembangkan keseluruhan aplikasi sistem yang dibutuhkan oleh *enterprise* secara efisien, maka perlu digunakan pendekatan konsep onjek terutama konsep *reusable component*, dimana komponen yang digunakan suatu aplikasi bisa digunakan dengan mudah pada aplikasi lain yang membutuhkannya. Karena kenyataannya dalam suatu arsitektur aplikasi yang terintegrasi akan sangat besar kemungkinan dibutuhkan modul pemrograman yang sama sehingga selain untuk efisiensi dan penyederhanaan proses pemeliharaan program, akan lebih baik modul pemrograman tersebut dibuat dalam bentuk *reusable component* yang bisa digunakan oleh aplikasi apa saja yang membutuhkannya<sup>[12, 17]</sup>.

Dalam pendekatan objek, alat bantu yang bisa digunakan untuk menggambarkan hubungan antara aplikasi dalam suatu *enterprise* adalah *component diagram* dan *package diagram*. Kedua diagram ini bermanfaat untuk menyusun arsitektur aplikasi yang lebih detail.

Ada lima tahapan dasar untuk menyusun arsitektur aplikasi yaitu sebagai berikut.

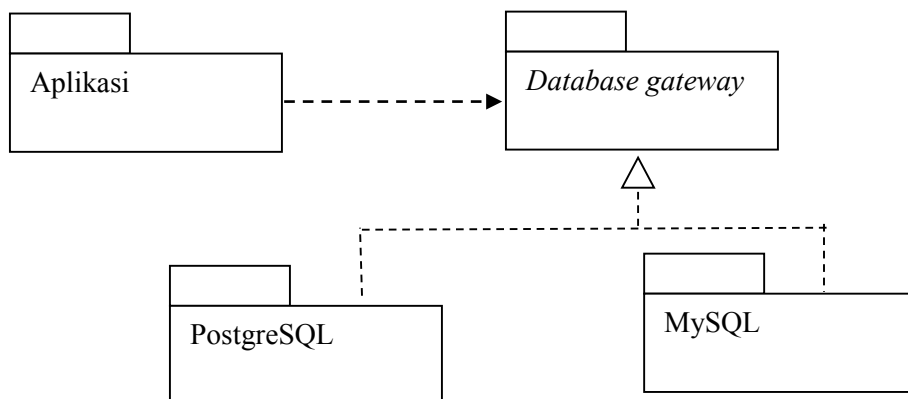
1. Membuat daftar kandidat aplikasi.
2. Mendefinisikan aplikasi yang diperlukan pengelolaan data
3. Merelasikan antara aplikasi dan fungsi bisnis.
4. Melakukan analisis dampak dari aplikasi yang ada.
5. Menyebarkan hasil dari arsitektur aplikasi.

Dalam tahapan analisis arsitektur *enterprise* sebagai hasilnya adalah sebagai berikut:

1. Daftar aplikasi  
Tabel yang berisi sistem-sistem informasi apa saja yang dibutuhkan oleh *enterprise* secara global dan menyeluruh.
2. Matriks aplikasi  
Tabel yang berisi matriks yang menghubungkan antara arsitektur bisnis dan arsitektur data. Matriks ini akan menggambarkan cakupan setiap aplikasi terhadap data dan proses bisnis yang terkait.

## 6.2 Diagram *Package*

Dalam pendekatan berorientasi objek *class* merupakan satu terkecil dari suatu aplikasi. Sebuah aplikasi bisa terdiri dari satu atau banyak *class* tergantung pada kompleksitas aplikasinya masing-masing. Semakin kompleks sistem semakin kompleks aplikasinya, dan semakin banyak *class* dibutuhkan untuk mengimplementasikannya.



Gambar 6.1 Diagram *Package*

*Package* adalah pengelompokan dari beberapa *class* dalam rangka pengelolaan *class* secara sistematis. Sebuah *package* bisa terdiri dari *class* dan bisa juga secara hirarkis terdiri dari beberapa *package* yang lebih rinci. Struktur *package* yang baik adalah yang dapat menunjukkan aliran ketergantungan yang jelas. Gambar 6.1 menunjukkan contoh penggunaan diagram *package*.

Pengelompokkan *class* ke dalam *package* bisa didasarkan pada dua cara yaitu

1. *Common closure principle*  
*Class-class* dalam *package* seharusnya berubah karena kesamaan alasan misalkan kesamaan fungsi.
2. *Common reuse principle*  
*Class-class* dalam *package* seharusnya digunakan secara bersama-sama.

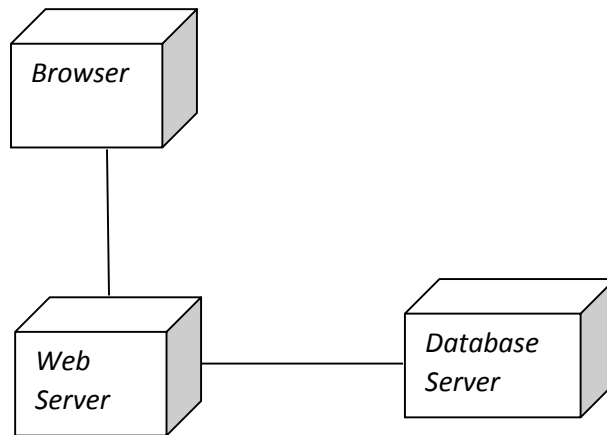
*Package* diagram berguna untuk menunjukkan paket aplikasi sistem dan saling ketergantungan diantara paket aplikasi tersebut. Manfaat utama dari diagram ini terutama dalam suatu aplikasi sistem informasi dalam skala *enterprise* untuk mendapatkan gambaran saling ketergantungan diantara komponen utama sistem.

### 6.3 Diagram *Deployment*

Diagram ini seperti pada gambar 6.2 menjelaskan tentang tata letak sebuah aplikasi sistem, serta menunjukkan bagian-bagian aplikasi yang berjalan pada bagian-bagian *hardware*.

Dalam UML, kubus menunjukkan *node* yang diberi nama dan ditambahkan *stereotype* untuk mengindikasikan tipe resource yang ada di dalamnya. Sebuah aplikasi sistem terdiri dari beberapa *node* yang digambarkan berupa kubus. Jika *node* adalah bagian dari *package* maka namanya bisa mengandung nama *package* tersebut. Kubus dari sebuah *node* bisa juga ditambahkan dengan kompartement yang berisi informasi seperti komponen yang disimpan di *node* tersebut. Jalur komunikasi antar *node* ditunjukkan dengan garis dan bisa ditambah dengan label yang menginformasikan protokol komunikasi yang dipakai.





Gambar 6.2 Diagram *Deployment*

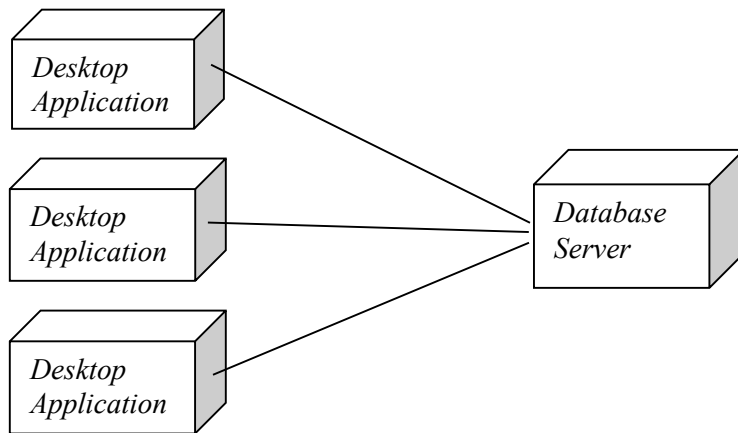
#### 6.4 Arsitektur *Client/Server*

Perkembangan teknologi informasi mendorong terjadinya perubahan konsep penerapan aplikasi dari sentralisasi di dalam satu server menjadi desentralisasi komponen aplikasi berupa bagian *client* dan bagian server.

Arsitektur aplikasi dengan menggunakan konsep *client/server* bisa dibagi menjadi beberapa tipe yaitu

1. Model *Two Tiers*

Konsep *client/server* yang paling sederhana dimana ada sisi *client* dan sisi server. Model seperti ini umumnya diterapkan untuk aplikasi yang tidak berbasis web atau sering disebut dengan *desktop application*, dimana aplikasi pengolah datanya dipasang di setiap *client* sedangkan sebagai servernya berupa *database server*, seperti yang ditunjukkan pada gambar 6.3



Gambar 6.3 Model *Two Tiers*

Model ini umumnya diterapkan untuk kepentingan aplikasi yang bersifat lokal atau hanya untuk kepentingan internal saja dengan memanfaatkan fasilitas intranet.

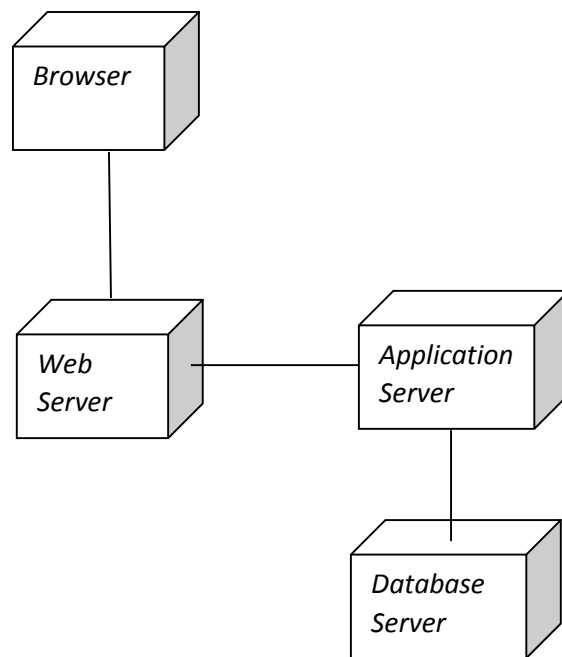
2. Model *Three Tiers*

Pada dasarnya server dipisah lebih spesifik sesuai dengan fungsinya masing-masing, tujuannya untuk lebih meringankan beban kerja setiap server, sehingga kinerja sistem secara keseluruhan bisa ditingkatkan.

Model ini banyak digunakan dengan memanfaatkan web sebagai basisnya, sehingga *client* cukup menggunakan *browser* untuk mengakses ke aplikasi yang disediakan oleh *web server* untuk mengolah data yang diperlukan oleh sistem.

3. Model *Multi Tiers*

Model lebih luas dari model three tiers, dimana biasanya program-program yang bersifat *logical program* disimpan terpisah dalam suatu server khusus yang disebut dengan *application server*. Untuk aplikasi yang menunjang sistem informasi yang kompleks model seperti ini bisa diterapkan. Terutama apabila sebagian *client* yang mengakses sistemnya berbasis *mobile* yang memiliki *resource* terbatas seperti *smart-phone*. Model ini juga bisa mengurangi beban *web server* karena bagian *application* di *web server* dipisahkan ke dalam *application server* seperti yang ditunjukkan pada gambar 6.4.



Gambar 6.4 Model *Multi-Tiers*

Tergantung pada pemrograman yang digunakan, khususnya yang berbasis teknologi java implementasi model *multi tiers* ini bisa menggunakan *entity bean*. Implementasi model ini bisa juga menggunakan *web service* yang biasanya untuk kepentingan kolaborasi antar sistem yang berbeda dan tidak tergantung pada bahasa pemrograman tertentu.

### 6.5 Model *View Controller* (MVC)

*Framework* yang populer sehingga banyak digunakan dalam pengembangan aplikasi sistem informasi. *Framework* ini memisahkan sistem menjadi tiga komponen utama sebagai berikut:

#### 1. *Model*

Bagian aplikasi yang khusus berkaitan dengan masalah pengelolaan data yang tersimpan dalam suatu *database*. Bagian akan melakukan berbagai proses pengolahan data/informasi berupa menambah, merubah, menghapus, dan memilah-milah data sesuai keperluan aplikasi.

2. *View*

Bagian aplikasi yang berkaitan langsung dengan pengguna. Bagian ini adalah bagian yang bertanggungjawab secara khusus untuk mengurus tampilan atau *interface* yang menghubungkan antara pengguna dan aplikasi sistem informasi.

3. *Controller*

Bagian yang merupakan penghubung antara dua bagian sebelumnya yaitu model dan view. Bagian ini biasanya berisi logika program dari suatu aplikasi sistem, menerima aksi yang dijalankan oleh pengguna dan menyalurkannya menjadi proses apa yang harus dilaksanakan terkait dengan data apa yang harus diproses lebih lanjut, dan data apa yang harus dikirimkan ke pihak pengguna untuk ditampilkan oleh bagian *view*.

Pembuatan arsitektur aplikasi dengan menggunakan konsep *framework* MVC bisa menghasilkan arsitektur sistem yang sistematis karena dilakukan pembagian komponen berdasarkan pada karakteristik fungsi yang harus dijalanannya.

# Bab 7

## ARSITEKTUR TEKNOLOGI

---

### 7.1 Infrastruktur Jaringan

Arsitektur aplikasi pada dasarnya menggambarkan daftar kandidat aplikasi yang diperlukan untuk memproses data yang teridentifikasi pada arsitektur data, dan ditujukan untuk mendukung proses bisnis.

Variasi infrastuktur jaringan komputer saat ini sangat banyak baik yang berbasis kabel maupun *wireless*. Sebagai *backbound* nya biasanya tetap menggunakan jaringan berbasis kabel, sedangkan untuk menambah akses poin dari pengguna menggunakan *wireless* karena lebih efisien dari sisi pemasangannya.

Faktor keamanan sistem merupakan hal yang terpenting dalam suatu infrastruktur jaringan. Oleh karena itu, harus dipasang beberapa sistem keamanan baik secara *software* maupun *hardware*, sehingga berbagai risiko *security hole* bisa diantisipasi sebaik mungkin. Walaupun demikian, sistem keamanan yang terlalu ketat akan menyebabkan kenyamanan pengguna dikorbankan. Perencanaan infrastuktur jaringan dengan mempertimbangkan faktor keamanan dan kenyamanan merupakan *trade-off* yang harus dilakukan dengan perhitungan yang benar, agar kebutuhan akses terhadap jaringan khususnya internet dapat berjalan dengan maksimal.

Langkah-langkah yang dilakukan dalam menyusun arsitektur teknologi adalah sebagai berikut:

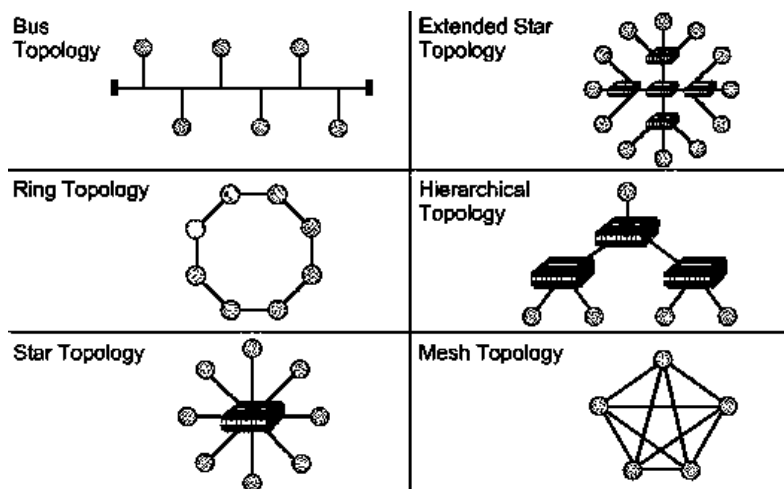
1. Mengidentifikasi *platform* teknologi dan prinsip-prinsipnya
2. Menentukan *platform* teknologi dan pendistribusian data dan aplikasi yang terkait
3. Merelasikan *platform* teknologi terhadap aplikasi dan fungsi bisnis yang telah ditetapkan

Penyusunan arsitektur teknologi yang paling utama adalah bagaimana menentukan dan menyusun keperluan *platform* teknologi jaringan yang layak

diterapkan dalam mendukung kepentingan arsitektur lainnya bisnis, data dan aplikasi.

Penggunaan infrastruktur jaringan komputer berbasis kabel bisa dilakukan dengan menggunakan pilihan berbagai topologi jaringan, diantaranya seperti yang ditunjukkan pada gambar 7.1.

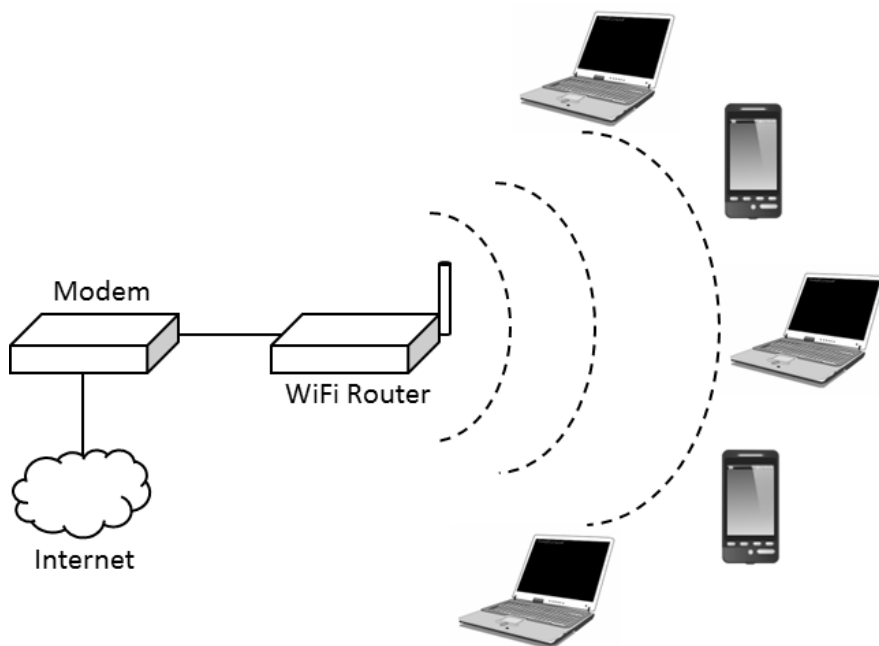
1. Topologi *Bus*
2. Topologi *Ring*
3. Topologi *Star*
4. Topologi *Hierarchy/Tree*
5. Topologi *Mess*



Gambar 7.1 Topologi Jaringan

## 7.2 Wireless Network

Perkembangan teknologi informasi khususnya teknologi jaringan telah berkembang dari berbasis kabel menjadi berbasis tanpa kabel (*wireless*), walaupun dalam hal kecepatan yang *wireless* relatif lebih lambat dibandingkan dengan yang *wire*.



Gambar 7.2 *Wireless Network Menggunakan Wifi*

Oleh sebab itu, biasanya jaringan *wireless* lebih bersifat pendukung untuk kepentingan di dalam suatu ruangan atau area tertentu. Kelebihan dari *wireless* adalah dalam pemasangan infrastrukturnya karena tidak memerlukan kabel sehingga bisa lebih fleksibel, mudah dan tidak memakan ruangan yang ada.

Seperti yang ditunjukkan pada gambar 7.2 penggunaan *wireless* semakin menunjang kebutuhan akses terhadap internet dengan menggunakan perangkat komputer jenis apa saja baik *personal computer* (PC), *laptop*, *tablet*, maupun *smartphone*, karena semua perangkat tersebut sudah mendukung akses internet via *WiFi*. Dengan demikian pengaksesan terhadap aplikasi sistem informasi bisa dilakukan dengan banyak pilihan.

Teknologi *WiFi* merupakan *icon* dalam penggunaan infrastruktur jaringan berbasis *wireless*. Dengan demikian untuk pemasangan jaringan internal atau intranet pilihan *wireless* dengan menggunakan teknologi *WiFi* menjadi pilihan yang tepat.

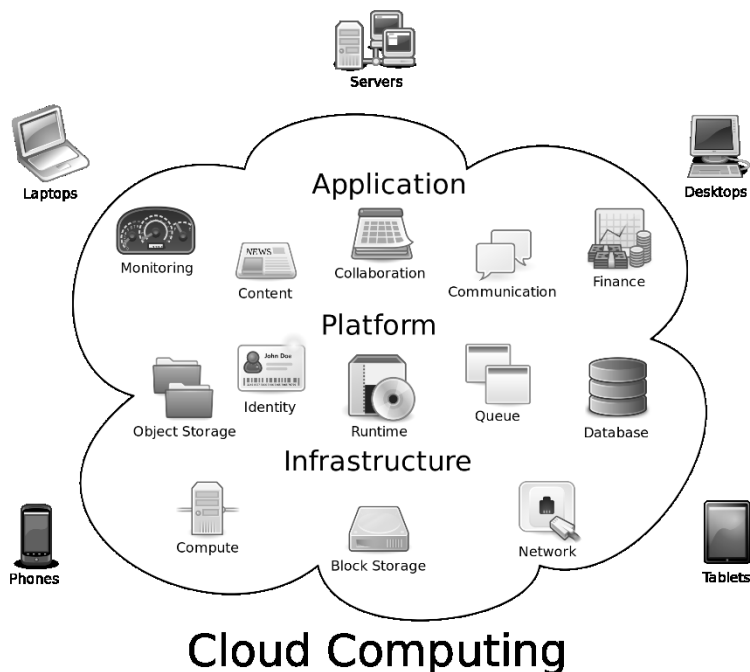
Selain teknologi *WiFi* sebenarnya ada teknologi *wireless* lainnya untuk jangkauan lebih luas yaitu teknologi *WiMax*. Tetapi, karena faktor tertentu (kebijakan pemerintah dalam penggunaan frekuensi) teknologi ini pada saat ini

tidak bisa diterapkan. Teknologi ini sebenarnya sangat penting untuk mempermudah akses internet di wilayah yang berbentuk kepulauan seperti Indonesia.

### 7.3 Cloud Computing

Perkembangan teknologi informasi di bidang infrastruktur jaringan yang menawarkan *bandwidth* yang lebar berkecepatan tinggi dan harga relatif terjangkau. Disamping itu semakin kompleksnya proses *maintenance* untuk perangkat infrastruktur *hardware* dan *software*, mendorong munculnya konsep baru *cloud computing* sebagai suatu fasilitas alternatif pada era saat ini dalam mengimplentasikan aplikasi sistem informasi berbasis web dan internet<sup>[34]</sup>.

Ide awal *cloud computing* sebenarnya dimulai dari istilah *network computing* sebagai konsep baru pengganti *desktop computing*, sehingga pengguna pengguna tidak memerlukan berbagai *software*, mulai dari sistem operasi dan berbagai *software* lainnya untuk diinstalasi di dalam komputer pengguna.



Gambar 7.3 Cloud Computing



Dengan demikian, *cloud computing* bisa dikatakan sebagai gabungan antara pemanfaatan teknologi komputer dan pengembangan sistem berbasis internet. Istilah *cloud* yang sering digunakan dalam diagram jaringan komputer, melambangkan abstraksi dari infrastruktur rumit yang disembunyikan.

Namun demikian, pertimbangan *cloud computing* dalam mendukung arsitektur teknologi harus dikaji dengan teliti, karena *cloud computing* sangat tergantung pada konsisi jaringan komputer yang stabil dan *bandwidth* yang mencukupi agar bisa mendukung penggunaan aplikasi dan pengolahan data secara normal.

Seperti yang ditunjukkan pada gambar 7.3, *cloud computing* menerapkan suatu metode komputasi yang disajikan dalam bentuk layanan/service sehingga pengguna dapat mengakses layanan tersebut melalui internet, dan tidak perlu mengetahui apa yang terdapat di dalamnya dan bagaimana pengendalian di dalamnya. Kata kunci dalam pemanfaatan *cloud computing* adalah tersedianya infrastruktur jaringan yang memadai (kecepatan akses atau *bandwidth*) bagi pengguna untuk bisa menggunakan berbagai aplikasi *software* yang diperlukan.

Secara umum layanan *cloud computing* terbagi menjadi tiga kategori yang ditawarkan oleh *Internet Service Provider* (ISP), yaitu:

1. *Infrastructure as a Service* (IaaS)

IaaS merupakan layanan yang menyediakan atau menyewakan sumberdaya teknologi informasi yang paling dasar. Pengguna tidak perlu mengetahui komputer apa dan bagaimana cara ISP menyediakan layana IaaS. Dengan demikian pengguna IaaS biasanya *enterprise* yang ingin mengembangkan sendiri aplikasi sistem informasi yang dibutuhkannya tanpa harus memiliki berbagai kebutuhan ruangan maupun *hardware* yang penunjang sistem tersebut.

IaaS adalah layanan komputasi awan yang menyediakan infrastruktur teknologi informasi berupa CPU, RAM, *storage*, *bandwith* dan konfigurasi lain. Komponen-komponen tersebut digunakan untuk membangun komputer virtual. Komputer virtual dapat diinstal sistem operasi dan aplikasi sesuai kebutuhan.

Keuntungan layanan IaaS ini adalah tidak perlu membeli komputer fisik sehingga lebih menghemat biaya. Konfigurasi komputer virtual juga bisa diubah sesuai kebutuhan. Misalkan saat *storage* hampir penuh, *storage* bisa ditambah dengan segera. Perusahaan yang

menyediakan IaaS adalah Amazon EC2, TelkomCloud dan BizNetCloud.

## 2. *Platform as a Service (PaaS)*

PaaS adalah layanan yang menyediakan *computing platform*. Biasanya sudah terdapat sistem operasi, *database*, *web server* dan *framework* aplikasi agar dapat menjalankan aplikasi yang telah dibuat. Perusahaan yang menyediakan layanan tersebutlah yang bertanggung jawab dalam pemeliharaan *computing platform* ini.

Keuntungan layanan PaaS ini bagi pengembang adalah mereka bisa fokus pada aplikasi yang mereka buat tanpa memikirkan tentang pemeliharaan dari *computing platform*. Contoh penyedia layanan PaaS adalah Amazon Web Service dan Windows Azure.

Kekurangan dari PaaS pengembangan aplikasi hanya dibatasi oleh platform yang memang disediakan oleh ISP, artinya pengguna dipaksa untuk menyesuaikan dengan platform yang ada dalam pengembangan aplikasinya.

## 3. *Software as a Service (SaaS)*

SaaS adalah layanan komputasi awan dimana kita bisa langsung menggunakan aplikasi yang telah disediakan. Penyedia layanan mengelola infrastruktur dan *platform* yang menjalankan aplikasi tersebut. SaaS memberikan kemudahan bagi pengguna untuk bisa memanfaatkan *software* dengan cara berlangganan tanpa perlu mengeluarkan investasi untuk *in house development* maupun pembelian lisensi. Contoh layanan aplikasi email yaitu gmail, yahoo dan outlook sedangkan contoh aplikasi media sosial adalah twitter, facebook dan google+.

Pengguna dapat langsung mengakses berbagai fitur yang disediakan oleh ISP dengan cara berlangganan via web. Dengan SaaS ini pengguna tidak memiliki kendali penuh atas aplikasi yang disewanya, tetapi hanya mengendalikan fitur-fitur aplikasi yang telah disediakan oleh ISP saja. Beberapa aplikasi ada yang mengharuskan pengguna untuk berlangganan agar bisa mengakses aplikasi yaitu Office 365 dan Adobe Creative Cloud. Sedangkan contoh penggunaan SaaS yang gratis adalah Google Docs dimana pengguna bisa menggunakan berbagai aplikasi untuk keperluan kantor seperti menulis dokumen, presentasi, dll.

Dengan penggunaan tiga jenis layanan tersebut para pengguna bisa fokus pada proses bisnisnya masing-masing tanpa harus memikirkan bagaimana pengelolaan infrastruktur baik maintenance maupun sistem keamanannya, dimana kedua hal tersebut merupakan pekerjaan yang cukup kompleks dan memerlukan penanganan yang profesional.

Beberapa konsep dalam *cloud computing* yang harus dipenuhi adalah sebagai berikut:

1. Layanan bersifat *on demand* artinya pengguna dapat berlangganan sesuai dengan kebutuhan dan hanya membayar untuk yang digunakannya saja.
2. Layanan bersifat elastis/*scalable* artinya pengguna bisa menambah atau mengurangi jensi dan kapasitas layanan yang dibutuhkan. Sistem selalu bisa mengakomodasi perubahan kebutuhan tersebut.
3. Layanan sepenuhnya dikelola oleh provider atau penyedia internet, yaitu yang dibutuhkan oleh pengguna hanyalah seperangkat komputer personal (PC, laptop, *desktop*, *tablet*, *smartphone*) dengan koneksi ke internet.

# Bab 8

## *Framework* Arsitektur

---

### 8.1 Pengertian *Framework*

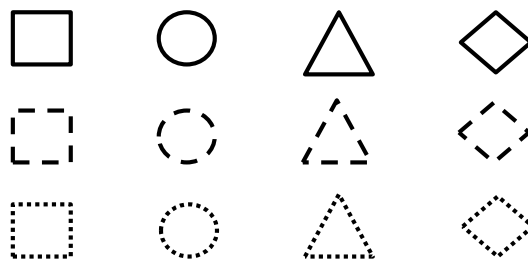
Dalam dunia teknologi informasi dikenal istilah *framework* yang mengandung pengertian yang berbeda. Pengertian *framework* dalam ruang lingkup arsitektur *enterprise*, sebagaimana yang didefinisikan oleh FEAF, adalah sebagai berikut:

*“a logical structure for classifying and organizing complex information.”*

Dalam perencanaan arsitektur *Enterprise* penggunaan *framework* merupakan hal yang penting, jika tidak maka akan menyebabkan dokumen-dokumen (artifak) yang dihasilkan terpisah-pisah (bersifat *stand-alone*) tidak terorganisir dan tidak jelas klasifikasinya. Akibatnya dokumen yang dihasilkan kurang memiliki makna untuk siapa dokumen tersebut bisa digunakan. Oleh karena itu, *framework* sangat berperan dalam pengorganisasian dokumen-dokumen hasil pengembangan sistem informasi dalam skala *enterprise*, sebagai alat kontrol untuk mengarahkan sistem-sistem informasi yang terdokumentasi dengan baik, sehingga memudahkan proses pemeliharaan sistem informasi *enterprise* secara keseluruhan. Dengan demikian, secara konsep seperti ditunjukkan pada gambar 8.1 bagaimana berbagai macam artifak dihasilkan dan dikelompokkan berdasarkan keterkaitan satu sama lain. *Framework* pada intinya mengatur pengelompokan produk-produk hasil penyusunan arsitektur *enterprise* (dokumen, laporan, grafik, diagram, dll) berdasarkan pada kategori tertentu dan mengatur hubungan antar kelompok tersebut.

Dalam pembahasan *framework* tidak dibahas bagaimana metodologi atau tahapan-tahapan yang harus dilakukan untuk menghasilkan semua artifak arsitektur dalam suatu *framework*. Tetapi lebih fokus pada pembahasan tentang apa yang harus dihasilkan dalam pengembangan arsitektur *enterprise*, dan bagaimana taxonomi dari artifak yang dihasilkan tersebut.

Tujuan dari pengelompokkan berdasarkan *framework* adalah untuk mempermudah dalam mengukur sejauh mana suatu metodologi menghasilkan produk-produk arsitektur *enterprise* yang lengkap. Disamping itu, *framework* bermanfaat untuk menyediakan informasi produk arsitektur *enterprise* yang disesuaikan dengan kebutuhan pihak-pihak terkait dengan pengembangan arsitektur *enterprise* berdasarkan pada peranannya dalam proyek arsitektur *enterprise*, sehingga melalui produk tersebut yang bersangkutan dapat memahami sejauh mana kelengkapan arsitektur yang harus dihasilkan.



Gambar 8.1 Konsep Dasar Framework Arsitektur *Enterprise*

*Framework* bisa menjadi alat bantu bagi tim pengembang untuk bekerjasama dalam menghasilkan suatu arsitektur *enterprise* disesuaikan dengan tugasnya masing-masing. *Framework* Zachman pertama kali dipublikasikan oleh John A. Zachman pada tahun 1987, dan kemudian dikembangkan lebih lanjut pada tahun 1992, dengan tujuan untuk menyediakan struktur dasar organisasi yang mendukung akses, integrasi, interpretasi, pengembangan, pengelolaan, dan perubahan perangkat arsitektural dari suatu sistem informasi berskala *enterprise*.

## 8.2 Framework Zachman

Zachman merupakan *framework* arsitektur *enterprise* yang menyediakan suatu cara bagaimana memandang dan mendefinisikan sebuah *enterprise* secara formal dan terstruktur. *Framework* ini pada dasarnya merupakan sebuah matriks dua dimensi untuk mengklasifikasi komponen-komponen arsitektur yang dibangun dengan mengkombinasikan beberapa pengelompokan. Pada awalnya Zachman hanya mendefinisikan tiga kolom saja yang sifatnya teknis yaitu *What*, *How*, dan *Where*. Kemudian Zachman mengembangkan lebih luas bahwa ternyata keberhasilan dari pengembangan suatu sistem informasi

*enterprise* memerlukan pandangan lain yang bersifat non-teknis tetapi sangat penting yaitu berupa *Who*, *When*, dan *Why*<sup>[32]</sup>.

*Framework Zachman* pada dasarnya bukan merupakan sebuah metodologi penyusunan arsitektur *enterprise*, karena *framework* ini tidak membahas tentang bagaimana metode dan proses secara spesifik untuk mengumpulkan, mengelola dan menggunakan informasi yang dituliskan pada Kerangka kerja tersebut. Oleh karena itu, *framework Zachman* lebih tepat merupakan salah satu alat untuk melakukan pengelolaan artifak arsitektur *enterprise* (dokumen perancangan, spesifikasi, model, dll) agar lebih memperjelas kepada siapa target dari masing-masing artifak tersebut (misalnya *planner*, *owner*, *developer*, dan lain-lain), dan isu utama apa yang terdapat pada artifak tersebut.

Implementasi *framework Zachman* dalam berbagai hal bisa diterapkan, diantaranya:

1. Kerangka kerja untuk mengorganisasi dan menganalisis data.
2. Kerangka kerja untuk arsitektur *enterprise*
3. Sistem klasifikasi atau skema klasifikasi
4. Matriks dalam bentuk 6x6.
5. Model dua dimensi atau model analitis

Setiap baris pada *Framework Zachman* mewakili tingkat abstraksi yang digunakan untuk melakukan analisis sistem.

1. *Scope* (ruang lingkup) lapisan abstraksi paling tinggi, diwakili dari ide-ide dan konsep-konsep idealistik.
2. Model *enterprise* menggambarkan tingkat konseptualitas, dimana pemodelan awal dilakukan untuk mendefinisikan konsep bisnis yang mengimplementasikan ruang lingkup.
3. Model sistem adalah tingkat dimana objek-objek yang konseptual diubah menjadi struktur-struktur logik .
4. Model Teknologi mendefinisikan obyek secara fisik yang akan mewakili struktur-struktur logik .
5. Representasi detail, lapisan ini terdiri dari implementasi-implementasi penuh dari spesifikasi secara fisik untuk setiap kategori .

Dengan demikian, setiap sel yang didefinisikan oleh interaksi dari tingkat abstraksi dengan lapisan aktivitas *enterprise*, akan memiliki berbagai arti dan isi berdasarkan subyek *framework* yang digunakan. *Framework Zachman* secara lengkap seperti yang ditunjukkan pada gambar 8.2.

<i>Layer</i>	<i>What (Data)</i>	<i>How (Function)</i>	<i>Where (Network)</i>	<i>Who (People)</i>	<i>When (Time)</i>	<i>Why (Motivation)</i>
<i>Scope Context Boundary (Planner)</i>	List of things important to the business	List of processes the business performs	List of locations in which the business operates	List of organizations important to the business	List of events significant to the business	List of business goals/ strategies
<i>Business Model Concepts (Owner)</i>	e.g., Semantic or Entity-Relationship Model	e.g., Business Process Model	e.g., Business Logistics System	e.g., Work Flow Model	e.g., Master Schedule	e.g., Business Plan
<i>System Model Logic (Designer)</i>	e.g., Logical Data Model	e.g., Application Architecture	e.g., Distributed System Architecture	e.g., Human Interface Architecture	e.g., Processing Structure	e.g., Business Rule Model
<i>Technology Model Physics (Builder)</i>	e.g., Physical Data Model	e.g., System Design	e.g., Technology Architecture	e.g., Presentation Architecture	e.g., Control Structure	e.g., Rule Design
<i>Component Configuration (Implementer)</i>	e.g., Data Definition	e.g., Program	e.g., Network Architecture	e.g., Security Architecture	e.g., Timing Definition	e.g., Rule Specification
<i>Functioning Enterprise Instances (Worker)</i>	e.g., Data	e.g., Function	e.g., Network	e.g., Organization	e.g., Schedule	e.g., Strategy

Gambar 8.2 *Framework Zachman*

Seiring dengan perkembangan teknologi *software*, pengembangan arsitektur *enterprise* juga mengadopsi pendekatan objek. Oleh karena, artifak-artifak pada *framework* Zachman dapat disesuaikan dengan pendekatan dengan berorientasi objek.

Pengembangan arsitektur *enterprise* berbasis *framework* Zachman bisa digunakan untuk menggambarkan arsitektur *enterprise* berkaitan dengan tujuan yang ingin dicapai. Model *framework* Zachman memberikan banyak sudut pandang sehingga memudahkan komunikasi dari seluruh *stakeholder enterprise* untuk memahami apa yang ada dalam *enterprise* (sumber daya, tujuan maupun perencanaan). *Framework* Zachman terbentuk dari matrik dua dimensi yang berisi enam baris dan enam kolom. Adapun sejauh mana artifak harus dihasilkan tentunya sangat tergantung pada tujuan pengembangan arsitektur *enterprise* itu sendiri apakah berupa perencanaan, perancangan, atau sampai dengan implementasi.

Keseluruhan baris dari enam baris pada *framework* Zachman merepresentasikan enam *perspective* (perspektif/sudut pandang) dimana pada prinsipnya suatu prespektif yang lebih bawah merupakan penjelasan terhadap

perspektif yang di atasnya, artinya menghasilkan artifak-artifak yang lebih rinci dan lebih implementatif.

1. Perencana (*Planner*)

Sudut pandang perencana berkaitan dengan segala aktifitas dalam menetapkan latar belakang, lingkup dan tujuan dari sistem informasi *enterprise*. Pada perspektif ini di definisikan arah dan tujuan bisnis *enterprise*.

2. Pemilik (*Owner*)

Sudut pandang pemilik berkaitan dengan penerima atau pemakai produk/jasa akhir dari *enterprise*. Pada perspektif ini digambarkan model-model terkait dengan kebutuhan bisnis, produk, jasa dari pemilik.

3. Perancang (*Designer*)

Sudut pandang perancang berkaitan dengan aktifitas perantara antara apa yang diinginkan pemilik dan apa yang dapat dicapai secara teknis dan fisik. Pada perspektif ini dibutuhkan suatu model yang dapat menggambarkan secara detail proses bisnis yang ada. Rancangan sistem bisa berupa rancangan umum dan rancangan detail sistem yang akan dikembangkan. Pada perspektif ini digambarkan model teknis perancangan yang menjadi dasar pedoman implementasi sistem informasi *enterprise* yang akan dibangun.

4. Pembangun (*Builder*)

Sudut pandang pembangunan arsitektur sistem informasi *enterprise* berdasarkan model rancangan yang telah ditetapkan. Pada perspektif ini digambarkan model teknis perancangan aplikasi dan data yang lebih realistis untuk pembuatan sistem informasi *enterprise*.

5. Pelaksana (*Integrator*)

Sudut pandang subkontraktor berkaitan dengan model-model yang menggambarkan bagian-bagian yang akan dimasukkan ke dalam produk akhir atau produk jadi dari berbagai elemen dalam suatu sistem informasi *enterprise*.

6. Pengguna (*user*)

Perspektif implementasi dari suatu sistem informasi *enterprise*. Perspektif pengguna dari suatu sistem yang berkaitan dengan hasil akhir berupa produk *software* dari sistem yang telah dikembangkan, termasuk antarmuka *software* yang turut menentukan sejauh mana *software* digunakan sebagai pendukung proses bisnis.



Sedangkan keenam kolom masing-masing menggambarkan jenis artifak yang khas dan berbeda satu sama lainnya yaitu sebagai berikut:

1. *What*

Kolom ini fokus pada entitas data, dan memberi gambaran tentang kebutuhan *enterprise* terhadap informasi yang dibuat dari sekumpulan data. Disamping itu, kolom ini juga menunjukkan hubungan antar entitas data, bagaimana efisiensi penyimpanan data, serta teknologi *database* apa yang bisa diterapkan dalam mendukung pengelolaan data menjadi informasi yang berkualitas.

2. *How*

Kolom ini fokus pada proses bisnis yang berjalan atau prosedur-prosedur yang berlaku untuk menjalankan roda bisnisnya sesuai dengan visi/misi dan tujuan yang ditetapkan. Kolom ini juga memberi gambaran tentang uraian fungsional terhadap komponen sistem informasi *enterprise*.

Kolom ini sangat penting dalam suatu *enterprise* karena merupakan rangkaian proses yang saling terkait dari bagian hulu sampai bagian hilir dari keseluruhan proses bisnis suatu *enterprise*. Bagian awal dari kolom ini yang akan menentukan model sistem informasi seperti apa yang benar-benar diperlukan untuk mencapai visi/misi *enterprise*. Selain sistem informasi disesuaikan dengan proses bisnis yang berjalan, sistem informasi juga harus didefinisikan dalam berbagai tipe TPS, MIS, DSS, atau EIS yang disesuaikan dengan tingkat manajerial memerlukannya.

3. *Where*

Kolom ini fokus pada *node-node* dan *link-link*, dan memberi gambaran bagaimana sebaran arus informasi dan proses dalam sebuah *enterprise*. Pembahasan juga mencakup keseluruhan area bisnis *enterprise* yang harus diselesaikan, dan bagaimana menyediakan kelengkapan infrastruktur seperti apa yang tepat bisa diterapkan dalam menunjang sistem informasi *enterprise*.

4. *Who*

Kolom ini fokus pada siapa yang berkontribusi terhadap pekerjaan yang terkait, dan menggambarkan bagaimana hubungan antara pekerjaan dan struktur tanggung jawab, serta otoritas dalam suatu *enterprise*.

5. *When*

Kolom ini fokus pada waktu dan siklus, dan digunakan untuk mendesain relasi dari serangkaian kejadian (*event-to-event*) yang menetapkan kriteria kinerja dan tingkatan kuantitatif untuk *resource enterprise*.

6. *Why*

Kolom ini fokus pada sasaran dan tujuan serta strategi atau metode bagaimana pengelolaan suatu sistem informasi *enterprise* bisa berjalan sesuai tujuan semula.

Walaupun *framework* Zachman merupakan *framework* arsitektur *enterprise*, tetapi *framework* ini tidak menunjuk suatu metodologi tertentu. *Framework* Zachman hanyalah model untuk mengorganisasikan artifak arsitektur *enterprise*, tujuannya adalah dalam rangka membantu para pengembang dan manajer mengisolasi masalah dan mengatur apa yang harus dibereskan dalam membangun suatu arsitektur *enterprise*.

*Framework* Zachman merupakan alat ukur untuk mengukur sejauh mana suatu metodologi menghasilkan kelengkapan artifak dalam pengembangan arsitektur *enterprise*. Sebagai contoh, pengembangan arsitektur *enterprise* dengan menggunakan EAP jika diterapkan kepada *framework* Zachman, maka artifak-artifak yang dihasilkan tidak akan mencakup keseluruhan *framework* Zachman, tetapi hanya mencakup enam sel saja yaitu dua baris pertama (*planner, owner*) dan tiga kolom pertama (*what, how, where*). Hal ini, karena EAP hanya bersifat khusus untuk perencanaan arsitektur *enterprise* saja, sehingga jika dipetakan ke dalam Zachman secara perspektif hanya mencakup dua baris pertama saja. Sedangkan kajian dari EAP fokus pada empat elemen utama dari suatu arsitektur *enterprise*, sehingga hanya mencakup tiga kolom pertama.

Pengembangan arsitektur *enterprise* dengan menggunakan *framework* seharusnya dilakukan secara berkelanjutan yang berbasis pada pendekatan per bagian/modul. Artinya memecah proyek arsitektur *enterprise* menjadi beberapa modul proyek. Dengan demikian, *enterprise* bisa fokus pada modul proyek yang dianggap lebih penting terlebih dulu sesuai dengan skala prioritas yang disepakati.

Beberapa alasan umum yang melatarbelakangi banyaknya penggunaan *framework* Zachman dalam pengembangan arsitektur *enterprise* adalah sebagai berikut :

1. Relatif sederhana karena hanya memiliki dua dimensi yang mudah untuk dipahami.

2. Kedua dimensi tersebut mengarahkan *enterprise* ke dalam cara yang komprehensif dalam mengelola arsitektur baik secara individu maupun departemen.
3. Menggunakan bahasa non teknis yang membantu orang untuk mudah mengerti dan berkomunikasi secara lebih tepat.
4. Dapat digunakan untuk mengelompokkan dan membantu memahami isu-isu yang kompleks.
5. Membantu menyelesaikan permasalahan desain dan fokus terhadap detil tanpa kehilangan jalur secara keseluruhan.

### 8.3 Framework TOGAF

TOGAF merupakan salah satu *framework* yang sudah banyak digunakan dalam pengembangan arsitektur *enterprise*. Hal ini dikarenakan TOGAF memiliki beberapa fitur yang tidak dimiliki oleh Zachman diantaranya selain menggunakan pendekatan objek, juga memperhatikan aspek lain selain empat elemen dasar arsitektur *enterprise* seperti analisis *gap*, tatakelola dan manajemen perubahan<sup>[7, 18, 21]</sup>.

*Framework* TOGAF seperti dijelaskan pada gambar 8.3, memiliki cukup banyak artifak-artifak baik berupa dokumen, gambar, diagram, matriks, dll yang harus diperhatikan dalam pengembangan arsitektur *enterprise* untuk menunjang kesempurnaan hasil akhir suatu arsitektur *enterprise*.

Metodologi TOGAF yang disebut dengan TOGAF *Architecture Development Method* (ADM), meliputi beberapa tahapan yang bersifat iteratif yaitu sebagai berikut:

1. *Preliminary Phase: Framework & Principles*

Mempersiapkan organisasi untuk kesuksesan proyek arsitektur *enterprise*, menentukan prinsip, framework dan *tool* yang akan digunakan.

2. *Requirements Management*

Pengelolaan kebutuhan sistem informasi *enterprise* yang akan dibangun, dengan meyakinkan bahwa setiap tahapan pengembangan arsitektur *enterprise* benar-benar didasarkan pada kebutuhan bisnis yang dijalankan.

3. *Phase A: Architecture Vision*

Mengatur ruang lingkup, batasan, dan harapan proyek, membuat visi pengembangan arsitektur *enterprise*, menentukan *stakeholder*, validasi konteks bisnis, dan membuat pernyataan pekerjaan arsitektur, dan meyakinkan bahwa visi *enterprise* sesuai dengan visi arsitektur *enterprise*.

4. *Phase B: Business Architecture*

Menganalisis proses bisnis yang dilakukan secara keseluruhan oleh *enterprise* dan menggambarkan proses bisnis secara visual agar mudah dipahami oleh berbagai pihak yang berkepentingan. Dengan demikian bisa diketahui dengan jelas siapa saja dan terlibat apa dalam pelaksanaan proses bisnis *enterprise*.

5. *Phase C: Information System Architectures*

Mengembangkan dua arsitektur utama sistem informasi yaitu arsitektur data dan arsitektur aplikasi yang dibutuhkan berdasarkan pada proses bisnis yang berjalan.

6. *Phase D: Technology Architecture*

Mengembangkan infrastruktur teknologi jaringan komputer yang merupakan landasan utama dalam mengimplementasikan sistem informasi *enterprise*.

7. *Phase E: Opportunitis & Solutions*

Mengevaluasi dan memilih opsi implementasi sebagai target pengembangan arsitektur *enterprise*, mengidentifikasi implementasi proyek yang besar kepentingannya. Setiap kasus pengembangan arsitektur *enterprise* selalu terjadi adanya perbedaan akibat perubahan dari arsitektur *baseline* (“*as is*”) menjadi arsitektur target (“*to be*”) sehingga perlu dilakukan analisis *gap* nya. Termasuk perlu dilakukannya risk assesment terhadap berbagai kendala yang akan menghambat pengembangan arsitektur *enterprise*.

8. *Phase F: Migration Planning*

Menganalisis *cost benefit* dan risiko, mengembangkan daftar prioritas proyek berdasarkan kepentingan dasar implementasi, serta membuat rencana migrasi.

### 9. Phase G: Implementation Governance

Mempersiapkan dan menerbitkan kontrak arsitektur, meyakinkan bahwa implementasi didasarkan pada arsitektur *enterprise* yang benar dengan didukung tata kelola teknologi informasi untuk memastikan bahwa sistem informasi bisa diterapkan sebagaimana mestinya.

### 10. Phase H: Architecture Change Management

Melakukan proses *monitoring* berkelanjutan untuk meyakinkan bahwa pengembangan arsitektur merespon kebutuhan *enterprise* secara dinamis.

Sejauh mana proses pengulangan metodologi TOGAF-ADM ini dilakukan, hal ini tergantung pada sejauh mana tingkat rincian pengembangan yang dimaksud, apakah arsitektur *enterprise* itu berupa perencanaan, perancangan, atau sampai dengan implementasi.

Preliminary				Architecture Vision					
Architecture Principle				Business Strategy	Technology Strategy		Business Principles, Objectives, and Drivers	Architecture Vision	Stakeholders
Architecture Requirements									
Requirements			Constrains			Assumptions		Gaps	
Business Architecture						Information System Architecture		Technology Architecture	
						Data	Application		
Motivation						Data Entities	Information System Services	Platform Services	
Drivers	Goals	Objectives	Measures						
Organization						Logical Data Components	Logical Application Components	Logical Technology Components	
Organization Units	Locations		Actors, Roles						
Function						Physical Data Components	Physical Application Components	Physical Technology Components	
Business Services, Contracs, Service Qualities		Processes, Events, Controls, Products		Functions					
Architecture Realization									
Opportunities, Solutions,and Migration Planning						Implementation Governance			
Capabilities		Work Packages		Architecture Contracs		Standars	Guidelines	Specifications	

Gambar 8.3 Framework TOGAF

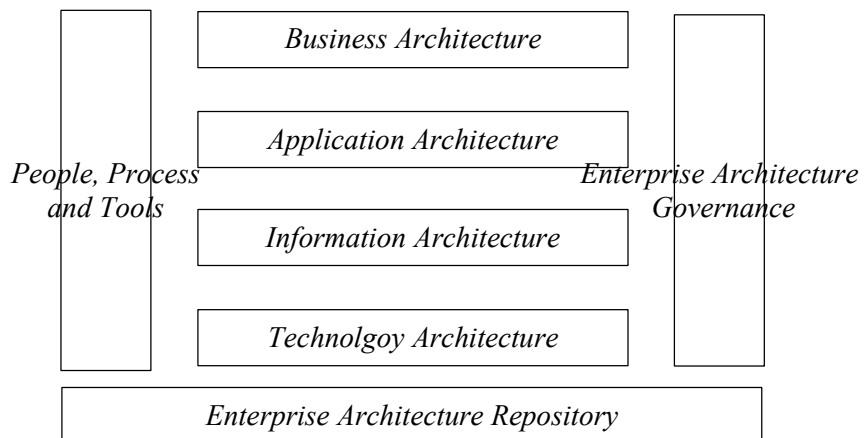
## 8.4 Framework Oracle

Sebagai vendor *database* Oracle mengembangkan juga produk *software* sejenis *Enterprise Resource Planning* (ERP). Dalam perkembangannya Oracle memperkenalkan salah satu *framework* yang sederhana, tetapi memiliki keunikan dalam pengembangan arsitektur *enterprise*.

Pada dasarnya *framework* Oracle bentuknya seperti Zachman, tetapi tidak memetakan ke dalam perspektif tertentu. Oracle lebih memusatkan pada apa yang harus dihasilkan artifaknya, minimal empat komponen dasar, dan dilengkapi dengan komponen lainnya seperti tata kelola *enterprise*, stakeholder, proses dan alat-bantu yang dibutuhkan untuk efisiensi pengembangan.

*Framework* Oracle ditunjukkan seperti pada gambar 8.4. Sedangkan tahapan dalam metodologi Oracle yang disebut dengan istilah Oracle *Architecture Development Process*, terdiri dari enam tahapan yang berlangsung secara iteratif sebagai berikut:

1. *Business context*
2. *Architecture vision*
3. *Current state*
4. *Future state*
5. *Roadmap*
6. *Governance*



Gambar 8.4 *Framework* Oracle

## 8.5 Framework FEAF

*Federal Enterprise Architecture Framework* (FEAF) pada intinya merupakan *framework* yang lebih sederhana daripada *framework* Zachman, atau merupakan sub-*framework* Zachman, seperti yang ditunjukkan pada gambar 8.5<sup>[16]</sup>.

FEAF seperti EAP lebih fokus pada empat komponen utama dari suatu arsitektur *enterprise*, karena merupakan pondasi yang paling harus diperhatikan dalam pengembangan suatu arsitektur *enterprise*.

*Framework* FEAF terdiri dari 15 sel yang menghasilkan artifak-artifak yang berbeda meliputi arsitektur data, arsitektur aplikasi, dan arsitektur teknologi, serta disesuaikan dengan lima sudut pandang (perspektif) sistem yang ada di *framework* Zachman, yaitu

1. *Planner*
2. *Owner*
3. *Designer*
4. *Builder*
5. *Subcontractor*

Data Architecture	Application Architecture	Technology Architecture
List of Business Objects	List of Business Processes	List of Business Locations
Semantic Model	Business Process Model	Business Logistics System
Logistic Data Model	Applications Architecture	System Geographic Deployment Architecture
Physical Data Model	Systems Design	Technology Architecture
Data Directory	Program	Network Architecture

Gambar 8.5 *Framework* FEAF

Tahapan-tahapan iteratif yang secara umum dilakukan dalam pengembangan arsitektur *enterprise* dengan menggunakan *framework* FEAF adalah sebagai berikut:

1. *Obtain Executive Buy-In and Support*  
Proses mendapat legalitas pekerjaan dan dukungan penuh dari manajemen puncak.
2. *Establish Management Structure and Control*  
Membuat struktur manajemen dan kontrol untuk pengendalian pekerjaan.
3. *Define an Architecture Process and Approach*  
Menentukan pendekatan dan metode yang paling sesuai digunakan dalam pengembangan.
4. *Develop Baseline Enterprise Architecture*  
Membuat kondisi arsitektur *enterprise* saat ini baik bisnis, data, aplikasi, dan teknologi.
5. *Develop Target Enterprise Architecture*  
Membuat target arsitektur *enterprise* yang akan dicapai disesuaikan dengan kebutuhan di masa datang.
6. *Develop Sequencing Plan*  
Menyusun rencana pengembangan sampai dengan implementasi sistem informasi *enterprise*.
7. *Use the Enterprise Architecture*  
Memanfaatkan dan menjalankan proses pengembangan arsitektur *enterprise* sebagaimana yang direncanakan.
8. *Maintain the Enterprise Architecture*  
Memelihara masterplan dari arsitektur *enterprise* disesuaikan dengan perubahan-perubahan yang bisa saja terjadi.

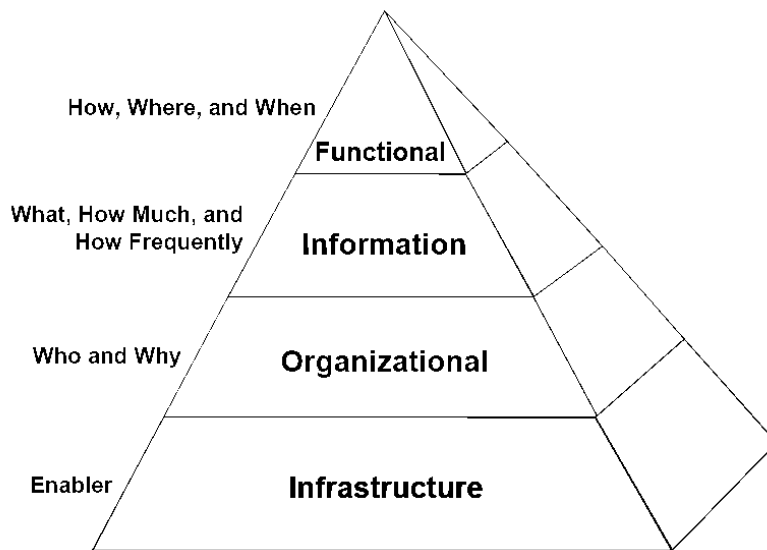
## 8.6 Framework TEAF

*Treasury Enterprise Architecture Framework* (TEAF) merupakan *framework* yang mengatur pengelompokan artifaknya hampir sama dengan



*framework* FEAF<sup>[16]</sup>. Seperti yang ditunjukkan pada gambar 8.6 artifak-artifak di dalam TEAF bisa dikelompokkan menjadi empat bagian yaitu

1. *Functional* (*how, where, dan when*)
2. *Informational* (*what,, how much, dan how frequently*)
3. *Organizational* (*who dan why*)
4. *Infrastructure* (*enabler*)



Gambar 8.6 *Framework* TEAF

Sedangkan, secara keseluruhan pengelompokan selnya lebih banyak jika dibandingkan dengan *framework* FEAF. TEAF meliputi empat baris perspektif dan empat kolom jenis seperti ditunjukkan pada gambar 8.7.

Adapun ke empat perspektif tersebut adalah sebagai berikut:

1. *Planner*  
Mendefinisikan artifaknya berupa daftar repositori arsitektur *enterprise*
2. *Owner*  
Mendefinisikan artifaknya berupa model level atas yang lebih rinci dari *planner*.

### 3. *Designer*

Mendefinisikan artifaknya berupa model rancangan logis yang rinci dan menjadi landasan implementasi.

### 4. *Builder*

Mendefinisikan artifak arsitektur *enterprise* berupa model rancangan fisik

<i>Functional View</i>	<i>Informational View</i>	<i>Organizational View</i>	<i>Infrastructure View</i>
<i>Mission &amp; Vision Statements</i>	<i>Information Dictionary</i>	<i>Organization Chart</i>	<i>Technical Reference Model</i>  <i>Standards Profile</i>
<i>Activity Model</i>  <i>Info Assurance Trust Model</i>	<i>Information Exchange Matrix (Conceptual)</i>	<i>Node Connectivity Description (Conceptual)</i>	<i>Info Assurance Risk Assesment</i>  <i>System Interface Description (level 1)</i>
<i>Business Process/ System Function Matrix</i>  <i>Event Trace Diagrams</i> <i>State Charts</i>	<i>Information Exchange Matrix (Logical)</i> <i>Logical Data Model</i> <i>Data CRUD Matrices</i>	<i>Node Connectivity Description (Logical)</i>	<i>System Interface Description (Level 2 &amp; 3)</i>
<i>System Functionality Description</i>	<i>Information Exchange Matrix (Physical)</i>  <i>Physical Data Model</i>	<i>Node Connectivity Description (Physical)</i>	<i>System Interface Description (Level 4)</i> <i>System Performance Parameters Matrix</i>

Gambar 8.7 *Framework TEAF*

## 8.7 *Framework DoD C4ISR*

*Framework* ini seperti ditunjukkan pada gambar 8.8<sup>[16]</sup>, dimana memiliki artifak-artifak yang secara garis besar dibagi ke dalam tiga *view* yaitu sebagai berikut:

### 1. *Operational view*

Menerangkan tugas dan aktivitas, elemen operasional, dan aliran informasi yang diperlukan untuk melaksanakan atau mendukung kegiatan operasional.

2. *Systems view*

Mengidentifikasi sistem-sistem yang bisa mendukung kebutuhan.

3. *Technical view*

Menentukan kriteria tatakelola implementasi dari setiap kemampuan sistem.

All View	Operational View	Systems View	Technical View
Overview & Summary Information	High-level Concept of Operation Graphic	Systems Interface Description	Technical Architecture Profile
Integrated Dictionary	Node Connectivity Description		
	Information Exchange Matrix		
	Command Relationship Chart  Activity Model  Operational Rules Model  State Transition Description  Event Trace Diagrams  Logical Data Model	Systems Communications Description  Systems Matrix  System Functionality Description  Operational Activity to System Function Traceability Matrix  System Performance Parameters Matrix  System Technology Forecast  Systems Rules Model  System State Transition  Systems Event Trace Diagrams  Physical Data Model	Standards Technology Forecast

Gambar 8.8 *Framework DoD C4ISR*

## Daftar Pustaka

1. Ambler Scott W. Nalbone John, Vizdos Michael J., 2005, The *Enterprise Unified Process: Extending the Rational Unified Process*, Prentice Hall PTR
2. Booch Grady, Maksimchuk R. A, dll, 2007, *Object-Oriented Analysis and Design with Applications Third Edition*, Addison-Wesley Publishing
3. Dhewanto Wawan, Falahah, 2007, *ERP Menyelaraskan Teknologi Informasi dengan Strategi Bisnis*, Penerbit Informatika
4. Fowler Martin, 2004, *UML Distilld , 3th Ed., A Brief Guide to the Standard Object*, Pearson Education, Inc.
5. Godinez Mario,Hechler Eberhard, Koenig Klaus, dll, 2010, *The Art of Enterprise Information Architecture*, IBM Press
6. Hamdani Dadi, 2014, *Model Enterprise Architecture Untuk Sekolah Menengah Atas Menggunakan Metode Enterprise Unified Process*, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
7. Harion Rachel, 2009, *TOGAF™ 9 Foundation Study Guide*, Van Haren Publishing
8. Hardiana Bella, 2013, *Pembuatan Enterprise Architecture di P2I-LIPI Bandung Menggunakan EAP*, Tesis Pasca Sarjana Sistem Informasi Unikom
9. Herdiana Rudi Bambang, 2014, *Perancangan Arsitektur Sistem Informasi Asuransi Kesehatan Berbasis Platform As a Service Menggunakan Framework Zachman*, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
10. Havey Mike, 2005, *Essential Business Process Modeling*, O'Reilly
11. Jogianto H. M., 2006, *Analisis dan Desain Sistem Informasi: Pendekatan Terstruktur Teori dan Praktek Aplikasi Bisnis*, Penerbit Andi
12. Juric Matjaz B., Loganathan Ramesh, Sarang Poornachandra, Jennings Frank, 2007, *SOA Approach to Integration*, Packt Publishing
13. Laplante Philip A., 2007, *What Every Engineer Should Know About Software Engineering*, CRC Press
14. Lam Wing, Shankaraman, 2007, *Enterprise Architecture and Integration*, Information Science Reference
15. Laudon Kenneth C., Laudon Jane P., 2012, *Management Information Systems (Managing the Digital Firm)*, Pearson Education, Inc.
16. Minoli Daniel, 2008, *Enterprise Architecture A to Z*, CRC Press
17. Nagappan Ramesh, Skoczylas Robert, Sriganesh Rima Patel, *Developing Java Web Services*, Wiley Publishing, Inc.
18. Maulana Firmansyah, 2015, *Perancangan Arsitektur Enterprise Untuk Pembuatan Blueprint Teknologi Informasi*, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
19. Munawar, 2005, *Pemodelan Visual*, Penerbit Graha Ilmu

20. Neves Gabriel, 2013, Pemanfaatan *Enterprise Architecture Planning* Sebagai Alat Perencanaan dan Pemodelan Proses Bisnis Guna Mendukung Sistem Informasi pada Institute of Business (IOB) Dili Timor-Leste, Tesis Pasca Sarjana Sistem Informasi Unikom
21. Prasetyo Tri Ferga, 2015, Perancangan Arsitektur Sistem Informasi Rumah Sakit Kabupaten Menggunakan Framework TOGAF, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
22. Rifai Mochamad Farid, 2011, Perencanaan Arsitektur *Enterprise* Perguruan Tinggi dengan Menggunakan *Enterprise Unified Process*, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
23. Rizky Soetam, 2011, Konsep Dasar Rekayasa Perangkat Lunak, Penerbit Prestasi Pustaka
24. Rosenberg Doug, Scott Kendall, 2001, Applying Use Case Driven Object Modeling with UML, Addison-Wesley Publishing
25. Roshen Waseem, 2009, SOA-BASED *Enterprise* Integration: A Step by Step Guide to Services-Based Application Integration, Mc Graw Hill
26. Rumapea Sri Agustina, 2010, Analisis Proses Bisnis Pada Distributor Xyz Menggunakan Tools Pemodelan Idef0, Seminar Nasional Aplikasi Teknologi Informasi (SNATI 2010)
27. Rumapea Sri Agustina, Surendro Kridanto, 2007, Perencanaan Arsitektur *Enterprise* Penyelenggaraan Pelayanan Terpadu Satu Pintu, Seminar Nasional Aplikasi Teknologi Informasi (SNATI 2007)
28. Salahuddin M., A. S. Rossa, 2011, Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek), Penerbit Modula.
29. Setiawan Ridwan, 2015, Perancangan Arsitektur *Enterprise* E-STTG Menggunakan TOGAF ADM, Tesis Pasca Sarjana Sistem Informasi STMIK-LIKMI
30. Sobiki Petrus, 2015, Perencanaan Arsitektur Sistem Informasi Perguruan Tinggi Menggunakan Metode *Enterprise Unified Process*, Tesis Pasca Sarjana Sistem Informasi Unikom
31. Spewak Steven H., Hill Steven C., 1992, *Enterprise Architecture Planning*, John Wiley & Sons, Inc.
32. Surendro Kridanto, 2009, Pengembangan Rencana Induk Sistem Informasi, Penerbit Informatika
33. Surendro Kridanto, 2009, Implementasi Tata Kelola Teknologi Informasi, Penerbit Informatika
34. Tim Elcon, 2012, Cloud Computing, Penerbit Andi
35. Theuerkorn Fenix, 2004, LightWeight *Enterprise* Architectures, Auerbach Publications
36. Yunis Roni, Surendro Kridanto, 2009, Perancangan Model *Enterprise* Architecture dengan TOGAF Architecture Development Method, Seminar Nasional Aplikasi Teknologi Informasi (SNATI 2009)

### **Biodata Penulis**

Nama Lengkap : Ana Hadiana  
Tempat/Tanggal Lahir : Tasikmalaya/29 Mei 1968  
Pendidikan : Tahun 1993, 1995 menyelesaikan program S1 dan S2 di bidang Teknik Informatika, Fukui University Jepang. Tahun 2004 menyelesaikan program S3 di bidang Teknik Informatika dari Shinsyu University Jepang.  
Kajian Penelitian : Information System, Kansei Engineering, Software Engineering