




BAB:04

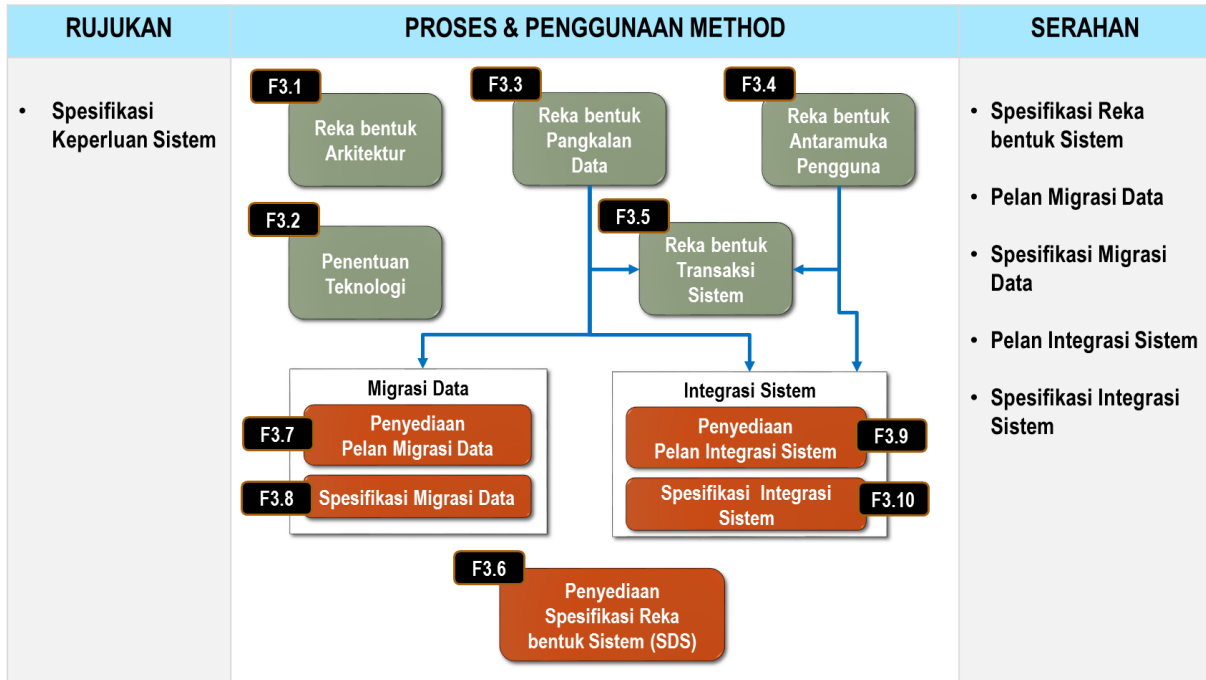
FASA REKABENTUK

Bab ini menerangkan berkenaan dengan aktiviti-aktiviti penyediaan Arkitektur Sistem, Rekabentuk Pangkalan Data, Rekabentuk Antara Muka Pengguna termasuk Pemetaan Data, Rekabentuk Transaksi Sistem, Rekabentuk Migrasi dan Rekabentuk Integrasi berdasarkan kepada maklumat keperluan sistem yang telah dikenalpasti.



4 FASA REKA BENTUK

4.1 Gambaran Keseluruhan



Rajah 56 : Gambaran Keseluruhan Fasa 3 - Reka bentuk

4.2 Pengenalan

Fasa reka bentuk adalah fasa bagi merancang penyelesaian masalah dan ekspetasi yang dinyatakan dalam **D03 Spesifikasi Keperluan Sistem**. Fasa ini adalah langkah permulaan untuk ترجمahkan dari domain masalah kepada domain penyelesaian iaitu alihan daripada 'Apa?' kepada 'Bagaimana?'. Reka bentuk sistem adalah faktor yang paling kritikal yang akan menjejaskan kualiti perisian dan mempunyai kesan yang besar kepada aktiviti pembangunan/pembinaan sistem. Fasa Reka bentuk Sistem menggariskan 7 aktiviti utama iaitu:

- Reka bentuk Arkitek
- Penentuan Teknologi
- Reka bentuk Pangkalan Data
- Reka bentuk Antaramuka Pengguna
- Reka bentuk Transaksi Sistem
- Migrasi Data
- Integrasi Sistem

Dokumen Rujukan kepada Fasa Reka bentuk adalah **D03 Spesifikasi Keperluan Sistem**.

Dokumen Serahan kepada Fasa Reka bentuk adalah seperti berikut:

- a) **D04 Spesifikasi Reka bentuk Sistem**
- b) **D05 Pelan Migrasi Data**
- c) **D06 Spesifikasi Migrasi Data**
- d) **D07 Pelan Integrasi Sistem**
- e) **D08 Spesifikasi Integrasi Sistem**

4.3 Penglibatan Pemegang Taruh

Pemegang Taruh utama yang patut terlibat dalam reka bentuk sistem adalah Juruanalisa Sistem. Juruanalisa sistem perlulah berkeupayaan untuk menterjemahkan keperluan sistem yang didokumen kepada logik pembangunan untuk menghasilkan sistem yang berkualiti dan memenuhi kehendak pelanggan.

Cadangan penglibatan kategori pemegang taruh adalah seperti berikut:

- a) Juruanalisis Sistem yang membangunkan reka bentuk
- b) SME untuk menyemak reka bentuk fungsian, data dan antaramuka sistem

4.4 Faktor Kejayaan

Untuk memastikan aktiviti reka bentuk berjaya dilaksanakan, faktor kejayaan utama yang perlu dipertimbangkan sebelum dan semasa aktiviti dilaksanakan adalah seperti berikut:

- a) **D03 Spesifikasi Keperluan Sistem** perlu mendapat pengesahan Pemilik Sistem.
- b) Pasukan Reka bentuk Sistem mempunyai kompetensi dan kemahiran untuk menterjemahkan keperluan sistem kepada reka bentuk.
- c) **D04 Spesifikasi Reka bentuk Sistem** disemak dan mendapat pengesahan pemegang taruh.

4.5 Reka bentuk Arkitektur [F3.1]

Pengenalan

Reka bentuk arkitektur adalah penyusunan dan pengaturan struktur-struktur bagi sesuatu sistem yang ingin dibangunkan. Reka bentuk arkitektur merupakan hubungan yang kritikal di antara reka bentuk dan kejuruteraan keperluan, di mana penyediaannya bertujuan untuk mengenal pasti komponen-komponen berstruktur yang utama di dalam sistem serta hubungan-hubungan di antara setiap komponen tersebut. Reka bentuk arkitektur adalah penting untuk memenuhi keperluan fungsian dan juga bukan fungsian oleh kerana impaknya kepada prestasi, keteguhan (*robustness*), pengagihan (*distributability*) dan kebolehsenggaraan sistem aplikasi.

Lapisan-lapisan bisnes, maklumat/data, aplikasi dan teknologi yang terkandung di dalam arkitektur *enterprise* boleh dijadikan sebagai input dan rujukan semasa penyediaan reka bentuk arkitektur yang dikehendaki. Output kepada proses reka bentuk arkitektur adalah arkitektur perisian yang terdiri daripada arkitektur perisian sistem aplikasi, arkitektur aplikasi dan arkitektur data. Arkitektur yang akan dihasilkan ini menerangkan bagaimana sesuatu sistem disusun atur sebagai set komponen yang saling berkomunikasi di antara satu sama lain.

Arkitektur Sistem Aplikasi

Arkitektur sistem aplikasi merupakan struktur bagi satu-satu sistem yang merangkumi komponen-komponen perisian, sifat-sifat (*properties*) komponen berkenaan serta hubungan di antara satu komponen dengan komponen-komponen yang lain. Arkitektur sistem aplikasi boleh diperincikan kepada sub-sub arkitektur seperti berikut :

- a) Arkitektur Aplikasi
- b) Arkitektur Pangkalan Data

Objektif

- o Menyediakan arkitektur sistem aplikasi yang terdiri daripada arkitektur keseluruhan sistem aplikasi, arkitektur aplikasi dan arkitektur pangkalan data berpandukan kepada keperluan-keperluan yang diperolehi di dalam fasa permulaan projek dan fasa analisis.

Langkah-langkah

Langkah 1 : Kenal Pasti Keperluan Bisnes Dan Sistem

Rujuk dokumen-dokumen **D02 Spesifikasi Keperluan Bisnes** dan **D03 Spesifikasi Keperluan Sistem** bagi mendapatkan keperluan fungsian dan bukan fungsian yang telah diperolehi daripada pemegang-pemegang taruh. Sila rujuk juga arkitektur *enterprise* untuk dijadikan sebagai input kepada arkitektur perisian yang ingin dibangunkan.

Langkah 2 : Laksanakan Pertimbangan Reka bentuk

Pertimbangan reka bentuk perlu dilakukan berdasarkan keperluan yang telah didapati semasa kajian keperluan sistem. Penilaian ke atas pertimbangan reka bentuk perlu mengambil kira perkara-perkara seperti berikut:

a) Penentuan Jenis Aplikasi

Pemilihan jenis aplikasi yang sesuai adalah penting bagi proses reka bentuk aplikasi. Pemilihan adalah tertakluk kepada keperluan yang telah diperolehi dan batasan infrastruktur serta kemampuan teknologi sedia ada. Contoh jenis-jenis aplikasi adalah seperti aplikasi mudah alih, *client-server application*, portal, aplikasi web tradisional atau aplikasi web moden (*Rich Internet Application*).

b) Penentuan Strategi Pelaksanaan

Pelaksanaan aplikasi boleh dilakukan di dalam pelbagai jenis persekitaran di mana setiap persekitaran mempunyai kekangan-kekangan tersendiri, Contohnya komponen-komponen yang perlu diasingkan secara fizikal dan perlu merentasi pelayan yang berbeza-beza, kekangan konfigurasi bagi peranti-peranti dan kekangan-kekangan yang lain. Keseimbangan di antara keperluan aplikasi dengan kemampuan perkakasan adalah penting di mana faktor-faktor ini boleh mempengaruhi kepada reka bentuk arkitektur yang dibangunkan.

c) Penentuan Ciri-ciri Kualiti

Ciri-ciri kualiti atau keperluan bukan fungsian seperti modular, keselamatan, prestasi, capaian, ketersediaan dan kebolehgunaan semula merupakan faktor-faktor yang perlu dipertimbangkan dalam penyediaan reka bentuk arkitektur. Pemilihan ciri-ciri kualiti yang ingin diutamakan adalah bergantung kepada keperluan yang telah diperolehi dan senario pelaksanaan aplikasi yang akan dijalankan kelak. Pemilihan keutamaan ini perlu dilakukan terlebih dahulu sebelum sebarang reka bentuk arkitektur dibangunkan bagi mengelakkan percanggahan di antara ciri-ciri kualiti yang perlu disediakan di dalam sistem. Sebagai contoh, keutamaan kepada keselamatan berkemungkinan boleh membebankan prestasi dan menyukarkan akses pengguna kepada aplikasi.

Langkah 3 : Kenal Pasti Corak Arkitektur Yang Sesuai

a) Corak arkitektur yang sesuai perlulah dikenal pasti berdasarkan kepada keperluan dan hasil pertimbangan reka bentuk. Corak reka bentuk arkitektur adalah bergantung kepada jenis sistem yang akan dibangunkan. Antara corak reka bentuk arkitektur yang sering digunakan adalah arkitektur lapisan (*layered architecture*) atau dikenali juga sebagai arkitektur *n-tier*. Berikut adalah fakta-fakta penting mengenai arkitektur lapisan:

i) Arkitektur lapisan merupakan gambaran aturan sistem dalam bentuk lapisan dan dihubungkan setiap lapisan tersebut dengan fungsi-fungsi yang berkaitan. Setiap

lapisan di dalam arkitektur mengandungi komponen-komponen sistem yang berbeza dan melaksanakan tugas serta logik yang tertentu. Arkitektur lapisan sesuai untuk digunakan bagi pembangunan aplikasi web atau aplikasi mudah alih berbentuk sistem maklumat kerana ia mudah untuk difahami serta pengasingan komponen-komponen sistem kepada lapisan yang berbeza meningkatkan kebolehsenggaraan dan keboleh-skalaan (*scalability*) sesuatu sistem.

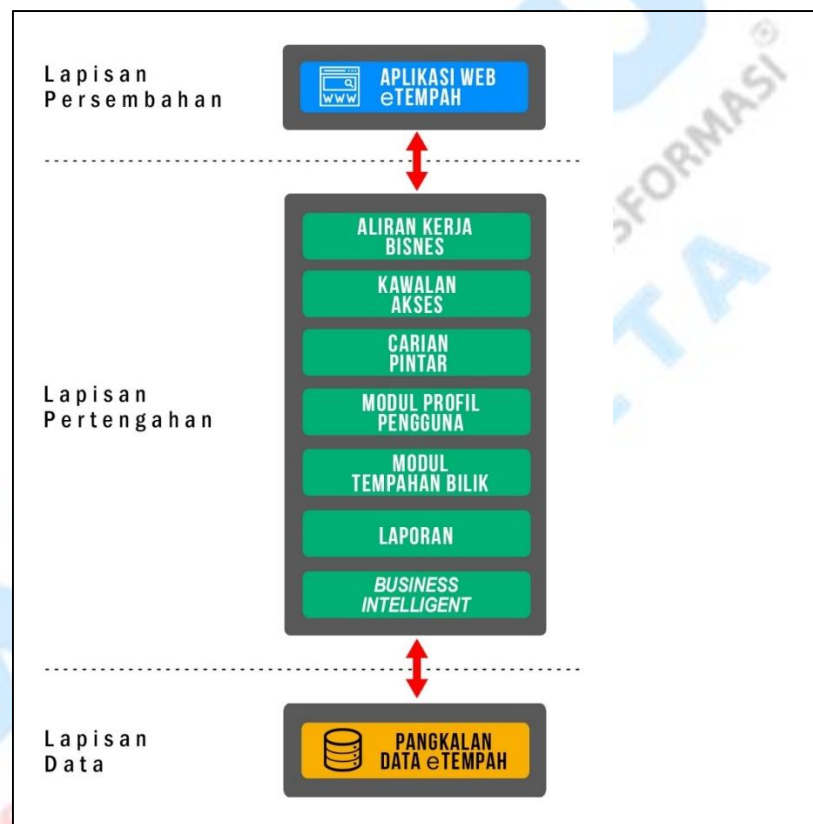
- ii) Secara amnya, arkitektur ini merangkumi lapisan-lapisan seperti lapisan persembahan (*presentation tier*), lapisan pertengahan (*middle tier*) dan lapisan data (*data tier*).
 - Lapisan persembahan merupakan lapisan di mana pengguna berinteraksi dengan aplikasi. Lapisan ini bertanggungjawab dalam mengendalikan kesemua antaramuka pengguna serta logik komunikasi pelayar web.
 - Lapisan pertengahan atau dikenali juga sebagai lapisan aplikasi (*application tier*) atau lapisan bisnes merupakan lapisan pengantaraan yang menghubungkan di antara lapisan persembahan dan lapisan data. Lapisan pertengahan ini terdiri dari komponen-komponen yang melibatkan fungsi logik bisnes aplikasi, perkhidmatan aplikasi atau/dan komponen utiliti (perisian utiliti) yang digunakan oleh komponen-komponen aplikasi yang lain.
 - Lapisan data pula merupakan lapisan di mana maklumat-maklumat aplikasi disimpan di dalam pelayan pangkalan data.
- b) Corak-corak arkitektur lain seperti *event-driven*, *microkernel*, *space-based* dan *client-server* boleh juga diguna pakai tetapi bergantung kepada kesesuaian dan juga keperluan yang telah diperolehi. Untuk mendapatkan keterangan lanjut berkenaan corak-corak arkitektur yang lain, buku-buku seperti di bawah boleh dirujuk:
 - i) *Software Architecture Patterns - Understanding Common Architecture Patterns and When to Use Them* (2015) oleh Mark Richards
 - ii) *Software Engineering – Chapter 6 Architectural Design (9th Edition 2011)* oleh Ian Sommerville

Langkah 4 : Sediakan Arkitektur Keseluruhan Sistem Aplikasi

- a) Arkitektur Keseluruhan Sistem Aplikasi adalah merupakan:
 - i) komponen-komponen perisian iaitu antaramuka sistem, aplikasi dan repositori (data)
 - ii) sifat-sifat luaran (*external properties*) komponen berkenaan
 - iii) hubungan di antara satu komponen dengan komponen-komponen yang lain
- b) Arkitektur Sistem Aplikasi boleh dibahagikan kepada dua jenis seperti berikut:

i) Arkitektur Monolitik

Arkitektur monolitik adalah arkitektur yang menggabungkan semua komponen fungsian perisian seperti kawalan akses, aliran kerja, modul profil pengguna dan laporan, menjadi satu unit sahaja. Perisian yang mengguna pakai arkitektur monolitik direka bentuk supaya ia bersifat *self-contained*, di mana komponen-komponen perisian berkenaan saling berhubung (*interconnected*) dan saling bergantung (*interdependent*) di antara satu sama lain. Dengan kata lain, arkitektur monolitik merupakan arkitektur yang bersifat *tightly-coupled*, setiap komponen yang berkaitan perlu disediakan bersama bagi membolehkan ia dilaksanakan. Berikut adalah contoh arkitektur bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur monolitik:

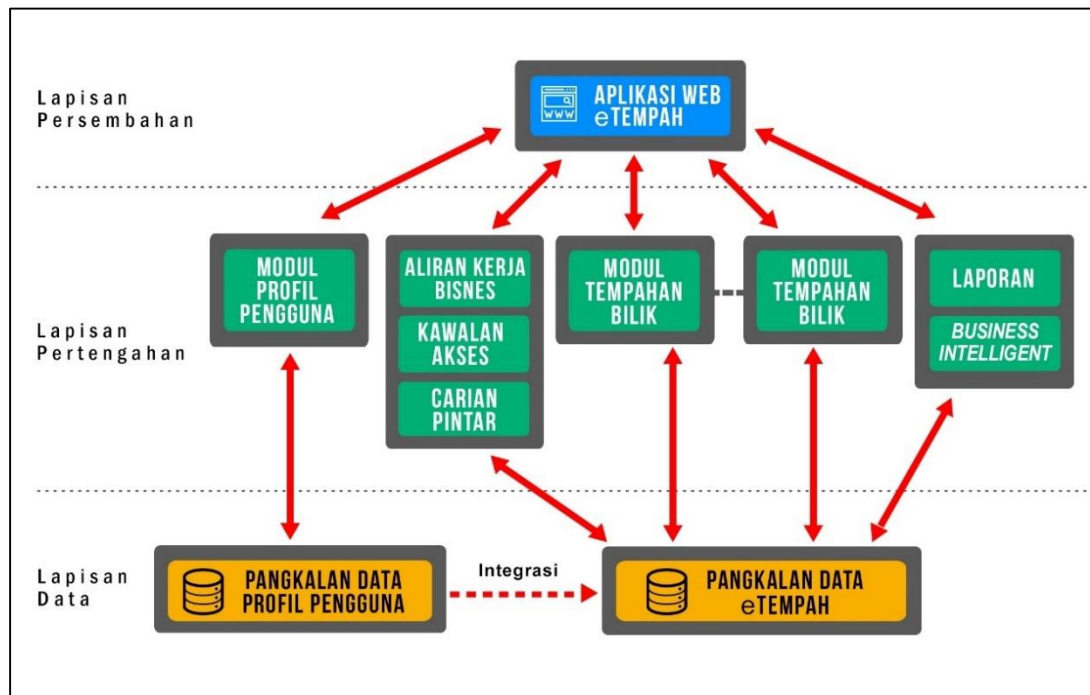


Rajah 57 : Arkitektur Monolitik Bagi Perisian Sistem Aplikasi

ii) Arkitektur Mikroservis

Arkitektur mikroservis adalah arkitektur yang mengagihkan perisian atau sistem aplikasi di pihak pelayan (*server side*) kepada servis-servis atau komponen-komponen perisian yang berasingan. Setiap servis atau komponen merupakan satu proses kecil yang tidak bergantung kepada mana-mana proses lain (*independent*). Mikroservis melaksanakan satu-satu tugas (*task*) secara autonomi tanpa mengganggu komponen-komponen lain di dalam sistem aplikasi. Komunikasi di antara satu servis dengan servis-servis yang lain dilaksanakan secara dalam

talian. Berikut adalah contoh arkitektur bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur mikroservis:



Rajah 58 : Arkitektur Mikroservis Bagi Perisian Sistem Aplikasi

c) Perbandingan di antara arkitektur monolitik dengan mikroservis adalah seperti berikut:

Jadual 31 : Perbandingan Arkitektur Monolitik dan Mikroservis

Ciri-ciri	Arkitektur Monolitik	Arkitektur Mikroervis
Pengurusan Sumber	Sukar untuk menentukan jumlah sumber yang diperlukan oleh setiap komponen-komponen perisian.	<p>i) Keperluan sumber bagi setiap komponen/servis boleh diukur secara berasingan.</p> <p>ii) Arkitektur ini juga membolehkan penggunaan sumber dilaksanakan secara optimum mengikut keperluan modul dan komponen/servis.</p>

Ciri-ciri	Arkitektur Monolitik	Arkitektur Mikrosevis
Pembangunan	Pembangunan perisian dilaksanakan secara konvensional. Variasi di dalam penggunaan teknologi, rangka kerja serta ahli pasukan yang terlibat dalam satu-satu projek sukar untuk dilaksanakan.	<ul style="list-style-type: none"> i) Pembangunan boleh dilakukan dengan menggunakan teknologi, bahasa pengaturcaraan dan rangka kerja yang berbeza oleh kerana setiap servis/komponen boleh berinteraksi dalam talian menggunakan kemudahan integrasi seperti <i>Application Programming Interface (API)</i>. ii) Disebabkan servis/komponen di dalam arkitektur ini berinteraksi secara dalam talian, ia juga boleh digunakan oleh perisian atau aplikasi yang lain. iii) Pengasingan komponen-komponen perisian membolehkan aplikasi dibangunkan oleh pasukan-pasukan kecil yang berlainan. iv) Pengaturcaraan lebih sukar kerana ia perlu melibatkan pengaturcaraan yang melibatkan interaksi di antara servis/komponen sama ada melalui panggilan API atau integrasi data.
Komunikasi dan Rangkaian	Perisian/Aplikasi monolitik adalah bersifat <i>self-contained</i> di mana semua komponen dikumpulkan dalam satu unit sahaja. Justeru, tidak timbul isu <i>latency</i> dan kebergantungan kepada prestasi rangkaian.	Komunikasi di antara servis/komponen berkemungkinan perlahan kerana bergantung kepada prestasi rangkaian.
Pengujian	i) Langkah pengujian adalah lebih ringkas kerana tidak memerlukan semakan lanjut kepada elemen-elemen tambahan yang melibatkan integrasi di antara komponen ataupun data.	i) Langkah pengujian lebih kompleks berbanding dengan arkitektur monolitik kerana semakan perlu melibatkan elemen-elemen integrasi di antara servis/komponen dan data.

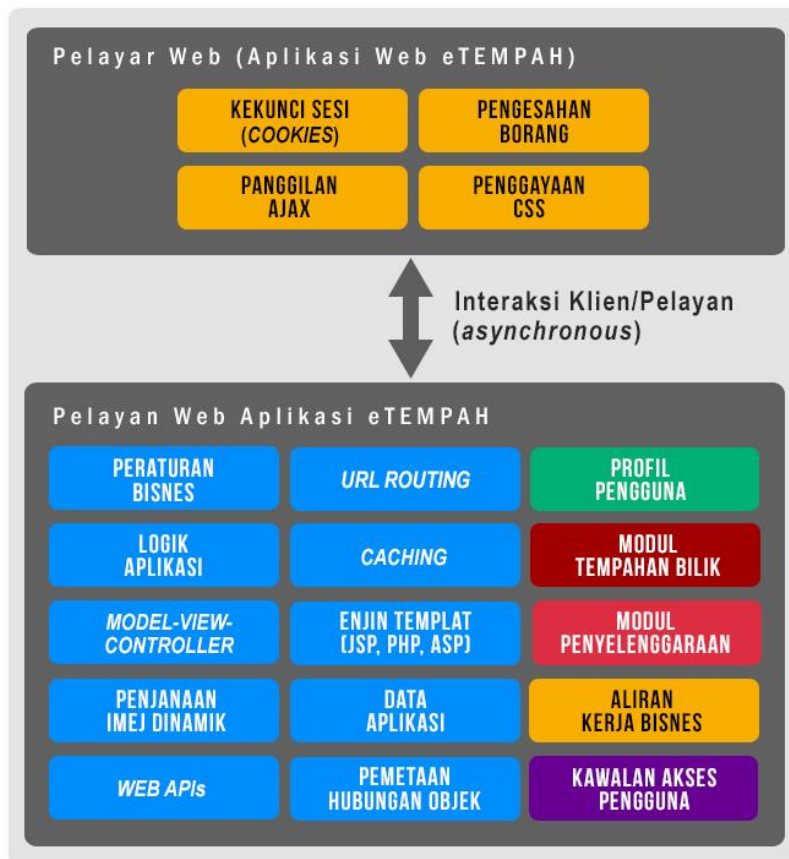
Ciri-ciri	Arkitektur Monolitik	Arkitektur Mikrosevis
	ii) Namun begitu, pengujian dan <i>debugging</i> secara berasingan untuk setiap komponen sukar untuk dilaksanakan.	ii) Pengujian dan <i>debugging</i> secara berasingan lebih mudah untuk dilakukan kerana servis/komponen sudah sedia teragih.
Kebolehskalaan (<i>Scalability</i>)	Peningkatan sumber (memori dan CPU) adalah tidak fleksibel kerana peningkatan perlu mengambil kira kesemua komponen di dalam sistem aplikasi.	Peningkatan sumber (memori dan CPU) secara penskalaan menegak (<i>vertical scaling</i>) dan mendatar (<i>horizontal scaling</i>) lebih anjal kerana peningkatan boleh dibuat mengikut keperluan modul dan komponen/servis.
Kebolehsediaan (<i>Availability</i>)	Arkitektur ini mempunyai ciri kebolehsediaan yang tinggi (<i>High Availability (HA)</i>) di mana semua modul dan komponen/servis perlu dimasukkan ke dalam kluster HA.	Arkitektur ini mempunyai ciri kebolehsediaan yang tinggi (HA) di mana modul-modul dan komponen/servis yang tertentu boleh dipilih untuk dimasukkan ke dalam kluster HA.
Pelaksanaan	Pelaksanaan kebiasaannya perlu dilaksanakan secara menyeluruh kerana setiap komponen perisian bergantung di antara satu sama lain.	Pelaksanaan boleh dilaksanakan secara berasingan tanpa perlu melibatkan servis/komponen lain yang masih belum selesai dibangunkan.
Penyelenggaraan	<p>i) Sebarang perubahan berkemungkinan memerlukan seluruh perisian/aplikasi untuk dibina semula (<i>rebuild</i>) atau <i>redeploy</i> disebabkan oleh kebergantungan komponen-komponen di antara satu sama lain.</p> <p>ii) Oleh yang demikian, masa yang diambil untuk sebarang penambahbaikan ataupun pembaikan (<i>error fixing</i>) adalah lama dan mudah terdedah dengan pelbagai kesilapan.</p> <p>iii) Sebarang kesilapan bukan sahaja boleh menjejaskan komponen/</p>	Sebarang perubahan kepada satu-satu servis hanya melibatkan penyelenggaraan kepada servis/komponen berkenaan sahaja dan tidak mengganggu kepada yang lain. Justeru, proses penambahbaikan dan pembaikan dapat diselesaikan dengan lebih pantas.

Ciri-ciri	Arkitektur Monolitik	Arkitektur Mikrosevis
	servis yang lain malah ia juga berkemungkinan akan menyebabkan seluruh perisian/aplikasi terjejas.	

- d) Tentukan sama ada arkitektur perisian sistem aplikasi yang ingin dibangunkan adalah berpandukan arkitektur monolitik atau mikroservis. Penentuan arkitektur ini perlu diseimbangkan di antara halatuju agensi, kekangan-kekangan dan infrastruktur sedia ada dengan keperluan fungsian dan bukan fungsian yang telah diperolehi di fasa Analisis Keperluan.
- e) Berdasarkan dua jenis arkitektur perisian sistem aplikasi seperti di atas dengan mengambil kira perbandingan di antara kedua-duanya, sediakan arkitektur yang ideal dan sesuai bagi reka bentuk perisian sistem aplikasi yang ingin dibangunkan.

Langkah 5 : Sediakan Arkitektur Aplikasi

- a) Arkitektur aplikasi merupakan subset kepada Arkitektur Keseluruhan Sistem Aplikasi di mana ia tertumpu kepada penyusunan dan pengaturan sistem aplikasi sahaja. Kemajuan teknologi di dalam pembangunan sistem aplikasi telah banyak mempengaruhi perubahan kepada arkitektur aplikasi.
- b) Dalam persekitaran web, arkitektur aplikasi boleh dibahagikan kepada dua jenis seperti berikut:
- i) **Arkitektur Aplikasi Tradisional**
- Arkitektur aplikasi tradisional lebih tertumpu kepada penggunaan dan utiliti di pihak pelayan (*server-side*), di mana pembentuk setiap muka web adalah berpandukan kepada skrip-skrip HTML dan kod pengaturcaraan lain yang dijana dan ditarik dari pelayan. Arkitektur ini adalah terdiri daripada aplikasi yang merupakan muka-muka web (*web pages*). Bagi mendapatkan kandungan yang dinamik, web aplikasi tradisional perlu sentiasa berhubung dengan pelayan web bagi menjana semula muka web dan memaparkan permohonan-permohonan yang dilakukan oleh pengguna (*user request*). Berikut adalah contoh arkitektur aplikasi bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur aplikasi tradisional:



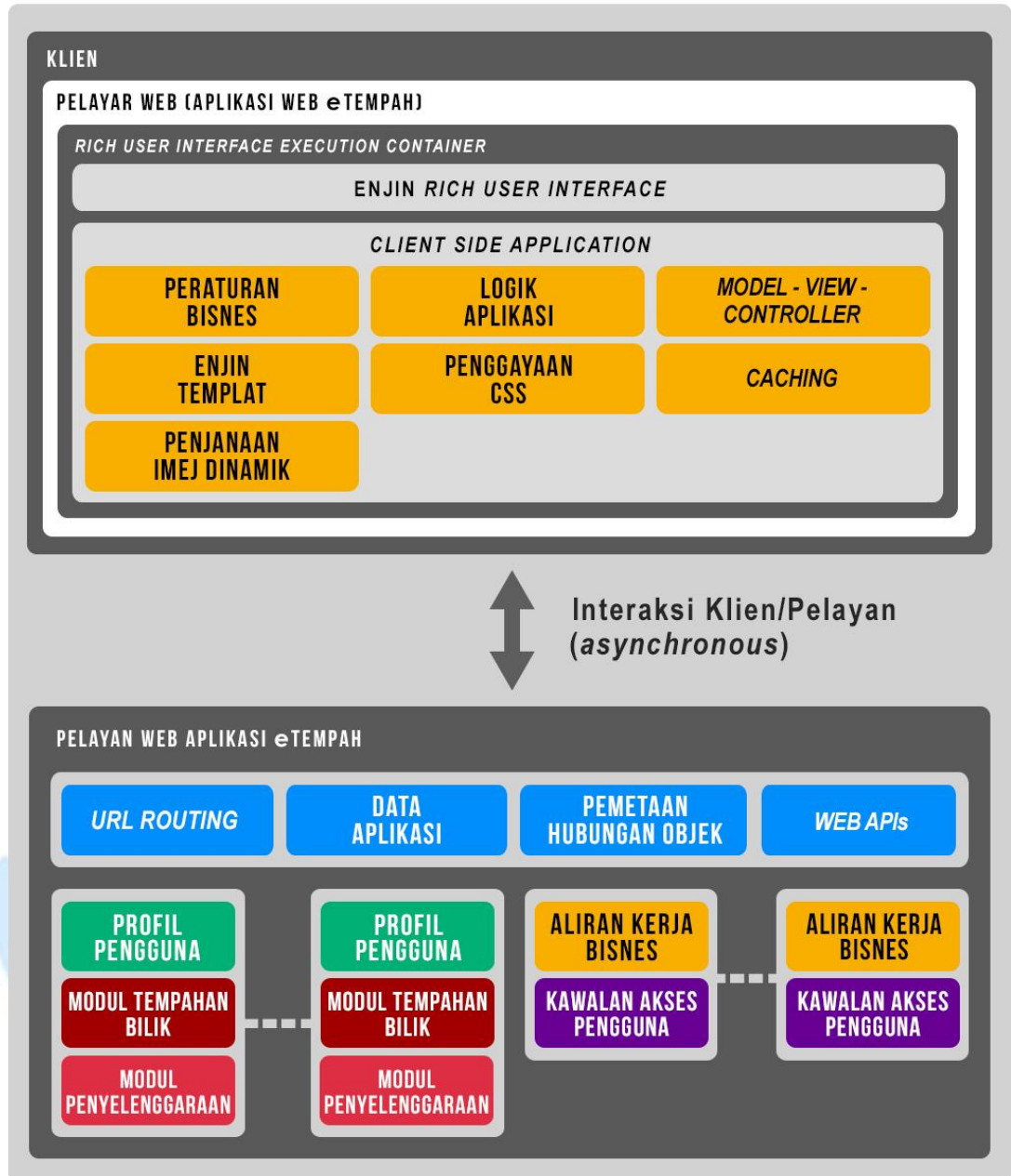
Rajah 59 : Arkitektur Aplikasi Tradisional

ii) Arkitektur Aplikasi Moden

Perkembangan teknologi bahasa pengaturcaraan seperti HTML5 dan CSS3, serta pengenalan kepada persekitaran *runtime* Javascript telah banyak merubah arkitektur aplikasi tradisional. Arkitektur aplikasi moden yang menggunakan teknologi terkini adalah lebih tertumpu kepada utiliti di pihak klien (*client-side*) dan meminimalkan penggunaan serta beban di pihak pelayan. Aplikasi dalam arkitektur ini bertindak sebagai aplikasi *single page* di mana :

- interaksi di antara klien dengan pelayan hanya melibatkan *unformatted data* sahaja
- pengesahan dapat dibuat secara langsung di klien (*live validation*)
- *unnecessary page reload*
- kemudahan *drag and drop*
- kemudahan animasi multimedia
- masa respon yang singkat
- *auto completion*,
- *periodic refresh*
- *rich text editors*
- *pull technology*

Ini menjadi aplikasi lebih kukuh (*robust*), meningkatkan *user experience*, prestasi serta tahap responsif sistem. Berikut adalah contoh arkitektur aplikasi bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur aplikasi moden:



Rajah 60 : Arkitektur Aplikasi Moden

- c) Perbandingan di antara arkitektur aplikasi tradisional dengan aplikasi moden adalah seperti berikut:

Jadual 32 : Perbandingan Arkitektur Aplikasi Tradisional Dan Aplikasi Moden

Ciri-ciri	Arkitektur Aplikasi Tradisi	Arkitektur Aplikasi Moden
Pengurusan Sumber	<p>i) Arkitektur web aplikasi tradisional membebankan dan meningkatkan kompleksiti di pihak pelayan. Pelayan aplikasi perlu melaksanakan sebahagian besar daripada pemprosesan perisian termasuk juga pemprosesan yang dilaksanakan di dalam pelayar web dan antaramuka pengguna (UI).</p> <p>ii) Keperluan sumber yang tinggi di pihak pelayan ini bukan sahaja memerlukan kos yang tinggi malah ia juga boleh menjejaskan prestasi keseluruhan aplikasi sekiranya keperluan tersebut tidak dapat dipenuhi.</p>	<p>Arkitektur web aplikasi moden menawarkan <i>native-app-like experience</i> kepada pengguna di mana sebahagian besar daripada <i>runtime</i> aplikasi dilaksanakan di pihak klien. Oleh yang demikian, utiliti di pihak server dapat dilaksanakan secara efisien tanpa memerlukan sumber yang tinggi untuk memenuhi keperluan operasi satu-satu aplikasi.</p>
Pembangunan	<p>i) Disebabkan arkitektur ini adalah bersifat <i>tightly coupled</i> dan komponen aplikasi kian bergantung di antara satu sama lain, pembangunan perlu dilaksanakan secara menyeluruh dan bersekali di pihak klien dan juga pelayan.</p> <p>ii) Pendekatan pembangunan lebih kepada pembangunan muka web (<i>web pages</i>).</p>	<p>i) Pengasingan dalam kebergantungan (<i>loosely coupled</i>) di antara pihak klien dan pelayan membolehkan pembangunan aplikasi dijalankan secara teragih. Pembangunan boleh dilaksanakan oleh dua pasukan yang berbeza, di mana satu kumpulan akan lebih fokus kepada pengaturcaraan di pihak klien (pengaturcaraan HTML, CSS dan Javascript boleh dilakukan tanpa perlu untuk mengaturcara terlebih dahulu teknologi</p>

Ciri-ciri	Arkitektur Aplikasi Tradisi	Arkitektur Aplikasi Moden
		<p><i>templating</i> aplikasi seperti JSP, ASP dan PHP) manakala kumpulan yang lain membangunkan aplikasi dan juga API web di pihak klien.</p> <p>ii) Pendekatan pembangunan lebih kepada pembangunan aplikasi.</p>
Komunikasi dan Rangkaian	Sebarang isu rangkaian yang timbul akan mengakibatkan aplikasi tidak boleh dicapai dan digunakan. Ini kerana, arkitektur aplikasi tradisional memerlukan komunikasi dengan pelayan untuk menjana fail-fail program seperti HTML dan imej dinamik (antaramuka pengguna) kepada pihak klien.	Arkitektur aplikasi moden boleh dimuatkan ke dalam pelayar web atau disimpan secara luar talian melalui <i>cache manifest</i> . Ini membolehkan fungsi-fungsi dalam aplikasi untuk terus digunakan walaupun terdapat gangguan rangkaian.
Pengujian	Pengujian aplikasi bagi arkitektur ini hanya boleh dilaksanakan secara <i>end-to-end</i> dan menyeluruh. Pengujian secara teragih atau <i>unit test</i> sukar untuk dilakukan oleh kerana kebergantungan yang tinggi di antara komponen-komponen aplikasi.	Arkitektur ini membolehkan pengujian dilaksanakan sama ada secara menyeluruh dan juga teragih (<i>unit test</i>) kerana kebergantungan di antara komponen-komponen aplikasi adalah rendah. Contohnya, pengujian API web atau <i>Javascript</i> boleh dilaksanakan secara berasingan tanpa perlu melibatkan komponen teknologi <i>templating</i> aplikasi seperti JSP, ASP dan PHP.
Pelaksanaan	Pelaksanaan dijalankan secara konvensional di mana aplikasi diakses melalui alamat URL di dalam pelayar web.	Pelaksanaan adalah lebih fleksibel dan menawarkan pelbagai opsyen. Pelaksanaan aplikasi boleh dijalankan dengan menggunakan pelayan web (sama seperti arkitektur aplikasi tradisional) dan ia juga boleh dipakejkan dengan rangka-rangka kerja seperti <i>Apache</i> dan <i>Cordova</i> bagi pengedaran melalui <i>App Store</i> atau <i>Google Play</i> .

Ciri-ciri	Arkitektur Aplikasi Tradisi	Arkitektur Aplikasi Moden
Prestasi	Aplikasi merupakan satu set <i>web pages</i> yang akan dijana di pelayan apabila pengguna klik pada alamat URL yang berkaitan. Proses ini akan memperlahankan sistem dan menjadi lebih kritikal apabila ramai pengguna membuat capaian secara serentak.	Merupakan aplikasi yang dimuatkan dalam pelayar web di mana hanya data sahaja yang ditarik daripada pelayan dan antaramuka pengguna (UI) dijana di pihak klien. Ini memberikan prestasi sistem yang lebih baik.
Penyelenggaraan	Penyelenggaraan aplikasi perlu dilaksanakan secara menyeluruh disebabkan oleh kebergantungan yang tinggi di antara komponen-komponen aplikasi (<i>tightly coupled</i>). Sebarang perubahan kod pengaturcaraan sama ada di klien atau pelayan perlu diselaraskan bagi membolehkan reka bentuk, ciri-ciri atau fungsi yang baru dipinda dapat berjalan dengan lancar.	Oleh kerana arkitektur ini adalah bersifat <i>loosely coupled</i> di mana kesemua logik persembahan diasingkan daripada komponen di pihak pelayan, sebarang pindaan di antaramuka aplikasi boleh dilaksanakan tanpa mengganggu kod pengaturcaraan di pihak pelayan. Justeru, sebarang perubahan dan evolusi kepada reka bentuk dan ciri-ciri (<i>features</i>) aplikasi dapat diselesaikan dengan lebih pantas.

- d) Tentukan sama ada arkitektur aplikasi yang ingin dibangunkan merupakan arkitektur aplikasi tradisional ataupun aplikasi moden. Penentuan arkitektur ini perlu diseimbangkan di antara halatuju agensi, kekangan-kekangan dan infrastruktur sedia ada dengan keperluan fungsian dan bukan fungsian yang telah diperolehi di fasa Analisis Keperluan.
- e) Berdasarkan dua jenis arkitektur aplikasi seperti di atas dengan mengambil kira perbandingan di antara kedua-duanya, sediakan arkitektur yang ideal dan sesuai bagi reka bentuk aplikasi yang ingin dibangunkan.

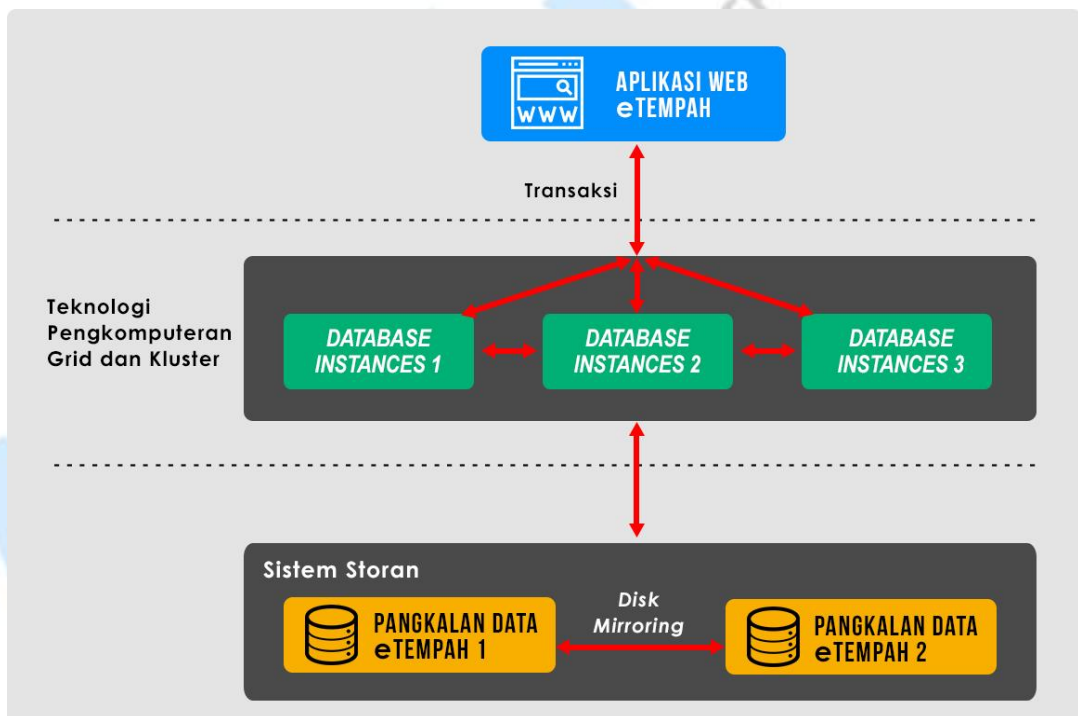
Langkah 6 : Sediakan Arkitektur Pangkalan Data

- a) Arkitektur pangkalan data merupakan subset kepada arkitektur perisian sistem aplikasi di mana ia tertumpu kepada penyusunan dan pengaturan pangkalan data sahaja. Penyediaan arkitektur tersebut adalah berdasarkan kepada keperluan fungsian dan bukan fungsian yang telah diperolehi serta corak arkitektur yang telah ditentukan.

b) Arkitektur pangkalan data boleh dibahagikan kepada dua jenis seperti berikut:

i) Arkitektur *Shared-Disk*

Sistem-sistem yang wujud di dalam persekitaran *shared-disk* berkongsi peranti cakera (*disk devices*) yang sama. Setiap pemproses mempunyai memori tersendiri dan ia boleh mengakses kesemua cakera peranti. Arkitektur *shared disk* membolehkan setiap nod mengakses keseluruhan set data di mana setiap nod berkenaan akan memberi maklum balas kepada sebarang permohonan yang diterima dari pangkalan data. Arkitektur *shared-disk* adalah sesuai bagi aplikasi yang hanya memerlukan *shared access* yang sederhana ke pangkalan data, serta aplikasi yang mempunyai beban kerja (*workload*) yang sukar untuk diagihkan (*partition*). Berikut adalah contoh arkitektur pangkalan data bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur *shared-disk*:

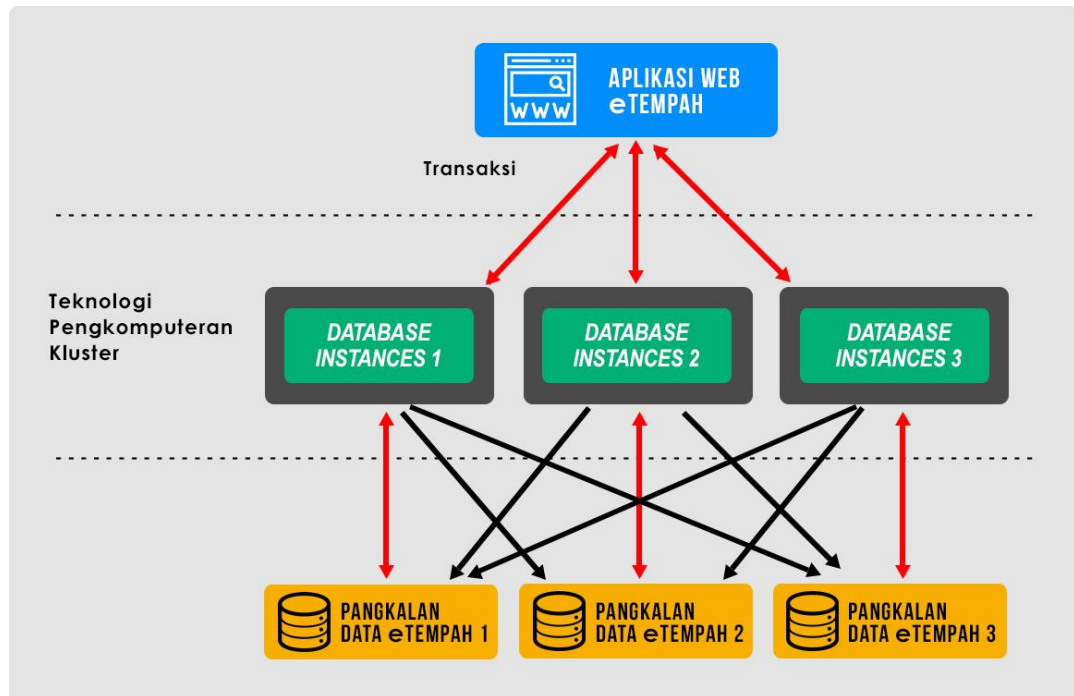


Rajah 61 : Arkitektur *Shared-Disk* Bagi Aplikasi Web eTempah

ii) Arkitektur *Shared-Nothing*

Sistem-sistem yang wujud di dalam persekitaran *shared-nothing* mempunyai memori dan cakera peranti yang tersendiri. Arkitektur *shared-nothing* mempunyai kemampuan penskalaan yang tinggi (*high scalability*) dan sesuai untuk aplikasi yang kerap mengakses dan melaksanakan transaksi ke pangkalan data. Berikut

adalah contoh arkitektur pangkalan data bagi Sistem Tempahan Bilik Mesyuarat (eTempah) yang direka bentuk berpandukan arkitektur *shared-nothing*:



Rajah 62 : Arkitektur *Shared-Nothing* Bagi Aplikasi Web eTempah

- c) Perbandingan di antara arkitektur *shared-disk* dengan *shared-nothing* adalah seperti berikut:

Jadual 33 : Perbandingan Arkitektur Shared-Disk Vs Shared-Nothing

Ciri-ciri	Arkitektur <i>Shared-Disk</i>	Arkitektur <i>Shared-nothing</i>
Pemasangan dan Penyelenggaraan	<p>Tidak Memerlukan Usaha Tambahan</p> <p>Arkitektur ini tidak bergantung kepada <i>data partitioning</i> kerana setiap nod di dalam kluster mempunyai akses penuh untuk memaparkan dan mengemaskini seluruh data.</p>	<p>Jadual <i>Re-Partitioning</i> Dan <i>Routing</i></p> <p>Berikut adalah langkah-langkah pemasangan bagi pangkalan data <i>shared-nothing</i>:</p> <ol style="list-style-type: none"> Sediakan skema <i>partitioning</i> untuk meminimakan atau menghapuskan secara terus mesej antara-nodal. Asingkan data di antara server-server dalam kluster. Selenggara jadual <i>routing</i>.

Ciri-ciri	Arkitektur <i>Shared-Disk</i>	Arkitektur <i>Shared-nothing</i>
		iv) Ulangi semula langkah-langkah di atas bagi setiap <i>re-partition</i> .
Prestasi dan <i>Overhead</i>		
Fungsi / <i>Data Shipping</i>	Meningkatkan Prestasi Tiada fungsi/ <i>data shipping</i> , <i>overhead</i> tambahan tidak diperlukan.	Mengehadkan Prestasi Dan Masalah Penalaan (<i>Tuning</i>) Arkitektur ini bergantung kepada fungsi/ <i>data shipping</i> untuk memenuhi permintaan (<i>request</i>) kepada pangkalan data (seperti <i>joins</i>) yang merentasi pelbagai nod. Arkitektur ini boleh mengehadkan prestasi serta kebolehskalaan nodal berdasarkan skema <i>partitioning</i> yang dibangunkan, saiz data dan bilangan nod yang terlibat.
<i>2-Phase Commit</i> (2PC)	Prestasi yang Lebih Pantas Setiap nod boleh menyelaras nod masing-masing tanpa perlu menunggu nod yang paling perlahan untuk dilaksanakan.	Prestasi 2PC Teragih (<i>Distributed</i>) Merentangi Pelbagai Nod Adalah Sangat Perlahan 2PC teragih menguncikan data pada seluruh nod yang terlibat. Ini mengakibatkan prestasi semua permintaan data-data menjadi perlahan.
Kelajuan Mengakses Data	Latensi Mengakses Data SAN Bagi kluster-kluster penskalaan secara luaran (<i>scale-out</i>), kelajuan antarahubungan (<i>interconnect</i>) <i>shared-disk</i> adalah lebih perlahan berbanding dengan kelajuan <i>bus</i> (<i>bus speed</i>).	Cakera Setempat (<i>Local disk</i>), Kelajuan Mengakses <i>Bus</i> Akses kepada data setempat (<i>local data</i>) dapat dilaksanakan dengan pantas.
<i>Load Balancing</i>	Mengoptimalkan Utiliasi CPU Merentasi Pelayan-pelayan (<i>Server</i>) Setiap pelayan dapat berfungsi pada tahap utiliasi	Mengoptimalkan Utiliasi CPU Pada Setiap Nod Setiap nod dan pelayan perlu menguruskan beban secara sendiri

Ciri-ciri	Arkitektur <i>Shared-Disk</i>	Arkitektur <i>Shared-nothing</i>
	CPU yang lebih tinggi kerana beban pemprosesan dapat diagihkan kepada pelayan-pelayan lain di dalam kluster yang sama.	tanpa sebarang <i>load balancing</i> yang dinamik.
Mesej Antara-Nodal (<i>Inter-Nodal Messaging</i>)	<i>Messaging Overhead</i> Arkitektur ini bergantung kepada mesej antara-nodal.	Tiada Mesej Antara-Nodal Arkitektur ini tidak memerlukan mesej antara-nodal.
Ketersediaan Tinggi (<i>High Availability</i>)	<i>Master-master Failover</i> a) Tidak memerlukan <i>partitioning</i> , justeru, <i>planned downtime</i> dapat dikurangkan secara dramatik. b) <i>Unplanned downtime</i> juga dapat dikurangkan. Sekiranya terdapat pelayan yang bermasalah (<i>failed</i>), arkitektur ini secara dinamik akan mengubah haluannya kepada server-server yang lain tanpa sebarang <i>downtime</i> .	Keperluan <i>Planned</i> dan <i>Unplanned Downtime</i> a) <i>Planned downtime</i> diperlukan untuk melaksanakan <i>re-partitioning</i> . b) Kegagalan (<i>failed</i>) kepada nod <i>master</i> akan mengakibatkan <i>downtime</i> sehingga satu-satu nod <i>slave</i> dinaik taraf menjadi nod <i>master</i> .
Konsistensi Data	Tiada Masalah Konsistensi Data Arkitektur ini tidak mengalami masalah konsistensi data oleh kerana ia mempunyai pelbagai salinan data sama dalam server yang berlainan.	Mempunyai Masalah Konsistensi Data Perlu melakukan konfigurasi kepada transaksi supaya ia merangkumi replikasi secara <i>synchronous</i> dalam transaksi berkenaan. Replikasi ini akan menyebabkan prestasi kelajuan satu-satu pangkalan data menjadi perlahan.

Ciri-ciri	Arkitektur <i>Shared-Disk</i>	Arkitektur <i>Shared-nothing</i>
Kebolehskalaan (<i>Scalability</i>)		
Penskalaan Secara Dalam (Scale-in) : Satu Pendekatan Kebolehskalaan Dalam Pelayan-pelayan Moden <i>MultiCore</i>	<p>Menyokong Pendekatan Penskalaan Secara Dalam</p> <p>Arkitektur ini menyokong penskalaan secara dalam kerana ia mengguna pakai dengan menyuluruh keupayaan <i>multiple cores</i> di dalam pelayan moden. Enjin <i>shared-disk</i> membolehkan permintaan kepada pangkalan data diagihkan kepada mana-mana <i>instances</i>.</p>	<p><i>Partitioning</i> Data Kurang Sesuai Bagi Penskalaan Secara Dalam</p> <p>Arkitektur ini perlu merangkumi serta melibatkan pangkalan data dan enjin storan untuk menggunakan pendekatan penskalaan secara dalam. Ia juga memerlukan capaian aplikasi kepada setiap <i>instance</i> secara berasingan. Selain itu, arkitektur ini memerlukan setiap cakera dibahagikan kepada storan yang berbeza bagi setiap <i>instance</i>. Oleh yang demikian, keperluan-keperluan ini akan mengakibatkan pangkalan data sukar untuk diuruskan.</p>
Pengedaran Berdasarkan Geografi (<i>Geo-Distribution</i>)	<p>Isu <i>Latency</i> Bagi Lokasi yang Berbeza</p> <p>Arkitektur ini mempunyai isu <i>latency</i> bagi lokasi pelayan-pelayan yang berbeza dari segi geografinya. Isu ini hanya boleh diselesaikan sekiranya dua atau lebih sistem <i>shared-disk</i> dibangunkan secara berasingan.</p>	<p>Isu <i>Latency</i> Bagi Lokasi yang Berbeza</p> <p>Sama seperti arkitektur <i>shared-disk</i>, arkitektur ini mempunyai isu <i>latency</i> bagi lokasi server-server yang berbeza dari segi geografinya. Isu ini boleh diminimakan dengan membahagi dan mengagih data berdasarkan geo-lokasi. Malah ia juga dapat mengurangkan atau menghapuskan fungsi/<i>data shipping</i> di antara dua lokasi berkenaan.</p>
Pengkomputeran Cloud : <i>Database-as-a-Service</i> (DaaS)	<p>Menyokong Kebolehskalaan Nodal Secara Dinamik</p> <p>Arkitektur ini adalah sesuai bagi penskalaan nodal atau keanjalan nodal (<i>nodal elasticity</i>). Memandangkan semua nod berhubung dengan <i>shared data store</i> yang</p>	<p>Pembahagian Tidak Optimal</p> <p>Pembahagian atau <i>partitioning</i> merupakan isu yang kerap dihadapi bagi pangkalan data <i>cloud</i>. Pembahagian ini dilaksanakan secara automasi dan sesuai bagi keanjalan nodal. Namun begitu, pembahagian secara automasi ini hanyalah separa optimal sahaja. Ia akan meningkatkan fungsi/<i>data-</i></p>

Ciri-ciri	Arkitektur <i>Shared-Disk</i>	Arkitektur <i>Shared-nothing</i>
	sama, bilangan nod-nod boleh diubah dengan mudah tanpa memberi sebarang implikasi kepada data.	<i>shipping</i> yang akan menyebabkan prestasi keseluruhan pangkalan data terganggu. Prestasi pangkalan data akan kian merosot sekiranya bilangan nod semakin bertambah.

- d) Tentukan sama ada arkitektur pangkalan data yang ingin dibangunkan merupakan arkitektur *shared-disk* ataupun *shared-nothing*. Penentuan arkitektur ini perlu diseimbangkan di antara halatuju agensi, kekangan-kekangan dan infrastruktur sedia ada dengan keperluan fungsian dan bukan fungsian yang telah diperolehi di fasa Analisis Keperluan.
- e) Berdasarkan dua jenis arkitektur pangkalan data seperti di atas dengan mengambil kira perbandingan di antara kedua-duanya, sediakan arkitektur yang ideal dan sesuai untuk reka bentuk pangkalan data yang akan dibangunkan.

Langkah 7 : Sediakan Arkitektur-Arkitektur Secara Iteratif

Penyediaan dan pembangunan arkitektur perlu dilaksanakan secara iteratif bagi meningkatkan tahap komprehensif dan memenuhi keperluan-keperluan yang telah diperolehi. Disyorkan supaya iterasi proses penyediaan arkitektur dilaksanakan sekurang-kurangnya sebanyak dua kali untuk meminimalkan sebarang kesilapan dan kekurangan di dalam reka bentuk arkitektur.

Rujukan

1. Narayan Prusty, Modern Javascripts Application (2016).
2. Mark Richards, Software Architecture Patterns (2015).
3. Danny Brian, Kirk Knoernschild. Modern Web Architecutre (2016).
<http://www.gartner.com/>.
4. Ben Stopford. Shared Nothing v.s. Shared Disk Architectures: An Independent View (2009). <http://www.benstopford.com/2009/11/24/understanding-the-shared-nothing-architecture/>.

4.6 Penentuan Teknologi [F3.2]

Pengenalan

Penentuan teknologi dan *tool* yang sesuai bagi pembangunan sistem aplikasi merupakan salah satu aspek yang penting untuk dipertimbangkan. Teknologi dan *tool* yang dipilih akan digunakan bagi memandu dalam reka bentuk antaramuka, reka bentuk proses dan reka bentuk logikal pangkalan data. Teknologi yang dipilih juga akan digunakan semasa pelaksanaan fasa pembangunan, pengujian, pengoperasian dan penyelenggaraan sistem aplikasi. Bagi menentukan teknologi yang akan digunakan, beberapa aspek perlu diambil kira:

- a) Memenuhi dan mematuhi reka bentuk arkitektur sistem aplikasi
- b) Keserasian dengan keperluan fungsian dan bukan fungsian
- c) Selaras dengan visi dan misi organisasi
- d) Teknologi-teknologi yang mudah diperolehi dan diselenggara

Matriks Alternatif

Matriks alternatif ialah salah satu kaedah yang digunakan untuk memilih teknologi yang memenuhi keperluan. Matriks alternatif digunakan sebagai kaedah untuk menentukan reka bentuk yang akan dibangunkan. Matriks alternatif juga digunakan untuk mengurus dan menyusun reka bentuk alternatif supaya penyelesaian yang terbaik dapat diperolehi. Matriks alternatif menggabungkan beberapa analisa kebolehlaksanaan seperti berikut:

- a) Kebolehlaksanaan Teknikal - penilaian kematangan atau keupayaan teknologi untuk berfungsi dengan teknologi yang lain.
- b) Kebolehlaksanaan Operasi - kesesuaian dan kesesuaian pihak pengurusan, pegawai dan pengguna dengan teknologi yang dicadangkan.
- c) Kebolehlaksanaan Ekonomi - penilaian sama ada teknologi yang digunakan berpatutan dan kos efektif.

Bagi setiap kategori *tools* yang diperlukan, analisis matriks alternatif perlu dijalankan. Ini kerana setiap kategori *tool* mempunyai kriteria-kriteria yang berbeza. Matriks alternatif boleh disediakan seperti dalam contoh di bawah. Tahap/Pemberat dan skor-skor dirangkumi bersekali bagi mewujudkan kad skor yang mengenalpasti kriteria-kriteria utama projek dan pilihan alternatif yang terbaik.

Contoh analisis matriks alternatif adalah seperti berikut:

Sebuah organisasi ingin membuat pilihan yang terbaik di antara bahasa pengaturcaraan yang akan digunakan dalam sistem aplikasi yang ingin dibangunkan. Sebanyak 10 kriteria telah ditentukan bagi membantu organisasi dalam membuat pemilihan tersebut. Terdapat tiga jenis skema penilaian yang merangkumi nilai skor dari 1 sehingga 10 bertujuan untuk menyediakan kad skor dalam menentukan alternatif yang terbaik. Jadual matriks alternatif

seperti di bawah digunakan untuk membantu organisasi berkenaan melakukan penilaian kepada bahasa pengaturcaraan yang sesuai. Bahasa pengaturcaraan **Java** telah dikenalpasti sebagai pilihan yang terbaik bagi sistem aplikasi yang ingin dibangunkan.

Jadual 34 : Nilai Skor Skema Penilaian Teknikal

Skema Penilaian	
Tinggi	8 sehingga 10
Sederhana	5 sehingga 7
Rendah	1 sehingga 4

Jadual 35 : Matriks Alternatif Bagi Penentuan Bahasa Pengaturcaraan

No.	Kriteria	Bahasa Pengaturcaraan		
		Java	C#	Phyton
		Nilai Skor		
1.	Pengetahuan, pengalaman dan kemahiran pasukan pembangun	8	6	1
2.	Ketersediaan (<i>availability</i>) pengaturcara	9	5	1
3.	Ketersediaan <i>Integrated Development Environment</i> (IDE) dan <i>tools</i> di pasaran	8	8	6
4.	Kemudahan integrasi	8	8	8
5.	Penjimatan kos	9	6	9
6.	Prestasi	8	8	8
7.	Sekuriti	8	8	8
8.	Sokongan dan komuniti	8	9	6
9.	Keanjalan bahasa pengaturcaraan	8	8	8
10.	Tren semasa	9	6	4
JUMLAH		83	72	59

4.7 Reka bentuk Pangkalan Data [F3.3]

Keterangan

Reka bentuk Pangkalan Data Logikal merupakan aktiviti untuk menterjemah model maklumat konseptual kepada model maklumat logikal. Model maklumat logikal merupakan model perantara yang akan digunakan untuk mereka bentuk pangkalan data fizikal, seterusnya membangunkan pangkalan data yang sebenar. Model ini menerangkan empat (4) komponen utama iaitu spesifikasi jadual (*table specification*), spesifikasi medan (*column specification*), spesifikasi kekunci primer (*primary key specification*) dan spesifikasi kekunci asing (*foreign key specification*).

Peristilahan yang digunakan dalam penterjemahan model maklumat konseptual kepada model maklumat logikal adalah seperti berikut.

Model Konseptual		Model Logikal
Entiti	→	Jadual
Atribut	→	Medan
UID Primer	→	Kekunci Primer
UID Sekunder	→	Kekunci Unik
Hubungan antara entiti	→	Kekunci Asing
Peraturan bisnes	→	<i>Check Constraint</i>

Objektif

- Menyediakan model maklumat logikal.
- Mengenalpasti spesifikasi jadual dan medan.
- Mengenalpasti spesifikasi kekunci utama dan kekunci asing.
- Implementasi entiti berjenis *Super-type* dan *Sub-type*.

Notasi

Model maklumat logikal adalah berdasarkan model maklumat konseptual rujuk **Pemodelan Keperluan Data [F2.2]**:

Jadual 36 : Notasi Reka bentuk Keperluan Data

Elemen	Keterangan
<p>Jadual</p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> <p style="text-align: center;">NAMA_JADUAL</p> </div>	<ul style="list-style-type: none"> • Digunakan bagi mewakili setiap jadual.

Medan				
NAMA_JADUAL				
jenis kekunci	simbol *	nama medan	jenis data	Panjang data

- Jenis Kekunci
Terdapat beberapa jenis kekunci bagi medan yang terdapat dalam jadual. Antaranya ialah:
P – Kekunci primer (*Primary key*)
U – Kekunci unik (*Unique key*)
F – Kekunci asing (*Foreign key*)
- Simbol * menunjukkan medan wajib diisi (*not null column*) manakala sekiranya tiada simbol * menunjukkan medan tidak wajib diisi (*null column*).
- Jenis data bagi setiap medan boleh sama ada dalam format numerik, alfanumerik, aksara, tarikh, masa atau fail (seperti fail imej/audio/video/multimedia).
- Panjang (*length*) data bagi jenis data tersebut dinyatakan dalam tanda kurungan '()' – jika ada.

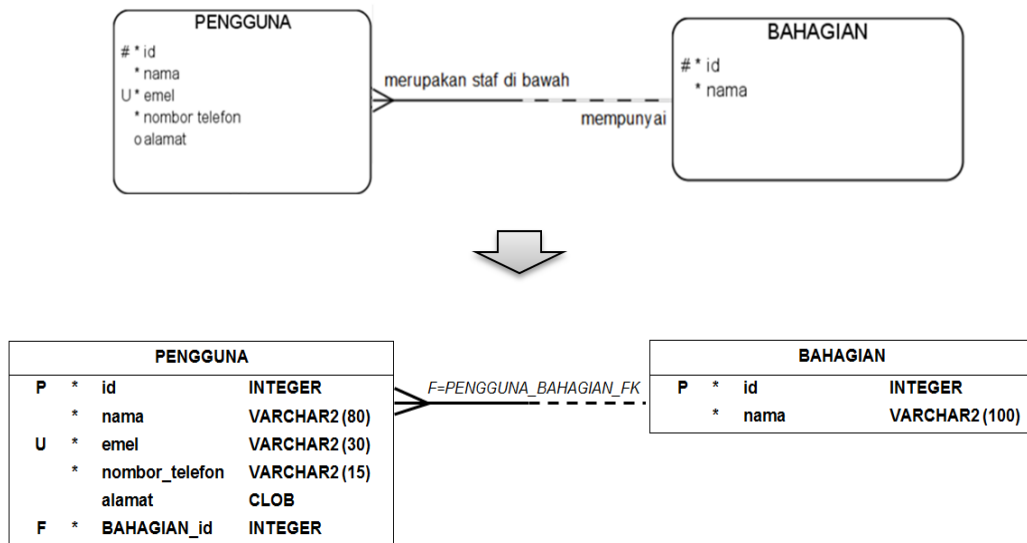
Langkah-Langkah

Langkah 1 : Sediakan Spesifikasi Jadual

- Kenalpasti entiti yang mempunyai maklumat yang tidak perlu disimpan dalam pangkalan data terlebih dahulu. Entiti tersebut tidak perlu diterjemahkan ke dalam model maklumat logikal.
- Setiap entiti tunggal (bukan berjenis Entiti *Super-type* atau *Sub-type*) akan diterjemahkan terus kepada jadual. Contoh penterjemahan adalah seperti berikut.

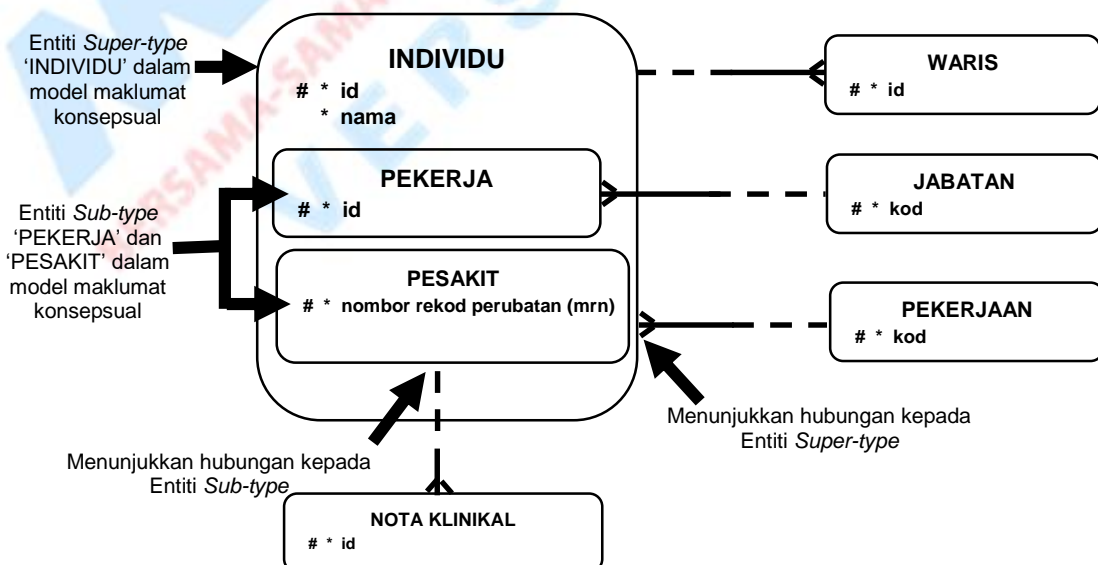
Model Konseptual		Model Logikal
Nama Entiti		Nama Jadual
PENGGUNA	→	PENGGUNA
ASET	→	ASET
KATEGORI ASET	→	KATEGORI_ASET
TEMPAHAN ASET	→	TEMPAHAN_ASET

Contoh entiti tunggal yang diterjemahkan terus kepada jadual adalah seperti di rajah dibawah.



Rajah 63 : ERD - Entiti Tunggal

- c) Sekiranya *Intersection Entity* diwujudkan bagi menyelesaikan hubungan banyak-ke-banyak (*many-to-many*) di antara dua entiti, terjemahkan model tersebut kepada model maklumat logikal adalah sama seperti entiti tunggal.
- d) Sekiranya Entiti Super-type atau Sub-type wujud, penterjemahan ke model logikal akan membabitkan hubungan dengan entiti lain sama ada kepada Entiti *Super-type* atau *Sub-type* tersebut.



Rajah 64 : ERD – Penggunaan Entiti Supertype

Terdapat **tiga pilihan** untuk menterjemahkan model maklumat konseptual ini kepada model maklumat logikal iaitu :

i) Pilihan 1 – Pelaksanaan *Super-type*

Pilihan ini akan menghasilkan satu jadual tunggal sahaja bagi pelaksanaan ketiga-tiga Entiti INDIVIDU, PEKERJA dan PESAKIT. Pelaksanaan ini juga dikenali sebagai **pelaksanaan satu jadual tunggal**.

ii) Pilihan 2 – Pelaksanaan *Sub-type*

Pilihan ini akan menghasilkan **satu jadual bagi setiap Entiti Sub-type**. Bagi contoh ini, dua jadual akan dihasilkan iaitu Jadual PEKERJA dan PESAKIT.

iii) Pilihan 3 – Pelaksanaan kedua-dua *Super-type* dan *Sub-type* (“Arc”)

Pilihan ini akan menghasilkan **satu jadual bagi setiap entiti** sama ada entiti tersebut berjenis *Super-type* atau *Sub-type*. Bagi contoh ini, tiga jadual akan dihasilkan iaitu Jadual INDIVIDU, PEKERJA dan PESAKIT

- e) Nama jadual mesti dimulakan dengan huruf. Gantikan ruang kosong atau aksara khas yang tidak dibenarkan seperti %, *, –, !, / dan lain-lain kepada aksara garis bawah ‘_’ (underscore). Elakkan penggunaan perkataan-perkataan rizab (reserved words) yang biasa terdapat dalam bahasa pengaturcaraan seperti *number*, *value*, *type* dan lain-lain.
- f) Nama jadual mestilah unik (nama yang sama tidak boleh digunakan berulang kali) dalam satu skema pangkalan data.

Langkah 2 : Sediakan Spesifikasi Medan

- a) Setiap atribut akan diterjemahkan kepada medan (*field*).
- b) Nama atribut akan menjadi nama medan (*field*). Nama medan mesti dimulakan dengan huruf. Gantikan ruang kosong/aksara khas yang tidak dibenarkan kepada aksara garis bawah ‘_’. Elakkan penggunaan perkataan-perkataan rizab, dan beri nama singkatan jika boleh.
- c) Nama medan mestilah unik (nama yang sama tidak boleh digunakan berulang kali) dalam satu jadual.
- d) Atribut Mandatori akan menjadi medan wajib diisi (*not-null column*), manakala Atribut Pilihan menjadi medan tidak wajib diisi (*null column*).
- e) Tentukan jenis data (*data type*) bagi setiap medan. Jenis data boleh sama ada dalam format numerik, alfanumerik, aksara, tarikh, masa atau fail (seperti fail imej/audio/video/multimedia). Nyatakan panjang (*length*) bagi jenis data tersebut dalam tanda kurungan ‘()’ – jika ada.
- f) Sesetengah peraturan bisnes diterjemahkan kepada CHECK Constraint bagi memastikan data yang sah sahaja diterima. Contohnya medan ‘tarikh_tamat_guna’

dalam jadual `TEMPAHAN_ASET` mestilah lebih besar atau sama dengan medan 'tarikh_mula_guna'.

1. tarikh_tamat_guna >= tarikh_mula_guna

- g) Bagi peraturan bisnes yang lebih kompleks, pengekodan/pengaturcaraan tambahan mungkin diperlukan sama ada di komputer pelayan (*server site*), di komputer pelanggan/klien (*client site*), atau kedua-duanya sekali.

Langkah 3 : Sediakan Spesifikasi Kekunci Primer

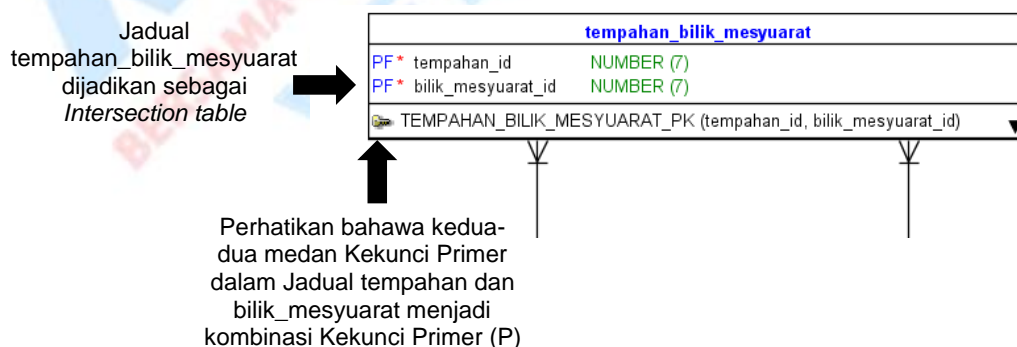
- a) Terjemahkan UID Primer menjadi Kekunci Primer (*Primary Key*).
- b) Terjemahkan UID Sekunder akan menjadi Kekunci Unik (*Unique Key*).

Langkah 4 : Sediakan Spesifikasi Kekunci Asing

- a) Hubungan antara dua entiti akan diterjemah menjadi Kekunci Asing (*Foreign Key*).
- b) Medan Kekunci Asing biasanya dinamakan dengan menggabungkan nama jadual yang dirujuk, dengan nama atribut yang menjadi Kekunci Primer dalam jadual yang dirujuk itu.

Langkah 5 : Sediakan Spesifikasi Medan *Intersection Entity*

- a) Bagi model yang mempunyai *Intersection Entity* yang menyelesaikan hubungan banyak-ke-banyak (*many-to-many*) di antara dua entiti, UID Primer daripada kedua-dua entiti akan digabungkan untuk menjadi UID Primer bagi *Intersection Entity* tersebut.



Rajah 65 : Kekunci Primer daripada *composite key*

- b) Dalam model logikal, gabungan UID Primer daripada kedua-dua entiti akan menjadi *Composite Primary key*.

- c) Merujuk kepada rajah di atas, medan-medan yang terdapat dalam *Intersection Table* 'ITEM_TEMPAHAN', hubungan antara dua entiti diterjemah menjadi Kekunci Asing F_1 dan F_2 , dan kedua-dua Kekunci Asing ini membentuk kombinasi Kekunci Primer (P) (*composite primary key*) dalam jadual berkenaan. Sama ada modaliti hubungan bersifat mandatori atau sebaliknya, medan Kekunci Asing dalam *Intersection Table* akan sentiasa menjadi medan wajib diisi (*not-null column*).

Langkah 6 : Sediakan Spesifikasi Medan Entiti Super-Type Dan Sub-Type

Terjemahan bagi atribut yang terlibat dalam Entiti *Super-type* atau *Sub-type*, terdapat beberapa peraturan bagi pilihan yang perlu dipatuhi mengikut jenis terjemahan model maklumat.

a) Pilihan 1 – Pelaksanaan *Super-type*



Rajah 66 : ERD – Penggunaan Entiti Supertype

- i) Atribut yang terdapat dalam Entiti *Super-type* diterjemahkan terus seperti biasa.
- ii) Kesemua atribut daripada Entiti *Sub-type* akan menjadi medan tidak wajib diisi (*null column*). Sekiranya atribut tersebut merupakan UID, atribut diterjemah menjadi Kekunci Unik (*Unique Key*) atau *CHECK Constraint*.
- iii) Satu medan mandatori **WAJIB** diwujudkan bagi membezakan antara Entiti *Sub-type* tersebut. Medan tersebut biasanya dinamakan dengan **<nama_entiti_supertype>_jenis@kategori** seperti **INDIVIDU_kategori** kerana medan inilah yang akan membezakan sama ada INDIVIDU tersebut kategori

PESAKIT atau PEKERJA. Penggunaan *CHECK Constraint* atau tambahan pengekodan dalam aturcara aplikasi boleh diaplikasikan di sini bagi menjamin integriti/kesahihan data.

- iv) Hubungan kepada Entiti *Super-type* juga diterjemahkan terus seperti biasa, manakala Hubungan kepada Entiti *Sub-type* akan diterjemah menjadi Kekunci Asing dan menjadi medan yang tidak wajib diisi (*null column*).

b) Pilihan 2 – Pelaksanaan Sub-type

Atribut yang terdapat dalam Entiti *Sub-type* diterjemahkan terus seperti biasa ke dalam jadual yang berasingan. Manakala Kekunci Primer bagi atribut yang terdapat dalam Entiti *Sub-type* adalah UID Primer Entiti *Super-type*.

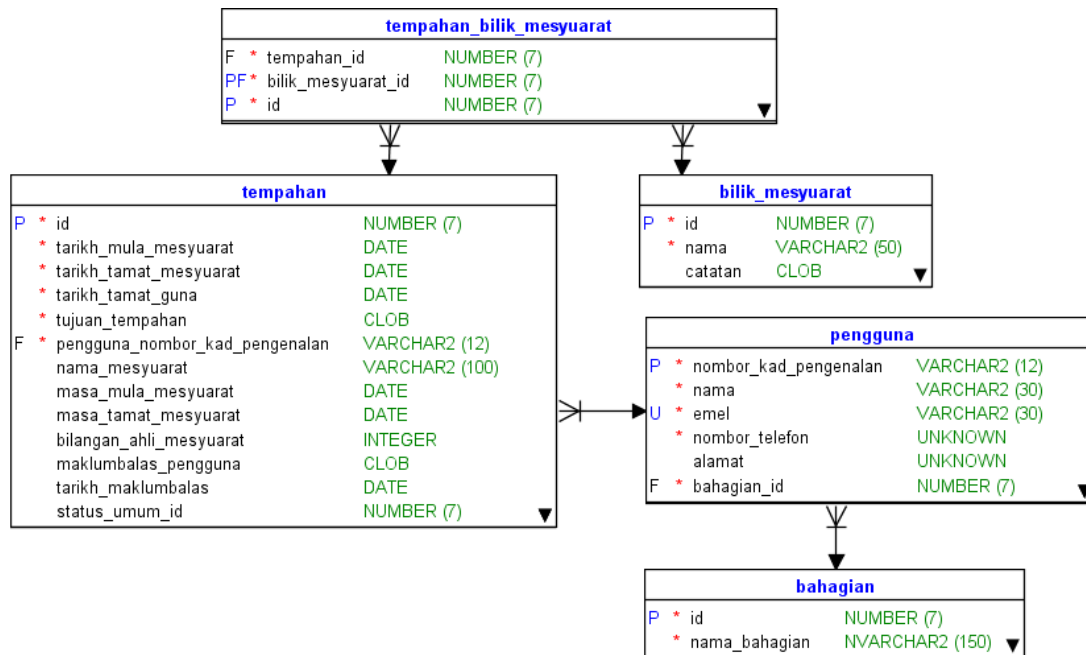
c) Pilihan 3 – Pelaksanaan kedua-dua Super-type dan Sub-type (Arc)

- i) Atribut yang terdapat dalam kesemua entiti tersebut diterjemahkan seperti biasa ke dalam jadual yang berasingan. Manakala Kekunci Primer bagi atribut yang terdapat dalam Entiti *Super-type* dan *Sub-type* adalah UID Primer Entiti *Super-type*.
- ii) Bagi Entiti *Super-type*, satu medan mandatori **WAJIB** diwujudkan bagi membezakan antara Entiti *Sub-type* tersebut. Medan tersebut biasanya dinamakan dengan *<nama_entiti_supertype>_jenis@kategori* seperti **INDIVIDU_kategori** kerana medan inilah yang akan membezakan sama ada INDIVIDU tersebut kategori PESAKIT atau PEKERJA. Penggunaan *CHECK Constraint* atau tambahan pengekodan dalam aturcara aplikasi boleh diaplikasikan di sini bagi menjamin integriti/kesahihan data.

Langkah 6 : Perkemaskan Model Maklumat Logikal

Setelah semua entiti siap diterjemah ke model logikal, lengkapkan dan perkemaskan lagi model maklumat logikal mengikut jenis teknologi yang hendak digunakan.

Contoh model maklumat logikal Sistem Tempahan Bilik Mesyuarat (eTempah) adalah seperti rajah berikut.



Rajah 67 : Contoh Model Maklumat Logikal Sistem

Langkah 7 : Dokumentasikan Model Maklumat Logikal

Dokumentasikan semua output yang dihasilkan sebagai hasil serahan proses reka bentuk pangkalan data logikal ke dalam **D04 Spesifikasi Reka bentuk Sistem**. Dokumentasi mengikut susunan berikut:

- Model Maklumat Logikal
- Rujuk **Apendiks 4 Template Skema Logikal Pangkalan Data** (*Database Logical Schema*).

Rujukan

- https://en.wikipedia.org/wiki/Data_modeling#Conceptual.2C_logical_and_physical_schemas.
- Jan Speelpenning, Patrice Daux and Jeff Gallus; Data Modeling and Relational Database Design (2001).

4.8 Reka bentuk Antaramuka Pengguna [F3.4]

Keterangan

Penyediaan Reka bentuk Antaramuka Pengguna (UI) adalah proses untuk menentukan kaedah interaksi di antara pengguna dengan sistem yang akan dibangunkan. Memberi keutamaan kepada reka bentuk antaramuka pengguna, khususnya kepada peningkatan *user experience* (UX), dapat menjadikan sesuatu aplikasi mudah untuk dilayari, efektif dan selesa untuk digunakan.

Di samping itu, antaramuka-antaramuka pengguna yang telah dibangunkan perlu dipadankan dengan medan-medan data di dalam jadual (*table*) pangkalan data. Pemetaan data ini bertujuan untuk memudahkan seseorang pembangun sistem mengetahui senarai medan data yang diperlukan bagi satu-satu antaramuka pengguna yang telah dibangunkan.

Objektif

- Membangunkan antaramuka pengguna yang berpandukan kepada rangka kerja, prinsip dan elemen asas UI/UX dan selaras dengan *trend* reka bentuk yang terkini.
- Menyediakan jadual rujukan bagi pemetaan di antara antaramuka pengguna dengan medan di dalam pangkalan data.

Langkah-langkah

Langkah 1 : Kenalpasti Dan Pilih Use Case

- a) Rujuk kepada rajah *Use Case* yang telah dibangunkan di dalam Pemodelan *Use Case* (Fungsian) [F2.1].
- b) Kenalpasti dan pilih setiap *Use Case* yang terlibat untuk menyediakan semua reka bentuk antaramuka pengguna yang terlibat.

Langkah 2 : Kenalpasti Aliran Data

- a) Rujuk kepada Rajah Aliran Data (DFD) yang telah disediakan di dalam Pemodelan Proses Sistem [F2.3].
- b) Kenalpasti aliran data yang keluar dan masuk dari fungsi yang berkaitan dengan reka bentuk antaramuka pengguna yang ingin dibangunkan

Langkah 3 : Kenalpasti Elemen-elemen Data

- a) Rujuk kepada rajah hubungan entiti (ERD) dan logikal pangkalan data yang telah disediakan di dalam Pemodelan Keperluan Data [F2.2] dan Reka bentuk Pangkalan Data [F3.4].

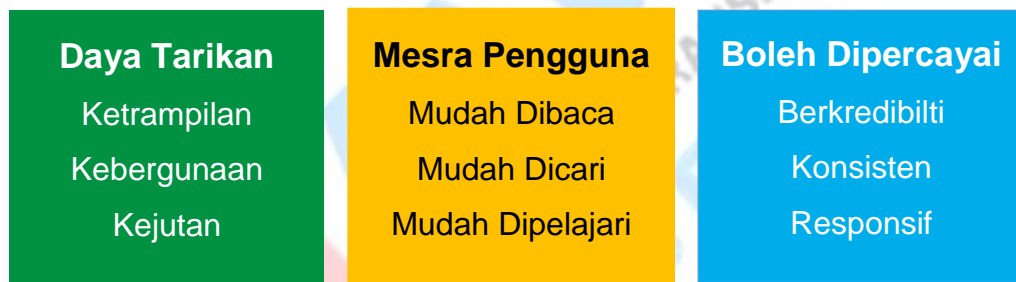
- b) Kenalpasti elemen-elemen data yang akan diguna pakai di dalam ruangan-ruangan teks reka bentuk antaramuka pengguna.

Langkah 4 : Pertimbangkan Keperluan *User Experience* (UX)

- a) Lakukan pertimbangan keperluan UX sebelum antaramuka-antaramuka pengguna dibangunkan. Pertimbangan keperluan UX ini boleh dilaksanakan dengan berpandukan kepada rangka kerja seperti berikut:

Rangka Kerja Ciri-ciri *User Experience* (UX)

Rangka Kerja Ciri-ciri UX merupakan panduan asas kepada pereka dan pembangun sistem dalam menyediakan reka bentuk antaramuka pengguna yang efektif serta memiliki tahap kebolegunaan yang tinggi. Rangka kerja ini mengambil kira kajian-kajian yang telah dilaksanakan berkenaan dengan *human-computer interaction* (HCI), *human factors* (HFs) dan *user-centered design* (UCD). Rangka kerja yang dimaksudkan adalah seperti berikut :-



Rajah 68 : Rangka Kerja Ciri-ciri *User Experience*

- i) Daya Tarikan

Daya tarikan adalah merupakan keupayaan untuk menarik perhatian dan membangkitkan minat pengguna untuk melayari aplikasi yang dibangunkan. Ciri-cirinya merangkumi kepada ketrampilan sesuatu antaramuka pengguna, kebolegunaan sesuatu aplikasi dan persembahan elemen-elemen kejutan kepada pengguna.

- Keterampilan

Keterampilan visual akan mempengaruhi kepada tanggapan dan ekspektasi pengguna kepada sesuatu aplikasi. Penekanan kepada warna, *font*, imej dan susun atur yang kemas dan konsisten mampu untuk meningkatkan keselesaan pengguna kepada aplikasi yang dilayari. Trend-trend terkini yang selaras dengan Reka bentuk *Flat 2.0*, seperti reka bentuk material yang dipopularkan oleh Syarikat Google dan Metro UI yang diguna pakai di dalam sistem pengoperasian Windows, boleh dijadikan sebagai rujukan dalam usaha untuk mempertingkatkan ketrampilan visual aplikasi yang dibangunkan.

- Kebergunaan

Kebergunaan antaramuka yang dibangunkan akan menentukan sama ada aplikasi akan terus dan kerap digunakan oleh pengguna. Aplikasi yang dibangunkan perlu sentiasa menitik beratkan nilai praktikaliti serta dapat membantu pengguna melaksanakan tugas dengan lancar.

- Kejutan

Inovasi di dalam reka bentuk antaramuka yang melangkaui ekspektasi pengguna mampu untuk meningkatkan daya tarikan sesuatu aplikasi. Inovasi di dalam reka bentuk antaramuka bukan sahaja tertakluk kepada kemajuan perisian terkini, malah kemudahan-kemudahan yang dimiliki oleh perkakasan pintar seperti kemudahan GPS atau kamera pada telefon bimbit boleh juga diintegrasikan dengan aplikasi untuk menjadikan sesuatu reka bentuk antaramuka kelihatan lebih moden dan canggih.

ii) Mesra Pengguna

Mesra pengguna merupakan keupayaan pengguna untuk mempelajari dan memahami penggunaan sesuatu aplikasi. Ciri-cirinya merangkumi kebolehan sesuatu aplikasi untuk memudahkan pengguna membaca kandungannya, melakukan carian dan mempelajari penggunaannya.

- Mudah Dibaca

Pereka bentuk perlulah terdahulu mengenal pasti kategori dan jenis pengguna aplikasi sebelum sesuatu reka bentuk antaramuka dibangunkan. Pengguna-pengguna aplikasi boleh terdiri sama ada daripada orang awam, kakitangan kerajaan, kumpulan profesional, golongan belia ataupun muda. Oleh yang demikian, seseorang pereka bentuk perlulah bijak dalam menentukan saiz dan warna *font* yang sesuai serta cerdik dalam menyusun atur kandungan aplikasi dengan kemas dan tidak mengelirukan.

- Mudah Dicari

Kebolehcarian sesuatu aplikasi diukur melalui bagaimana mudahnya seseorang pengguna melakukan carian kepada sesuatu maklumat yang tertentu. Contohnya, antaramuka yang dibangunkan perlulah dilengkapi dengan fungsi carian yang pintar, serta susun atur kandungan dan menu perlu diletakkan di ruangan yang strategik.

- Mudah Dipelajari

Reka bentuk antaramuka yang efisien perlu mempunyai konsep dan aliran kerja yang konsisten bagi membolehkan aplikasi mudah untuk difahami dan dipelajari oleh pengguna. Penyediaan manual pengguna atau fungsi bantuan

berbentuk interaktif bukan sahaja dapat membantu, malah dapat mempercepatkan lagi proses pembelajaran bagi seseorang pengguna.

iii) Boleh Dipercayai

Boleh dipercayai merupakan keupayaan sesuatu aplikasi untuk mendapatkan kepercayaan daripada penggunanya. Ciri-cirinya merangkumi kebolehan aplikasi untuk meningkatkan imej kredibiliti organisasi yang memiliki aplikasi berkenaan (*owner*), menunjukkan konsistensi di dalam reka bentuk antaramukanya serta memiliki tahap responsif yang tinggi.

- Berkredibiliti

Kualiti dan pemilihan visual yang bersesuaian bagi antaramuka pengguna sesuatu aplikasi, contohnya dari segi gambar yang dipaparkan, ikon yang digunakan serta susun atur yang kemas, mampu untuk meningkatkan imej kredibiliti sesuatu organisasi dari perspektif pengguna. Pemilihan dan kawalan kualiti visual ini bukan sahaja bertujuan untuk menarik perhatian pengguna, malah ia juga menggambarkan tahap profesional organisasi yang memiliki aplikasi berkenaan.

- Konsisten

Elemen-elemen antarmuka pengguna seperti warna, *font*, susun atur dan label-label perlu diselaraskan bagi mencapai aplikasi yang konsisten. Penyelarasan antarmuka pengguna juga dapat membantu pengguna untuk memahami dan mempelajari penggunaan aplikasi dengan lebih pantas.

- Responsif

Kelajuan maklumbalas dan prestasi sesuatu aplikasi perlu dititik beratkan di dalam reka bentuk antarmuka pengguna. Saiz imej yang dipaparkan serta teknologi antarmuka yang dipilih perlu bersesuaian dengan kebolehan infrastruktur sistem dan selari dengan keperluan yang diperolehi.

Langkah 5 : Bangunkan Reka bentuk Antaramuka Pengguna

Bangunkan semua reka bentuk antaramuka pengguna untuk memenuhi setiap *use case* yang telah dikenalpasti di dalam fasa analisa dengan berpandukan kepada Rangka Kerja Ciri-ciri *User Experience* (UX) termasuk Prinsip dan Elemen Asas Reka bentuk Antaramuka Pengguna seperti berikut:

Prinsip Dan Elemen Asas Reka bentuk Antaramuka Pengguna (UI)

Prinsip dan Elemen Asas Reka bentuk Antaramuka Pengguna adalah seperti berikut:

a) **Garisan**

Garis (*line*) merupakan asas pembentukkan semua komponen dalam reka bentuk visual. Penggunaan garisan di dalam antaramuka pengguna adalah bertujuan untuk memberi penekanan dan menarik perhatian pengguna kepada ruangan-ruangan tertentu, contohnya garisan disertakan dalam satu-satu muka web untuk memisah dan menonjolkan ruangan seperti kandungan, tajuk, label, pautan dan panel sisi.

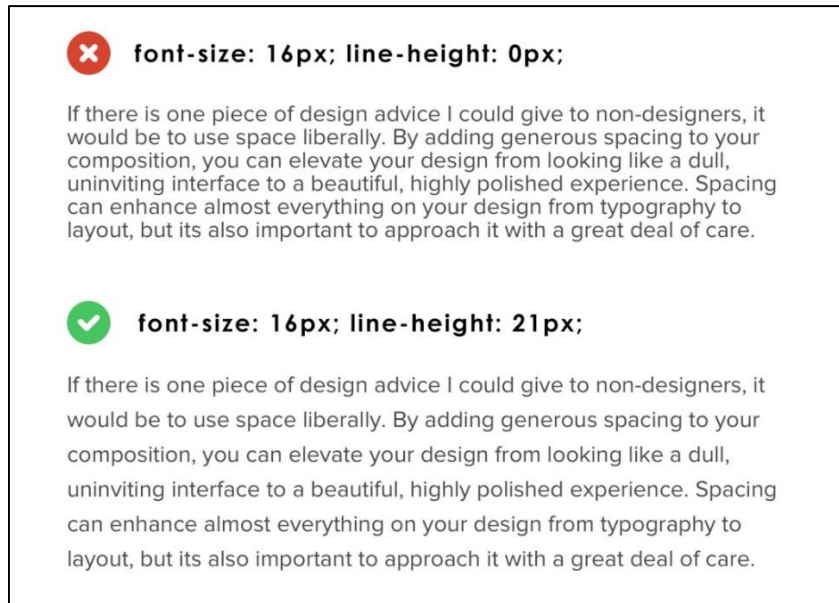
b) Warna

Penerapan warna di dalam antaramuka pengguna adalah bertujuan untuk mencetus suasana dan emosi yang spesifik kepada pengguna, malah ia juga merupakan satu mekanisme penceritaan berkenaan dengan satu-satu organisasi, produk dan perkhidmatan yang terlibat. Penggunaan warna di dalam antaramuka pengguna boleh berdasarkan kepada peraturan 60-30-10, di mana 60% ruangan dalam satu-satu muka web perlu mengandungi penggunaan warna dominan, 30% merupakan warna sekunder dan 10% pula adalah warna *accent*. Namun begitu, pematuhan kepada peraturan ini bukanlah mandatori oleh kerana bilangan dan jumlah peratusan warna yang ingin digunakan adalah bergantung kepada kompetensi dan kreativiti seseorang pereka antaramuka dalam mengharmoni serta mengimbangi komposisi warna-warna yang dipilih.

c) Tipografi

Tipografi merupakan elemen utama antaramuka untuk menyampaikan mesej dan maklumat kepada pengguna. Oleh yang demikian, pereka antaramuka perlulah memilih *typeface* atau *font family* yang sesuai mengikut ruangan-ruangan di dalam satu-satu antaramuka pengguna, sama ada ruangan tersebut merupakan tajuk, logo, label dan teks kandungan. *Font family* yang dipilih secara amnya perlu mudah untuk dibaca, jelas dan bersesuaian dengan tema yang ingin disampaikan. Sebagai contoh, *font family* yang standard seperti Arial, Sans Serif, Calibri dan Trebuchet MS sesuai digunakan bagi antaramuka yang berteraskan kepada aplikasi bisnes dan profesional. Bagi ruangan-ruangan yang ingin ditonjolkan seperti tajuk atau logo, pereka antaramuka boleh mempraktikkan kreativiti masing-masing dengan menggunakan *font family* yang lebih progresif dan unik bertujuan untuk menarik perhatian pengguna kepada aplikasi yang dibangunkan. Namun begitu, perlu diingatkan bahawa bilangan jenis *font family* dalam antaramuka pengguna tidak seharusnya digunakan secara berlebihan. Secara amnya, jumlah bilangan *font family* yang dipilih dalam satu-satu antaramuka aplikasi tidak seharusnya melebihi dari tiga jenis.

Salah satu aspek lain yang perlu juga dilihat dalam tipografi adalah jarak di antara satu baris ayat dengan baris ayat seterusnya, atau terma teknikalnya lebih dikenali sebagai *line-height*. Pereka antaramuka perlulah menyediakan jarak yang bersesuaian supaya kandungan maklumat atau label yang dipaparkan tidaklah terlalu rapat dan menyukarkan untuk dibaca oleh pengguna. Secara amnya, jumlah jarak di antara baris ayat atas dan bawah adalah 30% lebih besar dari saiz *font* yang ditetapkan. Contohnya, sekiranya saiz *font* yang diguna pakai adalah 16 piksel, nilai *line-height* atau jarak di antara baris adalah sebanyak 21 piksel. Rujuk rajah di bawah bagi melihat contoh perbezaan di antara penggunaan *line height* yang ideal dalam antaramuka pengguna.



Rajah 69 : Penggunaan *Line-Height* Yang Berkesan Dalam Antaramuka Pengguna

d) Ruang Negatif (*Negative Space*)

Ruang negatif, atau juga dikenali sebagai *white space*, adalah ruangan yang dibiarkan kosong di antara satu komponen antaramuka pengguna dengan komponen-komponen yang lain. Ruang negatif diimplementasi bertujuan untuk menarik fokus pengguna kepada maklumat atau mesej yang ingin ditonjolkan, memudahkan pengguna berinteraksi dan melakukan navigasi dalam aplikasi, serta ia dapat meningkatkan kekemasan satu-satu antaramuka. Penggunaan ruang negatif yang efisien adalah sejajar dengan pendekatan dan trend reka bentuk minimalis yang kini kian popular digunakan.

Penggunaan ruang negatif yang efisien boleh dilaksanakan dengan meningkatkan saiz margin di antara komponen-komponen antaramuka seperti gambar, logo, tajuk, label dan kotak teks. Dalam masa yang sama, komponen-komponen, informasi dan kandungan yang disertakan dalam setiap antaramuka tidak perlulah terlalu banyak sehingga membuatkan skrin tersebut menjadi terlalu sesak dan sukar untuk dibaca. Rujuk rajah di bawah untuk melihat contoh penggunaan ruang negatif yang efektif dalam satu-satu muka web atau aplikasi.

Rajah 70 : Contoh Penggunaan Ruang Negatif Yang Berkesan

Berikut adalah satu contoh reka bentuk antaramuka pengguna bagi menu Kemaskini Profil Pengguna di bawah Sistem Tempahan Bilik Mesyuarat (eTempah):

Rajah 71 : Contoh Reka bentuk Antaramuka Pengguna Bagi Menu Luluskan Tempahan Bilik Mesyuarat, Sistem Tempahan Bilik Mesyuarat (eTempah)

Langkah 5 : Padankan Reka bentuk Antaramuka Pengguna Dengan Elemen Data

- Padankan reka bentuk antaramuka pengguna dan medan data yang berkaitan dengan melengkapkan Apendiks 6 Templat Pemetaan Data Antaramuka seperti di.
- Selain daripada medan data, masukkan juga objek-objek lain, seperti butang-butang atau pautan, di dalam Jadual Pemetaan Data. Gunakan notasi '<Nama Objek>' apabila merekodkan objek berkenaan. Contoh penulisan menggunakan notasi objek butang adalah seperti <Hantar>, <Simpan> dan <Keluar>.
- Isikan maklumat-maklumat berikut:

Jadual 37 : Keterangan Medan-Medan Templat Pemetaan Data

Medan	Keterangan
Nama Label	Nama label bagi elemen-elemen seperti <i>textbox</i> , <i>textarea</i> atau <i>dropdown menu</i> di dalam reka bentuk antaramuka pengguna.
Nama Jadual	Nama jadual (<i>table</i>) di mana medan data disimpan bagi elemen di dalam reka bentuk antaramuka pengguna.
Nama Medan Data	Nama medan data yang terlibat bagi elemen di dalam reka bentuk antaramuka pengguna.
Create, Update, Read, Delete (CRUD)	<p>CRUD merujuk kepada empat fungsi utama yang dilaksanakan dalam pangkalan data. Isikan jenis fungsi utama berdasarkan keperluan proses dan reka bentuk antaramuka yang telah dibangunkan. Penerangan lanjut berkenaan fungsi-fungsi CRUD adalah seperti berikut :</p> <ol style="list-style-type: none"> <i>Create</i> (C) adalah merujuk kepada data yang baru diwujudkan atau direkodkan <i>Read</i> (R) adalah merujuk kepada data yang dikeluarkan untuk paparan <i>Update</i> (U) adalah merujuk kepada data yang telah wujud sebelum ini dan hanya perlu dikemaskini. <i>Delete</i> (D) adalah merujuk kepada data yang ingin dihapuskan

Catatan	Ruangan catatan adalah bertujuan untuk memasukkan keterangan tambahan ataupun peraturan yang berkenaan bagi medan-medan data yang terlibat. Ruangan ini adalah bukanlah mandatori dan boleh digabungkan dengan medan-medan data yang lain sekiranya catatan yang dimasukkan mempunyai ciri-ciri yang sama.
----------------	--

- d) Rujuk jadual di bawah bagi contoh pengisian Templat Pemetaan Data Bagi Menu Luluskan Tempahan Bilik Mesyuarat, Sistem Tempahan Bilik Mesyuarat (eTempah) berdasarkan kepada contoh antaramuka pengguna yang telah disediakan pada langkah 4.

Jadual 38 : Contoh Pengisian Templat Pemetaan Data

Nama Label	Jenis Objek	Nama Jadual	Nama Medan Data	CRUD	Catatan
Nama Mesyuarat	Data	tempahan	nama_mesyuarat	R	
Nama Bilik Mesyuarat	Data	tempahan tempahan_bilik_mesyuarat bilik_mesyuarat	id tempahan_id_bilik_mesyuarat_id id nama	R	Paparkan semua senarai bilik dalam satu-satu tempahan
Tarikh Mula Mesyuarat	Data	tempahan	tarikh_mula_mesyuarat	R	Format tarikh adalah dalam bentuk 'DD/MM/YYYY'.
Tarikh Tamat Mesyuarat	Data	tempahan	tarikh_tamat_mesyuarat	R	Format tarikh adalah dalam bentuk 'DD/MM/YYYY'.
Masa Mula Mesyuarat	Data	tempahan	masa_mula_mesyuarat	R	Format tarikh adalah dalam bentuk 'DD/MM/YYYY'.
Masa Tamat Mesyuarat	Data	tempahan	masa_tamat_mesyuarat	R	Paparan masa adalah dalam format 12-jam.
Bil. Ahli Mesyuarat	Data	tempahan	bil_ahli_mesyuarat	R	
Ditempah Oleh	Data	tempahan pengguna	no_kad_pengenalan no_kad_pengenalan nama	R	

Status Tempahan	Data	tempahan status_umum	status_umum_id id nama	C/R/U	
<Simpan>	Butang			C/U	Rekod atau pinda maklumat di dalam jadual yang berkaitan setelah butang Simpan diklik. Keluarkan notifikasi <i>pop-up</i> bagi mengesahkan pengguna ingin melakukan operasi simpan.
<Keluar>	Butang				Batalkan segala operasi dalam menu. Pengguna akan dikembalikan ke muka web/borang yang sebelumnya.

Langkah 6 : Dokumentasikan Antaramuka Pengguna dan Jadual Pemetaan Data

Dokumentasikan semua antaramuka pengguna dan Jadual Pemetaan Data yang telah disediakan ke dalam **D04 Spesifikasi Reka bentuk Sistem**.

Rujukan

1. Danny Brian. Attributes of a Great Web User Experience (2013).
<https://www.gartner.com/document/2447715>
2. Danny Brian. Building a Great User Experience (2012).
<https://www.gartner.com/document/2251816>
3. Ashley Gainer. Tips For Adding White Space To Your Website (2015).
<https://getflywheel.com/layout/tips-for-adding-white-space-to-your-website/>
4. Maryam Taheri. 10 Basic Elements of Design (2018).
<https://creativemarket.com/blog/10-basic-elements-of-design>
5. Mary Stribley. Design Elements & Principles.
<https://www.canva.com/learn/design-elements-principles/>

4.9 Reka bentuk Transaksi Sistem [F3.5]

Keterangan

Reka bentuk Transaksi Sistem merupakan spesifikasi terperinci sesuatu *Use Case*. Ianya juga dipanggil Senario *Use Case*. Senario *Use Case* terdiri daripada langkah-langkah aktiviti yang akan dilakukan dalam sesuatu *Use Case* dengan disokong oleh reka bentuk antaramuka pengguna dan syarat-syarat/kekangan bagi setiap langkah yang dinyatakan. Ia akan dijadikan sebagai panduan kepada pasukan pembangunan untuk membangunkan kod pengaturcaraan yang lebih tersusun bagi mencapai hasil yang diperlukan.

Objektif

Membangun Senario *Use Case* bagi setiap *Use Case* selari dengan reka bentuk antaramuka pengguna yang telah dibangunkan.

Langkah-langkah

Langkah 1 : Penerangan Ringkas *Use Case*

- a) Berdasarkan Rajah *Use Case* yang telah dibangunkan di dalam Pemodelan *Use Case* [F2.1], pilih salah satu Rajah *Use Case* (Modul) sistem.
- b) Ambil salah satu *Use Case* dan terangkan secara ringkas berkaitan *Use Case*
- c) Kenalpasti aktor yang akan berinteraksi dengan *Use Case* yang dipilih
- d) Nyatakan prasyarat atau aktiviti yang perlu dilaksanakan terlebih dahulu sebelum *Use Case* yang terlibat dilaksanakan.

Contoh pra syarat bagi Mohon Tempahan Bilik Mesyuarat adalah:

“Pengguna yang sah berjaya log masuk sistem”.

Langkah 2 : Tentukan Langkah-langkah Aktiviti *Use Case*

- a) Tentukan langkah-langkah aktiviti *Use Case* yang dipilih.
- b) Bagi setiap langkah aktiviti, kenalpasti apakah input (butiran), antaramuka pengguna dan keperluan/syarat-syarat/ kekangan.

Contoh :

Rajah *Use Case* Modul Tempahan Bilik Mesyuarat terdapat 9 *Use Case*, maka sebanyak 9 Senario *Use Case*. Ambil *Use Case* pertama iaitu **UC-BM-MT-TBM-01 Mohon Tempahan Bilik Mesyuarat** dan senaraikan aktiviti yang terlibat dalam memohon tempahan.

Langkah aktiviti Mohon Tempahan Bilik Mesyuarat:

- i) Kemasukan profil pengguna untuk dihubungi
- ii) Pengguna cari kekosongan bilik berdasarkan Tarikh yang diperlukan
- iii) Sistem akan paparkan senarai bilik mesyuarat dan kemudahan yang disediakan
- iv) Pengguna Pilih bilik mesyuarat yang dikehendaki dan tempoh penggunaan
- v) Pengguna hantar permohonan tempahan

Antaramuka Pengguna yang berkaitan:

UI-TBM-01 : Permohonan Tempahan Bilik Mesyuarat

PERMOHONAN TEMPAHAN BILIK MESYUARAT
✕

Butiran pegawai untuk dihubungi:

Nama :

Emel :

Carian kekosongan bilik

Tarikh Mula : Tarikh Tamat : Semak

Senarai Bilik Mesyuarat

	Nama Bilik	Kapasiti	Status
<input checked="" type="checkbox"/>	Bilik Mesyuarat Cyber 1	30 tempat	free
<input type="checkbox"/>	Bilik Bincang Cyber 3	15 tempat	free
<input type="checkbox"/>	Bilik Bincang Cyber Utara 5	15 tempat	free

Hantar

Rajah 72 : Contoh Antaramuka Pengguna Tempahan Bilik Mesyuarat

Langkah 3 : Kenalpasti Pasca Syarat Use Case

Nyatakan pasca syarat selepas semua langkah aktiviti *Use Case* dilaksanakan. Contoh pasca syarat *Use Case* Permohonan Tempahan Bilik Mesyuarat adalah:

“Permohonan tempahan bilik mesyuarat telah dihantar notifiaksi kepada pelulus bagi kelulusan”

Langkah 4 : Nyatakan Proses Alternatif Use Case

Proses alternatif merupakan proses pilihan kepada sesuatu langkah aktiviti yang tidak dapat dilaksanakan secara normal. Contohnya, sekiranya tempahan online tidak dapat dilaksanakan maka pengguna boleh dilakukan secara emel dengan menyatakan tarikh, nama bilik mesyuarat.

Langkah 5 : Sedia Dan Lengkapkan Templat Senario Use Case

- Lengkapkan dengan terperinci **Apendiks 7 Templat Senario Use Case** bagi setiap aktiviti *Use Case* yang telah dikenalpasti.
- Maklumat-maklumat Senario *Use Case* adalah seperti berikut :

Jadual 39 : Keterangan Medan-medan Templat Senario Use Case

Medan	Keterangan
Rujukan <i>Use Case</i>	Rujukan bagi setiap aktiviti <i>Use Case</i> berdasarkan konvesyen nama dan nombor yang selaras.
Nama <i>Use Case</i>	Nama bagi aktiviti <i>Use Case</i> yang terlibat berdasarkan Rajah <i>Use Case</i> yang telah dibangunkan.
Keterangan	Keterangan secara ringkas aktiviti <i>Use Case</i> yang terlibat.
Pra Syarat	Syarat atau operasi yang perlu dilaksanakan dahulu sebelum aktiviti yang terlibat dilaksanakan.
Aktor	Aktor yang terlibat dengan aktiviti <i>Use Case</i> berkenaan.
Input	Maklumat atau/dan dokumen yang diperlukan bagi setiap proses di dalam aktiviti <i>Use Case</i> yang terlibat.
Proses	Langkah untuk menavigasi dan melaksanakan operasi berdasarkan reka bentuk antaramuka pengguna yang berkaitan.
Rujukan Reka bentuk Antaramuka	Nama dan nombor rujukan reka bentuk antaramuka pengguna yang terlibat dengan aktiviti <i>Use Case</i> berkenaan.
Keperluan Keterangan / Syarat / Kekangan	Keterangan lanjut atau syarat tambahan atau kekangan yang dihadapi untuk melaksanakan operasi di dalam reka bentuk antaramuka pengguna yang berkaitan.

Pasca Syarat	Syarat atau operasi yang menyusuli selepas aktiviti <i>Use Case</i> berkenaan selesai dilaksanakan.
Proses Alternatif	Proses alternatif sekiranya aktiviti <i>Use Case</i> berkenaan tidak dapat dilakukan.

c) Sila rujuk jadual di bawah bagi contoh pengisian Jadual Senario *Use Case*.

Jadual 40 : Pengisian Templat Senario *Use Case*

Rujukan <i>Use Case</i>	UC-BM-TM-TBM-01		
Nama <i>Use Case</i>	Mohon Tempahan Bilik Mesyuarat		
Keterangan	Transaksi bagi memohon tempahan penggunaan bilik Mesyuarat		
Pra Syarat	Pengguna yang sah berjaya log masuk sistem		
Aktor	Pemohon		
Input	Langkah	Rujukan Antaramuka Pengguna	Keperluan Keterangan / Syarat / Kekangan
Nama dan emel pegawai untuk dihubungi	1. Kemasukan profil pengguna untuk dihubungi	UI-TBM-01	Secara <i>default</i> nama dan emel login yang dipaparkan. Nama & emel yang dikemaskini perlu divalidasi dengan senarai warga agensi. Sekira nama @ emel bukan warga agensi, mesej perlu dipaparkan
Tarikh Mula dan Tarikh Tamat	2. Pengguna cari kekosongan bilik berdasarkan Tarikh yang diperlukan. Masukkan Tarikh mula dan tamat dan tekan semak	UI-TBM-01	Perlu semak: a) Tarikh mula \geq Tarikh sistem semasa b) Tarikh tamat $>$ Tarikh mula
	3. Sistem akan paparkan senarai bilik mesyuarat dan	UI-TBM-01	Sistem akan menyemak semua aset bilik yang berstatus kekosongan dan paparkan butiran Nama, kapasiti dan status

	kemudahan yang disediakan		
	4. Pengguna memilih bilik mesyuarat yang dikehendaki dengan <i>tick</i> pada ruangan yang disediakan	UI-TBM-01	Pengguna boleh memilih lebih daripada satu bilik untuk sesuatu tempahan.
	5. Pengguna hantar permohonan tempahan	UI-TBM-01	Sistem perlu papar mesej sekiranya hantar permohonan tanpa tick pada senarai bilik.
Pasca Syarat	Sistem akan menghantar notifikasi kelulusan permohoann tempahan kepada pelulus.		
Proses Alternatif	Pengguna dapat mengemelkan tarikh dan kapasiti bilik yang diperlukan kepada pentadbir untuk tempahan.		

Langkah 4 : Dokumentasikan Senario Use Case

Dokumentasikan semua senario *Use Case* yang telah disediakan ke dalam **D04 Spesifikasi Reka bentuk Sistem**.

Rujukan

1. Dokumen Spesifikasi Reka bentuk Sistem (SDS) ePPAx.
2. Slaid Bootcamp Pembangunan Sistem Pasukan Perunding ICT MAMPU.

4.10 Penyediaan Spesifikasi Reka bentuk Sistem [F3.6]

Keterangan

Spesifikasi Reka bentuk Sistem (SDS) adalah penerangan terperinci berkenaan reka bentuk-reka bentuk arkitektur, fungsi sistem, pangkalan data, migrasi data dan integrasi data bagi sistem aplikasi yang akan dibangunkan. Dokumen SDS merupakan dokumen yang disediakan sebagai panduan utama kepada pasukan pengaturcara kepada senibina dan reka bentuk satu-satu sistem aplikasi.

Objektif

- Menyediakan Reka bentuk Arkitektur Keseluruhan Sistem Aplikasi, Arkitektur Aplikasi dan Arkitektur Pangkalan Data berdasarkan kepada langkah-langkah yang terkandung di dalam Reka bentuk Arkitektur [F3.1].
- Mengemaskini Model Fungsi Sistem yang telah dibangunkan di dalam Pemodelan Fungsi Sistem [F2.2] supaya selaras dengan keperluan reka bentuk sistem.
- Menyediakan Reka bentuk Transaksi Sistem, Reka bentuk Antaramuka Pengguna dan Pemetaan Data seperti yang dinyatakan dalam langkah-langkah di bawah Reka bentuk Transaksi Sistem [F3.5] dan Reka bentuk Antaramuka Sistem [F3.5].
- Menyediakan Reka bentuk serta Skema Pangkalan Data Logikal dengan merujuk kepada langkah-langkah di bawah Reka bentuk Pangkalan Data [F3.3].
- Menyertakan keterangan ringkas berkenaan Reka bentuk Migrasi Data dan Reka bentuk Integrasi Data.

Langkah-langkah

Langkah 1 : Sediakan Pengenalan Kepada Reka bentuk Sistem

Sila sedia dan lengkapkan pengenalan bisnes bagi perkara-perkara berikut:

a) Tujuan Reka bentuk

Terangkan tujuan, objektif dan matlamat yang ingin dicapai di dalam reka bentuk sistem aplikasi selaras dengan objektif bisnes dan keperluan sistem yang ingin dipenuhi.

b) Skop Reka bentuk

Nyatakan skop reka bentuk sistem aplikasi berdasarkan skop keperluan bisnes dan keperluan sistem yang telah diperolehi di dalam dokumen BRS dan SRS. Skop reka bentuk merupakan penentuan sempadan kepada reka bentuk fungsi sistem dan pangkalan data yang disediakan. Sempadan reka bentuk ini boleh ditentukan dengan merujuk kepada bilangan modul, menu dan submenu yang akan dirangkumkan di dalam sistem aplikasi.

Langkah 2 : Sediakan Reka bentuk Arkitektur Sistem Aplikasi

Reka bentuk Arkitektur

a) Arkitektur Keseluruhan Sistem Aplikasi

Sediakan Arkitektur Keseluruhan Sistem Aplikasi yang merupakan gambaran menyeluruh (*bird's eye view*) bagi komponen-komponen antaramuka sistem, aplikasi dan pangkalan data. Sila rujuk kepada langkah 4 di dalam Reka bentuk Arkitektur [F3.1] untuk menyediakan arkitektur berkenaan.

b) Arkitektur Aplikasi

Sediakan Arkitektur Aplikasi yang merupakan gambaran komponen-komponen aplikasi terperinci yang terkandung di dalam Arkitektur Keseluruhan Sistem Aplikasi. Sila rujuk kepada langkah 5 di dalam Reka bentuk Arkitektur [F3.1] untuk menyediakan arkitektur berkenaan.

c) Arkitektur Pangkalan Data

Sediakan Arkitektur Pangkalan Data yang merupakan gambaran komponen-komponen pangkalan data terperinci yang terkandung di dalam Arkitektur Keseluruhan Sistem Aplikasi. Sila rujuk kepada langkah 6 di dalam **Reka bentuk Arkitektur [F3.1]** untuk menyediakan arkitektur berkenaan.

Langkah 3 : Kemaskini Model Fungsi Sistem

a) Penggunaan Notasi

Senaraikan notasi-notasi yang akan digunakan untuk menyediakan Model Fungsi Sistem. Rujuk kepada Pemodelan Fungsi Sistem [F2.2] untuk menyediakan senarai notasi berkenaan.

b) Rajah Hierarki Fungsian Sistem

Rajah Hierarki Fungsian Sistem yang telah disediakan di dalam Pemodelan Fungsi Bisnes [F2.2] dan dipaparkan di dalam dokumen D03 Spesifikasi Keperluan Sistem perlu dikemaskini dan dibangunkan semula sekiranya terdapat perubahan dari segi struktur Fungsi Sistem dalam fasa reka bentuk ini. Namun begitu sekiranya tiada sebarang perubahan kepada struktur berkenaan, Rajah Hierarki Fungsian Sistem yang sama boleh diguna dan direkodkan di dalam dokumen SDS.

c) Jadual Pemandanan Aktor Dengan Fungsi Sistem

Kemaskini Jadual Pemandanan Aktor dengan Fungsi Sistem yang telah dibangunkan dalam **Pemodelan Fungsi Bisnes [F2.2]** sekiranya terdapat sebarang perubahan kepada struktur Fungsi Sistem. Sekiranya tiada sebarang perubahan, paparkan sahaja jadual yang sama yang telah dibangunkan dalam fasa sebelum ini. Jadual pemandanan

ini akan digunakan sebagai sumber rujukan bagi kawalan akses pengguna dengan sistem aplikasi yang akan dibangunkan.

Langkah 4 : Dokumentasikan Reka bentuk Fungsian

a) Reka bentuk Antaramuka Pengguna dan Pemetaan Data

Sertakan imej-imej Reka bentuk Antaramuka Pengguna bagi setiap skrin di bawah fungsi, modul, menu dan submenu aplikasi. Reka bentuk Antaramuka Pengguna yang telah disediakan kemudiannya akan dipadankan dengan entiti (jadual) dan atribut yang terkandung di dalam pangkalan data. Sila rujuk kepada langkah-langkah di bawah Reka bentuk Antaramuka [F3.4] bagi penyediaan reka bentuk tersebut dan pemetaan kepada data-data yang terlibat.

b) Reka bentuk Transaksi Sistem

Sertakan Reka bentuk Transaksi Sistem yang terdiri dari jadual-jadual Senario *Use Case*. Sila rujuk kepada langkah-langkah di dalam **Reka bentuk Transaksi Sistem [F3.5]** bagi penyediaan reka bentuk berkenaan.

Langkah 5 : Dokumentasikan Reka bentuk Pangkalan Data

a) Reka bentuk Pangkalan Data

Sertakan Reka bentuk Pangkalan Data Logikal yang merupakan perincian lanjut kepada Rajah Hubungan Entiti (ERD) yang telah disediakan di dalam dokumen SRS. Sila rujuk kepada langkah-langkah di dalam Reka bentuk Pangkalan Data [F3.3] bagi penyediaan reka bentuk berkenaan.

b) Skema Logikal Pangkalan Data

Sertakan juga Skema Pangkalan Data Logikal berdasarkan kepada langkah-langkah yang telah dinyatakan di dalam Reka bentuk Pangkalan Data [F3.3].

Langkah 6 : Nyatakan Secara Ringkas Reka bentuk Migrasi Dan Integrasi Data

a) Reka bentuk Migrasi Data

Sediakan keterangan ringkas berkenaan Reka bentuk Migrasi Data dan nyatakan juga rujukan kepada dokumen [D05] Pelan Migrasi Data dan [D06] Spesifikasi Migrasi Data bagi penerangan lanjut kepada reka bentuk berkenaan.

b) Reka bentuk Integrasi Data

Sediakan keterangan ringkas berkenaan Reka bentuk Integrasi Data dan nyatakan juga rujukan kepada dokumen [D06] Pelan Integrasi Data dan [D07] Spesifikasi Integrasi Data bagi penerangan lanjut kepada reka bentuk berkenaan.

Langkah 7 : Sertakan Dokumen-dokumen Sokongan Sebagai Lampiran

Sertakan dokumen-dokumen sokongan, sekiranya ada, yang perlu dirujuk sebagai penerangan lanjut kepada reka bentuk-reka bentuk yang disertakan di dalam dokumen SDS.

Langkah 8 : Lakukan Semakan Dan Pengesahan Ke Atas Dokumen SDS

Dokumen SDS perlu dilakukan semakan oleh Ketua Pasukan Analisis dan Reka bentuk, atau pegawai-pegawai yang lain yang bersesuaian. Setelah semakan dilakukan, dokumen SDS yang telah disediakan perlu disahkan oleh Pengurus Projek atau Pengarah Bahagian atau pegawai-pegawai yang lain yang bersesuaian.

Rujukan

1. ISO/IEC/IEEE 29148-2011 Systems and software engineering - Life cycle processes - Requirements engineering (2011).
2. IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications (1998).
3. IEEE 1233-1998 - IEEE Guide for Developing System Requirements Specifications (1998).
4. IEEE Std 1016-2009 : IEEE Standard for Information Technology - Systems Design - Software Design Descriptions (2009).

4.11 Migrasi Data

Migrasi data merupakan proses memindahkan data daripada satu sumber asal (sama ada daripada sumber sistem legasi, format excel, *hardcopies* atau sebagainya) ke destinasi baharu yang berbeza daripada sumber asal.

Proses migrasi ini biasanya dijalankan atas sebab-sebab seperti berikut:

- a) Pengembangan skop bisnes yang memerlukan pembangunan/peningkatan sistem;
- b) Penggabungan beberapa sistem kepada satu sistem;
- c) Penaiktarafan perkakasan dan perisian; dan
- d) Proses normalisasi/ semakan semula pangkalan data.

Objektif migrasi data adalah untuk memastikan data daripada sistem legasi dapat digunakan dalam sistem baharu selaras dengan keperluan bisnes.

2 aktiviti utama dalam migrasi data iaitu:

- a) Penyediaan Pelan Migrasi Data; dan
- b) Reka bentuk Migrasi Data.

4.11.1 Penyediaan Pelan Migrasi Data [F3.7]

Keterangan

Keperluan migrasi data dapat dikenal pasti semasa proses mengenal pasti proses bisnes sebagaimana diterangkan dalam **Pemodelan Proses Bisnes [F1.4]**. Analisis keperluan migrasi data perlu dijalankan bagi menentukan skop kerja proses migrasi data, impak migrasi kepada bisnes dan mengenal pasti risiko serta isu-isu berkaitan proses migrasi data.

Pelan Migrasi Data akan dibangunkan sebagai panduan dan rujukan bagi keseluruhan pelaksanaan migrasi data hasil daripada analisis keperluan data. Pelan ini menggariskan strategi, kaedah dan jadual pelaksanaan migrasi data yang perlu dipatuhi semasa pelaksanaan migrasi data. Penyediaan pelan ini dapat melancarkan pelaksanaan migrasi data.

Objektif

- Menjalankan analisis keperluan data bagi menghasilkan Pelan Migrasi Data sebagai sumber rujukan yang menetapkan strategi, kaedah dan jadual pelaksanaan yang akan digunakan dalam pelaksanaan migrasi data.

Langkah-Langkah

Langkah 1 : Analisis Keperluan

Analisis keperluan migrasi akan melibatkan kajian dan perbincangan antara pemilik proses, pembangun sistem dan pasukan migrasi data. Proses ini dijalankan bagi menentukan perkara-perkara seperti berikut:

Jadual 41 : Analisa Keperluan Migrasi Data

Bil.	Perkara	Penerangan
1	Apakah data yang ingin dipindahkan?	Kategori data yang ingin dipindahkan sebagai contoh, data maklumat peribadi pengguna bagi Sistem Tempahan Bilik Mesyuarat.
2	Apakah sistem yang terlibat?	Nyatakan sistem yang terlibat. Pemindahan data melibatkan data daripada sistem ke sistem atau manual ke sistem.
3	Bilakah data diperlukan?	Nyatakan bilakah data diperlukan dan maklumat ini perlu diambil kira dalam penyediaan jadual pelaksanaan.

4	Apakah impak migrasi kepada bisnes?	Impak sekiranya migrasi tidak dilaksanakan atau tidak dijalankan mengikut jadual kepada bisnes agensi. Adakah akan mengganggu aliran proses bisnes?
5	Apakah risiko proses migrasi ini?	Risiko dalam pelaksanaan migrasi sebagai contoh, sistem legasi perlu menjalani <i>downtime</i> untuk proses migrasi. Data perlu dipindahkan pada masa yang bersesuaian supaya tidak mengganggu aktiviti proses bisnes yang telah dirancang.
6	Apakah jenis pangkalan data yang digunakan di sistem legasi?	Maklumat persekitaran teknologi yang digunakan seperti jenis pangkalan data dan jenis data yang terdapat pada sistem legasi diperlukan sebagai input kepada strategi dan kaedah migrasi.
7	Apakah strategi dan kaedah yang bersesuaian digunakan untuk migrasi?	Strategi pelaksanaan migrasi data dilaksanakan sama ada secara <i>one-off</i> atau berfasa, maklumat penggunaan <i>tools</i> , secara <i>scripting</i> atau lain-lain kaedah dikenal pasti berdasarkan keperluan bisnes dan persekitaran semasa.
8	Siapa yang terlibat dalam proses migrasi data?	Pasukan migrasi yang akan terlibat dan peranan setiap ahli.

Langkah 2 : Bangunkan Pelan Migrasi Data

Pelan Migrasi Data dihasilkan selepas analisis keperluan migrasi data selesai dijalankan dan mengandungi **sekurang-kurangnya** perkara seperti berikut :

Jadual 42 : Isi Kandungan Pelan Migrasi Data

Tajuk	Isi Kandungan
Tujuan Dokumen	Penerangan tujuan dokumen dihasilkan adalah untuk dijadikan panduan bagi pelaksanaan migrasi data bagi sistem legasi kepada sistem baharu. Nama sistem perlu dinyatakan dengan jelas.
Latar Belakang	Pengenalan ringkas sistem iaitu objektif dan fungsi sistem serta impak pelaksanaan migrasi data kepada bisnes.

Objektif Migrasi	Penerangan objektif migrasi data dilaksanakan.
Skop Migrasi	<p>Penerangan sistem dan modul yang terlibat dan skop data yang terlibat.</p> <ol style="list-style-type: none"> Nama sistem terlibat – contoh : data daripada Sistem Sumber Manusia akan dipindahkan ke Sistem Tempahan Bilik Mesyuarat Modul yang terlibat – contoh : data bagi Modul Pengurusan Pengguna sahaja Julat data yang terlibat – contoh : data daripada tahun 2010 hingga 2017 sahaja atau data pegawai gred 41 dan ke atas sahaja.
Pendekatan Migrasi Data	<p>Penjelasan tentang pendekatan pelaksanaan migrasi data sama ada berfasa mengikut modul atau sebaliknya. Kaedah migrasi dijalankan iaitu sama ada menggunakan <i>tools</i>, <i>scripting</i> atau lain-lain kaedah juga diterangkan dalam bab ini.</p> <p>Bab ini juga mengandungi maklumat sekurang-kurangnya seperti berikut :</p> <ol style="list-style-type: none"> Penambahbaikan atau penyediaan <i>Service Level Agreement</i> (SLA) sekiranya perlu; Persediaan perkakasan (<i>hardware</i>) dan perisian (<i>software</i>) diperlukan; Aspek teknologi mengenai persekitaran semasa dan persekitaran migrasi yang perlu diambil kira; dan Kaedah pengujian dan verifikasi data yang akan digunakan
Pasukan Projek	Struktur organisasi bagi pasukan projek dan menyertakan nama-nama pegawai terlibat.
Jadual Pelaksanaan	<p>Penerangan jadual pelaksanaan migrasi data yang dicadangkan dalam bentuk gantt chart atau yang bersesuaian. Ia mengandungi aktiviti migrasi bermula daripada analisis keperluan sehingga penyediaan laporan. Jadual pelaksanaan juga perlu mengambil kira aspek seperti berikut :</p> <ol style="list-style-type: none"> Susunan keutamaan data yang ingin dipindahkan; dan Tempoh <i>downtime</i> yang dibenarkan untuk sistem semasa sekiranya melibatkan sistem yang sedang beroperasi.

Pelaksanaan migrasi data akan dilaksanakan berdasarkan pelan yang dihasilkan. Rujuk format **D05 Pelan Migrasi Data**.

Langkah 3 : Sahkan Pelan Migrasi Data

Pelan Migrasi Data yang didokumenkan perlu dibentang dan mendapat pengesahan pemilik sistem bagi memastikan strategi, kaedah dan jadual pelaksanaan migrasi data memenuhi keperluan dan mendapat sokongan serta kerjasama daripada pemilik sistem dan pembangun migrasi data.

Rujukan

1. Pelan Migrasi Data Sistem eRoses.
2. Oracle White Paper (2011). Successful Data Migration. <http://www.oracle.com/technetwork/middleware/oedq/successful-data-migration-wp-1555708.pdf>
3. Credosoft White Paper. Eight key steps which help ensure a successful data migration project: A white paper for inspection management professionals. <http://credosoft.com/wp/wp-content/uploads/2014/01/Eight-key-steps-which-help-ensure-a-successfu-data-migration-project.pdf>
4. SAGA Group (2012). Methods of Data Migration.

4.11.2 Reka bentuk Migrasi Data [F3.8]

Keterangan

Proses reka bentuk dan pemetaan akan dilakukan selepas selesai pembangunan pelan migrasi. Proses ini terbahagi kepada empat langkah utama iaitu :

- a) Pemetaan jadual (*table*);
- b) Pemetaan medan data (*field*);
- c) Pemetaan kod; dan
- d) Pemetaan rekod (*data*) disebabkan perubahan kod/id.

Proses ini boleh dilakukan secara manual atau menggunakan *tools*. Antara *tools* yang boleh digunakan dalam reka bentuk migrasi ialah Altova MapForce, Talend dan Navicat.

Objektif

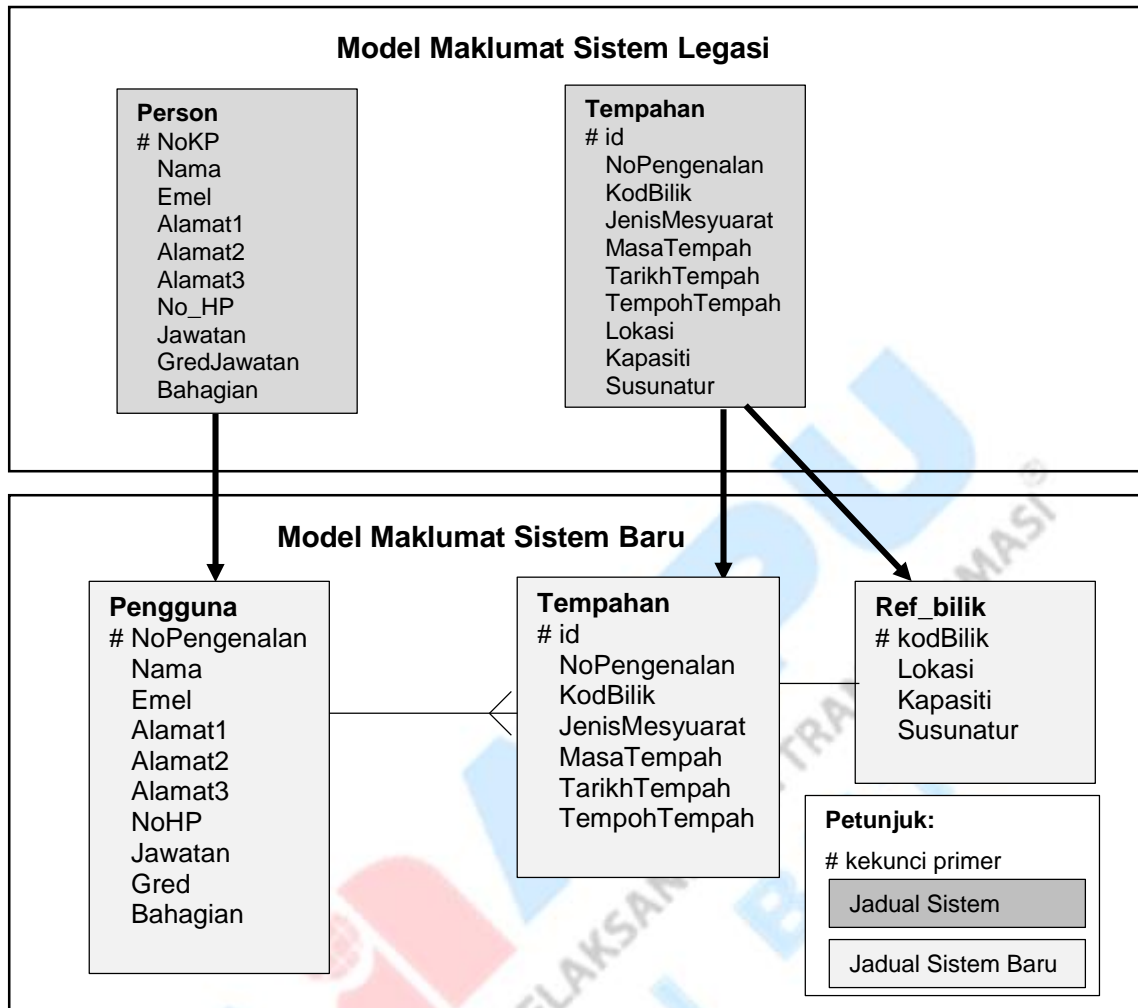
- o Proses reka bentuk dan pemetaan akan memastikan data yang dipindahkan dipetakan dari sumber asal ke destinasi baharu.

Langkah-Langkah

Langkah 1 : Pemetaan Jadual

Langkah pertama ini akan mengenal pasti bagaimana data daripada pangkalan data sistem legasi dipetakan kepada pangkalan data sistem baharu. Proses ini melibatkan pengenalpastian jadual yang terlibat pada kedua-dua pangkalan data sistem legasi dan sistem baharu. Sumber rujukan bagi proses ini adalah Model Maklumat Logikal bagi pangkalan data sistem yang terlibat.

Contoh pengenalpastian dan pemetaan jadual yang dijalankan dalam proses migrasi adalah seperti berikut:



Rajah 73 : Pemetaan Jadual antara Pangkalan Data Sistem Legasi dan Sistem Baharu

Langkah 2 : Pemetaan Medan Data

- a) Kenal pasti elemen data berdasarkan kepada Skema Logikal Pangkalan Data bagi sistem legasi serta sistem baharu.
- b) Bangunkan peraturan bisnes dalam mengendalikan kesemua proses ini. Sebagai contoh:
 - i) Alamat lokasi aset disimpan dalam 3 baris berlainan pada pangkalan data sumber iaitu Address1, Address2 dan Address3. Tentukan sama ada alamat lokasi aset di destinasi baharu akan mengikut format sumber atau disimpan dalam 1 baris sahaja pada *field* "Alamat".
 - ii) Tinggi dan lebar aset direkodkan dalam format sentimeter pada pangkalan data sistem baharu. Oleh yang demikian, data dalam format inci pada pangkalan data sistem legasi perlu ditukar kepada sentimeter.

iii) Kod aset pada pangkalan data sistem baharu akan diselaraskan menggunakan kod standard pada *Data Dictionary* Sektor Awam (DDSA).

c) Lengkapkan Apendiks 8a Templat Pemetaan Data Migrasi.

Contoh pengisian pemetaan data :

Jadual 43 : Pemetaan Medan Data antara Sistem Legasi dan Sistem Baharu

Bil.	MAKLUMAT PANGKALAN DATA SUMBER						MAKLUMAT PANGKALAN DATA DESTINASI						Catatan
	Penyedia (SISTEM SUMBER MANUSIA)						Penerima (SISTEM TEMPAHAN BILIK MESYUARAT)						
	Jadual	Medan	Jenis	P/F	Mandatori/Tidak	DDSA	Jadual	Medan	Keterangan	Jenis	P/F	Mandatori/Tidak	
1	PERSON	EPD_NAMA	Varchar (100)		Mandatori		PENGGUNA	nama	Nama pegawai	Varchar (100)		Mandatori	Tidak
2	PERSON	EPD_NOKP	Varchar (16)	P	Mandatori	Ya	PENGGUNA	no_pengenalan	No. Kad Pengenalan	Varchar (16)	P	Mandatori	Tidak
3	PERSON	EMEL	Varchar (50)		Tidak		PENGGUNA	emel	Alamat emel pegawai	Varchar (50)		Tidak	Tidak

Nota:

- i) P – Kekunci Primer
- ii) F – Kekunci Asing
- iii) DDSA – *Data Dictionary* Sektor Awam

Langkah 3 : Sediakan Keperluan Pemetaan Kod

Proses pemetaan kod dijalankan sekiranya ada keperluan untuk menyesuaikan dan menyelaraskan kod (atau ID) bagi atribut daripada sumber data dengan destinasi baharu data. Proses ini biasanya dijalankan ke atas jadual rujukan.

Jadual 44 : Contoh Pemetaan Kod

Kod Data Asal (sumber asal)	Penerangan data	Kod Data Baharu (destinasi baharu)
MJKP	Mesyuarat Jawatankuasa Pemandu	M01
MJKT	Mesyuarat Jawatankuasa Teknikal	M02
MBHGN	Mesyuarat Bahagian	M03

Aktiviti yang terlibat dalam proses keperluan pemetaan kod adalah seperti berikut.

- a) Kenal pasti kod yang berubah.

Kod yang berubah adalah seperti berikut:

MJKP -> M01
 MJKT -> M02
 MBHGN -> M03

- b) Kenal pasti jadual dan medan yang menyimpan maklumat kod tersebut. Contoh jadual yang dikenalpasti adalah jadual sumber asal: ref_Mesyuarat dan jadual destinasi baru: ruj_JenisMesyuarat (jadual baharu) seperti jadual dibawah.

Jadual 45 : Contoh jadual yang menyimpan maklumat kod

Jadual dan Medan Asal (Sumber Asal)	Jadual dan Medan Baru (Destinasi Baru)
Jadual: ref_mesyuarat	Jadual: ruj_jenisMesyuarat
Medan: kod, nama	Medan: kod, keterangan

Langkah 4 : Sediakan Keperluan Pemetaan Rekod (Data) Disebabkan Perubahan Kod/ID

- a) Adakalanya perubahan kod yang berlaku menyebabkan perubahan ke atas data dalam rekod maklumat. Perubahan yang berlaku ke atas data memerlukan pemetaan data berdasarkan perubahan kod (atau ID). Proses pemetaan data dijalankan bagi menyesuaikan dan menyelaraskan rekod data berdasarkan kod data baharu. Contoh bagi keperluan pemetaan rekod (data) disebabkan perubahan kod/ID adalah seperti berikut.

Dalam sistem lama, rekod skim dan perjawatan adalah dalam jadual sumber asal iaitu Jadual : ref_skimJwtn seperti dibawah.

Jadual 46: jadual sumber asal ref_skimJwtn

Kod	Keterangan
ICT1	Skim jawatan ICT untuk gred 1
ICT2	Skim jawatan ICT untuk gred 2
ICT3	Skim jawatan ICT untuk gred 3

Manakala dalam sistem baru pula, terdapat perubahan dalam kod skim jawatan ICT. Rekod skim dan gred perjawatan yang baru (dalam sistem baru) adalah dalam jadual destinasi baru iaitu Jadual : ref_skimGredJwtn seperti dibawah.

Jadual 47: Jadual destinasi baru ref_skimGredJwtn

Kod	Keterangan
X41	Skim jawatan ICT untuk gred 41
X44	Skim jawatan ICT untuk gred 44
X48	Skim jawatan ICT untuk gred 48
X52	Skim jawatan ICT untuk gred 52
X54	Skim jawatan ICT untuk gred 54
XX60	Skim jawatan ICT untuk gred 60
XX61	Skim jawatan ICT untuk gred 61
XX62	Skim jawatan ICT untuk gred 62

Kedua-dua jadual di atas adalah contoh jadual yang menyimpan rekod skim dan perjawatan bagi sumber asal dan jadual destinasi baru yang telah dikenalpasti.

Berikut adalah sebahagian daripada medan yang terdapat dalam jadual sumber asal Jadual : pekerja. Jadual ini menunjukkan rekod pekerja yang mengandungi maklumat rekod peribadi iaitu termasuklah skim dan gred jawatan serta gaji. Medan 'gred' menggunakan gred skim perjawatan yang lama. Jadual dibawah adalah contoh jadual sumber asal iaitu Jadual : pekerja yang telah dikenalpasti.

Jadual 48: Jadual sumber asal 'pekerja'

idPekerja	nama	tarikh_mula_bekerja	gaji	gred
0000001	Ali bin Abu	01 April 1985	12070.00	ICT3
0000002	Aminah binti Aziz	15 Januari 1988	10800.00	ICT3
0000003	Latifah binti Mazlan	01 Mac 2005	6500.00	ICT2
0000004	Suriya binti Abdullah	01 Mei 2010	3500.00	ICT1
0000005	Hassan bin Abu	01 Jun 2017	2100.00	ICT1

Jadual berikut adalah maklumat yang mengandungi perubahan gred perjawatan bagi skim ICT yang terlibat berdasarkan gaji dan tempoh perkhidmatan.

Jadual 49: Contoh Maklumat Perubahan Gred Perjawatan

Skim/Gred Lama (Sumber Asal)	Tempoh Perkhidmatan	Gaji (RM)	Skim/Gred Baru (Sumber Destinasi)
ICT1	Kurang 5 tahun	1,500.00 hingga 3,000.00	X41
ICT1	5 hingga 8 tahun	3,000.01 hingga 5,000.00	X44
ICT2	8 hingga 12 tahun	5,000.01 hingga 7,000.00	X48
ICT2	12 hingga 15 tahun	7,000.01 hingga 9,000.00	X52
ICT2	15 hingga 20 tahun	9,000.01 hingga 12,000.00	X54
ICT3	20 hingga 25 tahun	Lebih 12,000.00	XX60
ICT3	25 hingga 30 tahun	Lebih 12,000.00	XX61
ICT3	30 hingga 35 tahun	Lebih 12,000.00	XX62

b) Berdasarkan contoh di atas, berikut adalah langkah-langkah yang perlu diambil bagi melaksanakan pemetaan rekod (data) disebabkan perubahan kod/ID:

- i) Kenal pasti data atau rekod yang perlu dikemaskini disebabkan perubahan kod tersebut. Data atau rekod yang perlu dikemaskini disebabkan perubahan kod adalah data gred bagi setiap pekerja. Dalam pangkalan data sumber asal, rekod pekerja disimpan dalam jadual 'pekerja' dan medan yang akan mengalami perubahan kod adalah medan 'gred'.

Kenalpasti jadual dan medan yang terlibat dalam proses pemetaan data. Jadual yang terlibat dalam proses pemetaan data adalah jadual sumber asal iaitu Jadual: ref_skimJwtn dan Jadual: pekerja seperti yang telah dinyatakan di atas.

- ii) Sediakan peraturan pemetaan data bagi rekod yang terlibat seperti dibawah. Contoh pemetaan bagi kod baru (medan 'gred') untuk rekod kakitangan di jadual destinasi baru iaitu Jadual : PERSONEL. Peraturan yang digunakan adalah seperti yang terdapat dalam jadual perubahan gred perjawatan bagi skim ICT berdasarkan gaji dan tempoh perkhidmatan. Berdasarkan rajah diatas, peraturan yang terlibat dalam jadual pekerja adalah dalam medan 'tarikh_mula_bekerja' dan medan 'gaji'.

Contoh pengiraan bagi formula pemetaan data (andaian *current* tarikh adalah 15 Oktober 2018) bagi idPekerja '0000001' adalah seperti berikut.

$$\begin{aligned}
 \text{Tempoh perkhidmatan} &= \text{current tarikh} - \text{tarikh_mula_bekerja} \\
 &= 15 \text{ Oktober } 2018 - 01 \text{ April } 1985 \\
 &= 33 \text{ tahun}
 \end{aligned}$$

Contoh Jadual bagi idPekerja '0000001' adalah seperti berikut.

Jadual 50: Contoh Jadual Yang Menunjukkan Rekod idPekerja '0000001'

idPekerja	Tempoh perkhidmatan (current tarikh – tarikh_mula_bekerja)	Gaji terkini	Gred lama
0000001	15 Oktober 2018 – 01 April 1985 = 33 tahun	12070.00	ICT3

Contoh pemetaan data bagi idPekerja '0000001' berdasarkan jadual Pekerja dan maklumat perubahan gred perjawatan adalah seperti rajah berikut.

idPekerja	Tempoh perkhidmatan (current tarikh – tarikh_mula_bekerja)	Gaji terkini	Gred lama	Gred baru
0000001	15 Oktober 2018 – 01 April 1985 = 33 tahun	12070.00	ICT3	XX62

Skim/Gred lama (sumber asal)	Tempoh Perkhidmatan	Gaji (RM)	Skim/Gred Baru (sumber destinasi)
ICT1	Kurang 5 tahun	1,500.00 hingga 3,000.00	X41
ICT1	5 hingga 8 tahun	3,000.01 hingga 5,000.00	X44
ICT2	8 hingga 12 tahun	5,000.01 hingga 7,000.00	X48
ICT2	12 hingga 15 tahun	7,000.01 hingga 9,000.00	X52
ICT2	15 hingga 20 tahun	9,000.01 hingga 12,000.00	X54
ICT3	20 hingga 25 tahun	Lebih 12,000.00	XX60
ICT3	25 hingga 30 tahun	Lebih 12,000.00	XX61
ICT3	30 hingga 35 tahun	Lebih 12,000.00	XX62

↑ Tempoh perkhidmatan **33 tahun** ↑ Gaji terkini **12070.00** ↑ Gred baru **XX62**

Rajah 74 : Contoh pemetaan data

Pemetaan bagi rekod yang menunjukkan maklumat gred lama dan gred baru adalah seperti jadual di bawah.

Jadual 51 : Contoh Pemetaan Rekod Yang Menunjukkan Maklumat Perubahan Gred

idPekerja	Tempoh perkhidmatan (current tarikh – tarikh_mula_bekerja)	Gaji terkini	Gred lama	Gred baru
0000001	15 Oktober 2018 – 01 April 1985 = 33 tahun	12070.00	ICT3	XX62
0000002	15 Oktober 2018 – 15 Januari 1990 = 28 tahun	10800.00	ICT3	X54
0000003	15 Oktober 2018 – 01 Mac 2005 = 13 tahun	6500.00	ICT2	X48
0000004	15 Oktober 2018 – 01 Mei 2009 = 9 tahun	3500.00	ICT1	X44
0000005	15 Oktober 2018 – 01 Jun 2017 = 1 tahun	2100.00	ICT1	X41

Dalam pangkalan data destinasi, rekod pekerja disimpan dalam jadual pesonel dan medan yang perlu dikemaskini ialah 'gred'. Jadual dibawah (jadual: pesonel) adalah contoh rekod pesonel bagi jadual pesonel di destinasi baru. Medan 'gred' menggunakan skim gred perjawatan yang baru. Formula perubahan adalah

berdasarkan jadual perubahan gred perjawatan bagi skim ICT berdasarkan gaji dan tempoh perkhidmatan seperti di atas. Contoh data yang terlibat dalam perubahan dalam kod skim jawatan ICT adalah di jadual destinasi baru Jadual : Personel seperti jadual di bawah.

Jadual 52 : Contoh Jadual Destinasi Baru Pesonel

idPesonel	namaPesonel	tarikhMulaKhidmat	gaji	gred
0000001	Ali bin Abu	01 April 1985	12070.00	XX62
0000002	Aminah binti Aziz	15 Januari 1990	10800.00	X54
0000003	Latifah binti Mazlan	01 Mac 2005	6500.00	X48
0000004	Suriya binti Abdullah	01 Mei 2009	3500.00	X44
0000005	Hassan bin Abu	01 Jun 2017	2100.00	X41

- c) Lengkapkan Apendiks 8b) Templat Peraturan Pemetaan Data bagi rekod yang terlibat. Contoh pengisian peraturan pemetaan data seperti jadual dibawah.

Jadual 53 : Contoh Pengisian Peraturan Pemetaan Data

Kod Data Asal (Sumber Asal)	Keterangan Data	Peraturan /Nota	Kod Data Baharu (Destinasi Baru)
ICT1	Kod Skim	tempohPerkhidmatan = currentdate – PERSONEL.tarikhMulaKhidmat; gred = X41 If PERSONEL.kodGredSkim = ICT1 && PERSONEL.gaji is between 1500.00 and 3000.00 && tempohPerkhidmatan < 5 years;	X41
ICT1	Kod Skim	tempohPerkhidmatan = currentdate – PERSONEL.tarikhMulaKhidmat; gred = X44 If PERSONEL.kodGredSkim = ICT1 && PERSONEL.gaji is between 3500.01 and 5000.00;	X44

Langkah 5 : Sediakan Spesifikasi Migrasi Data

- a) Spesifikasi Migrasi Data perlu dibangunkan dan dijadikan rujukan dalam pelaksanaan migrasi data bersama-sama Pelan Migrasi Data. Spesifikasi ini akan mendokumentasikan langkah 1 hingga 3 yang telah diterangkan di atas iaitu mengandungi perkara seperti berikut:

Jadual 54 : Isi Kandungan Spesifikasi Migrasi Data

Tajuk	Isi Kandungan
Tujuan Dokumen	Penerangan tujuan dokumen dihasilkan adalah untuk merekodkan maklumat bagi reka bentuk migrasi data bagi tujuan memindahkan data daripada pangkalan data sumber ke destinasi pangkalan data baharu.
Maklumat Sistem	Maklumat sistem dan pangkalan data yang terlibat bagi tujuan migrasi data daripada sumber asal ke destinasi baharu direkodkan. Ini bertujuan bagi memudahkan penyelarasan dan konfigurasi sistem untuk tujuan migrasi data dijalankan.
Pemetaan Jadual	Penerangan dan rajah pemetaan jadual yang terlibat pada kedua-dua pangkalan data sistem legasi dan sistem baharu.
Peraturan Bisnes	Senarai peraturan bisnes yang perlu dipatuhi dalam proses migrasi data yang telah ditetapkan untuk menyelaraskan format data yang dipindahkan.
Pemetaan Data	Penerangan dan jadual yang mengandungi perincian data yang perlu dipindahkan perlu dikenal pasti berdasarkan kepada Skema Logikal Pangkalan Data bagi sistem legasi serta sistem baharu.
Pemetaan Kod	Penerangan dan jadual yang mengandungi perincian kod bagi atribut yang perlu diselaraskan antara pangkalan data sistem legasi dan sistem baharu.

- b) Rujuk format dokumen D06 Spesifikasi Migrasi Data. Dokumen ini juga boleh dijadikan sebagai lampiran dalam D05 Pelan Migrasi Data.
- c) Langkah seterusnya iaitu proses Pelaksanaan Migrasi Data [F6.1] yang mana terdiri daripada aktiviti Pelaksanaan, Pengujian dan Penyediaan Laporan Migrasi data.

Langkah 6 : Sahkan Spesifikasi Migrasi Data

D06 Spesifikasi Migrasi Data yang didokumenkan perlu dibentang dan mendapat pengesahan pemilik sistem bagi memastikan kesahihan dan spesifikasi yang dihasilkan memenuhi keperluan migrasi data.

Rujukan

1. Pelan Migrasi Data Sistem eRoses.
2. Oracle White Paper (2011). Successful Data Migration. <http://www.oracle.com/technetwork/middleware/oedq/successful-data-migration-wp-1555708.pdf>
3. Credosoft White Paper. Eight key steps which help ensure a successful data migration project: A white paper for inspection management professionals. <http://credosoft.com/wp/wp-content/uploads/2014/01/Eight-key-steps-which-help-ensure-a-successfu-data-migration-project.pdf>
4. SAGA Group (2012). Methods of Data Migration.

4.12 Integrasi Sistem

Terdapat pelbagai sistem aplikasi dibangunkan untuk menyokong fungsi sesebuah organisasi. Kewujudan pelbagai sistem aplikasi yang beroperasi secara silo menyebabkan pertindihan fungsi dan duplikasi data. Integrasi sistem dilaksanakan untuk membolehkan sistem-sistem aplikasi yang berasingan dapat melaksanakan tugas secara bersepadu dan *seamless*. Ini dapat meningkatkan produktiviti pekerja dan memudahkan organisasi mencapai matlamatnya. Selain itu, integrasi sistem juga dapat meningkatkan ketepatan dan kebolehpercayaan data.

2 aktiviti utama Integrasi Sistem dalam Fasa Reka bentuk iaitu:

- a) Penyediaan Pelan Integrasi Sistem; dan
- b) Reka bentuk Integrasi Sistem.



4.12.1 Penyediaan Pelan Integrasi Sistem [F3.9]

Keterangan

Pelan integrasi dibangunkan sebagai panduan dan rujukan bagi keseluruhan pelaksanaan integrasi sistem. Pelan integrasi menggariskan kaedah, strategi dan jadual pelaksanaan integrasi yang perlu dipatuhi.

Objektif

- Menghasilkan Pelan Integrasi berfungsi sebagai rujukan yang menetapkan kaedah dan strategi yang akan digunakan serta jangkamasa yang diperlukan semasa pelaksanaan integrasi.

Langkah-Langkah

Langkah 1 : Analisis Keperluan

Analisis Keperluan Integrasi perlu dilaksana untuk tujuan penyediaan Pelan Integrasi. Kajian dan perbincangan perlu melibatkan antara pemilik proses, pemilik sistem dan pasukan pembangun integrasi. Berikut adalah perkara-perkara yang perlu diambil kira untuk tujuan penyediaan Pelan Integrasi:

Jadual 55 : Analisa Keperluan Integrasi Sistem

Perkara	Keterangan
Apakah tujuan pengintegrasian? Apakah fungsi bisnes yang perlu disokong?	Fungsi bisnes yang ingin disokong melalui pengintegrasian sistem.
Apakah skop integrasi?	Sistem yang terlibat dan komponen sistem yang terlibat. Nyatakan juga integrasi yang melibatkan sistem luaran milik agensi luar.
Apakah strategi pelaksanaan integrasi?	Strategi yang akan dilaksanakan sama ada bersekali dengan pembangunan sistem utama ataupun selepas sistem utama siap. Strategi pelaksanaan integrasi perlu diselaraskan dengan D01 Pelan Pembangunan Sistem merangkumi fasa pembangunan, pengujian dan pelaksanaan.
Apakah keperluan perisian dan perkakasan untuk pembangunan integrasi?	Perisian dan perkakasan yang diperlukan untuk pembangunan integrasi.

Siapa yang terlibat dalam proses integrasi sistem?	Sumber manusia dan pihak berkepentingan yang akan terlibat dan peranan mereka dalam pembangunan, pengujian dan pelaksanaan integrasi
Apakah aktiviti yang terlibat?	Jadual masa bagi setiap aktiviti yang terlibat.

Langkah 2 : Sediakan Pelan Integrasi

Pelan Integrasi dihasilkan selepas analisa keperluan selesai dijalankan dan mengandungi sekurang-kurangnya perkara seperti berikut:

Jadual 56 : Isi Kandungan Pelan Integrasi Sistem

Kandungan	Keterangan
Objektif	Terangkan objektif pengintegrasian sistem ini
Skop kerja integrasi	Terangkan skop kerja integrasi yang terlibat
Pendekatan dan strategi	Pendekatan dan strategi yang digunakan untuk melaksanakan integrasi - Terangkan kaedah dan strategi yang dilakukan untuk mengenalpasti integrasi yang diperlukan dan pelaksanaan integrasi tersebut.
Kaedah integrasi, <i>tools</i> dan persekitaran	Terangkan kaedah integrasi dan <i>tools</i> yang terlibat dalam integrasi tersebut. Nyatakan juga persekitaran yang digunakan untuk pengujian integrasi tersebut
Tugas dan tanggungjawab	Terangkan tugas dan tanggungjawab pasukan integrasi dan pihak berkepentingan. Sertakan juga carta organisasi (jika berkaitan).
Jadual Pelaksanaan	Nyatakan tempoh masa yang diperlukan untuk setiap aktiviti yang dirancang. Aktiviti melibatkan fasa kajian, reka bentuk, pembangunan, pengujian dan pelaksanaan integrasi.
Andaian dan Risiko	Terangkan andaian yang dibuat sepanjang pelaksanaan integrasi dan potensi halangan yang akan memberi impak kepada pelaksanaan integrasi tersebut.

Pelaksanaan integrasi akan dilaksanakan berdasarkan pelan yang dihasilkan. Rujuk format **D07 Pelan Integrasi Data**.

Langkah 3 : Sahkan Pelan Integrasi

Pelan Integrasi yang didokumenkan perlu dibenteng dan mendapat pengesahan kesemua pemilik sistem yang terlibat dengan integrasi bagi memastikan aktiviti integrasi mendapat sokongan dan kerjasama.

4.12.2 Reka bentuk Integrasi Sistem [F3.10]

Keterangan

Integrasi sistem dilaksanakan bagi membolehkan sistem-sistem aplikasi yang berasingan bertukar maklumat secara automatik dan *seamless*. Ini kerana terdapat pelaksanaan proses bisnes yang memerlukan fungsi-fungsi daripada sistem aplikasi berlainan berinteraksi antara satu sama lain.

Service Orientation Architecture (SOA) merupakan kaedah integrasi yang menggunakan servis untuk berinteraksi antara sistem aplikasi. Servis merupakan logik fungsian yang dibangunkan mewakili proses bisnes. Ia boleh dikemaskini tanpa mengganggu servis lain dan ini menggalakkan pengintegrasian dilaksanakan secara *loosely coupled*. Selain itu, servis yang dibangunkan tidak terikat dengan platform, bahasa pengaturcaraan, sistem pengoperasian dan persekitaran sistem aplikasi. Ini kerana ia berkomunikasi menggunakan format data yang sama.

Objektif

- Menghasilkan spesifikasi integrasi bagi pelaksanaan keperluan proses bisnes yang merentasi fungsi bisnes agensi atau unit bisnes dalam agensi.

Langkah-Langkah

Langkah 1 : Kenal Pasti Keperluan Integrasi

Kenal pasti kebergantungan sistem yang sedang dibangunkan dengan sistem-sistem lain. Maklumat tersebut boleh diperolehi daripada hasil kajian keperluan sistem yang terdapat D03 Spesifikasi Keperluan Sistem. Pemodelan *Use Case* dan Rajah Konteks DFD turut digunakan dalam dokumen tersebut untuk menggambarkan skop interaksi sistem dengan pengguna atau sistem luaran secara menyeluruh.

Contoh Kajian Kes:

Rajah Konteks Sistem Pengurusan Tempahan Bilik Mesyuarat, **DFD-BM** menggambarkan secara keseluruhan hubungan sistem dengan entiti luaran. Berdasarkan Rajah Konteks tersebut, Sistem Pengurusan Tempahan Bilik Mesyuarat berintegrasi dengan Sistem Sumber Manusia dan Sistem Senggara Aset. Maka langkah seterusnya akan tertumpu kepada proses yang berlaku dan maklumat yang dikongsi antara sistem tersebut.

Langkah 2 : Kenalpasti Servis Integrasi Yang Diperlukan

- a) Senaraikan keperluan integrasi yang diperlukan. Nyatakan secara ringkas keperluan integrasi tersebut. Maklumat terperinci berkaitan servis tersebut boleh diperolehi daripada pemodelan *Use Case* dan *Data Flow Diagram* (DFD) dalam **D03 Spesifikasi**

Keperluan Sistem serta perincian Rajah Aliran Proses Bisnes (PFD) dalam **D02 Spesifikasi Keperluan Bisnes**.

- b) Maklumat yang perlu ada bagi keperluan servis integrasi adalah seperti di Apendiks 9 a) Maklumat Servis Integrasi. Jadual dibawah menunjukkan keterangan templat maklumat servis integrasi.

Jadual 57 : Keterangan Templat Maklumat Servis Integrasi

Nama Label	Keterangan
Rujukan Fungsi	Label dan nama Rajah Aliran Proses Bisnes yang dirujuk
Rujukan Aktiviti	Label dan nama <i>Use Case</i> yang dirujuk
Nama sistem sumber	Sistem yang membekalkan maklumat
Pemilik maklumat	Bahagian yang bertanggungjawab atas maklumat tersebut
Keterangan maklumat yang dihantar	Penerangan ringkas mengenai maklumat yang dihantar ke sistem lain
Tujuan penggunaan maklumat	Penerangan ringkas mengenai tujuan maklumat tersebut dihantar ke sistem lain

Contoh Kajian Kes:

Penguraian **DFD-BM-MA** menunjukkan bahawa integrasi berlaku semasa menguruskan aduan kerosakan. Perincian aktiviti berkaitan integrasi tersebut boleh diperolehi daripada Definisi Fungsi Bisnes pada Pemodelan Proses Bisnes (PFD), **PFD-BM-MA** dan Rajah *Use Case* **UC-BM-MA**. Senaraikan maklumat yang diperlukan seperti jadual berikut:

Jadual 58 : Contoh Maklumat Aktiviti yang Memerlukan Integrasi

Bil	Rujukan Fungsi	Rujukan Aktiviti	Nama Sistem Sumber	Pemilik Maklumat	Keterangan Maklumat yang Dihantar	Tujuan Penggunaan Maklumat
1	PFD-BM-MA-01 Mengurus Aduan Kerosakan	UC-BM-MA-01-001 Sediakan Aduan Kerosakan Baru	Sistem Pengurusan Tempahan Bilik Mesyuarat	Unit Sumber Manusia	Aduan baru bagi kerosakan bilik mesyuarat	Pengemaskinian status aset (bilik mesyuarat)
2	PFD-BM-MA-01 Mengurus Aduan Kerosakan	UC-BM-MA-01-002 Kemaskini Status Kesiediaan Bilik Mesyuarat	Sistem Senggara Aset	Unit Pentadbiran	Aduan kerosakan yang telah selesai	Pengemaskinian status kesiediaan bilik mesyuarat

Berdasarkan jadual di atas, terdapat dua servis integrasi yang perlu dibangunkan iaitu:

- i) Sistem Pengurusan Tempahan Bilik Mesyuarat akan menghantar maklumat aduan bagi kerosakan bilik mesyuarat.
- ii) Sistem Senggara Aset akan menghantar maklumat aduan kerosakan yang telah selesai.

Langkah 3 : Muktamadkan Format Pertukaran Data (*Data Exchange Format*)

- a) Berdasarkan senarai integrasi yang telah dikenalpasti, berikan nama servis untuk setiap integrasi tersebut dan pendekatan kaedah integrasi yang digunakan. Terangkan format pertukaran data yang akan digunakan. Rujuk Apendiks 9 b) Format Pertukaran Data. Keterangan bagi templat Format Pertukaran Data adalah seperti jadual di bawah.

Jadual 59 : Keterangan Templat Format Pertukaran Data

Nama Label	Keterangan
Nama Servis	Nama servis integrasi yang ingin dibangunkan.
Keterangan	Penerangan mengenai servis integrasi yang dibangunkan.
Kaedah Integrasi	Kaedah integrasi yang digunakan dan format pertukaran data yang digunakan.
URL <i>Web Service</i>	Alamat URL servis integrasi tersebut. Nyatakan URL yang digunakan semasa pembangunan, pengujian dan pelaksanaan (jika berkaitan).
<i>Request</i>	Kaedah permohonan maklumat.
<i>Respond</i>	Kaedah maklumbalas.
Data yang terlibat	Penerangan struktur data yang terlibat dalam integrasi.

Contoh Kajian Kes:

Berdasarkan contoh kajian kes, jadual dibawah menunjukkan keterangan servis integrasi dan data yang terlibat dalam integrasi tersebut.

Jadual 60 : Contoh Penerangan Servis Integrasi dan Data yang Terlibat

Nama Servis	Aduan Kerosakan				
Keterangan	Menghantar maklumat aduan kerosakan baru daripada Sistem Pengurusan Tempahan Bilik Mesyuarat				
Kaedah Integrasi	RESTful service Format: JSON				
Url Web Service	Testing: testing.company.com.my/sptbm/rest/integration Production: www.sptbm.my/rest/integration				
Request	Sistem luar perlu hantar permintaan kepada link berikut: <url web service>/aduan Arahan: GET				
Respond	<pre>{ "result" : { "aduan" : [{ "no_aduan" : , "tajuk_aduan" : , "keterangan_aduan" : , "nama_bilik_mesy" : , "status_bilik_mesy" : , "tarikh_daftar_aduan" : , "jenis_kerosakan" : , "status_penyelenggaraan" : , "tarikh_hantar_data" : }] } }</pre>				
Data yang terlibat	Nama	Jenis	Saiz	Nullable	Rules
	no_aduan	varchar	15	N	
	tajuk_aduan	varchar	100	N	
	keterangan_aduan	varchar	255	Y	
	nama_bilik_mesy	varchar	100	N	
	status_bilik_mesy	int	2	N	
	tarikh_daftar_aduan	date		N	dd/MM/yyyy
	jenis_kerosakan	varchar	100	N	
	status_penyelenggaraan	int	2	N	
tarikh_hantar_data	datetime		N	dd/MM/yyyy HH:mm:ss	

Jadual di atas menerangkan servis integrasi yang akan dibangunkan untuk membolehkan Sistem Pengurusan Tempahan Bilik Mesyuarat menghantar maklumat aduan kerosakan bilik mesyuarat ke sistem luar (Sistem Senggara Aset).

Langkah 4 : Sediakan Pemetaan Data (*Data Mapping*)

- a) Kenalpasti struktur data yang diperlukan oleh sistem dan struktur data yang dihantar oleh sistem sumber. Petakan struktur data tersebut. Apendiks 9 c) Pemetaan Struktur Data. Keterangan bagi templat pemetaan struktur data adalah seperti di jadual dibawah.

Jadual 61 : Keterangan Templat Pemetaan Struktur Data

Nama Label	Keterangan
Nama Servis	Nama servis integrasi.
Nama Sistem Penerima	Sistem yang memohon dan menerima data tersebut melalui servis integrasi.
Nama Sistem Pemilik	Sistem yang menyedia dan menghantar data tersebut kepada pemohon melalui servis integrasi.

Contoh Kajian Kes:

Berdasarkan contoh kajian kes, jadual dibawah menunjukkan contoh pemetaan struktur data yang terlibat.

Jadual 62 : Contoh Pemetaan Struktur Data

Nama Servis							
Aduan Kerosakan							
Nama Sistem Penerima				Nama Sistem Pemilik			
Sistem Selenggara Aset				Sistem Pengurusan Tempahan Bilik Mesyuarat			
Nama Medan	Jenis	Saiz	Keterangan	Nama Medan	Jenis	Saiz	Keterangan
no_siri	varchar	12		no_aduan	string	12	
tajuk	varchar	255		tajuk_aduan	string	100	
keterangan	text			keterangan_aduan	string	255	
nama_bilik_mesyuarat	varchar	255		nama_bilik_mesy	string	100	
status_bilik_mesyuarat	int	2		status_bilik_mesy	number	2	
tarikh_daftar	date		dd/MM/yyyy	tarikh_daftar_aduan	string		dd/MM/yyyy
jenis_kerosakan	varchar	255		jenis_kerosakan	string	100	

status_ penyelenggaraan	int	2		status_ penyelenggaraan	number	2	
tarikh_terima_data	datetime		dd/MM/yyyy HH:mm:ss	tarikh_hantar_data	string		dd/MM/yyyy HH:mm:ss

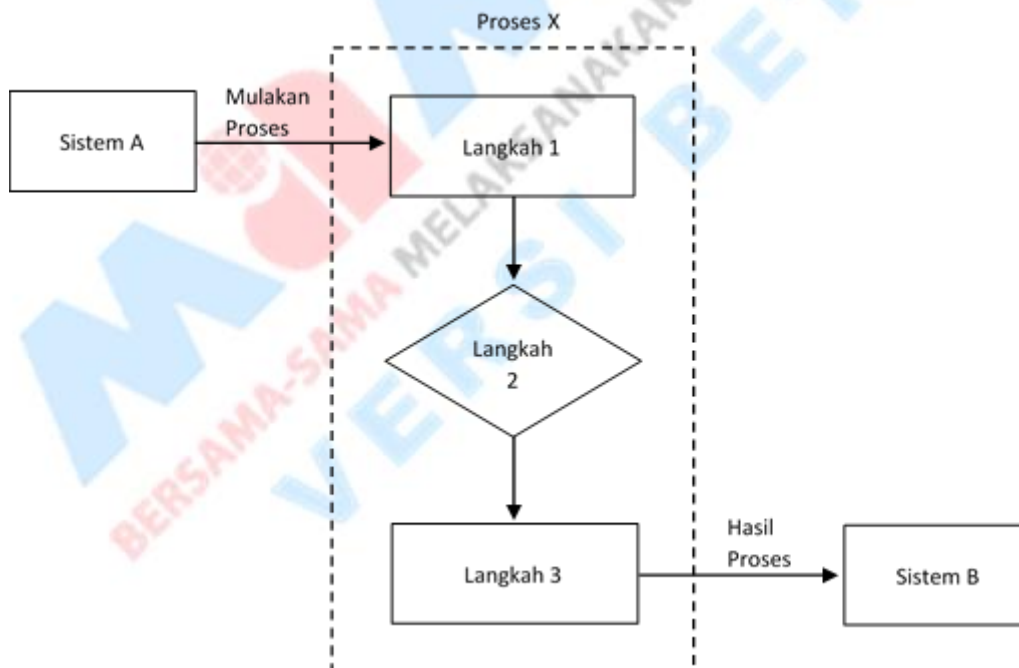
Jadual di atas menunjukkan pemetaan data aduan kerosakan bilik mesyuarat antara Sistem Senggara Aset dengan Sistem Pengurusan Tempahan Bilik Mesyuarat. Ini untuk memastikan data yang diterima daripada Sistem Pengurusan Tempahan Bilik Mesyuarat memenuhi keperluan Sistem Senggara Aset.

Langkah 5 : Sediakan Peraturan Integrasi Data

Kenalpasti aliran proses yang terlibat semasa pertukaran data berlaku. Nyatakan peraturan yang dilaksanakan semasa pertukaran data tersebut seperti logik integrasi dan transformasi data. Terdapat 2 jenis proses integrasi iaitu:

a) Proses Khusus (*Specialized Processes*)

Proses Khusus merupakan pengintegrasian untuk melaksanakan proses yang khusus untuk sistem tertentu. Ini bermakna hanya ada satu output sahaja yang dihasilkan oleh proses tersebut untuk diproses oleh sistem tertentu.

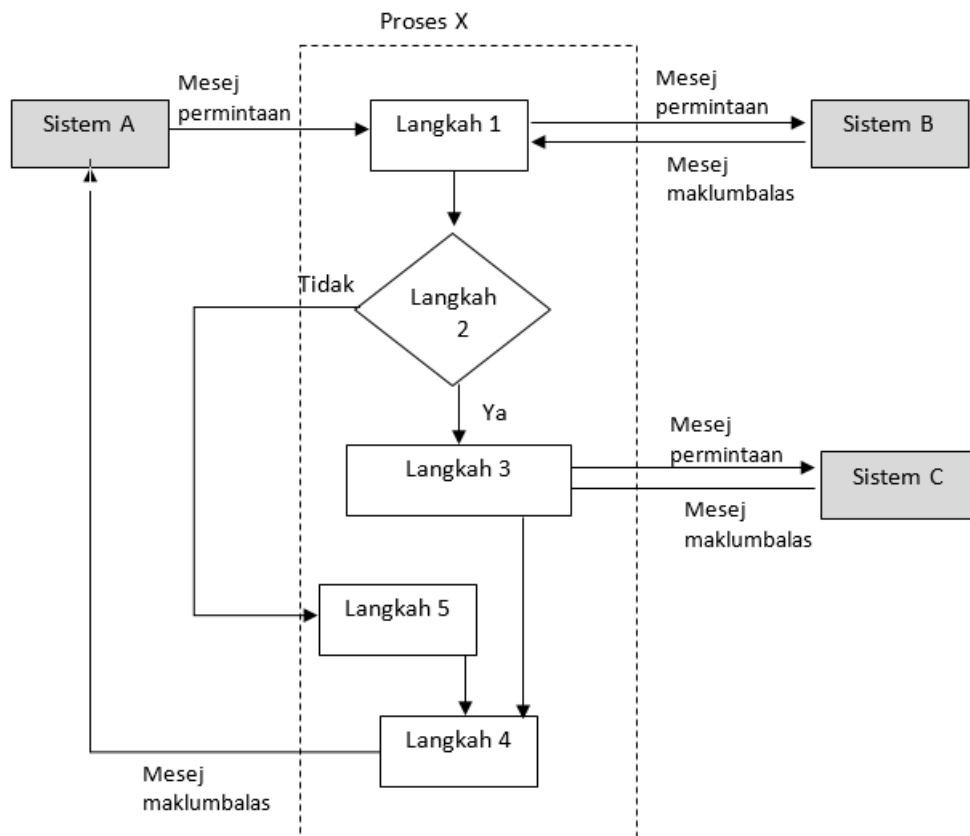


Rajah 75 : Proses Khusus

Berdasarkan rajah di atas, Sistem A akan memulakan Proses X yang terdapat pada server integrasi. Output yang dihasilkan oleh Proses X akan dihantar kepada Sistem B untuk diproses. Proses X ini diwujudkan untuk menyokong Sistem B agar sistem lain boleh melaksanakan prosedur tersebut tanpa perlu mengetahui senarai proses yang perlu dilakukan.

b) Proses Berbilang Langkah (*Multistep Process*)

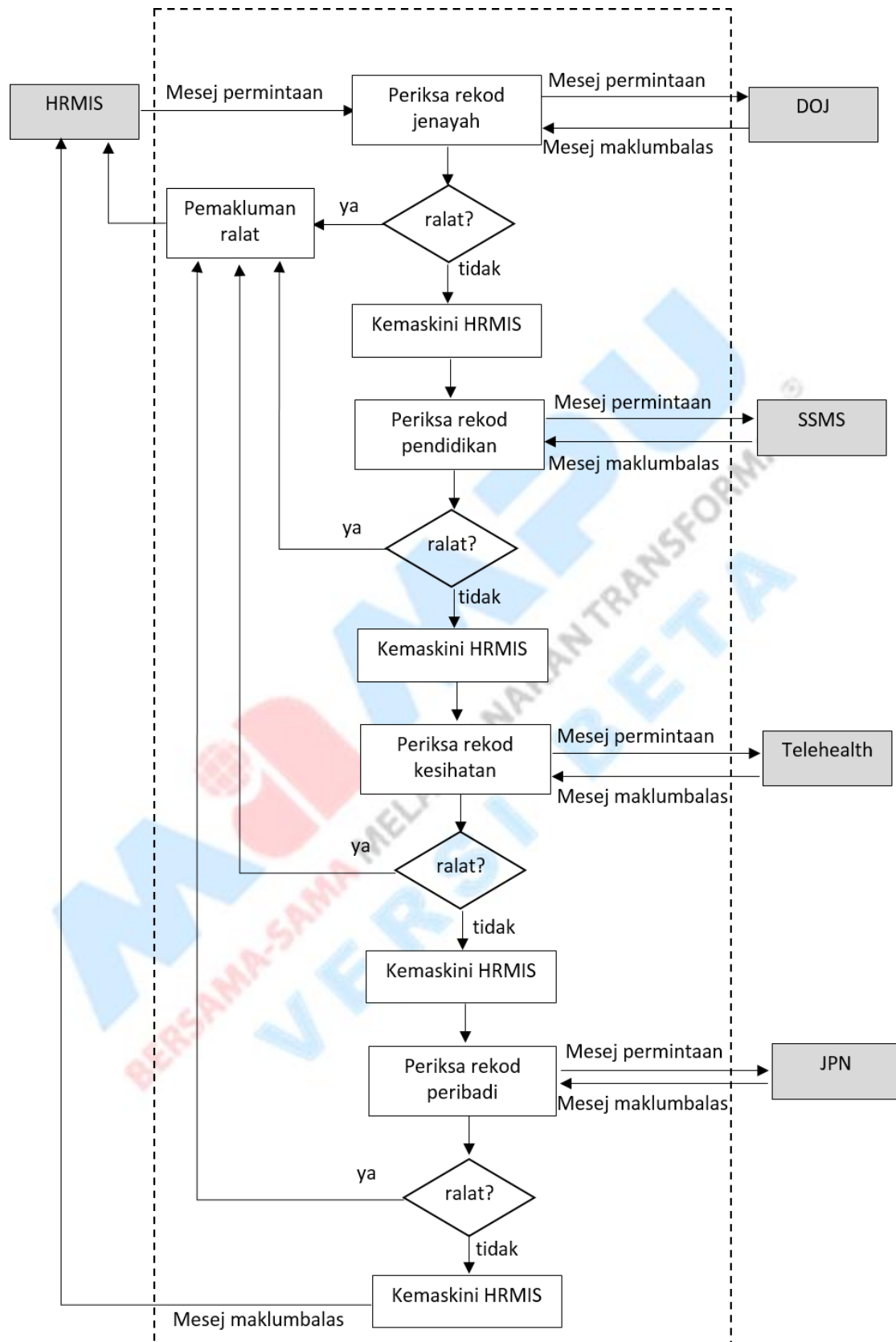
Proses Berbilang Langkah melibatkan pengintegrasian lebih dari satu sistem untuk melaksanakan proses tertentu. Aliran proses yang diwujudkan dalam server integrasi akan mengendalikan interaksi antara sistem tersebut. Semua transaksi antara proses dan sistem tersebut perlu mematuhi kekangan transaksi *atomic*. Ini bermakna sekiranya salah satu transaksi gagal, maka proses tersebut akan gagal. Maka semua perubahan yang telah berlaku kepada data perlu kembali kepada asal seolah-olah proses tersebut tidak berlaku. Mekanisma pengurusan pengecualian (*exception handling mechanism*) perlu diwujudkan untuk melaksanakan pembalikan (*rollback*) tersebut.



Rajah 76 : Proses Berbilang Langkah

Rajah di atas menunjukkan Sistem A perlu berintegrasi dengan Sistem B dan Sistem C untuk melaksanakan Proses X. Mesej permintaan digunakan untuk meminta menggunakan servis atau data yang dikongsi oleh sistem lain manakala mesej maklumbalas ialah hasil proses servis atau data yang diminta oleh mesej permintaan tersebut.

Contoh Senario Proses Integrasi:

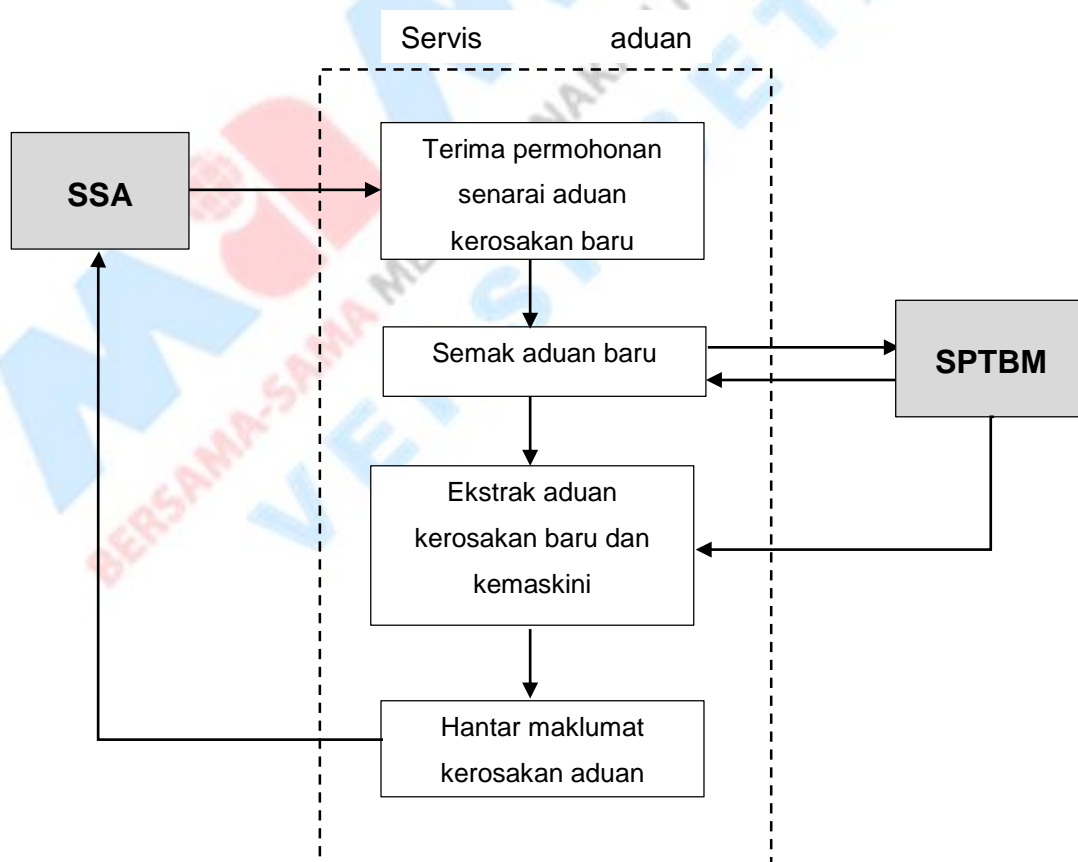


Rajah 77 : Contoh Senario Proses Integrasi

Berdasarkan proses di atas, Sistem Maklumat Rekod Pekerja (HRMIS) akan menghantar maklumat pekerja kepada proses integrasi untuk memeriksa rekod jenayah daripada Sistem Maklumat Kehakiman (DOJ), rekod pendidikan daripada Sistem Maklumat Pendidikan (SSMS), rekod kesihatan daripada Sistem Maklumat Kesihatan (Telehealth) dan rekod peribadi daripada Sistem Maklumat Pendaftaran (JPN). Jika semua rekod memenuhi kriteria, proses pengemaskinian akan dilakukan kepada sistem Telehealth dan HRMIS. Jika berlaku kegagalan semasa transaksi, mekanisma pengurusan pengecualian (exception handling mechanism) akan memainkan peranan untuk melaksanakan pembalikan (rollback) proses tersebut. Segala perubahan yang telah berlaku kepada sistem yang terlibat akan dibatalkan dan proses tersebut perlu diulang semula.

Contoh Kajian Kes:

Sistem Senggara Aset (SSA) akan memohon maklumat aduan kerosakan daripada Sistem Pengurusan Tempahan Bilik Mesyuarat (SPTBM). Sistem Pengurusan Tempahan Bilik Mesyuarat akan mendapatkan maklumat aduan kerosakan baru dan menghantar maklumat tersebut bersama tarikh_hantar_data yang telah dikemaskini. Sistem Senggara Aset akan menerima maklumat tersebut dan menyimpannya untuk tindakan seterusnya.



Rajah 78 : Contoh Proses Integrasi Servis Aduan Kerosakan

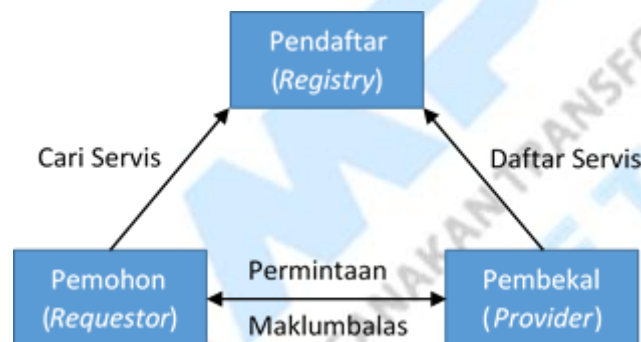
Langkah 6 : Sediakan Reka Bentuk Arkitektur Integrasi

Reka bentuk arkitektur integrasi adalah bergantung kepada cara dan peraturan integrasi tersebut dilakukan. Penerangan di bawah menunjukkan kaedah integrasi menggunakan SOAP, *RESTful* dan *messaging*.

a) Arkitektur SOAP

SOAP bermaksud *Simple Object Access Protocol*. Terdapat tiga peranan utama dalam arkitektur *web service*:

- i) Pembekal (*Provider*) - mewujudkan servis dan menjadikannya tersedia untuk sistem aplikasi yang ingin menggunakannya.
- ii) Pemohon (*Requestor*) - menggunakan servis sedia ada dengan menghantar permintaan kepada pembekal servis.
- iii) Pendaftar (*Registry*) – menyimpan maklumat servis yang disediakan oleh pembekal servis.



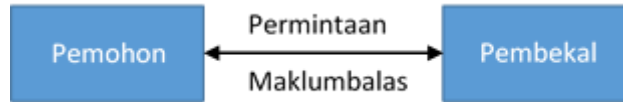
Rajah 79 : Arkitektur SOAP

Pembekal servis akan memaklumkan kepada pendaftar tentang servis yang disediakan dan cara menggunakannya. Pemohon akan mencari servis yang dikehendaki dalam pendaftar. Setelah mendapat maklumat mengenai servis yang diperlukan, pemohon akan menghantar permintaan kepada pembekal servis tersebut. Pembekal akan memberi maklumbalas berdasarkan jenis permintaan tersebut.

b) Arkitektur *RESTful*

REST bermaksud *REpresentational State Transfer*. Ia mempunyai ciri-ciri berikut:

- i) Setiap sumber seperti maklumat boleh diakses melalui URL.
- ii) Kata arahan GET, POST, PUT atau DELETE akan digunakan semasa mengakses sumber tersebut.



Rajah 80 : Arkitektur *RESTful*

Arkitektur *RESTful* berkonsepkan *client-server*. Permintaan akan dilakukan oleh klien/pemohon dengan cara menghantar URL sumber yang diperlukan dan kata arahan. Pelayan/pembekal servis akan memberi maklumbalas berdasarkan URL dan kata arahan permintaan tersebut.

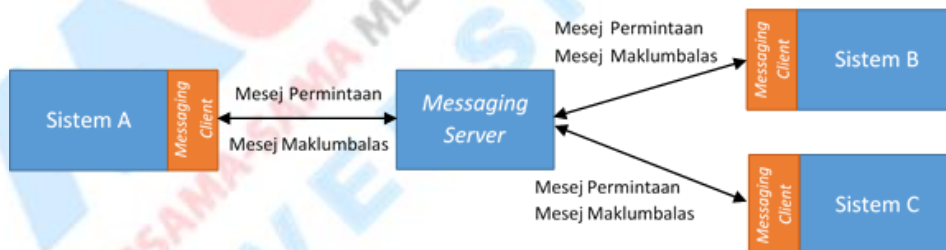
c) Arkitektur *Messaging*

Penghantaran Mesej (*Messaging*) membolehkan setiap sistem aplikasi berkomunikasi menggunakan mesej melalui saluran mesej (*message channel*) yang sama. Mesej yang dihantar oleh sistem aplikasi akan diuruskan oleh sistem penghantaran mesej (*messaging system*). Perkongsian data dan arahan adalah menggunakan mesej.



Rajah 81 : Arkitektur *Messaging*

Setiap sistem aplikasi akan dipasang dengan *messaging client* untuk membolehkannya menghantar dan menerima mesej. *Messaging server* bertindak sebagai orang tengah yang menguruskan penghantaran mesej.



Rajah 82 : Arkitektur *Messaging* untuk Pengintegrasian Berbilang Server

Arkitektur *messaging* memudahkan pengintegrasian yang melibatkan proses berbilang langkah (*multistep step*) kerana ia menyokong pengintegrasian berbilang sistem. Proses integrasi tersebut akan dibangunkan dalam *messaging server*.

Langkah 7 : Dokumentasikan Spesifikasi Integrasi Data

Kompilkan hasil langkah-langkah yang telah dilaksanakan ke dalam dokumen **D08 Spesifikasi Integrasi Data**.

Langkah 8 : Dapatkan Pengesahan Pengguna

Spesifikasi Integrasi Data yang didokumentasikan perlu dibentangkan dan mendapat pengesahan pemilik sistem bagi memastikan kesahihan dan spesifikasi yang dihasilkan memenuhi keperluan integrasi data.

Rujukan

1. Katalog Servis Pasukan Perunding ICT Sektor Awam (2013).
2. Dokumen SRDS Projek Perkongsian Maklumat Perkhidmatan Perguruan – Edu-XChange (2011).
3. Gregor Hohpe & Bobby Woolf (2003). Enterprise Integration Patterns. Addison-Wesley. ISBN: 0321200683.
4. MSC CFI Architecture – System Integration Architecture Version 3.0 (2004).
5. Dokumen uCustoms Penang Port Sdn Bhd (PPSB) Integration Specification Issue 1.0 (2015).