Seminar Nasional
Aplikasi Teknologi Informasi
2004
Yogyakarta, 19 Juni 2004

Perbandingan Model *Two-Tier* dengan *Three-Tier* dalam Arsitektur *Client/Server* untuk Mengolah Perintah *Query* pada Aplikasi Database

Wahyu Pujiyono, M. Idham Ananta Timur, Savitri
Jurusan Teknik Informatika, Universitas Ahmad Dahlan Yogyakarta
Jl. Prof. Supomo, Yogyakarta, Telp. (0274) 379418, Faks. (0274) 381523
e-mail: yywahyup@yahoo.com

Abstract

In the client/server architecture, the increasing numbers of users tend to add more complexity in database applications. As a result, performance will be the main reason for managing database server with many users. Altough two-tier model can work well by separating application from data sources, people always looking for another alternative to improve its performance and reability.

Three-tier model is the next development of two-tier model which add the third component within client application and application server, called "the middle tier." Middle tier is placed in separate machine which has several functions. Do business processing, and manage data are some of its useful functions. Separating business processing machine from database server will improve its performance by using its resources efficiently.

This paper will compare the performance of two-tier model with three-tier model in client/server architecture to manage query in database application developed by Delphi™. The connection process in three-tier model is focused on DCOM implementation. As conclusion, three-tier model's performance is more reliable and more efficient than two-tier model in client/server architecture.

Keywords: *client/server database, two-tier, three-tier, query, DCOM.*

1. Latar Belakang

Seiring dengan pesatnya kemajuan teknologi, perangkat lunak database atau biasa disebut dengan aplikasi database juga semakin berkembang, baik dalam hal penggunaan, ukuran, maupun kompleksitas. Hal ini secara langsung akan berdampak pada *database server*, sehingga konsekuensi dari semua itu adalah beban *database server* yang akan semakin bertambah. Dampak utama dari semua itu adalah kinerja *database server* yang akan menjadi lambat seiring banyaknya *client* yang ditangani. Untuk mengatasi hal itu, maka diperlukan antisipasi dalam perancangan arsitektur sistem database yang lebih baik di masa mendatang.

Pemusatan kerja pada komputer *server* mengakibatkan *server* harus melakukan dua pekerjaan sekaligus. Pertama komputer *server* bertindak sebagai *database server*, yang jelas hal ini membutuhkan alokasi sumber daya (*resource*) yang tidak sedikit. Kedua, komputer *server* juga dituntut untuk melayani permintaan dari pihak *client*, dalam hal ini bisa berupa transaksi, *query*, dan manajemen data.

Meski dalam hal ini terdapat dua aplikasi yang terpisah, akan tetapi arsitektur tersebut dirasakan belum mampu bekerja secara maksimal. Selain itu, arsitektur *two-tier* memiliki tuntutan tinggi dalam hal koneksi database, di mana koneksi harus selalu dijaga untuk masing-masing *client* sehingga menghabiskan sumber daya *server*. Terutama ketika aplikasi yang dibuat sudah menangani tajamnya peningkatan permintaan dari aplikasi *client* yang semakin hari semakin bertambah banyak, sehingga bisa berakibat kurang optimalnya kinerja *database server*.

2. Kajian Pustaka

2.1 Arsitektur Client/Server

Arsitektur *client/server* merupakan sebuah langkah maju karena menggunakan beban pemrosesan dari komputer sentral ke komputer *client*. Ini berarti semakin banyak *user* bertambah pada aplikasi *client/server*, maka kinerja *server* tidak akan menurun dengan cepat. Sistem *client/server* merupakan sistem yang cukup baik untuk digunakan, sistem ini mampu menghasilkan aplikasi database yang tangguh, serta mampu mengurangi kepadatan lalu-lintas jaringan.

Di dalam arsitektur *client/server*, pemrosesan pada sebuah aplikasi terjadi pada *client* dan *server*. *Client/server* merupakan tipikal sebuah aplikasi *two-tier* dengan banyak *client* dan sebuah *server* tunggal, di mana aplikasi ditempatkan pada komputer *client* dan mesin database dijalankan pada *server* jarak jauh. Proses yang terjadi adalah, aplikasi *client* mengeluarkan permintaan (*request*), kemudian server mengirimkan kembali data (*respons*) kepada client.

2.2 Model Client/Server

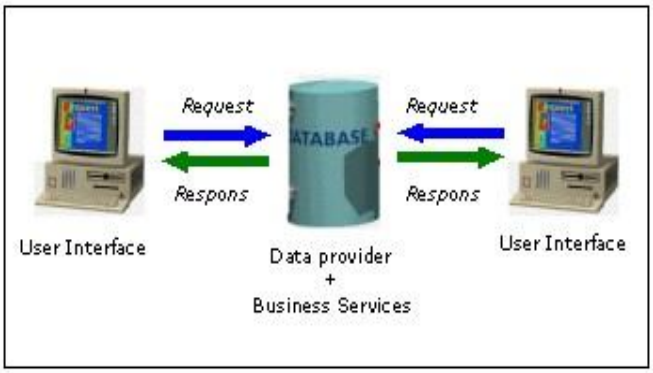
Secara umum, model atau bentuk dari *client/server* dapat dibagi menjadi beberapa bagian, namun dari garis besarnya terdiri dari dua bagian, yaitu model *two-tier* dan model *three-tier*.

a. Model Two-Tier

Model *two-tier* merupakan lingkungan *client/server* secara tradisional [10]. Pada model ini suatu aplikasi dibagi menjadi dua entitas, yaitu aplikasi *client* dan aplikasi *server*. Dalam konfigurasi yang tipikal, pembagian ini juga meliputi pembagian perangkat lunak dan perangkat keras. Aplikasi *client* umumnya diletakkan pada workstation yang digunakan oleh *user*, sedangkan *server* merupakan suatu komputer yang diletakkan di bagian lain pada jaringan. Model arsitektur dari *two-tier* terdiri dari dua bagian [9], yaitu:

- Layanan Presentasi (*Client Tier*)
  Layanan presentasi atau logika antarmuka pengguna ditempatkan pada mesin *client*. Lapisan ini berfungsi untuk menangani interaksi *user* dengan aplikasi.

- Layanan Data (*Data Source Tier*)
  Layanan data merupakan sebuah database *server* atau DBMS (*Database Management Systems*) yang menyediakan data bagi lapisan layanan *client* atau presentasi.



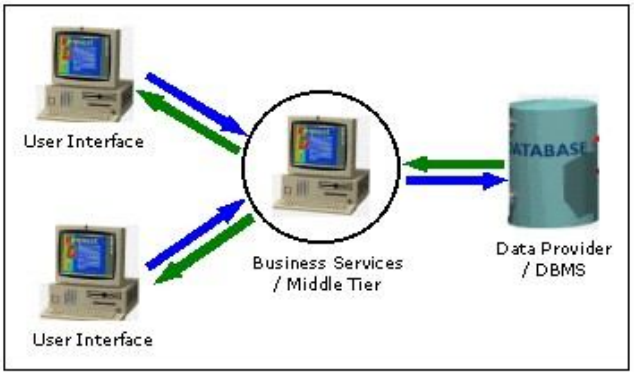
Gambar 1. Model two-tier client/server

b. Model Three-Tier

Model *three-tier* atau disebut juga *multi-tier* merupakan langkah pengembangan dari *two-tier client/server*. Hal ini berarti, setiap aplikasi *three-tier* adalah *client/server*, namun tidak semua aplikasi *client/server* adalah *three-tier*.

Model *three-tier* menambahkan komponen ketiga diantara aplikasi *client* dengan aplikasi *server* yang disebut *middle tier* atau layanan bisnis. Oleh karena itu, dalam model ini pemrosesan disebarkan di antara tiga lapisan (atau lebih).

- Layanan Presentasi (*Client Tier*)
  Sebagaimana dalam *two-tier*, layanan ini berfungsi untuk menangani semua interaksi *user* dengan aplikasi. Namun demikian, layanan ini tidak langsung mengakses database *server*.
- Layanan Bisnis (*Business Tier*)
  Layanan bisnis atau biasa disebut dengan *middle tier* merupakan sebuah aplikasi yang memberlakukan aturan-aturan bisnis, memproses data, dan mengelola transaksi. Logika yang semula ditempatkan pada *client* dipindahkan ke dalam komponen lapisan bisnis ini.
- Layanan Data (*Data Source Tier*)
  Layanan data merupakan sebuah DBMS yang mewakili satu atau lebih penyimpanan data. Lapisan ini menyediakan permintaan data bagi aplikasi *client* dengan melalui lapisan layanan bisnis.



Gambar 2. Model arsitektur pada three-tier client/server

3. Pembahasan

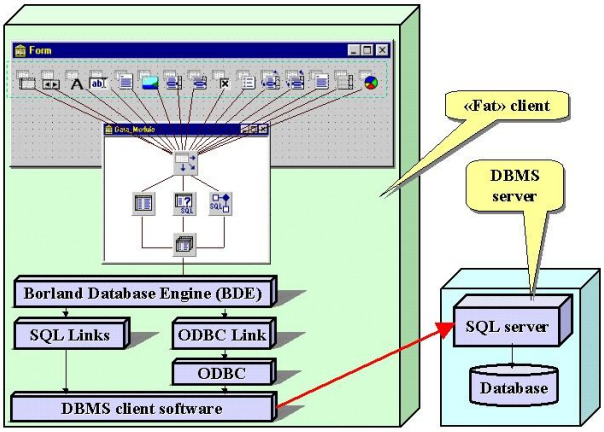
3.1 Analisis dan Kebutuhan Sistem

Dalam penelitian ini ruang lingkup aplikasi yang akan dibangun dilakukan menggunakan pemrograman Delphi dan diterapkan pada jaringan lokal (LAN). Aplikasi pada model *two-tier* serta *three-tier* berfungsi untuk melakukan perintah *query* dalam bentuk pernyataan SQL. Untuk operasi yang dilakukan meliputi penambahan data, pengambilan data, serta pengolahan data.

3.2 Implementasi Aplikasi Two-Tier

Pembuatan program aplikasi terdiri dari dua tahap, yaitu tahap desain dan tahap pembuatan prosedur program. Prosedur program digunakan agar program aplikasi dapat berjalan dan bekerja sebagaimana yang diharapkan. Tahap pembuatan program aplikasi dilakukan pada komputer yang bertindak sebagai aplikasi *client* dalam model *two-tier*, sedangkan aplikasi pada server hanya berupa database server, dengan demikian tidak perlu membuat *interface*.

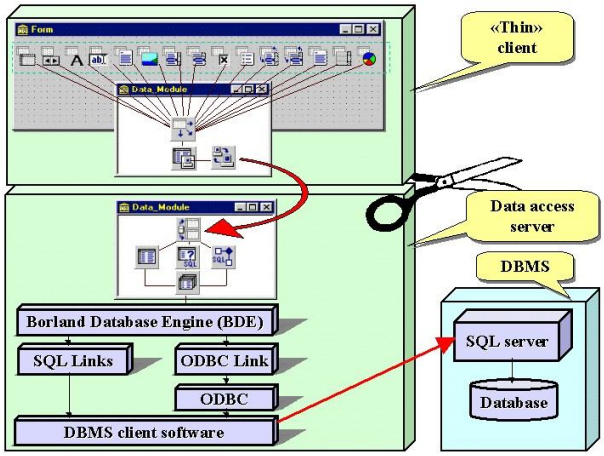
Untuk aplikasi *two-tier* yang diimplementasikan dengan Delphi, pembuatan program aplikasi dilakukan dalam sebuah *project*. Setelah pembuatan *project* selesai, selanjutnya aplikasi-aplikasi pada project dihubungkan dengan SQL server melalui BDE Delphi.



Gambar 3. Aplikasi two-tier menggunakan Delphi

3.3 Implementasi Aplikasi Three-Tier

Pembuatan aplikasi *three-tier* pada Delphi memerlukan tahapan yang lebih banyak, dengan kata lain diperlukan lebih dari satu *project*. Project pertama merupakan aplikasi *middle tier* dan *project* kedua adalah aplikasi *client.* Agar saling berkomunikasi antara project pertama dengan *project* kedua dihubungkan dengan *remote data module*. Meskipun masing-masing *project* diletakkan pada komputer yang terpisah, namun akan tetap saling mengenali.

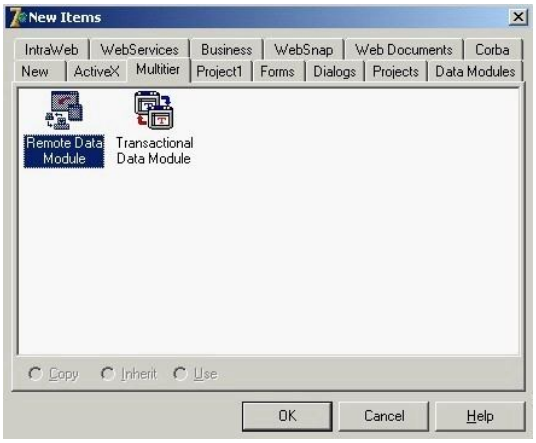


Gambar 4. Aplikasi three-tier menggunakan Delphi

Tahap pembuatan aplikasi *three-tier* terbagi ke dalam tiga bagian yaitu pembuatan aplikasi *middle tier*, registrasi *server*, dan pembuatan aplikasi *client*. Masing-masing bagian memiliki beberapa langkah dan tahap pelaksanaan.

a. Aplikasi Middle Tier

Layanan bisnis atau biasa disebut dengan *middle tier* merupakan sebuah aplikasi yang memberlakukan aturan-aturan bisnis, memproses data, dan mengelola transaksi. Secara khusus Delphi telah menyediakan fasilitas untuk membuat aplikasi *three-tier* yang terpaket dalam tab Multitier.



Gambar 5. Fasilitas pembuatan three-tier pada Delphi

b. Registrasi Server

Proses registrasi dilakukan dengan menjalankan aplikasi *middle tier* yang kemudian akan menghasilkan aplikasi program dalam bentuk file eksekusi (exe). Dari registrasi ini akan didapatkan nama *server* yang diambil dari nama file pada project serta nama *class* yang telah dibuat. Hasil registrasi inilah yang akan digunakan oleh aplikasi *client* untuk dapat berkomunikasi dengan *middle tier*, selain itu juga menggunakan alamat IP komputer *server*.

c. Aplikasi Client

Pembuatan aplikasi *client* pada model *three-tier* layaknya ketika membuat aplikasi *two-tier* yang menggunakan *client* dataset untuk melakukan *cache update*. Aplikasi *client* inilah yang akan dijalankan pada komputer *client* untuk mengakses data pada *server*. Hal yang membedakan antara client ini dengan *client two-tier* adalah penggunaan teknologi DCOM dari tab DataSnap.

3.4 Hasil Pengujian

Pengujian dilakukan dengan cara menambahkan data, mengambil data, serta mengolah data pada *three-tier* serta *two-tier*. Pengujian pertama dilakukan dengan memasukkan data acak sebanyak 10000 *record* pada masing-masing model aplikasi. Proses pengujian yang dilakukan adalah sebanyak lima kali dan hasilnya yang berupa waktu eksekusi dalam satuan menit dicatat berdasarkan waktu yang diperlukan.

Tabel 1. Hasil pengujian pemasukan data

No. Pengujian	**Model Two-Tier (*menit*)**	**Model Three-Tier (*menit*)**
1	13,09	9,00
2	12,20	10,10
3	12,50	10,40
4	12,20	10,40
5	12,40	10,50
Rata-rata	12,48	10,08

Pengujian untuk mengambil data dilakukan melalui perintah SQL (SELECT) untuk mengambil data dari tabel. Pengambilan data hanya dilakukan dalam satu tabel dan tidak melibatkan tabel lainnya.

Tabel 2. Hasil pengujian pengambilan data

No Pengujian	**Model Two-Tier (*menit*)**	**Model Three-Tier (*menit*)**
1	2,20	1,50
2	2,00	1,45
3	2,20	1,55
4	2,20	1,40
5	2,10	2,00
Rata-rata	2,14	1,58

Proses pengujian penghitungan data dilakukan untuk mendapatkan hasil dari pengolahan data yang dilakukan dengan mengalikan kolom dari tabel satu dengan kolom pada tabel lain.

Tabel 3. Hasil pengujian penghitungan data

No Pengujian	Model Two-Tier (*menit*)	Model Three-Tier (*menit*)
1	3,01	2,17
2	2,50	2,20
3	2,40	2,20
4	2,50	2,18
5	2,40	2,10
Rata-rata	2,56	2,17

Dari hasil pengujian yang telah dilakukan, diperoleh data dalam bentuk tabel serta grafik yang menggambarkan kinerja *database server* dalam merespon permintaan yang dilakukan oleh aplikasi *client.*

4. Analisis

Dari hasil penelitian dan pembahasan yang telah dilakukan mengenai konsep serta cara kerja kedua model, dapat diambil suatu analisis data dalam bentuk tabel yang menggambarkan perbandingan proses kerja yang terjadi antara model *two-tier* dengan *three-tier*.

Tabel 4. Perbandingan cara kerja aplikasi

Deskripsi	Model Two-Tier	Model Three-Tier
Proses koneksi *client* ke *server* dilakukan secara langsung	Ya	-
Program dan prosedur aplikasi dilakukan pada *client*	Ya	-
Menu dari aplikasi memiliki keterbatasan	-	Ya
Aplikasi *server* melakukan pengolahan dan pemrosesan data	Ya	-
Sumber daya dari *database server* dapat ditingkatkan	-	Ya

5. Kesimpulan

Aplikasi dalam model *three-tier* terbukti mampu meningkatkan kinerja sistem, hal ini disebabkan pada model *three-tier* ditambahkan sebuah komponen independen di antara aplikasi *client* dengan aplikasi *server* yang disebut sebagai *middle tier*. Komponen ini secara khusus berfungsi untuk melakukan pemrosesan bisnis atau pengolahan data, dengan demikian sumber daya (*resource*) *server* dapat dihemat sehingga kinerja *database server* akan dapat ditingkatkan.

Penerapan aplikasi model *three-tier* dengan memanfaatkan teknologi DCOM terbukti mampu menyediakan pendekatan koneksi yang lebih banyak. Hal ini terutama karena proses registrasi untuk menghubungkan antara *client* dengan *server* yang cukup praktis. Selain itu, fasilitas DCOM sudah termasuk dalam paket perangkat lunak visual seperti Borland Delphi.

Daftar Pustaka

[1] Cantu, M., 2001, *Mastering Delphi 6,* Sybex Inc, USA.

[2] Date, C.J., 2000, *An Introduction to Database Systems,* Addison Wesley Publishing Company, USA.

[3] Elmasri, R., Navathe, SB.,1994, *Fundamentals of Database System*, The Benjamin/Cummings, New York.

[4] Fathansyah, 1999, *Basis Data*, Informatika Bandung, Bandung.

[5] Mnich, M.A., 1998, *Multi-tier Architectures for Database Connectivity,* http://www.javaexchange.com/

[6] Martina, I., 2002, *Database Client/Server Menggunakan Delphi*, PT. Elex Media Komputindo, Jakarta.

[7] Petroutsos, E., 2002, *Pemrograman Database dengan Visual Basic 6,* Sybex Inc USA.

[8] Ramakrishnan, R., 1998, *Database Management Systems,* The Mc Graw-Hill Companies, Inc, Singapore.

[9] Savitri, 2004, *Perbandingan Model Two-Tier dengan Three-Tier dalam Arsitektur Client/Server untuk Mengolah Perintah Query pada Aplikasi Database,* Universitas Ahmad Dahlan, Yogyakarta.

[10] Siebold, D., 2001, *Visual Basic Developer's Guide to SQl Server,* Sybex Inc, USA.

[11] Wiryana, I.M., 2003, *Three-Tiers Client Server*, http://www.nakula.rvs.uni-bielefeld.de/made/.

[12] http://www.delphi.com/

[13] http://www.delphi.about.com/

[14] http://www.geo.yahoo.com/

[15] http://www.geocities.com/visiweb/