Data Storage Engine Sebagai Media Penyimpanan Dalam Jaringan Lokal

Laurentius Risal Subrata, Yohanes Adhi Nugraha
Program Studi Teknologi Informasi
Fakultas Teknologi Informasi, Universitas Kristen Maranatha
Jl. Prof. Drg. Suria Sumantri No. 65 Bandung 40164
Email: laurentius.risal@itmaranatha.org

Abstract

Julia Cage Bradley said, "Many computer applications require that data be saved from one run to the next. Although the most common technique is to use database, many times a database is overkill. Perhaps you just need to store a small amount of data ... " [2] *Store data with file indeed be used in programming because of consideration small amount of data. Programmer also using it and not using database because hopes for good security, optimalization and any exclusive aspects. Because of some opinion above, this project engineered file data manager called "Data Storage Engine". This application give developers a storage system for any application that works in local area network. The engineering has produce one application with server centric architechture, and the implementation is being demonstrated by LanMemo, memo application that works in local area network.*

***Keywords** : file data, data storage engine, data storage system, small data, small amount data*

1. Pendahuluan

Pada tahun 2005, *game* telah berkembang sangat pesat dalam banyak segi, seperti grafis, tantangan dan teknologi yang tertanam di dalamnya. Selain berkembang pada PC dan *console standalone* (*Standalone* = tidak terhubung dalam jaringan), *game* juga berkembang pesat dalam jaringan, baik jaringan kabel maupun nirkabel. Dalam perkembangan yang pesat sebuah *game*, tertanam teknologi *file data* yang menjadi media penyimpanan seluruh informasi yang diperlukan oleh *game* tersebut. Sampai kini, sebagian besar *game* yang dikembangkan, mengembangkan sendiri pula media penyimpanannya, tidak mengintegrasikannya dalam *database* yang sudah ada, seperti Microsoft Access atau SQL Server. Hal ini disebabkan kebutuhan para pengembang akan media penyimpanan yang aman, berkapasitas kecil dan eksklusif untuk optimalisasi.

Dalam pengembangan aplikasi-aplikasi pun *file data* banyak digunakan seperti untuk penyimpanan *setting* pada aplikasi, *file* pengolah data sementara (*temporary file*), *hit counter*, pengolah kata (*wordprocessor*) atau pengolah data. (*spreadsheet*)

Karena berbagai pendapat tersebut, dalam karya ilmiah berjudul "*Data Storage Engine* Sebagai Media Penyimpanan Data Dalam Jaringan Lokal" dirancang satu sistem penyimpan data alternatif, yaitu pengolah *file data* yang melayani kebutuhan untuk menyimpan data dalam sebuah *server*.

Pengolah *file data* ini dibuat khusus terutama untuk penyimpanan data berukuran kecil, dan didesain sedemikian rupa sehingga para pengembang dapat mengembangkan aplikasi mereka menggunakan perangkat lunak ini sebagai media penyimpanan.

2. Solusi Pengembangan Manajemen File Data

Database merupakan penyimpanan data terintegrasi, bekerja dengan mekanisme *Database Management System* (DBMS) untuk mengatur segala hal yang berhubungan dengan data [3], baik metoda penyimpanan, metoda pengiriman data, penerimaan permintaan data, perlindungan data dan elemen lainnya yang terintegrasi di dalamnya. Penyimpanan data dengan berbagai integrasi yang dilakukan ini mengakibatkan kurangnya efisiensi penggunaan untuk penyimpanan data yang memiliki sedikit *record* atau berkapasitas kecil, dikarenakan besarnya kapasitas *file* yang diperlukan untuk menyimpan data tersebut. Dalam perancangan manajemen *file data,* ruang lingkup pengembangan Sistem Manajemen (*Management System*) dipersempit sehingga data yang berukuran kecil tidak memerlukan kapasitas yang jauh lebih besar dari ukuran data aslinya ketika disimpan dalam media penyimpanan sekunder.

Dalam pengembangan aplikasi untuk mengelola *file data* yang diberi nama *Data Storage Engine*, data akan disimpan dengan teknik serialisasi, dan akan mengintegrasikan enkripsi dengan metoda *block cipher* "tripledes". Aplikasi ini dirancang sebagai aplikasi untuk keperluan umum (*general purpose application*) dan memiliki fleksibilitas pengembangan.

Aplikasi dikembangkan dengan menggunakan bahasa pemrograman C#. Kemudian *Data Storage Engine* akan dikoneksi ke dalam *web service*, yang dalam hal ini menggunakan IIS sebagai sistem *server*. Penggunaan *web service* dipertimbangkan karena merupakan satu *service* yang bersifat *platform independence*. *Web service* dikembangkan dengan harapan kompatibilitas, yaitu agar berbagai *platform* bahasa pemrograman dapat melekatkan (*embed*) aplikasi yang dikembangkan kepadanya.

Fitur berikut yang perlu dikembangkan dalam aplikasi ini adalah manajemen pemindahan dan pengaturan *folder* data. Data yang disimpan akan diletakkan pada posisi *folder* tertentu dalam *system drive*, namun posisi *folder* tersebut akan dapat dipindahkan ke *drive* dan *folder* yang ditentukan oleh pengelola aplikasi.

Sehingga secara umum aplikasi *Data Storage Engine* terdiri atas tiga bagian, yaitu:
1. Bagian penyimpanan data, yang adalah bagian utama dari perangkat lunak yang mengatur transaksi data
2. Antarmuka bagi *administrator* untuk melakukan *setting* aplikasi.
3. *Web service* yang menghubungkan penyimpanan data dengan bagian *server* aplikasi.

3. Model Proses dan Pengujian

Dalam bab 2 telah dijelaskan bahwa dalam pengembangannya, aplikasi *Data Storage Engine* harus memiliki satu fleksibilitas pengembangan. Karena alasan tersebut, aplikasi dikembangkan dengan metoda *Spiral*. Pengevaluasian yang dilakukan meliputi kesimpulan dari analisa atas kapasitas, kinerja, validasi, verifikasi, dan implementasi, dengan syarat bahwa dalam hasil akhir *cycle* pertama model proses ini, tidak ada *error* untuk penyimpanan satu data.

Test kapasitas data dilakukan pada *cycle* pertama model proses ini. Dalam perancangan awal harus sudah ditentukan bentuk data yang efisien untuk penyimpanan data kecil. Akhir dari *cycle* terakhir, besar kapasitas data pada aplikasi ini dibandingkan dengan kapasitas pada *database*.

Test verifikasi data banyak dilakukan pada akhir *cycle* pertama. Baik dalam penyimpanan data tanpa enkripsi dan dengan enkripsi, serta proses deserialisasi dan deskripsi data, tiap-tiap *class* yang dirancang harus menunjukkan *behavior* yang sesuai dengan desain aplikasi.

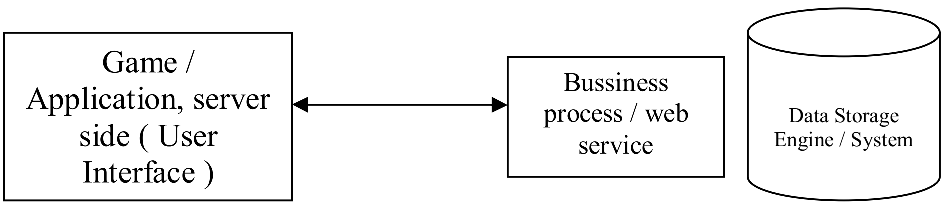
Melewati test verifikasi, dilakukan test kinerja. Kecepatan transaksi data diuji dengan *database* sebagai pembanding.

Test validasi dan implementasi merupakan test kelayakan penggunaan aplikasi, mengingat aplikasi *Data Storage Engine* merupakan aplikasi bersifat *general purpose*, harus dapat menunjukkan fleksibilitas ketika dilekatkan dengan aplikasi lain. Implementasi secara praktis aplikasi ini adalah aplikasi LANMemo, yaitu aplikasi memo yang bekerja dalam jaringan lokal.

4. Desain Aplikasi

Class diagram untuk bagian basis aplikasi akan dilampirkan dalam lampiran A.

Arsitektur aplikasi yang digunakan pada prinsipnya adalah 2Tier / *Server Centric*. Dari gambar 1 dapat dilihat ciri dari *server centric*, adalah *user interface* dibuat terpisah dari penyimpanan data, dan di bagian *server* tidak memiliki *user interface*, hanya penyimpanan data (dalam hal ini *Data Storage*) dan proses bisnis (yaitu *web service*) menjadi satu dengan *Data Storage*. Aplikasi *data storage* akan menjadi bagian *server* dari arsitektur aplikasi *server centric*, sedangkan untuk *user interface* akan direkayasa aplikasi "LAN Memo".



Gambar 1. Arsitektur aplikasi Data Storage Engine

5. Uji Hasil Kapasitas Data

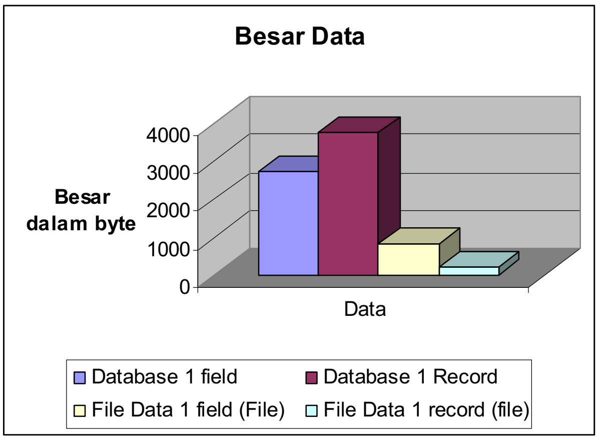
Pengujian kapasitas data dilakukan dengan melihat kapasitas yang diperlukan untuk menyimpan satu *record data* oleh *database* dan *file data.*

Dalam tabel 1 dapat terlihat bahwa hasil yang diperoleh dari pengujian kapasitas data antara *database* dan *file data* menunjukkan perbedaan yang signifikan.

Tabel 1. Hasil Pengujian Perbandingan Database dan File Data

Data	Database 1 field	Database 1 record	File Data 1 field (file)	File Data 1 record (file)
Besar Data (Byte)	2735	3789	848	240

Pada gambar 2, grafik menunjukkan sangat kecilnya kapasitas yang diperlukan oleh *file data* untuk menyimpan satu *record* data.



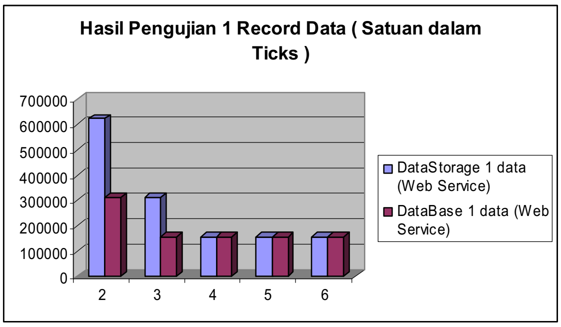
Gambar 2. Grafik Hasil Pengujian Perbandingan Database dan File Data

6. Uji Kinerja

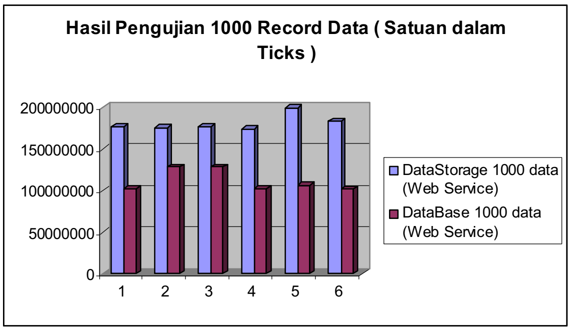
Kinerja diuji dengan mengkodekan suatu aplikasi *database* menggunakan *web service* sebagai pembanding. Kriteria yang digunakan adalah *database* tidak menggunakan *dataset* untuk bertransaksi, namun menggunakan metoda *direct access*. Kedua aplikasi diuji kinerjanya dalam satuan waktu 100 *nanosecond* atau *ticks* agar penghitungan dapat seteliti mungkin, dan dilakukan sebanyak enam kali percobaan.

Tabel 2. Hasil pengujian kinerja dalam satuan 100 nanosecond

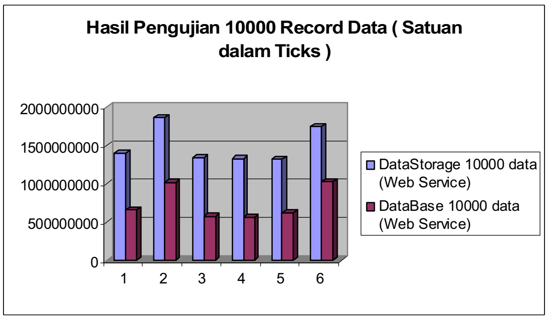
Aplikasi / Percobaan	DataStorage 1 data (Web Service)	DataBase 1 data (Web Service)	DataStorage 1000 data (Web Service)	DataBase 1000 data (Web Service)	DataStorage 10000 data (Web Service)	DataBase 10000 data (Web Service)
1	60781250	25937500	176718750	102031250	1402343750	659843750
2	625000	312500	175312500	128281250	1872031250	1022968750
3	312500	156250	176875000	128281250	1350000000	579218750
4	156250	156250	173750000	102187500	1329687500	566718750
5	156250	156250	198906250	106093750	1325781250	625468750
6	156250	156250	182968750	101562500	1754062500	1031406250
rata-rata	10364583.33	4479166.67	180755208.33	111406250.00	1505651041.67	747604166.67
Nilai terbesar	60781250.00	25937500.00	198906250.00	128281250.00	1872031250.00	1031406250.00
Nilai terkecil	156250.00	156250.00	173750000.00	101562500.00	1325781250.00	566718750.00



(1)



(2)



(3)

Gambar 3. Hasil pengunjian kinerja untuk (1) satu data, (2) seribu data, dan (3) sepuluh ribu data

Dalam gambar 3.(1) penyimpanan satu data antara *database* dan *data storage engine* masih menunjukkan kinerja yang sama cepat ketika kondisi sudah stabil. Dalam pengujian seribu data, pada gambar 3.(2), *data storage engine* menunjukkan penurunan kinerja, sehingga dalam gambar dapat terlihat kecepatan *database* antara dua pertiga waktu yang dibutuhkan *data storage engine* untuk menyimpan data. Dalam gambar 3.(3), kecepatan *data storage engine* menurun lagi dalam menyimpan data sebanyak sepuluh ribu, hingga memerlukan waktu dua kali kecepatan *database* untuk menyimpan data.

7. Uji Kelayakan Guna dengan Aplikasi LANMemo

Aplikasi data storage engine bersifat *general purpose*, dan digunakan dengan cara dilekatkan (*embed*) pada aplikasi yang berada dalam jaringan lokal. Oleh karena itu, perlu uji kelayakan, untuk melihat apakah aplikasi ini memiliki suatu fleksibilitas untuk digunakan oleh aplikasi lain.

Masing-masing *form* dalam LANMemo, kecuali *form Main*, menggunakan aplikasi *data storage engine* sebagai media penyimpanan dan pengambilan data.

Gambar 4 Menunjuk pada *form Login*. Setiap pengguna aplikasi ini memerlukan satu *username* untuk dapat bertukar pesan.

Gambar 4. LANMemo : Form Login

Pada gambar 5, f*orm Register* memperlihatkan bagaimana seseorang dapat melakukan *register* untuk dapat menjadi member aplikasi.

Gambar 5. LANMemo : Form Register

Gambar 6 memperlihatkan tampilan menu utama aplikasi LANMemo.

Gambar 6. LANMemo : Form Main

Form Compose yang terlihat pada gambar 7 digunakan untuk mengirim pesan kepada *user* yang lain.