Pemilihan Arsitektur *Smart Client* dan *Thin Client* pada Perangkat dan Jaringan Nirkabel

Hapnes Toba
Program Studi Teknologi Informasi
Fakultas Teknologi Informasi Universitas Kristen Maranatha
Jl. Prof. Drg. Suria Sumantri No. 65 Bandung 40164
Email: hapnes.toba@eng.maranatha.edu; hapnes.toba@itmaranatha.org

Abstract

Mobile solutions can be built in many different ways, used on many different devices, operate over many different networks, and integrate with many different back-end systems. The task of building a mobile solution can often be confusing given the many technology choices and implementation approaches. This paper explores smart and thin client architectures that have been successfully used by enterprises to build mobile solutions with new technologies in database management systems and different programming languages. The paper will also explore the differences between smart client and thin client mobile architectures, and when people would want to use each. It will then examine the details of common smart client architectures, including what type of back-end systems they integrate with, the types of devices they support, how data is managed within the architecture, and development languages that can be used to build the application.

Keywords**:** *mobile architecture, thin client, smart client, mobile solutions, application integration.*

1. Pendahuluan

Para pengembang aplikasi nirkabel seringkali harus memutuskan arsitektur aplikasinya dengan berbagai pertimbangan komponen arsitektur, yang terutama adalah:

- Lokasi tempat aplikasi digunakan;
- Kemampuan penyimpanan data;
- Kemampuan transportasi data.

Setiap pertimbangan akan berimbas pada pemilihan teknologi yang cocok dan penanganan teknik pemrograman yang diperlukan. Dalam artikel ini akan dilakukan analisis terhadap dua arsitektur yang paling sering digunakan dalam pengembangan aplikasi nirkabel, yaitu: *smart client* dan *thin client*.

Arsitektur *thin client* dalam solusi *web / client server* dikenal dengan kemudahannya dalam tahap pengembangan dan penggunaan karena hanya mengandalkan satu tempat pengendalian, yaitu di dalam *(web) server* saja. Penggunaan arsitektur *thin client* ini akan sangat handal jika diimplementasikan pada solusi berbasiskan PC, misalnya pada aplikasi-aplikasi *web*. Namun arsitektur *thin client* kadangkala menghadapi banyak kendala jika akan diimplementasikan dalam suatu solusi nirkabel.

Sebaliknya dengan penggunaan arsitektur *smart client*, aplikasi tidak dijalankan pada *server*, tetapi pada perangkat (*device*) bergerak. *Server* hanya dijadikan sebagai tempat untuk sinkronisasi data, dan hal ini menyebabkan beban *server* menjadi lebih ringan.

Dalam artikel ini akan dibahas masing-masing keuntungan dan kerugian penggunaan *thin client* dan *smart client*, dan pada bagian selanjutnya dari artikel akan dibahas secara lebih terperinci strategi-strategi khusus dalam pengembangan aplikasi nirkabel dengan penggunaan arsitektur *smart client*.

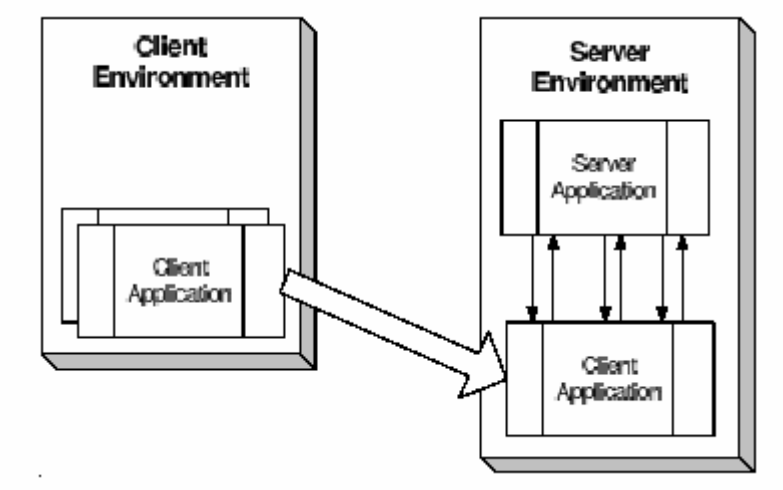
2. Perbandingan Antara Komponen Arsitektur *Smart client* dan *Thin client*

Dalam paragraf ini akan dituliskan berbagai aspek yang dapat dijadikan sebagai pembanding antara *smart* dan *thin client* untuk suatu pengembangan aplikasi nirkabel. Secara definisi dalam penjelasan mengenai *smart* dan *thin client*, dapat dibedakan ke dalam tiga komponen arsitektur, yaitu: lokasi jalannya aplikasi, kemampuan penyimpanan data dan kemampuan transportasi data (bandingkan juga Gambar 1 dan 2).

Tabel 1 Komponen Arsitektur *Smart* vs. *Thin Client*

Komponen Arsitektur	Smart Client	Thin Client
Lokasi aplikasi	Lokasi aplikasi adalah lokal pada perangkat nirkabel.	Penggunaan aplikasi berbasis web / Internet yang dijalankan pada suatu server. Biasanya client menggunakan *web-browser* pada aplikasi nirkabelnya. [2]
Penyimpanan / Storage	Data disimpan lokal dan diakses melalui aplikasi yang ada pada perangkat nirkabel.	Data disimpan pada Web server dan diakses melalui aplikasi web pada perangkat nirkabel.
Transportasi / Transport	Data ditransportasikan di antara perangkat nirkabel dan server pada saat proses sinkronisasi. Arsitektur transportasi data biasanya dirancang untuk menangani kapasitas *bandwidth* yang rendah ataupun koneksi nirkabel yang tidak terpercaya.	Data dikirimkan ke perangkat nirkabel ketika suatu request halaman *Web* dilakukan dari perangkat nirkabel. Setiap data yang diinputkan oleh pengguna aplikasi secara segera dikirimkan ke *Web server*. Penghantaran data menggunakan protokol HTTP, GPRS atau protokol aplikasi *web* lainnya.

Sebagaimana yang terlihat melalui tabel di atas, pembuatan aplikasi secara pendekatan *thin client* dibandingkan dengan *smart client* akan lebih sederhana. Pengembang aplikasi tidak memerlukan penyesuaian terhadap jenis perangkat nirkabel yang kemungkinan akan dipakai oleh si pengguna. Aplikasi cukup dibuat dalam suatu *Web server*, pemakaian aplikasi sepenuhnya tergantung pada *Web Browser* pada perangkat nirkabel yang digunakan. Pengembangan ataupun perubahan terhadap aplikasi juga cukup diinstalasikan pada *Web server*.



Gambar 1 Arsitektur Umum *Thin Client* [6]

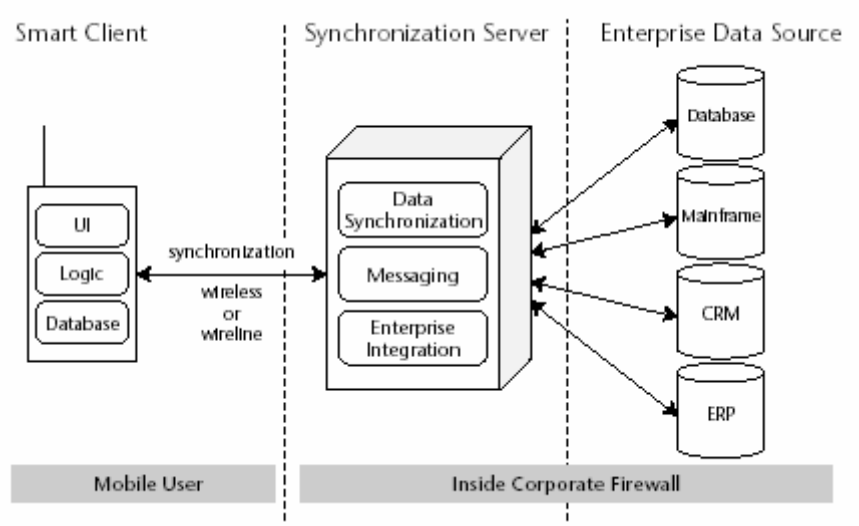
Penggunaan *thin client* sangat memerlukan koneksi jaringan yang handal sehingga pengguna dapat mengakses aplikasi secara konsisten, dan perihal koneksi ini merupakan salah satu faktor yang menjadi permasalahan dalam suatu lingkungan nirkabel. Biaya akses aplikasi bisa menjadi sangat mahal dan ditambah kapasitas jaringan (*bandwidth*) yang kadang-kadang tidak stabil untuk perangkat nirkabel.

3. Perbandingan Karakteristik Aplikasi *Smart Client* dan *Thin Client*

Ketergantungan yang berlebihan terhadap koneksi nirkabel akan menimbulkan situasi yang problematis. Hal ini disebabkan oleh beberapa alasan, di antaranya:

- Terbatasnya area (*coverage*) yang dilingkupi oleh suatu jaringan nirkabel;
- Interferensi transmisi yang menyebabkan turunnya kapasitas koneksi nirkabel karena pengaruh cacatnya peralatan fisik (seperti antena atau *access point*) dan adanya perbedaan ruang lingkup koneksi antar daerah;
- Kecepatan transmisi jaringan nirkabel lebih rendah dibandingkan dengan koneksi yang paten (*landline*);
- Biaya pengaksesan jaringan nirkabel yang lebih tinggi dibandingkan dengan jaringan paten.

Dengan alasan-alasan di atas, produktifitas pengguna aplikasi dalam suatu arsitektur *thin client* diperkirakan akan menjadi lebih rendah dibandingkan dengan *smart client*, dan hal ini menyebabkan penggunaan arsitektur *thin client* untuk suatu lingkungan nirkabel menjadi tidak menguntungkan.



Gambar 2 Arsitektur Umum *Smart Client* [3]

Berikut ini akan dituliskan beberapa karakteristik aplikasi yang berhubungan dengan solusi nirkabel, dan bagaimana arsitektur *thin client* dan *smart client* mempengaruhinya:

Tabel 2 Karakteristik Solusi dan Pengaruh Arsitektur *Smart –Thin Client*

Karakteristik Solusi	Smart Client	Thin Client
Ketersediaan jaringan: aplikasi nirkabel memerlukan akses jaringan untuk dapat melakukan pertukaran data.	Pengguna dapat produktif setiap waktu karena aplikasi berada lokal pada perangkat nirkabel, tidak selalu bergantung pada ketersediaan jaringan. Pertukaran data dapat dilakukan (jika tersedia, tidak harus selalui nirkabel) melalui jaringan paten, nirkabel ataupun dengan kabel data.	Pengguna aplikasi nirkabel hanya produktif ketika tersambung dengan jaringan. Sayangnya setiap jaringan nirkabel memiliki perbedaan kemampuan hantaran data yang berbeda-beda, dan ini akan sulit jika pengguna benar-benar "*mobile*". Koneksi yang tidak terpercaya / *drop* akan sangat mempengaruhi produktifitas pengguna.
Responsi aplikasi: semakin lama waktu yang diperlukan aplikasi untuk merespon pengguna, akan semakin tidak produktifnya pengguna.	Responsi aplikasi terhadap input / interaksi dari pengguna langsung terasa karena data dan antarmuka tidak ditransportasikan melalui jaringan pada saat digunakan dari *Web server*.	Penundaan / *delay* antara interaksi pengguna dengan aplikasi bisa menjadi sangat lama, karena data dan antarmuka harus ditransmisikan melalui jaringan nirkabel yang kemungkinan lambat dan tidak terpercaya.
Penggunaan baterai: perangkat nirkabel sangat bergantung pada ketersediaan asupan listrik.	Karena koneksi dengan jaringan nirkabel tidak harus selalu terjadi, penggunaan baterai akan menjadi lebih irit.	Hubungan yang konstan untuk mengirim dan menerima data dengan jaringan nirkabel memerlukan asupan tenaga

Karakteristik Solusi	Smart Client	Thin Client
		yang besar, dan ini menyebabkan baterai bisa menjadi boros.
Sekuritas: data-data suatu organisasi / enterprise yang dapat diakses melalui perangkat nirkabel, harus terproteksi dengan baik untuk menghindari penyalahgunaan.	Data yang disimpan di dalam perangkat ataupun yang ditransmisikan dapat dienkripsi. Data yang telah dienkripsi tersebut tidak dapat diakses tanpa adanya password yang valid.	Data hanya disimpan dalam *Web server*, dan secara teratur diakses oleh perangkat nirkabel. Untuk menjamin keamanan data dapat menggunakan HTTPS ataupun kontrol terhadap *user sessions*.
Pengembangan: aplikasi nirkabel harus dapat diakses dengan berbagai perangkat nirkabel: handphone, PDA, laptop. Masing-masing memiliki tipe dan karekteristinya sendiri-sendiri.	Aplikasi *smart client* harus terinstal secara lokal di dalam perangkat nirkabel, dan biasanya memerlukan trik-trik khusus pada saat instalasinya.	Karena aplikasi berada pada *Web server*, pengembangan aplikasi dapat dilakukan secara leluasa tanpa perlu melakukan instalasi pada setiap perangkat nirkabel, selama menggunakan protokol *web* / Internet yang berlaku.
Portabilitas: aplikasi nirkabel dimungkinkan untuk digunakan pada banyak perangkat ataupun sistem operasi.	Aplikasi *smart client* biasanya dibuat khusus untuk sistem operasi tertentu. Jika ingin digunakan pada perangkat dengan sistem operasi yang berbeda harus dilakukan penyesuaian, dan ini memerlukan keahlian khusus.	Aplikasi *thin client* biasanya memanfaatkan Web browser yang terinstal pada perangkat nirkabel. Selama browser tersebut mendukung protokol yang diperlukan, aplikasi akan dapat dijalankan tanpa memperdulikan sistem operasi pada perangkatnya. [1]

Karakteristik yang tertulis pada Tabel 2 di atas, akan mempengaruhi implementasi dan keberhasilan penggunaan aplikasi nirkabel di organisasi / perusahaan pada dua aspek:

- Pengaruh terhadap produktifitas pengguna aplikasi nirkabel;
- Pengaruh terhadap investasi teknologi informasi pada saat pengembangan, dan perawatan aplikasi nirkabel.

Aristektur *thin client* akan memberikan keuntungan kepada saat tahap pengembangan, namun karena kemungkinan kurangnya produktifitas pengguna akan menyebabkan lamanya masa balik modal (*return on investment*). Sementara kalau dilihat dari sisi arsitektur *smart client*, akan memerlukan banyak campur tangan pihak pengelola teknologi informasi dalam pengembangan dan pemeliharaan, dan ini akan mempengaruhi tingkat penggunaan aplikasi dan produktifitas penggunanya.

4. Penggunaan Arsitektur *Smart Client*

Memperhatikan berbagai keunggulan arsitektur *smart client*, dibandingkan dengan arsitektur *thin client*, maka dalam sisa artikel ini, penulis tertarik untuk

menelaah secara lebih dalam lagi berbagai arsitektur dan teknologi pendukung arsitektur *smart client*.

Secara garis besar, terdapat empat jenis implementasi arsitektur untuk mengembangkan solusi nirkabel. Keempat arsitektur berikut sebagaimana telah diteliti oleh Sybase, Inc. pada implementasi jaringan nirkabel, menonjolkan pengelolaan data serta jenis sistem pendukung (*back-end*) yang digunakan [5]:

1. Sinkronisasi basis data;
2. Penyimpanan dan penghantaran pesan;
3. Sinkronisasi basis data dengan penyimpanan dan penghantaran pesan;
4. Replikasi berkas untuk pertukaran data.

Sinkronisasi Basis Data

Dengan arsitektur *smart client* jenis ini, data dengan mudah dikembangkan dari basis data organisasi / perusahaan menggunakan sistem basis data *remote* yang berada pada perangkat nirkabel. Data yang dibuat, dimodifikasi atau dihapus dalam suatu basis data *remote* dapat disinkronisasikan kembali dengan basis data aslinya pada *server* ataupun disinkronisasikan dengan basis data *remote* pada perangkat nirkabel yang lainnya [3]. Model arsitektur dapat dilihat pada Gambar 3.

Aplikasi nirkabel dapat dikembangkan menggunakan berbagai macam perangkat lunak pendukung dan bahasa pemrograman. Perangkat nirkabel yang dapat dimanfaatkan untuk arsitektur ini di antaranya *laptop, PDA, smartphone* dan *desktop* dengan sistem operasi Windows 32, Windows Mobile, Windows CE ataupun Palm OS.

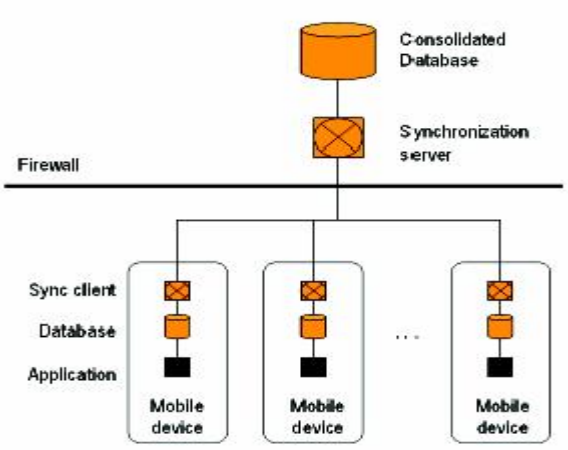
Konfigurasi umum untuk arsitektur sinkronisasi basis data misalnya:

Perangkat Lunak pada Perangkat Nirkabel:

- Mesin basis data seperti: UltraLite atau SQL Anywhere
- MobiLink Synchronization Client
- Perangkat lunak aplikasi bergerak

Perangkat Perantara (*Middleware*):

- MobiLink Synchronization pada server



Gambar 3 Smart Client dengan Sinkronisasi Basis Data

Integrasi *Enterprise*:

- Basis data *remote* yang dapat disinkronisasikan dengan basis data asli, seperti SQL Anywhere, Oracle, SQL Server, IBM DB2 atau Sysbase ASE

Bahasa Pemrograman:

- C/C++, VB, VB .Net, C#, ASP, PHP, Perl, Java, Python, Delphi, Pocket Builder, PowerBuilder

Perangkat "*Remote*":

- Pocket PC, Palm OS, Smartphone, Windows, Solaris, Linux

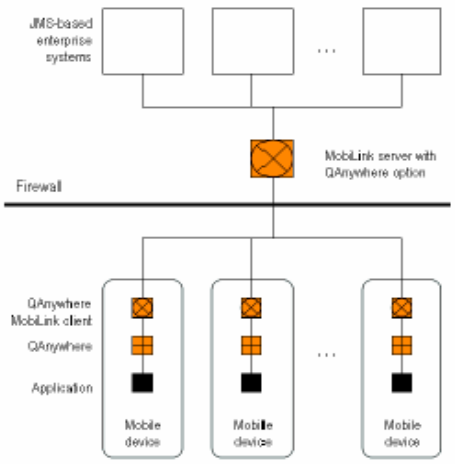
Konektifitas Jaringan:

- Nirkabel, Wifi 802.11, W-LAN, GPRS, GSM, CDMA, Wired Ethernet, Modem

Penyimpanan dan Penghantaran Pesan

Penghantaran pesan merupakan suatu strategi untuk mengkoneksikan sekumpulan sistem teknologi informasi melalui pertukaran data yang sifatnya tidak sinkron (*asynchronous*). Dengan model ini, sistem-sistem yang terdistribusi dapat dibandingkan dengan basis data aslinya pada *server* dimana data secara terpusat disimpan. Gambaran arsitektur selengkapnya dapat dilihat pada Gambar 4.

Informasi disalurkan antar sistem dengan cara menempatkan pesan pada antrian pada suatu pusat penyimpanan pesan atau pada suatu sistem *remote*. Pada setiap antrian pesan akan diinformasikan untuk aplikasi-aplikasi yang memerlukan sinkronisasi (dapat dianalogikan dengan *log* perubahan). Dengan cara demikian pengguna akan mengetahui perubahan yang terjadi melalui pesan pada antrian. Contoh yang paling nyata adalah pada penggunaan Java Message Service (JMS).



Gambar 4 Smart Client dengan Penyimpanan dan Penghantaran Pesan

Konfigurasi umum untuk arsitektur penyimpanan dan penghantaran pesan misalnya:

Perangkat Lunak pada Perangkat Nirkabel:
- Mesin basis data seperti: SQL Anywhere
- QAnywhere Synchronization Client
- Perangkat lunak aplikasi bergerak

Perangkat Perantara (*Middleware*):
- MobiLink Synchronization Server dengan QAnywhere

Integrasi *Enterprise*:
- JMS-based Messaging System seperti MQ Series atau server aplikasi berbasis JMS, seperti Sybase EAServer, BEA WebLogic, atau IBM WebSphere
- TIBCO, SAP, PeopleSoft, Siebel, dan lain-lain dengan menggunakan Sybase Unwired Orchestrator

Bahasa Pemrograman:
- Aplikasi dapat ditulis dengan C/C++, C#, Java

Perangkat "*Remote*":
- Pocket PC, Palm OS, Smartphone, Windows, Solaris, Linux

Konektifitas Jaringan:
- Nirkabel, Wifi 802.11, W-LAN, GPRS, GSM, CDMA, Wired Ethernet, Modem

Sinkronisasi Basis Data dengan Penyimpanan dan Penghantaran Pesan

Beberapa organisasi / perusahaan melakukan kontrol terhadap perubahan-perubahan yang dilakukan pada basis data asli dengan cara mengirimkan modifikasi yang terjadi melewati *middle-tier* yang berjalan pada *server* aplikasi.

Dengan melakukan strategi kombinasi antara sinkronisasi dengan penyimpanan dan penghantaran pesan, data dapat disinkronisasikan sepenuhnya antara server dan aplikasi. Setiap modifikasi yang terjadi terlebih dahulu disimpan pada *middle-tier*, dan melengkapinya dengan antrian pesan, untuk kemudian dapat diteruskan pada basis data asli pada saat pesan perubahan disetujui. Lihat Gambar 5 untuk gambaran arsitektur selengkapnya.

Konfigurasi umum untuk arsitektur penyimpanan dan penghantaran pesan misalnya:

Perangkat Lunak pada Perangkat Nirkabel:
- Mesin basis data seperti: SQL Anywhere
- QAnywhere Synchronization Client
- Perangkat lunak aplikasi bergerak

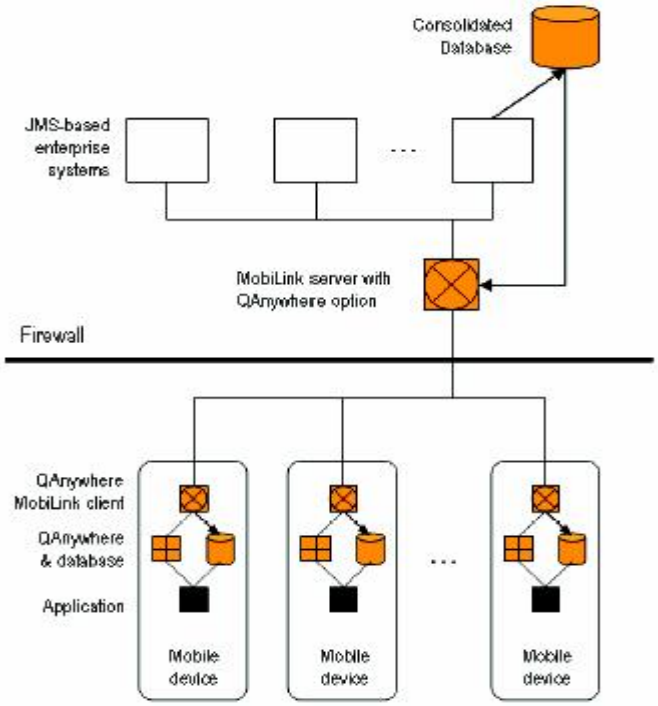
Perangkat Perantara (*Middleware*):
- MobiLink Synchronization Server dengan QAnywhere

Integrasi *Enterprise*:

- Basis data untuk aplikasi nirkabel, seperti: SQL Anywhere, Oracle, Microsoft SQL Server, IBM, DB2 atau Sybase ASE;
- *Send and receive messages* dengan JMS-based *messaging system* seperti MQ Series atau server aplikasi berbasis JMS, seperti Sybase EAServer, atau IBM WebSphere;
- *Send and receive messages* dengan TIBCO, SAP, PeopleSoft, Siebel atau sistem lainnya.

Bahasa Pemrograman:

- Aplikasi dapat ditulis dengan C/C++, C#, Java



Gambar 5 Smart Client Sinkronisasi Basis Data dengan Penyimpanan dan Penghantaran Pesan

Perangkat "*Remote*":

- Pocket PC, Palm OS, Smartphone, Windows, Solaris, Linux

Konektifitas Jaringan:

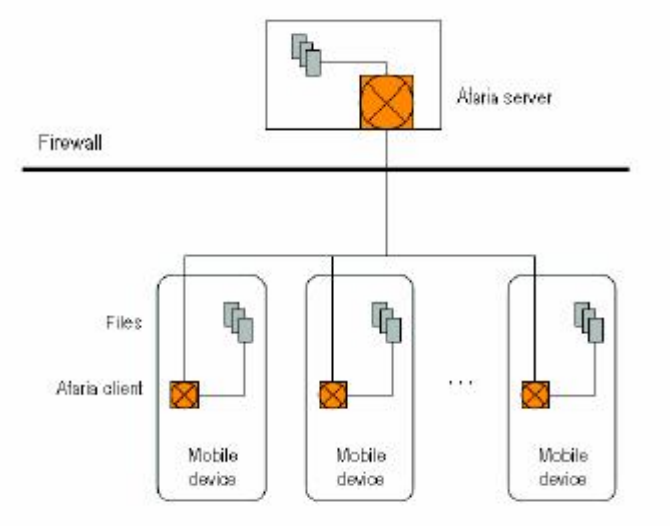
- Nirkabel, Wifi 802.11, W-LAN, GPRS, GSM, CDMA, Wired Ethernet, Modem

Replikasi berkas untuk pertukaran data

Melalui adanya replikasi berkas maka data yang tersimpan pada basis data asli akan tetap terjaga kualitasnya. Hal ini dapat dilakukan dengan dibentuknya berkas *back-up* terhadap perubahan yang dilakukan, sehingga jika terjadi kesalahan

ataupun hasil yang minim dari modifikasi, data asli tetap dapat digunakan. Sinkronisasi dilakukan jika berkas hasil modifikasi sudah "disetujui" untuk digunakan.

Salah satu cara yang dapat digunakan untuk menjalani strategi replikasi berkas adalah dengan adanya "*session manager*" (sebagai contoh: pada Afaria Server), yang mengatur hak akses perubahan yang dilakukan oleh pengguna server aplikasi. Konsekuensi dilakukannya pengelolaan sesi adalah terjaminnya hasil perubahan yang dilakukan terhadap basis data asli. Gambar 6 menunjukkan gambaran arsitektur dengan replikasi data.



Gambar 6 Smart Client dengan Replikasi Data

Konfigurasi umum untuk arsitektur penyimpanan dan penghantaran pesan misalnya:

Perangkat Lunak pada Perangkat Nirkabel:
- Afaria Client dengan Session Manager
- Perangkat lunak aplikasi bergerak

Perangkat Perantara (*Middleware*):
- Server Afaria dengan Session Manager

Integrasi *Enterprise*:
- Data dapat dipertukarkan dengan sistem lainnya melalui kustomisasi script (JScript atau VBScript) atau program kustomisasi lainnya yang dibuat oleh pemrogram.

Bahasa Pemrograman:
- C/C++, VB, VB .Net, C#, ASP, PHP, Perl, Java, Python, Delphi, Pocket Builder, PowerBuilder

Perangkat "*Remote*":
- Pocket PC, Palm OS, Smartphone, Windows, Solaris, Linux, Symbian

Konektifitas Jaringan:

- Nirkabel, Wifi 802.11, W-LAN, GPRS, GSM, CDMA, Wired Ethernet, Modem

5. Simpulan

Dalam artikel ini telah dibahas berbagai pertimbangan pemilihan arsitektur *smart client* atau *thin client* dalam pengembangan aplikasi nirkabel. Di antara pertimbangan yang telah dibahas akan dapat menjawab pertanyaan-pertanyaan sebagai berikut:

- Perbaikan alur kerja (*workflow*) seperti apa yang akan diakibatkan pengimplementasian aplikasi nirkabel ?
- Tipe jaringan seperti apakah yang dapat mengakomodasikan alur kerja yang baru tersebut ?
- Tipe perangkat bergerak/ nirkabel apakah yang akan dipakai oleh pengguna ?
- Dengan sistem apa sajakah aplikasi nirkabel akan terintegrasi ?
- Seperti apakah pengelolaan data yang terjadi pada organisasi / perusahaan ?

Dengan memilih arsitektur dan teknologi yang tepat, pengembang aplikasi nirkabel akan dapat fokus pada pengembangan aplikasi dan kompleksitas yang biasanya merupakan hambatan pada pengembangan aplikasi nirkabel dapat dihindari karena telah "ditangani" oleh infrastruktur arsitektur sistem itu sendiri.

Referensi

[1] Kim J., Barrato R.A, Nieh J. (2006); *pTHINC: A Thin Client Architecture for Mobile Wireless Web.*
Available: http://www.ncl.cs.columbia.edu/publications/www2006_fordist.pdf. Accessed April: 2006.

[2] *Manolescu D.A., Santamarina G.F (2006);* Java Thin Clients Revisited: An Architecture for Responsive, Live Wireless Application.
Available: http://micro-workflow.com/PDF/mcbdemo.pdf. Accessed: April 2006.

[3] _____. (2006) *Smart Client Overview.*
Available:
http://media.wiley.com/product_data/excerpt/91/04712141/0471214191.pdf.
Accessed: April 2006.

[4] *Sybase iAnywhere Homepage*
Available: http://www.ianywhere.com/. Accessed: April 2006.

[5] _____. *(2006)* Best Practices for Mobile Application Architectures. Whitepaper iAnywhere, May 2006.

[6] Wang, A.I. , Sorensen, C.F., Indal. E. (2003); *A Mobile Agent Architecture for Heterogeneous Devices.;* Dept. Computer and Information Science Norwegian University of Science and Technology.
Availble: http://www.idi.ntnu.no/grupper/su/publ/alfw/woc2003-mobile-agent.pdf . Accessed: April 2006.