Pembuatan Aplikasi Mobile Commander pada Handphone dengan Menggunakan J2ME

Tjatur Kandaga, Fandy Chandra
Program Studi S1 Teknik Informatika
Fakultas Teknologi Informasi, Universitas Kristen Maranatha
Jl. Prof. Drg. Suria Sumantri No. 65 Bandung 40164
Email: tjatur.kandaga@itmaranatha.org, vahn_dee@yahoo.com

Abstract

These days handphone technology advances rapidly. Now handphones has faster processors, bigger memory, faster connections to internet, and touch-screen technology. These better handphone technology gives handphone more power to execute application, so we can build more advanced and complex application. In turn it gives us a chance to use handphones in more areas of our lives. Bigger memory means we can store more files on our handphone and better application to manage those files. Many handphones doesn't have convenient file manager, so this application can fill the gap.
This application built with J2ME (Java 2 Micro Edition) technology, and use GPRS, 3G or HSDPA technology to connect to FTP Server.
The Mobile Commander application has features to manage files on local folders (localhost) as well as files on FTP Server. It has features to copy and paste files, delete files/directory, rename files, change file attributes, encrypt and decrypy files, and upload/download files to FTP Server. When uploading or downloading files the handphones act as a FTP Client.

Keywords: handphone, file manager, FTP client

1. Pendahuluan

Perkembangan teknologi handphone sangatlah cepat akhir–akhir ini. Bahkan ada handphone yang memiliki memory lebih dari 8 GB. Hampir setiap orang sudah memiliki handphone. Tetapi untuk handphone saat ini terutama handphone–handphone yang sudah canggih seperti Nokia series N, *Sony Ericsson*, dan lain lain, kita sering mengalami kesulitan untuk mengatur *file–file* serta folder–folder yang ada dengan rapi. Selain itu kita tidak bisa mengubah property dari sebuah *file* atau folder. Handphone generasi sekarang juga menyediakan fasilitas koneksi internet dengan memanfaatkan jaringan GPRS, 3G atau HSDPA. Koneksi ini dapat dimanfaatkan untuk mengakses FTP Server untuk menyimpan file secara *online*, sehingga dapat meningkatkan aksesibilitas data.

Aplikasi ini dikembangkan karena adanya kondisi seperti yang diceritakan diatas. Dengan dibuatnya aplikasi yang dapat mengatur sistem *file* yang ada pada handphone dan FTP Server, diharapkan pengguna handphone dapat dengan mudah untuk mengatur *file* handphonenya serta pengguna bisa menjaga keamanan informasi dari *file* tersebut dan diharapkan dengan dibuatnya aplikasi ini.

2. Tujuan

Tujuan pembuatan aplikasi ini adalah membantu pengguna handphone dalam mengatur *file – file* atau folder – folder yang berada pada handphone tersebut. Dengan tampilannya yang berbentuk *explorer* akan lebih cepat dan mempermudah bagi penggunanya untuk dilihat dan digunakan. Aplikasi ini juga memungkinkan pengguna untuk melindungi *file – file* yang penting di *handphone*-nya. Selain itu apikasi ini juga menyediakan akses ke FTP server, bisa disebut sebagai aplikasi FTP Client. Aplikasi ini akan berhubungan dengan FTP server yang ada dan kita bisa melakukan proses *login*, upload serta download file terhadap FTP Server. Dengan adanya aplikasi ini, kita bisa meng-update FTP Server kita melalui handphone.

3. Pembatasan Masalah Aplikasi

Beberapa hal yang merupakan batasan pada aplikasi ini yaitu :

- Perangkat *Mobile* atau *handphone* yang digunakan adalah perangkat *mobile* yang mendukung jalannya aplikasi *java* (*J2ME*)
- Fitur tertentu pada Aplikasi ini menggunakan koneksi *WAP* ataupun *GPRS,* sehingga hanya dapat dijalankan pada perangkat *Mobile* atau handphone yang memiliki koneksi ke internet (*GPRS/3G*).
- Aplikasi ini hanya mengatur *file*, maka untuk membuka *file* tertentu hanya dapat dilakukan menggunakan aplikasi yang sesuai.
- Proses Enkripsi dan Dekripsi hanya bisa dilakukan untuk *file* yang berada pada *localhost*, tidak dapat dilakukan untuk *file* yang berada di *FTP Server*
- Hanya bisa mengakses (*remote*) 1 buah FTP Server dalam satu buah koneksi

4. J2ME *File*Connection (JSR-75)

Kita dapat mengakses *file-file* yang berada pada handphone (*localhost*) dengan menggunakan sebuah *library* yang telah disediakan oleh *java* yaitu *FileConnection* (JSR-75). Tidak semua handphone yang mendukung aplikasi *java* memiliki *library* ini. Maka dari itu aplikasi ini hanya berjalan terhadap handphone-handphone yang mendukung atau memiliki *JSR*-75. Kita tidak bisa mengetahui apakah handphone tersebut mendukung *JSR*-75 secara langsung hanya dengan melihat isi didalam handphone tersebut, tetapi kita bisa mengetahuinya dengan mengunjungi *website provider* yang mengeluarkan atau memproduksi handphone tersebut.

FileConnection merupakan sebuah class *interface* dan turunan dari *class Connection*. Fungsi dari class *File*Connection ini adalah memungkinkan programmer untuk berhubungan dengan sistem *file* yang ada pada handphone. Selain bisa berhubungan dengan *file-file* yang ada pada handphone, bisa juga mengatur *file-file* tersebut dengan method-method yang telah disediakan pada library ini seperti membuat direktori baru, menghapus *file* atau direktori, mengganti nama *file* atau direktori dan lain-lain.

Untuk membuka atau melakukan sebuah koneksi ke *file* pada handphone kita membutuhkan sebuah objek *File*Connection.

Sintaks diatas berfungsi untuk membuka sebuah koneksi menggunakan *class* connector, dan mengubah koneksi tersebut menjadi objek *File*Connection.

Setelah koneksi terbuka, kita bisa menampilkan list *file* pada direktori saat ini (currDirName). Tetapi untuk melakukannya, kita membutuhkan bantuan sebuah objek Enumerasi. Objek ini berfungsi untuk menampung list *file* yang ada pada *File*Connection tersebut. Selain itu, kita juga membutuhkan sebuah objek List. Fungsi dari list ini sendiri adalah untuk menampung tiap-tiap list *file* untuk ditampilkan ke layar. Tetapi mengapa kita tidak bisa langsung menampilkan list *file* langsung dari *File*Connection? Hal ini tidak memungkinkan karena pada J2ME, library *File*Connection itu tidak berhubungan dengan object Screen (sebuah object pada J2ME untuk menampilkan sesuatu ke layar). Enumerasi juga tidak berhubungan dengan Screen. Tetapi yang kita inginkan adalah sebuah List *file*, maka dari itu kita menggunakan class List untuk menampung list *file* dan menampilkannya kelayar. Prosesnya bisa dilihat pada sintaks dibawah ini:

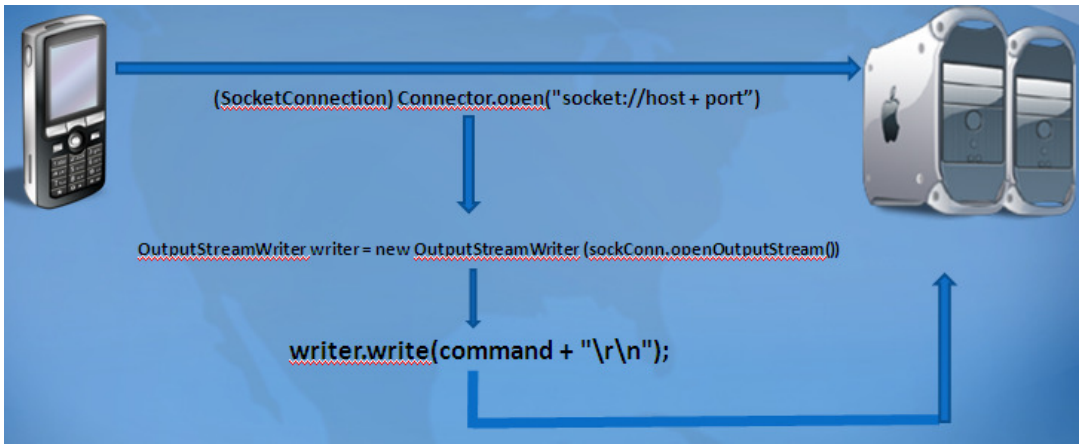
Pertama kita membuat sebuah objek enumerasi dan objek list. Kemudian dari objek *File*Connection sebelumnya yang telah dibuka, kita daftar list *file*nya menggunakan method *list("*", true).* Method tersebut memiliki 2 parameter, parameter pertama menandakan kita mengambil semua *file* pada direktori tersebut, parameter kedua merupakan *Boolean*, apabila *true* maka *file* yang bersifat *hidden* juga akan ikut masuk kedalam *list* dan sebaliknya.

5. Mengirim Command FTP melalui SocketConnection

Metode untuk melakukan *remote* atau akses terhadap *FTP Server* yaitu dengan mengirimkan *command-command* yang dimengerti oleh *FTP server* tersebut. Untuk jenis-jenis *command-command* yang bisa dimengerti oleh *FTP Server* bisa dilihat seperti website http://en.kioskea.net/internet/FTP.php3. Sehingga dapat dikembangkan method-method untuk mengirimkan *command-command* tersebut dalam bahasa pemograman java di J2ME.

Untuk mengakses *FTP Server*, kita harus menggunakan *SocketConnection* dengan melakukan koneksi terhadap port 21, karena *FTP* berjalan di port 21. Setelah membuka *SocketConnection*, untuk membaca dan menulis data melalui *SocketConnection* yang telah dibuka tersebut, kita membutuhkan *InputStreamReader* dan *OutputStreamWriter*. Sesuai dengan namanya, *InputStreamReader* berfungsi untuk menerima paket-paket data yang diterima, sedangkan *OutputStreamWriter* berfungsi untuk mengirimkan *command-command* ke *FTP server*. [MAH04]

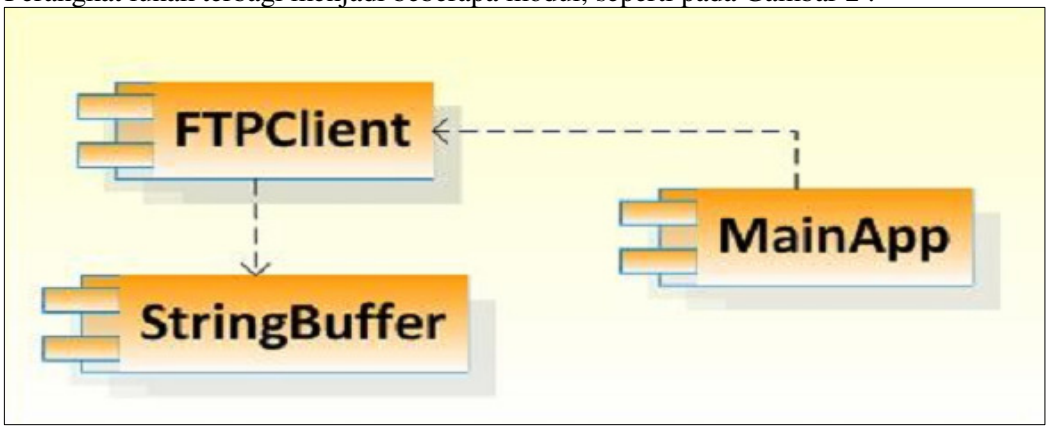
Respon yang diberikan oleh *server* itu berupa paket-paket data (*byte*), untuk itu diperlukan dilakukan proses perubahan menggunakan Modul *String Byte Buffer* untuk mengubah paket-paket data tersebut menjadi *String*.



Gambar 1 Mekanisme Pengiriman Command FTP menggunakan SocketConnection

6. Modul

Perangkat lunak terbagi menjadi beberapa modul, seperti pada Gambar 2 :



Gambar 2 Pembagian modul perangkat lunak

- Modul *Main Application*
  Modul ini merupakan implementasi dari antarmuka pengguna aplikasi. *Form* utama yang terdapat dalam modul ini berupa serangkaian *List* dari *file-file* yang berada di *Localhost* ataupun di *FTP Server* yang telah di-*remote*. Selain itu terdapat beberapa *form* lain yaitu :
  1. *Form Properties*
  Form ini menampilkan *properties* dari sebuah *file* yang terpilih. Berisi informasi-informasi seperti nama *file*, ukuran *file*, tanggal *file* dan hak akses (*Hidden* atau *Read Only*)
  2. *Form* Minta Input
  Form ini menampilkan sebuah *textfield* untuk diisi oleh pengguna sebagai nama baru untuk *file* atau folder

- Modul *FTP Client*

Modul *FTP Client* adalah modul yang berfungsi sebagai perantara untuk melakukan *command-command* untuk *FTP Server*. Dalam modul ini *String* perintah atau *command* yang dikirim oleh pengguna melalui *method-method* diubah terlebih dahulu menjadi paket *stream* dan kemudian dengan menggunakan *SocketConnection* akan mengirim paket-paket tersebut yang kemudian akan dibaca oleh *server*. Modul ini juga berfungsi untuk membaca respon yang dikembalikan oleh *server*. Contoh sintak-sintak yang ada dalam modul ini:

1. Membuka *Socket Connection*

Sintak diatas memerintahkan Connector untuk membuka socket connection dengan host sebagai alamat *FTP* yang dituju, dan port untuk nomor port yang digunakan untuk melakukan koneksi.

2. Membuka *Input* dan *Ouput Stream*

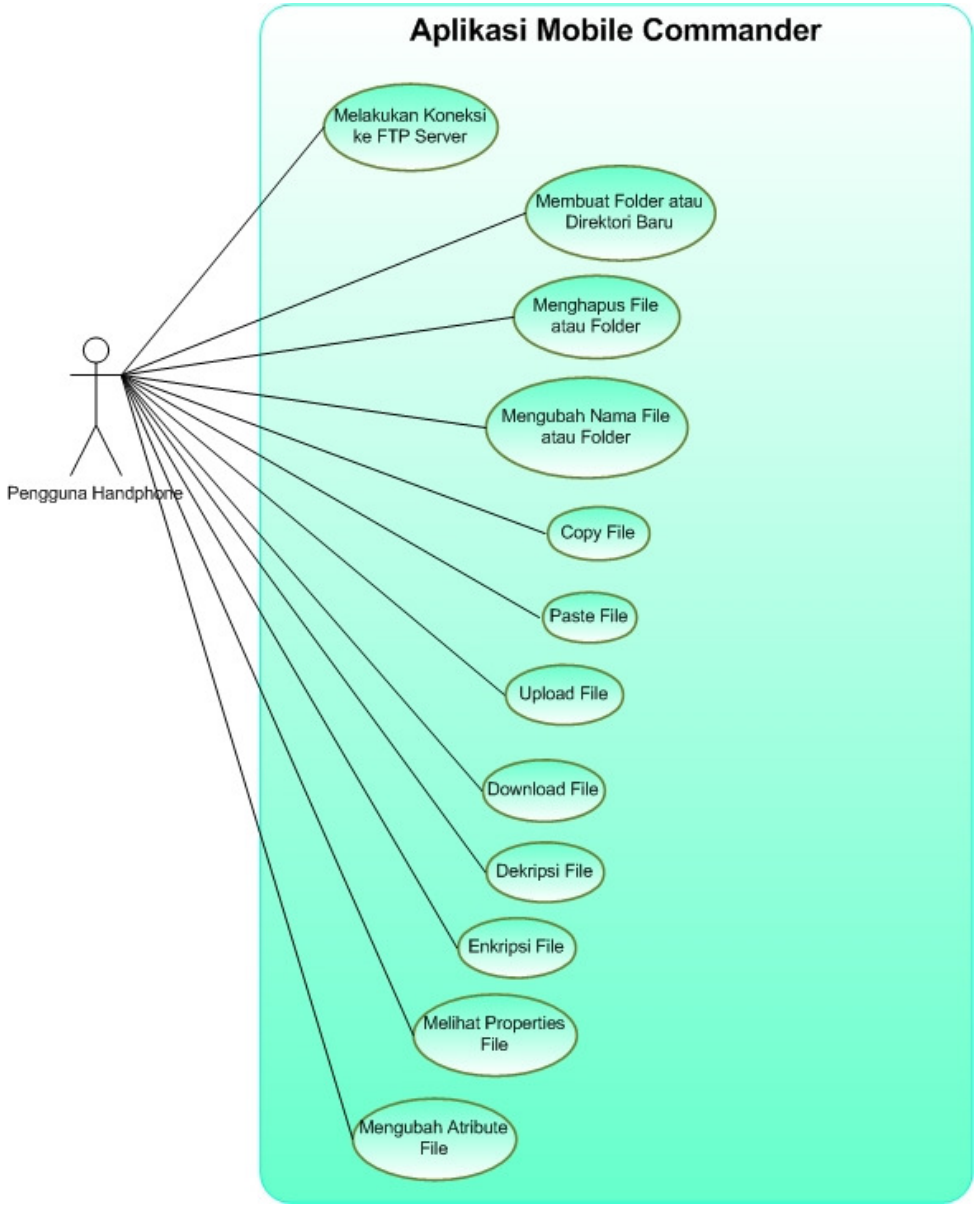
Sintak diatas memerintahkan untuk membuka *Input Stream Reader* dan *Output Stream Reader* dari *SocketConnection* yang telah dibuka sebelumnya.

- Modul *String ByteBuffer (Util)*
  Modul ini berfungsi untuk membuka *stream* yang diterima hasil dari perintah-perintah *FTP Server* dan mengubahnya menjadi *byte* yang kemudian untuk diproses lagi menjadi sebuah *file*

7. Desain Sistem

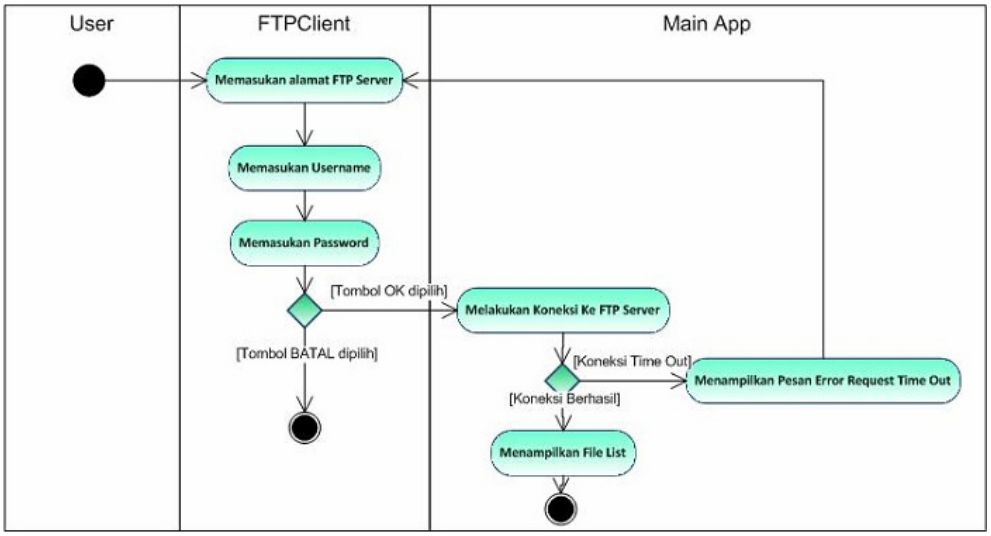
Pembuatan Aplikasi MobileCommander ini memerlukan adanya perancangan proses. Perancangan proses dalam pembuatan aplikasi ini menggunakan *UML (Unified Modeling Language)*

Pengguna aplikasi ini adalah pengguna handphone itu sendiri dimana pengguna handphone bisa melakukan proses-proses seperti membuat direktori baru, mengubah nama direktori atau *file*, menghapus direktori atau *file*, meng-*copy* dan *paste file*, *upload* dan *download file*, proteksi file, melihat *property file,* serta melakukan koneksi ke FTP Server (Lihat gambar 2).



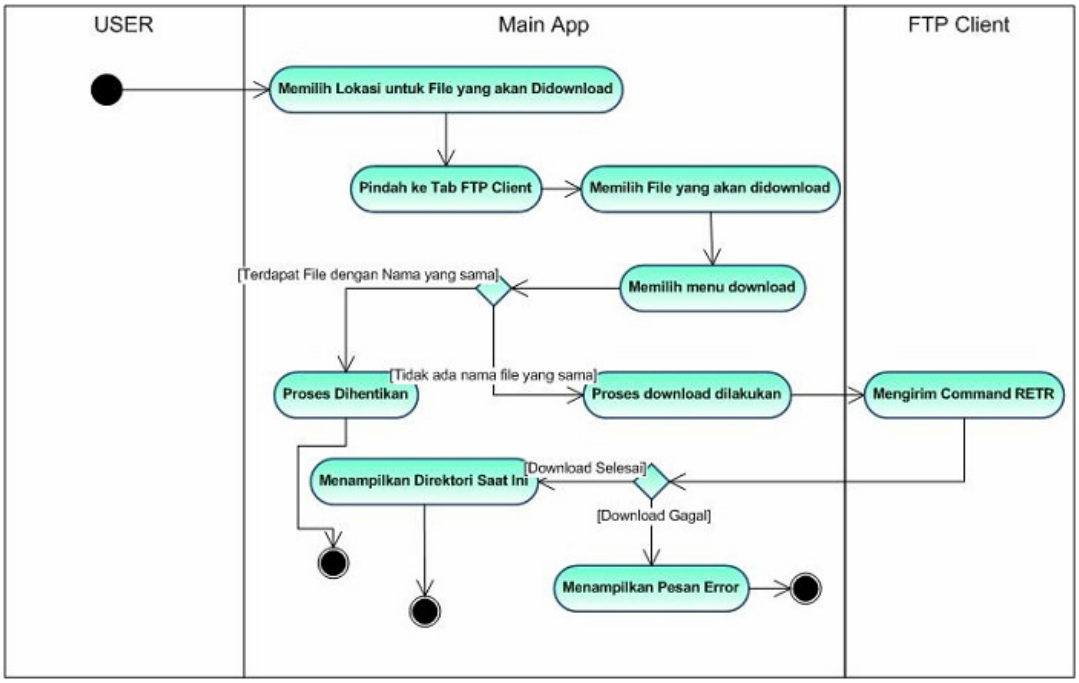
Gambar 2 ***Use Case Diagram***

Gambar 3 menunjukan *Activity diagram* untuk *use case* melakukan koneksi ke FTP Server. Pada saat aplikasi dijalankan, pengguna memilih untuk melakukan koneksi ke FTP Server. Kemudian pengguna memasukan alamat FTP Server yang ingin di remote, username serta password untuk login ke FTP Server tersebut. Ketika tombol OK dipilih, program akan melakukan akses ke FTP Server berdasarkan alamat yang dituliskan oleh pengguna. Bila aplikasi berhasil melakukan koneksi dengan baik, maka aplikasi akan menampilkan list file yang ada di FTP Server tersebut, dan bila tidak (Time Out), maka sistem akan menghentikan prosesnya dan kembali ke Form Input.



Gambar 3 *Activity diagram* melakukan koneksi ke FTP Server

Gambar 4 menunjukan diagram activity dari use case download file. Pada dasarnya proses download file ini sama seperti proses copy dan paste file. Tetapi hanya berbeda lokasi file yang di-copy dan tujuan tempat file tersebut akan di-paste. Jadi proses download ini merupakan proses copy file dari FTP Server dan paste file tersebut pada localhost. Pengguna terlebih dahulu menentukan lokasi untuk file yang akan di download pada Localhost, dan kemudian pengguna pindah ke Tab FTP Client. Pengguna kemudian memilih file yang akan di-download dan menekan menu download. Apabila tidak terdapat file dengan nama yang sama pada localhost, maka proses download akan dilakukan, tetapi apabila terdapat file dengan nama yang sama, maka proses akan dihentikan.

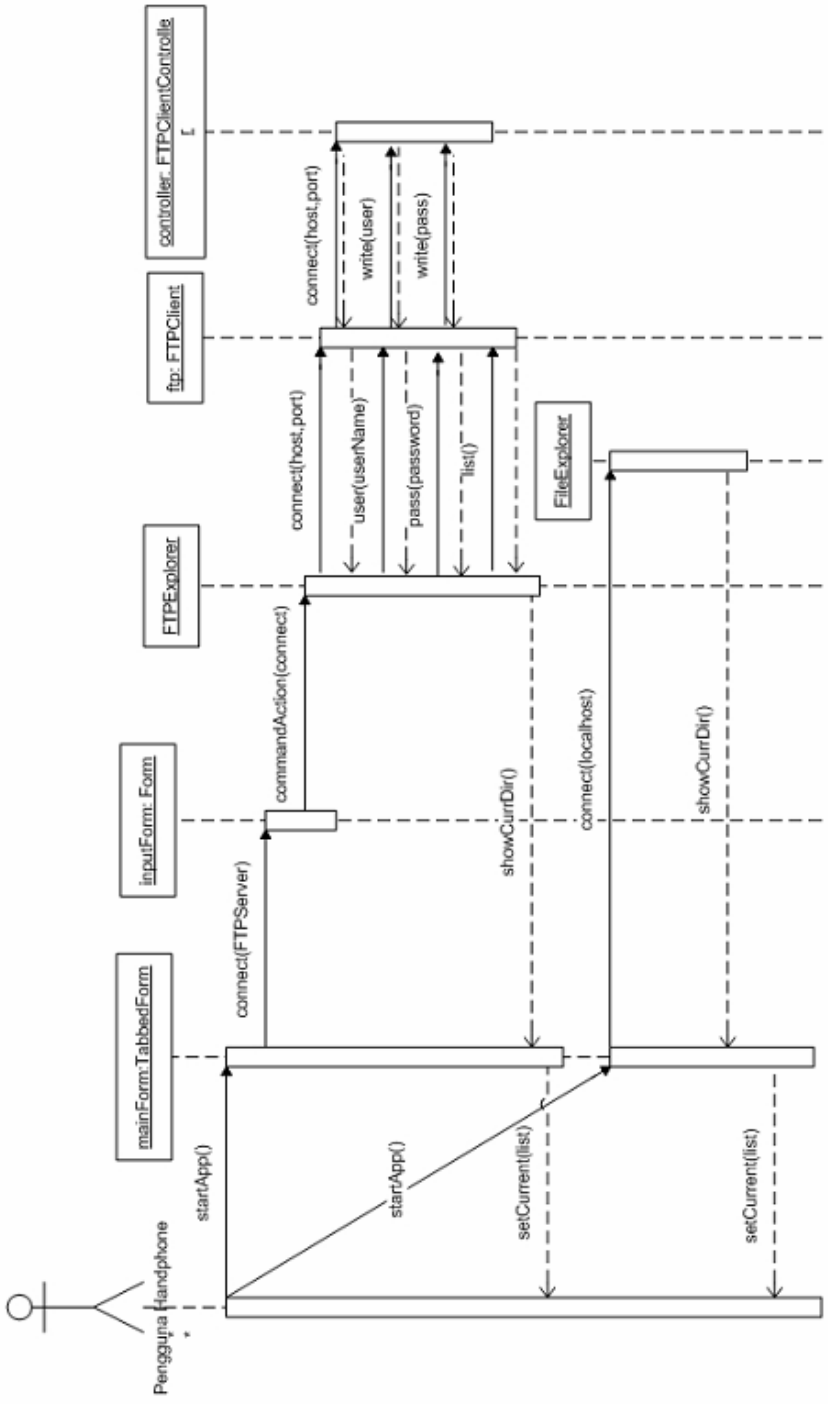


Gambar 4 *Activity diagram* download file dari FTP Server

Bila proses download selesai, system akan menampilkan direktori saat ini pada localhost, tetapi apabila download gagal, system akan menampilkan pesan error.

Sequence diagram untuk proses koneksi ke FTP Server atau localhost dapat dilihat pada Gambar 5, penjelasan prosesnya sebagai berikut:

- Ketika pengguna menjalankan aplikasi ini, secara otomatis pengguna menjalankan Method startApp().
- Pengguna bisa memilih untuk melakukan 2 koneksi, yang pertama koneksi ke localhost dengan menjalankan Method connect(localhost) atau melakukan koneksi ke FTP Server dengan Method connect(FTPServer).
- Ketika pengguna menjalankan Method connect(FTPServer), pengguna akan diminta untuk mengisikan beberapa kriteria untuk mengakses FTP Server. Kemudian sistem akan menjalankan commandAction (connect).
- Sistem akan membuat objek ftp dari class FTPClient, sistem akan melakukan koneksi melalui objek ftp.
- Setelah koneksi berhasi dibuat , maka sistem akan menampilkan list kepada pengguna.
- Ketika pengguna memilih localhost, sistem akan langsung menampilkan list file pada handphone kepada pengguna



Gambar 5 *Sequence diagram* koneksi ke FTP Server atau localhost

8. Implementasi

Aplikasi yang telah dirancang kemudian diimplementasikan dengan menggunakan J2ME (*Java 2 Micro Edition*). Beberapa detail implementasi dijelaskan dibawah ini:

- Membuat Direktori Baru
  Fungsi ini untuk membuat sebuah direktori baru didalam handphone *user* ataupun pada *FTP Server*. Pengguna diminta untuk meng-input sebuah *String* untuk dijadikan sebagai nama dari direktori tersebut
- Menghapus *File* atau Direktori
  Fungsi ini untuk menghapus sebuah *file* atau direktori. Apabila direktori tersebut di dalamnya sebuah terdapat direktori lain atau *file* lain, maka sistem akan menghapusnya secara rekursif
- Mengganti nama *file* atau direktori
  Berfungsi untuk mengubah nama dari sebuah *file* atau direktori. Pengguna diminta untuk meng-input sebuah *String* untuk dijadikan sebagai nama baru untuk *file* atau direktori tersebut
- Melihat properti *file*
  Fungsi ini membantu pengguna untuk melihat informasi *file* seperti nama *file*, waktu dan tanggal *file* tersebut terakhir kali dimodifikasi, besarnya ukuran *file* terseut (dalam Kb). Pengguna juga bisa mengubah atribut *file* (*Read – only* dan *Hidden*) pada *file-file* yang berada pada *localhost*.

- *Copy File*
  Berfungsi untuk meng-*copy* sebuah *file* dengan tujuan untuk menggandakan *file*. Sistem akan menyimpan nama *file* dan objek dari koneksi *file* akan tersimpan didalam sebuah variabel yang bersifat sementara (temporary)
- *Paste File*
  Berfungsi untuk menduplikasikan sebuah *file*. *File* menjadi 2 dengan sifat – sifat yang sama dengan *file* yang sebelumnya. Pada aplikasi ini terdapat 4 proses *Copy Paste* yang berbeda, yaitu Localhost – localhost, localhost – *server* (upload), *server* – localhost (download), serta *server – server.*
- Enkripsi *file*
  Berfungsi untuk mengamankan sebuah *file* pada *localhost*, sehingga *file* tersebut tidak bisa dibuka, dan informasi dari *file* tersebut tidak terbaca. Informasi *file* tidak bisa dilihat dan *file* tidak bisa dibuka oleh pengguna
- Dekripsi *file*
  Membuat *file* pada *localhost* yang sudah diamankan agar bisa terbaca lagi. Informasi *file* bisa dilihat kembali dan *file* bisa dibuka.

Form utama pada aplikasi ini dibagi menjadi 2 bagian yaitu Localhost dan *FTP Client.*

- Localhost

Gambar 6 *List File* pada *Localhost*

- *FTP Client*

Gambar 7 *List File* pada *FTP Client*

Gambar 6 dan gambar 7 merupakan realisasi dari rancangan antarmuka untuk *Main Form* atau *Current Directory*. *Form* ini adalah tampilan utama yang muncul ketika pengguna melakukan koneksi ke *localhost* atau *FTP Server*. *Form* ini terdiri dari:

- *List* dari *file-file* yang berada pada direktori tertentu
- Menu-menu manajemen *file*, dekripsi *file* (pada *localhost*), download (pada *FTP Client*) serta upload *file* (pada *localhost*).

Pada saat kita memilih menu properties, maka kita akan melihat informasi dari *file* yang kita pilih. Terdapat 2 perbedaan yaitu pada tampilan property *file* pada *file-file* yang berada pada localhost, kita dapat mengganti atribut *file* tersebut, tetapi pada FTP Client, kita hanya bisa melihat informasi *file* tersebut saja. Gambar 8 dan gambar 9 merupakan antarmuka pengguna untuk *Properties Form*. *Form* ini muncul ketika pengguna menekan tombol *Properties* pada *Main Form* atau *Current Directory* pada *Localhost* atau *FTP Client*.

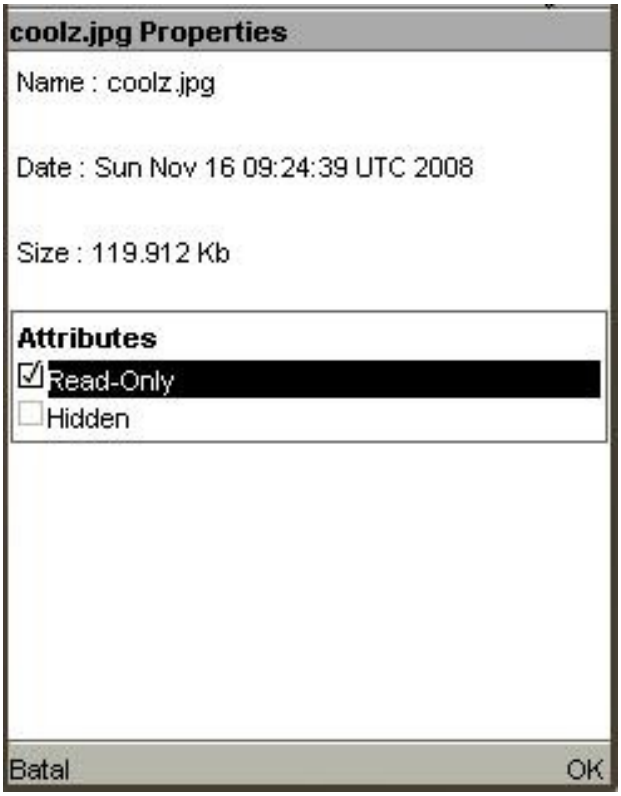
Form Properties ini terdiri atas:

1. Beberapa *Textfield* yang berisikan informasi tentang *file* tersebut.
2. *ChoiceGroup* untuk menampung atribut (*Hidden*, *Read only*) pada *localhost*

3. *Command OK* untuk mengubah nilai atribut bila ada perubahan pada *localhost*
4. *Command* Kembali untuk kembali ke *Current Directory*

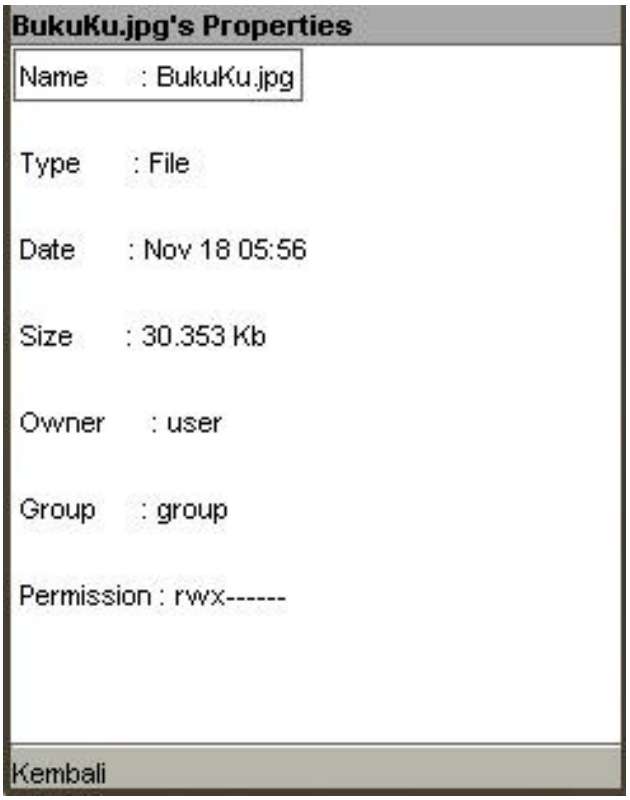
Berikut adalah tampilan dari form properties dari Localhost dan FTP Client:

- Localhost



Gambar 8 *Properties File* pada *Localhost*

- *FTP Client*



Gambar 9 *Properties File* pada *FTP Client*

9. Kesimpulan

Kesimpulan yang dapat ditarik dari hasil evaluasi yaitu secara umum aplikasi ini menghasilkan nilai guna yang cukup tinggi, dimana aplikasi ini memberikan solusi pada masalah pengaksesan terhadap FTP Server yang lebih fleksibel dengan menggunakan handphone. Beberapa hal yang ditawarkan pada aplikasi ini yaitu kemudahan dalam *management file* seperti membuat direktori baru, menghapus *file* atau direktori, mengganti nama *file* atau direktori, meng-*copy file*, dan *paste file*. Selain itu aplikasi ini juga dapat mengamankan *file* pengguna handphone dengan fitur enkripsi dan dekripsi, serta pengguna handphone dapat melihat *property* dari *file* yang ada di handphone-nya.

Aplikasi ini juga membantu dalam mengakses sebuah FTP Server, sehingga memudahkan para pengguna handphone untuk melakukan perubahan (update) terhadap FTP Server kapan saja dan dimana saja selama handphone tersebut masih terhubung dengan jaringan GPRS.

10. Saran

Beberapa hal yang dapat dilakukan supaya aplikasi ini lebih berdaya-guna lagi:

1. Pengembangan aplikasi dengan tampilan yang lebih menarik lagi dan mudah dimengerti oleh user.
2. Penambahan *help* dan *user* manual untuk aplikasi.
3. Aplikasi lebih fleksibel dan bisa berjalan pada *device* yang berbeda-beda.
4. Dibuat supaya dapat mengakses lebih dari satu FTP Server pada suatu saat.

Daftar Pustaka

[AMI07] Amity. (2007, May). SocketConnection in CLDC. J2ME Forums. Retrieved June 26, 2008, from http://www.j2meforums.com/forum/index.php?topic=15840.0

[KEO03] Keogh, James. (2003). *Socket Connection*. Retrieved July 11, 2008, from http://www.java2s.com/Code/Java/J2ME/Socketconnection. htm

[KNU05] Knudsen, Jonathan. (2005). *Beginning J2ME – Form Novice to Professional third edition.* New York: Apress

[MA04] Ma. Frank. (2004, August). SocketConnection *Problems.* J2ME Forums. Retrieved June 26, 2008, from http://www.j2meforums.com/forum/index.php?topic=2023.0

[MAH03] Mahmoud. Qusay. (2003, April). *J2ME Low-Level Network Programming with MIDP 2.0.* Retrieved July 12, 2008, from http://developers.sun.com/mobility/midp/articles/midp2network/

[MAH04] Mahmoud, Qusay. (2004) . *Getting Started with the File Connection APIs.* Retrieved January 22, 2008, from http://developers.sun.com/ mobility/apis/articles/fileconnection/index.html