Sinkronisasi File Pada Google Drive Berdasarkan Perbandingan File Modification Date

Kevin Darmawan Limantoro[1],Justinus Andjarwirawan[2],Agustinus Noertjahyana[3]
Program Studi Teknik Informatika,Fakultas Teknologi Industri, Universitas Kristen Petra
Jl. Siwalankerto 121-131, Surabaya 60236
Telp (031) – 2983455, Fax. (031) - 8417658
vinlimantoro@gmail.com,justin@petra.ac.id,agust@petra.ac.id

Abstrak

Online storage such Google Drive is created to answer current lifestyle of people all over the world. Nowadays, most people use more than one device, especially for those who lived on advance country. Using online storage, not only their data is safe but also can be accessed anytime online. The thing is most of these people want to have the latest version from a file in all their device. To always keep and update the latest version of file, these people can use synchronization.

Because of that, this application is able to do two way file synchronization which is from android device to Google Drive and also otherwise from Google Drive into the android device is made on this final assignment. The method I will use is date comparation, so file with latest modification date will be duplicated to replace the older file. With this method both storage will always have the latest updated file. All synchronization and other processes like compare, upload, download, share, and update will be use based on function Google Drive API. Application will be made using Java version of Eclipse IDE combined with Android Development Tool, because this application will run on android platform devices.

Kategori dan Deskripsi Subjek

D.3.3 [Java]: Language Constructs and Features – Abstract Data Type, Classes and Object, Data Type and Structures.

Istilah Umum

Algoritma, Eksperimen.

Kata Kunci

Android, Google Drive, Sinkronisasi File, Google API, Drive SDK, Java.

1. PENDAHULUAN

Google Drive merupakan cloud storage buatan Google yang dibuat untuk menjawab permasalahan kebanyakan orang seperti kemanan, rawan rusaknya media penyimpanan seperti CD atau flashdisk. Pengguna tidak ditarik biaya sama sekali untuk bisa menggunakan Google Drive ini. Dengan Google Drive, user dipermudah ketika akan melakukan transfer dokumen menuju PC dan juga untuk sebaliknya yaitu dari PC menuju Android.

Namun, hingga saat ini di Android belum terdapat aplikasi yang dapat melakukan sinkronisasi file secara otomatis, sehingga untuk memperbarui file user harus melakukan upload atau download file secara manual. Hal ini tentunya merugikan karena bisa menjadi sangat time-consuming. Dengan sinkronisasi ada begitu banyak keuntungan yang bisa didapat, salah satunya adalah risiko kehilangan dan kerusakan file dapat dikurangi[2].

2. STRUKTUR FILE

Metadata merupakan sebuah data yang memberikan informasi tentang aspek-aspek dari sebuah data atau file. Aspek tersebut diantaranya:

- Tanggal pembuatan file.
- Tanggal modifikasi terakhir dari file.
- Creator atau author dari file.
- Ukuran dari file.

Sebagai contoh sebuah file gambar akan memiliki metadata mengenaik ukuran dari gambar, kamera yang digunakan untuk mengambil gambar, panjang focus dari kamera, dll. Dengan membaca metadata dari file informasi lengkap dari file akan dapat diketahui. Teori sinkronisasi yang akan diterapkan adalah berdasarkan perbandingan waktu modifikasi terakhir. Oleh sebab itu metadata dari yang akan digunakan dalam proses hanyalah data modifikasi terakhir. Dengan menggunakan java.io.file dan juga query dari API Google Drive tanggal modifikasi dari masing-masing file bisa didapatkan.

2.1. Pengambilan Tanggal Modifikasi terakhir File Lokal

Untuk melakukan sinkronisasi digunakan data tanggal modifikasi terakhir dari metadata file. Tanggal modifikasi terakhir merupakan data yang berisi informasi kapan terakhir kali file telah dimodifikasi. Setiap file dapat dipastikan memiliki metadata ini. Tanggal modifikasi inilah yang akan digunakan melakukan perbandingan Untuk bisa mendapatkan tanggal modifikasi terakhir, maka file harus dijadikan objek yang baru. Dari objek file ini akan dipanggil fungsi lastModified(). Yang akan me-return tanggal modifikasi terakhir dari file dalam wujud long. Untuk dapat melakukan komparasi maka tanggal dalam wujud long ini dapat diubah

menuju format RFC3339. Konversi dalam format RFC3339 dapat dilakukan dengan fungsi toStringRFC3339() yang terdapat dalam Java Android. Tanggal modifikasi terakhir dari file ini akan secara otomatis terupdate ketika file mengalami modifikasi atau mengalami edit. Tanggal modifikasi ini juga akan berubah ketika file ditumpuki ketika proses download. Tanggal modifikasi akan sama dengan tanggal pembuatan ketika file pertama kali dibuat.

2.2. Pengambilan Tanggal Modifikasi terakhir File Google Drive

Metadata ini juga dapat ditemui pada file Google Drive. File Google Drive memiliki data yang jauh lebih banyak daripada file lokal. File metadata ini digunakan untuk menyimpan data share dari file terhadap user lain, selain itu metadata ini juga berisi data link menuju file. Untuk melakukan sinkronisasi berdasarkan perbandingan waktu diperlukan data dari fungsi lastModified(). Hasil return dari fungsi ini adalah waktu modifikasi terakhir dari file, dalam format yang berbeda dari file lokal. Tanggal modifikasi terakhir dari Google Drive merupakan DateTime yang diformat dalam wujud RFC3339. Format ini merupakan standard untuk dateTime dari semua file pada Google Drive. Berikut adalah contoh tanggal yang ditulis dalam format RFC3339:

2012-12-31T23:59:59.023Z

Tanggal modifikasi dicatat pertama kali ketika file diupload menuju Google drive danakan secara otomatis terupdate ketika file dimodifikasi atau diedit dalam Google Drive. Tanggal modifikasi ini juga akan berubah otomatis ketika terjadi update konten file.

3. ALGORITMA SINKRONISASI FILE

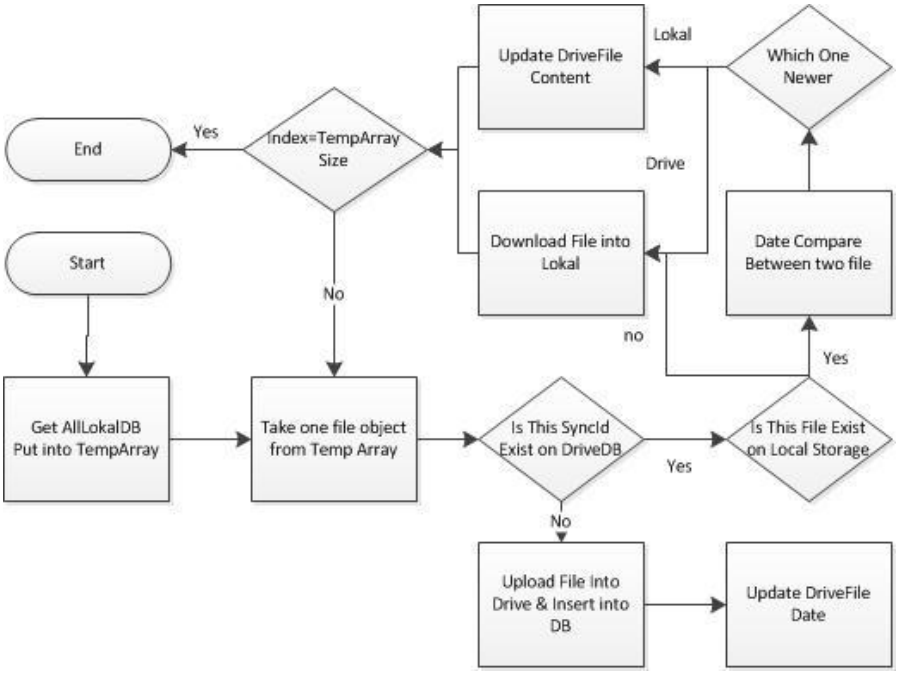
Pada bagian awal algoritma sinkronisasi dilakukan pembacaan database untuk mengetahui file apa saja yang harus di sinkronisasi. Penyediaan database ini sangat diperlukan karena memiliki fungsi untuk membedakan file yang akan atau tidak disinkronisasi. Terdapat dua table dalam database yaitu tabel FileList yang digunakan untuk menyimpan file dalam device yang akan di sinkronisasi menuju Google Drive, file disimpan dalam wujud string path menuju file. Alasan dari penggunaan string path ini adalah kemudahan yang didapat ketika akan membuat objek file. Selain table lokal juga terdapat tabel DriveFileList yang berfungsi untuk menyimpan file google drive yang akan disinkronisasi menuju device. File google drive disimpan dalam wujud file id Google Drive. Dalam Google drive, file dan folder dikenali menggunakan kombinasi karakter yang disebut sebagai file id. Namun, untuk memudahkan user file direpresentasikan kepada user menggunakan nama dari file. Dalam proses sinkronisasi dilakukan proses komparasi waktu modifikasi terakhir dari dua file. Disinilah kedua table yaitu tabel FileList dan file google drive menjalankan tugasnya. Setiap file baik lokal maupun Google Drive di identifikasi menggunakan ListId oleh aplikasi. ListId ini menjadi primary key dari kedua table. Sebuah file dapat mencari file pasangan sinkronisasinya berdasarkan ListId dari tiap-tiap file. Bila dua file pada tiap-tiap table memiliki ListId yang sama maka kedua file tersebut adalah pasangan file sinkronisasi. Pasangan file sinkronisasi inilah yang nantinya akan dibandingkan mana yang lebih baru[3][5]. Database akan dibaca 2 kali, seperti dapat dilihat pada gambar 1 yaitu flowchart proses sinkronisasi. Pertama dibaca tabel FileList, kedua akan dibaca tabel DriveFileList.

FileList		DriveFileList	
PK	ListID	PK	ListID
FK1	UserId FilePath Sync	FK1	UserId FileId Sync

Gambar 1 Struktur Database penyimpan file

Pembacaan database sebanyak 2 kali ini dimaksudkan agar file yang masih belum memiliki pasangan dapat diupload menuju google drive atau di download menuju device untuk pertama kalinya. Setelah dilakukan upload atau download ini maka akan dilakukan input menuju table dengan ListId yang dimiliki file source. Dengan begitu maka akan tercipta sebuah pasangan file.

Bila masing-masing dari kedua file telah terdapat dalam table maka proses sinkronisasi dapat dilakukan dengan melakukan perbandingan tanggal modifikasi terakhir dan bukan upload atau download lagi. Database yang digunakan untuk proses sinkronisasi ini adalah sqlite yang memang sudah terdapat dalam OS Android, sehingga tak perlu menambahkan ekstensi tambahan.setelah dilakukan pembacaan file pada database lokal, maka semua file yang terdapat dalam database akan ditampung dalam sebuah array penampung sementara. Dari array ini selanjutnya file akan diambil satu persatu. Setiap file lokal akan dicek apakah telah memiliki pasangan pada table Google Drive. Seperti telah dibahas sebelumnya pasangan file dikenali dengan memiliki ListId yang sama. Bila ternyata file belum memiliki pasangan pada sisi Google Drive maka file akan diupload dan setelah diupload maka akan di inputkan menuju tabel DriveFileList. Setelah input maka file yang barusan diupload akan disamakan tanggal modifikasi terakhirnya dengan file yang terdapat pada lokal atau sumber upload. Tujuan dari penyamaan tanggal sinnkronisasi ini adalah agar file yang baru diupload tidak dianggap lebih baru dari file yang berada pada device.



Gambar 2 Flowchart proses sinkronisasi.

Hal ini dikarenakan tanggal modifikasi terakhir dari file Google Drive dipengaruhi oleh waktu ketika file diupload. Sehingga, bila tanggal tidak disamakan maka file yang baru saja terupload akan dianggap lebih baru dan lalu di download ulang. Setelah pengecekan pasangan file sinkronisasi maka proses selanjutnya adalah pengecekan apakah file masih terdapat dalam lokal memori karena menggunakan database maka file yang dihapus tidak akan hilang begitu saja dari tabel FileList, dibutuhkan pengecekan apakah file telah dihapus. Sehingga bila ada file lokal yang telah terhapus maka file tersebut akan dihapus dari tabel FileList. Karena pasangan file Google Drive yang terdapat dalam table lokal telah tidak ada maka file akan di download dan di input langsung menuju tabel FileList, untuk menciptakan sebuah pasangan file. Sedangkan bila kedua file yaitu file lokal dan google drive exist, maka proses selanjutnya adalah perbandingan file mana yang lebih baru bila file yang lebih baru adalah file lokal maka file lokal akan diupload untuk mengupdate konten dari file google drive. Sedangkan sebaliknya bila file google drive lebih baru maka file tersebut akan didownload menuju lokal storage, dan secara ototmatis akan menumpuki file lokal yang lebih lama. Proses ini akan diulangi terus hingga semua file yang terdapat dalam array penampung sementara selesai di cek.

4. IMPLEMENTASI

Implementasi dari aplikasi yang dijelaskan pada bagian ini hanyalah pada bagian sinkronisasinya. Pertama sesuai dengan flowchart yang terdapat pada gambar 2 adalah dilakukan pembacaan semua file lokal yang harus disinkronisasi. Pembacaan dilakukan pada table file list

Pseudocode 1. Fungsi pembacaan file lokal yang harus di sinkronisasi

```
Return GET ALL FROM TABLE FileList THAT userId IS EQUAL currentUser AND Synchronize.
```

Dari hasil pseudocode 1 maka akan dihasilkan sebuah list data dari table FileList. Tiap-tiap data dari list memiliki komponen-komponen yaitu:

- ListId : Merupakan komponen utama dan juga primary key dari FileList, listId ini digunakan untuk mengidentifikasi pasangan dari file lokal yang terdapat dalam table DriveFileList. Bila terdapat file dengan ListId yang sama pada DriveFileList maka file itulah yang menjadi pasangan file lokal.
- Username : Merupakan atribut yang menjelaskan kepemilikan dari file yang akan disinkronisasi. Adanya atribut ini dikarenakan aplikasi mendukung pergantian account yang akan digunakan untuk mengakses Google Drive.
- Filepath : Atribut ini berisi string path dari file. String path dapat dengan mudah dijadikan objek file, karena objek file dibuat salah satunya dengan menggunakan string path.
- Sync(Bool) : Sync merupakan satu-satunya atribut yang berwujud Boolean. Sync digunakan untuk mengidentifikasi apakah sebuah file harus disinkronisasi atau tidak.

Dari semua file lokal yang didapat dari database akan dilakukan proses selanjutnya. Satu buah data dari table ini disimpan dalam sebuah class yang disebut CLocalFile, CLocalFile terdiri dari ListId,UserName,FilePath, dan SyncBool dan memiliki fungsi set dan get untuk tiap-tiap variable. Pada bagian selanjutnya tiap-tiap objek CLocalFile

akan dicek. Pengecekan pertama akan dilakukan dengan melakukan cek terhadap pasangan file yang terdapat dalam table DriveFileList(). Bila file pasangan tersedia pada table dan juga file tersebut memiliki nilai true pada atribut Sync, maka akan dilakukan proses pembandingan file. Proses pembandingan file dilakukan dalam proses SynchronizeFile() yang dieksekusi dengan mengirimkan objek CLocalFile sebagai parameter. Proses synchronizeFile() akan dibahas pada bagian selanjutnya. Dalam fungsi yang terdapat pada Pseudocode 2 juga dilakukan fungsi upload yang dipanggil ketika file tidak tersedia pada table DriveFileList, dan ketika file yang dibaca bukanlah folder.

Pseudocode 2. Proses pengecekan file.

```
int i = 0;

while (i <total CLocalFile) {

if(CLocalFile IS Exist on DriveFileList
and Have True on SyncBool Value)
{
SynchronizeThisFile()
}
Elseif (CLocalFile IS NotExist on
DriveFileList and Not A Directory(must be
File))
{
UploadThisFile();
}
i++;
}
```

Seperti dapat dilihat pada Pseudocode 2, file yang belum terdapat pada table DriveFileList akan diupload dengan menggunakan fungsi UploadThisFile(). Dalam proses upload ini dilakukan beberapa proses diantaranya:

1. Upload file menuju Google Drive.
2. Mengupdate file last modification date, karena file modification date file Google Drive memiliki default yaitu tanggal upload.
3. Melakukan insert data file Google Drive menuju table DriveFileList.

Pseudocode3. Fungsi upload file.

```
Function UploadThisFile{

create null DriveFile for upload;

Get Mimetype,Title, and FileContent from
source file save to Temp;

Initialize temp on DriveFile

Make INSERT API Call from Drive SDK on
DriveFile;

CATCH FileId returned from upload process
put on CDriveFile;

UpdateDriveFile(FileId Catched);

Return CDriveFile;

}

Function onInsertFinish(CDriveFile){
insert into DriveFileList CDriveFile;
}
```

pada proses awal upload dilakukan inisialisasi terhadap file yang hendak diupload. Selain itu juga harus dibuat sebuah tipe file Google Drive(`com.google.api.services.drive.model.File`),file Google Drive ini akan digunakan untuk menginisialisasi body dari file, seperti nama file, mimetype, atau juga tanggal modifikasi bila akan mengupdate tanggal secara manual. Setelah itu dibuat sebuah objek fileContent, objek ini digunakan untuk menyimpan konten dari file sesuai dengan mimetypenya. Mimetype adalah tipe file, mimetype biasanya dapat diketahui dari ekstensi file(contoh : txt memiliki mimetype "text/plain"). Selanjutnya dilakukan query upload pada objek drive yaitu service. Untuk melakukan upload digunakan query `files().insert(fileBody,fileContent).execute()`, untuk file yang akan diconvert menjadi Google Document dapat digabungkan fungsi `.convert()`.Query ini akan mereturn metadata file yang telah berhasil diupload, metadata yang telah berhasil didapat nantinya akan direturn dan akan berjalan pada onInsertFinish(CDriveFile) pemanggilan API Upload ini dapat dipelajari lebih lanjut pada website official Google Drive. Pada onPostExecute cukup dilakukan input menuju database. Dalam fungsi upload terdapat proses updateDriveFile.Proses ini digunakan untuk mengupdate tanggal modifikasi dari file yang baru diupload. Dapat dilihat pada Pseudocode 4 fungsi update file. Fungsi update ini juga digunakan melakukan update konten file, sehingga dengan kata lain fungsi update ini juga digunakan ketika proses sinkronisasi, yaitu ketika file dari device lebih baru dan harus menggantikan file pada Google Drive.

Pseudocode 4. Fungsi update file.

```
FunctionUpdateDriveFile(FileId)
{
create null DriveFile for upload;
Get Mimetype,Title, lastModifiedDate, and
FileContent from source file save to
Temp;

Convert lastModifiedDate to String
RFC3339;

Initialize temp on DriveFile

Make UPDATE API Call from Drive SDK on
DriveFile based on FileID,
setSetModifiedDate as True, Disable
onView Changing LastModifiedDate;

}
```

Pada dasarnya proses yang dilakukan ketika update cukup mirip dengan yang dilakukan dengan proses upload. Harus dilakukan inisialisasi awal file Google Drive yaitu untuk nama file dan mimetype, kmeudian disediakan sebuah filekonten yang berasal dari file lokal dan terakhir adalah datetime untuk tanggal modifikasi terakhir yang dikehendaki untuk file Google Drive. Yang membedakan adalah query yang dipanggil yaitu update dengan parameter awal yaitu id dari file google drive yang hendak diupdate kontennya. Pada

proses insert atau upload tidak tersedia id file yang akandiupload. Query yang dipanggil juga lebih panjang karena selain `files().update(ID,Body,FileContent)`, juga terdapat `setSetModifiedDate(true)` dan `.setUpdateViewedDate(false)`. Fungsi dari keduanya adalah:

- `setSetModifiedDate(true)` : query ini digunakan untuk mengenable update tanggal modifikasi terakhir. Query ini hanya bisa dipanggil bersamaan dengan update(), tidak bisa dengan insert().
- `setUpdateViewedDate(False)` : karena sinkronisasi berdasarkan waktu, maka query ini harus dipanggil agar file yang hanya dilihat tidak dianggap lebih baru. Hanya file yang dimodifikasi saja yang dianggap lebih baru.

Seperti telah dibahas sebelumnya bahwa pembacaan database dilakukan 2 kali. Pertama untuk membaca semua file lokal yang harus disinkronisasi, lalu semua file tersebut akan diproses seperti terlihat pada Pseudocode 1-4. Dan kemudian baru dilanjutkan dengan membaca semua file Google Drive yang harus di sinkronisasi. Proses awal yang dilakukan mirip dengan pembacaan lokal adalah dengan memanggil select terhadap table DriveFileList, fungsi ini akan me-return semua id dari file Google Drive yang harus disinkronisasi. Setelah itu satu-persatu file akan dicek. Pengecekan yang dilakukan jauh lebih simple daripada ketika melakukan cek terhadap file lokal. Pengecekan hanya dilakukan untuk mengetahui apakah file Google Drive telah terdapat dalam device. Bila file belum terdapat maka akan dipanggil download. Proses yang hanya dilakukan hanya ini karena sinkronisasi dengan perbandingan waktu sebelumnya dilakukan pada bagian file lokal, sehingga agak sia-sia untuk melakukan proses itu 2 kali. Proses pengecekan dapat dilihat pada Pseudocodes 5 dibawah.

Pseudocode 5. Awal proses pengecekan file Google Drive.

```
Return GET ALL FROM TABLE FileList THAT
userId IS EQUAL currentUser AND
Synchronize -> CDriveFile
int i = 0;

while (i < total CDriveFile) {
if(CLocalFile IS NotExist on
DriveFileList and Not A Directory(must be
File))
{
DownloadThisFile(ListId,FileId);
}
i++;
}
```

Proses ini cukup mirip dengan pengecekan file lokal yang membedakan hanya proses ini lebih singkat, karena hanya mengecek untuk download saja tanpa perbandingan waktu. Untuk mendownlaod ada dua parameter yang dikirim, namun dua parameter ini disatukan dalam 1 array. Parameter pertama adalah fileId dari file Google Drive, dan parameter kedua adalah ListId yang harus diinput kedalam database.

Pseudocode 6. Proses Download terhadap file Google Drive.

```
Function DownloadThisFile(ListId,FileId){

Create DriveFile using GET API Call on
FileId;

If(DriveFile downloadURL not Empty){

Download using HttpGet with downloadURL
as parameter};

Else(DriveFile downloadURL empty){

Download using HttpGet with
ExportLink(Mimetype) as parameter}

}

Function onDownloadFinish(){

CreateFile with FileOutputStream;

Input into FileList with ListId;

}
```

Proses download diawali dengan menginisialisasi sebuah drivefile dari fileId yang dikirimkan. Kemudian dibuat sebuah mekanisme HttpGet dan request menggunakan akses token, dan juga download url dari file Google Drive. Untuk mendapatkan url file terdapat fungsi getDownloadURL() yang tentu saja mereturn URL menuju file. Response yang diterima dari HttpGet akan dikirimkan untuk proses createFile. Pada proses ini terjadi pengecekan apakah harus dilakukan input menuju database lokal atau tidak. Bila parameter yang diparse hanya satu array maka tidak diperlukan input database, sedangkan bila lebih dari satu berarti harus input menuju database dengan fungsi addLokal. Khusus untuk file google document tidak bisa menggunakan GetDownloadURL(), karena akan me-return null. Sebuah Google Document bisa dikonversi menjadi berbagai jenis file lain. Contoh diatas menunjukkan sebuah file google document yang dikonversi menjadi sebuah file text. Setelah filecontent berhasil didapat maka akan diparse menuju fungsi createFile, fungsi ini akan menciptakan sebuah file dengan menggunakan fungsi java Fileoutput Stream yang menulis byte demi byte filecontent[1]. Untuk Pseudocode terakhir akan dibahas untuk algoritma sinkronisasi, algoritma ini melakukan 3 proses yaitu:

1. melakukan perbandingan waktu antara file lokal dan juga file Google Drive.
   a. Bila file pada Google Drive lebih baru maka file tersebut akan di download untuk menggantikan file yang berada pada device.
   b. Bila file pada Lokal lebih baru, maka file tersebut akan di upload untuk menggantikan file yang berada pada Google Drive.

2. Melakukan pengecekan keberadaan existensi dari file untuk lokal dan Google Drive.
    a. Untuk file lokal yang sudah tidak ada atau sudah dihapus maka akan data yang terdapat dalam database FileList juga akan dihapus.
    b. Sedangkan bila file Google Drive yang sudah di trash maka file akan dihapus pula dari database DriveFileList
3. Melakukan upload dan download untuk file yang masih eksis disalah satu, yaitu lokal atau Google Drive.
    a. Bila hanya file pada Google Drive yang masih tersedia, maka file akan didownload dan diinput ulang ke database FileList.
    b. Bila hanya file pada device yang masih tersedia, maka file akan diupload dan diinput ulang ke database DriveFileList.

Pseudocode 8.Fungsi untuk melakukan perbandingan waktu.

```
Initialize CDriveFile;
Initialize CLocalFile;
If(CLocalFile is Newer than
CDriveFile){
UpdateDriveFile();}
Else if(CLocalFile is Older than
CDriveFile){
DownloadThisFile(FileId)}
Redownload or reupload if file only
exist in one side;
Delete completely from database if
both deleted;
}
```

Tanggal modifikasi terakhir ini berwujud string RFC3339 date standard. String RFC 3339 pada umumnya memiliki wujud seperti terlihat pada bab 2.2 pada jurnal ini. Namun, untuk memudahkan proses maka dilakukan pemotongan string yaitu pada bagian ".000Z" atau 5 character bagian terakhir dari RFC3339. Dilakukan pemotongan ini adalah karena pengecekan untuk mengetahui file mana yang lebih baru tidak perlu tingkat ketelitian hingga millisecond. Selanjutnya dua string waktu ini dikomparasi dengan fungsi komparasi yang dimiliki oleh string.Bila hasilnya adalah <0 maka string waktu yang memanggil fungsi komparasi lebih lama daripada waktu yang diparse. Sebaliknya bila menghasilkan >0 maka berarti string waktu yang memanggil fungsi komparasi lebih baru daripada waktu yang diparse. sedangkan untuk file yang sudah tidak berada lagi di dua tempat maka akan hilang dari database[4].

5. PENGEMBANGAN

Meskipun sinkronisasi dari aplikasi telah dapat berjalan dengan baik, masih da beberapa hal yang dapat dikembangkan untuk membuat aplikasi ini menjadi lebih baik:

5.1. Perbandingan MD5 Checksum

Satu hal lagi permasalahan yang ditemui pada aplikasi ini adalah adanya kecenderungan untukngoogle drive tiba-tiba google drive untuk melakukan auto-touch pada file yang barundiupload pada Google drive.Akibat dari jalannya sistem auto-touch ini adalah perubahan tanggal modifikasi terakhir yang menjadi terbarui. Seperti telah dibahas sebelumnya bahwa proses sinkronisasi dilakukan dengan mengkomparasi tanggal modifikasi terakhir dari file, bila tanggal modifikasi menjadi lebih baru padahal file sebenarnya sama, maka bisa terjadi download yang sia-sia dan bahkan merugikan bila sampai menumpuki file yang seharusnya diupload. Yang dapat dikembangkan dalam aplikasi ini adalah untuk algoritma sinkronisasi yang dilakukan, untuk mengatasinya dapatfitambahkan algoritma untuk menghasilkan MD5 checksum pada file lokal dan juga file Google Drive, setelah itu dilakukan komparasi atau perbandingan dari dua MD5 Checksum ini. Bila keduanya sama maka file berarti tidak megalami perubahan, bila berbeda maka salah satu file telah diedit.

6. Kesimpulan

Kesimpulannya adalah aplikasi telah berhasil mensinkronisasi setiap file termasuk dokumen selama dokumen tidak memiliki gambar. Sinkronisasi telah berhasil mengupdate konten file, sehingga file yang berada pada dua tempat ini akan selalu sama dan pada versi yang terbaru. Namun, masih ada banyak yang bisa dikembangkan untuk algoritma sinkronisasi, salah satunya adalah pengembangan untuk metode perbandingan file, karena seperti telah dibahas sebelumnya dimana Google Drive terkadang melakukan auto-touch file, sehingga tanggal modifikasi file mengalami perubahan meskipun tidak dibuka ataupun di edit. Hal ini terkadang bisa membaut kekacauan pada saat proses sinkronisasi karena terkadang file yang harusnya diupload malah akhirnya didownload dan menumpuki file yang seharusnya diupload. Sedangkan untuk fitur upload picture kemungkinan besar akan disediakan Google di masa mendatang pada SDK Drive yang terbaru.

7. Daftar Pustaka

[1] McLain, Jay F. (1998), Offline viewing of internet content with a mobile device. Washington : United States Patent.

[2] Alam, Salim, Bhalerao, Vinayak A., Wu, Charles, & Hu, George (1998), File Object Synchronization between a desktop computer and a mobile device. Washington : United States Patent.

[3] Cane, David, Hirschman, David, Speare, Philip, & Vaitzblit, Lev (1998), File Comparison for backup and file synchronization. Mass : United States Patent.

[4] Hung, Shih Ben, Larson, Richard C. (1996), The document synchronizer : a partial solution for real-time replication. Massachusetts : Thesis (M. Eng.)-- Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science.

[5] Cox, Russ, Josephson, William (2005), File Synchronization with vector time pairs. Massachusetts : Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory Technical Report.