スマートフォン間の大容量ファイル送受信アプリケーションの試作

2012SE021 古田 将士 2012SE214 櫻井 友輔
指導教員 後藤 邦夫

1　はじめに

近年スマートフォンが普及し 20 代の普及率は 9 割を超えている [6]．この先さらにスマートフォンの普及率は上がっていくと思われる．それに伴いスマートフォンアプリケーションの需要は上がっている．必要なアプリケーションがインストールされていることで，スマートフォンはより便利なものになる．またスマートフォンのカメラ性能がよくなってきているため，画像，動画を撮る機会が増え，その画像，動画をやりとりすることも増えている．こうした中，動画などの大容量ファイルを送受信するアプリケーションはあまり見当たらない．クラウドという手段もあるが，クラウドは動画の送受信を目的としたものではないため，送受信を第一の目的とし，動画などのやりとりを無駄が無く簡単にするため大容量ファイル送受信アプリケーションを試作しようと考えた．

そこで本研究では Web ブラウザをつかってファイルを交換する方法で実現を試みた．具体的には片方のスマートフォンで Web サーバを動かしもう片方のスマートフォンでそこにアクセスし，ファイルを交換する．Web サーバを起動させる方法は，Python のモジュールの 1 つである SimpleHttpServer[4] を使う．しかしこれだけではうまくファイルを交換することはできない．理由として，CGNAT(Carrier GradeNetwork Address Translation)[1] の関係でグローバル IP アドレスが取得できず，うまく通信ができない．CGNAT を通過するために Web サーバとは別に IP アドレス登録サーバを立てる．Web サーバを動かすスマートフォンは，いったん IP アドレス登録サーバに通信してから Web サーバを動かす．その結果，CGNAT を通過した状態で Web サーバを動かすことができるので，スマートフォン同士の通信が可能になる．

また本研究では動画などの大容量ファイルの送受信を考えているので，Wi-Fi 通信で実装したいと考えている．しかし，データ通信でもできた方が便利なので，Wi-Fi，データ通信ともに動作するアプリケーションを試作する．

それぞれの役割分担は古田が iOS アプリケーションの試作を担当し，櫻井が Android アプリケーションの試作を担当した．

2　概要

アプリケーションの概要を図 1 に示す

本研究では 1 対 1 でのファイル交換を考えている．図 1 のスマートフォン A が送信側，スマートフォン B が受信側である．また，サーバーの役割としては，スマートフォンの IP アドレスの取得を簡単にするためのもので，送信側受信側ともにいったんサーバの方に接続したのち，直接通信しファイルを交換する．

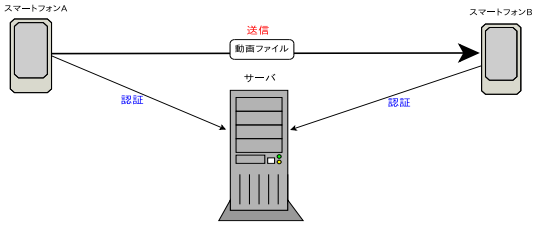

図 1　概要

このアプリケーションを実装する上で問題となることは 2 つあり，1 つはファイルを交換する相手を特定する事であり，もう 1 つは相手を特定した後のファイルを送る通信手段である．本節ではこれらの解決方法を説明する．

2.1　相手の特定

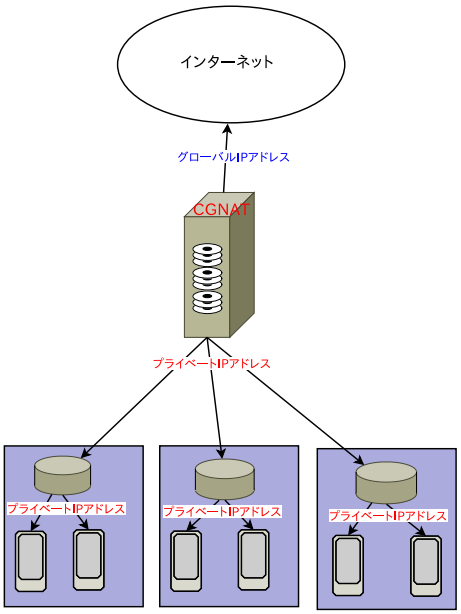

図 2　CGNAT

普通，相手のグローバル IP が分かれば直接通信することが可能である．スマートフォンにはグローバル IP アドレスが割り振られているはずなので，発売当初から 2010 年頃 [3] まではグローバル IP アドレスが割り振られいた．しかし，それ以降スマートフォンの普及に伴い IPv4 アドレスが枯渇してきたために，現在スマートフォンの IPv4 アドレスにはグローバル IP アドレスが割り振られてない．本研究で使うスマートフォンのキャリアは CGNAT が使われている．CGNAT で使用されるアドレスは RFC6598[2] で定義されている CGNAT 専用の

100.64.0.0/10 が使用される．このアドレスはグローバルアドレスでもプライベートアドレスでもないものであると定義されている．CGNAT が使われることで，一つのグローバル IP アドレスを 100.64.0.0/10 とポート番号の組み合わせで複数の端末で使っていることになる．その結果，スマートフォンの特定が難しくなっている．研究をはじめた当初は Dynamic DNS(Dynamic Domain Name System) サービスを使用しよう考えていた．

Dynamic DNS サービスのしくみは，図 3 のように接続先の IP アドレスをドメイン名にあらかじめ紐付けしておき，接続するときにドメイン名を入力するだけで相手との通信が可能とするものである．しかし，スマートフォンに CGNAT が使われているので，Dynamic DNS サービスを使って相手を特定するのは困難である．

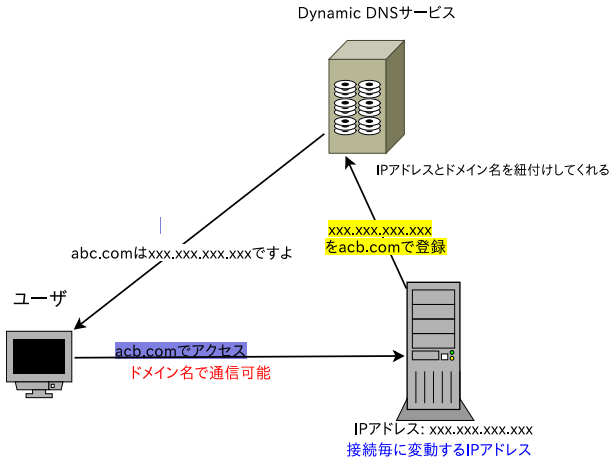

図 3　DDNS

本研究では，アクセスしたスマートフォンの IP アドレスとポート番号を登録する TCP サーバを立て，そこに送信側が TCP ソケットを使い通信したのち，同じソケットを使い HTTP サーバを起動させることでスマートフォン同士の通信を可能にさせるプログラムを作る．プログラムについては第 4 節の実装で詳しく説明する．

2.2　通信方法

ファイルを交換するための通信プロトコルは HTTP を利用する．送信側のスマートフォンで HTTP サーバを起動させる．その後受信側が送信側の IP アドレスを取得し web ブラウザを経由して送信側のスマートフォンにアクセスする．この IP アドレスを取得するときに TCP ソケットを利用する．

プログラムの詳細については第 4 節の実装で説明する．

3　開発環境

本研究では Android OS と iOS の両方で動作するアプリケーションを作成したい．

普通アプリケーションを開発する場合 Android OS と iOS で使うプログラミング言語が異なる．また開発環境も Android は Android Studio，iOS は Xcode が提供されているが，AndroidOS，iOS それぞれの OS で動作するアプリケーションを試作した場合，そのアプリケーションを異なる OS 同士で通信できるように改良する必要がある．その方法だと時間がかかるので作業効率が悪い．また通信が成功するとは限らないので，始めから共通で動作するような方法は無いのかと探した結果，Python を使用する方法が良いとアドバイスを受けた．Python を動かすためのアプリケーションは Android OS では QPython，iOS では Python2.7 が提供されていたので，本研究では Python を使用してアプリケーションを試作する．Python のバージョンは 2.7．実験で使用する AndroidOS のバージョンは Android 5.02 で，iOS は iOS9.2，PC は Ubuntu 10.04LTS を使用する．

4　実装

ここでは作成したプログラムの仕組みとアプリケーションの流れを説明する

4.1　プログラムについて

ファイルのやりとりを行なうために

- 登録サーバ
- ユーザ情報登録プログラム
- ファイル提供プログラム

の 3 つのプログラムを作成した．

登録サーバでは，アクセスがあったときにその宛先の IP アドレスとポート番号を記録しておくことができるプログラムである．ユーザは Web ブラウザを経由してこのサーバにアクセスできる．アクセスすると登録されたユーザ名リストある．そのユーザ名のリンクをたどることで受信側は送信側のスマートフォンにアクセスすることができる．このプログラムは PC 上で動作させておく．

図 4 で，コマンド/が入力されると登録してあるユーザ情報を送信する．コマンド/USER が入力されるとアドレスとポート番号を新しく記録する．

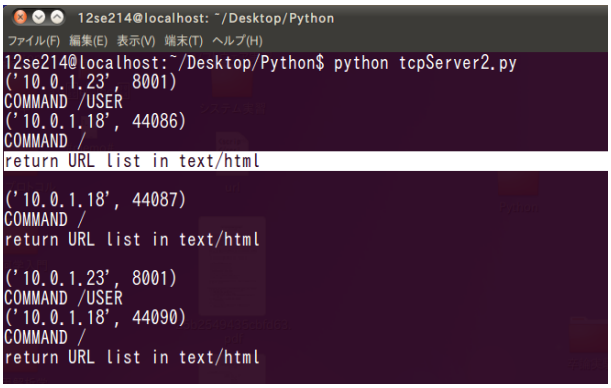

図 4　登録サーバを起動したときの画面

ユーザ情報登録プログラム

```
s = socket
  .socket(socket.AF_INET,socket.SOCK_STREAM)
s.setsockopt
  (socket.SOL_SOCKET,socket.SO_REUSEADDR, 1)
s.bind(('0.0.0.0',8000))
s.connect(('10.0.1.18',8888))
```

ユーザ情報登録プログラムでは，TCP ソケットを作成し s.setsockopt() でソケットのオプションを設定する．s.bind() でアドレスとポートを指定する．s.connect() でサーバに接続する．s.connect() のアドレスは登録サーバが動いているところに指定する．

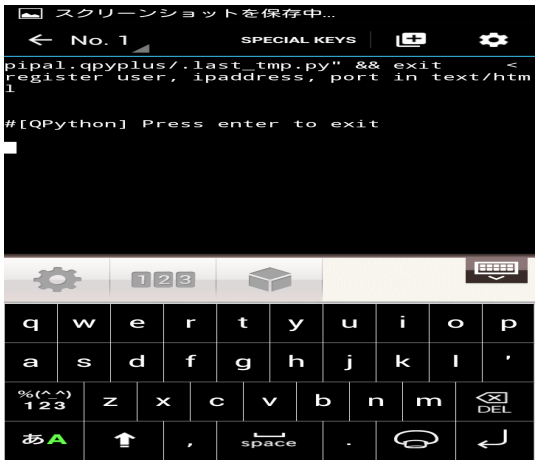

図 5　実行画面

ファイル提供プログラム

```
DOCUMENT_ROOT = '/strage/sdcard1/MyRoot'
os.chdir(DOCUMENT_ROOT)
PORT =8000
Handler = SimpleHTTPServer
          .SimpleHTTPRequestHandler
httpd = SocketServer
        .TCPServer(("", PORT), Handler)
print "serving at port",PORT
```

ファイル提供プログラムは，普通に動かしてしまうとスマートフォン内のすべてのファイルが見えてしまう．そのため os.chdir() を使用して自分が指定したフォルダのみを相手に見せられるよう設定した．DOCUMENT_ROOT は自分の好きなフォルダに設定できるが，送信したいファイルを DOCUMENT_ROOT で設定したフォルダの中に保存しておく必要がある．

今回の実験ではスマートフォン内の/strage/sdcard1/MyRoot に送信したいファイルをおいておく．

4.2　アプリケーションの流れ

ファイルを交換するときの流れは図 6 の左上のようにリストからユーザ名のリンクをクリックする．クリックされたユーザがファイル提供サーバを動かしていると，図 6 の右上のような画面に切り替わり，ダウンロードできるファイルのリンク一覧が表示される．その中からほしいファイルを選択しダウンロードする．

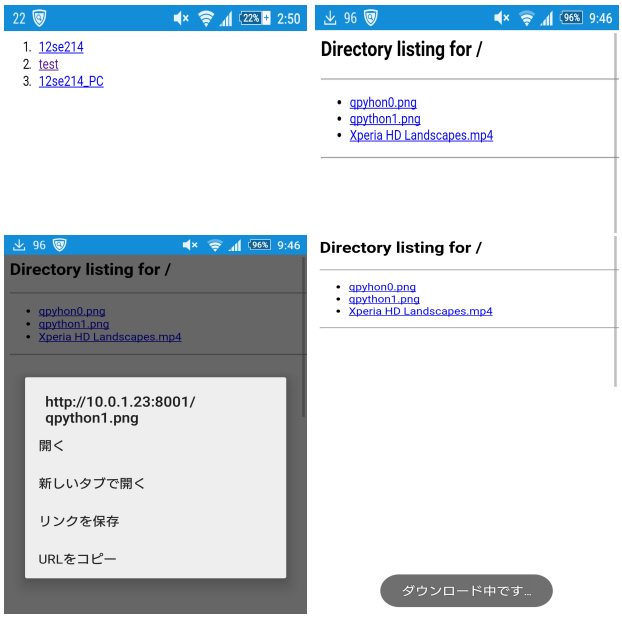

図 6　実験画面

4.3　iPhone での不具合

iPhone で上記のプログラムを動かそうとしたがうまくいかなかった．iPhone ではアプリケーションごとにファイル空間が分離されていることが原因だった．そのため Python プログラムを動かすことはできるが，提供するファイルを選択して Web 上に置くことが困難だった．また，Web ブラウザでファイルは保存できるが，動画の再生ができなかった．そのため iPhone で動作するアプリケーションを別で作成することを試みた．調べてみたところ MultipeerConnectivity[5] という仕組みが iOS7 から追加されていた．これは擬似的な P2P 通信を行なうための仕組みで，近距離通信に限られるがサーバを介さずデータ通信を行うことができる．iPhone ではこの仕組みを使い swift で iPhone 同士で通信と送受信ができるアプリケーションを試作した．

クライアント側のプログラムでは，MCAdvertiserAssistant クラスの start メソッドを使うことで，クライアントがホストに検索されるようになる．

ホスト側のプログラム MCBrowserViewController クラスを使うことで，検索可能なクライアントを表示できる．

ブラウズ画面で通信したいクライアントを選択し，クライアント側がホストと通信することを許可すると通信が成功し，データの送受信が可能になる．

ホスト側プログラム

```
Controller = MCBrowserViewController
   (serviceType: SERVICE, session: session)
Controller.delegate = self
Controller.minimumNumberOfPeers = 1
Controller.maximumNumberOfPeers = 1
self.presentViewController(Controller,
         animated: true, completion: nil)
```

5　実験と考察

作成したプログラムをスマートフォン実機で実験をする．Wi-fi 通信で実験し，登録サーバプログラムは PC で動作させる．

登録サーバ，ユーザ情報登録プログラム，ファイル提供プログラムは，すべて同じ Wi-fi ネットワークで動作させる．

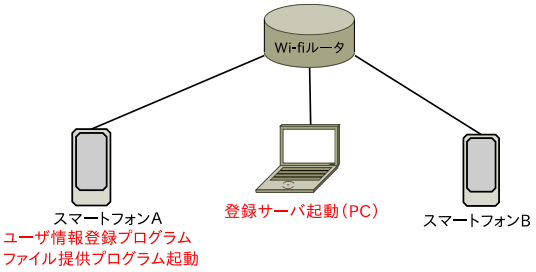

図 7　実験環境

5.1　実験結果

データ通信での実験は成功した．

ファイル交換する実験は Wi-Fi 通信でしたが，画像ファイル, 動画ファイルともにファイルのダウンロードに成功した．約 100MB の動画ファイルも 20 秒程度とそこまで時間はかからなかったので利用するのに不便は無い．また，自分が DOCUMENT_ROOT で指定したフォルダ以外は中身が見えていないことも確認できた．

Android 同士，Android と PC 間では送受信ともに成功した．Android から iPhone への送信は画像ファイルはできたが動画ファイルはダウンロードできなかった．また，iPhone から android への送信は画像ファイル，動画ファイルともにできなかった．

Python プログラムが動く環境があればスマートフォン同士だけでなく，PC とスマートフォン間，PC 同士でもファイルの交換が可能である．実際に図 6 のスクリーンショットはスマートフォンから PC に送ったものである．

iPhone 同士の MultipeerConnectivity の仕組みを使った実験では，通信と文字データを送ることに成功した．画像データや動画データを送信するにはカメラロールへのアクセスをしなければいけないが，Apple の制限により特別なクラスを使わないといけないなどの問題がある．

5.2　考察

同じ Wi-fi ネットワークに繋がっている端末同士でないと通信ができず，違う Wi-fi ネットワークに繋がっている場合，登録サーバには接続ができたが，登録サーバからユーザ名のリンクをたどってもソケットエラーになってしまい，うまく接続ができなかった．理由として，Wi-fi ルータが外から中への TCP の接続を拒否しているからではないかと考えられる．

Python を使ったプログラムでは, Apple がファイル保存場所を自由にできないよう制限をかけているため、iPhone で動作させることができなかった．

6　おわりに

AndroidOS 同士ではデータ通信 Wi-fi 通信ともに実験は成功した．iOS 同士でも通信は成功し，文字データは送受信できた．

課題としてはファイルを交換するのにプログラムを一つ一つ動かさなければならないので，そこを手間なくできるように GUI でうまくまとめる必要がある．また，iOS から AndroidOS への通信は課題が残る結果となった．

参考文献

[1] Japan Network Information Center.: インターネット 10 分講座：大規模 NAT(Large Scale NAT:LSN) あるいはキャリアグレード NAT(CGN) (acceced Nov. 2015). https://www.nic.ad.jp/ja/newsletter/No41/0800.html.

[2] Kuarsingh, V.: IANA-Reserved IPv4 Prefix for Shared Address Space (acceced Jan. 2016). https://tools.ietf.org/html/rfc6598.

[3] NTT DOCOMO, INC.: 2010 年スマートフォン新サービス・機能スマートフォン向けサービス提供基盤 (acceced Nov. 2015). https://www.nttdocomo.co.jp/binary/pdf/corporate/technology/rd/technical_journal/bn/vol18_3/vol18_3_038jp.pdf.

[4] Python Software Foundation.: SimpleHTTPServer 簡潔な HTTP リクエストハンドラ (acceced Nov. 2015). http://docs.python.jp/2/library/simplehttpserver.html.

[5] UP-frontier, Inc: iOS 7 特集 (acceced Jan. 2016). www.gaprot.jp/pickup/ios7/vol3/.

[6] 不破雷蔵 : 携帯電話の普及率現状をグラフ化してみる (2015 年)(最新) (accessed Apr. 2015). http://www.garbagenews.net/archives/2157553.html.