卒業論文 2009 年度 (平成 21 年度)

Uni-Fi: 異種モバイルデバイスのための ファイルアクセスシステムの構築

指導教員

慶應義塾大学環境情報学部

徳田　英幸

村井　純

楠本　博之

中村　修

高汐　一紀

重近　範行

Rodney D. Van Meter III

植原　啓介

三次　仁

中澤　仁

武田　圭史

慶應義塾大学 環境情報学部

井村 和博

iphoo@ht.sfc.keio.ac.jp

卒業論文要旨 2009 年度 (平成 21 年度)

Uni-Fi: 異種モバイルデバイスのためのファイルアクセスシステムの構築

論文要旨

近年，計算機や情報通信技術の発達により，様々な機能を搭載し，かつネットワーク到達性を持つモバイルデバイスが普及している．例えば携帯電話は，今日では音声ファイル，映像ファイル，写真ファイルなどのメディアファイルを作成・再生できるようになっている．他にもメディアファイルを作成・再生可能かつネットワーク到達性を持つモバイルデバイスとして，ポータブルミュージックプレイヤーやデジタルカメラなども挙げられる．これらのデバイスは，携帯電話とデジタルカメラを所持する，といったように，一人のユーザが異なるデバイスを一台ずつ所持している場合が多い．また，これらのデバイスは大きなデータ保存領域を持ち，数千から数万といった膨大な量のメディアファイルをデバイス内に保持する事ができる．このような環境下で，ユーザはデバイス内のファイルにアクセスするために多くの手間を割いている．

例えば以前撮影した写真を再生したいと思った場合，ユーザがどのデバイスで写真を撮影したかを記憶していなければファイルがどのデバイスに入っているかわからないため，全てのデバイスを起動して一つづつファイルを探さなければならない．また，デバイスごとに搭載するユーザインタフェースが異なるため，ユーザはデバイスを操作してファイルを再生するために，複数のデバイスのユーザインタフェースの扱い方を覚えなくてはならない．さらに，複数人，もしくはそれ以上でファイルを共有するために，家庭内用のデバイスでは UPnP や，それをベースに作成された DLNA といったプロトコル，また Web 上では Flikr や YouTube といったサービスが存在する．

しかし，これらの手段は自宅で利用する事を想定しているため，環境側にアクセスポイントが必要であったり，またはインターネット接続環境が必要である．モバイルデバイスは外出先で利用される事が多く，外出先では自宅のようにネットワークインフラが整っているとは限らないため，既存のファイルアクセス手法ではモバイルデバイスに対応出来ない．そこで，本研究では無線アドホック通信を用いた異種モバイルデバイスのためのファイルアクセスシステムを構築する．IEEE802.11b/g 規格の無線インタフェースを持つモバイルデバイスを対象として，異なる機能を持つ複数のデバイスが存在する際，ネットワークインフラの無い環境でもデバイス内のファイル一覧，デバイス横断的なファイル検索，ファイルの転送，といったメディアファイル再生時に必要なファイルアクセスを行えるシステムを構築する．

キーワード

モバイルデバイス，ユビキタスコンピューティング，デバイス連携，ファイルアクセス

慶應義塾大学 環境情報学部

井村 和博

Abstract of Bachelor's Thesis Academic Year 2009

Uni-Fi : File Access System for Heterogeneous Mobile Devices

Summary

Recently, a mobile device that is able to connect to the Internet and has multi functions are widespread because of the development of computer and communication engineering.For example, mobile phones sold in these days are able to generate or play a music file, movie file and photo file. Not only mobile phones but also portable music players and digital still cameras are able to connect to the Internet and has multi functions. Users are tend to have these devices in combination, like having a mobile phone and a digital still camera each.

Keywords
Mobile Device, Ubiquitous Computing, Device Collaboration, File Access

Keio University Faculty of Environment and Information Studies
Kazuhiro Imura

目次

図目次

表目次

第 1 章

序論

1.1 研究背景

近年，電子工学や情報通信技術の発展により，様々なデバイスやモジュールが高性能になり，かつ小型化された．コンピュータのストレージとして代表的なハードディスクドライブは，10 年前の 1999 年当時，最大容量が約 40GB だったのに対し，現在では最大容量 2TB のものが販売されている．コンピュータの頭脳である CPU も，10 年前と比べ格段に性能が向上している．このような条件が揃い，多くのデバイスが単一の機能のみではなく，複数の機能を持ち合わせるようになった（表 1.1）．例えば，携帯電話は当初，通話機能しか持たなかったが，現在では写真や動画を撮影する事に加え，インターネット上からファイルやアプリケーションをダウンロードし，実行する事ができるようになっている [6]．また，ポータブルオーディオも同様に，写真や動画を撮影したり，Wi-Fi などを用いてインターネットに接続することができるようになっている [5]．

表 1.1　各デバイスの持つ機能

デバイスの種類	元の機能	近年追加された機能
ポータブルミュージックプレイヤ	音楽の再生	写真の撮影，動画の撮影 写真の表示，動画の再生
携帯電話	音声通話	写真の撮影，動画の撮影 音楽の再生，写真の表示，動画の再生
デジタルスチルカメラ	写真の撮影，写真の表示	動画の撮影，動画の表示
デジタルビデオカメラ	動画の撮影，動画の再生	写真の撮影，写真の表示

本研究では，近年普及しているデバイスの中でも

- コンテンツファイル (写真ファイル，映像ファイル，音楽ファイル) のいずれかを再生もしくは生成することができる
- 無線ネットワークインタフェースを有している
- 複数のアプリケーションが実行可能である
- ユーザが携帯可能である

という条件に当てはまるデバイスをスマートモバイルデバイスと定義する．これは，携帯電話やポータブルオーディオ，デジタルスチルカメラやデジタルビデオカメラに当てはまる．

図 1.1 はポータブルオーディオ iPodtouch の仕様表である．楽曲であれば 14000 曲をデバイス内に保持できる．このように，数千から数万のコンテンツファイルをデバイス内に保存するデバイスが普及して行くと予想される今後，ユーザはより多くのコンテンツファイルを扱うようになると考えられる．

図 1.1　ポータブルオーディオ iPodtouch の仕様 [5]

スマートモバイルデバイスの普及を背景として，デバイス同士がコンテンツファイルを共有するために発展したファイルアクセス手法がいくつか存在する．

YouTube [3] は「個人が番組を作り，配信して楽しんで欲しい」というコンセプトのもと運用を開始した．このサービスでは個人がコンピュータ内に保存されている映像コンテンツファイルをアップロードし，YouTube にアクセスした他のユーザが，それを閲覧できるようにしている映像コンテンツ共有サービスである．ここでは 5000 万を超える映像ファイルが共有されており，一日平均で 35,000 本もの映像ファイルが追加されている．

Flickr [1] は，デジタルスチルカメラや，携帯電話で撮影した写真コンテンツファイルを共有するためのサービスである．ユーザがアップロードした写真コンテンツファイルは，第三者が閲覧できるとともに，タグと呼ばれるメタデータを用いてユーザ自身や，Flickr にアクセスした他のユーザが分類を行える．Flickr のようなサービスの提供と，カメラ付き携帯電話ユーザの増加により，ユーザは日常的に写真を共有するようになっている．このように，ユーザが作成したコンテンツを他者に公開し共有したいといったニーズが近年顕在化した．

DLNA [4] は，デバイス連携を行うための規格である．DLNA 規格に対応したデバイス同士は，ホームネットワーク上でコンテンツを共有し，お互いに機能を制御する事が可能となる．例えば，コンピュータ上に存在する映像ファイルをネットワークテレビに転送し，さらにその再生の制御をコンピュータ上から行う，といった利用例が挙げられる． ここでは，ホームネットワーク上でのコンテンツ共有が実現されている．

ネットワークを通じてコンテンツを共有する手法が存在し，それらはユーザのコンテンツを共有したいといったニーズを満たしている．しかし，ユーザはその他にも，モバイルデバイス同士で一時的にコンテンツファイルを共有したいといった要求を持つ．例えば，友人と旅行に行き写真を撮影した際に，その場で写真を共有したいといった要求がある．

また，スマートモバイルデバイスはデバイスに搭載されているディスプレイやスピーカといったモジュールが小さく，テレビや据え置き型のスピーカといった据え置き型デバイスと比べると性能的に劣る場合が多い．そのため，コンテンツを複数人で同時に視聴したい時には，スマートモバイルデバイス内のコンテンツファイルを据え置き型デバイスに転送し，より性能の良いデバイスでコンテンツファイルを再生したいといった要求

を持つ．

図 1.2 はファイルアクセス手法を分類した図である．YouTube や Flickr，DLNA などが提供するファイルアクセス手法は，ユーザがコンピュータの前に座り，コンピュータはインターネットに接続され，それを用いてコンテンツにアクセスを行うホームユース寄りの手法である．

複数人のユーザが無線ネットワークモジュールを搭載したデバイスを持って集まり，一時的なネットワークを構築してお互いのコンテンツファイルに対してファイルアクセスを行うモバイルユースな場面には対応できない．そのため，このような場面で利用できるファイルアクセス手法の確立が望まれる．

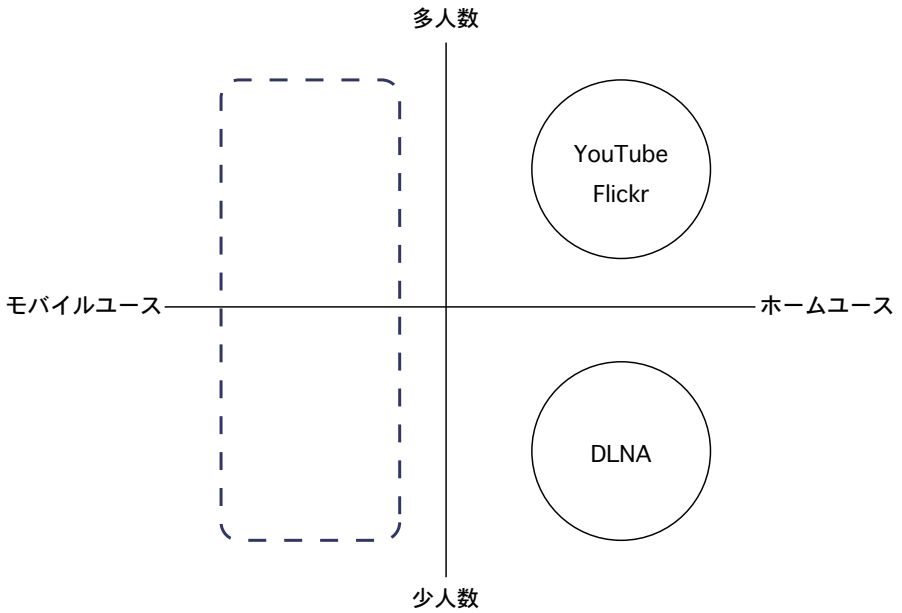

図 1.2　ファイルアクセス手法の分類

1.2　研究目的

本研究では，背景で述べたモバイルユースのファイルアクセス手法として，アドホックネットワークを用いたモバイルデバイス間ファイルアクセスシステム Uni-Fi を提案する．Uni-Fi では，デバイス同士がアドホックネットワークを構築し，その上で互いにコンテンツのメタデータとファイルを交換する．これにより，インターネット接続環境の無い場所でも，デバイス間におけるファイルアクセスを可能とし，ユーザのコンテンツ共有における要求を満たす．

1.3　本論文の構成

本論文は全８章で構成される．本章では本研究の背景と目的を示した．次章では既存のファイルアクセス手法について言及し，それらが抱える問題点を挙げ，モバイルデバイスのためのファイルアクセスの機能要件を述べる．　　第３章では，第２章で取り上げた機能要件から，モバイルデバイスにおけるファイルアクセス手法を提示する．第４章では本研究と関連研究の比較を行う．第５章では提示した手法に対する設計を解説する．第６章では設計に対する実装を解説する．第７章ではシステムの評価基準と評価結果を述べる．最終章で

本論文のまとめを行い，今後の課題を述べる．

第 2 章

ファイルアクセス手法

本章ではファイルアクセスの概念とファイルアクセス手法を整理する．次に，スマートモバイルデバイス上で既存の手法によってファイルアクセスを行った場合に，どのような問題が生じるかをシナリオを挙げて解説する．また，それらの問題から，スマートモバイルデバイスに必要なファイルアクセスシステムのための機能要件を導く．

2.1 ファイルアクセス手法

本節ではファイルアクセス手法の概念について詳しく述べ，概念の整理を行う．次に，ファイルアクセス手法の概念から，ファイルアクセス手法の機能を整理する．

2.1.1 ファイルアクセスの定義

ユーザが目の前のコンピュータで音楽を再生したいと考えた時，ユーザはまず目的の音楽ファイルを探すためにアプリケーションを起動する．アプリケーション上でファイルの一覧を見て，目的のファイルがあればそれを選択する．結果，流れた音楽が，選択したものであると確認する．

本研究では，ユーザがファイルにアクセスする場合のファイルアクセスに必要な機能として

1. ファイルの一覧を取得する機能
2. ファイルに対してデータを読み書きする機能
3. ファイルを移動する機能

の 3 つを挙げる．以下で，これらの機能について詳しく述べる．

1. ファイルの一覧を取得する機能
   ユーザがファイルを探す際には，ユーザ自身が目的のファイルについて名前や作成日などのメタデータ，または内容についての情報を断片的に持っている場合と，それらの情報を全く持っていない場合とに分けられる．ユーザが過去に目的のファイルを開いていれば，その情報を断片的にユーザが記憶している場合が多い．例えば，過去に撮影した写真を再び見たいと検索する場合は，これにあてはまる．この場合は目的のファイルについて情報を持っているので，ファイル名やファイルの作成日時といった情報を一覧として表示すれば，ユーザは求めているファイルと探しているファイルとの同一性を確認できる．しかし，ユーザが何らかの目標を達成するために過去に一度も開いたことの無いファイルを検索している場合には，目的のファイルが定まっていない．そのため，ディレクトリやデバイスなど，指定した場所に存在する全てのファイルの名前や，場合によってはサムネイルなどの内容を一目で確認出来る情報を一覧として表示する必要がある．

2. ファイルに対してデータを読み書きする機能
   ファイルはデータの連続を任意の場所で区切るための単位である．データは何らかの処理を行った際に，読み出される場合と，その結果を出力される場合がある．その際，それらのデータをファイルとして読み書きする機能がファイルアクセスには必要となる．

3. ファイルを移動する機能
   ファイルの保存場所は，主にユーザの操作によって変更される．ディレクトリ構造を持っているファイルシステムでは，ユーザがどこに何を保存してあるかをユーザ自身が覚えておくために，任意のディレクトリ名のディレクトリを作成し，そこにファイルを保存できるようになっている．また，以前作成したファイルを整理するために，既に記録されているディレクトリから，他のディレクトリに移動できる機能を持っている．ファイルを移動する機能は，このようにファイルを別のディレクトリに移動できる

機能である．

2.1.2 ファイルアクセス手法の定義

ファイルアクセスに対してファイルアクセス手法とは，どのようにファイルアクセスを行うかといった概念である（図 2.1）．例えば，デバイス間でファイルを移動する際に，USB メモリなどの外部記憶装置を使用して行う事が可能である．また，インターネットに接続していればネットワーク上のサーバにファイルをアップロードし，それを他のデバイスがダウンロードする事によってファイルを移動できる．あるいは，デバイス間で telnet などを利用して一対一で通信を行い，そこでファイルを移動するといった事も可能である．

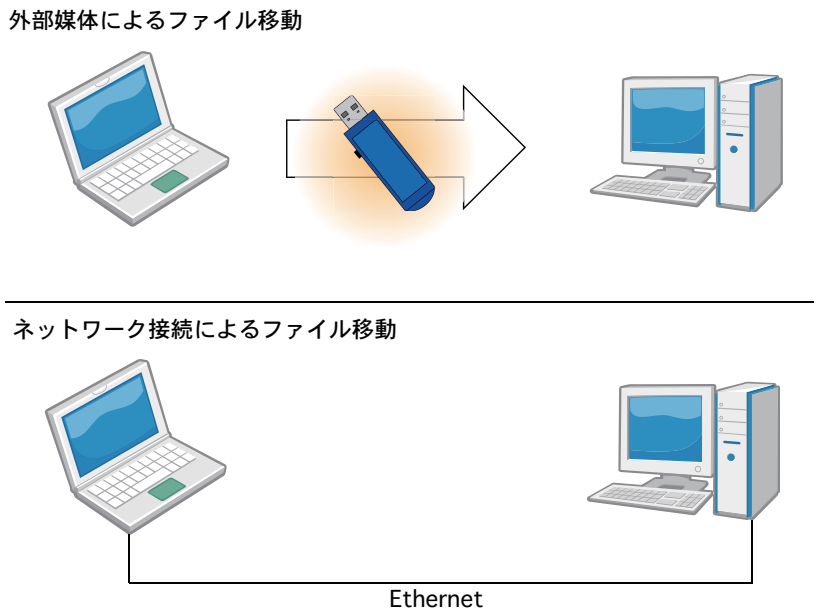

図 2.1　複数のファイルアクセス手法

本研究では，ファイルアクセスを行う際に取る手法をファイルアクセス手法と定義する．

2.2 問題意識

序論で，近年大容量のストレージを搭載するデバイスが増え，さらにユーザがそのようなデバイスを複数台所持しており，各デバイスがネットワーク到達性を持っている事を述べた．しかし，スマートモバイルデバイスがファイルアクセスを行う際，インターネット接続環境が無い場合や，複数のデバイス間でファイルアクセスを行う場合，に，様々な問題が発生する．以下で，本研究が想定する問題を，シナリオを挙げつつ述べる．

2.2.1 インフラストラクチャへの依存

- シナリオ：旅先での写真共有
  A君は友人の B 君，C 君と旅行に出かけた．このとき，３人とも写真を撮影・再生できるスマートモバイルデバイスを持っている．旅先でA君が写真を撮影し，とても良い写真が撮れたので，それをB君

とC君が欲しいと言った．しかし，写真を共有して閲覧するために Web サービスにアクセスしようとしたが，インターネット接続環境が旅先には無かったため，写真を共有する事ができなかった．

このシナリオでは，デバイスがインターネットに接続することができず，Web サービスに接続することができなかったため，写真ファイルを共有することができなかった．しかし，カメラや携帯電話といったスマートモバイルデバイスは，頻繁に家の外に持ち出され，場所を問わず利用される．そして，その場にインターネット接続環境があるとは限らない．既存のサービスでは複数台のスマートモバイルデバイス間でファイルを共有するシナリオに対応出来ない．

2.2.2 複数デバイス間におけるファイルアクセスのコスト

- シナリオ：デバイス内のファイル検索
  A君は自宅で以前撮影した写真を探していた．A君はその時，その写真を携帯電話で撮ったのかポータブルオーディオで撮ったのか，それともコンパクトカメラで撮ったのかを忘れてしまっていた．そこで，全てのデバイスを起動して，一つづつファイルを探していった．しかし，一つのデバイスに数千のファイルが入っていて，かつ写真を確認するためのディスプレイがとても小さく，さらに操作方法もデバイスにより別々だったので，A君は目的の写真を探す作業が非常に大変だと感じ，探すことを諦めてしまった．

このように，身の回りに複数のデバイスが存在している時，それらのデバイスは別々のユーザインタフェースを持つ．例えば，携帯電話のユーザインタフェースは小さいディスプレイとテンキーと十字キーであり，コンピュータのユーザインタフェースは大きいディスプレイとキーボードとマウスである．各デバイスのユーザインタフェースはハードウェア面だけではなくソフトウェア面も異なる．デバイスを駆動するための OS は各デバイスによって異なるため，どのボタンを押してどのような画面遷移を経ればファイルにアクセスできるか，といった点も異なる．そのため，ユーザがデバイスごとに扱い方を学習しなければならない．デバイスごとにかかる学習時間はユーザにとって煩わしさを感じさせる．仮にユーザが全てのデバイスの扱い方を覚えても，デバイスはそれぞれメモリｘｘードやハードディスクといったメディアを搭載しており，それらに含まれるファイルを一括して検索したりする手法は無い．そのため，ファイルを検索するためには全てのデバイスを起動して確認しなければならない．これはユーザにとって大きな労力と時間を要求する．

2.2.3 デバイスのオフライン化

- シナリオ：デバイス内のファイル検索
  A君は自宅で以前撮影した写真を探していた．A君はその時，その写真を携帯電話で撮ったのかポータブルオーディオで撮ったのか，それともコンパクトカメラで撮ったのかを忘れてしまっていた．A君は通常，所持しているコンピュータに撮影した写真を全て転送して保持させているが，今回探しているファイルに関しては転送することを忘れてしまったようだ．A君がいくらコンピュータ上でファイルを検索しても，目的のファイルは候補に上がってこなかった．

現在，多くのユーザはデスクトップコンピュータなどの大きいストレージを持つ機器にファイルを転送し，そこでファイルの一括管理を行う．しかし，スマートモバイルデバイスはデスクトップコンピュータなどの据え置き型デバイスと異なり，電源としてバッテリーを用いている．バッテリーの供給できる電力は有限であ

るため，デバイスを使用しない時には電源を落とす．また，電力の消費を抑えるために無線ネットワークモジュールを必要なときにだけ使用し，それ以外の時はモジュールの電源を切る事がある．このような理由により，スマートモバイルデバイスはネットワーク到達性を失う場合がある．ネットワーク到達性を失っているデバイス内のファイルにアクセスすることや，どのようなファイルが中に入っているかを調べることは困難である．

2.3 機能要件

前節で述べた通り，モバイルデバイスがコンテンツファイルにアクセスする上で 3 つの問題が存在する．本節ではスマートモバイルデバイスがモバイル環境でコンテンツファイルにアクセスする上で必要な機能要件を述べる．

2.3.1 インフラストラクチャに依存しないファイルアクセスの実現

モバイル環境では，インターネットへの接続環境や，無線アクセスポイントといったインフラストラクチャが存在しない場合がある．そのような環境下でも，複数のデバイス同士がファイルアクセスを行える必要がある．これを実現するにあたり，デバイスが自律的に動作し，他のデバイスとインフラストラクチャを用いずにネットワークを構築して，ファイルアクセスを行う必要がある．

2.3.2 複数のデバイスに対する統一的なファイルアクセス手法

複数のデバイスが存在し，それぞれが別々のユーザインタフェースを持っている場合でも，ユーザはそれを意識せずに，普段使い慣れたユーザインタフェースを利用してファイルアクセスを行える必要がある．また，複数のデバイスがそれぞれ別のメディアを内蔵していたとしても，ユーザはそれを意識せずに，ファイルアクセスを行える必要がある．この機能が達成されれば，ユーザから見た複数デバイス間におけるファイルアクセスを簡素化し，ユーザインタフェースの学習にかかるコストと，ファイルの検索を行う際に毎回デバイスを起動するといった労力を省く事ができる．

2.3.3 オフラインになってしまったデバイスへの対応

スマートモバイルデバイスは電源が常時入っていてネットワーク到達性があるとは断言できない．よって，デバイスがネットワーク到達性を持っていない時でも，記録されているファイルを他のデバイスから検索出来る必要がある．もしネットワーク到達性が無いデバイス内にユーザの求めているファイルが存在したときに，そのデバイス内にファイルがあるということをユーザが確認できれば，デバイスを起動してファイルを取り出すことができる．オフラインになっているデバイスにファイルが存在するかどうかを一括して検索できれば，ユーザにとってファイルアクセスにかかる時間的なコストや労力的なコストを大幅に減らすことができる．

2.4 本章のまとめ

本章ではまず，ファイルアクセスの概念を整理し，ファイルアクセスに必要な機能としてファイルの一覧を取得する機能，ファイルに対してデータを書き込む機能，ファイルを移動する機能の 3 つを挙げた．次に，

ファイルアクセス手法の例を挙げた上で，現状のファイルアクセス手法の抱える問題をシナリオと共に挙げた．また，シナリオからインフラストラクチャへの依存，複数デバイス間におけるファイルアクセスのコスト，デバイスのオフライン化という 3 つの問題点を導き．これらの問題点に対処するための機能要件を述べた．次章では，本研究と関連研究を機能要件をもとに比較する．

第 3 章

関連研究

本章ではスマートモバイルデバイス間での連携やファイルアクセスを行っている cogma, サマリを用いたコンテンツ集約機構，HomeShare，Mobile Media Content Sharing の 4 つの研究を挙げ，それらの特徴と問題点を述べる．また，機能要件をもとに Uni-Fi との比較を行う．

3.1 関連研究

本節では関連研究の概要を述べる．

3.1.1 cogma

cogma [13] は名古屋大学の河口らによって開発されたアドホックネットワーク環境における組み込み機器を対象としたデバイス間連携用のミドルウェアである．河口らが開発した MAGNET を利用して無線ネットワークを自己形成し，通信を行うことができる．外部からの通信によりデバイスを制御することができるが，デバイスが持つファイルに対するアクセスなどに関しては触れられていない．したがって，Uni-Fi で挙げている機能要件を満たしていない．

3.1.2 サマリを用いたコンテンツ集約機構

関西大学大学院の近藤らが開発したこのシステム [14] では，複数のデバイスが存在するアドホックネットワーク上で，各デバイスがサマリと呼ばれるファイルのメタデータを記録する XML ファイルを自動生成し，それらを共有する．これにより，インフラストラクチャに依存しないファイルアクセスを実現している．しかしサマリの共有を，ユーザがリクエスト送信した際に行っているため，どのデバイスにどのファイルが入っているかはリクエストを送信するまでわからない．また，サマリをリクエストが終了した段階で破棄してしまうため，Uni-Fi の機能要件であるオフラインになってしまったデバイスに対するファイル検索は実現されていない．

3.1.3 HomeShare

Bilhanan Silverajan らが開発を行った HomeShare [11] は，無線アクセスポイントが予め設置されている家庭環境で，デバイス間の Peer to Peer 通信により，サーバなどの中間ノードを必要としないデバイス同士のファイル共有を可能にしている．HomeShare でプロトタイプ実装されたアプリケーションでは，HomeShareClient と呼ばれる，自らの持つファイルのメタデータを作成するプロセスと，HomeShareHub と呼ばれる HomeShareClient の作成したメタデータを集約するプロセスが存在する．HomeShareHub が他のデバイスで作成されたメタデータも集約することで，家庭環境に存在するデバイスのファイル検索が可能となっている．しかし，デバイス同士が通信を行う際にアクセスポイントが存在することが前提となっているため，Uni-Fi の機能要件であるインフラストラクチャに依存しないファイルアクセスを達成出来ていない．

3.1.4 Mobile Media Content Sharing

Chih-Lin Hu らが開発したこのシステム [9] では，上述の HomeShare と同様に予め無線アクセスポイントが設置されている家庭環境で，UPnP [7] に対応したデバイスが写真や音楽，映像ファイルといったコンテンツファイルを共有することを実現している．他のデバイス上にコンテンツを転送し，再生をコントロールするといった機能や，UPnP との親和性にフォーカスが当てられているため，Uni-Fi の機能要件であるインフラストラクチャに依存しないファイルアクセス，オフラインになってしまったデバイス内のファイル検索などは満たされていない．

3.2 関連研究との機能要件による比較

本節では，Uni-Fi と関連研究の比較を，Uni-Fi の機能要件をもとに行う．

表 3.1　Uni-Fi と関連研究の比較表

研究名	インフラレス	統一的なファイルアクセス	オフラインデバイス対応
cogma	△	△	×
サマリを用いたコンテンツ集約機構	○	○	×
HomeShare	×	○	×
MobileMedia Content Sharing	×	○	×
Uni-Fi	○	○	○

表 3.1 は機能要件を比較した結果を表す表である．この表で示す通り，Uni-Fi で挙げた 3 つの機能要件を満たす研究は無い．インフラストラクチャに依存しないファイルアクセスを実現しているのはサマリを用いたコンテンツ集約機構のみである．近藤らが開発したこのシステムでは，この他にも複数のデバイスに対する統一的なファイルアクセスを実現しているが，電源の入っているデバイスしか対象とならないため，オフラインになってしまったデバイスに対するファイルアクセスには対処できない．HomeShare と MobileMedia Content Sharing では，統一的なファイルアクセスを実現しているが，図 3.1 のように，システムを利用する環境に無線アクセスポイントが設置してあることを前提としているため，インフラストラクチャに依存しないファイルアクセスを達成していない．また，オフラインになってしまったデバイスにも対応しない．このよう

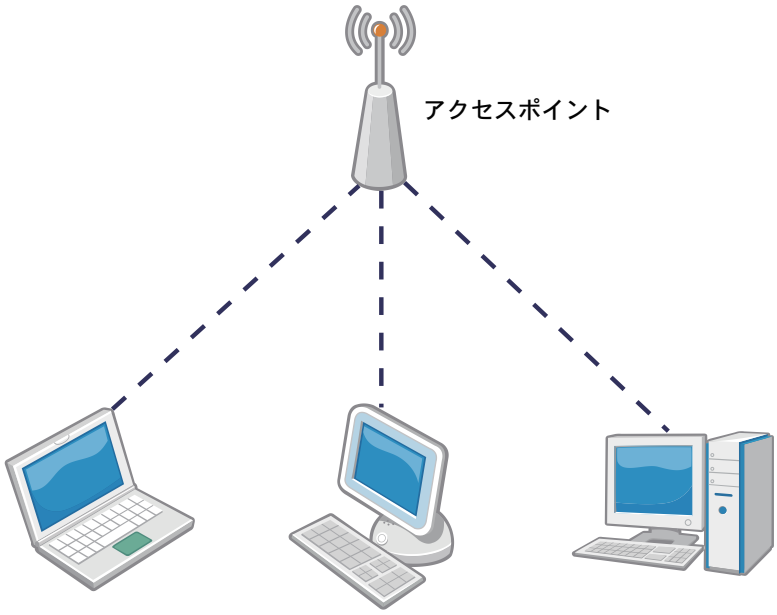

図 3.1　関連研究でのデバイス間ネットワーク構成

に，既存の研究では，Uni-Fi の挙げる問題点を全て解決している研究が存在しない．

3.3 本章のまとめ

本章では，4 つの関連研究を挙げ，本研究との比較を行った．関連研究ではインフラストラクチャに依存しないファイルアクセス，複数のデバイスに対する統一的なファイルアクセス手法，オフラインになってしまったデバイスへの対応といった本研究で定義した機能要件を全て満たす研究が存在しないことを確認し，本研究の優位性を示した．次章では，前章で挙げた機能要件を達成するためのアプローチについて述べる．

第 4 章

モバイルデバイスのためのファイルアクセス手法

本研究では，2 章で挙げた機能要件であるインフラストラクチャに依存しないファイルアクセス，複数デバイスに対する統一的なファイルアクセス，オフラインになってしまったデバイスへの対応を実現するアプローチとして，デバイス間の自律的なネットワーク構築によるファイルアクセス，仮想的な単一ストレージの生成，ファイルメタデータのキャッシュという 3 つのアプローチを取る．本章では，これらのアプローチを取るに至った経緯と，各アプローチの詳細を述べる．

4.1 Uni-Fi：モバイルデバイスのためのファイルアクセス手法

4.1.1 デバイス間の自律的ネットワーク構築によるファイルアクセス

インフラストラクチャに依存しないネットワーク構築方法として，近距離無線通信を用いたアドホック・ネットワークの構築が挙げられる．序論で述べた通り，スマートモバイルデバイスは無線ネットワークインタフェースを有している．スマートモバイルデバイスに搭載されるインタフェースは主に Bluetooth [10] や IEEE802.11b/g [2] の二つであり，これらはどちらも近距離無線通信に使用される．また，どちらもインフラストラクチャに依存しないアドホック・ネットワークを構築する事が可能である．

Bluetooth は，ポータブルオーディオにヘッドセット接続するといった用途で用いられている．それに対して IEEE802.11b/g は，ブラウザを使用して Web を閲覧する，音楽をインターネット上からダウンロードする，といった用途で用いられることが多い．Bluetooth は IEEE802.11b/g と比べネットワークの帯域が小さいため，映像などのサイズの大きいファイルを転送することには不向きである．よって，Uni-Fi では IEEE802.11b/g 規格の無線ネットワークインタフェースを搭載したデバイスを想定する．

IEEE802.11 にはインフラストラクチャ・モード [8] とアドホック・モード [12] の 2 つの通信形態がある．インフラストラクチャ・モードではアクセスポイントが空間内にあることが前提となる．よって Uni-Fi の機能要件を満たさない．アドホック・モードではアクセスポイントが無い空間で，他のデバイスと通信を行う事がサポートされている．これはインフラストラクチャに依存せずアドホックな通信を行うことが可能であるため，Uni-Fi の機能要件を満たす．よって，Uni-FI ではアドホック・モードを用いて通信を行う．

インフラストラクチャに依存しないネットワークが構築された上で，そのネットワーク上でデバイス同士がファイルアクセスを行う手法について議論したい．ネットワーク上でデバイス同士がファイルアクセスを行う際に，現状では二つのアプローチが考えられる．一つは YouTube や Flickr，DLNA のように，ネットワークのどこかに全てのファイルを集約しているサーバが存在する，サーバ集約モデルである．

- サーバ集約モデル
  図 4.1 はサーバ集約モデルの概念図である．サーバ集約モデルでは，中央にサーバが存在し，そのサーバがファイルを集中的に管理し，ファイルを要求するデバイスに対してファイルを転送する．また，各デバイスは新しいファイルを生成した場合は，サーバに接続した際に同期を取る．

もう一つは，接続しているデバイス同士で自律的にファイルアクセスを行う，Peer to Peer（以下，P2P）モデルを用いたファイルアクセスである．

- Peer to Peer モデル
  図 4.2 は P2P モデルを用いたファイルアクセスの概念図である．P2P モデルでは，中央で集中管理を行うサーバが存在しない．デバイス同士が自律的に動作し，互いにファイルを転送しあうことによりファイルアクセスを行う．

サーバ集約モデルでは，全てのファイルをサーバが所持しているので，サーバになるデバイスのストレージやネットワークに負荷がかかりがちである．また，サーバになっているデバイスがネットワークから切断されてしまった際に，他のデバイスはファイルアクセスを行うことができなくなってしまう．P2P モデルでは，各デバイスにかかる負荷は均等であり，一つのデバイスがオフラインになってしまっても，そのデバイスに入って

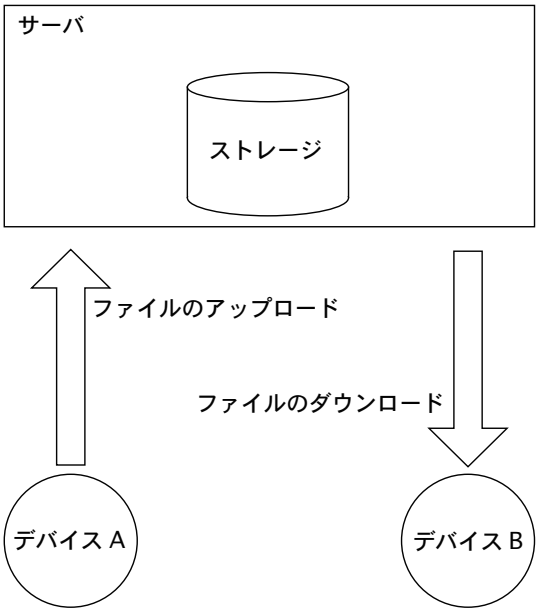

図 4.1　サーバ集約モデル

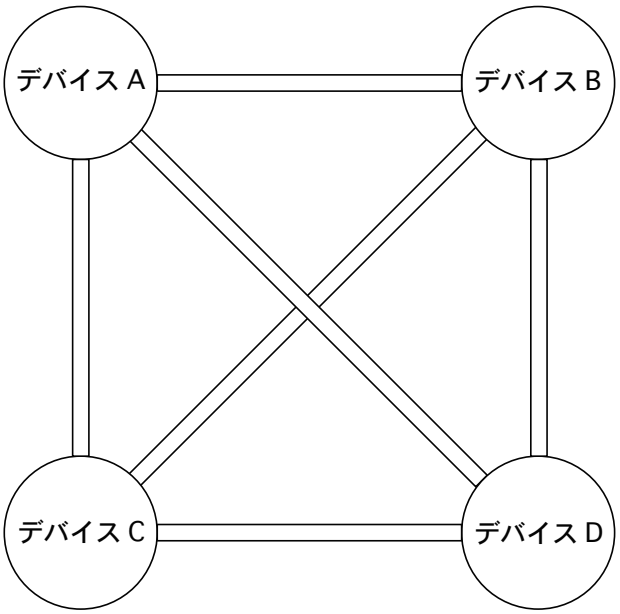

図 4.2　Peer to Peer ファイルアクセスモデル

いるファイルにしか影響が出ない．したがって，ネットワークへの接続解除やデバイスの電源断などが頻繁に生じるモバイル環境におけるファイルアクセスには，P2P モデルの方がメリットが大きい．よって，Uni-Fi では P2P モデルを用いてファイルアクセスを行う．

Uni-Fi では，IEEE802.11b/g のアドホック・モードを用いることで，アクセスポイントやインターネット接続環境などのインフラストラクチャに依存しないネットワーク構築を行う．また，その上でデバイス同士が P2P な通信を行うことで，サーバなどのインフラストラクチャに依存しないファイルアクセスを実現する．

4.1.2 仮想的な単一ストレージの生成

複数のデバイス間でのファイルアクセス方法が定義された上で，それを統一的に扱うアプローチを述べる．図 4.3 は複数のデバイスが存在した際の，既存のファイルアクセス手法における概念図である．この図では，

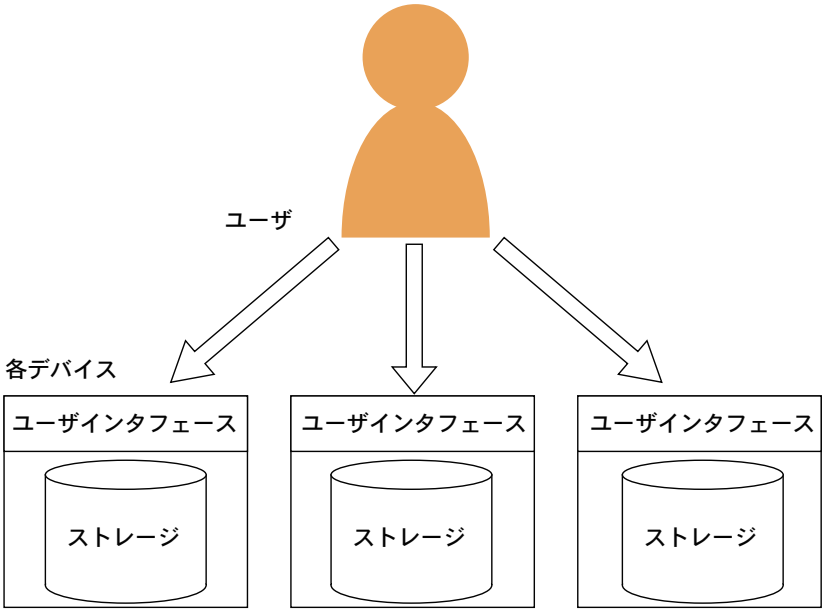

図 4.3　複数デバイス存在時のファイルアクセス概念図

ユーザから見た際に，デバイスに対して一つずつのストレージとユーザインタフェースが存在しているため，複数のデバイスにまたがってファイルアクセスを行う際に，ユーザはそれを意識してファイルアクセスを行う必要がある．例えば，デバイスＡで作成したファイルを，デバイスＢ上で再生したい時，デバイス間でファイルを移動する必要がある．この際，ユーザはどちらのデバイスに対してもファイルを移動させるための操作をしなければならない．これは 2 章で述べた通り，ユーザインタフェースの学習コストや，一つの目的に対して必要以上に多くの操作があるといった点で，ユーザに大きな負荷をかけている．

そこで，Uni-Fi では，複数デバイスにまたがったファイルアクセスを行う際に，図 4.4 のように仮想的な単一ストレージを生成する．ユーザから見た際，複数のデバイスに対して一つのストレージが共有されているように見える．このアプローチを採用したシステム上でユーザがデバイス間のファイル移動を行うとする．デバイスＡのファイルをデバイスＢに移動するにあたり，デバイスＢのユーザインタフェース上ではデバイスＡのファイルが確認出来るため，ユーザは複数存在するストレージを意識せず，一つのデバイスから全てのデバイスに含まれるファイルに対してアクセスすることができる．

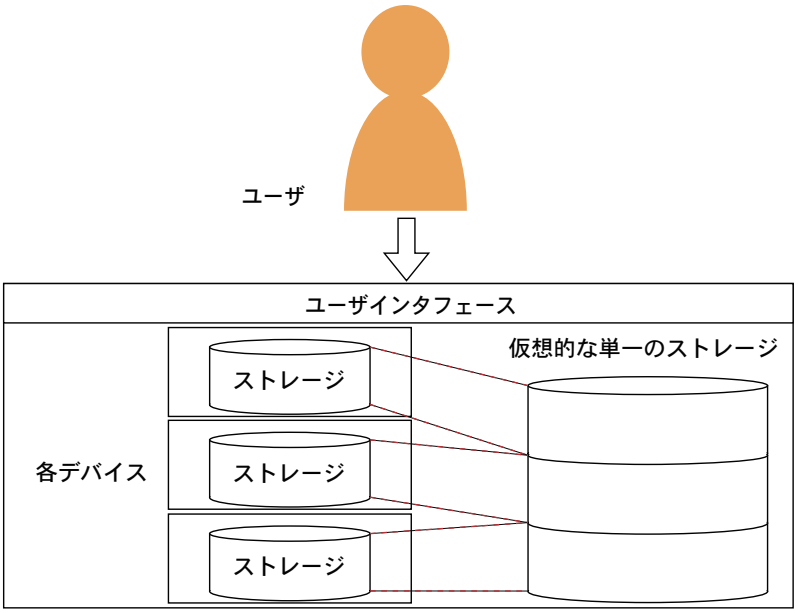

図 4.4　Uni-Fi におけるファイルアクセス概念図

仮想的な単一ストレージを作成するにあたり，Uni-Fi では各デバイス内に存在する全てのコンテンツファイルのメタデータの共有を行う．このアプローチの正当性に関しては次の 4.1.3 項で述べる．

4.1.3　ファイルメタデータのキャッシュ

前章で，スマートモバイルデバイスは，据え置き型デバイスと比べて頻繁にネットワーク到達性が絶たれ，それがファイル検索を行う際の障害になると述べた．また，前項にて，仮想的な単一ストレージを生成すると述べた．これらを踏まえ，Uni-Fi では，デバイス同士がお互いの持つファイルのメタデータを共有することにより，この問題を解決する．

P2P におけるファイル検索方法での分類

P2P 通信でのファイル共有方式として，ハイブリッド P2P，スーパーノード P2P，ピュア P2P の 3 種の方式が存在する．これらは，主にファイル検索を行う部分を異なるアプローチで実現している．

ハイブリッド P2P は，サーバにファイルのメタデータと，それを持っているデバイスを記述したインデックスを持たせる．デバイスは検索を実行する際にサーバにアクセスし，インデックスから目的のファイルを持っているデバイスを検索する．ここまではサーバクライアントモデルに基づく方法である．ファイルを持っているデバイスを特定した上で，そのデバイスとは P2P 通信によってファイル転送を行う．

スーパーノード型 P2P では，特定の選ばれたデバイスがファイルのメタデータとデバイス情報のインデックスを持ち，他のデバイスはスーパーノードに選ばれたデバイスに対して検索クエリを発行する．

ピュア P2P は，ネットワーク上に存在する全てのデバイスに対して検索クエリを発行し，ファイルを持っているデバイスがそれに対して返答を行う．ファイルの転送も P2P 通信で行われる．

表 4.1 では，これらの P2P のファイル共有方式の特徴から，本研究の機能要件に対応するかどうかを示した表である．これらの 3 つの方式でデバイス間のファイル検索を行う際，検索クエリを発行する側のデバイス

表 4.1　検索実行方式の比較

	ハイブリッド P2P	スーパーノード型 P2P	ピュア P2P
オフラインデバイスへの対応	△	×	×
アドホックネットワークとの親和性	×	△	○

は，クエリを他のデバイスもしくはサーバが受け取り，返答を行うまで，どのデバイスにどのファイルが入っているかを把握できない．よって，ユーザが要求しているファイルを持っているデバイスがオフラインになってしまった場合には対応できない．

ハイブリッド P2P は，条件付きでオフラインデバイスに対応する．例えば，デバイスがオフラインになる前にファイルのインデックスをサーバに対して共有していれば，そのデバイスがオフラインになってしまっても，ファイルの検索は可能である．しかし，ハイブリッド P2P はインデックスをサーバに集中させて管理を行っているため，インターネット接続環境の無いアドホックネットワークではサーバに接続できず，ファイル検索を行えない．よって本研究ではこの方式を採用しない．

スーパーノード型 P2P は，完全に P2P な通信のみを用いてファイル検索と転送を行う．スーパーノードと呼ばれる，広帯域のネットワークに接続されていて，かつ処理能力が高く，長時間ネットワークに接続されているデバイスにインデックスが集中するように設計されている．サーバを用いないため，ハイブリッド型よりアドホックネットワークとの親和性が高いが，スマートモバイルデバイスを常時ネットワークに接続して使用することは考えにくいため，スーパーノードの決定が難しい．また，スーパーノードが決定されても，そのデバイスがオフラインになってしまった場合にはファイル検索を行えなくなってしまう．よって本研究ではこの方式を採用しない．

ピュア P2P は，デバイス間の完全な P2P 通信に加え，全てのデバイスは並列に動作し，その上でファイル検索と転送が行われる．アドホックネットワーク上でピュア P2P によるファイル検索を行った際，各デバイスは自らの持つファイルのインデックスのみを持つため，一つのデバイスがオフラインになった場合でも，そのデバイスの持つファイルに対する検索クエリしか損なわれない．よって，この方式はスマートモバイルデバイスにおけるアドホックネットワークとの親和性が高い．本研究ではこの方式を改良し，ファイル検索を行う．

ピュア P2P の改善

ピュア P2P 方式でファイル共有を行うアプリケーションでは，どのデバイスがどのコンテンツを持っているか，ネットワーク上で検索を行うまで未知である．よって，ネットワークに接続されていないオフラインなデバイスには対応できない．そこで，Uni-Fi では予めファイルのメタデータを全てのデバイス間で共有し，キャッシュすることにより，オフラインになってしまったデバイスに対応する．検索用インデックスのキャッシュを行うことによりオフラインデバイスへの対応は行えるが，デバイスが新しくファイルを生成した際に各デバイスのキャッシュに対してその通知を行わなければ，新しく生成されたファイルは検索結果に現れない，といったトレードオフの関係にある．Uni-Fi ではオフラインデバイスへの対応に重点を置くため，予めファイルメタデータのキャッシュを行う方法を選択した．

2 章でファイルアクセスに必要な機能は

1. ファイルの一覧を取得する機能

2. ファイルに対してデータを読み書きする機能
3. ファイルを移動する機能

だと述べた．ネットワーク上で任意のデバイス内の任意のファイルを特定するにあたり必要な情報は，デバイス名，ネットワークアドレスとファイルパスである．また，ユーザがファイルの一覧を閲覧する際に，ファイルの種類や作成日時，最終更新日時などによってソートを行う場合が多い．これを考慮し，それらをストレージ内から抽出し，記述する必要がある．本研究では，これらの情報を XML ファイルに記述し，接続しているデバイス同士で共有する．これをもとに，前項で述べた仮想的な単一ストレージを生成する．仮想的なストレージ内でファイルを移動する際に，移動する範囲がデバイスにまたがっている場合は，どのデバイスのどのファイルをどのデバイスに対して転送するかを解決するためにも利用する．メタデータの記述方法とデバイス，ファイル名解決については５章で詳しく述べる．

4.2 本章のまとめ

本章では，２章で述べた機能要件を達成するために，デバイス間の自律的ネットワーク構築によるファイルアクセス，仮想的な単一ストレージの生成，ファイルメタデータのキャッシュといった３つのアプローチを行う事を述べた．また，スマートモバイルデバイス間でのファイルアクセスにおいて，それらのアプローチを取ることの正当性を述べた．次章では，これらのアプローチをシステムに落としこむための設計について述べる．

第 5 章

Uni-Fi の設計

本章では，前章で挙げたアプローチをシステムに落としこむための設計について述べる．まず，設計概要として想定環境とソフトウェア構成を述べ，次にシステムの動作概要を述べる．最後に，ソフトウェア構成で挙げたソフトウェアモジュールの設計について詳しく述べる．

5.1 設計概要

本節では，Uni-Fi の設計概要を述べる．

5.1.1 想定環境

Uni-Fi は図 5.1 のように，複数のスマートモバイルデバイスが存在する環境で動作することを想定している．また，環境にインターネット接続環境やアクセスポイントが無いことが前提となっている．その上で，各デバイスは他のデバイスを発見し，メディアファイルメタデータの共有を行う．また本研究では，全てのデバイスが Uni-Fi をインストールしている事を前提とする．

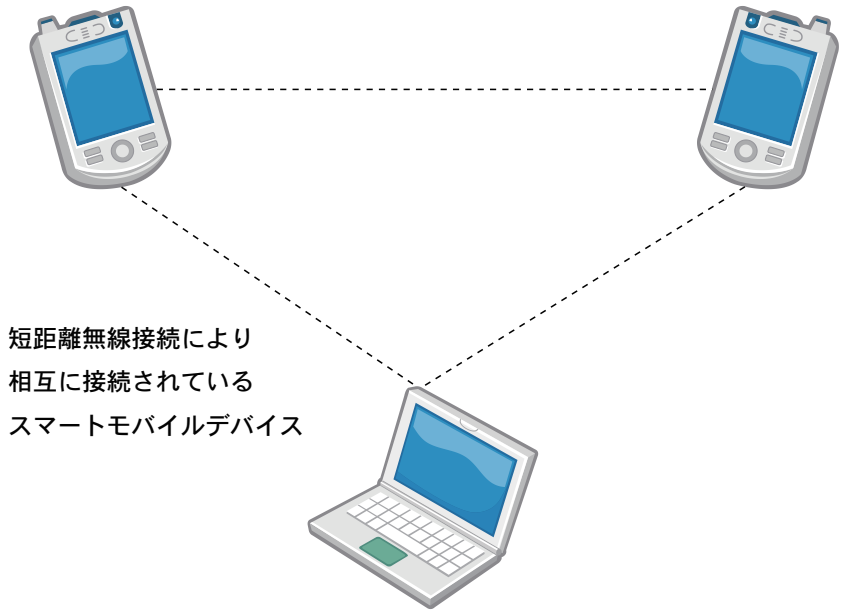

図 5.1　Uni-Fi の想定環境

5.1.2 ソフトウェア構成

Uni-Fi は 4 つのモジュールから構成される．他のデバイスとの通信を管理するネットワーク管理部，他のデバイスに対する通信に必要な情報を保持するデバイス情報管理部，仮想的な単一メディアを生成するメタデータ管理部，他のデバイスの要求によりファイルの転送を行うファイル転送管理部の 4 つである．図 5.2 はそれらの関係性を表したソフトウェア構成図である．また，各部の詳細は 5.2 節以降で述べる．

5.2 Uni-Fi の動作概要

本節では，Uni-Fi の動作について述べる．まずフロー図を用いてデバイス同士の挙動を確認した後，シーケンス図を用いて，各デバイスに搭載される Uni-Fi システムのモジュールの挙動を確認する．

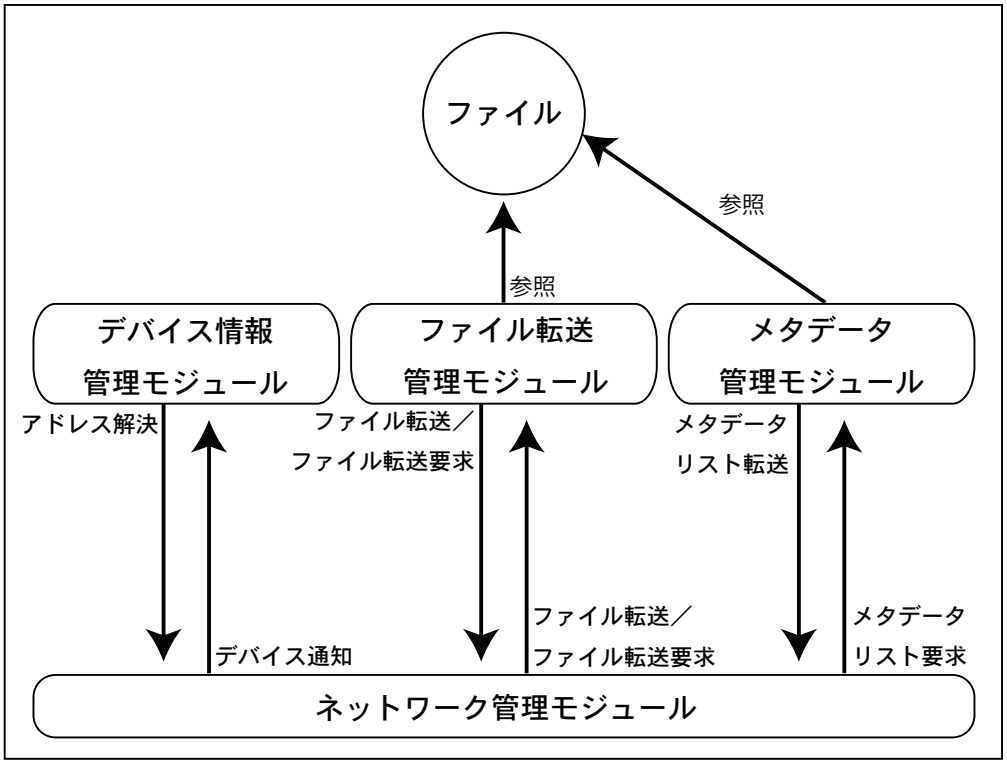

図 5.2　Uni-Fi のソフトウェア構成図

5.2.1　デバイス間の挙動

図 5.3 は Uni-Fi を用いたデバイス同士の動作フロー図である．この図では，既にアプリケーションを立ち上げて待機していたデバイスＢと，新しくアプリケーションを立ち上げネットワークに参加したデバイスＡがファイルアクセスを行うまでの流れを記した．この図の流れに沿って，アプリケーションの動作概要を説明する．

ファイルアクセス準備

図 5.3 のファイルアクセス準備では，デバイス同士がファイルアクセスを行うための準備を行う．準備には自己通知，メタデータ送付，メタデータ返送が含まれる．以下で詳しく解説する．

- 自己通知
  デバイスＡはアドホックネットワークに参加した際に，デバイスＢに対して自らのデバイス名を送信する．この際，パケットを受け取ったデバイスＢは，デバイスＡの名前と IP アドレスの登録を行う．その後，デバイスＢはデバイスＡに対して，自らの名前を送信する．デバイスＡはそのパケットを受け取った際に，デバイスＢの名前と IP アドレスの登録を行う．なお，自己通知はアプリケーション起動後も定期的に送信される．
- メタデータ送付
  メタデータ送付は，ネットワークに新たに参加したデバイスＡがデバイスＢに対してメタデータリス

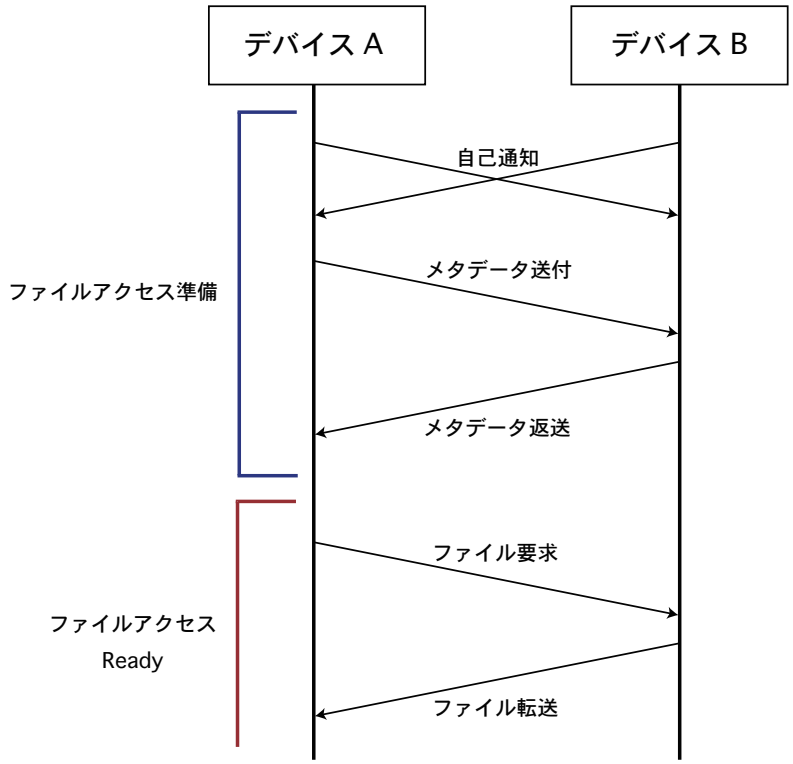

図 5.3　Uni-Fi の動作フロー図

トを送信する．メタデータを受信したデバイスＢは，それを自らの持つ統合メタデータリストに結合する．

- メタデータ返送
  メタデータを受信したデバイスＢは，送信元のデバイスＡに対して自らのメタデータリストを返送する．それを受信したデバイスＡは，自らの持つ統合メタデータリストに結合する．

ファイルアクセス

図 5.3 のファイルアクセスでは，ファイルアクセスの準備が整った上で，ユーザの要求に応じてファイルアクセスを行う．ファイルアクセスにはファイル転送要求とファイル転送が含まれる．これらについて以下で詳しく解説する．

- ファイル転送要求
  デバイスＡ上でユーザがファイルを要求した時，もし目的のファイルがデバイスＡ上に存在しなければ，デバイスＡはデバイスＢに対してファイル転送要求を送信する．
- ファイル転送
  ファイル転送要求を受け取ったデバイスＢは，要求の送信元であるデバイスＡに対してファイルを転送する．

5.2.2 各モジュールの動作概要

図 5.4 は Uni-Fi がファイルアクセス準備が完了した段階から，実際にファイルアクセスを行い，デバイス間でのファイル転送が完了するまでを示したシーケンス図である．

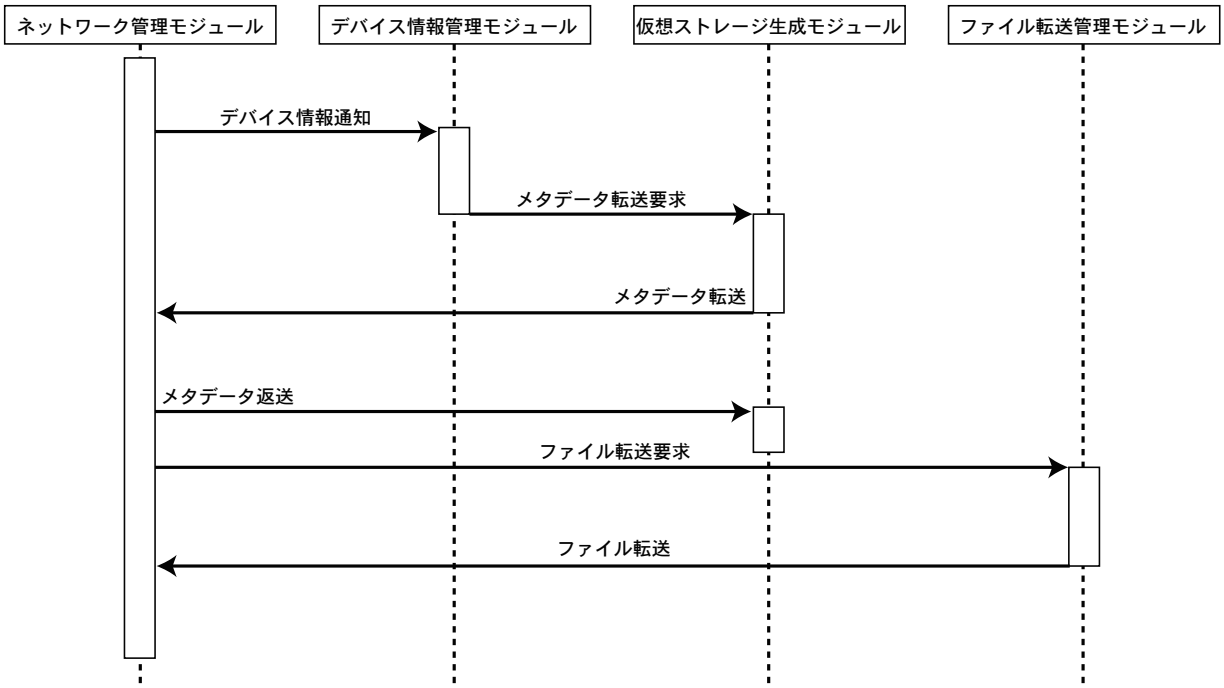

図 5.4　Uni-Fi のシーケンス図

この図に示されているモジュールに関して以下で詳しく述べる．

ネットワーク管理モジュールの設計

ネットワーク管理モジュールは，アプリケーションの起動時にネットワーク上に他のデバイスが存在するかを調べる．また，アプリケーションが起動している間は，常に自分がネットワークに存在を他のデバイスに知らせるために，パケットのマルチキャストを行う．同時に，ネットワーク上に他のデバイスが存在するかを調べ続ける．他のデバイスを発見した際には，デバイス情報管理モジュールに，発見したデバイスの IP アドレスとデバイス名を通知する，デバイス追加要求を行う．デバイス管理モジュールがメタデータの転送を行ったり，メタデータの返送を受ける場合，またファイル転送モジュールがファイル転送要求を受け取る際や，ファイル転送を行う際には，全てのデータがネットワーク管理モジュールを通り送受信される．

デバイス情報管理モジュールの設計

デバイス情報管理モジュールでは，ネットワーク管理モジュールからの通知された情報をもとに，現在ネットワークに接続されているデバイス情報のリストを作成し保持する．図 5.5 はデバイス情報管理モジュールの動作概要図である．

デバイス情報リストには，デバイスの名前，IP アドレス，メタデータリストを共有したかどうかのフラグ

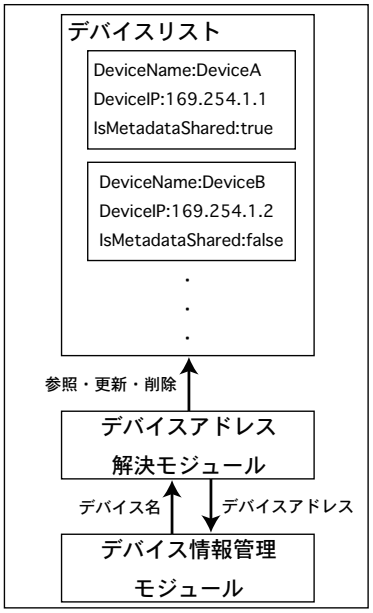

図 5.5　デバイス情報管理モジュールの動作概要図

があり，もしメタデータリストを共有していないデバイスが存在すれば，仮想ストレージ管理モジュールにメタデータ転送要求を行う．仮想ストレージ管理モジュールやファイル転送管理モジュールから要求があった際には，デバイス名とアドレスの解決を行う．デバイスが検出されなくなった場合には，デバイス情報リストからデバイスの情報を削除する．また，Uni-Fi の実行を終了した際にはデバイス情報リストからデバイスの情報を削除する．そのため，どのデバイスとメタデータを共有したか，といった情報は Uni-Fi 実行の終了と共に消失し，次回起動時には，再度，環境内に存在する全てのデバイスとメタデータリストの共有を行う．

仮想ストレージ管理モジュールの設計

仮想ストレージ管理モジュールでは，他のデバイスが所持するメタデータリストの共有と結合を行う．また，統合メタデータリストをもとに，仮想ストレージを生成し，それに対するファイルアクセスの管理を行う．仮想ストレージ管理モジュールは，５つの小さなモジュールから成る．図 5.7 は仮想ストレージ管理モジュールのモジュール構成図である．

- メタデータ管理モジュール
  メタデータ解決モジュールは，XML で記述されている自己メタデータリストと，統合メタデータリストへのアクセスを行う．仮想ストレージ管理モジュール群は，全てメタデータ管理モジュールを通して XML ファイル対するアクセスを行う．
- メタデータリスト生成モジュール
  メタデータリスト生成モジュールは，Uni-Fi が動作するデバイスの OS 上で，ルートディレクトリから再帰的にディレクトリを巡回し，全てのコンテンツファイルのメタデータを収集する．また，収集したメタデータを自己メタデータリストに記録する．さらに，Uni-Fi の起動中には常にデバイス内で新しいコンテンツファイルが生成されたかを調べ，もし新しいコンテンツファイルが生成されれば，自己メタデータリストの更新を行う．

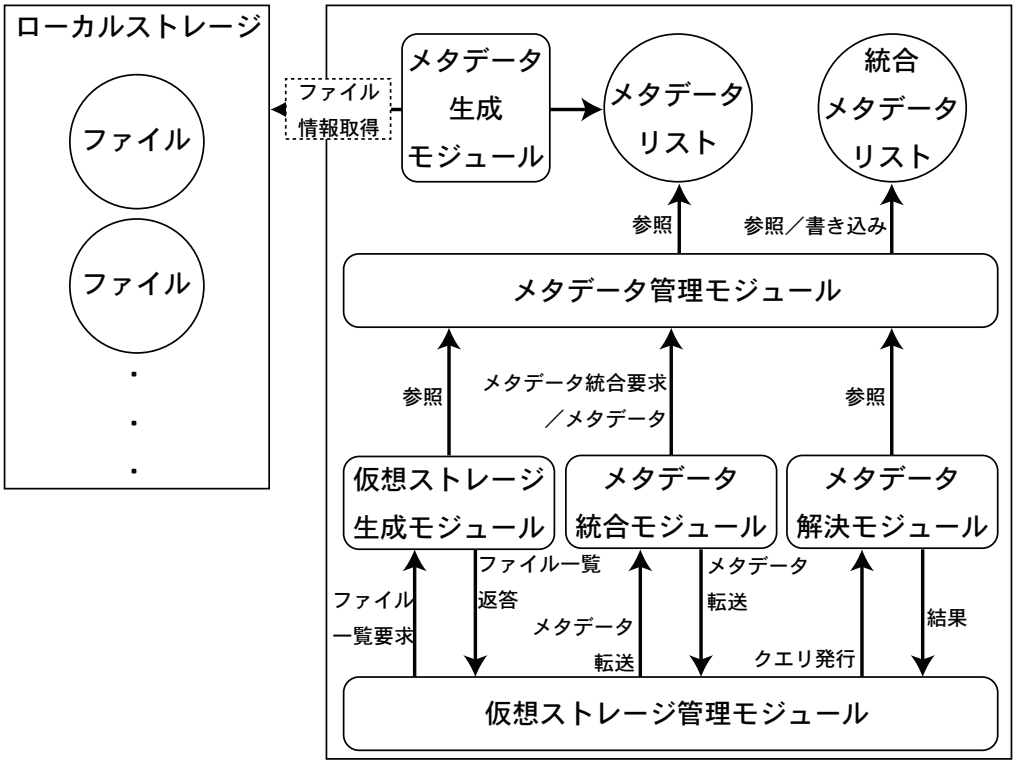

図 5.6　仮想ストレージ管理モジュールのモジュール構成

- メタデータ統合モジュール
  メタデータ統合モジュールは，デバイス管理モジュールと連携して，環境内の他のデバイスからメタデータリストを収集し，それを統合メタデータリストに対して結合する．また，デバイス管理モジュールからメタデータ転送要求を受けた際には，自己メタデータリストを取得し，転送要求元に返送を行う．Uni-Fi は，プロセスを終了しその後もう一度起動した際に，一度メタデータを共有したデバイスとも再度メタデータの共有を行う．これは，あるデバイス内で生成されたコンテンツファイルを全てのデバイス上で認識させるために必要な更新の機能を担っている．何度も同じデバイスとメタデータ共有を行った場合，統合メタデータリストには既にメタデータを共有したデバイスの情報が書き込まれている可能性がある．そのため，メタデータ統合モジュールは，メタデータが完全に重複している XML ノードを更新しない．
- 仮想ストレージ生成モジュール
  仮想ストレージ生成モジュールは，統合メタデータをもとに，環境内全てのデバイスのストレージを仮想的に一つのストレージに統合する．例えば，デバイスＡ上のファイルシステムでファイルパスが”/User/username/music/artistname/albumname/music.mp3”のような形式だった場合，Uni-Fi では ” /DeviceA/User/username/music/artistname/albumname/music,mp3 ” という形式に変換される．
- メタデータ解決モジュール
  メタデータ解決モジュールは，仮想ストレージに対するファイル検索機能を提供する．例えば，ファイル名をメタデータ解決モジュールに与えた場合，そのファイル名に対応するメタデータを統合メタデー

タリストから抽出する．これにより，ファイルがローカル上ではなく他のデバイスに存在した際に，デバイス名と，そのデバイス上でのファイルパスを把握し，ファイル転送要求を発行することができる．

ファイル転送管理モジュールの設計

ファイル転送管理モジュールは，ファイル転送要求が他のデバイスから送信された際，要求クエリに対応するファイルを，要求送信元のデバイスに転送する．また，ユーザがローカル上ではなく，他のデバイス上に存在するファイルを要求した際に，要求されたファイルが存在するデバイスに対してファイル転送要求を送信し，転送されたファイルを受け取る．図 5.8 は，ファイル転送管理モジュールと，ローカル上でのファイル保存ディレクトリの構成を示した図である．ファイル転送要求を受けた場合は，仮想ストレージ管理モジュール

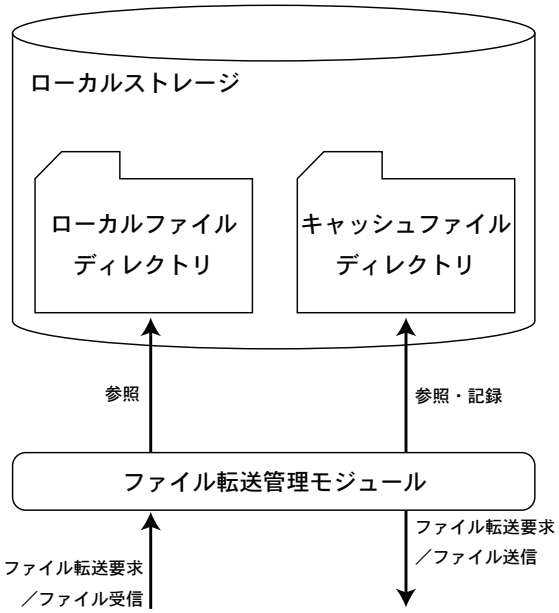

図 5.7　ファイル転送管理モジュールとローカルディレクトリの構成

で，ファイル名からファイルパスの解決を行い，ファイルの転送を行う．自らがファイル転送要求を発行する場合は，ファイルを受け取った後，キャッシュファイルディレクトにファイルを移動する．キャッシュファイルディレクトリは，Uni-Fi 起動時に初期化され，全てのキャッシュファイルが削除される．これは，デバイス間でファイル交換を行う度に，デバイス間のファイルコピーにより，同一のファイルが複数存在する事態を避けるために行っている．

5.3　本章のまとめ

本章では Uni-Fi システムの想定環境が，インターネット接続環境やアクセスポイントの無い場所で複数のスマートモバイルデバイスが存在する環境であることを述べた．また，そのような環境で動作するシステムの設計として，システムの動作概要とソフトウェア構成を述べた．さらに，Uni-Fi システムでは大きく分けてネットワーク管理モジュール，デバイス情報管理モジュール，仮想ストレージ管理モジュール，ファイル転送管理モジュールの 4 つのモジュールが存在することを述べ，それらの詳細を述べた．次章では，これらの設計

をもとにどのような実装を行ったかを述べる．

第 6 章

Uni-Fi の実装

本章では，前章で述べた設計に対し，具体的にどのような実装を行ったかを述べる．まず，実装環境として使用したハードウェアや開発環境について述べる．次に前章で挙げたネットワーク管理モジュール，デバイス管理モジュール，仮想ストレージ管理モジュール，ファイル転送管理モジュールをどのように実装したかを詳しく述べる．

6.1 実装環境

本節では，本研究における Uni-Fi の実装概要について述べる．表 6.1 は Uni-Fi の実装環境をまとめた表である．

表 6.1 Uni-Fi の実装環境

使用デバイス	VAIO Type-U
OS	Windows XP
実装言語	JAVA SE 6

表 6.1 の示す通り，プラットフォームに Windows XP を用い，実装言語として JAVA SE 6 を用いた．また，ハードウェアとして Vaio Type-U を選択した．Vaio Type-U のハードウェア仕様は表 6.2 の通りである．

図 6.1 Vaio Type-U

表 6.2 Vaio Type-U の仕様

CPU	Intel CoreSolo 1.06GHz
メモリ	512MB
メディア	HDD 30GB 4200rpm
ネットワークインタフェース	IEEE802.11 b/g

6.2 各モジュールの実装

本節では，Uni-Fi を構成する各モジュールの実装について述べる．各モジュールは JAVA 言語を用いて記述されている．

6.2.1 ネットワーク管理モジュール

ネットワーク管理モジュールは，アプリケーション上で生じる全ての無線通信を管理するモジュールである．Uni-Fi において，デバイス間の接続は IEEE802.11b/g アドホック・モードを利用して構築される．また，IP アドレスの割り当てとして AutoIP を用いた．デバイス発見と自己通知の際のパケットはコネクションレスな UDP を使用し，デバイス間通信でデータを送受信する際にはコネクション型の TCP を用いて通信を行っている．今回の実装において，自己通知のパケット送信はマルチキャストで 15 秒おきに行っている．また，パケットのデータ部分には自らのデバイス名を入れて送信している．自己通知を受信した他のデバイスは，ヘッダから IP アドレス，データ部からデバイス名を取得し，デバイス管理モジュールへそれらの情報を通知する．

6.2.2 デバイス情報管理モジュール

デバイス情報管理モジュールは，IEEE802.11b/g によって構成されたアドホックネットワーク上に存在し，かつ Uni-Fi を起動しているデバイスの情報を収集する．メタデータリスト送付，返送や，ファイル転送要求，ファイル転送などを行う際に，仮想ストレージ管理モジュールと連携する．仮想ストレージ管理モジュールでは，ファイルを保持しているデバイスの名前が解決される．ファイル転送を行う際や，ファイル転送要求を発行する際には，宛先デバイスの IP アドレスを，デバイス名をもとにデバイス情報管理モジュールが解決するため，IP アドレスを用いて通信を行うことができる．リスト 6.1 は，デバイス情報を保持するためのデータクラスである．

リスト 6.1　デバイス情報保持クラス

```
1  package jp.ac.keio.sfc.ht.iphoo.unifi.device;
2
3  import java.net.InetAddress;
4
5  /**
6   * デバイスの情報を保持するクラスです．
7   * デバイス一台につき一つがインスタンス化されます．
8   * @author iphoo
9   *
10  */
11 public class DeviceInfo {
12   private String deviceName;
13   private InetAddress deviceAddress;
14   private int lostTimes;
15
16   /**
17    * コンストラクタ
18    */
19   public DeviceInfo(String deviceName, InetAddress deviceAddress){
20     this.deviceName = deviceName;
21     this.deviceAddress = deviceAddress;
22     this.lostTimes = 0;
23   }
24
25   public String getDeviceName(){
26     return deviceName;
```

```
27    }
28
29    public InetAddress getDeviceAddress(){
30      return deviceAddress;
31    }
32
33    public int getLostTimes(){
34      return lostTimes;
35    }
36
37    public void setLostTimes(int lostTime){
38      lostTimes++;
39    }
40
41    public void clearLostTimes(){
42      lostTimes = 0;
43    }
44  }
```

DeviceInfo クラスは，デバイス一台につき一つインスタンス化され，そのデバイスがネットワーク上で何度検出不可能な状態になったかを記録している．

リスト 6.2 は，デバイス情報管理モジュールにおいて，デバイスの接続性を確認するメソッドである．

リスト 6.2　デバイスの接続性確認

```
1   public class DeviceManager {
2     private static ArrayList<DeviceInfo> DEVICE_LIST;
3
4     public DeviceManager(){
5       DeviceManager.DEVICE_LIST = new ArrayList<DeviceInfo>();
6     }
7
8     public void addNewDevice(InetAddress deviceAddress, String deviceName){
9       DeviceInfo info = new DeviceInfo(deviceName, deviceAddress);
10      DEVICE_LIST.add(info);
11    }
12
13     （中略）
14
15    /**
16     * デバイスの接続性を調べます．
17     * もし接続されていたデバイスがオフラインになっていたら，デバイスリストから削除します．
18     * @param inetAddressList
19     */
20    public void setDeviceConnection(InetAddress inetAddressList[]){
21      for(int i=0; i<DEVICE_LIST.size(); i++){
22        DEVICE_LIST.get(i).setLostTimes();
23      }
24      for(int i=0; i<DEVICE_LIST.size(); i++){
25        for(int j=0; j<inetAddressList.length; j++){
26          if(DEVICE_LIST.get(i).getDeviceAddress().equals(inetAddressList[j])){
27            DEVICE_LIST.get(i).clearLostTimes();
28          }
29        }
30        if(DEVICE_LIST.get(i).getLostTimes()>5){
31          DEVICE_LIST.remove(i);
32        }
33      }
34    }
35  }
```

環境内に他のデバイスが存在した際，ネットワーク管理モジュールから定期的に通知が行われる．その際，IP アドレスとデバイス名が通知されているので，デバイスリストに記録されているデバイスと，通知されたデバイス情報の照合を行う．もし，通知されたデバイス情報がデバイスリストに登録されていなければ，登録を行う．それと同時に，ネットワーク管理モジュールを通じて，メタデータの送信と受信を行う．逆に，デバイスリストに記録されているデバイスが一定時間検出されず，通知が行われなかった場合は，そのデバイスがオフラインになったと解釈し，デバイスリストから該当する情報を削除する．

6.2.3 仮想ストレージ管理モジュール

仮想ストレージ管理モジュールの持つ機能はメタデータリストの生成，受信したメタデータリストの結合，仮想ストレージの生成，メタデータ解決の４つである．以下でメタデータの記述形式と，仮想ストレージ管理モジュールの持つ機能に対応する実装について解説する．

メタデータリストの記述形式

メタデータリストは，デバイス内の全てのコンテンツファイルのメタデータを収集し，記述した XML 形式のファイルである．ファイルのメタデータとして，以下の情報を抽出して記述している．

- 全ファイル共通
  ファイル名
  そのファイルを保持するデバイス上でのファイルパス
  そのファイルを保持するデバイスの名前

これらの情報は，デバイス間でファイルを転送する必要が生じた際に使用する．また，メタデータリストにはファイルの種類により以下の付加情報も記述している

- 音楽ファイル
  アーティスト名，アルバム名，タイトル，ファイル継続時間（秒）
- 映像ファイル
  タイトル，ファイル継続時間（秒）
- 写真ファイル
  撮影日

これらの情報は，ユーザがファイルを一覧する際に，ソートを行うことを想定して記述した．実際のファイルメタデータリストは，XML のフォーマットに則って以下のように記述される．

リスト 6.3　メディアファイルメタデータ記述形式

```
<files>
  <file>
    <filetype>music</filetype>
    <filename>01 Feel That.mp3</filename>
    <filepath>/Users/iphoo/Documents/workspace/UniFiPC/mediafiles/01 Feel
      That.mp3</filepath>
    <devicename>iphooMacBook</devicename>
    <artist>ArtistName</artist>
    <album>AlbumName</album>
    <title>MusicTitle</title>
    <time>125</time>
  </file>
  <file>
    <filetype>movie</filetype>
    <filename>sunday morning.3gp</filename>
    <filepath>/Users/iphoo/Documents/workspace/UniFiPC/mediafiles/sunday
      morning.3gp</filepath>
    <devicename>iphooMacBook</devicename>
    <title>sunday morning.3gp</title>
    <time>600</time>
  </file>
  <file>
    <filetype>photo</filetype>
```

```
24      <filename>PHM08_0559.JPG</filename>
25      <filepath>/Users/iphoo/Documents/workspace/UniFiPC/mediafiles/PHM08_0559.JPG
26      </filepath>
27      <devicename>iphooMacBook</devicename>
28      <date>2009/11/22 04:01:42</date>
29    </file>
30  </files>
```

メタデータリスト生成モジュール

メタデータリストの生成は，アプリケーション起動時に行われる．ユーザが指定したディレクトリから再帰的にディレクトリを読み込み，その中に存在する mp3，MP4，JPG 形式のコンテンツファイル全てに対してメタデータの収集を行い，XML ファイルに記述する．記述形式はリスト 6.3 と同じフォーマットを用いる．

メタデータ統合モジュール

他のデバイスから受信したメタデータリストは，統合メタデータリストに対して結合を行う．

仮想ストレージ生成モジュール

仮想ストレージ生成モジュールでは，統合メタデータリストからファイル情報を読み込み，仮想的なストレージを生成し，そこに含まれるファイルに対するアクセスを実現する．

メタデータ解決モジュール

メタデータ解決モジュールでは，ファイルの断片的な情報をもとに，どのファイルが対応するか解決する．ファイルの断片的な情報とは，完全ではない，一つ以上のメタデータである．例えば，任意のファイルを持つデバイスのみわかる場合や，ファイルの名前だけがわかる場合がこれにあたる．メタデータ解決モジュールは，このような断片的な情報をもとに，メタデータリスト内のメタデータとの照合を行う．

6.2.4 ファイル転送管理モジュール

ファイル転送管理モジュールでは，他のデバイスから送信されたファイル転送要求を受け取り，対応するファイルを，転送要求送信元に転送する．この際には，転送要求に含まれるファイルパスで指定されたファイルをローカルディレクトリ上から取得し，ネットワーク管理モジュールにファイルストリームとして渡す．逆に，仮想ストレージ管理モジュールにファイルのメタデータを渡されて，ファイル転送要求をこちらから行う場合には，ネットワーク管理モジュールを通じて，該当するデバイスにファイル転送要求を送った後，ファイルを受け取る．この際，受け取ったファイルはキャッシュファイルディレクトリに保存する．

6.3 本章のまとめ

本章では，Uni-Fi システムの実装に関して述べた．前章の設計から必要な実装を検討し，Uni-Fi を構成する 4 つのソフトウェアモジュールであるネットワーク管理モジュール，デバイス情報管理モジュール，仮想ストレージ管理モジュール，ファイル転送管理モジュールの実装について詳しく述べた．次章では本研究で実装を行った Uni-Fi システムに対して定性的，定量的に評価を行った結果を述べる． .

第 7 章

実験と評価

本章では，Uni-Fi システムの評価を行った結果を述べる．定性的評価では関連研究との比較を行う．また，定量的評価として，スマートモバイルデバイス間でファイルアクセスを行った際の所要時間やファイル検索の所要時間などを計測し，Uni-Fi システムの性能を示す．

7.1 定性的評価

本節では，Uni-Fi の定性評価として，３章で挙げた関連研究との比較を確認する．図 7.1 は Uni-Fi の機能要件をもとに関連研究との比較を行った表である．３章でも挙げた通り，Uni-Fi はインフラストラクチャー

表 7.1 Uni-Fi と関連研究の比較表

研究名	インフラレス	統一的なファイルアクセス	オフラインデバイス対応
cogma	△	△	×
サマリを用いたコンテンツ集約機構	○	○	×
HomeShare	×	○	×
MobileMedia Content Sharing	×	○	×
Uni-Fi	○	○	○

に依存せず，かつ，複数のデバイスに対して統一的なファイルアクセス手法を備え，かつ，オフラインになってしまったデバイス内に存在するファイルも検索対象となる，唯一のシステムである．

7.2 定量的評価

本節では，Uni-Fi の動作実験を行い，ファイルアクセスに関する計測を行った結果を述べる．

7.2.1 実験概要

Uni-Fi でのファイルアクセスが使用に耐えうるものと検証するために，今回の実装において作成したシステムのテストを行った．実験では，Uni-Fi のメタデータリスト生成時間と，ファイルの検索から終了までの所要時を計測した．実験環境を表 7.1 に示す．実験ではプラットフォームとして Windows XP，JAVA SE6

表 7.2 実験環境

使用デバイス	VAIO Type-U
OS	Windows XP
アプリケーション実行環境	JAVA SE 6

のインストールされた Vaio Type-U を使用した．

7.2.2 実験内容

メタデータリスト生成時間の計測

メタデータリスト生成時間の計測では，一台の Vaio Type-U を用いて，デバイス内の mp3，jpg，mp4 形式のファイルのメタデータを全て XML ファイルに記述するまでの時間を計測した．デバイスの中には予め

mp3 形式の音楽ファイルと，jpg 形式の写真ファイルを 1000 個，また MP4 形式の動画ファイルを 200 個用意した．各ファイルの概要は以下の通りである．なお，ファイルサイズの小数点 2 桁以下は切り上げてある．

表 7.3　写真ファイルの概要

形式	jpg ファイル
平均ファイルサイズ	4.2MB
最大ファイルサイズ	7.1MB
最小ファイルサイズ	0.5MB

表 7.4　映像ファイルの概要

形式	MP4 ファイル
平均ファイルサイズ	13.9MB
最大ファイルサイズ	302.4MB
最小ファイルサイズ	0.1MB

表 7.5　音楽ファイルの概要

形式	mp3 ファイル
平均ファイルサイズ	3.3MB
最大ファイルサイズ	14.6MB
最小ファイルサイズ	1.5MB

ファイル検索所要時間の計測

ファイル検索所要時間の計測では，一台のデバイス上で任意のファイル名を指定し，そのファイルの検索結果を表示するまでの時間を計測した．なお，メタデータの情報は重複しておらず，XML ファイル内に記録されているファイルの総数は 6600 個である．

ファイル転送所要時間の計測

ファイル転送所要時間の計測では，3 台の VAIO Type-U がファイルアクセス Ready の状態で，2 つのデバイス間で 5MB のファイルを転送した際の所要時間を計測した．

7.2.3　実験結果

メタデータリスト生成所要時間計測

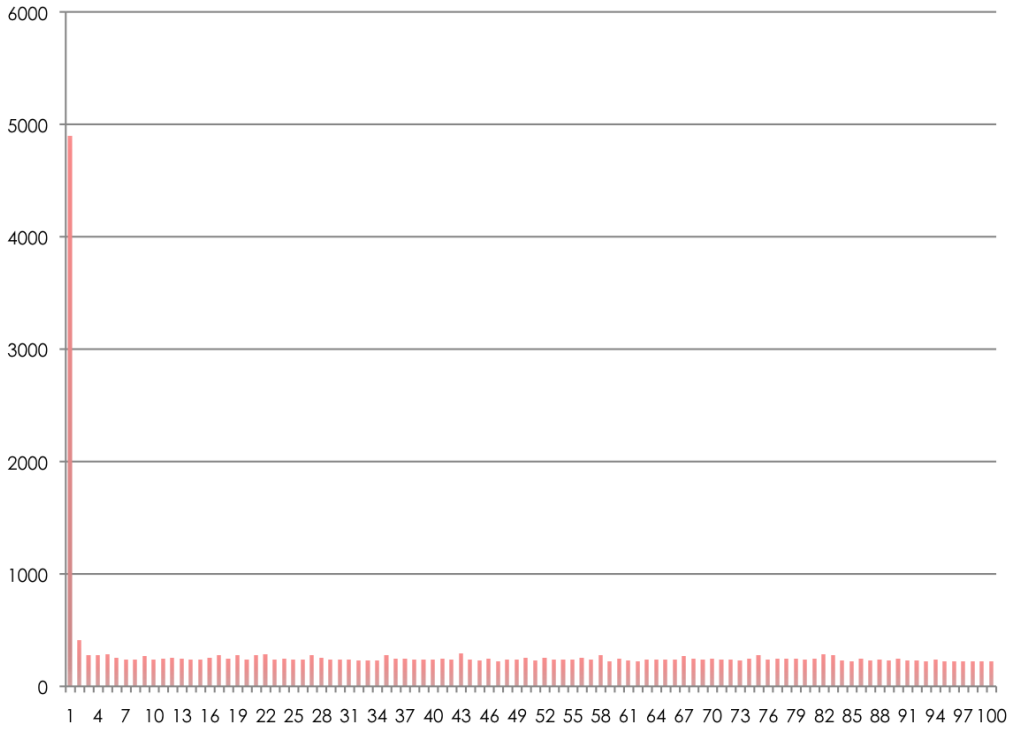

図 7.1　メタデータリスト生成時間計測結果グラフ

図 7.1 では，メタデータリストの生成を 100 回実行し，それぞれにかかった時間を計測した結果をグラフにした．X 軸の単位は回数であり，Y 軸の単位は所要時間（ミリ秒）である．Uni-Fi の起動直後の一回目は 4890 ミリ秒と，他の回に比べ時間がかかっているが，2 回目以降は 344 ミリ秒，3 回目以降はおおむね 250 ミリ秒前後で生成が行われている．なお，メタデータリスト生成を 100 回実行した際の平均値実行時間はは 285 ミリ秒であった．

ファイル検索所要時間計測

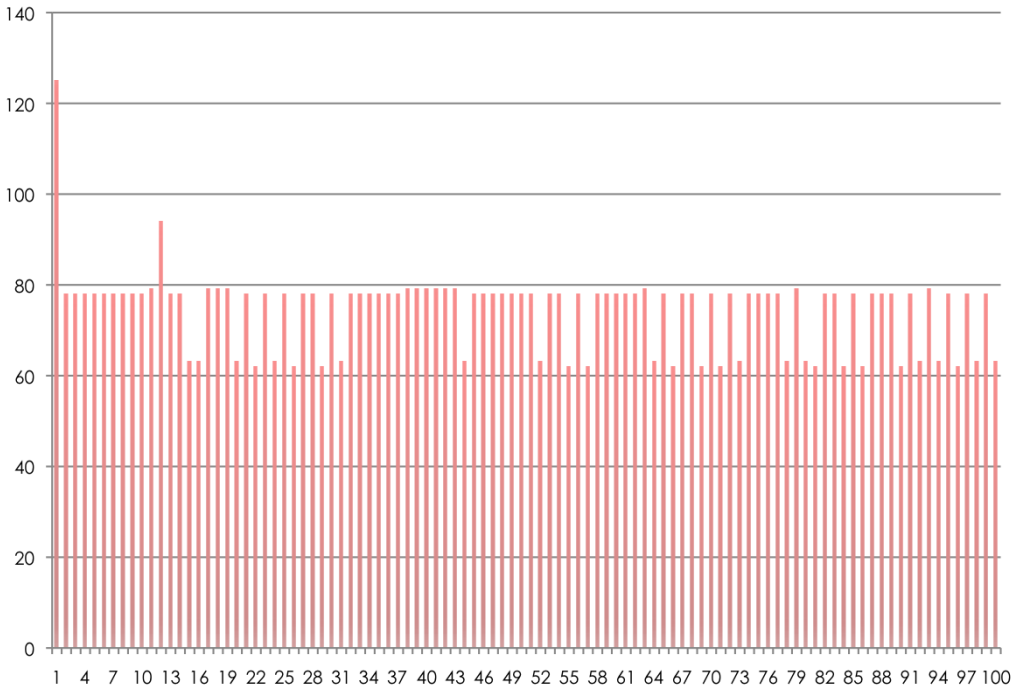

図 7.2　ファイル検索所要時間計測結果グラフ

図 7.2 はランダムに実在するファイルのファイル名を選択し，それをメタデータ解決モジュールに与え，メタデータを取り出した際にかかった時間を計測しグラフにした．X 軸の単位は回数であり，Y 軸の単位は所要時間（ミリ秒）である．一度目の検索で 125 ミリ秒かかっているが，それ以外の回では，おおむね 60 から 80 ミリ秒で推移している．統合メタデータリストからメタデータを取り出す操作にかかった平均実行時間は，100 回実行時で 74 ミリ秒であった．

ファイル転送所要時間の計測

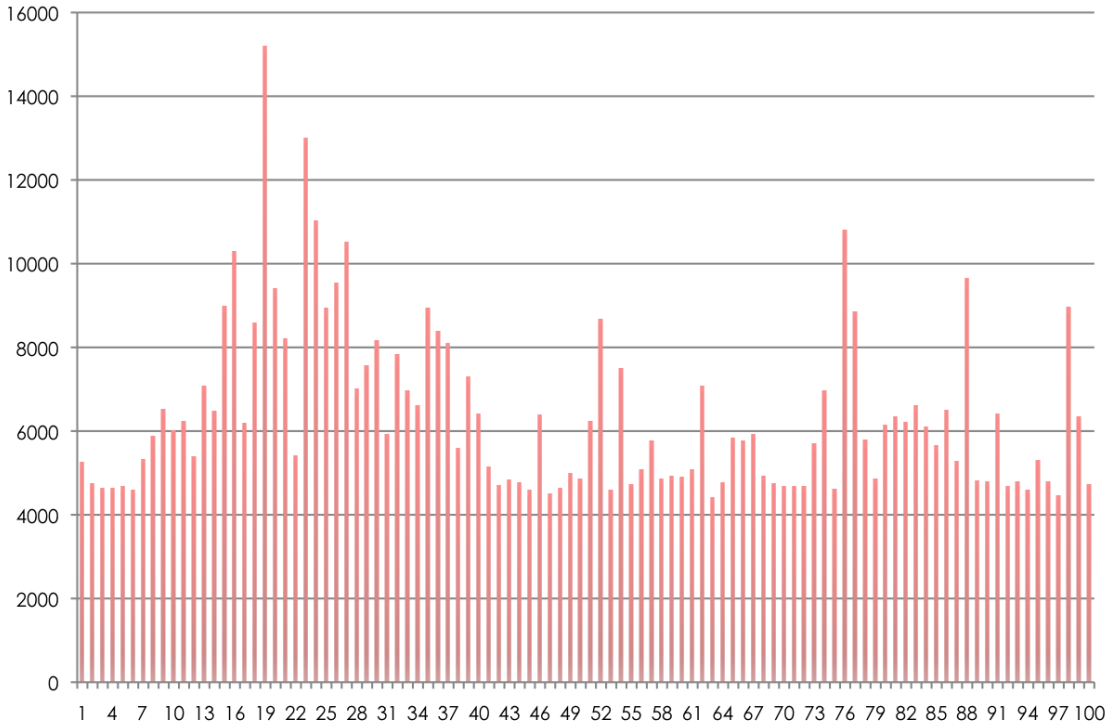

図 7.3　ファイル転送所要時間計測結果グラフ

図 7.3 は 3 台のデバイスがファイルアクセス Ready の状態で，2 つのデバイス間で 5MB のファイルを転送した際の所要時間を計測した結果を表すグラフである．X 軸の単位は回数であり，Y 軸の単位は所要時間（ミリ秒）である．所要時間の最大値は 15184 ミリ秒であり，最小値は 4408 ミリ秒となっている．また，100 実行時の平均所要時間は 6361 ミリ秒であった．

7.3　本章のまとめ

本章では，定性的評価として既存研究との比較を行い，Uni-Fi が新しい価値を提供していることを確認した．また定量的評価では，本研究でのアプローチ，設計，実装の結果，機能要件を満たし，かつ実用に耐えるシステムが構築された事を確認するために，メタデータリスト生成所要時間の計測，ファイル検索所要時間の計測，ファイル転送所要時間の計測の 3 つの実験を行った．結果として，ファイルの一覧表示や検索，転送などの実行速度が実際の使用に耐える可能性を示した．次章では，本論文のまとめと今後の課題について述べる．

第 8 章

結論

本論文では，スマートモバイルデバイスを対象としたファイルアクセス手法の構築と評価を行った．本章では，本論文のまとめを行った上で，今後の課題について述べる．

8.1 本論文のまとめ

本論文では，ネットワーク到達性を持つスマートモバイルデバイスを対象として，アドホックネットワーク上において複数デバイス間でファイルアクセスを実現するファイルアクセスシステム Uni-Fi を構築した．Uni-Fi は，ピュア P2P と，仮想単一ストレージ生成，ファイルメタデータのキャッシュにより，インフラストラクチャに依存しないファイルアクセス，複数デバイスに対する統一的なファイルアクセス手法，オフラインデバイス内のファイル検索を可能にした．これは，今後さらに増加してゆくコンテンツファイルと，普及してゆくスマートモバイルデバイスをモバイル環境において活用する上で，ユーザに対して負荷の低いファイルアクセスを実現する．

8.2 今後の課題

本研究では，今後，以下の研究課題に取り組む．

8.2.1 ファイル一覧機能の充実

Uni-Fi のファイル一覧機能では，ファイル名やファイルのメタデータを使用して，文字列ベースでの表示を行っている．しかし，Web サービス上でのコンテンツファイルの一覧は，写真ファイルであればサムネイルファイル，動画ファイルであれば短縮・縮小動画ファイルの表示などが行われている．これは，ユーザが具体的な目的を持たずにファイルを一覧している際に，ファイルの内容を確認するための手順を短縮している．したがって Uni-Fi においても，サムネイルや短縮・縮小動画などを一覧で表示できるように改善することが望まれる．

8.2.2 より多くのデバイスへの対応

今回の実装では，JAVA 言語を用いて Uni-Fi のプロトタイプ実装を行った．JAVA による実装で，携帯電話やネットブックといった，ユーザの身近に存在する多くのスマートモバイルデバイスに対応する事が可能だが，より多くのスマートモバイルデバイスに対応するために，他の言語での実装を行う必要がある．

8.2.3 ファイルアクセスのセキュリティ向上

Uni-Fi におけるファイルアクセスでは，ファイルの権限設定を行っていないため，基本的には全ての Uni-Fi を搭載するデバイスが同じ権限でファイルアクセスを行える．また，ネットワーク上のセキュリティーについても，IEEE802.11b/g の SSID による認証など，非常に限られている．モバイル環境では，ユーザがファイルアクセスを許可したくないデバイスも存在する可能性がある．そこで，通信の暗号化や，より柔軟なファイルアクセス権限の設定が必要となると考えられる．

8.2.4 よりスケーラブルなファイルアクセスの実現

Uni-Fi は，アドホック・ネットワーク構築に関する機能を IEEE802.11b/g のプロトコルに委ねている．よって，アドレス割り当てやマルチホップ時のルーティング制御など，未解決な問題が多い．これは，さらに多くのスマートモバイルデバイスを接続しファイルアクセスを実現する上で問題となる．そこで，これらの問題を吸収するためのモジュールの実装が求められる．

謝辞

本研究の機会を与えてくださり，絶えず丁寧なご指導を賜りました，慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝致します．また，慶應義塾大学環境情報学部准教授高汐一紀博士，慶應義塾大学環境情報学部専任講師中澤仁博士，慶應義塾大学政策・メディア研究科特別研究助教由良淳一博士，伊藤昌毅博士には，本論文の執筆にあたって，貴重なご助言を賜りましたことを感謝致します．

慶應義塾大学徳田研究室の諸先輩方には，折りに触れ貴重なご助言とご指導を頂きました．特に慶應義塾大学政策・メディア研究科修士課程伊藤友隆氏，生天目直哉氏には本論文を執筆する上で多くの議論の時間を割いて頂き，ご指導，叱咤激励を頂きました．ここに深い感謝の意を表します．また，日々の研究生活を支えて下さった慶應義塾政策・メディア研究科修士課程河田恭平氏，橋爪克弥氏，徳田義幸氏，山本純平氏，米澤祐紀氏，研究の日々を共に過ごした慶應義塾大学環境情報学部４年唐津豊氏，米川賢治氏，大日野舞氏，天野雅哉氏，多大な協力をして下さった瀧本拓哉氏，望月剣氏，西條晃平氏に心から感謝致します．

最後に，大学４年間に渡る生活を支え続けてくれた母，兄と，七夕祭・秋祭実行委員会の友人，その他多くの湘南藤沢キャンパスの友人に深く感謝し，謝辞と致します．

2010 年 2 月 12 日

井村　和博

参考文献

[1] Flickr. http://www.flickr.com/.

[2] Ieee802.11. http://www.ieee802.org/11/.

[3] Youtube. http://youtube.com/.

[4] Digital Living Network Alliance. Dlna. http://www.dlna.org/.

[5] Apple. ipodtouch 仕様. http://www.apple.com/jp/ipodtouch/.

[6] NTT docomo. Android 端末仕様. http://www.nttdocomo.co.jp/product/foma/pro/ht03a/spec.html/.

[7] UPnP FORUM. Upnp. http://www.upnp.org/.

[8] Andrea Goldsmith. *Wireless Communications*. 2005.

[9] Chih-Lin Hu, Wei-Shun Liao, and Yen-Ju Huang. Mobile media content sharing in upnp-based home network environment. *Journal of Information Science and Engineering 24*, 2008.

[10] Bluetooth SIG Inc. Bluetooth. http://www.bluetooth.com/.

[11] Bilhanan Silverjan, Antti Vekkeli, Tuure Vatiainen, and Jarmo Harju. Facilitating content exchange among homes, ad-hoc communities and mobile users. In *The 13th IEEE International Symposium on Consumer Electronics*, 2009.

[12] C.-K. Toh. *AD HOC MOBILE WIRELESS NETWORKS:PROTOCOLS AND SYSTEMS*. 2002.

[13] 河口信夫. cogma:動的ネットワーク環境における組み込み機器間の連携用ミドルウェア. 情報処理学会コンピュータシステム・シンポジウム, 2001.

[14] 近藤育雄. サマリの概念によるアドホックグループ上でのコンテンツ多目的集約機構. Technical report, 情報処理学会　データベース・システム研究会報告書, 2003.