慶應義塾大学学術情報リポジトリ
Keio Associated Repository of Academic resouces

Title	ふ*らいふ コミュニケーションと連動する花育成システムと推薦システムを利用したツイッタークライアントアプリケーション
Sub Title	Fu*life Twitter-client application system using growing flower system linked with communication and recommender system
Author	吉牟田, 陽平(Yoshimuta, Yohei) 稲蔭, 正彦(Inakage, Masahiko)
Publisher	慶應義塾大学大学院メディアデザイン研究科
Publication year	2010
Jtitle	
JaLC DOI	
Abstract	本論文では、ソーシャルネットワーク上でツイートを投稿することで花が育成され、かつ、ツイートの推薦を取得することが可能なソーシャルアプリケーション"ふ*らいふ"について述べる。ふ*らいふでは、ユーザーは、ツイートを投稿することで、蒔いた種に水や肥料を与えてお花を育て、育てた花をプレゼントする、コミュニケーションに連動する花育成システムを楽しむことが可能である。また、ユーザーは、自らの投稿をきっかけに、それに関連する投稿および投稿元である見知らぬ人を推薦するシステムを通して見知らぬ人と出会うことが可能である。ふ*らいふの目的は、コミュニケーションと連動する花育成システムと推薦システムによって、ツイッター上のコミュニケーションを促進することである。ふ*らいふのシステムフレームワークは、Client-Server ArchitectureやObject-Orientedアプローチ、CocoaにおけるMVC Design Patternを採用することで、システム開発の複雑さを軽減する。また、iOS SDKが提供するUI Kitフレームワークを拡張することで、iOS SDKの持つベストプラクティスを継承しつつ柔軟な描画能力を持つユーザーインターフェイスを可能としている。これらの特長は、直感的な操作性、独特の世界観を表現したインターフェイス、コミュニケーションへの報酬の付与、見知らぬ人の発見、システムの柔軟性および拡張性確保が可能であることを意味する。
Notes	修士学位論文. 2010年度メディアデザイン学 第122号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40001001-00002010-0122

慶應義塾大学学術情報リポジトリ(KOARA)に掲載されているコンテンツの著作権は、それぞれの著作者、学会または出版社/発行者に帰属し、その権利は著作権法によって保護されています。引用にあたっては、著作権法を遵守してご利用ください。

The copyrights of content available on the KeiO Associated Repository of Academic resources (KOARA) belong to the respective authors, academic societies, or publishers/issuers, and these rights are protected by the Japanese Copyright Act. When quoting the content, please follow the Japanese copyright act.

2010 年度　　修士論文

ふ＊らいふ
コミュニケーションと連動する花育成システムと推薦システムを利用したツイッタークライアントアプリケーション

吉牟田　陽平

慶應義塾大学大学院
メディアデザイン研究科

本論文は慶應義塾大学大学院メディアデザイン研究科に
修士 (メディアデザイン学) 授与の要件として提出した修士論文である。

吉牟田　陽平

指導教員：

稲蔭 正彦 教授　　(主指導教員)

砂原 秀樹 教授　　(副指導教員)

審査委員：

稲蔭 正彦 教授　　(主査)

砂原 秀樹 教授　　(副査)

徳久 悟 特別研究講師　(副査)

ふ＊らいふ
コミュニケーションと連動する花育成システムと推薦システムを利用したツイッタークライアントアプリケーション

内容梗概

本論文では、ソーシャルネットワーク上でツイートを投稿することで花が育成され、かつ、ツイートの推薦を取得することが 可能なソーシャルアプリケーション”ふ*らいふ”について述べる。ふ*らいふでは、ユーザーは、ツイートを投稿することで、蒔いた種に水や肥料を与えてお花を育て、育てた花をプレゼントする、コミュニケーションに連動する花育成システムを楽しむことが可能である。また、ユーザーは、自らの投稿をきっかけに、それに関連する投稿および投稿元である見知らぬ人を推薦するシステムを通して見知らぬ人と出会うことが可能である。ふ*らいふの目的は、コミュニケーションと連動する花育成システムと推薦システムによって、ツイッター上のコミュニケーションを促進することである。ふ＊らいふ のシステムフレームワークは、Client-Server Architecture や Object-Oriented アプローチ、Cocoa における MVC Design Pattern を採用することで、 システム開発の複雑さを軽減する。また、iOS SDK が提供する UI Kit フレームワークを拡張することで、iOS SDK の持つベストプラクティスを継承しつつ柔軟な描画能力を持つユーザーインターフェイスを可能としている。これらの特長は、直感的な操作性、独特の世界観を表現したインターフェイス、コミュニケーションへの報酬の付与、見知らぬ人の発見、システムの柔軟性および拡張性確保が可能であることを意味する。

キーワード

ソーシャルアプリケーション、ツイッター、推薦システム、iOS(iPhone) SDK、システムフレームワーク

慶應義塾大学大学院　メディアデザイン研究科

吉牟田　陽平

Fu*life
Twitter-Client Application System Using Growing Flower System Linked with Communication and Recommender System

Abstract

In this paper, we presented "Fu*life", social application system which allow users to grow flowers with posting tweets and to receive recommendation tweets. Fu*life incorporate the flower growing system linked with user communication which allow users to grow various flowers with posting tweets. And users can find unknown people with tweets recommendation system which learn users' tweets. Our purpose is to promote users' communication in twitter. Fu*life system framework is to apply Client-Server Architecture, Object-Oriented Approach and Cocoa Model-View-Controller Design Pattern to decrease developing complexity and cut down on system response time. And Fu*life extend UI Kit Framework in iOS SDK to make better User Interface. These features mean simple operation, expressive User Interface, reward to communication and, simplicity and augmentability of system development framework.

Keywords:

social application, twitter, recommender systems, iOS (iPhone) SDK, system frameworks

Graduate School of Media Design, Keio University

Yohei Yoshimuta

目　　次

第 1 章　研究目的 **1**

第 2 章　関連研究 **2**

2.1. 友人とのコミュニケーションを支援するシステム 2

2.2. 見知らぬ人と知り合えるシステム 3

第 3 章　アプローチ **4**

3.1. 概要 4

3.1.1　プロトタイプ 4

3.1.2　ターゲット 6

3.2. 友人とのコミュニケーション促進 6

3.3. 見知らぬ人とのコミュニケーション促進 8

第 4 章　システム **9**

4.1. システム構成 9

4.1.1　データ構造 12

4.2. コミュニケーションと連動する花育成システム 15

4.3. ユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステム 18

4.4. その他のコミュニケーション促進の機能 18

4.4.1　花の贈与に連動した音楽の贈与機能 18

4.4.2　花の贈与とその効果 20

4.4.3　ツイートへのワンタッチアクション「見たよ！」機能 . . . 21

4.4.4　ふ＊らいふへ誘う機能 21

第 5 章　実装 **24**
5.1. 基本実装 25
5.1.1　クライアントサーバーアーキテクチャ 25
5.1.2　MVC デザインパターン 27
5.1.3　モデル管理 28
5.1.4　画面構成/遷移 30
5.1.5　ユーザーインターフェイス 32
5.2. 花の育成と管理 40
5.2.1　花の育成モデルの構築 40
5.2.2　花の贈与モデルの構築 43
5.2.3　花の贈与に連動した音楽の贈与モデル 49
5.2.4　見たよ！モデルの構築 51
5.3. 推薦システムの構築 51

第 6 章　評価実験 **54**
6.1. 実験デザイン 54
6.2. 友人とのコミュニケーション促進 55
6.3. 見知らぬ人とのコミュニケーション促進 60

第 7 章　考察 **64**
7.1. 実験結果から見るふ*らいふの考察 64
7.2. 今後の展望 65

第 8 章　おわりに **69**

図　目　次

3.1　プロトタイプ外観 5

4.1　ふ＊らいふ外観 10
4.2　ふ＊らいふにおけるクライアントサーバーアーキテクチャ 11
4.3　花育成システムのユーザーフロー 1 16
4.4　花育成システムのユーザーフロー 2 17
4.5　推薦システムのユーザーフロー 19
4.6　見たよ！の付加とその効果 22
4.7　ふ＊らいふへ誘う機能のユーザーフロー 23

5.1　クライアントとツイッターサーバーを直接繋がない場合 26
5.2　MVC デザインパターンのふ＊らいふにおける適用図 28
5.3　ふ＊らいふの画面構成およびその遷移 33
5.4　UI コンポーネントの比較 (左は Ecofon[17]) 34
5.5　TimelineView、TweetView、DetailTweetView の関係 35
5.6　ArrowIndicatorButton の外観 38
5.7　メモリ使用量と生成時間への影響 39
5.8　花の贈与を通知するハチの吹き出し 44
5.9　ありがとうの新着件数の通知 48

6.1　ユーザー A の実験結果 1 56
6.2　ユーザー B の実験結果 1 57
6.3　ユーザー C の実験結果 1 58
6.4　ユーザー D の実験結果 1 59
6.5　ユーザー A の実験結果 2 60

6.6 ユーザー B の実験結果 2 61
6.7 ユーザー C の実験結果 2 62
6.8 ユーザー D の実験結果 2 63

7.1 Admob の導入例 67

表　目　次

5.1　クライアントサイドでのデータ管理 27
5.2　MVC デザインパターンのふ＊らいふにおける適用リスト 29
5.3　データモデルの管理に関わる関数群 1 30
5.4　データモデルの管理に関わる関数群 2 31

第1章
研 究 目 的

本研究では、コミュニケーションと連動する花育成システムと推薦システムによって、ツイッター上のコミュニケーションを促進することを研究目的とする。本研究では、コミュニケーションの相手を、友人と見知らぬ人という 2 タイプに区別する。したがって、研究目的は、友人とのコミュニケーションを支援するシステムと見知らぬ人と知り合えるシステムの 2 つから成る。

本研究では、これら 2 つのシステムを適用した、花育成型ツイッタークライアント ”ふ＊らいふ” の実装を通して研究目的を証明する。”ふ＊らいふ” は、iOS SDK[8] を用いた iOS アプリケーションとして Apple Store を通じてリリースが予定されている。

”ふ＊らいふ” は、2 つの理由からツイッタークライアント [15] として開発されている。1 つめの理由は、公開されているソーシャルネットワークサービスとして世界的に広く利用されており [12]、日本でも多くのユーザーが利用している点である。2 つめの理由は、公開されているソーシャルネットワークでありながら、ほぼすべての機能にアクセスすることを可能とする開発者への豊富な API の提供 [15] があるという理由からである。

第2章
関 連 研 究

2.1. 友人とのコミュニケーションを支援するシステム

ソーシャルネットワーク上の友人とのコミュニケーションを支援するシステムとして、まず挙げられるのが、2010年に発表された、地域侵攻型モンスター育成アプリ『Twimon（ツイモン）』[1]である。ツイモンは、ツイッタークライアントとしての基本的な機能は備えながらも、タイムライン上でツイートすることで、60種類以上の「モンスター」を育成でき、位置情報を使って実在の建物などを攻略ポイントに見立てた「拠点」を攻撃・制圧するシステムを備えたiPhoneアプリケーションである。ツイートと連動することでモンスターを成長させることができる点で興味深いが、友人のツイートへの返信を含む双方向からのツイートのやりとりをサポートしていない点で問題である。

また、2010年に発表された、ツイッターのつぶやきを読んで言葉を覚えながら成長していくペット育成ゲーム「meromero park for モバツイ」[2]は、ツイッターと連携して、フォロワーの輪を広げて行けることを特徴としたペット育成システムである。「meromero park for モバツイ」では、ユーザーの分身である「メロシー」のお部屋をフォロワー同士で行き来し、お部屋にある「メロシーノート」にコメントを書くことによってフォロワー同士のコミュニケーションを支援することを可能とする。もっとも、友人のツイートへの返信を含む双方向からのツイートのやりとりに関しては、視覚化するにとどまり、育成には関与しない点で、支援の範囲が狭く問題である。

本研究では、これらの関連研究を踏まえた上で、友人のツイートへの返信を含む双方向からのツイートのやりとりを含めた支援を可能とするシステムを提案する。

2.2. 見知らぬ人と知り合えるシステム

ソーシャルネットワーク上の見知らぬ人と知り合えるシステムに、2005 年に発表された、Event-Based Mobile Social Network Services[3] がある。Event-Based Mobile Social Network Services は、位置情報やイベントの参加状況の重なる頻度などを解析して、興味関心の近い見知らぬ人を推薦するシステムを備えている。位置情報等のユーザーの行動履歴を利用して、興味を持つに違いない見知らぬ人を発見することができる点で興味深いが、位置情報をもとにして出会える見知らぬ人の範囲にはかなりの制限がある点で問題である。

また、2010 年に発表された、ツイッターのソーシャル推薦エンジン [4] は、ユーザーがすでにフォローしているユーザーやそのユーザーの情報に基づいて、おすすめのフレンドを推薦するシステムである。ユーザーのフォロー関係を重視するシステムによって、知り合いだけれどフォローしていなかったユーザーを発見するのには効率的であるが、知り合いではないけれど興味関心の近い見知らぬ人を発見することまではサポートしていない点で問題である。

本研究では、ユーザーの発言内容を利用して、ユーザーが興味を持つに違いない見知らぬ人を推薦するシステムを提案する。

第3章
アプローチ

3.1. 概要

本研究の研究目的である、ソーシャルネットワーク上のコミュニケーションを促進するためには、友人と見知らぬ人とのコミュニケーションを区別する必要がある。なぜなら、見知らぬ人とコミュニケーションをとるためには、まず発見して出会わなければならないからです。したがって、コミュニケーションの相手が友達か見知らぬ人かで異なるアプローチを採用する必要がある。

3.1.1 プロトタイプ

本研究で提案する”ふ＊らいふ”を開発する以前に、その前身となるプロトタイプを開発した [5][6]。本プロトタイプは、コミュニケーションに連動した花視覚化システムによる、コミュニケーションの促進を目的に開発された。ユーザーは、ツイッターコミュニケーションとタッチアクションを利用してお花とのインタラクションを楽しむことができる。具体的には、自身とフォロワーの発言がお花の形で視覚化する機能、お花をドラッグして飾り付ける機能が搭載されている。また、本プロトタイプは、クライアントが当初 Flash(ActionScript3.0) により実装され、後に JavaScript により実装された PC 向けアプリケーションであった。

本プロトタイプでは、非公式にユーザーテストを実施した。その結果、コミュニケーションに連動した視覚化のみでは、コミュニケーションに対する報酬と感じられず、ユーザーがコミュニケーションを行う動機付けとして欠けていることがわかった。この結果は、コミュニケーションと連動して、花の視覚化からさら

に花の育成にまで踏み込んだシステムを検討するきっかけとなった。また、見知らぬ人を発見して出会うことを可能とする機能がなかったために、ふ＊らいふを通じて友人の輪を広げることができないことに対する不満があることがわかった。そこで、現在知り合っている友人だけでなく、新しく友人を開拓できるシステムを検討するきっかけになった。

その他に、デスクトップ型 PC クライアントは、ツイッタークライアントとしての利用上の敷居が高いことがわかった。そのため、”ふ＊らいふ”はモバイルアプリケーションとして iOS SDK を用いた iPhone クライアントとして開発されることになった。本研究を進めるにあたり、本プロトタイプの存在が大きく影響している (図 3.1)。

図 3.1　プロトタイプ外観

3.1.2 ターゲット

ふ＊らいふのメインターゲットは、20 代の女性である。女性が持つ親しい人とコミュニケーションをしたいという気持ちやかわいいものに惹かれる感受性を刺激し、ソーシャルネットワーク上のコミュニケーションのツールとしても情報を発見する実用的なツール、遊びのツールとしても利用可能なシステムを目指す。たとえば、女性とそのともだちが一緒に花の育成を楽しみ、コミュニケーションと連動する花育成システムを通して両者がコミュニケーションを行うことを想定する。誰か一人がふ＊らいふで花の育成を開始すると、そこに別のともだち達が集まり、自律的にコミュニケーションがはじまるような使い方を想定する。また、見知らぬ人とも推薦システムを通して出会い、コミュニケーションが生まれることを想定する。より楽しいコミュニケーション体験をユーザーに提供し、ユーザーのコミュニケーションを促進する。

3.2. 友人とのコミュニケーション促進

本研究では、友人とのコミュニケーションを促進させるために、コミュニケーションと連動する花育成システムを提案する。コミュニケーションと連動する花育成システムとは、ユーザーが友人とコミュニケーションをすることに対応して、花の生育状態が変動するシステムである。これによって、友人とのコミュニケーションを促進させることが可能となる。なぜなら、システムによって友人とのコミュニケーションを促進させるためには、ユーザーが友人とコミュニケーションをすることに対応して、ユーザーに報酬を提供することが必要だからである。

本研究で表現される、”コミュニケーションを促進する”とは、ユーザーと友人との双方向からのツイートのやりとりを促進させることを意味する。すなわち、ここで意味する”コミュニケーション”には、実際にユーザーが友人のツイートへ返信することだけでなく、友人からの返信を呼び起こすきっかけとしての返信以外のツイートを含めるべきである。したがって、ふ＊らいふが備えるコミュニケーションと連動する花育成システムは、ユーザーが行ったツイートの通常投稿

／返信／引用またはユーザーに届いた返信/引用と連動する花育成システムである。

また、”友人”とはユーザーとコミュニケーションをする関係を意味する。したがって、”友人”とは、ツイッターにおいてはユーザーがツイートを取得することを申請する（以下フォローする）ことで成立する間柄であるフレンドのことである。

ふ＊らいふが備える、ユーザーが行ったツイートの投稿／返信／引用またはユーザーに届いた返信/引用と連動する花育成システムとは、ツイートの投稿／返信／引用またはユーザーに届いた返信/引用をきっかけにして花が生育するシステムである。ユーザーはツイートを通常投稿／返信／引用のどれかをする、または返信/引用がユーザーに届くことによって、その投稿種類に応じた程度で花を育成することができる。ツイッターは、通常投稿／返信／引用の 3 種類の投稿を可能としている。通常投稿とは、返信と引用以外のすべての投稿である。返信とは、他者のツイートに応答するツイートである。引用とは、他者のツイートをフレンドに広めるツイートである。ツイートの投稿には種類があり、その種類に応じてツイートの投稿が持つ意味が異なるのであれば、それに連動した花の育成度合いにも変化をつけるのは自然である。本研究では、引用、返信、通常投稿の順番に、花の育成度合いを大きく設定している。なぜなら、ツイッターという一つのエコシステムとして俯瞰すると、ツイートの引用は、引用するユーザーを仲介して、出会うはずのなかったツイートが届けられるを意味するからである。これによって、引用するユーザーとフレンドとのコミュニケーションは生まれないかもしれないが、異なるフレンド同士のコミュニケーションが生まれるきっかけになる。一対一の最大 2 人が関わる返信も重要ではあるが、10 人、100 人、1000 人、10000 人、100000 人を超えるフレンド同士に影響を与える引用の方がより重要である。

3.3. 見知らぬ人とのコミュニケーション促進

本研究では、見知らぬ人とのコミュニケーションを促進させるために、ユーザーの発言をきっかけとしてユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステムを提案する。見知らぬ人とのコミュニケーションを促進するためには、システムに推薦された後も関係が継続されると思われる見知らぬ人を推薦するシステムでなければならない。システムに推薦された後も見知らぬ人との関係が継続されると思う場合とは、ユーザーが自身と同様の関心を見知らぬ人が共有していると思う場合である。そのためには、ユーザーの関心を特定する必要がある。ユーザーの発言は、自らの関心に沿った内容にしたがっていると思われる。ここで本研究では、ユーザーの関心とは、ユーザーの発言に含まれる単語から構成されると仮定する。したがって、ユーザーが自身と同様の関心を共有していると思う見知らぬ人とは、ユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人である。

また、ユーザーの関心は時間とともに変化するものであるから、前述の仮定から、ユーザーの発言とその発言に含まれる単語を含む発言を行った見知らぬ人の推薦との時間間隔は少なければ少ないほど、良い。なぜなら、時間の変化に伴ってユーザーの関心が変化する以上、以前のユーザーの発言と現在のユーザーの関心との齟齬は時間間隔に比例して大きくなるからである。したがって、ユーザーの発言とそれを根拠とする推薦との時間間隔を少なくするために、ユーザーの発言をきっかけに見知らぬ人を推薦するシステムを提案する。

そして、”見知らぬ人”とは、このシステムがなければ、ユーザーとコミュニケーションをしなかったであろう関係を意味する。したがって、”見知らぬ人”とは、ツイッターにおいてはフレンド以外のツイッターユーザー（以下一般ユーザー）のことである。

第4章
シ ス テ ム

4.1. システム構成

ふ＊らいふは、ツイッタークライアントとしての基本的な機能を備えつつも、コミュニケーションと連動する花育成システムとユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステムを持つ iOS アプリケーションである。ユーザーは、ツイッタークライアントとしての基本的な機能として、ツイートの通常投稿／返信／引用、一度に最大 200 件のツイートのダウンロードおよび閲覧、他のユーザーのフォロー、一度に最大 100 件のフォローフォロワーの閲覧ができる。また、ユーザーは、花育成システムとしての機能として、花の植え付け、花の育成、花の贈与ができる。そして、ユーザーは、推薦システムとしての機能として、推薦一回あたり 20 件の見知らぬユーザーの閲覧ができる。

ふ＊らいふのシステム構成は、クライアントサーバーアーキテクチャにより構成される。クライアントは iPhone である。サーバーはふ＊らいふ用のサーバーと、ツイッターサーバーの 2 つである。iPhone とツイッターサーバーはツイッター API を通してデータの送受信を行う。iPhone とふ＊らいふサーバーで配列を送受信する際には、JSON データに一度変換してから行われる。iPhone からふ＊らいふサーバーへは配列を JSON データにエンコードし、ふ＊らいふサーバーで受け取った JSON データをデコードする。ふ＊らいふサーバーから配列を JSON データにエンコードして、iPhone で JSON データをデコードして配列を取り出している (図 4.2)。

[1] メイン画面　　[2] タネ詳細画面

図 4.1　ふ＊らいふ外観

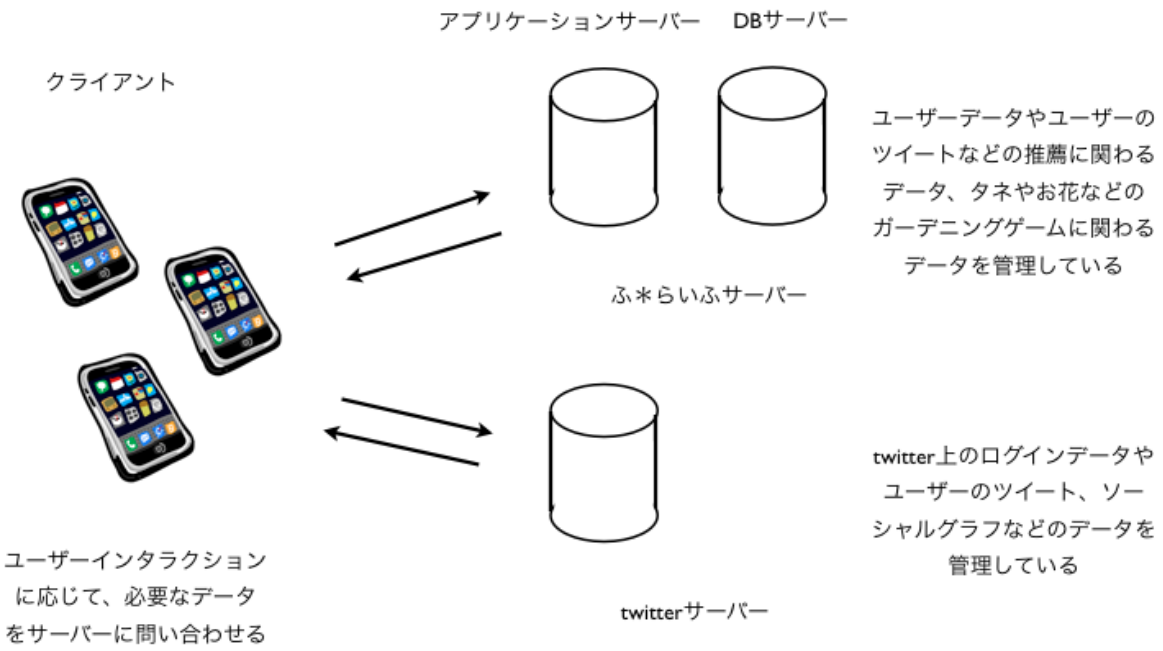

図 4.2 ふ＊らいふにおけるクライアントサーバーアーキテクチャ

4.1.1 データ構造

ふ＊らいふは、コミュニケーションと連動する花育成システムとユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステムを持つ、ツイッタークライアントである。そのため、ツイッターに関するデータモデルだけでなく、花の育成や推薦に関するデータモデル等を管理する必要がある。ふ＊らいふの管理するデータモデルには、[1]Flower、[2]Tane、[3]Tweet、[4]Bouquet、[5]FlowerHistory、[6]TaneCollection、[7]User、[8]BouquetMusic、[9]Mitayo、[10]Wordがある。

[1]Flower は、ユーザーの花の保有と生育状態を管理するデータモデルであり、id、t_user_id、tane_id、pos、status、is_bloom、is_used、is_deleted、created_at を属性として持つ。それぞれ、プライマリ ID、その花を持つユーザー ID、種の種類を示す種 ID、生育位置、生育状態、開花フラグ、贈与フラグ、削除フラグ、生育開始時間を意味する。例えば、62/39738761/12/0/5/1/1/0/2010-10-27 22:31:53 の形で記録される。

[2]Tane は、ユーザーの種の保有状態を管理するデータモデルであり、id、t_user_id、tane_id、get_type、is_used、is_deleted、created_at を属性として持つ。それぞれ、プライマリ ID、その花を持つユーザー ID、種の種類を示す種 ID、取得原因、植え付けフラグ、削除フラグ、所得開始時間を意味する。例えば、59/42382363/11/buy/1/0/2010-10-27 21:45:00 の形で記録される。

[3]Tweet は、ユーザーのツイートを管理するデータモデルであり、id、t_user_id、t_screen_name、t_status_id、t_profile_image_url、t_text、t_created_at、created_at、updated_at を属性として持つ。それぞれ、プライマリ ID、ツイッターのユーザー ID、スクリーン名、ステイタス ID、画像 URL、テキスト、ツイートの生成時間、管理開始時間、更新時間を意味する。例えば、151/115300477/sugm1/8256274760212480/http://a1.twimハロウィンパターン/0000-00-00 00:00:00/2010-11-27 05:30:24/0000-00-00 00:00:00の形で記録される。

[4]Bouquet は、ユーザーの花の贈与関係を管理するデータモデルであり、id、t_user_give_id、t_user_given_id、tane_ids、is_read、is_thank、is_action、is_deleted、created_at を属性として持つ。それぞれ、プライマリ ID、花を贈与するユーザー ID、花を受け取るユーザー ID、花の種類を示す種 ID、既読フラグ、ありがとう受信フラグ、ありがとうを原因とする種配布フラグ、削除フラグ、管理開始時間を意味する。例えば、124/39738761/115300477/9/1/0/0/0/2010-11-07 04:10:56 の形で記録される。

[5]FlowerHistory は、ユーザーの花の音楽 ID を管理するデータモデルであり、id、flower_id、t_user_id、m_user_id、is_deleted、created_at を属性として持つ。[4.4.1 花の贈与に連動する音楽の贈与] にて詳述される、贈与される花の音楽 ID を計算するのに利用される。それぞれの属性は、プライマリ ID、Flower のプライマリ ID、花の所有ユーザー ID、花の生育原因となったツイートを投稿したユーザー ID、削除フラグ、管理開始時間を意味する。例えば、1/186/115300477/115300477/0/2010-11-16 07:20:51 の形で記録される。

[6]TaneCollection は、花の種を管理するデータモデルであり、id、name、description、limit、grade を属性として持つ。それぞれ、プライマリ ID、名前、説明、開花までの生育期待値、通常の種か特別な種かのフラグを意味する。例えば、2/あさがお/花言葉は 「愛情」「平静」「愛情の絆」「結束」「短い愛」「明日もさわやかに」「はかない恋」だよ。/5/normal の形で記録される。

[7]User は、ふ∗らいふのユーザーを管理するデータモデルであり、id、t_user_id、t_screen_name、t_profile_image_url、t_token_key、t_token_secret、seed_id、garden_upper、is_delete、created_at を属性として持つ。それぞれ、プライマリ ID、ツイッターでのユーザー ID、ツイッターでのスクリーン名、ツイッターでのプロフィール画像の URL、ツイッター API にアクセスするための OAuth Token Key、ツイッター API にアクセスするための OAuth Token Secret、ユーザーに紐づいた種

ID、植えられる本数の限界値、削除フラグ、管理開始時間を意味する。例えば、33/115300477/sugm1/http://a1.twimg.com/profile_images/720563317/bluer.../115300477-WZldBmbsF7NF7KldkbtaXFxWU2oshrj5zkeOSvVA/zvAV7Ogdd6vqTasMUNTc0eWwK2xPtBuik 12-02 14:38:39 の形で記録される。

[8]BouquetMusic は、ユーザーの花の贈与に連動する音楽の贈与を管理するデータモデルであり、id、bouquet_id、t_user_give_id、music_ids、is_deleted、created_at を属性として持つ。それぞれ、プライマリ ID、Bouquet のプライマリ ID、音楽を贈与するユーザー ID、音楽 ID、削除フラグ、管理開始時間を意味する。例えば、1549/229/115300477/24,24,24,24,24,24,12,12,12/0/2010-11-28 16:02:37 の形で記録される。

[9]Mitayo は、ユーザーのツイートに対する「見たよ！」の付加関係を管理するデータモデルであり、id、t_user_give_id、t_status_id、t_screen_name、t_profile_image_url、t_text、t_created_at、created_at、updated_at、is_action、is_end、is_deleted を属性として持つ。[4.4.3 ツイートへのワンタッチアクション「見たよ！」機能] にて詳述される機能に利用される。それぞれ、プライマリ ID、「見たよ!」を付加したユーザーのユーザー ID、ツイートのステイタス ID、スクリーン名、プロフィール画像の URL、テキスト、ツイートの生成時間、管理開始時間、更新時間、「見たよ!」を原因とする種配布イベント発生フラグ、イベントの終了フラグ、削除フラグを意味する。例えば、242/115300477/27577200110731264/Asugm/http://a1.twimg.com/a/1294874399/images/d 花もらったヽ(￣д￣;)ノ=3=3=3/2011-01-19 13:04:51/2011-01-19 13:19:57/2011-01-19 13:19:57 の形で記録される。

[10]Word は、ユーザーのツイートとそこに含まれる単語を管理するデータモデルであり、id、tweet_id、t_user_id、t_screen_name、t_status_id、t_text、t_word、t_cnt_links、t_created_at、created_at、updated_at を属性として持つ。Word は、推薦システムで、ユーザーのツイートに含まれる単語のうち、そのユーザーにとって最も特徴的な単語を算出する際に利用される。それぞれ、プライマリ ID、

Tweet のプライマリ ID、ユーザー ID、スクリーン名、ツイートのステイタス ID、テキスト、単語、リンク数、ツイート生成時間、管理開始時間、更新時間を意味する。例えば、1539/605/237600112/Asugm/27218917176909825/今日は、あったかいココアを飲むべし♪———O（≧∇≦）O————♪/ココア/0/2011-01-18 13:21:10/2011-01-18 13:21:13/2011-01-18 13:21:13 の形で記録される。

4.2. コミュニケーションと連動する花育成システム

ふ＊らいふは、友人とのコミュニケーションを支援するために、コミュニケーションと連動する花育成システムを備えている。ユーザーはツイートの通常投稿／返信／引用のいずれかをする、または返信/引用がユーザーに届くことによって、花を育成することができる。(図 4.3、図 4.4) その際、ふ＊らいふは、ツイートの送受信と花育成システムとの連動を視覚的にユーザーが経験できるよう、工夫をこらしている。一般的に、花の育成原因としては、肥料あげ、雨などの天気を含めた天からの恵みなどが挙げられる。ふ＊らいふでは、通常の投稿に雨を、返信の投稿および受信に肥料あげを、そして引用の投稿および受信に天からの特別な恵みという効果をあてがっている。そのために、ふ＊らいふのツイートを表現するタイムラインの描画は、雲の形を連結した横型のタイムラインの形式で生成している。ユーザーは、雲の形をしたツイートを横にスライドすることでタイムラインを閲覧する。ユーザーがツイートを通常投稿すると、ユーザーの新規に投稿したツイートから雨が降り注いで花が育成される。これによって、ふ＊らいふは、コミュニケーションと連動する花育成システムをユーザーインターフェース面からも可能としている。

ユーザーは、他のふ＊らいふユーザーのユーザーページへ遷移することで、そのユーザーの花の育成状態を確認することができる。

図 4.3 花育成システムのユーザーフロー 1

図 4.4 花育成システムのユーザーフロー 2

4.3. ユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステム

ふ＊らいふは、見知らぬ人とのコミュニケーションを促進するために、ユーザーの発言をきっかけにユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステムを備えている。ユーザーはツイートをすることで、そのツイートに関係するツイートおよび一般ユーザーのプロフィールを一度に 20 件閲覧することができる。さらに、ユーザーは推薦されたツイートおよび一般ユーザーをきっかけにして、一般ユーザーをフォローすることができる (図 4.5)。

推薦システムでは、特定の単語の特徴量を算出するために、単語の出現数、その単語を含むツイートのリンク数を参照している。ユーザーが特定の単語を過去に多く利用していることは、その単語に関して強い関心があることを示すものである。また、ツイート中の特定の単語の出現とともにリンクを付加している場合には、特に強い関心を示していると解釈できる。

4.4. その他のコミュニケーション促進の機能

4.4.1 花の贈与に連動した音楽の贈与機能

ふ*らいふは、ユーザーが育てた花を組み合わせて友人に贈与することができる。この際、贈る花を構成する花の本数、その花の生成過程に関わったユーザー、それらの花の並び順という複合的な要素を通じて生成される音楽が花の贈与と一緒に付与される。これら生成手法は太田智美氏の ’おところりん’ を利用している。

おところりんでは、贈与される花に音楽 ID を設定する。その音楽 ID は、花の生育原因となるツイート (@, RT, それ以外) の発信者が登録時に選択した種 (以下、マイタネ) が持つ固有の花 ID を、関係する全発信者のそれを加算した総計で

図 4.5 推薦システムのユーザーフロー

ある。ユーザーが花を贈与する際の花の選択順とその花の音楽 ID をおところりん指定の表と対応させることで、それら花の本数分の mp3 ファイルを特定する。そして、それら mp3 ファイルを連結することで、花の贈与に連動した音楽を生成することを可能とする。

4.4.2 花の贈与とその効果

ふ＊らいふは、花を育成する前提として、花を植える機能を実装している。そのために、植える花の種をユーザーが受け取れる機会をいくつか実装する必要がある。ふ＊らいふでは、花の贈与の効果として植える花の種を確率的に受け取れる機能を実装している。確率は、基本的に、1 本あたり 3 回の贈与で 1 つ種が受け取れるものとして設定している。ただし、ユーザーの種が枯渇することは目的に反するので、ユーザーの現在所持している種が 0 個の場合は、救済措置として必ず 1 つは受け取れるようにしている。

また、ふ＊らいふは、植える本数を最初 3 本からユーザーの熟達に応じて 16 本まで植えられる仕様になっている。その際、花の贈与を促すために、花の贈与の効果として、贈与の回数に応じて植えられる本数が拡張する機能を実装している。具体的には、贈与の回数がある一定数に達すると植えられる本数が 1 本増加する。その一定数とは、現在植えられる本数-3 を n とした場合、2 の n 乗として計算する。例えば、植えられる本数が現在 3 本の場合、2 の 0 乗であるから 1 回贈与すれば、植えられる本数は 4 本に拡張する。植えられる本数が現在 4 本の場合、2 の 1 乗であるから 2 回贈与すれば、植えられる本数は 5 本に拡張する。植えられる本数が現在 10 本の場合、2 の 7 乗であるから 128 回贈与すれば、植えられる本数は 11 本に拡張する。

ふ＊らいふでは、花の贈与を受け取ったユーザーは、ありがとうを返すことができる機能を実装している。ふ＊らいふは、花の贈与の間接的な効果として、ありがとうを受け取った回数が 5 回を超えると、ありがとうを受け取った総数に応じた特別な種を受け取れる機能を実装している。ありがとうは、受け取った花の

贈与の確認画面からありがとう画像をタッチすることで送ることができる。ユーザーが、ありがとうを受け取った回数が5回を超えると、特別な種を受け取ることができる。さらに、ありがとうの総数が1回、10回、100回、1000回を超えるたびに、特別な種の種類が変化する。

4.4.3 ツイートへのワンタッチアクション「見たよ！」機能

ふ＊らいふでは、ユーザーは閲覧するツイートに対して、「見たよ！」を付け加えることができる。「見たよ！」はツイッターに存在しない、ふ＊らいふの独自拡張機能である。「見たよ！」は見たよ！画像へのタッチだけで、ツイートの発言元ユーザーへ読んだことを通知することができる機能である。この機能は、ユーザーとフレンドなどの他のユーザーとの簡単なコミュニケーション手段を提供することで、コミュニケーションを促進させるために実装している。

ふ＊らいふは、「見たよ！」機能を推奨するために、一つのツイートあたり「見たよ！」が5つたまると、そのツイートに「見たよ！」をタッチしたユーザーに対して、特別な種を配布する機能を実装している。「見たよ！」されたユーザーではなく、「見たよ！」を行ったユーザーにのみ、種を配布しているのは、「見たよ！」されたユーザーは「見たよ！」されること自体が報酬として有効であるからである。また、他の人が「見たよ！」をする可能性のある、意味のあるツイートの発掘を促す効果が期待されるからでもある (図 4.6)。

4.4.4 ふ＊らいふへ誘う機能

ふ＊らいふでは、花の育成や花の贈与、「見たよ！」機能等、ふ＊らいふ独自の機能を実装している。なかでも、花の贈与や「見たよ！」機能は、ふ＊らいふに登録しているユーザー同士でしか、コミュニケーションできない機能である。そこで、ユーザーのフレンドを効率的にふ＊らいふに登録させる誘導を施す必要がある。ふ＊らいふでは、ユーザーが他のユーザーのユーザーページへ遷移した際、ふ＊らいふに登録していない場合には、「ふ＊らいふに誘う」ボタンを描画して、

図 4.6 見たよ！付加とその効果

ふ＊らいふへ誘う機能を実装している。「ふ＊らいふに誘う」ボタンをタッチすると、書き込み画面へ遷移し、ツイートの Reply を通して他のユーザーに対してふ＊らいふのアプリ紹介ページへ誘導する。例えば、Asugm が BarackObama のユーザーページから「ふ＊らいふに誘う」ボタンをタッチすると、「@BarackObama Asugm さんが BarackObama さんをふ＊らいふへ誘っています。http://fulife.jp」を投稿する画面へ遷移する (図 4.7)。

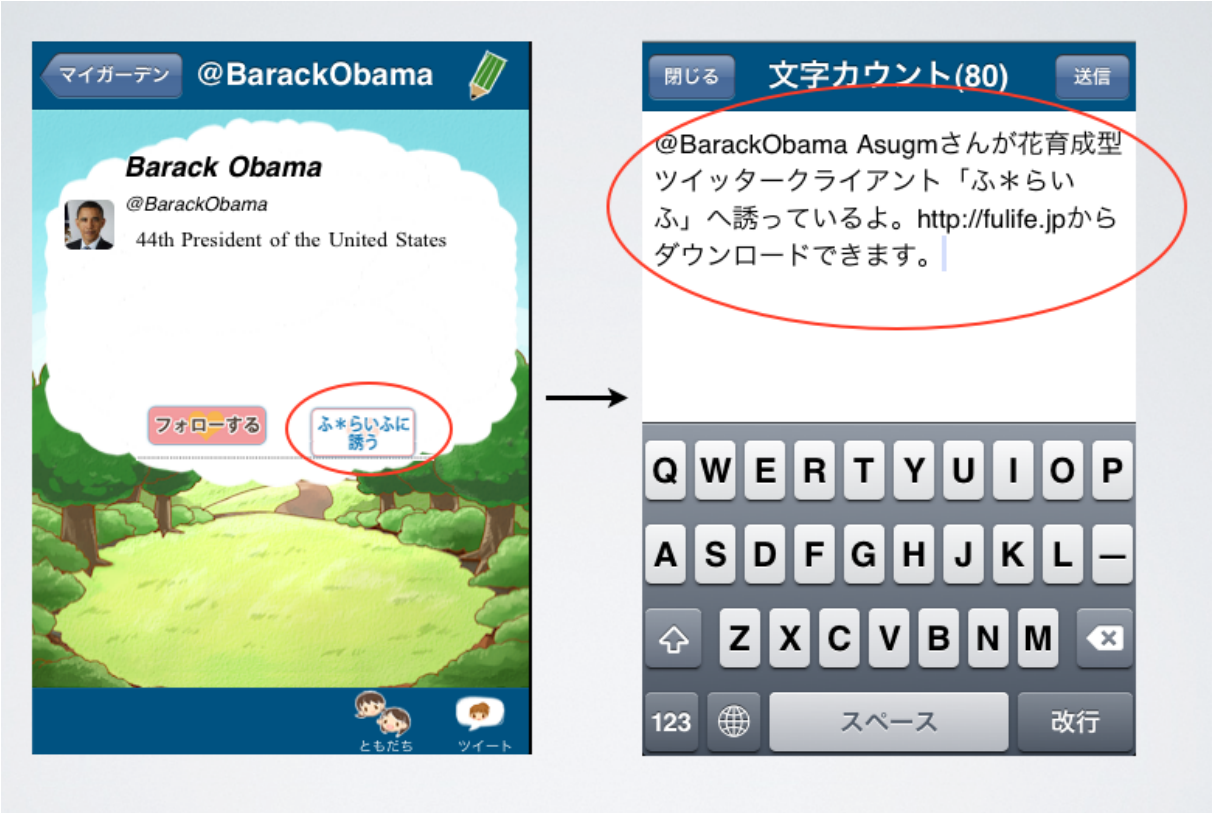

図 4.7　ふ＊らいふへ誘う機能のユーザーフロー

第5章
実　　装

ふ＊らいふは、データ処理等を集中的にこなすサーバーとそれを利用するクライアントとで処理の分散を行うクライアントサーバーアーキテクチャーを採用している。そして、特定の処理を記述したコンポーネントの再利用性やシステムの拡張性を担保することを可能とするMVCデザインパターンを用いて、システムを実装している。もっとも、ツイートの処理にツイッターAPIを利用することやアプリケーションにおけるレスポンスタイムを縮減したシームレスな操作感を実現するためのアーキテクチャ、デザインパターンを構築している。

そこで、[5.1.1 クライアントサーバーアーキテクチャ] では、アーキテクチャについて、[5.1.2 MVCデザインパターン] では、デザインパターンについて、説明する。また、それにともない、MVCデザインパターンにおける、データ管理を担当するModel、画面の描画/管理を担当するViewおよびModelとViewのインタラクションを管理するControllerについて、それぞれ、[5.1.3 モデル管理]、[5.1.4 画面構成、遷移] で説明する。

ふ＊らいふは、iOS SDKを利用して開発している。iOS SDKは、画面の描画/管理に関するフレームワークとしてUI Kit Frameworkを提供している。もっとも、UI Kit Frameworkとふ＊らいふが実現しようとする画面の描画/管理に関する機能との間には大きなギャップがある。そこで、[5.1.5 ユーザーインターフェイス] では、ふ＊らいふが提供するツイートデータの表現手法、画面遷移の手法に関して議論する。

ふ＊らいふは、コミュニケーションと連動した花育成システムを実装している。そのためには、コミュニケーションとどのように連動するのかを花の育成と管理のモデルとして実装する必要がある。そこで、[5.2 花の育成と管理] では、[5.2.1 花の育成モデルの構築]、[5.2.2 花の贈与モデルの構築]、[5.2.3 花の贈与に連動した音楽の贈与モデルの構築]、[5.2.4 見たよ！モデルの構築] に関して説明する。

ふ＊らいふは、ユーザーのツイートをきっかけにユーザーのツイートに含まれる単語と同じ単語を含むツイートを行った見知らぬ人を推薦するシステムを実装している。[5.3 推薦システムの構築] では、ユーザーのツイートの中から最も特徴的な単語を抽出する手法およびその結果取得された推薦結果の表示に関して説明する。

ふ＊らいふのクライアントは、iOS SDK を利用して Objective-C で記述されている。またサーバーは、PHP5.0 で記述されており、Yii Framework[16] を利用して構築している。データベースは MySQL を利用している。サーバーは CentOS release 5.5[10] をインストールして構築している。

5.1. 基本実装

5.1.1 クライアントサーバーアーキテクチャ

ふ＊らいふは、ユーザー同士がデータのやりとりを行い、そのデータをお互いに共有することを前提としたシステムであるため、ソーシャルアプリケーションに一般的なソフトウェアアーキテクチャの一つであるクライアントサーバーアーキテクチャーを採用している。

クライアントサーバーアーキテクチャーは、データ処理等を集中的にこなすサーバーとそれを利用するクライアントとで構成することによって、処理を分散することを可能としている。したがって、ふ＊らいふは、種やお花、音楽などのデータ管理をサーバーサイドに処理させ、画面の描画/管理をクライアントサイ

ドに処理させている。もっとも、ツイッター API を利用してツイッターに関するデータを取得している点およびレスポンスタイムを縮減したシームレスな操作感を実現する観点から、例外的な処理を施している。

ツイッターに関わるデータ処理の管理

ツイート単体やツイート郡から成る各種タイムライン、プロフィール情報、フォロワーやフォローなどのソーシャルグラフ情報などのデータを取得して、クライアントサイドの画面に描画して管理するまでには、大きく 2 つの経路がある。1 つは、クライアントからサーバーサイドにツイッターに関するデータをリクエストして、サーバーサイドからツイッター API をリクエスト、レスポンス結果をクライアントに返すという経路 (図 5.1)。もう 1 つは、クライアントから直接ツイッター API をリクエストしてレスポンス結果を取得する経路である。

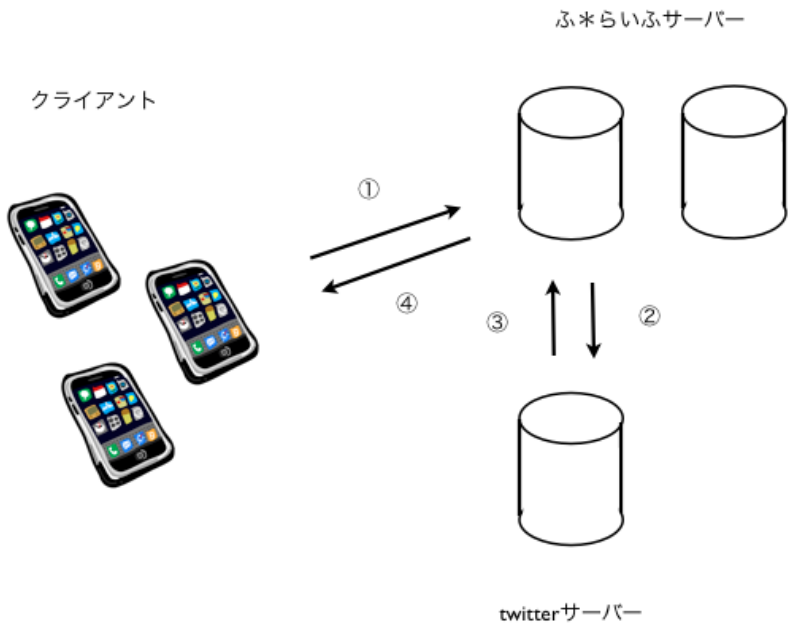

図 5.1 クライアントとツイッターサーバーを直接繋がない場合

ふ＊らいふでは、処理回数が少なくその分ツイッターデータのリクエストに対するレスポンスタイムがより少ない後者を採用している。それにともない、次節の内容とも関連して、クライアントサイドも同様にツイッターデータの管理を担当する。

更新頻度の少ないデータモデルの管理

更新頻度の少ないデータに関しては、サーバーサイドにリクエストにともなうネットワークの遅延を防ぐために、一度取得したデータをクライアント側で永続データ処理を施し、管理している (表 5.1)。更新頻度の少ないデータとは、ツイッター API における oauth 情報やプロフィール情報等のユーザーデータと各種のツイートデータである。これによってユーザーデータやツイートデータの管理を伴う処理時間は軽減されるが、サーバーサイドのデータがクライアントサイドに複製されることになるために、データの削除や更新等の変更を同期させるデータ管理処理が要求されることになる。

表 5.1 クライアントサイドでのデータ管理

クライアント	サーバー	データ内容
UserModel	ふ＊らいふサーバーの User モデル	ユーザーログイン情報
TweetModel	ツイッターサーバーの Tweet モデル	ツイート情報

5.1.2 MVC デザインパターン

ふ＊らいふは、iOS 上で動作するアプリケーションの実行環境を提供するオブジェクト指向フレームワーク'Cocoa'[9] を利用して開発されている。したがって、ふ＊らいふは、Cocoa が前提とするデザインパターンの一つである MVC デザインパターン [7][9] を採用している (図 5.2 は代表的なクラスを中心に紹介している)。詳細は表 5.2 に記載している。もっとも、ふ＊らいふは、データ管理をクライアントサイドだけでなく、サーバーサイドにも分散させているため、クライアント

からのサーバーサイドへのデータリクエストを専門で処理する'Manager' という概念を導入している。Model の詳細、および、Manager と Model の関係については、[5.1.3 モデル管理] で詳述する。

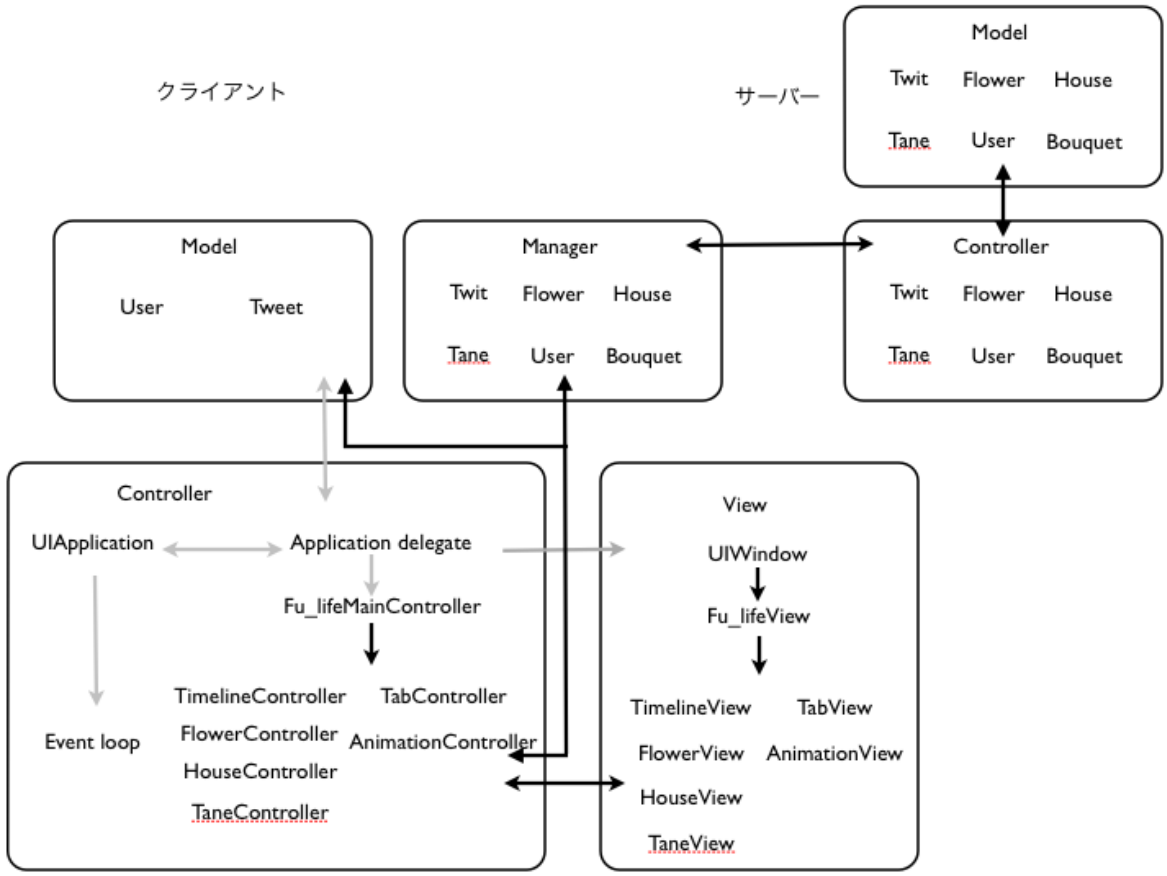

図 5.2 MVC デザインパターンのふ＊らいふにおける適用図

5.1.3 モデル管理

ふ＊らいふは、コミュニケーションと連動する花育成システムとユーザーの発言に含まれる単語と同じ単語を含む発言を行った見知らぬ人を推薦するシステムを実装している。したがって、ふ＊らいふは、ツイッターに関するデータだけで

表 5.2 MVC デザインパターンのふ＊らいふにおける適用リスト

Model(cliend/server)	Manager	Controller	View
cUserModel	UserManager	UserController	UserView
cTweetModel	TweetManager	HomeTimelineView	TimelineView
cTweetModel	TweetManager	RelatedTimelineView	TimelineView
cTweetModel	MitayoManager	MitayoTimelineView	TimelineView
cTweetModel	TweetManager	SocialGraphTimelineView	TimelineView
sTaneModel	TaneManager	TaneController	TaneView
sFlowerModel	FlowerManager	FlowerController	FlowerView
sFlowerHistoryModel	FlowerHistoryManager	FlowerController	
sBouquetModel	BouquetManager	BouquetController	BouquetView
sFFriendModel	FFriendtManager	FFriendController	TimelineView
sRecommendModel	RecommendManager	RecommendController	RecommendView

なく、そのデータと密接に関係する、推薦やガーデニングゲームに関するデータを取り扱う。表 5.3 および表 5.4 は、データ管理モデルが持つ関数を示している。

表 5.3 データモデルの管理に関わる関数群 1

class	server	client	client_manager
User	getUser	getUser	do_get
User	postUser	postUser	do_post
User	Twitter API	changeUser	do_change
Tweet	Twitter API	getTweet	do_get
Tweet	Twitter API	postTweet	do_post
Tweet	Twitter API	deleteTweet	do_delete
Tweet	Twitter API	favoriteTweet	do_favorite
Mitayo	getMitayo		do_get
Mitayo	postMitayo		do_post
Mitayo	getMitayoTimeline	getTweet	do_getMitayoTimeline
Mitayo	getMitayoHistory	getTweet	do_getMitayoHistory
Mitayo	checkEvent		do_checkEvent

5.1.4 画面構成/遷移

Cocoa は、アプリケーション内におけるユーザーインタラクションを通知するために、target-action デザインパターンを前提としている [7][9]。target-action デザインパターンとは、あらかじめ定められた方式に沿ったユーザーインタラクション (クリックやスクロール) をコントロール (ボタン等) が検知した際に、特定のオブジェクト (target) に対してメッセージ (action) を通知するパターンである。ふ＊らいふは、target-action デザインパターンを利用して、画面構成にコントロールを配置して、それに対するユーザーインタラクションを契機とした画面の遷移

表 5.4 データモデルの管理に関わる関数群 2

class	server	client	client_manager
Tane	getTanes		do_get
Tane	postTane		do_post
Tane	useTane		do_use
Tane	checkIsDaily		do_checkIsDaily
Tane	getTaneCollectionSheet		do_getTaneCollectionSheet
Flower	getFlowers		do_get
Flower	postFlower		do_post
Flower	postFlower		do_grow
Flower	postFlower		do_isPlant
Flower	deleteFlowers		do_delete
FlowerHistory	postFlowerHistory		do_post
Word	getRecommend		do_get
Word	postUsersTimeline		do_postUsersTimeline
FFriend	getFFriend		do_get
Bouquet	getBouquets		do_get
Bouquet	posBouquet		do_post
Bouquet	getBouquetHistory		do_getHistory
Bouquet	getBouquetThanks		do_getBouquetThanks
Bouquet	checkThanksEvent		do_checkThanksEvent
Bouquet	getBouquetMusicPath		do_getBouquetMusicPath
Bouquet	updateIsRead		do_updateIsRead

を実現している。また、推薦システムや花育成システムが持つ一定のルールを満たした場合のイベント処理にともない、画面が切り替わる (図 5.3)。

5.1.5 ユーザーインターフェイス

ふ＊らいふのユーザーインターフェイスは、iOS SDK の提供する UI Kit Framework を基礎として構築されている。iOS 上で動作する多くのツイッタークライアントアプリケーションが描画するタイムラインやツイートの表現は、UI Kit Framework が提供するテーブル描画を管理する UITableView やテーブルの中の一画面単位を描画し管理する UITableViewCell などのコンポーネントを利用している。また、複数の画面を管理するアプリケーションの多くが、UIScrollView を利用してスクロールする画面や UITabBar を利用してタブで切り替わる画面遷移を管理している。ふ＊らいふは、ツイッターを利用したコミュニケーションを可能とするユーザービリティを確保しつつも、女性をターゲットとした特定の世界観を持った花育成システムという特長を持つアプリケーションである。したがって、それにともない、大きく 2 つのコンポーネントを UI Kit Framework の関係するコンポーネントに機能追加する形で作成した。タイムラインとツイートを描画し管理するコンポーネント、タブバーを描画しタブのクリックによる画面遷移を管理するコンポーネントである (図 5.4)。

タイムライン/ツイートの描画コンポーネント

ふ＊らいふは、ツイッターを使ったコミュニケーションを楽しむことができるツイッタークライアントアプリケーションである。ツイッターの提供するサービスのうち、最も本質的な要素は、ユーザーが以前にフォローした、フレンドのプロフィール画像とテキストコンテンツから構成されたツイートとそれらの一連のツイートから構成されたタイムラインである。したがって、ふ＊らいふは、それらツイートおよびタイムラインとユーザーとのインタラクションを可能とするものでなければならない。
ふ＊らいふは、ツイートのタイムラインを流れていく雲に見立てて、表現し、ユー

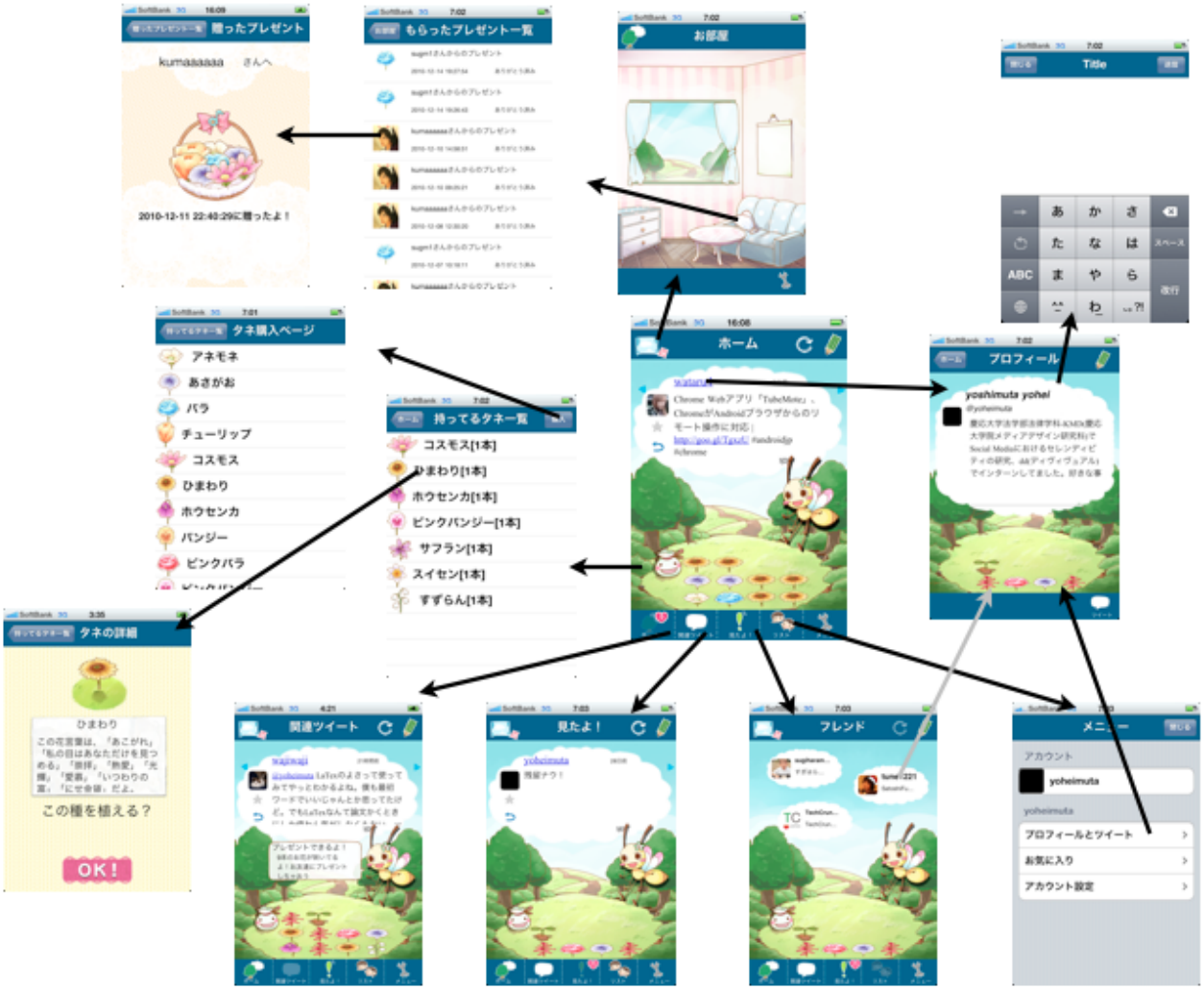

図 5.3　ふ＊らいふの画面構成およびその遷移

図 5.4　UI コンポーネントの比較 (左は Ecofon[17])

ザーインタラクションを可能なものとしている。ユーザーは、雲の描かれた画面部位を横にスクロールすることで、雲の上に描かれたツイート (以下、通常ツイート) を閲覧する。そして、ツイート上に存在する詳細切り替えボタンをクリックすることで、返信/RT/見たよ/削除/お気に入りボタンを描画した詳細ツイートに切り替わる。そこで、ユーザーはツイートに対して、返信したり削除したりお気に入りに登録したりすることができる。タイムラインを管理するコンポーネントを TimelineView、通常ツイートを管理するコンポーネントを TweetView、詳細ツイートを管理するコンポーネントを DetailTweetView として、実装している (図 5.5)。

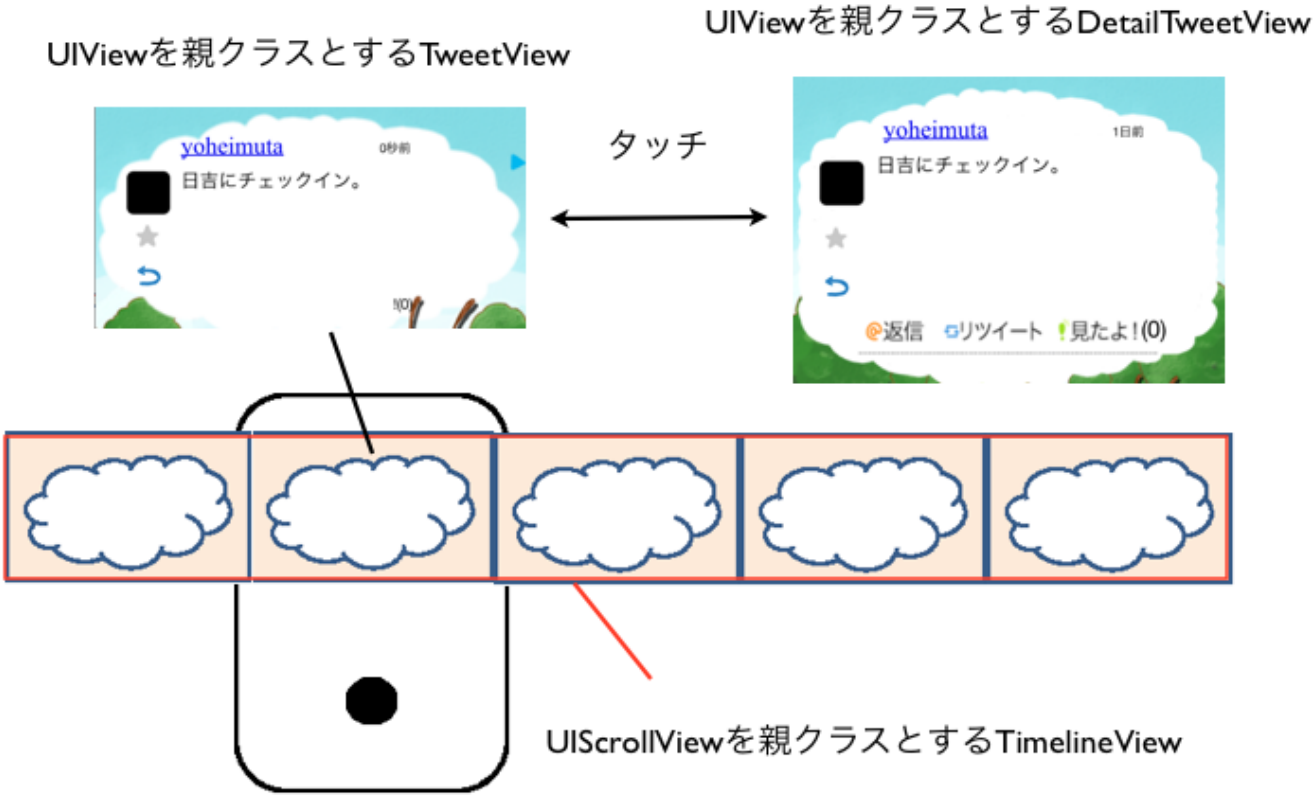

図 5.5 TimelineView、TweetView、DetailTweetView の関係

TimelineView は、UIScrollView の上に TweetView を重畳させて、横に引き延ばすことでタイムラインを表現する。また、TimelineView は、TweetView と DetailTweetView の切り替え処理およびそれに伴う画面構成の変更を管理する。TweetView は、UIView の上に、リンク付きのユーザースクリーン名 (UIWebView)、雲の画像 (UIImageView)、ユーザープロフィール画像 (UIImageView)、URL や@username の自動リンク機能を付加されたテキストコンテンツ (UIWebView)、投稿時間 (UILabel)、お気に入り状況 (UIImageView)、詳細ツイート切り替えボタン (UIButton)、見たよ個数 (UIImageView/UILabel) を重畳して、通常ツイートを表現している。TweetView の詳細ツイート切り替えボタンをクリックすることで切り替わる DetailTweetView は、TweetView に重畳している要素に加え、返信ボタン (UIButton)、RT ボタン (UIButton)、見たよボタン (UIButton)、削除ボタン (UIButton)、お気に入りボタン (UIButton) を重畳して、詳細ツイート画面を表現している。

このようにして作成されたコンポーネントには、当初、大きく 4 つの問題が存在した。TimelineView のスクロール速度が遅いこと、TimelineView に自動スクロール機能がないことでユーザビリティを損なうこと、URL や@付きの特定ユーザーへのリンクがないこと、およびメモリ使用量が多く生成までの時間が長いことの 4 つである。

[問題 1:TimelineView のスクロール速度の遅延]
TimelineView のスクロール速度が遅いのは、サーバーとのネットワーク処理に原因があることがわかった。そこで、2 つの方策を採用した。1 つ目は、サーバーとのネットワーク処理を同期処理から非同期処理に変更することである。ツイートの描画処理の中でサーバーとのネットワーク処理を行っていたのは、ツイッターサーバーからユーザーのプロフィール画像を取得する部分とふ * らいふサーバーから見たよされた回数を取得する部分である。これには、それぞれ、ImageLoader および ASIHTTPRequest というライブラリを利用して実現した。2 つ目は、これらネットワーク処理を TimelineView の生成時に一度にすべてのツイートに関して実

行するのではなく、ユーザーがTimelineViewをスクロールして当該TweetViewを閲覧するタイミングでその都度、ネットワーク処理を実行することである。

[問題2:TimelineViewの操作のわかりにくさ]
TimelineViewは、一度のスクロールの度に、一つのツイートを閲覧することができるタイムライン画面である。200件のTweetViewを重畳させている場合、200件目以降を閲覧したいという場合、1件目から200件目の画面に向かうまでにユーザーは200回スクロール動作をしなくてはならない。また、一見ガーデニングゲームにおける雲が描かれた画面にしか見えないTimelineViewにおいて、スクロールできる性質のものであることをヒントを示すことなく、直感的に気づくことはまれである。したがって、ふ∗らいふは、ユーザビリティの観点から、スクロール可能であることを示しクリックすることで最初と最後に自動スクロールするArrowIndicatorButtonを実装している(図5.6)。そして、これを実装するためには、最初と最後という概念が必要であり、ページを特定する必要があるため、TimelineViewにページング判定処理を実装している。

[問題3:TweetViewおよびDetailTweetViewにおけるリンクの生成]
TweetViewおよびDetailTweetViewは、テキスト整形処理の中で、正規表現を用いて、URLや@付きの特定ユーザーを示す文字列に対して動的にリンクを生成している(ソースコード??)。

[問題4:メモリ使用量の縮減と生成時間の短縮]
TimelineViewの実装は、当初生成したTweetViewが画面から外れた場合もすべて保持する仕様となっていた。これは実装が容易であるという利点があるが、反面、生成するTweetViewの数が増えるにしたがってメモリ使用量が増大する上に、一度に多くのツイートを含むタイムラインを生成する場合には多くの時間がかかってしまう点で問題である。そこで、ユーザーのスクロール動作に際して一度に目につく、真ん中のTweetViewとその前後のTweetViewの合計3つだけを生成して、それらをデータだけ書き換えて再利用する仕様により実装している。

図 5.6　ArrowIndicatorButton の外観

このTweetViewの再利用化とは、左右のスクロールアニメーションに必要な、画面中央と左右のTweetViewのみを生成しておき、常に生成された3個のTweetViewを使い回すことである。これによって、例えば、200個のTweetViewをそれぞれ生成してTimelineViewに重畳させるのではなく、3個のTweetViewのみを生成してユーザーのスクロール操作に応じて動的にデータを書き換えるので、ロードにかかる時間は短縮され、また使用メモリを縮減させることが可能となる（図5.7）。

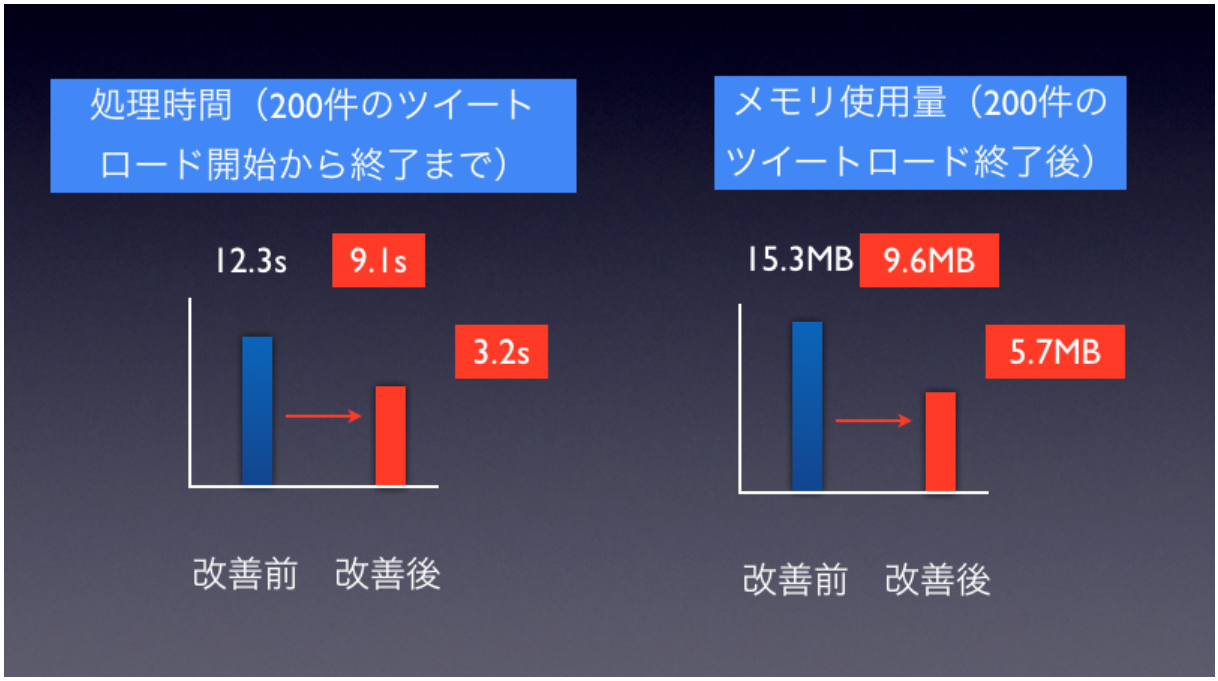

図 5.7　メモリ使用量と生成時間への影響

タブバーの描画/タブクリックによる画面遷移を管理するコンポーネント

ふ＊らいふは、タイムラインを閲覧する画面に5つのタブボタンを配置し、それによってそれぞれの機能を持った画面に切り替わる仕組みを提供している。多くのアプリケーションが実装しているタブによる画面切り替えの機能は、iOS SDKが提供しているUITabBarViewControllerを利用している。しかし、UITabBarViewControllerは、タブの画像に白黒処理およびクリックに伴う色彩の反転処理を施

す仕様になっており、タブの画像をカラーで表現することができない。すなわち、UITabBarViewController のそれらに関するプロパティやメソッドは private であり、UITabBarViewController に対する、オブジェクト指向プログラミングにおけるサブクラス化や Cocoa におけるサブクラス化の代わりにもなるカテゴリやプロトコルという処理による拡張というアプローチ [7][9] では解決できない。そこで、ふ＊らいふは、独自に UITabBarViewController と同様の機能を持ち、かつ、タブの画像をカラーで表現することができるコンポーネント'TabBarView' を実装している。

TabBarView は、複数のタブボタンで構成されたタブバーを描画し、タブボタンがクリックされると当該ボタンにアニメーションによる画像効果を施し、それに対応した画面に切り替える機能を持ったコンポーネントである

5.2. 花の育成と管理

5.2.1 花の育成モデルの構築

ふ＊らいふは、コミュニケーションと連動する花育成システムである。ユーザーはツイートを通常投稿／返信／引用のどれかをする、または返信/引用がユーザーに届くことによって、花を育成することができる。それを可能とするためには、[1] 花が植えられているか、[2] 植えられているとして育つか、[3] 育たないとしてなにを提示するか、[4] 育つとしてどの程度育つか、[5] 植えられていないとしてそもそも植えることができるか、[6] 植えられるとしてなにを植えることができるのかというロジックおよびそのロジックツリーを含む、花の育成モデルが必要である。

[1] 花が植えられているか

ユーザーがツイートを投稿または自分あてのツイートを受け取ることによって、ふ＊らいふは、[1] 花が植えられているかを確認する。なぜなら、花が植えられて

いないのに育成するのは不自然であるからである。もっとも、ユーザーがツイートを投稿しても花が育成しないのは、ふ＊らいふの目的に適うものではない。したがって、ふ＊らいふでは、ユーザーがログインするたびに一日あたり一つの花の種を贈与する機能を実装している。これによって、いつでもユーザーが花を植えることができるようにしている。

[2] 植えられているとして育つか

次に、[2] 植えられているとして育つかを確認する。ふ＊らいふの扱う花には、生育状態が存在する。生育状態とその進行は、花の種類によって異なり、通常の種と特別な種の場合で異なる。int 型の整数値として扱われる生育状態は、通常の種の場合で、1 が種を、2 が芽を、3 が苗を、4 がつぼみ、5 が咲いた花を意味する。特別の種の場合だと、通常の種の生育状態の倍数として、2 が種を、4 が芽を、6 が苗を、8 がつぼみ、10 が咲いた花を意味する (以下、表記上、断りがない限り生育状態は通常のものとして扱う)。すでに生育状態が 5 の場合は、咲ききった花として育たない。なぜなら、咲いた花がさらに育つというのは不自然だからである。もっとも、[1] の場合と同様、ユーザーがツイートを投稿しても花が育成しないのは、ふ＊らいふの目的に適うものではない。この点に関しては、[3] 育たないとしてなにを提示するかにつながる。

[3] 育たないとしてなにを提示するか

これ以上育たないという場合には、[3] 育たないとしてなにを提示するかを確認する。ふ＊らいふは、ツイッター上のコミュニケーションを促進することを目的とするアプリケーションである。したがって、咲いた花は、育成できない代わりに、フレンドに贈与できる対象として扱われる。ふ＊らいふでは、植えられている花の生育状態が 5 の場合には、フレンドへの花の贈与を促すアイコンの描画を開始する。

[4] 育つとしてどの程度育つか

植えられている花の生育状態が 5 未満の場合、[4] 育つとしてどの程度育つかを確認する。ふ＊らいふでは、ツイートの種類に応じて、通常投稿では雨のアニメーション、返信では少女の肥料あげのアニメーション、引用では虹のアニメーションが描画されて、花が生長する。その際、引用、返信、通常投稿の順番に、花の育成度合いが大きく設定されている。具体的には、引用がプラス 3、返信がプラス 2、通常投稿がプラス 1 である。問題は、複数本の花が植えられている場合である。育つとしてどのように育つかを検討する必要がある。当初、複数本の花が植えられている場合も単本の場合と同様、一律に花の育成度合いがプラスされる方法を採用した。なぜなら、雨や虹などの天候をきっかけとした生育原因において、複数本のうち、一部だけが生育するというの不自然だからである。もっとも、非公式のユーザーテストを通して、一度のツイートの投稿または受信で多くの花が育成されてしまうことは、コミュニケーションに連動する花育成システムを通して、ユーザーへの達成感を提供することを困難にすることが判明した。それを受けて、ふ＊らいふは、複数本の花が植えられている場合は単本の場合とは異なり、一本単位で育成度合いを増加させる方法を採用している。その場合に対象となる花の選び方は、植えた時間の降順でその花の生育状態が 5 未満である花を優先して選択する。また、それに付随して、引用が複数件届いた場合など、生育可能なプラス幅が一本あたりの生育可能余地を上回る場合に対応する必要が生じる。この場合、一本あたりの生育可能余地に生育可能なプラス幅を充当した上で、残った生育可能なプラス幅を次の花へ割り当てている。

[5] 植えられていないとしてそもそも植えることができるか

また、花が植えられていない場合は当然花は生育しないため、ユーザーは植えようとするわけであるが、[5] 植えられていないとしてそもそも植えることができるかを確認する必要がある。ふ＊らいふでは、植えられる花の本数に上限が設定されている。最初は 3 本のみ植えられ、花の贈与を繰り返すことによって、最大 16 本まで植えることができる。これは、前述の [3] 育たないとしてなにを提示するかに関係するが、育たない花を滞留させることなく循環させるためである。

すなわち、ユーザーは、新しい種を植えるために、花の贈与を繰り返し、贈与によって植えられる本数が広がることによって、また新しい種を植えるという好循環を生み出す。花の贈与モデルの構築に関しては、次節で詳述する。

[6] 植えられるとしてなにを植えることができるのか

最後に、花が植えられていなくて、かつ花を植えることができるという場合、[6] 植えられるとしてなにを植えることができるのかを確認する。

5.2.2 花の贈与モデルの構築

ふ＊らいふは、コミュニケーションと連動する花育成システムである。ユーザーは、育成した花を友人に贈与することができる。それを可能とするためには、[1] 咲いた花があるか、[2] 贈る花をどのような順序で選択したか、[3] 贈れる相手が存在するか、[4] 贈る音楽をどのように合成するか、[5] 贈与の効果として贈った花から種を受け取れるか、[6] 贈与の効果として植える本数が増えるか、[7] 贈与の効果としてありがとうを受け取るか、[9] ありがとうを受け取ったとして、その効果として特別な種を植え取れるか、というロジックおよびそのロジックツリーを含む、花の贈与モデルが必要である。

[1] 咲いた花があるか

まず、ふ＊らいふは、アプリの起動直後にサーバーからユーザーが現在植えている花の種類、生育状態、植えた時間を取得することで画面上に花を描画している。その際、生育状態が 5 の花があるかを確認することで、[1] 咲いた花があるかを確認している。咲いた花がある場合には、ユーザーが花を贈与できることを通知してアクションを引き起こすトリガーを用意する必要がある。したがって、ふ＊らいふは、咲いた花の存在を確認した場合、「花をプレゼントすることができるよ。プレゼントしたい場合は、この吹き出しをタッチしてね」という、ハチの吹き出しを描画する。ユーザーがハチの吹き出しをタッチすることで贈る花の選択画面に移行する (図 5.8)。

図 5.8　花の贈与を通知するハチの吹き出し

[2] 贈る花をどのような順序で選択したか

次に、ユーザーに贈る花を選択してもらう必要がある。咲いた花は複数本ある場合があるので、贈る花は複数本選択することができる。ふ＊らいふは、花の贈与に連動した音楽の贈与機能を実装している。贈与される音楽の生成は、贈与される花の生育過程と本数、および贈与時に選択された花の順序に影響される。したがって、贈与される花の本数とその順序を取得するために、[2] 贈る花をどのような順序で選択したかを確認している。具体的には、縦に並んだテーブル上に、贈る花の候補を描画し、それにタッチすることでチェックを描画して、ユーザーに贈る花を選択させている。

[3] 贈れる相手が存在するか

ふ＊らいふでは、育成した花を友人に贈与することができる。「友人」とは、本来ユーザーのフレンドのことである。花の贈与機能は、ツイートの送受信などとは異なる、ふ＊らいふの独自拡張機能である。したがって、花の贈与の際の「友人」とは、ユーザーのフレンドであり、かつ、ふ＊らいふに登録しているユーザーである。そこで、[3] 贈れる相手が存在するかを確認する必要がある。この場合、当初構築したシステムでは、まず、ツイッター API によりユーザーのフレンドを取得する。次にその取得したフレンドのユーザー ID のみを JSON 形式に変換した後、サーバーで配列にデコードした上で、データベースにふ＊らいふユーザーかどうか照会する。サーバーはふ＊らいふユーザーと確認できたユーザー ID だけを返す。これによって、☒贈れる相手が存在するかを確認していた。もっとも、当初構築したこのような方法では、処理に大幅な時間がかかることが判明した。キャッシュを利用すれば、時間が短縮するというのは、自明のことであったが、問題はいつの時点のデータをキャッシュとしてそれをどのタイミングで更新するかである。それによって、キャッシュのデータと実データとの齟齬が少なくなるかが左右されるからである。思案の末、ユーザーがフレンドのリストを閲覧する際のデータをキャッシュする方法を採用した。なぜなら、ユーザーがふ＊らいふのみをツイッタークライアントとして利用しているという前提に立つかぎりにおいて、ユーザーが最新のフレンドを確認する手段はフレンドリストの閲覧時である

以上、その際のデータを常に最新のデータとしてキャッシュすることで、仮に実データとは異なっていても、ユーザーの認識している実データとの齟齬はなくなるからである。また、ユーザーがフレンドを追加削除した際には、これをキャッシュデータに反映させる処理も追加した。これによって、実データとの齟齬が実質的にない形で、ツイッター API によりユーザーのフレンドを取得する過程を省くことにより、処理時間が短縮されている。

[4] 贈る音楽をどのように合成するか

このようにして、贈る花とその選択順序、贈る相手が決定した後、ふ＊らいふは、贈る花と贈る音楽の確認ページに移行する。ユーザーは、音符画像をタッチすることで贈る音楽を確認することができる。その際、[4] 贈る音楽をどのように合成するかが問題となる。この点に関しては、[5.2.3 花の贈与に連動した音楽の贈与モデル] にて詳述する。

[5] 贈与の効果として贈った花から種を受け取れるか

ふ＊らいふでは、花の贈与の効果として植えられる花の種を確率的に受け取れる機能を実装している。そこで、[5] 贈与の効果として贈った花から種を受け取れるかを確認する。まず、ユーザーが現在所持している種の個数を取得し、その個数が 1 個以上あるかを判定する。0 個であれば、贈った花と同じ種類の種を配布する。1 個以上ある場合は、乱数を取得し、その乱数を 3 で割りきれるかの判定を個数分実行する。3 で割り切れる場合は、その花と同じ種類の種を配布する。

[6] 贈与の効果として植える本数が増えるか

ふ＊らいふでは、花の贈与の効果として、贈与の回数に応じて植えられる本数が拡張する機能を実装している。そこで、[6] 贈与の効果として植える本数が増えるかを確認する。クライアントは、ユーザー ID をサーバーに送信する。サーバーは、そのユーザー ID をもとにそのユーザーの贈与回数と現在の植えられる本数を取得する。そして、ユーザーの贈与回数が、現在植えられる本数-3 を n とした

場合の 2 の n 乗を超えるかを判定する。超える場合は、ユーザーの現在植えられる本数を更新し、サーバーからクライアントへ拡張を可能とするフラグをオンにして返す。クライアントは返り値を確認し、拡張が可能な場合には、ユーザーへ植えられる本数が拡張したことを通知する。

[7] 贈与の効果としてありがとうを受け取ったか

ふ＊らいふでは、花の贈与を受け取ったユーザーは、ありがとうを返すことができる。そこで、[7] 贈与の効果としてありがとうを受け取ったかを確認する。ふ＊らいふは、起動直後に、ユーザー ID をもとに受け取ったありがとうを更新する。最新のありがとうを確認すると、ふ＊らいふは、家のアイコンにありがとうの新着件数を描画してユーザーに通知する (図 5.9)。

[8] ありがとうを受け取ったとして、その効果として特別な種を取得できるか

そして、ふ＊らいふは、花の贈与の間接的な効果として、ありがとうを受け取った回数が 5 回を超えると、ありがとうを受け取った総数に応じた特別な種を受け取れる機能を実装している。そこで、[8] ありがとうを受け取ったとして、その効果として特別な種を取得できるかを確認する。クライアントは、起動直後に、ユーザー ID をサーバーに送信する。サーバーは、そのユーザー ID をもとにそのユーザーのありがとう総数と効果の原因となっていないありがとう（ありがとうに伴う効果フラグがオフ）の個数を取得する。そして、効果の原因となっていないありがとうの個数が 5 個を超えるかを判定する。超える場合は、そのありがとうに伴う効果フラグをオンにする。そして、ありがとう総数が 1,10,100,1000 を超えるかを判定し、それに対応した特別な種の ID を特定してクライアントに返す。クライアントは返り値を確認し、特別な種の ID がある場合には、ユーザーへ特別な種を受け取れることを通知する。

図 5.9　ありがとうの新着件数の通知

5.2.3 花の贈与に連動した音楽の贈与モデル

’おところりん’による、花の贈与に連動した音楽を生成するためには、[1] 花の音楽 ID がいくつか、[2] 音楽の合成をどのようにして実行するか、[3] 音楽の再生をどのようにして行うか、[4] 保存された音楽をどのように管理するかを確認する必要がある。

[1] 花の音楽 ID がいくつか

まず、[1] 花の音楽 ID がいくつかを確認する必要がある。そのために、ふ＊らいふでは、花が生育する際に、その生育原因となるツイート (@, RT, それ以外) の発信者のマイタネの固有の花 ID を取得して、その花の音楽 ID に加算して保存する処理を行っている。

次に、ふ＊らいふでは、贈る花とその選択順序、贈る相手が決定した後、贈る花と贈る音楽の確認ページに移行する。そして、音楽を確認する際のユーザーのストレスになるようなタイムラグを最小化するために、処理を音楽合成と音楽再生の 2 つに分けている。

[2] 音楽の合成をどのようにして実行するか

まず、音楽合成は、ユーザーが贈る花を選択して贈る相手を選択するページに移行した際に、実行を開始する。その際、[2] 音楽の合成をどのようにして実行するかを確認する必要がある。具体的には、贈る花のそれぞれの持つ音楽 ID、贈る花の選択順序、その総数、ユーザー ID をサーバーに送信する。そして、サーバーでは、’おところりん’の対応表にしたがって、対応した音楽を特定して保存先パスを明らかにする。その保存先パスにある mp3 を合成する。合成は、ひとつひとつの mp3 ファイルを文字列として書き出した上で、ID3 tag を削除し、それら文字列を連結することで実装している。その合成した mp3 を「ユーザー ID.mp3」の形でサーバーに保存し、その保存先パスをクライアントに返す。

もっとも、ひとつひとつの mp3 ファイルを文字列として書き出した上で、ID3 tag を削除し、それら文字列を連結する方式は単純な方式ではあるが、再生時間が間違って認識されるという問題があった。mp3 を扱う多くの音楽プレイヤーは、mp3 の再生時間を最初のビットレートから判断する。これは固定ビットレート (CBR) では適切な結果が得られるが、可変ビットレート (VBR) でエンコードされた mp3 では適切な結果が得られないため、mp3 の再生時間の把握には演奏時間等が記録された ID3 tag が用いられる。そのため、ID3 tag を削除し演奏部分のみを連結する当初の実装方法では、ID3 tag が書き変わらないために iTunes が誤作動を起こすことがわかる。これに対して、ふ＊らいふでは vbrfix[11] というオープンソースソフトウェアを利用することで解決している。すなわち、ID3 tag を削除することなく mp3 を文字列として連結した上で、vbrfix を用いて、連結された mp3 にある ID3 tag から連結された mp3 の内容に即した ID3 tag に修正している。これによって、再生時間が間違って認識されるという問題が解決されている。

[3] 音楽の再生をどのようにして行うか

ユーザーは、贈る花と音楽の確認ページで、音符画像をタッチすることで贈る音楽を確認することができる。ふ＊らいふでは、音楽合成により保存先パスを受け取るまでは、アクティビティインジケーターを描画することでロード中を表示し、受け取り次第、音符画像を描画している。ここでは、[3] 音楽の再生をどのようにして行うかを確認する。ふ＊らいふでは、ストリーミング再生によりこれを実行している。mp3 の受け渡しには時間がかかり、ユーザーのストレスになるようなタイムラグが発生するためである。

[4] 保存された音楽をどのように管理するか

最後に、[4] 保存された音楽をどのように管理するかを確認する。保存される音楽ファイルは、6 秒から 30 秒ほどの短い音楽が刻まれた mp3 ファイルではあり、また、保存を常に「ユーザー ID.mp3」の形で 1 ユーザー 1 ファイルにして上書

き処理を施している。とはいえ、それをそのまま保存していては、ユーザー数の増加に伴ってサーバーの容量が足りなくなってしまう。また、ユーザーが贈与される音楽の確認をした後、保存する必要はない。なぜなら、贈与を受け取る際にも、贈与される花の構成を受け取る以上、音楽合成処理をその都度施すことは可能だからである。したがって、ふ＊らいふでは、音楽を贈与したかまたは取り消した時点で音楽を削除している。

5.2.4 見たよ！モデルの構築

ふ＊らいふは、ワンタッチでツイートの発言元ユーザーへ読んだことを通知することができる機能「見たよ！」を実装している。クライアントはユーザーが「見たよ！」をタッチすると、そのツイートに関する情報をユーザー ID とともにサーバーへ送信し、ユーザーの「見たよ！」したツイートとして保存する。そして、「見たよ！」が一つのツイートにつき 5 つ集まると、そのツイートの発言元ユーザーに対して特別な種を配布する機能を実装している。ユーザーが「見たよ！」タブをタップすると、ふ＊らいふは、「見たよ！」したツイートか「見たよ！」されたツイートを選択する画面を描画する。ユーザーが「見たよ！」されたツイートを選択する場合、それらのツイートに対して何個の「見たよ！」が付加されているかが問題となる。クライアントはユーザー ID をふ＊らいふサーバーへ送信する。ふ＊らいふサーバーは、ユーザー ID をもとに「見たよ！」されたツイートを取得する。「見たよ！」されたツイートひとつひとつに何個の「見たよ！」が付加されているかを計算する。その個数が 5 個以上で、かつそのツイートを原因とする配布が未成立の場合、サーバーはそのツイートを原因とする配布のフラグをオフにして保存した上で、配布が可能であることを返り値で示す。クライアントは、配布が可能な場合、そのユーザーへ特別な種を配布する。

5.3. 推薦システムの構築

ふ＊らいふは、見知らぬ人とのコミュニケーションを促進させるために、ユーザーの発言をきっかけにユーザーの発言に含まれる単語と同じ単語を含む発言を

行った見知らぬ人を推薦するシステムを備えている。ユーザーは、ツイートを投稿することで、そこに含まれる特徴的な単語に関連したタイムラインを取得することができる。その際、[1] ユーザーのツイートは成功したか、[2] ユーザーのツイートのうち、どの単語が最も特徴的か、[3] ユーザーは推薦結果をどのように閲覧可能か、が問題となる。

[1] ユーザーのツイートは成功したか

ふ＊らいふの推薦システムは、ユーザーがツイートを投稿することをきっかけに、そこに含まれる特徴的な単語に関連したツイートを提示するシステムである。したがって、クライアントはユーザーの投稿を API を通して送信した後、API の返り値から、[1] ユーザーのツイートは成功したかを確認する必要がある。

[2] ユーザーのツイートのうち、どの単語が最も特徴的か

次に、[2] ユーザーのツイートのうち、どの単語が最も特徴的かを判定する必要がある。そこで、ユーザーのツイートに含まれる特徴的な単語を特定するために、ユーザーの過去のツイートログを集計する必要がある。したがって、ふ＊らいふは、ユーザーがふ*らいふに登録時にプロフィール情報を取得し、ツイートの閲覧時にそれが本人のものである場合はそのツイートをサーバーに送信し、推薦システムの学習に利用している。また、ユーザーがふ＊らいふを使用して投稿したテキストもその都度、サーバーに送信され、推薦システムの学習に利用される。

そして、ユーザーがツイートを投稿すると、クライアントはツイートを構成する配列を JSON 形式に変換してサーバーに POST 通信を行う。サーバーは、JSON 形式のデータを配列に変換し、ツイート本文を取得して、日本語形態素解析ツール Mecab[13] を利用して、名詞を抽出する。抽出された名詞の特徴度を計算して、最も特徴度の高い名詞を、そのツイートの中でそのユーザーの関心にとって最も特 徴的な単語であると判定する。名詞の特徴度とは、その名詞に対するユーザー

の 関心の強さを意味しており、以下のような TF-IDF 法 [14] を用いて算出している。 TF で単語の出現頻度を算出し、IDF で単語の逆出現頻度を算出している。また、特徴度が同じ場合には、その単語と一緒に出現したリンクの数を比較することで優劣をつけている。

TFIDF(w) = TF(w) × IDF(w)
TF(w) = 当該テキストに占める単語 w の頻度
IDF(w) = log[(ふ∗らいふに登録している全ユーザー数)/(w を一度でも使ったユーザー数)]

このようにして特定された特徴的なキーワードは、Twitter search API を利用して、類似ツイート結果を取得している。独自の検索アルゴリズムを実装してはいないが、本システムにおける推薦システムの構築において重要な点は、ユーザーのツイートを構成する雑多なキーワードの中から、ユーザーが頻繁につぶやき、かつ、他のユーザーがあまりつぶやいていないキーワードを特定することである。したがって、本システムは検索アルゴリズムに Twitter search を利用している。

[3] ユーザーは推薦結果をどのように閲覧可能か

クライアントは、サーバーから推薦結果を取得すると、それらを描画する必要がある。そこで、[3] ユーザーは推薦結果をどのように閲覧可能かが問題となる。ふ∗らいふは、推薦結果を取得すると、それをユーザーに通知するためにハチの吹き出しに推薦結果の第一番目を埋め込んで描画する。ユーザーをそれをタッチすると、より詳細にすべての推薦結果ツイートを閲覧することができる。

第6章
評 価 実 験

6.1. 実験デザイン

調査対象は、意図的なサンプリング手法により抽出された20代の女性4名に対して行った。サンプリングに際しては、ツイッターの使用歴とツイッターの使用目的を基準に行った。抽出された4名は、それぞれ、ツイッター使用頻度が高くツイッターを情報収集目的で利用しているユーザー (以下ユーザー A)、ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザー (以下ユーザー B)、ツイッター使用頻度が低くツイッターを情報収集目的で利用しているユーザー (以下ユーザー C)、ツイッター使用頻度が低くツイッターをコミュニケーション目的で利用しているユーザー (以下ユーザー D) である。被験者には初めに簡単な”ふ＊らいふ”の説明を行い、平日の特定の一日を使ってプレイしてもらった。平日の特定の一日に限定しているのは、周期的な同様の生活スタイルの中で利用することで、過去の平日の特定の一日に”ふ＊らいふ”以外のツイッタークライアントのプレイ中のユーザーログという比較対象との整合性を担保するためである。

友人とのコミュニケーションが促進したかを確認するために、ユーザーに一日自由にプレイしてもらった上で、プレイ中のユーザーログとして、ツイートの通常投稿／返信／引用の回数を集計する。これを過去1ヶ月間を対象とした、実験時と同様の曜日のユーザーのツイートの通常投稿／返信／引用の回数の平均と比較することで、ふ＊らいふが友人とのコミュニケーションを促進したといえるか証明する（以下実験結果1）。もっとも、被験者はふ＊らいふを初めて使った

ユーザーであるため、慣れるために投稿した可能性は配慮しなければならない。そのため、実験では、プレイ前の説明の際に実際に体験してもらい、そのような可能性を限りなく少なくするよう努めた。

また、見知らぬ人とのコミュニケーションを促進したかを確認するために、ユーザーに10回ツイートの推薦を閲覧してもらった。そして、ユーザーログとして、1回のツイートの推薦毎に、フォロー／お気に入り／ツイートの更なる閲覧の回数を集計する（以下実験結果2)。ツイートの更なる閲覧とは、推薦システムにより推薦されたツイートからそのツイートの発言元ユーザーのリンクを辿り、同じユーザーの別のツイートを探索する行為と定義する。これはフォローやお気に入りと同様、そのツイートに対する関心の現れとして評価できるからである。

6.2. 友人とのコミュニケーション促進

友人とのコミュニケーションが促進したかを確認するために、ふ＊らいふをプレイ中のツイートの通常投稿／返信／引用の回数を平均化した過去のログデータとそれぞれ比較する。

まず、ツイッター使用頻度が高くツイッターを情報収集目的で利用しているユーザーは、過去のログデータとして、1日あたり平均1回の通常投稿、0回の返信、4回の引用を投稿していた。対して、ふ＊らいふを1日プレイしてもらった間に2回の通常投稿、1回の返信、10回の引用を投稿した (図6.1)。

つぎに、ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザーは、過去のログデータとして、1日あたり平均1回の通常投稿、8回の返信、0回の引用を投稿していた。対して、ふ＊らいふを1日プレイしてもらった間に1回の通常投稿、10回の返信、1回の引用を投稿した (図6.2)。

ツイッター使用頻度が低くツイッターを情報収集目的で利用しているユー

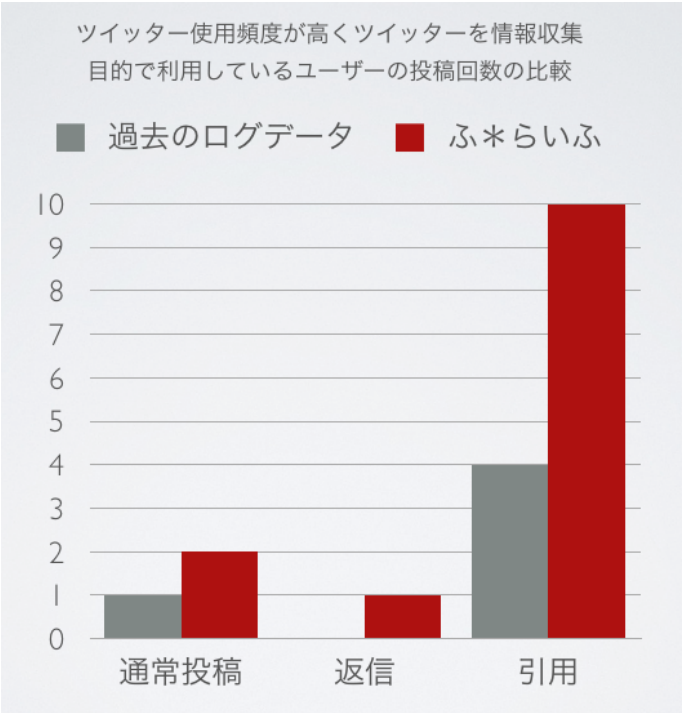

図 6.1　ユーザー A の実験結果 1

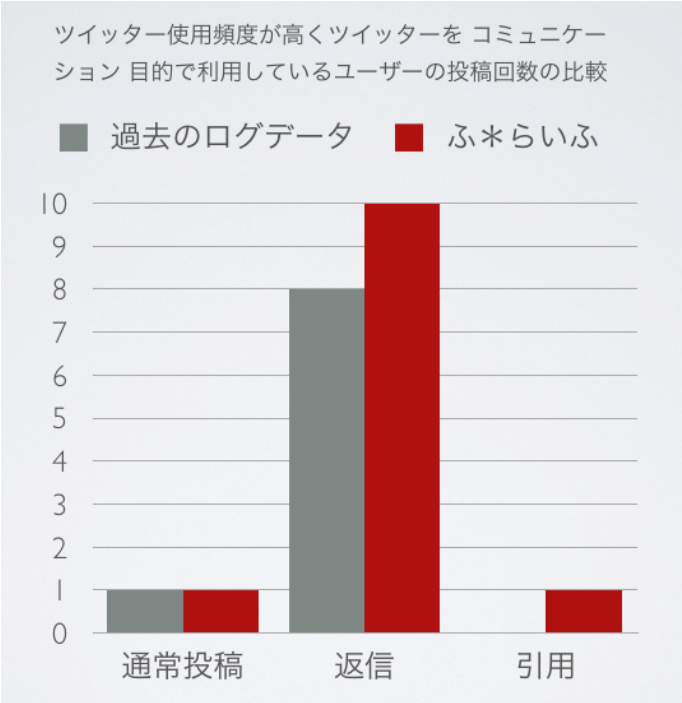

図 6.2　ユーザー B の実験結果 1

ザーは、過去のログデータとして、1 日あたり平均 1 回の通常投稿、0 回の返信、4 回の引用を投稿していた。対して、ふ＊らいふを 1 日プレイしてもらった間に 2 回の通常投稿、1 回の返信、6 回の引用を投稿した (図 6.3)。

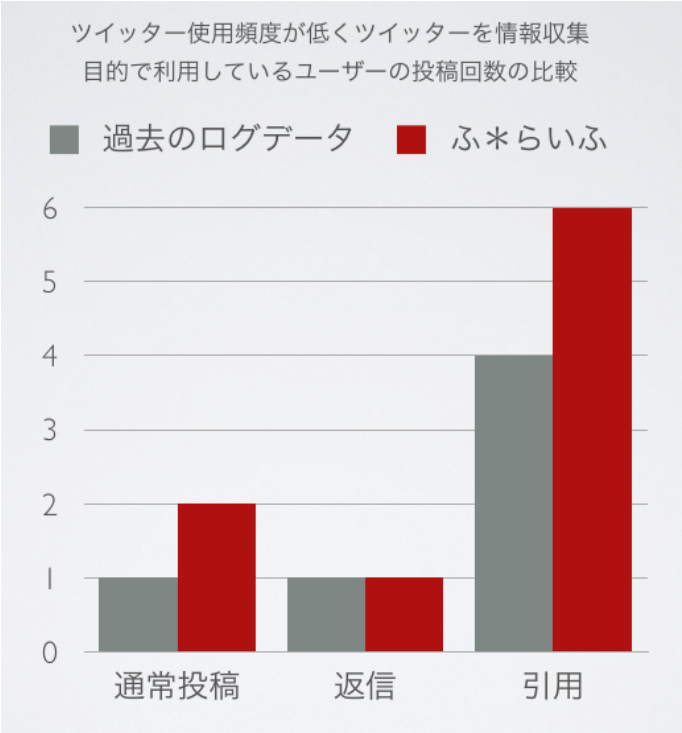

図 6.3 ユーザー C の実験結果 1

ツイッター使用頻度が低くツイッターをコミュニケーション目的で利用しているユーザーは、過去のログデータとして、1 日あたり平均 1 回の通常投稿、3 回の返信、1 回の引用を投稿していた。対して、ふ＊らいふを 1 日プレイしてもらった間に 2 回の通常投稿、2 回の返信、2 回の引用を投稿した (図 6.4)。

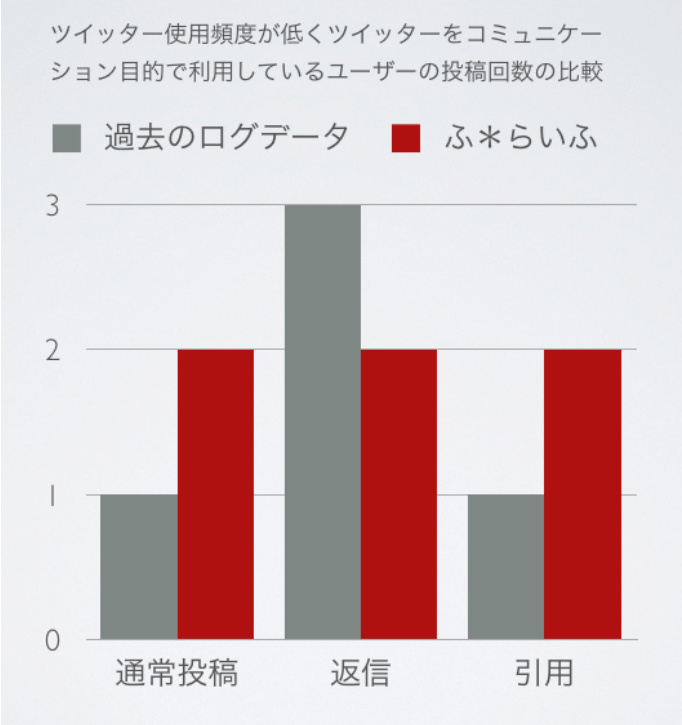

図 6.4　ユーザー D の実験結果 1

6.3. 見知らぬ人とのコミュニケーション促進

見知らぬ人とのコミュニケーションを促進したかを確認するために、ユーザーに10回ツイートの推薦を閲覧してもらい、1回のツイートの推薦毎に、フォロー／お気に入り／ツイートの更なる閲覧の回数を集計した。

まず、ツイッター使用頻度が高くツイッターを情報収集目的で利用しているユーザーは、10回のツイートの推薦に対して、2人をフォロー、お気に入りを3つ、ツイートの更なる閲覧を5回行った (図6.5)。

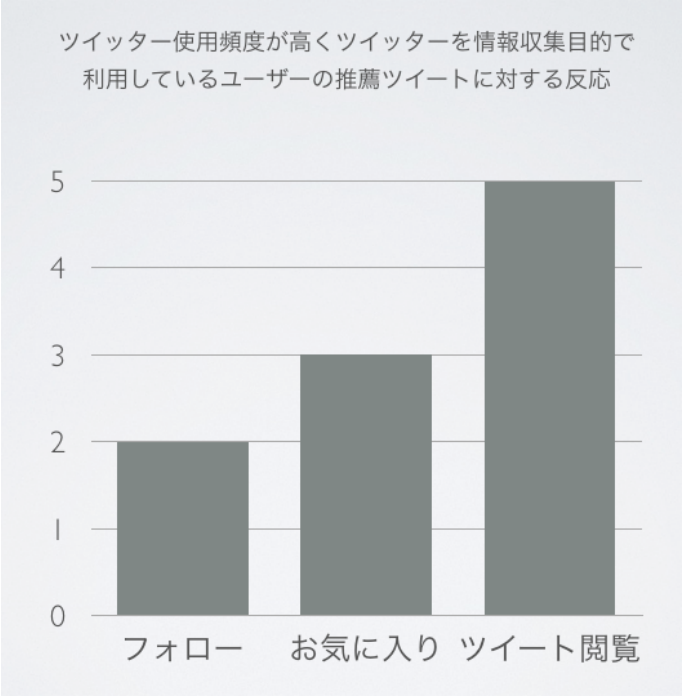

図 6.5 ユーザー A の実験結果 2

つぎに、ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザーは、10回のツイートの推薦に対して、0人をフォロー、お気に入りを1つ、ツイートの更なる閲覧を3回行った (図6.6)。

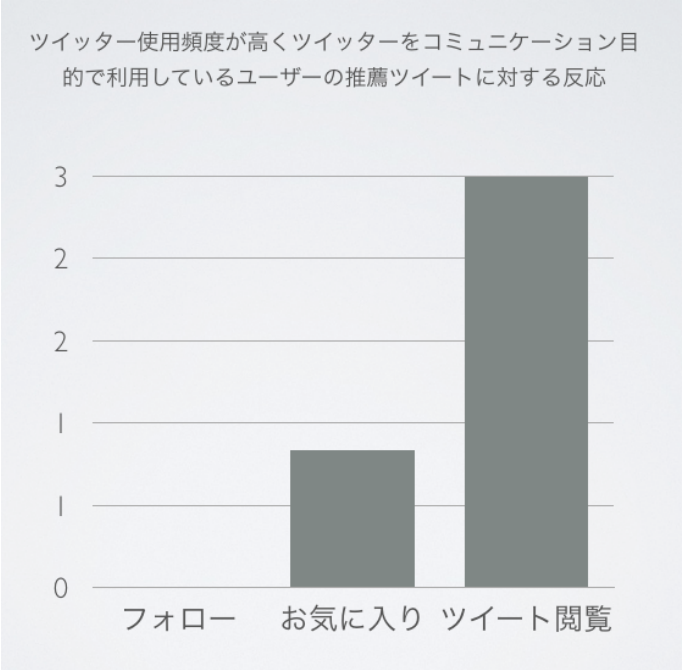

図 6.6 ユーザー B の実験結果 2

ツイッター使用頻度が低くツイッターを情報収集目的で利用しているユーザーは、10 回のツイートの推薦に対して、2 人をフォロー、お気に入りを 2 つ、ツイートの更なる閲覧を 4 回行った (図 6.7)。

ツイッター使用頻度が低くツイッターをコミュニケーション目的で利用しているユーザーは、10 回のツイートの推薦に対して、1 人をフォロー、お気に入りを 4 つ、ツイートの更なる閲覧を 1 回行った (図 6.8)。

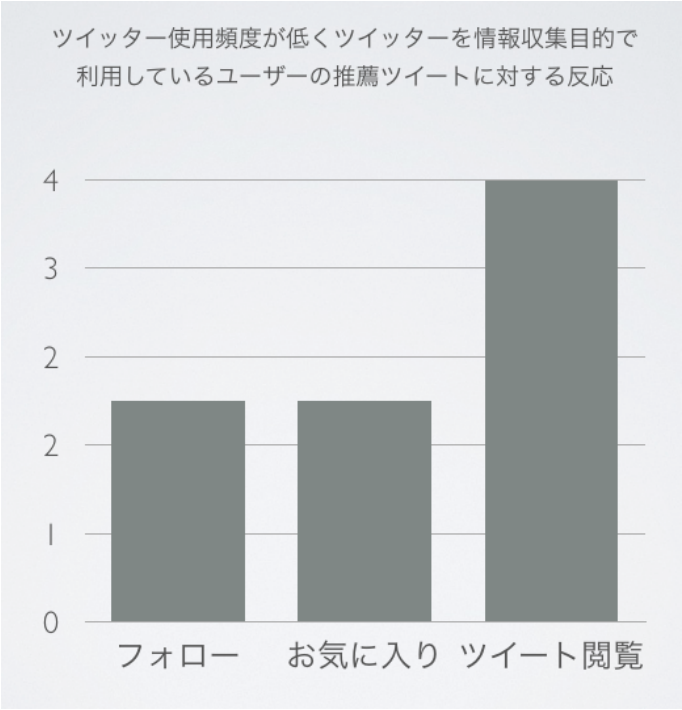

図 6.7 ユーザー C の実験結果 2

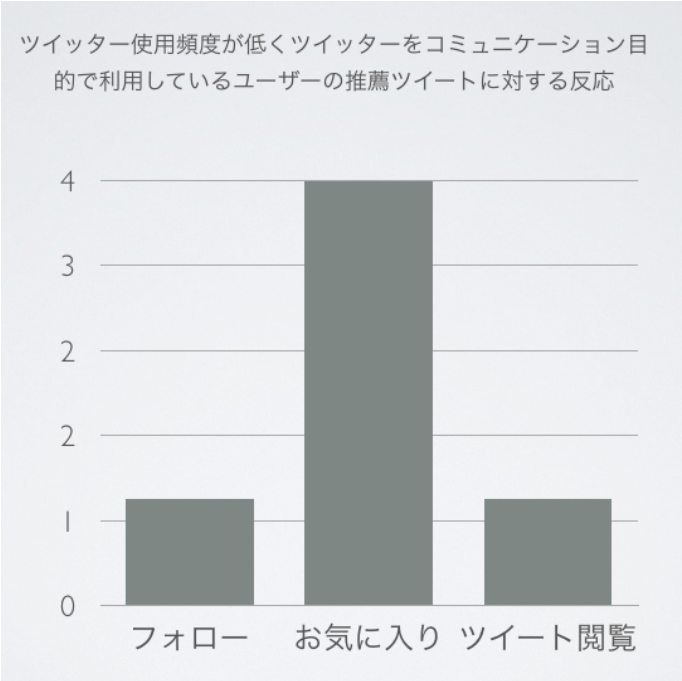

図 6.8 ユーザー D の実験結果 2

第7章
考　　察

7.1. 実験結果から見るふ*らいふの考察

ふ＊らいふが友人とのコミュニケーションを促進したかを確認するための実験では、平均化した過去のログデータに対して、ふ＊らいふをプレイ中のツイートの通常投稿／返信／引用の回数がそれぞれ、ほとんどの場合上回った。もっとも、ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザーの通常投稿は平均と同じであった。また、ツイッター使用頻度が低くツイッターをコミュニケーション目的で利用しているユーザーの返信では下回っていた。これらの結果は、ツイッター使用頻度の高低に関わらず、使用目的をコミュニケーションに置いているユーザーでは、通常投稿と返信のツイート投稿を促進しない場合があることがわかった。したがって、ふ＊らいふは、コミュニケーションに連動する花育成システムが友人とのコミュニケーションを促進するためのシステムとして、コミュニケーション目的のユーザーに対するものとしては改善の余地はあるものの、効果的に機能していることが示された。

次に、ふ＊らいふが見知らぬ人とのコミュニケーションを促進したかを確認するための実験では、ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザーが１人もフォローしなかった場合を除いて、１人以上をフォローし、お気に入りを１つ以上、ツイートの更なる閲覧を１回以上行っている。このことは、ふ＊らいふが備えるツイートの推薦システムによって推薦されたツイートが、ユーザーが関心を寄せられる内容であったことを示すものであり、見知らぬ人とのコミュニケーションを促進するためのシステムとしての有

効性を示している。ツイッター使用頻度が高くツイッターをコミュニケーション目的で利用しているユーザーは、推薦されたツイートをきっかけにしたフォローをだれにもしなかった。これは、ユーザーのツイッター上でのコミュニケーションの相手がフレンドを中心としたものであり、ユーザーのフレンドではない見知らぬ人を対象としたツイートの推薦がツイッターをする目的と乖離していたからであると考えられる。

7.2. 今後の展望

本研究の今後の展望として、機能拡張と実用化の二つの方向性から今後の研究が必要であると考える。

ふ＊らいふは、コミュニケーションと連動する花育成システムとツイートを推薦するシステムを基本的な機能として持つ。まず、コミュニケーションと連動する花育成システムに対する機能拡張として、ログインしないと日照りで花が枯れたり、友人と一緒に一本の花を育成できるなど、コミュニケーションと連動する花育成の要素を増やすことが挙げられる。これは、花を育成する自由度を高めることで、ユーザーに自身で操作可能な工夫の余地を与えるためである。現状では、ツイートを投稿するにしたがって花は一方的に生育するだけであるが、生育が後退するシステムに拡張することにより、より柔軟なコミュニケーションが可能となる。同様に、特定の友人とのコミュニケーションに特に連動する花育成システムを導入することで、実験結果から判明したコミュニケーション目的のユーザーの通常投稿と返信を促すことも可能となる。また、ツイートを推薦するシステムに対する機能拡張として、テキストデータだけでなくツイートに付加された位置情報を学習に利用することが挙げられる。これは、ツイートの推薦に位置を考慮することで、ツイートの推薦の精度を向上する可能性があるためである。現状の推薦システムは、ユーザーのツイートテキストのみを学習対象としているだけであるが、位置等のコンテキスト情報にまで対象を広げることで、場所や行動を共有する見知らぬ人の推薦を行うことも可能となる。

技術的な観点からは、システムのスケーラビリティを高める機能を拡張することが必要である。スケーラビリティとは、利用者や仕事の増大に対してサーバー等のリソースを追加することでシステムを大規模化することを可能とする適応能力のことである。ふ＊らいふは、ソーシャルアプリケーションであり、口コミや友人からの誘い等によるユーザー数の飛躍的な増加の可能性を抱えるシステムである。そのため、ふ＊らいふのシステムは、実装上スケーラビリティの確保を考慮しなければならない。現状、ふ＊らいふは、”APC(Alternative PHP Cache)”と分散メモリキャッシュサーバー”memcached”の PHP クライアントを実装している。PHP のようなインタプリタ言語は、実行の度にソースコードをコンパイルするため、C 言語のようなコンパイル言語に比べてコンパイルにかかるオーバーヘッドの影響による実行速度の遅延を招く。”APC”は、PHP の中間コードのキャッシュや最適化をすることで、アプリケーションサーバーを高速に実行することを可能とする。また、”memcached”は、データベースへのアクセス結果を一時的にキャッシュしてデータベースへのアクセス回数を減らすことで、RDBMS の負荷を減少させるために実装している。もっとも、推薦システムなどのような、ユーザー数の増加に伴い計算すべきデータが増大するような大量データのバッチ処理の分散処理を実現するための機能は、今後の技術的課題として存在する。ふ＊らいふは、推薦システムを実装しているため、大量データのバッチ処理の分散化は重要である。この点、大規模情報の分散応用処理を支援するソフトウェアとして、”Hadoop”を利用することが考えられる。”Hadoop”は、GFS(Google File System) と MapReduce に触発されたオープンソースモデルで実装されたソフトウェアであり、分散ファイルシステムと分散処理フレームワークを利用することができる。これによって、ふ＊らいふは大量のユーザーの増加や推薦処理の複雑化に対してもスケールアウトすることで、スケーラビリティを確保することが可能となる。

ふ＊らいふを実用化する際には、収益化手段と配布手段を考慮する必要がある。ふ＊らいふは、収益化手段として、広告収入とアイテム課金を採用する。広告収

入は、アプリケーションのトラフィックを収入に転換する手段として簡便であるため、ふ＊らいふは、AdMob ネットワークを用いて広告収入を確保する手段を用意している (図 7.1)。また、ふ＊らいふは花育成システムであり、花の種が貴重なアイテムとして価値を持たせるシステムとなっている。そこで、花の種を容易に入手できる手段としてアイテム課金を用意する。

図 7.1　Admob の導入例

現状、iOS SDK を利用して開発されたアプリケーションを配布する手段には、大きく二つの方向性がある。一つ目は、Apple Store を通した配布であり、二つ目は、Web アプリケーションに移植して特定の組織を通さず配布する場合である。Apple Store を通した配布の利点は、単一のマーケットを通じて全世界のすべての iPhone ユーザーにアプリケーションを提供できる点である。また、アプリ

ケーションを有料で販売し、アプリケーション内部で課金する際の決済が開発者とユーザー双方にとって容易である点も重要である。もっとも、アプリケーションの配布はApple Storeの審査を通過する必要があり、その審査に1週間から4週間かかることから、アップデートに時間がかかるという欠点がある。この点、ふ＊らいふは、日本にとどまらないサービスであるツイッタークライアントであるため、世界中のユーザーに配布できる利点を享受することができる。また、ふ＊らいふは、多くのソーシャルネットワークアプリケーションと同じく、無料アプリケーションとして配布する予定であるが、収益化手段としてアプリケーション内部でアイテム課金をする予定である。したがって、ふ＊らいふは、課金決済の手段が容易である利点を享受することができる。アップデートに時間がかかるという欠点は、現状のシステムにおいては、毎月毎週アプリケーション内でのイベントを実施する等の構造的な更新を繰り返す必要はないため、問題にならない。もっとも、Apple Storeを通した配布による実績が悪かった場合には、ユーザーの反応を見ながら素早くアップデートを繰り返すことが可能なWebアプリケーションとして配布する手段は有効である。

第8章
お わ り に

本論文では、ソーシャルネットワーク上でツイートを投稿することで花が育成され、かつ、ツイートの推薦を取得することが 可能なソーシャルアプリケーション ”ふ*らいふ ”を提案した。

ふ*らいふでは、ユーザーは、ツイッターを投稿することで、蒔いた種に水や肥料を与えてお花を育て、育てた花をプレゼントする、コミュニケーションに連動する花育成システムを楽しむことができる。また、ユーザーは、自らの投稿をきっかけに、それに関連する投稿および投稿元である見知らぬ人を推薦するシステムを通して見知らぬ人と出会うことができる。ふ*らいふの目的は、コミュニケーションと連動する花育成システムと推薦システムによって、ツイッター上のコミュニケーションを促進することである。コミュニケーションの相手を、友人と見知らぬ人という 2 タイプに区別することで、友人とのコミュニケーションを支援するシステムと見知らぬ人と知り合えるシステムを実装している。

ふ＊らいふを自由にプレイしてもらったログデータを集計した実験結果では、ツイート投稿を促進する点に関してのコミュニケーションと連動する花育成システムの有効性、見知らぬ人と知り合える点に関してのツイートの推薦システムの有効性をそれぞれ確認することができた。もっとも、コミュニケーションと連動する花育成システムの有効性に関しては、平日の一日にプレイしてもらったログデータであるため、持続的に効果を保つことまでを確認するものではなく、あくまで一時的な効果を確認したものである。現状の課題として、コミュニケーションと連動する花育成の要素の追加、推薦に使用する学習データへの位置情報の追

加が挙げられる。これらの改善を行うことにより、ユーザーに自律的な操作の可能性を提示することができ、また、高い精度を持って見知らぬ人と知り合うことが可能となる。技術的な課題として、”Hadoop”などの導入によりスケーラビリティを向上させることが挙げられる。これによって、大量データのバッチ処理を高速に実行することができ、推薦システムの安定稼働や精度向上を図ることが可能となる。

謝　　辞

本研究の指導教員であり、幅広い知見から的確な指導と暖かい励ましやご指摘をしていただきました慶應義塾大学大学院メディアデザイン研究科の稲蔭 正彦教授に心から感謝いたします。

研究の方向性について様々な助言や指導をいただきました慶應義塾大学大学院メディアデザイン研究科の砂原 秀樹教授に心から感謝いたします。

研究指導など数多くの助言を賜りました慶應義塾大学大学院 メディアデザイン研究科の徳久 悟 特別研究講師に心から感謝いたします。

研究活動、学生生活全般にわたり、数多くの貴重な助言、ご指導をいただいた、株式会社博報堂の小田部巧さんに心より感謝いたします。

研究を行う上で多くの助言や助力をいただいた NetworkMediaProject の皆様に心から感謝いたします。

さまざまな面から研究活動を支えていただき、時に苦楽を共にした慶應義塾大学大学院メディアデザイン研究科 SUGM プロジェクトの皆様に心から感謝いたします。

最後に、研究活動に関するご理解とともに、経済面や生活面において支援していただきました家族に心から感謝いたします。

参 考 文 献

[1] GungHo Online Entertainment, Inc., “TWIMON(http://www.gungho.jp/twimon/),“

[2] MicroAd, Inc, “meromero park for モバツイ (http://m.movatwi.meropar.jp/),“

[3] Tom Lovett, Eamonn O’Neill, David Pollington, James Irwin, “Event-Based Mobile Social Network Services“, MobileHCI ’09, September 15 - 18, 2009, Bonn, Germany. ACM 978-1-60558-281-8.

[4] Twitter inc., “Discovering Who To Follow(http://blog.twitter.com/2010/07/discovering-who-to-follow.html)“

[5] （株）博報堂, SUGM project , “ふ＊らいふワークショップ (http://www.keio.ac.jp/ja/press_release/2009/kr7a43000002f18e.html)“

[6] （株）博報堂, SUGM project , “ふ＊らいふワークショップ (http://www.wsc.or.jp/6th/21_40/27.html)“

[7] Apple inc., “iOS Application Programming Guide(http://developer.apple.com/jp/documentation/Cocoa/Conceptual/CocoaFundamentals/Introduction/chapter_1_section_1.html),”

[8] Apple inc., “iOS Technology Overview(http://developer.apple.com/library/ios/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898),”

[9] Apple inc., “Cocoa Fundamentals Guide(http://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/CocoaFundamentals/Introduction/Introduction.html#//apple_ref/doc/uid/TP40002974),”

[10] CentOS, “CentOS(http://www.centos.org/)“

[11] VBRFix, “VBRFix homepage(http://home.gna.org/vbrfix/)“

[12] Huberman, B. A., Romero, D. M., Wu, F, “Social networks that matter: Twitter under the microscope,”, First Monday, Vol. 14, No. 1, January 2009.

[13] 京都大学情報学研究科－日本電信電話株式会社コミュニケーション科学基礎研究所 共同研究ユニットプロジェクト, “MeCab: Yet Another Part-of-Speech and Morphological Analyzer(http://mecab.sourceforge.net/)“,

[14] Salton. G, Buckley. C, 1988, “Term-weighting approaches in automatic text retrieval. “Information Processing and Management 24 (5): 513-523.

[15] Twitter inc., “Twitter API Documentation(http://apiwiki.twitter.com/w/page/22554679/Twitter-API-Documentation),”

[16] Yii Software LLC., “Yii Framework(http://www.yiiframework.com/),”

[17] naan studio, inc. ., “ecofon for iPhone(http://www.echofon.com/twitter/iphone/),”