Vol.2012-CDS-5 No.25
Vol.2012-DCC-2 No.25
2012/10/18

ヒューマンセントリックコンピューティングにおけるスマートフォンの動的デスクトップシステムの開発

富田 達夫[1,a)]　角田 忠信[1,b)]　伊藤 栄信[1,c)]　藤野 信次[1,d)]　飯田 一朗[1,e)]

概要：ヒューマンセントリックコンピューティングでは，人の所有する携帯端末が起点となりクラウドサービスと連携し，周囲のシステムが人に対して的確なサービスを提供する．今日，スマートフォンとクラウドとを連携させた SaaS や BaaS を提供する企業が登場しているが，個人のスマートフォンを業務利用する形態（BYOD）において，これらのサービスを利用することは困難である．なぜならば，現状のスマートフォンはセキュリティが脆弱であり，業務専用端末としての利用にリスクが伴うためである．本論文では，この問題を解消するために我々が考案した，状況に応じてスマートフォンのデスクトップを動的に変更するシステムを提案する．本システムは，クラウドがユーザの状況を監視し，ユーザが実行すべきアプリケーションを適切なタイミングでスマートフォンにプッシュして，不要になり次第削除する．これにより，必要なアプリケーションのみが端末にある状態をつくり，セキュリティを確保する．また，利用可能なアプリケーションの絞り込みにより，ユーザによるスマートフォンの操作性を向上させ，作業支援を実現する．我々は，動的デスクトップシステムの実用性を評価し，製品化に繋げるためプロトタイプを開発した．本プロトタイプシステムの評価により，ユーザが短時間で容易にアプリケーション起動が可能であることを確認し，本システムの有効性を示した．

1. はじめに

科学技術，とりわけ情報通信技術の発展は目覚しいものがあり，我々の生活を一変させた．人同士のコミュニケーションツールは電話やＦＡＸから電子メール，SNS やブログ，マイクロブログ（Twitter など）へと発展を遂げ，コンピュータを介した通信により会話することが当たり前となった．コミュニケーションに用いるコンピュータも家庭やオフィスに据え置きのデスクトップＰＣからそれと同等の機能を持つスマートフォンを主としたものへ変化しつつあり，大人から子供までスマートフォンを一人一台以上所有することが近い将来普通になると考えられる．

このような状況においても，人間自身は進化しているわけではない．これら情報通信技術（ICT）が齎す恩恵を享受するためには，人間は機器毎に異なる動作の違いを理解し，各機器をを使いこなすための相当の努力や準備が必要となる．言い換えれば，現在は ICT システムの都合が中心の世界であり，人はそれに振り回される形になっている．

我々はヒューマンセントリックコンピューティングというコンセプトを基にこの問題を解消する．ヒューマンセントリックコンピューティングでは ICT システムではなく，人の活動が中心となり，周囲のシステムが的確なサービスを提供する．人が必要になったときに人が必要な手順を踏んでシステムを使用する従来とは異なり，ICT が人の状況を常に監視し，人の都合を理解して，人が ICT のサービスを自然な形で利用できるようにするのである．

ヒューマンセントリックコンピューティングでは，人の状況（コンテキスト）と ICT システムとを結びつける仕組みが必要となる．近年普及しつつあるスマートフォンは，GPS や加速度センサ，NFC リーダライタ等のハードウェアを搭載しており，それを常に携帯することにより，ユーザの行動データを採取して状況把握に利用することが可能である．また，高速広帯域無線通信機能の搭載により，ユーザは何処にいてもクラウドサービスをスマートフォンを通じて利用できる．すなわち，スマートフォンを起点とし，クラウドと連携して的確なサービス提供を行うことにより，ヒューマンセントリックコンピューティングは実現するのである．

スマートフォンとクラウドとを連携させたサービスを，SaaS（Software as a Service） や BaaS（Back-

[1] 株式会社富士通研究所 〒 211-8588 川崎市中原区上小田中 4-1-1
FUJITSU LABORATORIES LTD. Kamikodanaka 4-1-1, Nakahara-ku, Kawasaki, Japan
a) tomita.tatsuo@jp.fujitsu.com
b) ta.tsunoda@jp.fujitsu.com
c) itou.hidenobu@jp.fujitsu.com
d) fujino@jp.fujitsu.com
e) iida.ichirou@jp.fujitsu.com

© 2012 Information Processing Society of Japan

end as a Service） として既に多くの企業が提供している [1], [2], [3], [4], [5]．また，個人で使用するスマートフォンを仕事の場でも使用する形態であるBYOD（Bring Your Own Device）[6] を採用する企業も増加している．そのため，個人のスマートフォンから業務用クラウドサービスを利用することが仕事を遂行する上で今後さらに必要になると考えられる．

しかし，現状のスマートフォンは多機能化の一方で，セキュリティの脆弱性は解消されず，業務専用の端末として利用することが困難である．ユーザはスマートフォンをプライベートで利用している際に，さまざまなアプリケーションを App Store[7] や Google Play Store[8] 等のアプリストアや，インターネット上からインストールする．アプリストア上のアプリケーションは一定の審査が行われるが，それらの完全なセキュリティ保証はない．万が一ユーザが誤ってマルウェアをインストールした場合，業務用クラウドサービスで使用する機密情報を漏洩させるリスクがある．また，ユーザによる機密情報の故意の持ち出しや，盗難による漏洩の危険性もある．従来のクラウドサービスではこのようなリスクに対応できない．

本論文では，これを解決するために我々が考案した，ユーザのコンテキストに応じてスマートフォンのデスクトップを動的に変更するシステムを提案する．本システムは，クラウドとスマートフォンとが連携し，クラウドがユーザのコンテキストに応じて，必要となるアプリケーションを適切なタイミングでプッシュ追加し，不要になった時点でスマートフォンから完全に削除する．スマートフォンのデスクトップ画面上では，ユーザの現在のコンテキストにおいて，実行可能なアプリケーションが絞り込まれて提示させる．これにより，ユーザが自宅にいる場合は自宅専用，会社にいる場合は会社専用といった，ユーザのコンテキストに特化した専用端末となり，不正なアプリケーションによる機密情報へのアクセスや，不要な機密情報の持ち出しを防止することが可能になる．また，アプリケーションの絞り込み提示により，ユーザのスマートフォン操作性を向上させ，作業支援を実現することが可能になる．

我々は，動的デスクトップシステムの実用性を評価し，そのフレームワークを弊社で開発中のヒューマンセントリック基盤へ搭載して製品化に繋げるため，プロトタイプを開発した．本論文では動的デスクトップシステムのアーキテクチャおよびプロトタイプ設計指針を述べ，本システムについて実施した評価結果を記す．

本論文の構成は以下の通りである．まず，2 章にて関連技術の紹介をし，他技術と本研究との違いを明確にする．3 章にて動的デスクトップシステムのコンセプトについて説明する．4 章にて本システムのアーキテクチャを示す．5 章にて開発したプロトタイプの詳細を述べ，6 章にて本システムの評価結果を示す．7 章で本稿をまとめる．

2. 関連技術

近年多くの企業がスマートフォン向けのクラウドサービスを提供している．アップル社の iCloud [1] は，端末とクラウドとの間でアプリケーションやデータの同期を容易に可能にする．また，グーグル社が提供する SaaS（Software as a Service）の製品群である Google Apps[2] では，専用のネイティブアプリをインストールせずとも，ブラウザのみでオフィスアプリケーションをスマートフォン上で動作させることが可能である．Microsoft 社は，Windows Azure Mobile Service[3] の提供を発表した．これは，モバイルアプリの開発者向けに提供されるプラットフォームで，クラウド上のデータベース，ユーザ管理，プッシュ配信機能を使用するモバイルアプリケーションを容易に作成可能とする．このようなスマートフォン向けプラットフォームの提供形態は BaaS（Backend as a Service）と呼ばれ，Parse[4] や appiaries[5] など，現在多くの企業が参入しつつある．

今日の SaaS や BaaS において，クラウドサービスへのユーザ認証機能は提供されているものの，スマートフォン上で動作するアプリケーションそのものへのユーザのアクセス制限を行うことは困難である．BYOD への対応には，ユーザのコンテキストに応じてアクセス可能なデータやアプリケーションを制御する仕組みが必要である．

これを実現する関連技術として，文献 [9] では，ユーザの状況に対応するリモートデスクトップ環境を構築し，コンテキストに応じてその環境へアクセス可否を決定するシステムについて述べられている．本システムではスマートフォン上に不要なアプリケーションをインストールする必要がないというメリットはあるものの，サーバとの通信を常に保つ必要があり，ネットワークへの高い負荷が問題となる．文献 [10], [11], [12] では，仮想化技術を用いた手法について述べている．ハイパバイザを予め端末にインストールし，端末イメージをマイグレーションさせることにより，コンテキストに応じて端末の環境を丸ごと変更することが可能になる．本手法では，リモートデスクトップ方式のようにサーバと常時接続する必要はないが，セキュリティを考慮した場合，コンテキスト切替時にイメージをサーバ側へ退避させ，機密情報が含まれるデータを端末に残さないようにする仕組みが必要である．イメージのサイズは大きいため，サーバへの退避時および端末への復元時にネットワーク負荷が高くなる．また，退避や復元に失敗した場合，コンテキスト変化時に画面を切り替えられない可能性がある．文献 [13] では，コンテキスト毎にポリシーを切り替え，アプリケーションの動作を制限する．本手法はアプリケーションを端末に格納しておくため，ネットワーク負荷の問題は発生しない．しかし，端末に機密情報は格納されたままとなるため，セキュリティのリスクはゼロではない．

© 2012 Information Processing Society of Japan

Vol.2012-CDS-5 No.25
Vol.2012-DCC-2 No.25
2012/10/18

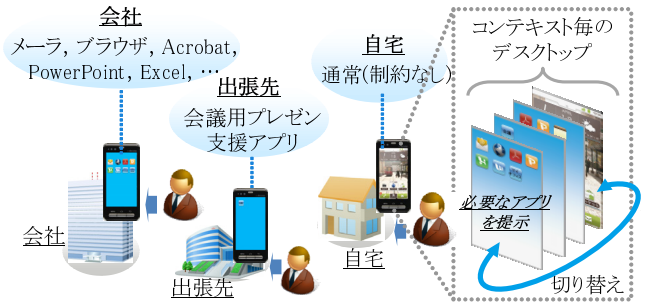

図 1 動的デスクトップシステムの動作イメージ
Fig. 1 An operation image of the dynamic desktop system

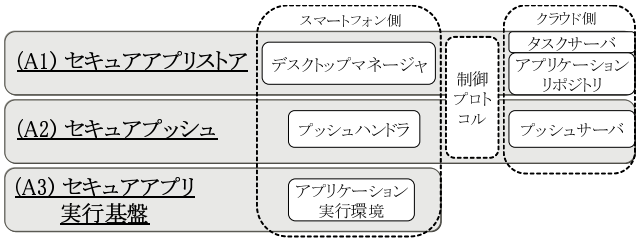

図 2 動的デスクトップシステムのフレームワーク
Fig. 2 The framework of the desktop system

3. 動的デスクトップシステムのコンセプト

我々が提案する動的デスクトップのコンセプトは，スマートフォンとクラウドとを連携させ，ユーザのコンテキストに応じたデスクトップ画面切り替えとアプリケーション追加・削除による，スマートフォンのセキュアな BYOD 対応システムの実現である．それに加えて，本システムはユーザの所望するアプリケーションへ素早くアクセス可能にし，ユーザによるスマートフォンの操作性も向上させる．

今日のスマートフォンでは，必要となるアプリケーションが端末にインストールされていない場合，何度もアプリストアにアクセスし，所望のアプリケーションを検索し，ダウンロードを行う必要がある．動的デスクトップシステムは，そのようなユーザへの負担を強いないものにする．

たとえば，**図 1** のように，コンテキスト毎にデスクトップ画面を用意してクラウドの指示により動的に切り替える．BYOD を採用する企業向けとして，会社ではメーラーやオフィスアプリケーションなど，機密情報を含むアプリケーションにアクセスできるが，家に戻ると，デスクトップ画面が切り替わり，機密情報へのアクセスを不可にすることができる．さらに，特定のコンテキストにおいてのみ必要なアプリケーションを追加・削除することにより，不要な機密情報を端末に残さず，よりセキュアに個人用のスマートフォンを会社内でも使用できるようになる．

このように，ユーザがあるコンテキストにある場合，ユーザが担う役割が決まる．役割が決まれば，タスクが絞られる．そして，コンテキストの推定により，デスクトップに表示すべきアプリケーションの絞り込みが可能である．これはすなわち，コンテキストに応じたスマートフォンの専用端末化とも言える．動的デスクトップシステムは，従来のスマートフォンの問題を解消して BYOD への対応を可能にし，ユーザが容易に許可されたアプリケーションへのアクセスを可能とする，ヒューマンセントリックコンピューティングにおける重要な機構なのである．

また，2 章で述べた従来の BYOD 技術の問題を解消するため，我々の提案するシステムは，アプリケーションをロジックのみを含むパッケージとしてプッシュ通知することにより，ネットワークへの負荷を抑え，アプリケーションの追加・削除を高速かつ容易に実行できるようにする．

4. システムアーキテクチャ

我々は，動的デスクトップ環境を実現するためのシステムアーキテクチャを検討した．スマートフォン上で 3 章で述べたような動作を実現させるのに必要なシステム要件は，(i) クラウドの指示により，コンテキストに応じて利用可能なアプリケーションの制限を行い，セキュリティを確保すること，(ii) 必要なアプリケーションがインストールされていない場合は，クラウドからのプッシュにより追加が行われること，(iii) 不要になった際はアプリケーションを削除し，情報漏洩を防止すること，(iv) 必要なアプリケーションを絞り込んで表示することにより，スマートフォンの操作性を向上させること，が挙げられる．

以上を踏まえて我々は**図 2** に示すフレームワークを考案した．これは次に挙げる 3 つの主要機構から構成される．

(A1) セキュアアプリストア
コンテキスト毎にユーザがアクセス可能なアプリケーションをセキュアに制御するための仕組みである．デスクトップ画面をどのような構成にするかの設定情報はクラウド側からプッシュ通知される．クラウドがユーザのコンテキストを観測し，適切なタイミングでスマートフォンへ通知することにより，ユーザのコンテキストに応じた画面切り替えを実現する．

(A2) セキュアプッシュ
クラウドからの制御メッセージやアプリケーション本体をスマートフォンへセキュアにプッシュするための仕組みである．端末にプッシュされたアプリケーションは暗号化され，他のアプリケーションからのアクセスができないように格納する．また，制御メッセージのプッシュにより，アプリケーションの完全な削除を実行可能にする．

(A3) セキュアアプリ実行基盤
スマートフォンにインストールされたアプリケーションを端末上で実行するための仕組みである．アプリケーションの実行に必要な基本機能（実行環境）はスマートフォンに予めインストールしておき，アプリ

© 2012 Information Processing Society of Japan

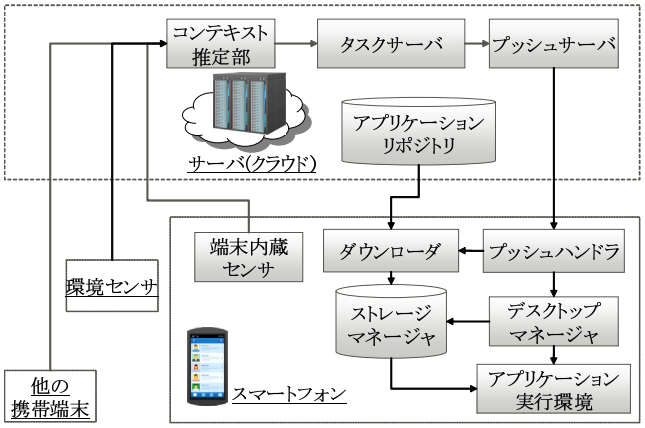

図 3 プロトタイプシステムの構成
Fig. 3 The components of the prototype system

ケーションのロジックのみが含まれるパッケージをサーバからダウンロードし，実行環境上で動作させる．これにより，アプリケーションの追加削除が容易に実現できるうえ，アプリケーションの基本部を省くことによるパッケージサイズの削減により，ダウンロードを短時間で実行することが可能になる．

以上のフレームワークに基づき，我々は Android 端末上で動作する動的デスクトップシステムのプロトタイプを作成した．次章にてその詳細を述べる．

5. プロトタイプの開発

5.1 設計指針

我々が考案した動的デスクトップシステムの実用性を評価し，そのフレームワークを弊社で開発中のヒューマンセントリック基盤へ搭載し製品化に繋げるため，我々は動的デスクトップシステムのプロトタイプを作成した．本システムは，次の設計指針に基づき構築したものである．

- プッシュ通知によるアプリケーションインストール，および，削除を容易に実行可能であること．
- スマートフォンユーザにとって高い操作性を保つこと．
- ハードウェアやネットワークのリソース消費を極力抑えること．
- アプリケーション開発者が，端末上で動作するアプリを容易に開発可能であること．
- マルチプラットフォームに対応すること．

本指針を基に，我々が開発したプロトタイプの構成を**図 3** に示す．次節以降では，本システムの主要な構成要素について説明していく．

5.2 メッセージ・アプリケーションプッシュ

制御メッセージやアプリケーションのスマートフォンへのプッシュ通知を実現するため，本システムではスマートフォン側にプッシュハンドラ，サーバ側にプッシュサーバを用意する．プッシュハンドラはスマートフォンのバックグラウンドで動作し，プッシュサーバとの直接のメッセージ送受信を担うモジュールである．プッシュハンドラはまず，プッシュサーバ接続のための認証を行い，接続が許可されると，プッシュサーバと TCP コネクションを張り続ける．そして，プッシュサーバからのプッシュメッセージを受信すると，その内容を解析し，ダウンローダへのアプリケーションパッケージの取得指示や，デスクトップマネージャへの画面の切り替え，および，アプリケーションアイコンの追加・削除命令を行う．

表 1 アプリパッケージの構成例
Table 1 An example of components of an application package

ファイル名	説明
index.html	アプリ起動時に表示するトップページ
style.css	アプリのスタイルの定義
lib/app.js	ロジックが記述された JavaScript
webapp-manifest.json	設定ファイル

5.3 アプリケーション実行環境とパッケージ

スマートフォンで実行可能なアプリケーションは，ネイティブアプリ，WEB アプリ，ハイブリッドアプリの 3 つに分類できる [14]．ネイティブアプリは，スマートフォンのハードウェアが直接実行可能なアプリケーションであり，その本体はスマートフォンにインストールする．一方，WEB アプリは，WEB ブラウザを通じて実行可能なアプリケーションであり，その処理は基本的にサーバ上で行われる．ハイブリッドアプリは，一見 WEB アプリのように動作するが，HTML や JavaScript をスマートフォンに格納し，ローカルで実行することによりネイティブアプリと同様の動作を実現する．W3C により策定が進められている HTML5 [15] は，ネットワーク切断時にディスクキャッシュを用いてローカルで WEB アプリを実行可能な仕組みが提供されている．これを用いることにより，HTML5 をハイブリッドアプリとして扱うことも可能である．

我々が開発したシステムは，アプリケーションプッシュに適していると考えられるハイブリッドアプリを採用した．具体的には，HTML5 をあらゆるプラットフォームでネイティブアプリと同様に扱える PhoneGap[16] を基盤として，アプリケーション実行環境を開発した．これをスマートフォンにプリインストールしておくことにより，アプリケーション実行制御を行う．アプリケーションは HTML5 により記述されたロジックのみでパッケージングできる．本システムで用いるアプリケーションパッケージの構成例を**表 1** に示す．HTML や JavaScript，CSS，webapp-manifest.json という設定ファイルなどが含まれる．このようなパッケージ構成により，本システムでは軽量で容易なアプリケーションプッシュを可能にする．

© 2012 Information Processing Society of Japan

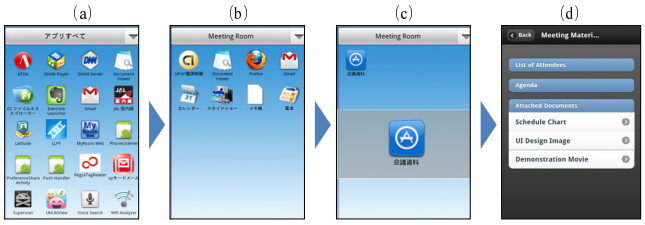

図 4 スマートフォン上の動的デスクトップの動作例
Fig. 4 An example of operation of the dynamic desktop of a smartphone

5.4 デスクトップマネージャによる画面変化

作成したプロトタイプにおける動的デスクトップは，サーバからのプッシュメッセージに呼応してデスクトップ画面を変化させる．スマートフォンにサーバから画面切り替え通知がプッシュされると，デスクトップ画面はユーザの現在のコンテキスト専用のものに切り替わり，アプリケーションのアイコンの配列はそれに応じて変化する．各画面のアプリケーションアイコンの配列に関しては，アイコンの配列情報をサーバからプッシュすることにより，後から動的に変更することが可能である．この仕組みにより，デスクトップ上にユーザに必要なアプリケーション一覧を提示することが可能である．

また，ユーザがアプリケーションにアクセスすべきでないコンテキストとなる際に，画面切り替え通知をサーバからプッシュ通知することにより，アイコンを隠して当該アプリケーションを起動不可にすることができる．さらに，ビジネス用途などにおいて，より高いセキュリティを考慮する必要がある場合，別のコンテキストに持ち出せないよう，サーバからのプッシュ通知によりアプリケーション本体を動的に削除することも可能である．

図 4 は本プロトタイプにおいて開発したスマートフォン用デスクトップ画面の変化の様子を示す．図の (a) はスマートフォンにインストールされたアプリケーション全てのアイコンが配置されたホーム画面である．サーバからの特定の指示がない場合はこの画面が表示される．

図の (b) はサーバから「会議室」というデスクトップ画面切り替え情報を受信した際に表示される画面である．ここでのアイコンの配列は，タスクサーバにおいて会議で使用するアプリケーションとして設定され，プッシュ通知により提示されるものであり，(a) で表示されていたアプリケーションから必要なもののみを絞り込んだものになる．

図の (c) と (d) は，会議資料に相当する会議アプリケーションがサーバからプッシュされ，アプリケーションが起動される様子を示す．(c) はアプリケーション起動時にスプラッシュ画面が表示され，アプリケーションが立ち上がりつつあるときの画面である．(d) は本アプリケーションが起動し，実行されているときの様子である．このように，本システムは画面の切替だけでなく，サーバからのプッシュメッセージによりアプリケーションのアイコンを動的に追加したり，アプリケーションを自動実行させたりすることも可能である．また，ここでプッシュされたアプリケーションは，「会議室」ではないコンテキストへ移動したことをタスクサーバが検知すると，プッシュ通知により会議アプリケーションを完全に削除し，機密情報の漏洩を防止することも可能である．

5.5 実行手順の概要

(1) アプリケーションプッシュ準備
アプリケーション提供者は，アプリケーションパッケージをアプリケーションリポジトリに登録する．また，システム管理者は，各コンテキストとそれに応じて提示するアプリケーションおよびその配列情報の紐付けをタスクサーバへ登録する．

(2) プッシュサーバへの制御メッセージ通知
本システム運用時では，まず，スマートフォンの端末内蔵センサや，他の携帯端末，および環境センサからの情報がサーバに送信される．これらの値は，たとえば，位置測位を行うための GPS 情報や WiFi の電波強度，加速度センサ・ジャイロセンサの値，NFC リーダが読み取ったタグ ID などである．そして，コンテキスト推定部がそれらの値からユーザのコンテキストを推定する．推定されるコンテキストとは，たとえば，ユーザが会社の会議室にいる，外出して車で移動中といったユーザの状態を示す情報である．推定したコンテキストはタスクサーバに通知される．タスクサーバは，コンテキストに紐付けられたアプリケーションや，アイコン配列情報をプッシュサーバへ通知する．

(3) スマートフォンへのメッセージプッシュ
プッシュサーバはタスクサーバからのプッシュ通知を受けると，メッセージをプッシュ独自のプロトコルを用いて，端末のプッシュハンドラへ通知する．

(4) アプリケーションのダウンロード
スマートフォンがプッシュ受信したメッセージがアプリケーションプッシュ通知である場合，その中に含まれる URI 情報を用いて，ダウンローダがアプリケーションパッケージをアプリケーションリポジトリからダウンロードし，スマートフォンのストレージマネージャにインストールする．

(5) アプリケーションの削除
スマートフォンがプッシュ受信したメッセージがアプリケーション削除通知である場合，ストレージマネージャに格納していたアプリケーション本体，および，デスクトップ上に表示させていた当該アプリケーションのアイコンを削除する．

(6) デスクトップ画面切り替え
スマートフォンがプッシュ受信したメッセージが画面

© 2012 Information Processing Society of Japan

表 2 サーバのハードウェア仕様
Table 2 The specifications of hardware of the server

機種名	Fujitsu PRIMERGY TX100
CPU	Xeon Processor (E3120 3.16 GHz)
メモリ	8GB DDR2
HDD	1TB SATA HDD
OS	Cent OS 5.5

表 3 スマートフォン N と T の仕様
Table 3 The specifications of smartphone N and T

機種名	スマートフォン N	スマートフォン T
CPU	S5PC110 1GHz	QSD8250 1GHz
内部メモリ	16GB iNAND	512MB RAM
外部メモリ	（内蔵メモリ使用）	2GB Micro SD
OS	Android 2.3	Android 2.1

切り替え通知である場合，その中に含まれるアイコン配列情報を用いて，デスクトップマネージャが画面上に表示させるアイコンの切り替えを行う．

(7) アプリケーションの起動
ユーザがアプリケーションアイコンをクリックする，もしくは，サーバからアプリケーション自動起動メッセージを受信すると，アプリケーション実行環境は，ストレージマネージャに格納したアプリケーションロジックを抽出し，その起動を行う．

6. 評価

6.1 評価環境概要

スマートフォンの動的デスクトップシステムの実用性を確認するため，開発したプロトタイプの評価を行った．本プロトタイプのサーバは**表 2**，スマートフォンは**表 3** に示すハードウェアで構築した．プッシュサーバ，タスクサーバ，アプリケーションリポジトリ，アプリケーションプロバイダ，コンテキスト推定部および各種ＤＢはひとつのサーバ上に構築した．スマートフォンにはプッシュハンドラ，ダウンローダ，デスクトップマネージャ，アプリケーション実行環境をプリインストールしておき，評価を行った．

6.2 アプリケーションプッシュの評価

スマートフォンの利用において，ユーザ操作性を向上させることは非常に大きな課題である．本システムにおいては，コンテキスト情報の検知後，スマートフォン上に実行すべきアプリケーションが提示され，実際に実行可能になるまでの時間が重要となる．スマートフォンでの実行において，どの処理がボトルネックとなるかを評価するため，次の４つのパターンに関して実験を行った．

I) ZIP でアーカイブされたアプリケーションパッケージをそのまま外部メモリに一旦格納．その後，パッケージを展開し，インストール実行．
II) ZIP でアーカイブされたアプリケーションパッケージを ZipInputStream を用いて展開処理を行いつつ直接インストール．
III) 非圧縮の TAR でアーカイブされたアプリケーションパッケージをそのまま外部メモリに一旦格納．その後，パッケージを展開し，インストール実行．
IV) 非圧縮の TAR でアーカイブされたアプリケーションパッケージを TarInputStream を用いて展開処理を行いつつ直接インストール．

これらの条件の下に，我々はプッシュサーバにアプリケーションプッシュ通知が行われてから，スマートフォン上で当該アプリケーションが提示され，起動が可能になるまでの時間を測定した．アプリケーションパッケージは，会議用ドキュメント表示アプリを使用し，そのサイズは，非圧縮の TAR で 655KB，ZIP 圧縮後のアーカイブファイルで 167KB （圧縮率 27% ） であり，25 個のファイルと 5 個のフォルダが格納されている．アプリケーションパッケージのダウンロードは一般的なオフィスの無線 LAN を経由して実行した．

6.3 アプリケーションプッシュ評価の結果

図 5 はそれぞれのテストケースにおいて，プッシュサーバにアプリケーションプッシュ通知がされてからアプリケーションの起動が可能になるまでの経過時間の比較を示す．また，**図 6** はスマートフォンにおける単位時間当たりのデータ処理量（スループット）の比較を示す．これらの結果から次のことが示された．

- 各パターンにおいて，スマートフォン N はスマートフォン T と比較して 2 倍以上高速に処理を実行．
- 2 機種ともアプリケーションパッケージを ZIP でアーカイブ化し，かつ，直接インストールする方式 （パターン II ）が最も高速に処理実行．
- スマートフォン N においては，アプリケーションパッケージを一旦 SD カードに格納する方式（パターン I, III）と格納しない方式（パターン II, IV）との差は僅かである．一方，スマートフォン T においては，それらの差は大きい．
- スマートフォン N においては，アプリケーションパッケージを ZIP 圧縮する方式（パターン I, II）が圧縮しない方式（パターン III, IV）と比べて約 2 倍高速に処理を実行する．スマートフォン T においても，N ほどではないものの，ZIP 圧縮を行う方式の方が高速に処理を実行する．

それぞれのパターンにおいてスマートフォン N が T よりも高速に実行する理由は，ストレージデバイスの違いにある．アプリケーションはインストール時に外部メモリに格納される．スマートフォン T は SD カードにデータが

© 2012 Information Processing Society of Japan

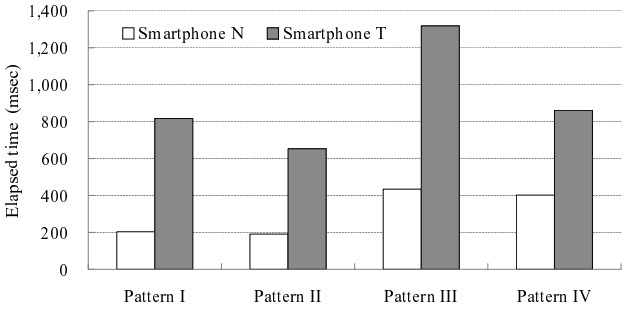

図 5 経過時間の比較
Fig. 5 The comparison of the elapsed times

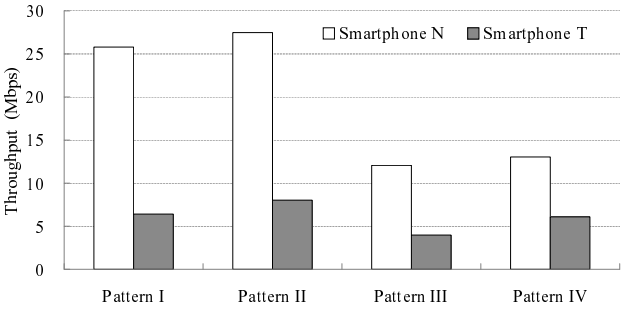

図 6 スループットの比較
Fig. 6 The comparison of throughputs

書き込まれるが，スマートフォン N の場合，内蔵メモリ（NAND フラッシュメモリ）に記録される．そのため，これらのデバイスの違いによる入出力処理時間の差がスループットとして大きく影響する．また，Android 2.1 と 2.3 の違いによる影響もある．Android 2.3 では JIT コンパイラ (Just-In-Time Compiler) を使用するため，従来の OS と比較して高速に動作する．

アプリケーションパッケージは ZIP により圧縮してアーカイブ化する方が圧縮しない場合より処理完了が早い．これは, 圧縮によるサイズ削減のため短時間でパッケージのダウンロードが完了することや, 大きなファイルサイズのデータ処理を TarInputStream が行うより ZipInputStream で直接処理を行う方が高速で効率的に処理が実行できることが理由として挙げられる．

これらの結果より，アプリケーションを ZIP 圧縮してアーカイブ化して処理を行い，かつ，外部メモリに一旦格納せず，直接インストールする方式が最も高速に処理が実行可能であることを確認した．また，スマートフォン N と比較し低速であるスマートフォン T の場合でも，パターン II を適用することにより 1.0 秒以内に処理を完了する．よって，本システムはスペックの高くないデバイスであっても実用的な時間内で処理を行うことができ，ユーザによる操作性を損なうことはないことが示された．

6.4 動的デスクトップによるアプリケーション絞り込みの効果に関する評価

動的デスクトップがコンテキストに応じてアプリケーションを絞り込むことによる，ユーザによるスマートフォン操作性向上を確認するため，開発したプロトタイプを用いて評価を行った．

本評価はスマートフォン T を用いて実施した．本端末に予め異なる 80 個のアプリケーションをインストールしておき，タスクサーバの設定により，画面切り替えのプッシュ通知毎にデスクトップに提示するアプリケーションの配列を変更できるようにした．

評価実験では，タスクサーバと接続可能な実験用 PC を別に用意し，本 PC がまず上記の 80 個のなかから指定した数のアプリケーションをランダムに選択し，そのアイコン配列情報をタスクサーバに設定するようにした．そして，サーバからのプッシュ通知に応じて，スマートフォン上で指定した数のアイコンがランダム配列で画面切替時に提示されるようにした．また，プッシュ通知と同時に，実験用ＰＣは，先に選択したアプリケーションのうちのひとつをさらにランダムに選択し，そのアプリケーション名を本ＰＣのディスプレイに表示させる．被験者はＰＣの表示を参照し，スマートフォンのデスクトップ上で指定されたアプリケーションのアイコンを発見して，当該アプリケーションを起動する．本評価では，この一連の操作において，プッシュ通知時にスマートフォンの画面切替後，被験者が指定されたアプリケーションを発見し，起動するまでの時間を測定した．なお，被験者はスマートフォンの操作に慣れたユーザであり，上記 80 個のアプリケーションの名前やアイコンの形状を事前に把握しているものとする．

スマートフォンのデスクトップマネージャは，一般的なスマートフォンのランチャー画面と同様に，アプリケーションのアイコンを 1 画面内に横に 4 個（列），縦に 5 個（行）配列させることができ，1 画面に同時に最大 20 個のアイコンを表示させることが可能である．入りきらなかったアプリケーションのアイコンは，画面を縦にスクロールさせることにより表示される．スマートフォン T のディスプレイのサイズは約 4 インチ，解像度はフルワイド VGA（480 × 854）である．

6.5 アプリケーション絞り込み効果の評価結果

本評価結果を**図 7** に示す．横軸は，実験用 PC が指定する，スマートフォン上に提示させるアプリケーションの数であり，縦軸は，プッシュ通知後アプリケーション起動までの時間を示している．グラフの各点は，提示アプリケーション数を固定して 30 回ずつ試行した際の平均値であり，グラフのエラーバーは 95% の信頼区間を示す．

本結果より，ユーザがアプリケーションを発見して起動するまでの時間は，提示するアプリケーション数にほぼ比例して増大することが分かった．アプリケーション数が 80 個，すなわち，動的デスクトップシステムを用いない場合，

© 2012 Information Processing Society of Japan

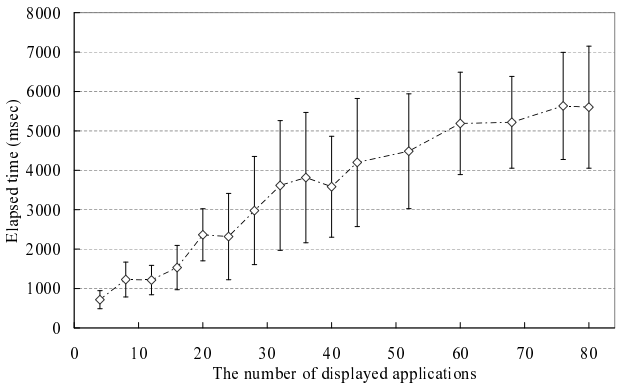

図 7 画面切り替わりからアプリケーション起動までの経過時間とアイコン数との関係

Fig. 7 The number of applications vs. elapsed time from displaying the desktop to starting a specified application

発見・起動するまでに約 4〜7 秒かかるが，アプリケーションを 12 個（横 4 ×縦 3）に絞り込むと，1.6 秒以下で起動することが可能になる．さらに，4 個（横 4 ×縦 1）まで絞り込むと，1 秒以下で起動可能になる．これより，動的デスクトップによるアプリケーション絞り込みは非常に効果があり，ユーザの操作性向上に大きく貢献すると言える．

また，システム管理者が，コンテキスト毎に必要なアプリケーションを最小限に絞ってタスクサーバに登録しておくことは難しい作業であると考えられるが，アプリケーション 12 個程度の絞込みでも効果があるため，ユーザが使用すると考えられるアプリケーションがある程度予想がつくのであれば，考えられるものを 12 個程度登録しておけばよい．システム管理者は，最適なアプリケーションの登録に頭を悩ませることはないため，そのような観点からも本システムは実用的であると言える．

7. まとめ

我々は，スマートフォンがセキュアに業務用クラウドサービスを利用可能にし，企業向け BYOD に対応させるため，スマートフォンとクラウドとを連携させた動的デスクトップシステムのアーキテクチャを考案し，そのプロトタイプを開発した．本システムは，特定のコンテキストにおいて，そこで必要となるアプリケーションのみ利用可能である状態をつくりだし，スマートフォンをそのコンテキストにおける専用端末にする．

また，評価により，開発したプロトタイプシステムは，ユーザのコンテキスト変化時に 100KB 程度のアプリケーションパッケージを 1 秒以内にプッシュ追加可能であり，アプリプッシュを実施してもユーザの操作性を大きく損なうことはないことを確認した．さらに，アプリケーションの絞り込みの効果の評価により，提示するアプリケーションを 12 個まで絞り込むと，ユーザは 1.6 秒以下で起動することが可能であることを確認した．これは，多くのアプリケーションがインストールされた従来のスマートフォンを使用する場合に比べて，非常に短時間でユーザがアプリケーションを起動可能であることを示しており，スマートフォンの操作性を大きく向上させるものになる．

以上を踏まえると，動的デスクトップシステムは，BYOD への対応のみならず，ユーザのスマートフォン操作性を向上させる，ヒューマンセントリックコンピューティングを実現する重要な基本機構であると結論付けることが可能である．本研究の今後の課題として，開発したシステムを実際にユーザに利用してもらい，その使用感を評価することにより，実用化にむけて本システムの改良をさらに行っていくことが挙げられる．

参考文献

[1] Negrino, T.: *iCloud: Visual QuickStart Guide*, Peachpit Press (2012).
[2] Google Inc.: Google Apps for Business, *http:// www.google.com / enterprise / apps / business /*.
[3] Microsoft Corporation: Windows Azure Mobile Services Dev Center, *http:// www.windowsazure.com / en-us / develop / mobile /*.
[4] Parse: *https://www.parse.com/*.
[5] appiaries: *http://www.appiaries.com/en/*.
[6] Bradley, T.: Pros and Cons of Bringing Your Own Device to Work, *http://www.pcworld.com /businesscenter /article /246760 /pros_ and_ cons_ of_ bringing_ your_ own_ device_ to_ work.html* (2011).
[7] McCann, T.: *The Art of the App Store: The Business of Apple Development*, Wrox Press (2011).
[8] Meier, R.: *Professional Android 4 Application Development*, Wrox Press (2012).
[9] Stieghahn, M. and Engel, T.: Law-aware access control for international financial environments, *Proceedings of the Eighth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 33–40 (2009).
[10] Heiser, G.: Virtualizing Embedded Systems - Why Bother ?, *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, pp. 901–905 (2011).
[11] Heiser, G.: The Motorola Evoke QA4: A case study in mobile virtualization, *White paper, Open Kernel Labs, http:// www.ok-labs.com / _assets / image_library / evoke.pdf* (2009).
[12] Barr, K., Bungale, P., Deasy, S., Gyuris, V., Hung, P., Newell, C., Tuch, H. and Zoppis, B.: The VMware mobile virtualization platform: is that a hypervisor in your pocket ?, *ACM SIGOPS Operating Systems Review*, pp. 124–135 (2010).
[13] Russello, G., Conti, M., Crispo, B. and Fernandes, E.: MOSES: supporting operation modes on smartphones, *Proceedings of the 17th ACM symposium on Access Control Models and Technologies*, pp. 3–12 (2012).
[14] Gardner, L. D. and Grigsby, J.: *Head First Mobile Web*, Oreilly & Associates Inc. (2011).
[15] Hickson, I.: A vocabulary and associated APIs for HTML and XHTML, *W3C Working Draft 29* (2012).
[16] Adobe Systems Inc.: PhoneGap, *http://phonegap.com/*.

© 2012 Information Processing Society of Japan