スマートデバイス向けアプリケーションのための共通アーキテクチャの提案

M2013SE006 黒田航平
担当教員：野呂昌満

1 はじめに

スマートデバイスの多様化に伴い，それらが前提とする実行時環境や開発環境も多様化している．フレームワークとその開発用のプログラミング言語を例にすると，Android SDK と Java，Cocoa Touch と Objective-C というように，実行時環境と開発環境は多岐にわたる．開発者は，既存の環境ごとにフレームワークや開発言語を調査，理解し，その使い方を習熟することが求められる．さらに要求の多様化や開発期間，開発予算などの制約により，開発者の負担は増加する．この問題に対して，ワンソースでの開発を可能とするクロスプラットフォーム開発環境の開発が行なわれている．しかしそれ自身も独自の環境を持ち，環境の差異に起因する問題の解決には至っていない．また，実行時環境はアプリケーションアーキテクチャにより定義され，開発環境は前提とする実行時環境と参照アーキテクチャが含意する開発プロセスにより規定される．したがって，これら環境の差異による問題はソフトウェアアーキテクチャに起因するものだと考える．

本研究の目的は，スマートデバイスアプリケーションのための共通アーキテクチャを提案することである．共通アーキテクチャは，既存の実行時環境が前提とするアプリケーションアーキテクチャ間が相互に対応付くことを保証する．開発者は共通アーキテクチャを参照することで，任意の環境についての理解を他の任意の環境での開発に活用できる．よって，環境の多様化による習熟コストの削減およびアプリケーションの仕様変更などの場面における理解容易性の確保に繋がる．

共通アーキテクチャは共通参照アーキテクチャと共通アプリケーションアーキテクチャの 2 つの視点に基づいて設計する．共通参照アーキテクチャの設計では，スマートデバイスアプリケーションがインタラクティブソフトウェアであることから，MVC アーキテクチャとその派生に基づく．これらのアーキテクチャが分離を試みている関心事を特定し，これをアスペクトとして分離したアスペクト指向アーキテクチャとして定義した．アスペクト間記述 (以下，IAD) とアスペクトとの関連を様々なビューから捉えることで，それぞれのアーキテクチャに矛盾なく対応付けることが可能となる．共通アプリケーションアーキテクチャは，共通参照アーキテクチャの詳細化により設計する．共通アーキテクチャがさまざまな実装技術に矛盾なく対応することを示し，共通参照アーキテクチャで定義されたアスペクトの詳細化によって柔軟性を担保する．共通参照アーキテクチャと MVC アーキテクチャとその派生，共通アプリケーションアーキテクチャと実行時環境の前提とするアプリケーションアーキテクチャそれぞれについて比較し，その関係を考察する．

2 関連研究

MVC アーキテクチャに基づく既存の実行時環境が前提とするアプリケーションアーキテクチャを分析し，共通参照アーキテクチャおよび共通アプリケーションアーキテクチャの設計において参考にした．既存の環境を共通的に扱うという本研究と同様の目的で開発されているクロスプラットフォーム開発環境についても調査を行なった．これについても，共通アプリケーションアーキテクチャの設計において参考にした．

2.1 スマートデバイスアプリケーション

Sokolova ら [1] は，Android SDK が MVC アーキテクチャに基づいて設計されていないとし，Android SDK のための MVC アーキテクチャとして Android Passive MVC を提案している．Android SDK が MVC アーキテクチャに基づいていない理由として，Android SDK が提供する実行時ライブラリである Activity が View と Controller を横断する構造であることが挙げられている．既存の MVC アーキテクチャとそのである PAC，MVP，HMVC，MVVM との比較を行ない，Android SDK のための MVC アーキテクチャについて検証されている．iOS SDK については，Mark ら [2] によって MVC アーキテクチャに基づいて設計されていると説明されている．しかし，iOS SDK が提供する実行時ライブラリである ViewController についても，View と Controller を横断する構造である．したがって本研究の見解では iOS SDK も MVC アーキテクチャに基づいて設計されていないものとする．

2.2 クロスプラットフォーム開発環境

クロスプラットフォーム開発環境はスマートデバイスアプリケーションをワンソースでの開発を可能としており，商用として数多く提供されている [3]．Monaca[4] や PhoneGap[5] に見られるように，これらは HTML や CSS，JavaScript などの標準の Web 技術を利用することで共通的な開発を支援している．

3 共通アーキテクチャ

共通アーキテクチャの設計指針とアーキテクチャの構成要素について説明する．共通アーキテクチャを 2 つの視点に基づいて設計したこと，アスペクト指向の適用により柔軟性の担保と実装技術に矛盾しない設計を実現したことを説明する．

3.1 共通アーキテクチャの設計指針

一般にリファレンスアーキテクチャに基づいてアプリケーションアーキテクチャが設計されることから，共通アーキテクチャは共通参照アーキテクチャと共通アプリ

ケーションアーキテクチャにの 2 つの視点から記述する．図 1 に共通アーキテクチャ設計についての概念図を示す．

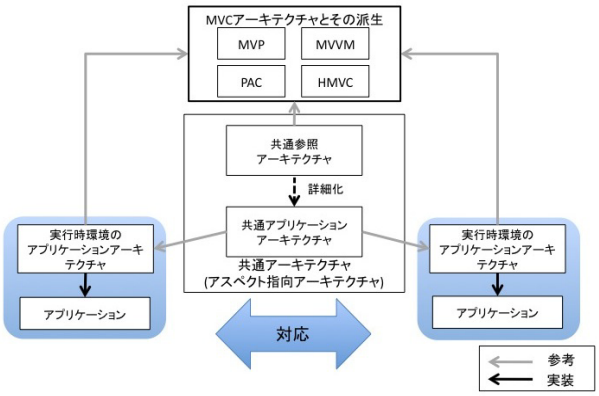

図 1　共通アーキテクチャ設計の概念図

共通参照アーキテクチャは，全ての MVC アーキテクチャとその派生に対応する．インタラクティブソフトウェアは一般に，MVC アーキテクチャとその派生に基づき設計される．したがって，共通参照アーキテクチャも MVC アーキテクチャとその派生である PAC，MVP，HMVC，MVVM に基づき設計する．MVC アーキテクチャとその派生が分離を試みている関心事を特定し，これをアスペクトとして分離したアスペクト指向アーキテクチャとして定義する．IAD とアスペクトの関連をさまざまな視点で捉えることで，それぞれのアーキテクチャに矛盾なく対応付けることが可能になる．

共通アプリケーションアーキテクチャは，共通参照アーキテクチャに基づき詳細化することにより設計する．詳細化は，共通参照アーキテクチャで定義したアスペクトに着目して行なう．これにより，共通アプリケーションアーキテクチャも共通参照アーキテクチャと同様に，アスペクト指向アーキテクチャとして定義したといえる．さらに，詳細化したアスペクトに基づき，実装を考慮した柔軟性を担保する．以下に，共通アプリケーションアーキテクチャが担保する柔軟性とその必要性を示す．

- Model
  - ビジネスロジックとデータ処理：画面構築との分離
  - 状態と状態におけるイベントに対応して実行される処理：アプリケーションの部分的な自動生成
- View
  - 内部表現と外部表現：モデルとのバインディングとその表現を分離
  - 内容，役割，見栄え：標準の Web 技術に基づく，部分的な再利用
- Controller
  - 内部イベント，外部イベント：イベントの形式を分離
- Model，View，Controller 間の関連
  - プッシュ型，プル型の切換え：既存の実装技術はどちらかに固定

3.2　共通参照アーキテクチャの設計

共通参照アーキテクチャの設計は，前節で述べた設計指針に基づき，アスペクト指向アーキテクチャとして定義した．これにより，複数の視点からそれぞれの MVC アーキテクチャとその派生が対応付く．図 2 に共通参照アーキテクチャを示し，表 1 にコンポーネントの責務を示す．

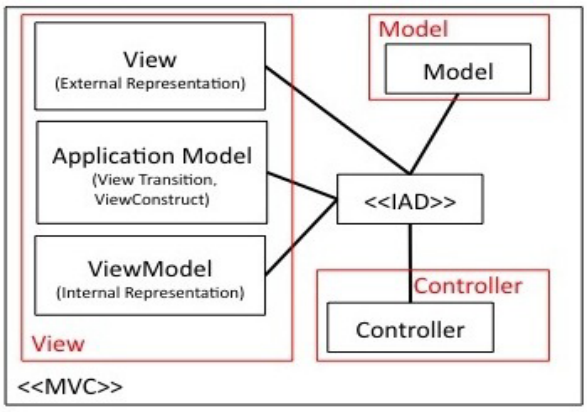

図 2　共通参照アーキテクチャ

表 1　共通参照アーキテクチャのコンポーネントの定義

コンポーネント	責務
Model	ビジネスロジック
	データ処理
View	ビジュアルコンポーネント
ApplicationModel	プレゼンテーションロジック
ViewModel	モデルへのデータバインディング
	内部表現と外部表現に基づく画面構成
Controller	イベントハンドリング

3.3　共通アプリケーションアーキテクチャの設計

共通アプリケーションアーキテクチャは共通参照アーキテクチャに基づき詳細化することで設計される．したがって，共通アプリケーションアーキテクチャも同様にアスペクト指向アーキテクチャとして定義される．また，実装を考慮したさいの柔軟性を担保するために，共通アプリケーションアーキテクチャで定義されるアスペクトは共通参照アーキテクチャで定義されたアスペクトの詳細である．図 3 に共通アプリケーションアーキテクチャを示し，表 2 にコンポーネントの役割を示す．また，共通アプリケーションアーキテクチャが共通参照アーキテクチャから矛盾なく詳細化されていることを確認するために，その対応関係を図 4 に示す．

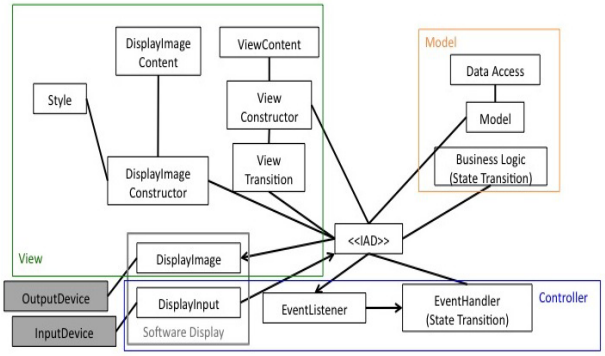

図 3 共通アプリケーションアーキテクチャ

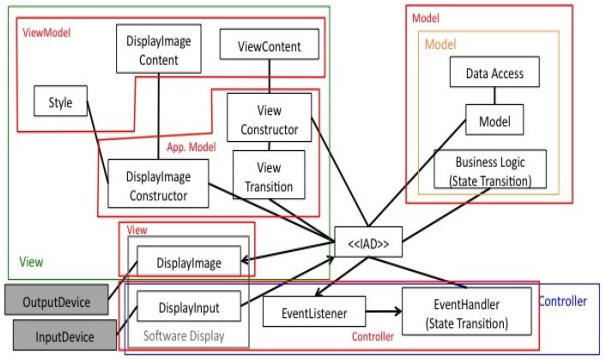

図 4 共通参照アーキテクチャと共通アプリケーションアーキテクチャの対応関係

表 2 共通アプリケーションアーキテクチャのコンポーネントの定義

コンポーネント	責務
DisplayInput	操作対象ウィジェットを検知
EventListener	外部，内部のイベント変換
EventHandler	メッセージの振分け
BusinessLogic	ビジネスロジック
Model	データ構造
Data Access	OR マッピング
ViewContent	内容
DisplayImageContent	役割
Style	見た目
ViewConstructor	内部表現のためのプレゼンテーションロジック
DisplayImageConstructor	外部表現のためのプレゼンテーションロジック
DisplayImage	レンダリングのためのファイル

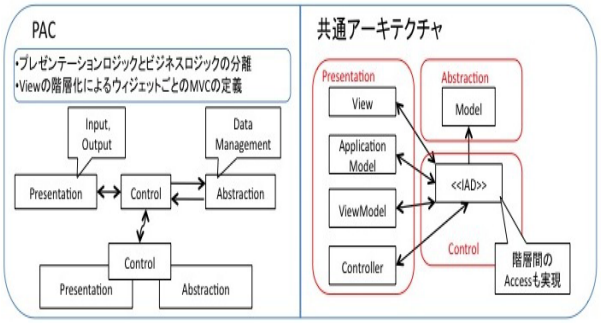

図 5 共通参照アーキテクチャと PAC アーキテクチャの対応関係

4 共通アーキテクチャと既存のアーキテクチャの対応関係

共通アーキテクチャが適切に定義され，全てのスマートデバイスアプリケーションを説明可能であることを検証するために，既存のアーキテクチャとの対応関係を整理する．これらの対応関係について構造と役割で適切に示すことができれば，共通アーキテクチャをスマートデバイスアプリケーションのための共通アーキテクチャとして利用可能であると考える．

4.1 共通参照アーキテクチャと MVC アーキテクチャとそのとの対応関係

共通参照アーキテクチャと MVC アーキテクチャとそのであるアーキテクチャとの対応関係を整理し，既存の参照アーキテクチャが分離を試みている関心事を考慮したアスペクト指向アーキテクチャであることを考察する．本稿では PAC アーキテクチャを例に取り上げ，対応関係を説明する．図 5 に，共通参照アーキテクチャと PAC アーキテクチャの対応関係を示す．

4.2 共通アプリケーションアーキテクチャと実行時環境のアプリケーションアーキテクチャとの対応関係

共通参照アーキテクチャが MVC アーキテクチャを説明可能であるという前提のもと，共通アプリケーションアーキテクチャが実行時環境のアプリケーションアーキテクチャについて説明可能であることを示す．共通アプリケーションアーキテクチャのアスペクトは，前提とする共通参照アーキテクチャのアスペクトの詳細である．したがって，共通アプリケーションアーキテクチャが定義するアスペクトと実行時環境のアプリケーションアーキテクチャのコンポーネントの役割についての同一性に基づき対応関係を整理する．共通アプリケーションアーキテクチャは，IAD との関連によって複数のアスペクトのモジュール化が可能である．したがって，共通アプリケーションアーキテクチャと実行時環境のアプリケーションアーキテクチャの対応関係は多対多の関係になることもある．図 6 に Android のアプリケーションアーキテクチャに基づく実装例との対応関係，図 7 に iOS のアプリケーションアーキテクチャに基づく実装例との対応関係を示す．

5 考察

5.1 共通アーキテクチャと既存のアーキテクチャとの対応関係による考察

共通参照アーキテクチャは，既存の参照アーキテクチャである MVC アーキテクチャとそのであるアーキテクチャが分離を試みている関心事をアスペクトとして定義し，IAD との関連を定義することで対応付けが可能であることを示した．共通アプリケーションアーキテクチャのアスペクトが共通参照アーキテクチャのアスペクトの詳細であり，また多様な要求を実現するための柔軟性を担保

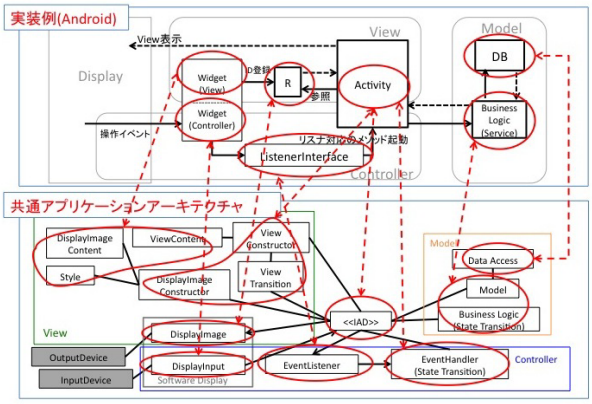

図 6 Android のアプリケーションアーキテクチャに基づく実装例との対応関係

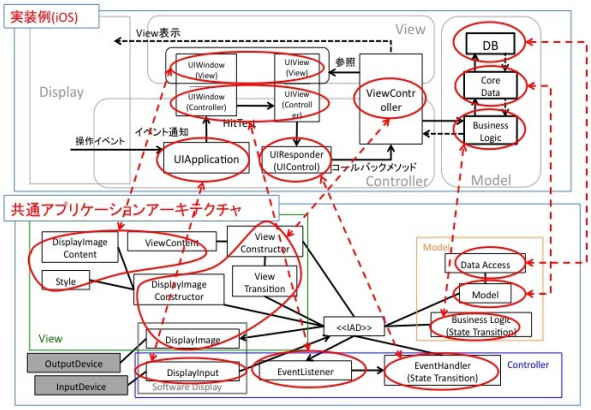

図 7 iOS のアプリケーションアーキテクチャに基づく実装例との対応関係

し，かつ，実装技術と矛盾なく対応できることを示した．

5.2 共通アーキテクチャの利用手段

共通アーキテクチャが，MVC アーキテクチャに基づく全てのスマートデバイスアプリケーションの共通アーキテクチャとして利用が可能であることを考察する．スマートデバイスの多様化に伴う実行時環境や開発環境の多様化による習熟コストが増加する問題の解として提案する．図 8 に，共通アーキテクチャの共通アーキテクチャとしての利用の概要を示す．開発者は，共通アーキテクチャが全てのスマートデバイスアプリケーションを説明可能であることから，任意の環境についての理解を任意の環境での開発に活用することが可能となる．

6 おわりに

スマートデバイスの多様化に伴う実行時環境と開発環境の多様化について，共通アーキテクチャの提案による解決を試みた．共通アーキテクチャは，全ての実行時環境のアプリケーションアーキテクチャについて，構造と役割の対応関係から説明可能である．アーキテクチャ間の対応関係の根拠を示すために，MVC アーキテクチャとその派生を参考に共通参照アーキテクチャを設計した．既

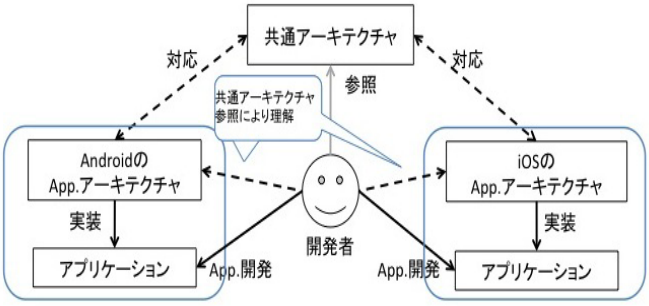

図 8 共通アーキテクチャとしての共通アーキテクチャ

存の MVC アーキテクチャとその派生であるアーキテクチャを参照アーキテクチャとし，これらが分離を試みている関心事を特定してアスペクト指向アーキテクチャとして定義した．共通アーキテクチャが多様な要求に対応するための柔軟性を担保するために，共通参照アーキテクチャを詳細化し，共通アプリケーションアーキテクチャを設計した，さらに，共通参照アーキテクチャと MVC アーキテクチャとその派生との対応関係を考察し，共通参照アーキテクチャがインタラクティブソフトウェアのための参照アーキテクチャであることを確認した．共通アーキテクチャがさまざまな実装技術と矛盾なく対応することを確認するために，共通アプリケーションアーキテクチャと実行時環境のアプリケーションアーキテクチャとの対応関係を確認した．最後に，共通アーキテクチャの参照アーキテクチャとしての利用手段を考察し，共通アーキテクチャが環境間の差異を解消することを示した．今後の課題として，共通アーキテクチャを全てのスマートデバイスアプリケーションの共通アーキテクチャとして利用できることを目的とし，残る全ての実行時環境のアプリケーションアーキテクチャについて考察することが挙げられる．

参考文献

[1] K. Sokolova, M. Lemercier, and L. Garcia, ”Android Passive MVC: a Novel Architecture Model for Android Application Development, ” *PATTERNS 2013: The Fifth International Conferences on Pervasive Patterns and Application*, pp. 7-12, 2013.

[2] D. Mark, and J. LaMarche, *More IPhone 3 Development, ser. Tacking IPhone Sdk 3. Berkely: Apress*, Jan. 2010.

[3] S. Allen, V. Graupera, and L. Lundrigan, *Pro Smartphone Cross-Platform Development: IPhone, Blackberry, Windows Mobile and Android Development and Distribution*, 1st ed. Berkely: Appress, Sep. 2010.

[4] アシアル株式会社,“ *Monaca* ドキュメント，” http://docs.monaca.mobi/3.5/ja/，2015.

[5] アシアル株式会社，PhoneGap 入門ガイド，翔泳社，2011.