携帯電話システムにおけるファイルシステムのアスペクト指向実現

2001MT006 坂野 将秀　　2001MT018 長谷川 智美　　2001MT053 小林 鋼司
指導教員　野呂 昌満

1　はじめに

多機能化が進む近年の携帯電話は，ファイルシステムを持っているのが一般的である．携帯電話のような組込み機器においては，記憶領域の大きさが限られているので，汎用計算機と同様のファイルシステムを適用することは困難である．一方で，携帯電話などの組込み機器は，開発サイクルが短いので，それらを制御するソフトウェアは柔軟性が必要となる．実現の方法として，柔軟な構造を構築できると言われているオブジェクト指向の適用が考えられる．

我々は，組込み機器である携帯電話のファイルシステムにおいて，オブジェクト指向を用いるとコンサーン横断問題 [4] が発生することを確認した．携帯電話特有のファイルシステムに関する処理 (メモリ制限など) が散在している構造は，再利用性が低い．携帯電話のファイルシステムの再利用性を向上させるためには，横断的に関連している処理を分離する必要がある．

本研究の目的は，アスペクト指向で携帯電話のファイルシステムを実現し，変更における柔軟性，および再利用性を考察することである．POP(Post-Object Programming) 技術のひとつであるアスペクト指向により，携帯電話における横断的に関連している処理を分離できると考えた．

本研究では，携帯電話システムをオブジェクト指向で実現し，アスペクト指向によりコンサーンに関するアスペクトの分離をおこなった．抽出したコンサーンは，データ永続性，およびメモリストレージ，インスタンス生成，例外処理である．我々は，横断的な処理を分離したことにより，柔軟性の高い携帯電話におけるファイルシステムを開発することができた．

小林はファイルシステム，坂野はアドレスシステム，長谷川はカメラシステムを主に担当した．

2　オブジェクト指向による開発

携帯電話のファイルシステムを，オブジェクト指向で設計し，コンサーン横断問題が発生することを確認した．携帯電話のファイルシステム，およびデータを扱うサブシステムを実現した．本節では，実現したファイルシステム，および実現したサブシステムの構造を述べる．オブジェクト指向による携帯電話のファイルシステムを開発する際，一般的にオブジェクト指向言語として広く用いられている Java [3] を使用した．

2.1　ファイルシステムの実現

オブジェクト指向で実現したファイルシステムについて述べる．実現には，デザインパターン [1] のひとつである，Singleton パターン，および Composite パターン，Command パターンを用いた．ファイルシステムは，整合性をとるために，Singleton パターンを用いている．また，データの多相性を実現するために，Composite パターンを用いて階層化した．Command パターンにより，要求をオブジェクトとしてカプセル化する．実現したファイルシステムの構造を**図 1** に示す．

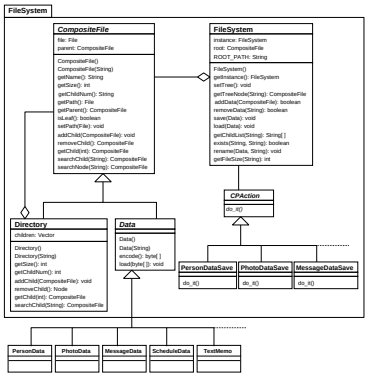

図 1　ファイルシステムのクラス図

2.2　ファイルシステムを扱うサブシステムの実現

ファイルシステムを扱うサブシステムとして，アドレスシステム，およびカメラシステムを例にとり，オブジェクト指向により実現した．実現したアドレスシステム，およびカメラシステムの構造を**図 2**，**図 3** に示す．

アドレスシステム

我々が実現したアドレスシステムについて述べる．個人情報を複数のオブジェクトの集合として持つアドレスシステムを実現した．個人情報データには永続性を持たせる必要があるので，アドレスシステムはファイルシステムと関連がある．

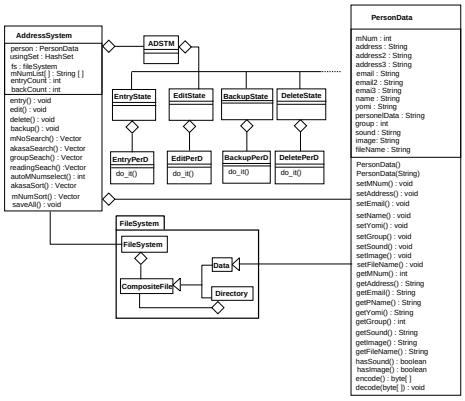

図 2　アドレスシステムのクラス図

カメラシステム

我々が実現したカメラシステムについて述べる．実現したカメラシステムの構造は，デジタルカメラの仕組みをもとに実現したので，カメラシステムがデータを保持している．撮影した画像には永続性を持たせる必要があるので，カメラシステムはファイルシステムと関連がある．

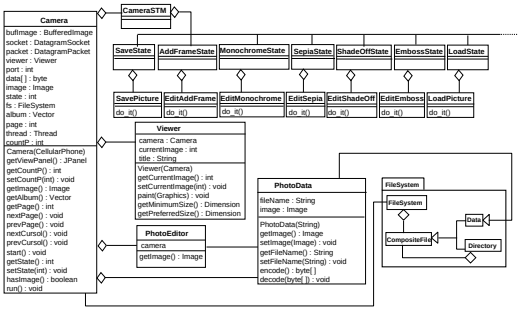

図 3　カメラシステムのクラス図

2.3　一般のファイルシステムとの比較

実現した携帯電話のファイルシステムでは，携帯電話特有の処理が散在しているので，再利用性が低い．携帯電話特有の処理を分離し，再利用性を向上させる．

実現したファイルシステムにおいて，横断的に関連している処理を確認した．確認した処理は，データに永続性を持たせる処理，メモリを制限する処理，インスタンス生成を制限する処理，例外処理である．確認した処理の詳細を次章に示す．

3　アスペクト指向による開発

実現したアドレスシステム，およびカメラシステム，ファイルシステムからコンサーンを抽出し，アスペクト指向により実現する．抽出したコンサーンは，データ永続性，メモリストレージ，インスタンス生成，例外処理である．抽出したコンサーンに関するアスペクトを分離することでシステムの柔軟性を向上させる．今回実現するシステムでは，Java で作成したプログラムを利用するために AspectJ[2] を用いる．

携帯電話特有のコンサーンに関するアスペクトの分離をおこなうことで，システムの再利用性を向上させる．携帯電話特有のコンサーンとは，メモリストレージなどのことである．一般のファイルシステムと携帯電話のファイルシステムの共通部分は変更せずに，特有な部分を分離することでファイルシステムの汎用性を向上させる．

3.1　データ永続性コンサーン

携帯電話は，データをファイルに保存するという方法でデータに永続性を持たせる必要がある．アドレスシステムをオブジェクト指向により実現した構造では，個人情報のデータに永続性を持たせる処理が散在してしまう．データに永続性を持たせる処理が，個人情報の新規登録時，変更時に横断的に関連している．複数のオブジェクト間を横断する処理を**図 4** に示す．

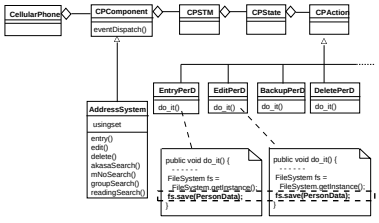

図 4　データ永続性に関する処理の詳細

データに永続性を持たせる処理の記述は，アスペクト指向により分離できる．複数のオブジェクト間に横断する処理を分離することにより，変更の影響が局所化される．データの保存を必要とする処理が追加されても，データを保存する処理を記述する箇所は局所的である．実現したアスペクト間記述を**図 5** に示す．

```
pointcut atPersonDataSavePoint():
 execution(void AddressSystem.entry(..));
 after():atPersonDataSavePoint(){
  AddressSystem trg =
  (AddressSystem)thisJoinPoint.getThis();
  PersonData p = trg.getTargetPersonData();
  FileSystem.getInstance().save(p);
}
```

図 5　データ永続性コンサーンに関するアスペクト間記述

アドレスシステムにおける，データの永続性コンサーンに関するアスペクトを分離することができた．ジョインポイントは，データに永続性を持たせる必要がある登録・変更のメソッドである．

3.2　メモリストレージコンサーン

携帯電話は記憶領域が限られているので，メモリの使用を制限する必要がある．カメラシステムをオブジェクト指向により実現した構造では，メモリを制限する処理が散在してしまう．メモリを制限する処理が，撮影した画像の登録，編集時に横断的に関連している．複数のオブジェクト間を横断する処理を**図 6** に示す．

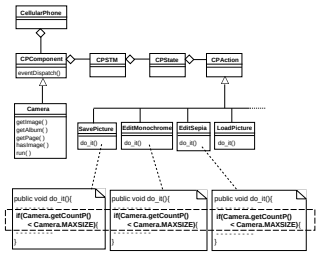

図 6　メモリストレージに関する処理の詳細

メモリを制限する処理の記述は，アスペクト指向により分離できる．複数のオブジェクト間を横断する処理を分離することにより，変更の影響が局所化される．メモリの制限を必要とする処理が追加されても，メモリを制限する処理を記述する箇所は局所的である．実現したアスペクト間記述を**図 7** に示す．

```
pointcut atPhotoDataMemoryManager():
 within(camera.*) &&
 execution(void SavePicture.do_it());
 void around():atPhotoDataMemoryManager(){
  CameraContextFS cfs = new CameraContextFS();
 if(cfs.getFileSize(PATH)<memoryMAX){
   proceed();
  }else Camera.exception();
}
```

図 7　メモリストレージコンサーンに関するアスペクト間記述

カメラシステムにおける，メモリストレージコンサーンに関するアスペクトを分離することができた．ジョインポイントはデータをファイルに残す処理があるメソッドである．

3.3 インスタンス生成コンサーン

ファイルシステムは整合性をとるために，インスタンスがひとつしか生成されないことを保証する必要がある．オブジェクト指向による実現では，Singleton パターンによりファイルシステムのインスタンス生成を制限している．インスタンス生成時には，インスタンス生成を制限する getInstance() メソッドを呼び出す必要がある．アドレスシステムでは，ファイルシステムのインスタンスを生成する処理が，個人情報の新規登録時，変更時に横断的に関連している．複数のオブジェクト間に横断する処理を**図 8** に示す．

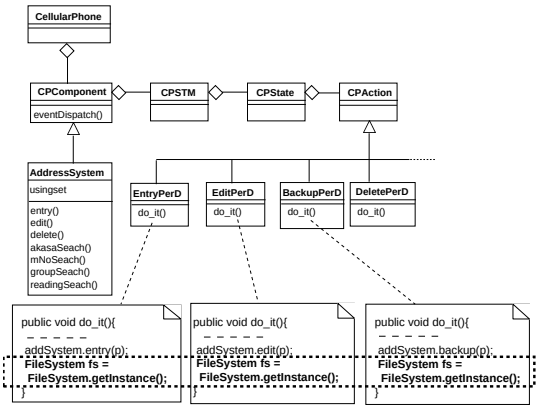

図 8　インスタンス生成に関する処理の詳細

インスタンス生成を制限する処理の記述は，アスペクト指向により分離できる．実現したアスペクト間記述を**図 9** に示す．

```
pointcut newInstance():
  !within(filesystem.*) &&
  call(FileSystem.new());
private static final FileSystem
FileSystem.instance = new FileSystem();
FileSystem around(): newInstance(){
 return FileSystem.instance;
}
```

図 9　インスタンス生成コンサーンに関するアスペクト間記述

アドレスシステムにおける，インスタンス生成コンサーンに関するアスペクトを分離することができた．ジョインポイントはファイルシステムのインスタンス生成の処理をおこなうメソッドである．

3.4 例外処理コンサーン

本研究では，例外処理を Java 言語の Exception を用いず実現した．組込みソフトウェアでは例外が発生した際に，明確におこなう処理が決まっていないと，致命的な欠陥となる可能性がある．例外が発生した際に，明確におこなう処理を決定するには，例外が発生する可能性がある全ての処理に try，catch を記述しなければならない．例外処理を Java 言語の Exception を用いず記述し，アスペクト指向により分離し，局所化する．

携帯電話は記憶領域が限られているので，画像，および音楽，個人情報などの登録をする際，メモリ残量がないと例外処理をする必要がある．実現したアドレスシステムの構造では，メモリ不足時に新規登録，バックアップの処理をおこなった際の例外処理が横断的に関連している．複数のオブジェクト間に横断する処理を**図 10** に示す．

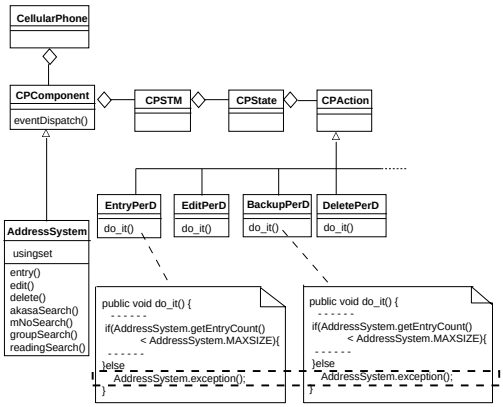

図 10　例外処理に関する処理の詳細

例外処理をする処理の記述は，アスペクト指向により分離できる．実現したアスペクト間記述を**図 11** に示す．

```
pointcut addrException():
  within(address.*) &&
   call(AddressSystem.exception(..));
void around(): addrException(){
 return AddressSystem.toIdle();
}
```

図 11　例外処理コンサーンに関するアスペクト間記述

アドレスシステムにおける，例外処理コンサーンに関するアスペクトを分離することができた．ジョインポイントは，データを保存する処理の中でメモリ残量がない場合におこなう例外処理のメソッドである．

4 考察

アスペクト指向で実現したファイルシステムを用いて柔軟性，および再利用性について考察する．本節では，コンサーン同士の関連の整理をおこなった後，実現したファイルシステムについて議論する．柔軟性，および再利用性が向上したことを示すために，オブジェクト指向で実現した構造と，アスペクト指向で実現した構造の比較をする．

4.1 コンサーンに関するアスペクト同士の関連

本研究では，分離したコンサーンに関するアスペクト同士にも関連がある事を確認した．コンサーン同士に関連があると変更における柔軟性が低下するので，分離することでモジュール性が高まると考えた．

メモリストレージコンサーンとデータ永続性コンサーン，例外処理コンサーンの関連

データを保存する際にメモリの残量がなければ，例外処理が必要である．データ永続性コンサーンに関するアスペクトの中に，メモリストレージコンサーンに関する処理が存在する．また，メモリストレージコンサーンに関するアスペクトの中に，例外処理コンサーンに関する処理が存在する．コンサーンに関するアスペクト同士の関連を**図 12** に示す．

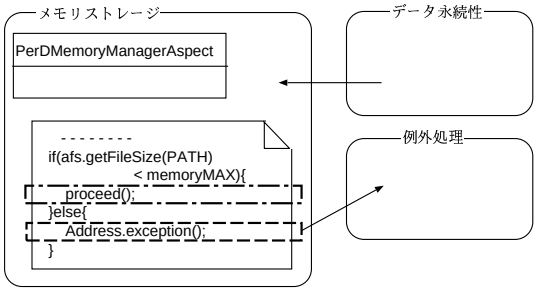

図 12　コンサーンに関するアスペクト同士の関連

データ永続性コンサーンとインスタンス生成コンサーンの関連

データを保存する際，ファイルシステムのインスタンス生成が必要である．データ永続性コンサーンに関するアスペクトの中にインスタンス生成コンサーンに関する処理が存在する．コンサーンに関するアスペクト同士の関連を**図 13** に示す．

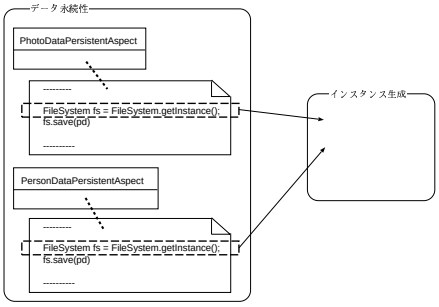

図 13　コンサーンに関するアスペクト同士の関連

4.2 アスペクト指向開発による利点

コンサーン横断問題に対して，アスペクト指向を適用した利点について述べる．本研究では，データ永続性，メモリストレージ，インスタンス生成，例外処理をモジュール分割した．変更の影響が局所化される構造を構築することができた．例えば，ファイルシステムのデータの保存をおこなうアクションの中にメモリの上限を超えていないか確認する confirm() メソッドを加えても，**図 14** のように記述する箇所が局所的である．また，保存するデータの項目が追加されても，変更されたデータクラスの encode() メソッドの変更のみである．既存のシステムでデータの保存を必要とする処理が追加されても，ジョインポイントとして指定するだけで実現できる．

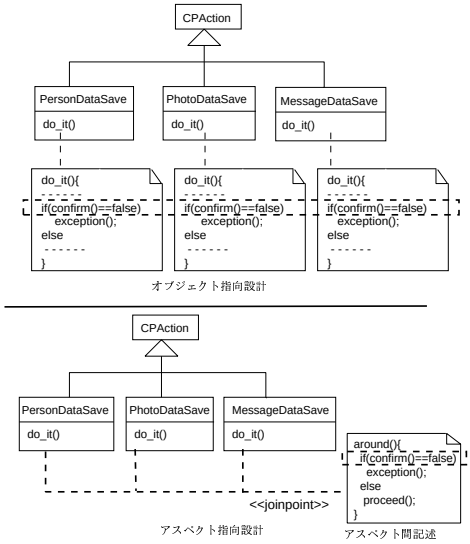

図 14　オブジェクト指向設計とアスペクト指向設計の比較

5 おわりに

本研究では，携帯電話のファイルシステムをアスペクト指向により実現することで柔軟性の高いシステムを構築した．アドレスシステムとカメラシステムを例に，システムの変更における柔軟性について考察した．アスペクト指向を用いることで変更における柔軟性が向上することを確認した．

今後の課題は，本研究で実現した構造を抽象化し，他の組込み機器に適用できるアーキテクチャを構築することである．

6 謝辞

本研究を進めるにあたり，二年間熱心に御指導をいただいた野呂昌満教授，有益なアドバイスをいただいた熊崎敦司先生，大学院生の後藤修平さん，石見知也さん，八木晴信さん，小久保佳将さんに深く感謝いたします．また，二年間をともに励まし支え合い頑張ってきた野呂研究室一同の方々にも感謝いたします．

参考文献

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vissides: Design Patterns Elements of Reusable Object-Oriented Software, AddisonWesley，p.395 (1995).
[2] AspectJ http://eclipse.org/aspectj/
[3] Java http://java.sun.com/
[4] COMMUNICATIONS of the ACM, Vol.44, No.10, p.168 (2001).