携帯電話における通信ソフトウェアのアスペクト指向実現

2000MT020 本多 克典　2000MT030 伊藤 裕康　2001MT001 安形 知美　2001MT079 緒方 秀次
指導教員　野呂 昌満

1　はじめに

携帯電話における通信ソフトウェアは，通信事業者で異なるプロトコルやプロトコルの改訂に対応し，開発をおこなわなければならない．各通信事業者は，一般の計算機で利用されている POP や HTTP を使用せず各通信事業者独自の通信プロトコルを使用している．既存の通信ソフトウェアのプロトコルを容易に変更するためには，プロトコル変更に柔軟に対応できる構造が求められる．柔軟性の高いソフトウェアの構築方法としてオブジェクト指向による開発が有効であるとされているが，複数のクラスに横断的に関連する処理が存在する問題 [4] が指摘されている．

本研究の目的は，アスペクト指向の適用により，組込みソフトウェアにおいて高い柔軟性と再利用性をもつ通信ソフトウェアの構造を考察することである．プロトコル処理と横断的に関連する処理をアスペクト指向で分離することで，通信プロトコルの変更に柔軟に対応できる通信ソフトウェアを開発できると考える．本研究では，アプリケーション層の通信プロトコルを対象として研究をおこなった．

我々は携帯電話の通信ソフトウェアをオブジェクト指向技術を用いて試作し，プロトコル処理と横断的に関連している処理を確認した．横断的に関連する処理をアスペクト指向技術の適用により分離することで，高い柔軟性と再利用性をもつソフトウェアとしてモジュール化する．オブジェクト指向とアスペクト指向での通信ソフトウェアの実現を比較し，通信プロトコルの変更に対する柔軟性と再利用性について考察する．複数の処理を並行におこなうことを組込みソフトウェアの本質と考え，組込みソフトウェアの典型的な例として携帯電話を取り上げた．本研究では携帯電話の通信ソフトウェアとして web 閲覧システム，通話システム，メイルシステムを題材とした．

本多，伊藤は主に web 閲覧システムを，安形は主に通話システムを，緒方は主にメイルシステムを担当した．

2　オブジェクト指向による開発

携帯電話における通信ソフトウェアをオブジェクト指向で設計，実現し，横断的に関連する処理が存在することを確認する．本研究で題材とする通信ソフトウェアの基本構造は同じであると考え，携帯電話における通信ソフトウェアの共通構造を設計する．設計した構造をもとに，メイルシステム，web 閲覧システム，通話システムの設計，実現をおこなう．

2.1　通信ソフトウェアに共通な構造の設計

携帯電話における各通信ソフトウェアの共通部分を，オブジェクト指向で設計した．通信ソフトウェアをサーバとの通信処理をおこなう部分と，通信処理以外の処理をおこなう部分に分離した．各通信ソフトウェアは複数の状態をもつ．状態遷移機械を実現させるために State パターン [2] を用いた．また状態遷移時に，おこなわれる要求をひとつのオブジェクトとしてカプセル化するために Command パターン [2] を適用した．設計したクラス図を**図 1** に示す．

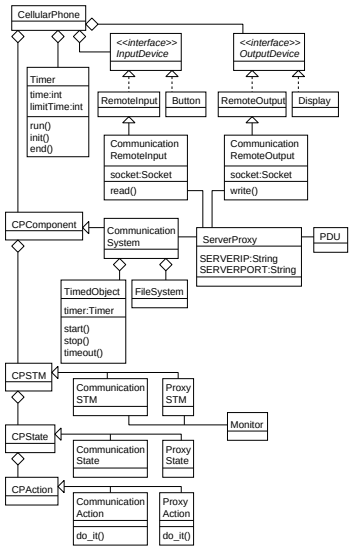

図 1　通信ソフトウェアにおける共通構造のクラス図

図 1 に示したクラスとその役割は以下のとおりである．

- CommunicationRemoteInput
  遠隔サーバからデータを受信する
- CommunicationRemoteOutput
  遠隔サーバにデータを送信する
- ServerProxy
  サーバの代理として振舞う
- PDU
  プロトコル処理で扱うデータ
- Timer
  時間計測をおこなう
- FileSystem
  データの保存，呼び出しをおこなう
- Monitor
  入力と状態遷移機械を監視する
- CommunicationSTM
  通信ソフトウェアの状態遷移機械
- CommunicationState
  通信ソフトウェアの各状態
- CommunicationAction
  通信ソフトウェアで状態遷移のさいに実行される振舞い

2.2　各システムのオブジェクト指向による設計と実現

図 1 で示した通信ソフトウェアの共通構造をメイルシステム，web 閲覧システム，通話システムに適用する．

電気通信事業者や携帯電話製造メーカーに共通のユースケースを，携帯電話の取扱説明書などを参考に分析し，各通信ソフトウェアをオブジェクト指向で設計，実現した．設計，実現した通信ソフトウェアの一例としてメイルシステムのクラス図を**図 2** に示す．本研究で実現したメイルシステムでは特定のサーバと通信をおこなう．サーバの通信をおこなう部分のインタフェースを Proxy クラスに記述した．

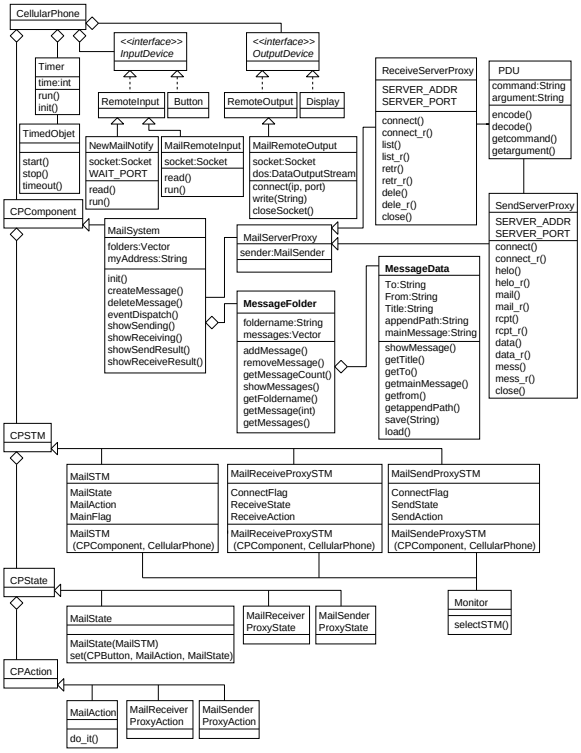

図 2　メイルシステムのクラス図

	クラス	クラスの説明
メイル システム	MessageFolder MessageData	送受信メッセージを管理 送受信メッセージを 表す
web閲覧 システム	BookmarkEditor Bookmark	URL情報を管理 URL情報を保持
通話 システム	HistoryEditor History CPMicrophone CPSpeaker	発着信履歴を管理 発着信履歴情報を保持 音声入力装置 音声出力装置

図 3　各通信ソフトウェア独自のクラス

図 1 で示した通信ソフトウェアに共通のクラス以外に，各通信ソフトウェアは独自のクラスをもつ．各通信ソフトウェア独自のクラスを**図 3** に示す．
各通信ソフトウェアに実時間処理，セキュリティ処理などの横断的に関連する処理の存在を確認した．

3　アスペクト指向による開発

携帯電話における通信ソフトウェアをアスペクト指向で設計する．複数のオブジェクトと横断的に関連している処理を，オブジェクト指向で設計した通信ソフトウェアから抽出し，構造を整理する．これに加えて PDU は別の指針である協調場 [3] による分割にしたがい分割する．

3.1　コンサーンの抽出

オブジェクト指向による設計をおこなった携帯電話の通信ソフトウェアから，複数のオブジェクトと横断的に関連している処理をコンサーンとして抽出した．抽出したコンサーンを以下に示す．

- 実時間処理コンサーン
- セキュリティ処理コンサーン
- 例外処理コンサーン
- データ永続性コンサーン
- 並行処理コンサーン
- イベント処理コンサーン
- 外部出力コンサーン
- コアコンサーン

実時間処理コンサーン

実時間処理として，サーバからのレスポンスタイム，通話システムのダイヤル後の時間，通話時間の計測をおこなう．実時間処理は時間計測をおこなう複数のクラスに横断的に関連する．レスポンスタイムの計測では，メッセージを送信し，応答待ちの状態へ遷移するときに計測を開始する．サーバからの応答が来たら計測を終了する．一定時間内にサーバからの応答がなく状態が遷移しない場合，例外を発生させる．例外処理コンサーン，イベント処理コンサーン，コアコンサーンと関連をもつ．

セキュリティコンサーン

セキュリティ処理は，パケット盗聴に関してのプライバシー保護のための暗号化と復号化をおこなう．SSL などの通信プロトコルレベルと，APOP などのアプリケーションレベルでの暗号化，復号化が存在する．通信プロトコルレベルではサーバとのメッセージ送受信をおこなう RemoteInput クラスと RemoteOutput クラスに横断的に関連する．アプリケーションレベルでは，セキュリティ処理が必要なコマンドを送信する Action に関連する．サーバから受信したメッセージを復号化できない場合，例外を発生させる．例外処理コンサーン，イベント処理コンサーン，外部出力コンサーンと関連をもつ．

例外処理コンサーン

例外処理として，携帯電話利用者の例外入力や，通信に関する例外に対する処理がある．例外処理をおこなう通信ソフトウェアの複数のクラスと横断的に関連する．利用者の例外入力とは，web 閲覧システムの閲覧したい web ページの URL 指定で，決められている構文に合わない URL を指定したときなどが該当する．通信に関する例外とは，実時間処理の送受信時間切れや，復号化の失敗などが該当する．セキュリティ処理コンサーン，実時間処理コンサーン，イベント処理コンサーン，コアコンサーンと関連をもつ．

データ永続性コンサーン

データ永続性に関する処理では，携帯電話の電源を切ってもデータを残すために，データをファイルに保存する．データ永続性に関する処理は，送受信メッセージを管理

するクラスと，ブックマークを管理するクラス，発着信履歴を管理するクラスに横断的に関連している．

コアコンサーン

コアコンサーンは受け取ったイベントに対応する処理をおこない，状態を遷移させる．コアコンサーン内でさらに，サーバとの通信処理の状態遷移と，通信処理以外の状態遷移を分離した．同様に通信処理のアプリケーションロジックと，通信処理以外のアプリケーションロジックを分離した．

3.2 PDU の協調場による分割

複数の役割が存在する PDU を，横断的に関連する処理の分離とは別の指針である協調場による分割をおこなう．PDU はプロトコル処理中の場所によって役割を変える．今回実現した通信ソフトウェアの PDU には以下の役割を確認した．

- LocalChar：ユーザからの入力文字列
- Com：サーバへ出力するメッセージ
- Security：暗号化，復号化をおこなうメッセージ
- RemoteChar：サーバからの入力文字列
- Res：サーバからの応答メッセージ

オブジェクト指向実現における PDU クラスには複数の役割が存在するので，変更が容易ではないなどの問題ある．本研究では，複数の役割が存在することによる問題を解決するために，プロトコル処理中の場所を協調場として，PDU クラスを分割する．

3.3 アスペクト指向による設計

抽出したコンサーンに基づいてアスペクトを分離した．分離したアスペクトとアスペクト間の関連を**図 4** に示す．

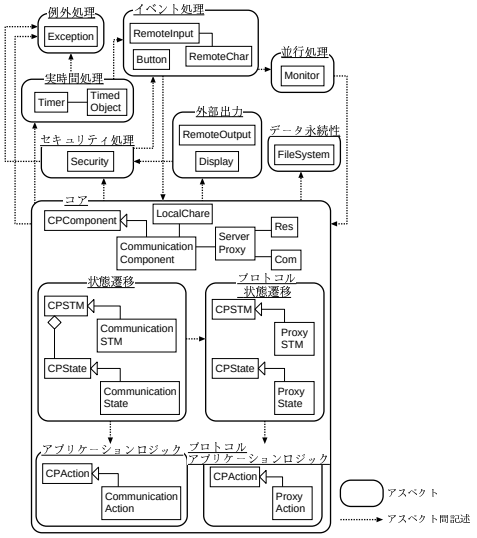

図 4　アスペクト間の関連図

4 アスペクト指向による実現と考察

通信ソフトウェアをアスペクト指向実現し，オブジェクト指向実現した通信ソフトウェアと比較する．通信プロトコルの変更に対する通信ソフトウェアの柔軟性と再利用性について考察する．アスペクト指向実現には AspectJ[1] を用いた．

4.1 実時間処理のアスペクト指向実現と考察

アスペクト指向により，時間計測に関する処理をプロトコル処理から分離し，実時間処理とプロトコル処理の独立性を高める．オブジェクト指向による実現とアスペクト指向による実現を比較し，考察をおこなう．

オブジェクト指向実現では時間計測に関する記述が複数のクラスに散在している．通信ソフトウェアはサーバからのレスポンスタイムの計測を各コマンドに応じておこなう．通信プロトコルの変更にともないコマンドの追加，変更をおこなうさいプロトコル処理をおこなうオブジェクトに時間計測に関する記述も必要となる．メイルシステムの各コマンドの送信を例にとり，時間計測に関する処理を記述をした様子を**図 5** に示す．

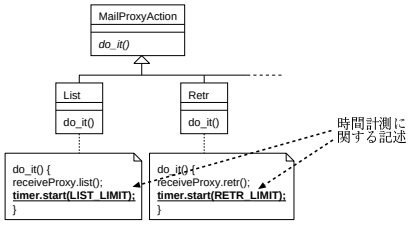

図 5　オブジェクト指向による時間計測

アスペクト指向によって時間計測に関する処理を分離した．アスペクト指向による実時間処理の実現を**図 6** に示す．時間計測に関する記述を実時間処理アスペクト内に局所化した．通信プロトコルの変更にともないコマンドの追加，変更をおこなうさいプロトコル処理の中に時間計測に関する記述を必要としない．プロトコル処理をおこなうオブジェクト内に時間計測に関する記述がなくなったので，柔軟性と再利用性が向上したと言える．

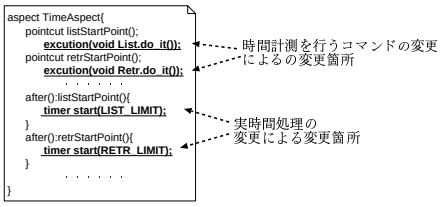

図 6　アスペクト指向による実時間処理の実現

4.2 セキュリティ処理のアスペクト指向実現と考察

オブジェクト指向とアスペクト指向でのセキュリティ処理の実現を比較し，セキュリティ処理と通信プロトコルの変更に対する柔軟性と再利用性について考察する．

オブジェクト指向による実現では，アプリケーションレベルでのセキュリティ処理に関する記述がプロトコル処理をおこなうクラスに複数存在する．セキュリティ処理を変更するさいに，複数のクラスで変更が必要になるので，変更に対する柔軟性が低い．メイルシステムにおけるセキュリティ処理の変更による変更箇所を**図 7** に示

す．また，プロトコル処理をおこなうオブジェクトは，通信プロトコルに依存する記述が含まれるので，再利用性に優れているとは言えない．

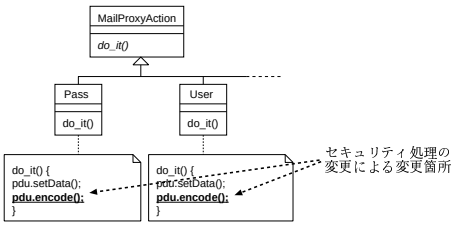

図 7　オブジェクト指向によるセキュリティ処理の実現

アスペクト指向の適用により，セキュリティ処理に関する処理を分離した．セキュリティ処理のインタフェースなどの変更による影響はアスペクト間記述の advice 内に局所化される．通信プロトコルの変更によるセキュリティ処理の付加は joinpoint の指定によっておこなえる．メイルシステムにおけるセキュリティ処理の変更と通信プロトコル変更のさいの変更箇所を**図 8** に示す．セキュリティ処理に関する変更の影響は全てセキュリティ処理アスペクト内に局所化されたので，柔軟性が向上した．また，プロトコル処理をおこなうオブジェクトは，通信プロトコルに依存するセキュリティ処理の記述が無くなったので，再利用性が向上したと言える．

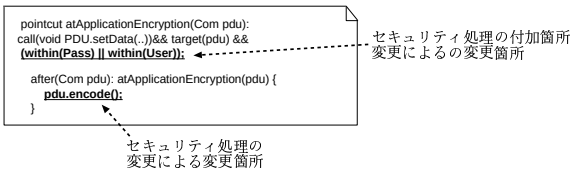

図 8　セキュリティ処理アスペクトのアスペクト間記述

4.3　PDU の協調場による分割と考察

オブジェクト指向による PDU の実現と協調場により分割した PDU を比較し，プロトコル変更による PDU クラスの変更と，PDU のモデル化をおこなうさいの労力について考察をおこなう．

オブジェクト指向による PDU の実現では，PDU クラスに複数の役割に関する処理が存在していた．オブジェクト指向による PDU の実現を**図 9** に示す．PDU クラスに複数の役割に関する処理が存在しているので，PDU クラスのソースコードの可読性が低く，変更のさいに労力がかかる．また，PDU クラスの構造が複雑になるので，モデル化にも労力がかかる．

PDU の協調場による分割の実現では，PDU の役割毎に PDU クラスのサブクラスを作成する．役割毎にサブクラス化をおこなった PDU を**図 10** に示す．役割毎に処理を分割したことにより，ソースコードの可読性が高くなり，コマンドの追加や変更などによる処理の変更が容易になった．また，複雑な構造の原因となっていた複数の役割を，クラス単位で分割することによって，モデル化をおこなうさいの労力が省力化されたと言える．

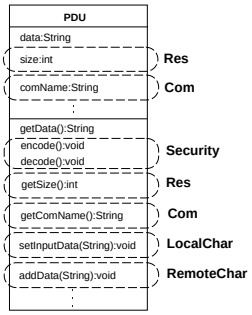

図 9　オブジェクト指向による PDU の実現

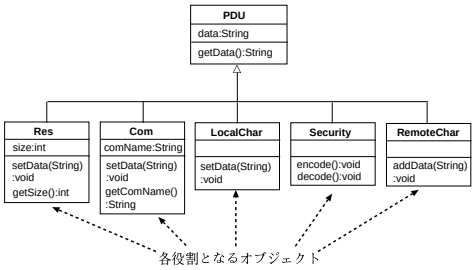

図 10　PDU の協調場による分割の実現

5　おわりに

本研究では，オブジェクト指向により携帯電話の通信ソフトウェアに共通な構造を設計し，各通信ソフトウェアに適用した．通信ソフトウェアの設計をもとに横断的コンサーンを抽出した．抽出した横断的コンサーンをアスペクト指向で分離し，通信プロトコルの変更に対する通信ソフトウェアの柔軟性と再利用性について考察をおこなった．

今後の課題は，今回取り上げた通話システム，web 閲覧システム，メイルシステム以外の携帯電話の通信ソフトウェアへの適用を考える．また，カーナビゲーションやデジタルテレビなど，他の組込み機器の制御ソフトウェアにおける通信ソフトウェアへの適用を考える．

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，小久保佳将さん，八木晴信さんに深く感謝いたします．

参考文献

[1] AspectJ http://eclipse.org/aspectj/

[2] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides： Design Patterns Elements of Reusable Object-Oriented Software, Addiso-Wesley，p.395 (1995).

[3] Tetsuo Tamai： Evolvable Programming based on Collaboration-Field and Role Model, IWPSE (2002).

[4] COMMUNICATIONS of the ACM, Vol.44, No.10, p.168 (2001).