新人研修報告書

平成 20 年 6 月 30 日
筒井　雅敏

1.目的

Java 言語及び、Struts の学習を通して必要知識と基礎技術を身に付け、開発現場で通用するレベルのプログラミング能力を習得する事を目的とする。

2.方法

Robocode を利用し、Java 言語で動作する戦車のプログラムを作成することで、Java 言語の基本的な構文と、オブジェクト指向基礎知識の習得を目指す。また、Struts を利用し、学校給食会システムのプログラムを作成することで、web アプリケーション開発における Struts の基礎知識と技術習得を目指す。

3.結果

Robocode を利用した戦車のプログラムを 5 本作成することで、Java の基本的な構文やオブジェクト指向の概要、デザインパターンについての基礎を理解した。また、Struts を利用した学校給食会システムのプログラムを 6 本作成することで、web アプリケーション開発におけるフレームワーク（Struts）及び、MVC モデルの利点と、Java からの SQL 操作や Excel 出力操作の基礎知識を習得することができた。

4.結論

研修を通して Java 言語の基本的な構文や、オブジェクト指向に対する考え方、デザインパターンの意義、Struts を利用した web アプリケーションの開発方法、フレームワーク（Struts）と MVC モデリングの利点、Java からのデータベースや Excel 出力の操作方法などについて知識と基本的な技術を習得することができた。今後も常に知識習得と技術向上を心掛け、1 日でも早く開発現場で恥じることのないようなプログラミング能力を身に付けるよう精進する。

5.コメント

3 ヶ月前にプログラミングというものの第一歩からスタートいたしましたので、研修期間に学んだこと全てを完璧に習得し実践できるレベルまでには到底及んでおりませんが、基礎となる知識や技術は習得できましたので、この研修で学んだことを礎に今後も知識習得や技術向上の為の努力を怠らず、精進していこうと思っております。

目次

頁

[1] 目　的

Java 言語は OS に依存せず、どのようなプラットフォームでも動作する汎用性の高いプログラム言語として、幅広い分野で必要とされている。

このような Java 言語及び、JSP ＆ Servlet を学習し、必要知識と基礎技術を習得することで、プログラマーとしての間口を広げるとともに、実際の開発現場で通用するレベルのプログラミング能力を身に付ける事を目的とする。

[2] 対象システム

Java 教育で使用する PC 及び、開発環境をいかに示す。

OS	：	Microsoft Windows XP Professional Version2002 ServicePack2
メモリ	：	1.00GB RAM
JDK	：	Java 2 Platform Standard Edition Development Kit 5.0
IDE	：	Eclipce_SDK_ Version3.3.1.1
Robocode	：	Robocode_Version1.0.6
MySQL	：	MySQLServer_5.0
Tomcat	：	Apach Tomcat_6.0
Struts	：	Struts_1.3.8

[3] 方　法

3-1　　Java 言語基礎習得

Robocode を用いて Java 言語の基本的な構文及び、オブジェクト指向基礎の習得を目標とし、Java プログラムで以下の動作をする戦車を作成する。() 内に習得するスキルを示す。

- 三角形と四角形の動きを交互に行うロボット　(継承及び、オーバーライド)
- 様々な方向にスピンしながら移動するロボット (継承及び、オーバーライド)
- フィールド中央に移動して 360 度敵を索敵し、攻撃するロボット (継承及び、オーバーライド)
- 被弾する毎に三角形と四角形の動きを交互に切替えるロボット (インターフェース・抽象クラス)
- オリジナルロボット (デザインパターン)

3-1-1　　継承及び、オーバーライドについて

参考書「やさしい Java 第 3 版」(第 11 章 1 項「継承」、第 11 章 3 項「オーバーライド」)、その他関連 web サイトを参考に、継承及びオーバーライドについての概要と使い方について学習する。

以下の動作をする戦車を作成し Robocode 上で動きを確認しながら、プログラムの流れと継承及び、オーバーライドについての知識を習得する。

- 三角形と四角形の動きを交互に行うロボット
- 様々な方向にスピンしながら移動するロボット
- フィールド中央に移動して 360 度敵を索敵し、攻撃するロボット

3-1-2　　インターフェースについて

参考書「やさしい Java 第 3 版」(第 12 章 2 項「インターフェース」)、その他関連 web サイトを用いて、インターフェースの定義や用途、多態性について学習し、サンプルプログラムの作成や章末問題を通して基礎理解を固める。

よりオブジェクト指向的なプログラムへの変更の 1 つとして、すでに作成済みの三角形と四角形の動きを交互に行うロボットのプログラムを、インターフェースを用いて拡張し、被弾する度に動作を交互に切替えるロボットへ修正することで、オブジェクト指向プログラミングにおけるインターフェースの役割をより深く理解し、習得する。

3-1-3　抽象クラスについて

参考書「やさしい Java 第 3 版」(第 12 章 1 項「抽象クラス」)、その他関連 web サイトを参考にし、主に以下の項目について、調査しサンプルプログラムを作成、動作させることで知識習得を行う。

- 抽象クラスについて
- インターフェースと抽象クラスの違いについて

よりオブジェクト指向的なプログラムへの変更のもう 1 つの方法として、デザインパターンを用いて、すでに作成済みの被弾する度に三角形と四角形の動作を交互に切替えるロボットを Factory パターンで作成し、オブジェクト指向とデザインパターンの密接な関係を学ぶ。

＊ デザインパターンの学習内容については 3-1-4 章を参照。

3-1-4　デザインパターンについて

参考書「サルでもわかる逆引きデザインパターン 第 2 章 ロジック編」、Web サイト「TECHSCORE モデリング デザインパターン」を主として参考にし、以下 2 つのオブジェクト指向の実践的なプログラミングパターンを学習する。

- Factory パターン
- Strategy パターン

Java 言語基礎習得の総復習としてオリジナルの動作、攻撃をするロボットを作成。Web サイト「@IT UML/モデリング」を主として参考にし、UML（Unified Modeling Language：統一モデリング言語）について知識を深めた上で、オブジェクト指向基礎理解の確認と共に、各項目に対する自身の理解度を確認する。

3-2　Struts 基礎習得

Struts を用いて、web アプリケーション開発におけるフレームワーク（Struts）の知識と基礎技術習得を目標とし、以下 6 本の「学校給食会システム」プログラムを作成する。

1. 学校マスタ保守
2. 学校マスタリスト
3. 製品区分マスタリスト
4. 納品書発行
5. 材料受払台帳
6. 原料入出庫一覧表

3-2-1　Struts について

参考書「Apache Struts 入門」、その他関連 web サイトを参考に、フレームワーク（Struts）を用いた web アプリケーション開発について、基礎と概要を学習し、以下の項目について知識習得を行う。

- フレームワークについて
- MVC（Model View Control）について

習得した知識を用いて「学校給食会システム」各プログラムを作成することで、web アプリケーション開発におけるフレームワーク（Struts）の具体的な使用方法を習得する。

3-2-2　JSP について

参考書「Java の道 Servlet・JSP」、その他関連 web サイトを参考に、JSP について以下の項目を調査、web アプリケーション開発における JSP の基礎と概要を学習し、知識習得を行う。

- JSP について
- HTML（Hyper Text Markup Language）について

習得した知識を用いて「学校給食会システム」各プログラムの JSP ファイルを作成することで、HTML 言語と JSP ファイルへの具体的な記述方法を習得する。

3-2-3　SQL について

Web サイト「@IT SQL 実践講座 SQL の基礎」を主として参考にし、SQL 及び、JDBC について調査し、SQL 文の基礎知識と Java から MySQL への基本的な操作方法を習得する。

- データベースについて
- JDBC（Java Database Connectivity）について

習得した知識を用いて学校給食会システムの各プログラムにおいて、必要なデータを扱うロジックを記述し動作確認することで、Java からの SQL 操作について基礎技術を習得する。

3-2-4　POI について

Web サイト「IT アーキテクト POI で Excel を操る」、その他関連 web サイトを参考に、POI ついて調査し、Java から Excel を操作する知識の習得を行う。

- POI（Poor Obfuscation Implementation）について

習得した知識を用いて「学校給食会システム」の帳票出力プログラムで POI を用いた Excel 出力ロジックを記述し動作確認することで、Java ファイルから Excel ファイルへ出力する基礎技術を習得する。

[4] 結　果

4-1　　Java 言語基礎習得

4-1-1 ～ 4-1-4 章の各内容について参考書及び、web サイトで調査を行い、それぞれの定義と使い方、プログラムの流れを学び、理解することで知識として習得することができた。

また、Robocode を用いて、それぞれ違う動作をする戦車のプログラムを 5 本、Java 言語で作成することにより、Java の基本的な構文やオブジェクト指向の基礎について理解が深まり、習得した知識を技術として身につけることができた。

4-1-1　　継承及び、オーバーライドについて

・「継承（inheritance）」

元となるスーパークラスを土台とし、拡張（extends）したサブクラスを実装することで、スーパークラスの機能をサブクラスでも使用できるようにする方法。継承の使用方法及び、注意点について以下に記す。

- 継承するサブクラスの宣言は以下の通りに行う。
  「class　サブクラス名　extends スーパークラス名」
- final 修飾子が付加されたスーパークラスは継承できない。
- スーパークラスで private 修飾子が付加された変数、メソッドは継承したサブクラスから参照できない。
- スーパークラスで宣言されたコンストラクタは、サブクラスに継承されない

・「オーバーライド」

スーパークラスで定義されているメソッドを、継承したサブクラスにて同じ名前かつ、同じ引数リスト（数、型）で実装すること。これにより、継承したサブクラスでそれぞれ異なる独自の処理を行うことができる。オーバーライドを定義する際の主な規定について以下に記す。

- 戻り値の型、メソッド名、引数の型、引数の数が同じであること。
- final 修飾子が付加されていないこと。
- 付加されてある修飾子のアクセス制限が、スーパークラスで定義されているメソッドよりも低いこと

上記 2 項目について理解した上で、以下に示す 3 本のロボットを作成した。作成したプログラムの概要を図 4-1 に示す。

1. **三角形と四角形の動きを交互に行うロボット**
2. **様々な方向にスピンしながら移動するロボット**
3. **フィールド中央に移動して 360 度敵を索敵し、攻撃するロボット**

スーパークラスである「Robot クラス[*1]」及び、「AdvancedRobot クラス[*2]」を継承したクラス（サブクラス[*3]）を作成し、独自の動作を行うメソッド[*4] を定義。3 本のロボットにおいて、共通のメソッド（run()[*5]）をオーバーライドした結果、それぞれ異なる 3 つの動作を行う戦車が完成した。

この一連の動作を確認することで、java 言語における継承とオーバーライドについての知識をより深く習得し、スキルとして身に付けることができた。

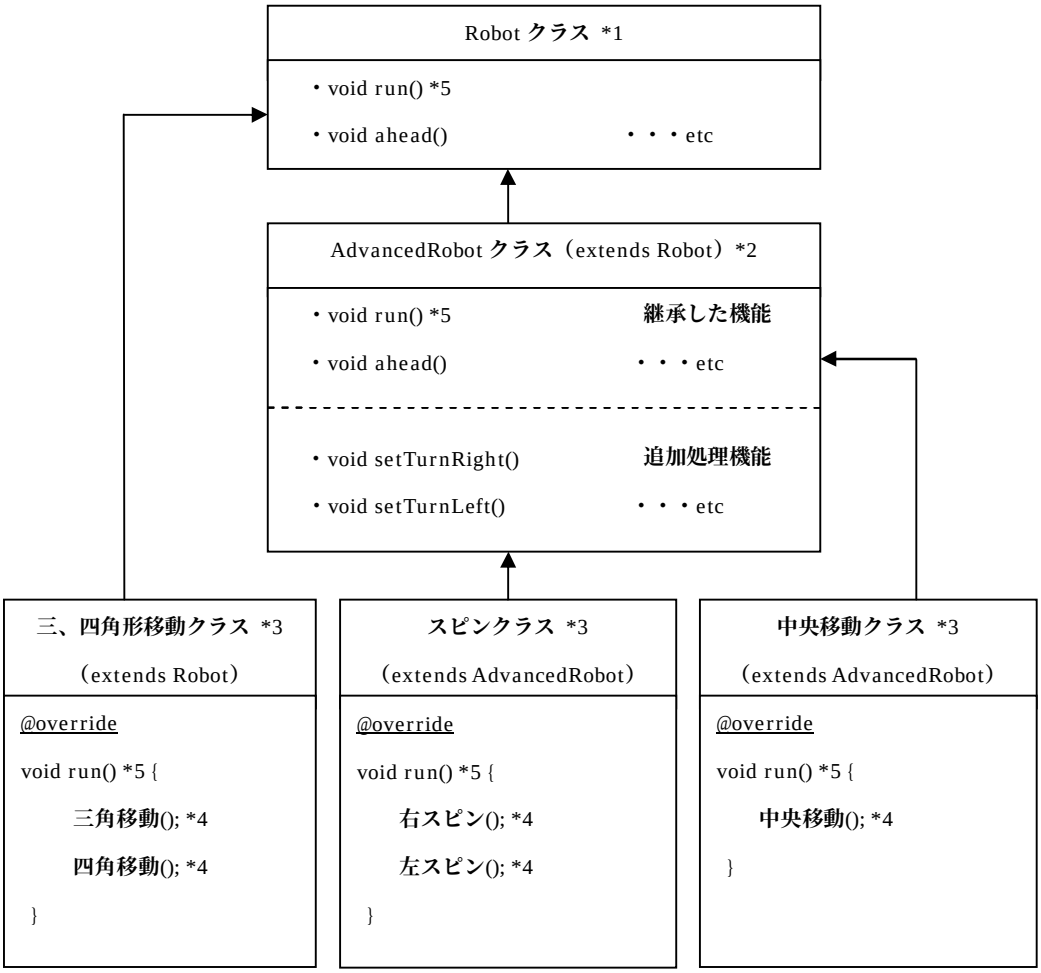

図 4-1　継承及び、オーバーライドについて

4-1-2　インターフェースについて

・「インターフェース」

ある特定の機能（メソッド）の概要のみを記述したものであり、インターフェースを実装する各クラスにおいて機能の詳細（処理内容）を定義する。これにより、プログラムの利用者側は「何をするのか」のみを定義し「どうするのか」といった実際の処理内容を意識せず、プログラムを利用することができる。またコーディング側は、処理を依頼されるクラス（実装クラス）に対して処理内容を明確にでき、機能拡張、縮小、追加、削除といったメンテナンス性も向上する。

インターフェースの主な特徴として以下に示す。

- フィールドには自動的に final public static 修飾子が付加
- メソッドには自動的に abstract public が付加
- メソッドは処理内容を定義しない抽象メソッドのみ
- 複数のインターフェースを多重継承可能
- 実装クラスはインターフェースの全メソッドを必須で実装

上記を理解した上で、4-1-1 章で作成した三角形と四角形の動きを交互に行うロボットに対して、インターフェースを用いて被弾時に動作を切り替えるロボットへ拡張する。作成したプログラムの概要を図 4-2 に示す

抽象的な動作（abstract move()）のみを定義したインターフェース*1 を作成。このインターフェースを実装（implements）させるクラス*2 を 2 つ作成し、それぞれに三角形と四角形の動きをする処理内容を定義したメソッド*3 を実装させる。Robot クラスを継承したクラス*4 の被弾時に呼ばれるメソッド（onHitByBullet()）*5 で動作を切り替える処理を定義し、メインメソッド（run()）*6 で動作処理（move()*7 メソッドの実行）を定義することで、被弾時に動作を切り替える戦車が完成した。

動作を切替える処理として、Robot クラスを継承したクラス*4 において、インターフェースを実装したクラス*2 のオブジェクトをそれぞれ生成し*8、インターフェース型の変数*9 に初期動作として設定。被弾時に呼ばれるメソッド*5 において、現在設定されている動作オブジェクトと異なる動作オブジェクトをインターフェース型の変数に代入する*10 ことで動作を切替える処理として定義した。これにより、run()メソッドの実装は動作を意識することなく行えるようになった。

この一連の動作を確認することで、java 言語におけるインターフェースについての知識をより深く習得し、スキルとして身に付けることができた。

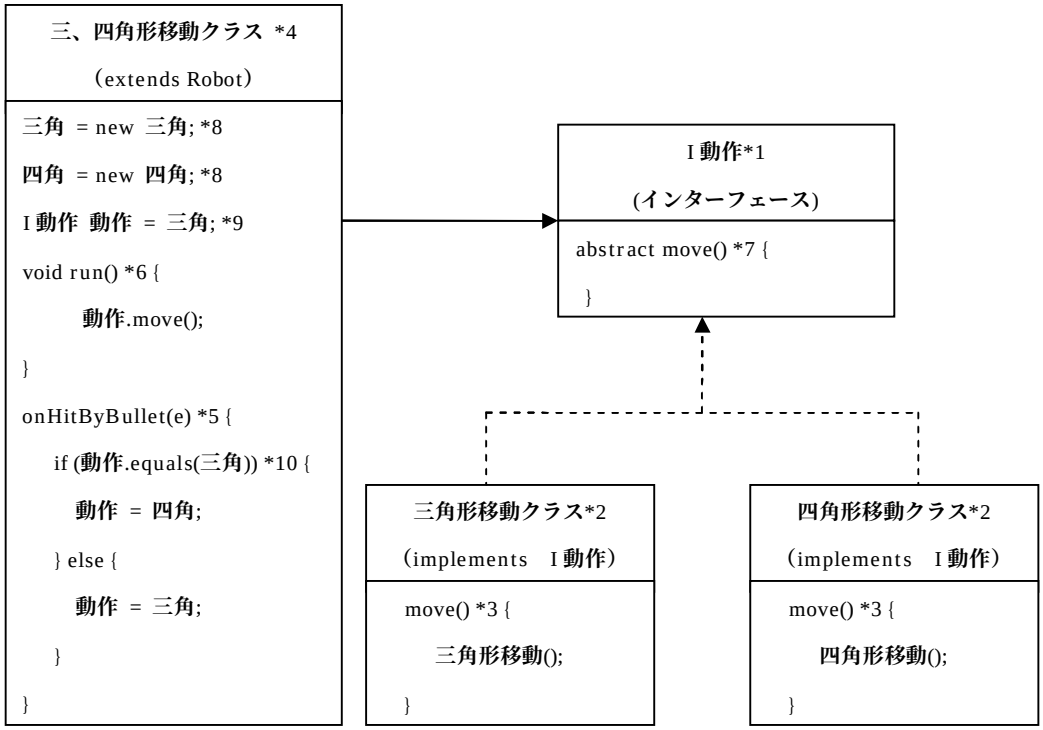

図 4-2　インターフェースについて

4-1-3　　抽象クラスについて

・「抽象クラス」

抽象メソッドを持つクラスのこと。抽象メソッドを持つ、インスタンス生成できないなどインターフェースと類似する点は多いが、主な違いとして以下のことがあげられる。

・抽象クラスは処理内容を記述した具象メソッドも定義可能

・多重継承できない（インターフェースは実装のみ複数可）

抽象クラスの場合、継承する複数のサブクラスに対して、共通して持たせたい機能を具象メソッドとして定義し、それぞれ独自の処理を持たせたい機能は抽象メソッドとして定義することで、継承したサブクラスにある特定の共通機能（具象メソッド）を保障し、かつ独自の処理を持たせることができる。

上記を理解した上で、4-1-2 章で作成した被弾時に動作を切り替えるロボットを抽象クラスと関連するデザインパターンの Factory パターンを用いて作成した。作成したプログラムの概要を図 4-3 に示す。

＊ Factory パターンの学習内容については 4-1-4 章を参照。

オブジェクト生成を行う抽象メソッド（abstract create()）[*1] を定義した抽象クラス[*2] を作成。継承したサブクラス[*3] で抽象メソッドを実装し、三角形と四角形の動作を行うクラスのオブジェクトをそれぞれ生成[*4]。オブジェクトの生成処理[*5] と使用処理[*6] を分離することで、利用者側に処理過程を意識させないことができる。またソースコードのメンテナンス性も向上する。よりオブジェクト指向を重視した Factory パターンを用いることで、java 言語における抽象クラスと Factory パターンの概念を理解、習得することができた。

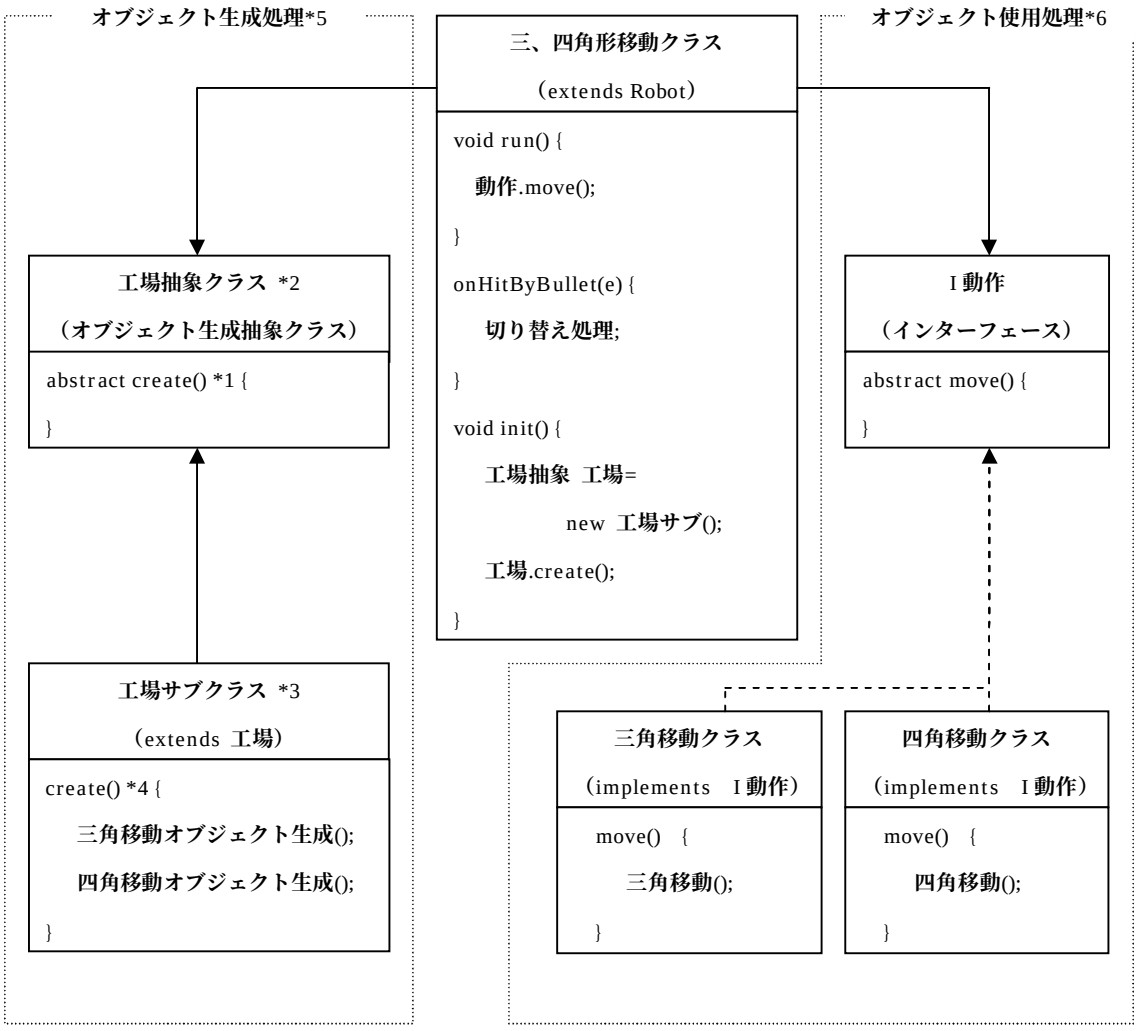

図 4-3　抽象クラスについて

4-1-4　デザインパターンについて

・「Factory パターン」

オブジェクトの生成処理と使用処理を分離させるパターン。利用者側は処理過程を意識せずプログラムを使用でき、開発者側は生成するオブジェクトの追加や削除、処理内容の追加や削除をメインのロジックを大幅修正することなく変更できるため、メンテナンス性が向上する。

・「Strategy パターン」

インターフェースを窓口として、処理の内容毎にクラスを用意し、実行内容に適応する処理を持つクラスに切り替えるパターン。利用者側は、処理過程を意識することなくプログラムを利用でき、開発者側は処理内容の追加や削除、修正などを容易に行えるメリットがある。

・「UML（Unified Modeling Language：統一モデリング言語)」

オブジェクト指向でのプログラム開発における設計図の統一表記法。今回のオリジナルロボットに関して作成したクラス図とは、プログラムの静的構造を表したものであり、属性、操作といったクラスの内部構造と、クラス間の関係を図にしたものである。

上記したデザインパターン及び、UML について概要を理解した上で、Java 言語基礎習得の総復習として、Robocode においてオリジナルロボットを考案し、4-1-1～4-1-3 章で学習した内容を用いて作成した。

基本動作

1. フィールドに対し周回移動を行い続ける。
2. 周回移動を続けながらフィールド内を索敵する。
3. 発見した敵機と自身の距離に応じて攻撃力を切替え、砲撃する。
4. 敵機接触及び、規定回数以上被弾した場合のみ、周回移動から逃避移動へ動作を切替える動作を行う。

上記の戦車を考案、作成した。

以下、図 4-4 に作成したオリジナルロボットの概要を示す。

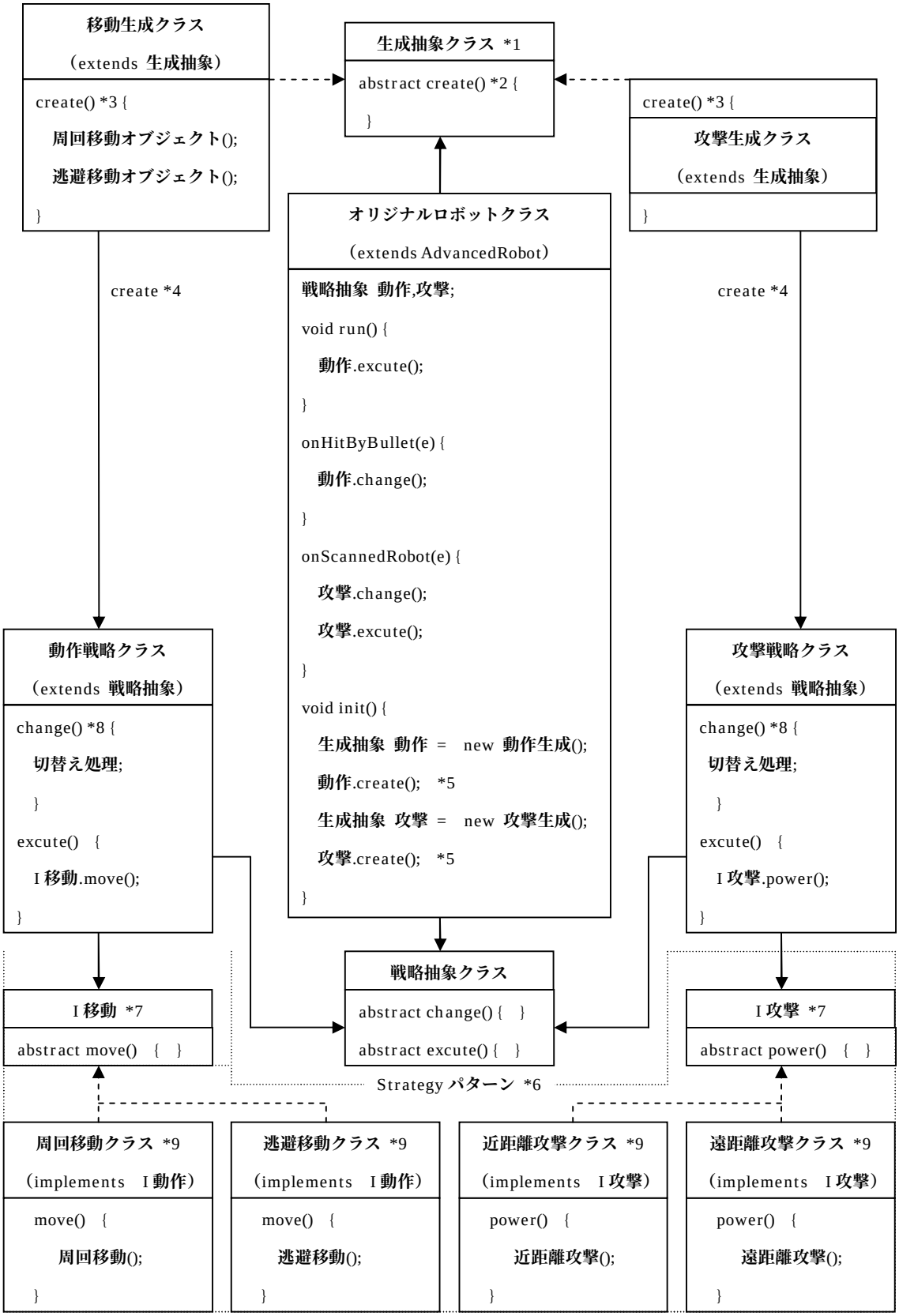

図 4-4 オリジナルロボットについて

図 4-4 では、Factory パターンを用いて、オブジェクト生成抽象クラス[*1]を作成。抽象メソッド（abstract create()）[*2]を定義し、継承した各サブクラスで実装して戦略選択の元となる各動作、攻撃クラスのオブジェクトをそれぞれ生成した[*3]。生成したオブジェクトの戻り値としてそれぞれ動作戦略クラス、攻撃戦略クラスを設定[*4]。これにより、メイン側では抽象メソッド[*5]を呼び出すのみで、動作オブジェクトは動作戦略クラス型へ、攻撃オブジェクトは攻撃戦略クラス型へと内部の処理を意識せずに適応したオブジェクトを生成できるようになった。

オブジェクトの使用処理としては、Strategy パターン[*6]を用いて、動作、攻撃を切り替える処理を行った。動作、攻撃ともにそれぞれインターフェース[*7]を作成し、状況に応じた戦略へと切り替える際[*8]に、適応する処理を行うクラス[*9]を呼出し実行させることで、複雑なアルゴリズムを用いることなく目的の戦略に切り替えることが可能になった。また、プログラム作成の際に、当初他にも動作を考案していたが、無駄な動作であると判断。Strategy パターンを用いて作成していたことにより、メインのロジックを大幅に修正することなく容易に、無駄な動作処理を定義したクラスを削除することができたことで、デザインパターンの利便性を実感することができた。

4-2　　Struts の基礎習得

4-2-1～4-2-4 章の各内容について参考書及び、web サイトで調査を行い、それぞれの概要と使い方、web アプリケーションにおけるプログラムの流れを学び、理解することで知識として習得することができた。

Struts を利用した学校給食会システムの web アプリケーションプログラムを 6 本作成することで、オブジェクト指向基礎習得で学んだ Java の基本的な構文の復習やデザインパターン、Java プログラムにおけるデータベースの操作や Excel への帳票出力など、より実践的な分野への理解が深まり、習得した知識を技術として身につけることができた。

4-2-1　　Struts について

・「フレームワーク」

アプリケーションを開発する際、頻繁に必要とされる機能をまとめて提供する土台として用いられるソフトウェア。メリットとして、開発時に独自に必要とされる部分のみを開発すれば済むので開発効率の向上が見込める。Struts も web アプリ開発におけるフレームワークの 1 つ。

・「MVC(Model View Control)」

アプリケーションの役割分担を M(Model)、V(View)、C(Control)の 3 つに分割するシステムモデル。

M(Model)	データ構造やビジネスロジックを実装するシステムの本体的役割を担う。
V(View)	入力、結果などの画面表示を担う。JSP で実装。
C(Control)	View と Model の制御を担う。

MVC モデルのメリットとして、機能ごとの分離が明確になり、それぞれの独立性が保たれることで、開発側の分業が容易になり、開発効率の向上に繋がる。また、それぞれの依存性が最小限に抑えられることで、他部分を変更した場合などによる影響を受けにくい。

Struts における MVC モデルの概要を図 4-5 に示す。

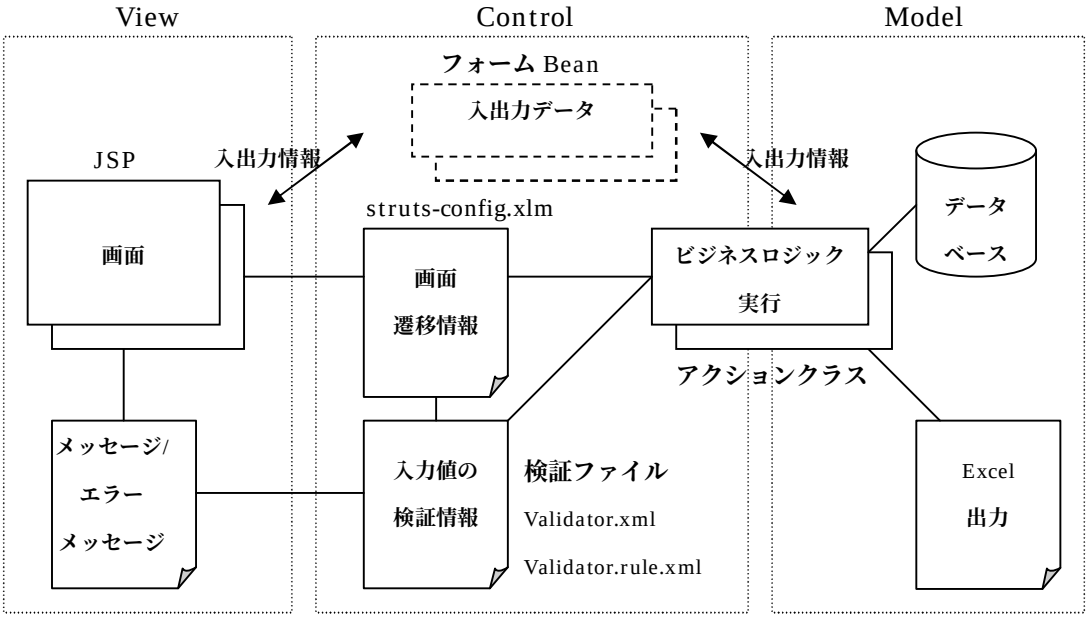

図 4-5 Struts における MVC モデル

・画面 （JSP）

入力させたものに対して、ビジネスロジックの実行結果を表示する必要があるため、Struts で用意されてあるタグライブラリを用いて JSP ファイルを作成する。学校給食会システムにおいては、入力するテキストボックス（html タグ）や、項目選択のためのセレクトボックス、入力された項目へのエラーメッセージの表示（message タグ）など全ての項目に対してタグライブラリを利用し、理解を深めた。

・画面遷移情報

各画面のフォームが submit されたときに実行するビジネスロジックや、その結果など、次にどのページへ遷移するのかといった遷移情報を struts-config.xml に集約する。学校給食会システムにおいては、あらかじめ遷移先が指定してあったが、新しく追加した初期表示のための初期アクションクラスなどへの遷移に関して、struts-config.xml を修正し遷移先を変更することで理解を深めることができた。

・ビジネスロジックの実行（アクションクラス）

定義したビジネスロジックの処理をアクションクラスで実行する。学校給食会システムにおいては、主にデータベースへの登録、検索、削除などの操作や、Excel ファイルへの出力を操作することで、Java の情報保持についてリストやマップの定義と使い方、また Java からの SQL 操作、Excel 出力方法を学ぶことができた。

・入出力情報（フォーム Bean）

画面での入力情報やビジネスロジックの実行結果などはフォーム Bean を介してやりとりされる。学校給食会システムにおいては、画面での入力項目をフォーム Bean を介してアクションクラスへ送り、送られた情報を元にデータベースから必要な情報を取得、取得した情報は画面表示のためフォーム Bean へ送ったり、リスト作成のための RecordBean へ送ったりすることで、情報の遷移とフォーム Bean の役割について理解することができた。

・入力値の検証、エラーメッセージ

Struts では利用者が画面で入力した値を検証する Validator を利用することで容易に入力値のチェックを行うことができる。学校給食会システムにおいては、各入力項目に対して Validator の役割を果たす入力値検証メソッドを定義したクラスを作成し、予め struts-config.xml で遷移先を指定。そのクラスを利用することで、入力値に誤りがあった場合、プロパティで指定したエラーメッセージを Validator 使用時と同じように表示することができた。

4-2-2　JSP について

・「JSP」

HTML ファイルの中に Java プログラムを埋め込むことで、Java 言語を利用した動的な web ページを作成できる技術。メリットとして、利用者側から要求された処理を Java プログラムで行い、結果を HTML 形式で返すため、プログラムを実行するための特殊な機能を web ブラウザに組み込むことなく web アプリケーションを構築することができる。以下に JSP の構成を示す。

・ディレクティブ

JSP 実行時の制御。サーブレットコンテナへの指令文

・スクリプティング要素

「宣言」「式」「スクリプトレット」で構成。Java 言語記述部

・アクション

Bean の処理や転送などの命令文を記述

・暗黙オブジェクト

使用頻度の高いメソッドなどを定義したクラスで、変数宣言せずに使用可

・「HTML（Hyper Text Markup Language）」

テキストを web ブラウザに表示させるための言語であり、「タグ」を用いて様々な設定の指示（要素）を記す。

学校給食会システムにおいて 6 本の JSP ファイルを作成した。各プログラムにおいて Struts で用意されてあるタグライブラリを利用し、仕様書のレイアウト通りに画面表示されているか、action の遷移先や、入力された項目に対応する property の遷移先は正しく設定してあるかなどを確認、意識しながら作成することで、MVC モデルにおける View の役割と、Struts を用いた web アプリケーションプログラムの流れを理解することができた。

4-2-3　SQL について

・「データベース」

多数の情報を整理、分類して管理し、SQL 言語を用いて必要な情報を検索、登録、削除することができるシステム。

・「JDBC（Java Database Connectivity）」

Java プログラムからデータベースへアクセスするための API であり、Java プログラムから SQL 言語を発行することでデータベースの操作を行うことができる。学校給食会システムにおいて、MySQLConnector/J を使用することでデータベースへのアクセスが可能になった。

学校給食会システムの web アプリケーション開発過程において、MySQL で管理されている情報に対し検索、登録、削除などの処理を行う以下のビジネスロジックを作成することで、Java プログラムから MySQL を操作する基本的な方法を習得した。

・該当データの削除

削除時の SQL 構文：delete from テーブル名 where 検索条件

データベースに該当するデータが存在すれば DELATE 文を用いて削除、該当するデータが存在しなければプロパティファイルで定義してあるエラーメッセージを画面に出力するビジネスロジックを定義した。以下に「学校給食会システム」の学校マスタ保守プログラムで用いた SQL 文を例として示す。

```
delete from m_school where scode = ?
```

＊m_school = 学校マスタテーブル　scode = 学校コード

?=画面で入力された値

・該当データの登録及び、更新

登録時の SQL 構文： insert into テーブル名 (項目名) values (データ)

更新時の SQL 構文： update テーブル名 set 項目名=データ where 検索条件

データベースに該当するデータが存在すれば UPDATE 文で、該当データが存在しなければ INSERT 文をそれぞれ用いて MySQL へ登録するビジネスロジックを定義した。以下に「学校給食会システム」の学校マスタ保守プログラムで用いた SQL 文を例として示す。

insert into m_school (scode,sname...) values (?,?,...)
＊m_school = 学校マスタテーブル　scode = 学校コード sname = 学校名称　?=画面で入力された値

update m_school set sname=? Where scode=?
＊m_school = 学校マスタテーブル　scode = 学校コード sname = 学校名称　?=画面で入力された値

・該当データの検索

検索時の SQL 構文： select 項目名 from テーブル名 where 検索条件

データベースに該当するデータが存在すれば検索した各データを Bean クラスにセットし検索結果として web 画面に表示させ、該当データが存在しなければプロパティファイルで定義してあるエラーメッセージを画面に出力するビジネスロジックを定義した。以下に「学校給食会システム」の学校マスタ保守プログラムで用いた SQL 文を例として示す。

select * from m_school where scode = ?
＊m_school = 学校マスタテーブル　scode = 学校コード *=全ての項目　?=画面で入力された値

・複雑な条件のデータ検索

サブクエリ構文：

select 項目名 from テーブル名 where exist

(select 項目名 from テーブル名 where 条件)

利用者の求める情報が MySQL 内で複数のテーブルと関連し、かつ複雑な検索条件であった場合に対応する措置として、必要なテーブル同士の key となる項目を WHERE 句で結ぶ結合方法と、必要な情報のみを予め抽出した SQL 文をテーブルとして利用するサブクエリを学んだ。学校給食会システムにおいては、当初外部結合及び、サブクエリを利用し複数テーブルにまたが

る情報を一括で抽出する SQL 文を記述し、処理を行うビジネスロジックを定義することで、MySQL における結合とサブクエリの使用方法を身に付けることができた。また、検索条件や関連するテーブルがあまりにも複雑になったプログラムにおいては、SQL 文が複雑になり過ぎない程度に小分けして情報を抽出する処理に変更することで、より正確な情報抽出に近づけるよう心がけた。

4-2-4　POI について

・「POI（Poor Obfuscation Implementation）」

Java プログラムから複合ドキュメント形式（Excel、Word など）を扱うための API であり、すべて Java 言語で記述されているため、Java プログラムへの組み込みも容易に行えるメリットがある。

学校給食会システムにおいて、5 本の帳票出力プログラムを作成。以下の手順で各プログラムの帳票出力を行うことで、Java プログラムにおける POI を用いての Excel 出力方法を習得した。

1. 原本となる Excel ファイルの作成。各出力内容に応じた項目及び、仕様書のレイアウトに沿った 5 つの Excel ファイルを作成。
2. 5 本の帳票出力プログラムのアクションクラスにて、事前に作成しておいた PropertyFileAccess クラス*1 を介し、作成した Excel ファイルへのパスを指定。
3. 項目 2.で指定したパスのファイルオブジェクト(POIFSFileSystem)を生成。
4. 項目 3.で生成したファイルのワークブックオブジェクト(HSSFWorkbook)を生成。
5. 項目 4.で生成したワークブックから原本となるワークシート（HSSFSheet）を取得。
6. 取得したワークシートに出力するセルを設定(CellSeting クラス*2 を利用)

*1. PropertyFileAccess　ファイル読込の為、プロパティファイルに設定した Excel ファイルのプロパティを取得するクラス。

*2. CellSeting　セルの結合や設定、罫線の出力やシートのコピーなど、POI を用いた Excel 出力の処理を多数定義し、まとめたクラス。

[5] 結　論

Robocode プログラムの作成を通して、Java 言語における基本的な構文や、オブジェクト指向に対する考え方、デザインパターンの意義などについて基礎知識を習得することができた。また、学校給食会システムのプログラム作成を通して、web アプリケーション開発における MVC モデルを利用した開発の流れとフレームワークの利用、Java プログラムからのデータベースや Excel 出力の基本となる操作方法についても理解することができた。しかし、今回のプログラム作成に関しては、他の人が事前に作成し、用意してくれていたクラスやメソッドを利用したり参考にしたりして作り上げたプログラムがほとんどなので、自分一人の知識と技術で 1 から作り上げたものではない。実際の開発に携わるようになれば、今回のようなことであれば確実にチームの足を引っ張ってしまうことになり兼ねないので、今後は今まで以上に知識習得と技術向上に精進し、1 日でも早く開発現場で恥じることのないようなプログラミング能力を身に付ける。

[6] 参考文献

6.1 参考書

・やさしい Java 第 3 版

・Java 言語プログラミングレッスン（上）

・Java 言語プログラミングレッスン（下）

・Java 言語で学ぶデザインパターン入門

・Apache Struts アプリケーション開発入門

6.2 参考 Web サイト

・Java で Hello World

[http://www.hellohiro.com/basic/]

・浅煎り珈琲 Java アプリケーション入門

[http://www.nextindex.net/java/index.html#abstract]

・Java の道_クラス

[http://www.javaroad.jp/index.htm]

・ロボコードについて

[http://ara.moo.jp/robocode/index.htm]

・ロボコード API ドキュメント

[http://www.solar-system.tuis.ac.jp/Java/robocode_api/]

・2006 年度 デザイン情報セミナーⅠ（Robocode）

[http://www.wakayama-u.ac.jp/~fukuyasu/dis1-2006/robocode/]

・サルでもわかる 逆引きデザインパターン

[http://www.nulab.co.jp/designPatterns/designPatterns1/designPatterns1-1.html]

・TECHSCORE

[http://www.techscore.com/]

・Java ではじめる UML

[http://www.ogis-ri.co.jp/otc/hiroba/technical/JavaWorld_UML/chap2/index.html]

・ごく簡単な HTML の説明

[http://www.kanzaki.com/docs/htminfo.html]

・@IT 第 12 回 Struts 開発を IDE で効率化

[http://www.atmarkit.co.jp/fjava/rensai2/jakarta12/jakarta12_01.html]

・javadrive サーブレット/JSP 入門

[http://www.javadrive.jp/servlet/index.html]