ディプロマシーfor Moodle 最終ドキュメント

明石敬，森岡亮一，鬼頭孝行

2005 年 4 月 5 日

ドキュメントの構成

- 第一部
  - システム分析報告書

- 第二部
  - Moodle 環境構築方法
  - Moodle フレームワークの説明
  - ディプロマシーアーキテクチャの説明
  - ディプロマシーforMoodle マニュアル
- 第三部
  - 明石個人レポート
  - 森岡個人レポート
  - 鬼頭個人レポート

はじめに

1. プロジェクトの発足

プロジェクト発足時は，LMS を使ったシステム開発を目的としていた．システム開発を行う前に，LMS に触れた事のない森岡，鬼頭の両者に LMS に慣れてもらおうと LMS の上に乗せる簡単なシステムを考えてもらった．すると，森岡がディプロマシーというゲームを LMS 上で作って見たい，と企画を考えてきた．森岡が探したディプロマシーというゲームがおもしろそうなので，プロジェクトメンバー全員でディプロマシーを開発することになった．

2. 背景

ディプロマシーとは、ボードを使って遊ぶゲームである。ボードを使った場合，以下に述べる課題が存在する．

・ ゲームの過程を振り返ることが難しい
・ 秘密の交渉を行う事が難しい

以下にそれぞれの詳細を述べる．

2.1. ゲームの過程を振り返ることが難しい

ボードを使ったディプロマシーでは，ゲームの戦況の移り変わりを，毎回紙に書き記さない限りゲームの過程を振り返る事が出来ない．ゲームの過程を振り返り分析することで，どのような交渉が重要であり，また，必要とされているのかを理解することができる．

2.2. 秘密の交渉を行う事が難しい

ディプロマシーには二種類の交渉がある．他のプレイヤーに知られてもよい公の交渉と他のプレイヤーに知られては困る秘密の交渉がある．

ボードを使ったディプロマシーの場合，公の交渉に問題はないのだが，秘密の交渉を行う為に，他のプレイヤーがいない場所へ行き交渉を行うか，筆談を行うなどといった工夫をしなければならない．しかし，いくら工夫をしても他のプレイヤーに怪しい動作を見られてしまえば，交渉を察知されてしまう可能性がある．結果，逆に不利な立場になってしまう，という場合もある．

3. ディプロマシーfor Moodle とは

背景で述べたような課題を解決する為に，我々はディプロマシーfor Moodle の開発を行った．ディプロマシーを Moodle 上で実現する事で，背景で述べられた「ゲームの過程を振り返る事が難しい」といった課題は，ゲームの過程を記録する仕組みを作る．また，「秘密の交渉を行うことが難しい」といった課題は，Moodle に標準で付属しているチャット機能を利用する事で簡単に解決できる．このようなシステムを導入することで，より楽しくより効果的にディプロマシーを行う事ができるはずである．

4. ディプロマシーfor Moodle の開発

我々はディプロマシーfor Moodle の開発をこの学期を通して行った．残念ながら全ての実装を終える事が出来ず，完全な形でディプロマシーを行う事が出来なかった．

しかし，ディプロマシーを開発する上で必要なフレームワークの構築や必要なドキュメントの整備などを完了させることができたので，これらの成果を足がかりにすれば，ディプロマシーfor Moodle を完全に実装する事ができるだろう．

5. 最終ドキュメントについて

本ドキュメントは二部構成になっている．第三部では，我々のプロジェクトを通しての成長を見る事が出来，第二部では，ディプロマシーfor Moodle を開発する上での必要な情報をドキュメントにまとめた．第三部はプロジェクトメンバーの個人レポートとなっている。

本ドキュメントは, 我々のプロジェクトを評価することができ, 誰でもディプロマシーfor Moodle の開発に参加でき，また，新たに Moodle のモジュール開発が取り組めるようになることを目指す．

第一部

第一部の構成

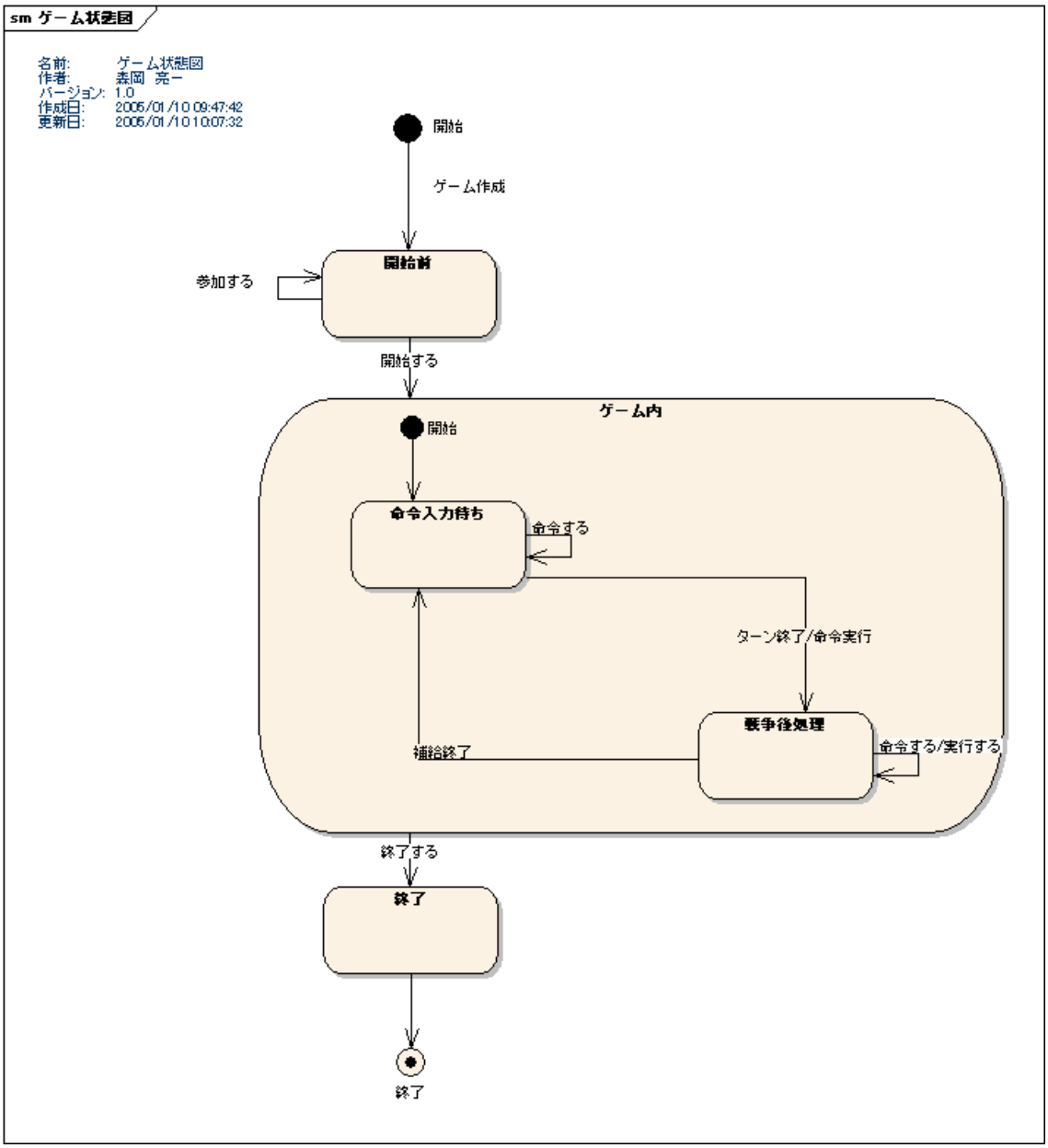
第一部のドキュメントの構成は下記の通りである.

- システム分析報告書
  - ディプロマシーfor Moodle のシステム分析の報告書である．研究会で発表をしたシステム分析報告書とそれを清書した分析報告書が示されている.

システム分析書

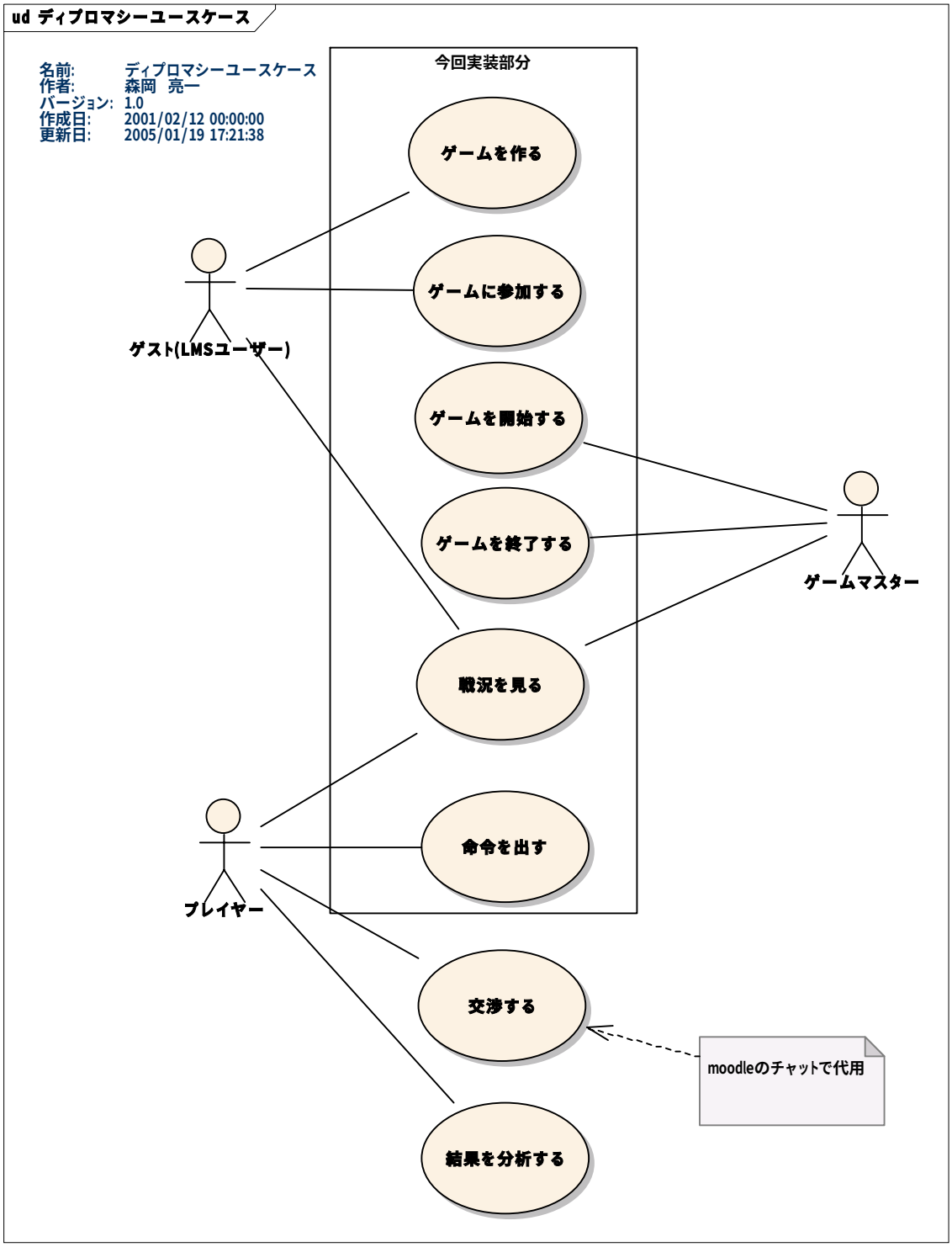
1 ゲーム状態図

　ディプロマシーというゲームが持つ状態とその遷移を表した図である。ゲームクラスがこの状態を保持する。



2 ユースケース図

今回作成したアプリケーションのユースケースである。システム分析報告書では優先度をつけたが、この図では実装部分と未実装の部分を図示している。



3 ユースケース文章

ユースケース：ゲームを作る
目的：ゲームをするため
事前条件：ＬＭＳにログインしていること
事後条件：ゲームが作成されていること
イベントフロー：
　ゲームを作るメニューを選ぶ。（ゲーム作成画面に切り替わる）
　ゲームの名前を入力する。
　ゲームの交渉時間を入力する。
　国の選択方法（ランダム、選択）を入力する。
　作成ボタンを押す。（ゲームを作成し、そのゲームの設定画面（参加申し込み受付状態）に切り替わる）

ユースケース：ゲームに参加する
目的：ゲームをするため
事前条件：ゲームが参加申し込み受付状態であること
事後条件：プレイヤーがゲームに登録されていること
イベントフロー：
　ゲームを選ぶ（ゲーム設定画面（参加申し込み受付状態）に切り替わる）
　国を選ぶモードであれば、国を選ぶ。
　参加ボタンを押す。（プレイヤーを登録し、ゲーム設定画面を更新する）

ユースケース：ゲームを開始する
目的：ゲームをするため
事前条件：参加申し込み受付状態で、一人以上のプレイヤーが登録されていること
事後条件：ゲーム中状態になること
イベントフロー：
　ゲームを選ぶ。（ゲーム設定画面（参加申し込み受付状態）に切り替わる）
　ゲーム開始ボタンを押す。（ゲーム中状態へ遷移し、戦況画面（命令入力待ち状態）に切り替わる）

ユースケース：ゲームを終了する
目的：ゲームをやめるため
事前条件：ゲーム中状態であること
事後条件：ゲーム終了状態になっていること

イベントフロー：

ゲームを選ぶ。（戦況画面（命令入力待ち状態）に切り替わる）

ゲーム終了ボタンを押す。（確認画面に切り替わる）

終了ボタンを押す。（ゲーム終了状態へ遷移し、戦況画面（ゲーム終了状態）に切り替わる）

ユースケース：命令を出す

目的：軍隊を動かすため

事前条件：命令入力待ち状態であること

事後条件：命令が記録されていること

イベントフロー：

ゲームを選ぶ。（戦況画面（命令入力待ち状態）に切り替わる）

命令を出す軍隊を選ぶ。

命令を選ぶ。（を繰り返す。つまり、命令は一度に複数設定できる）

なお、移動命令の場合は、移動先を選ぶ。

なお、サポート命令の場合は、サポート先を選ぶ。

命令入力ボタンを押す（命令を記録し、戦況画面を更新する）

ユースケース：戦況を見る

目的：戦況を観戦するため、戦略を練るため、交渉の判断をするため、etc...

事前条件：ゲーム中状態であること

事後条件：なし

イベントフロー：

ゲームを選ぶ。（戦況画面（いずれかのゲーム中状態）に切り替わる）

（一つ前の履歴だけは、戦況画面に表示されている）

過去の履歴ボタンを押す。（履歴画面に切り替わる）

ユースケース：交渉する

目的：戦略を練るため

事前条件：命令入力待ち状態であること

事後条件：交渉内容（メッセージのやりとり）が記録される

イベントフロー：

ゲームを選ぶ。（戦況画面（命令入力待ち画面 A）に切り替わる）

交渉相手を選択し、交渉ボタンを押す（交渉画面に切り替わる　図なし）

メッセージを入力し、送信ボタンを押す（交渉相手にメッセージが送られる）

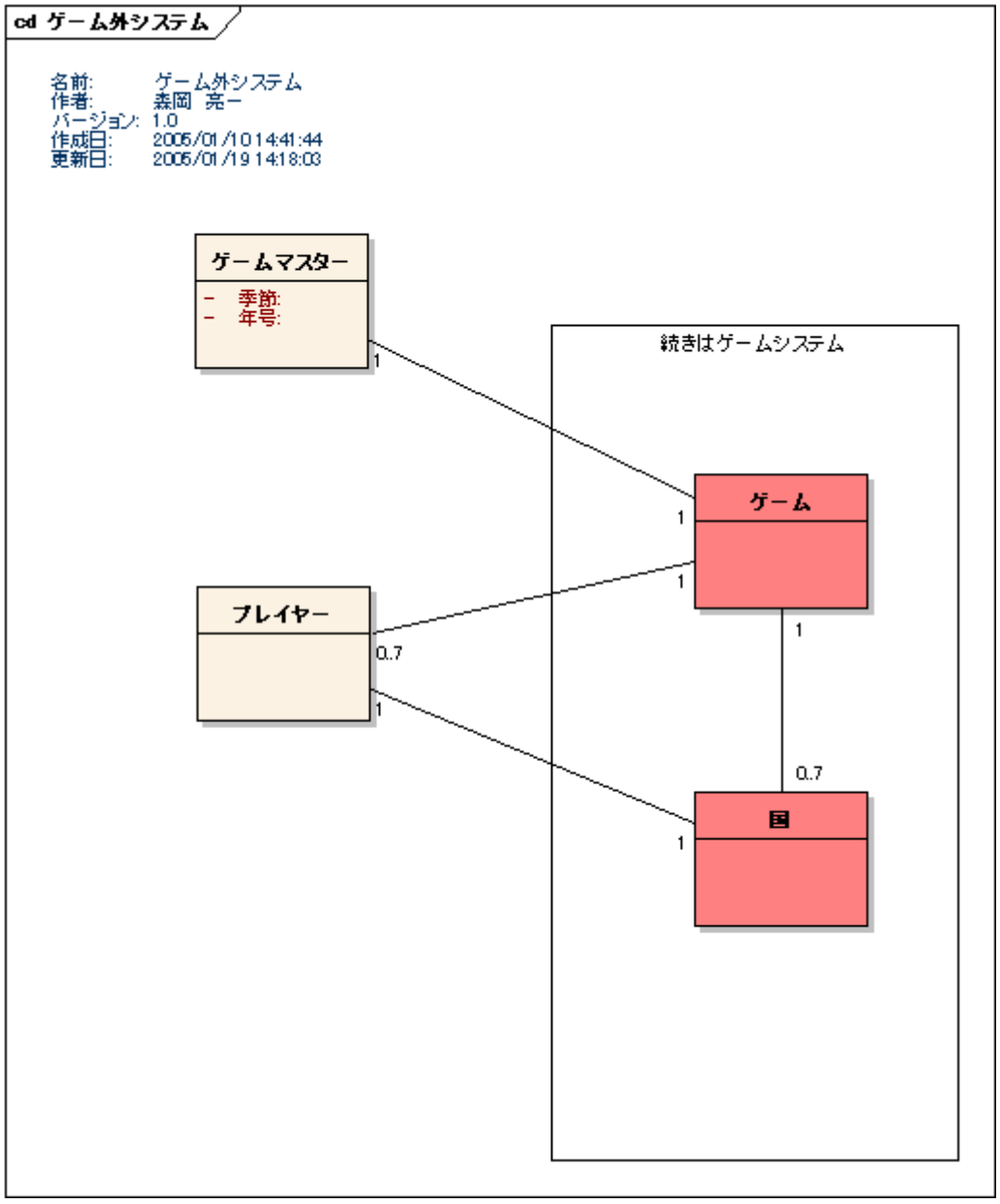
（相手からメッセージが送られてくるとメッセージが表示される）

ユースケース：分析する
目的：何故そういう結果になったのか知りたいから
事前条件：ゲームが終了していること
事後条件：なし
イベントフロー：
　　以下未定

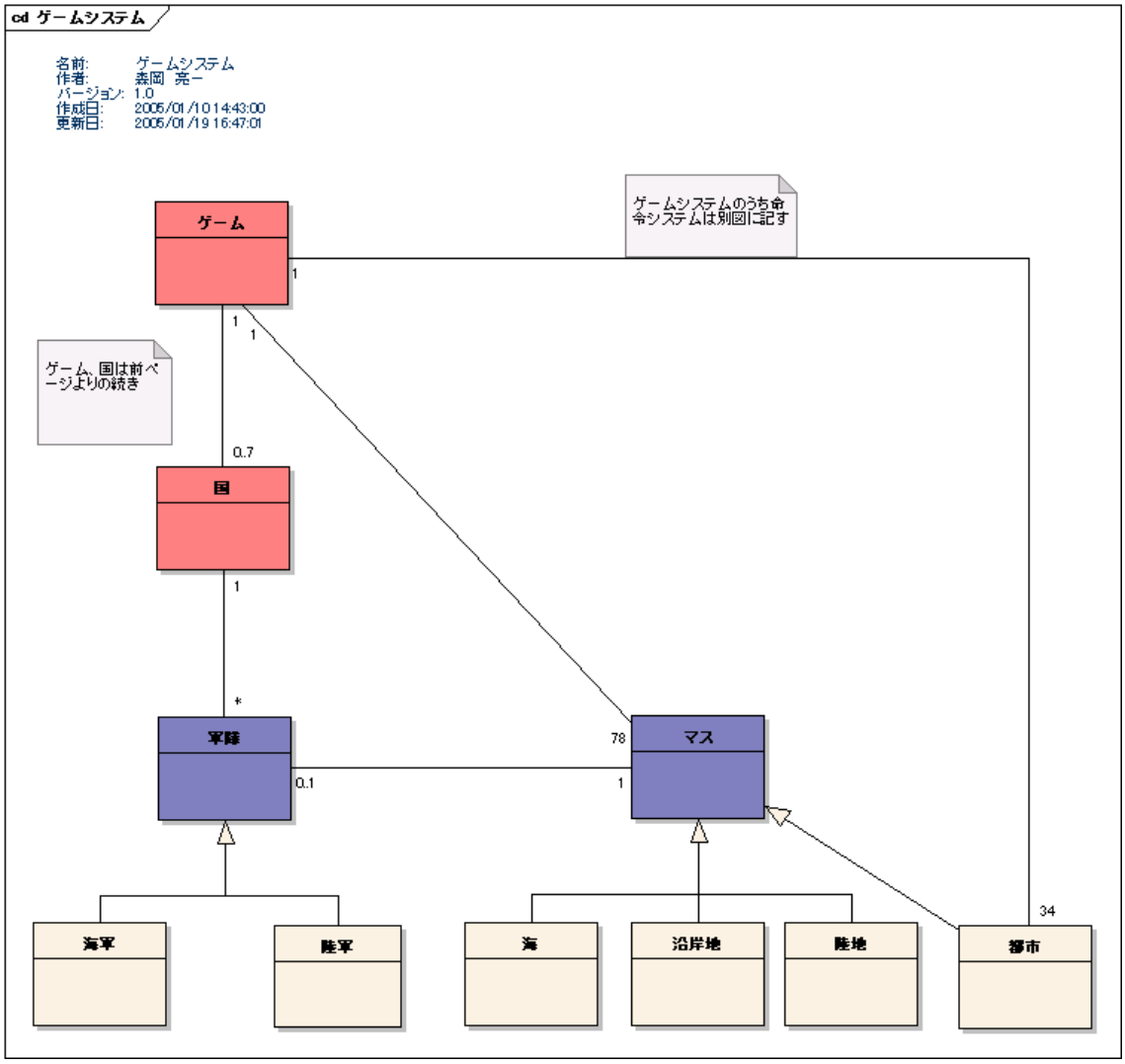
4 概念モデル

概念モデルはゲーム外システム、ゲームシステム、命令システムの三つの図に分けられる。それぞれの相関関係は図内に記述されている。

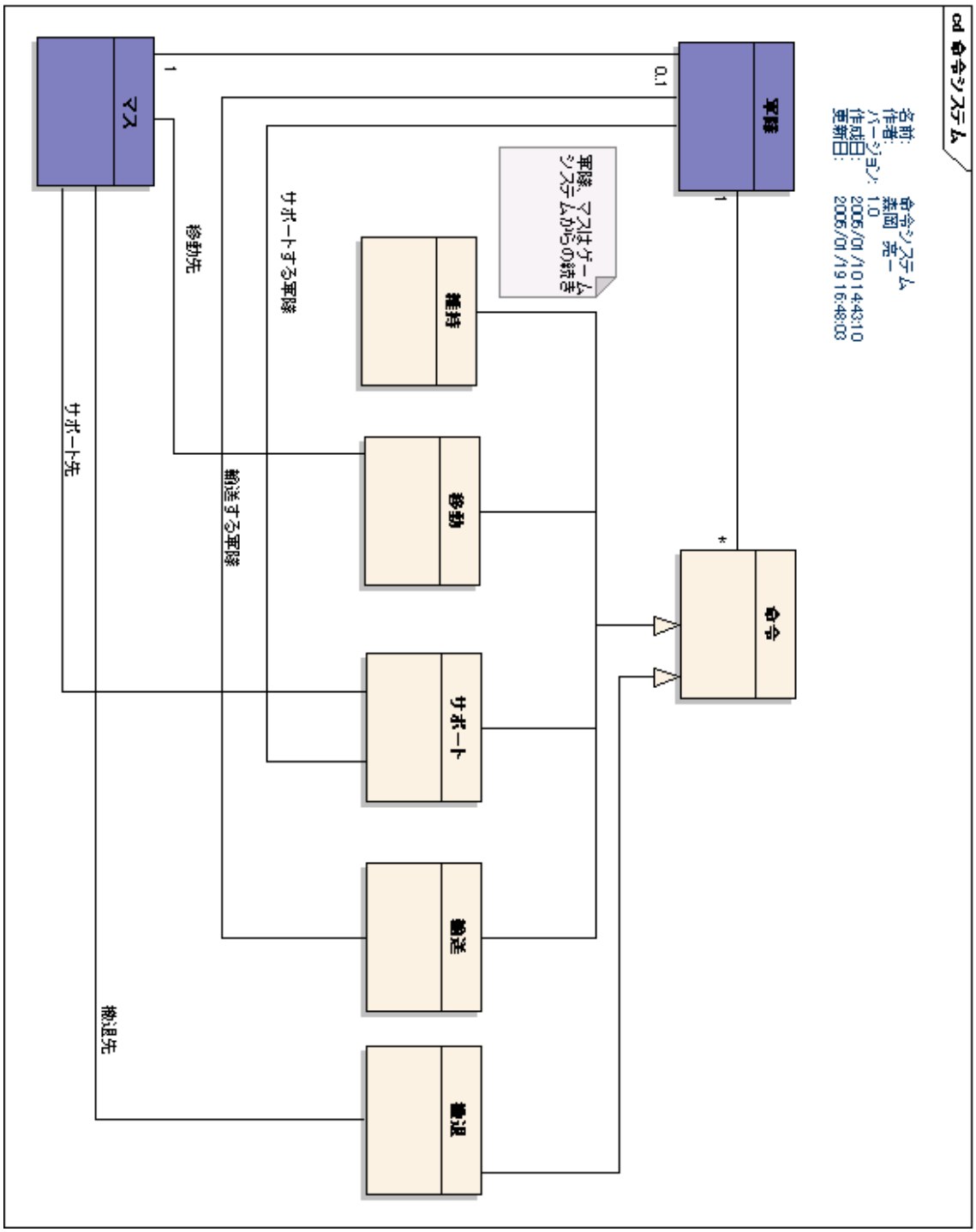
ゲーム外システム



ゲーム内システム


cd ゲームシステム
名前: ゲームシステム
作者: 森岡 亮一
バージョン: 1.0
作成日: 2005/01/10 14:43:00
更新日: 2005/01/19 16:47:01
ゲーム
ゲームシステムのうち命令システムは別図に記す
ゲーム、国は前ページよりの続き
国
軍隊
マス
海軍
陸軍
海
沿岸地
陸地
都市
1
1
1
0..7
1
*
78
0..1
1
34

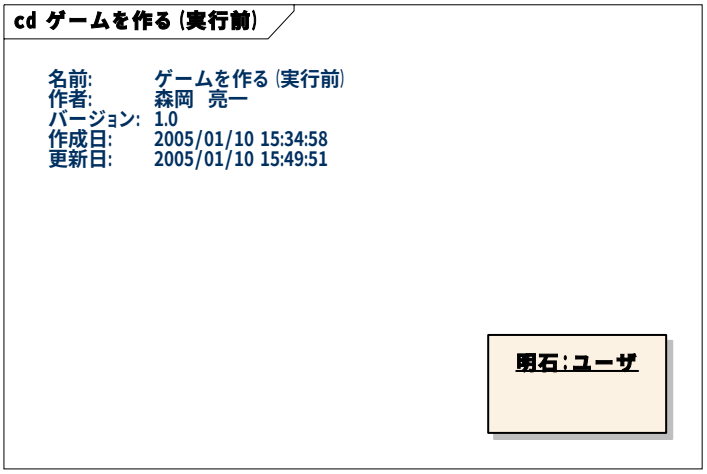
命令システム


cd 命令システム
名前: 命令システム
作者: 高岡 宏一
バージョン: 1.0
作成日: 2006/01/10 14:43:10
更新日: 2006/01/19 16:48:03
軍隊
マス
命令
維持
移動
サポート
輸送
撤退
0.1
1
*
1
軍隊、マスはゲームシステムからの続き
移動先
サポートする軍隊
輸送する軍隊
サポート先
撤退先

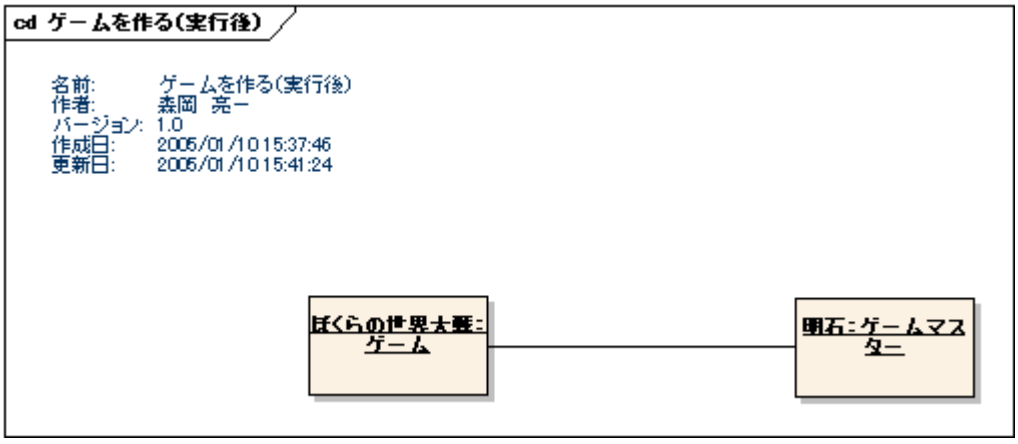
5 概念モデルインスタンス図

各ユースケースの実行前と実行後のインスタンスの変化を表している。

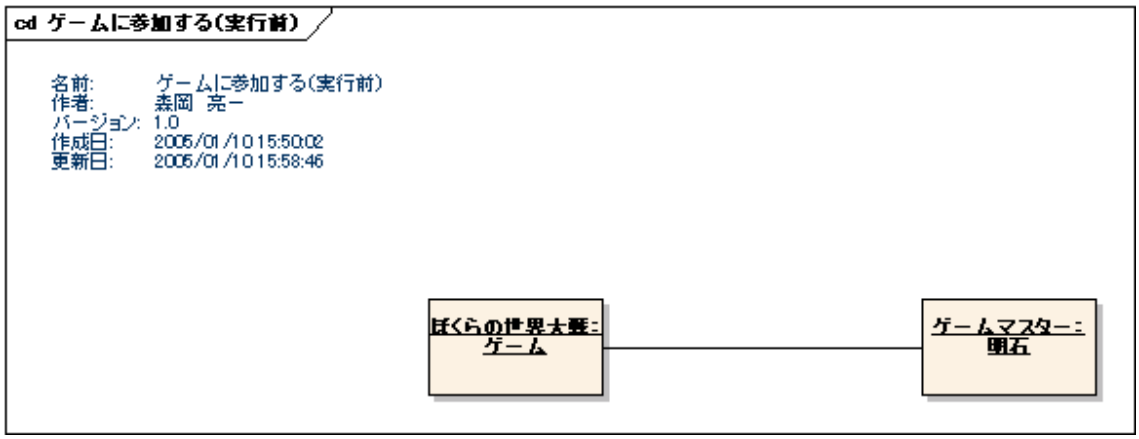
ゲームを作る



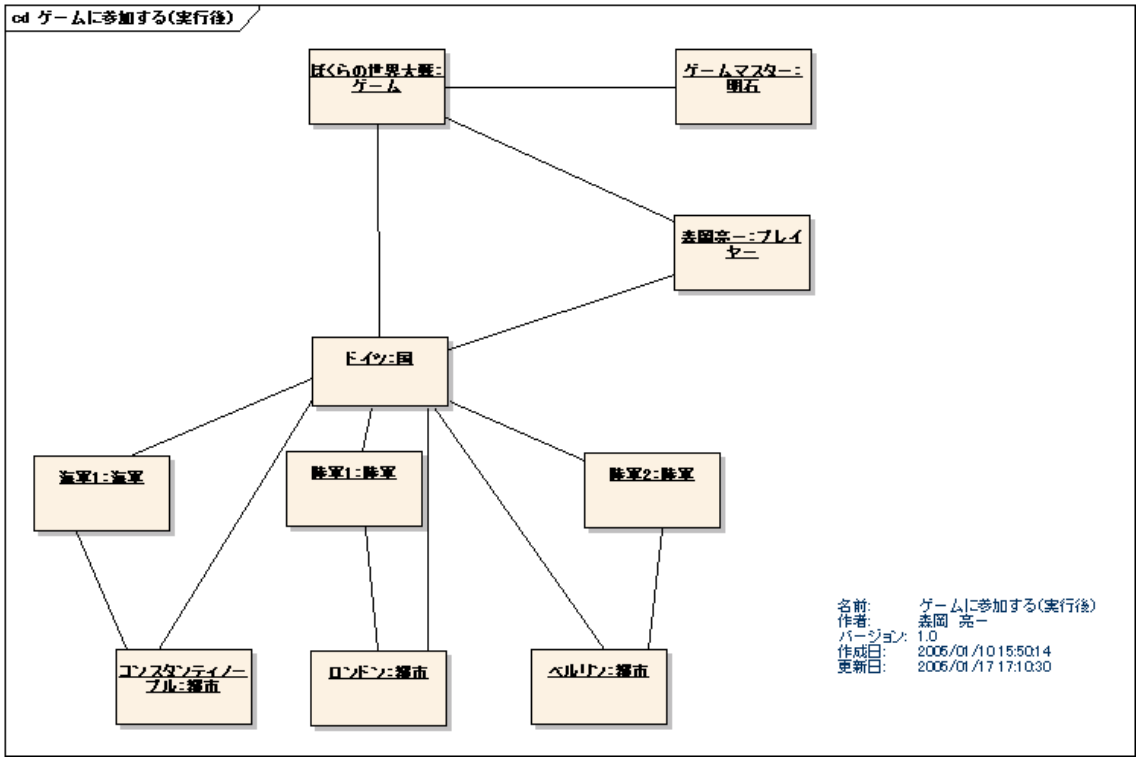
↓　明石さんがゲームを作成する。



ゲームに参加する

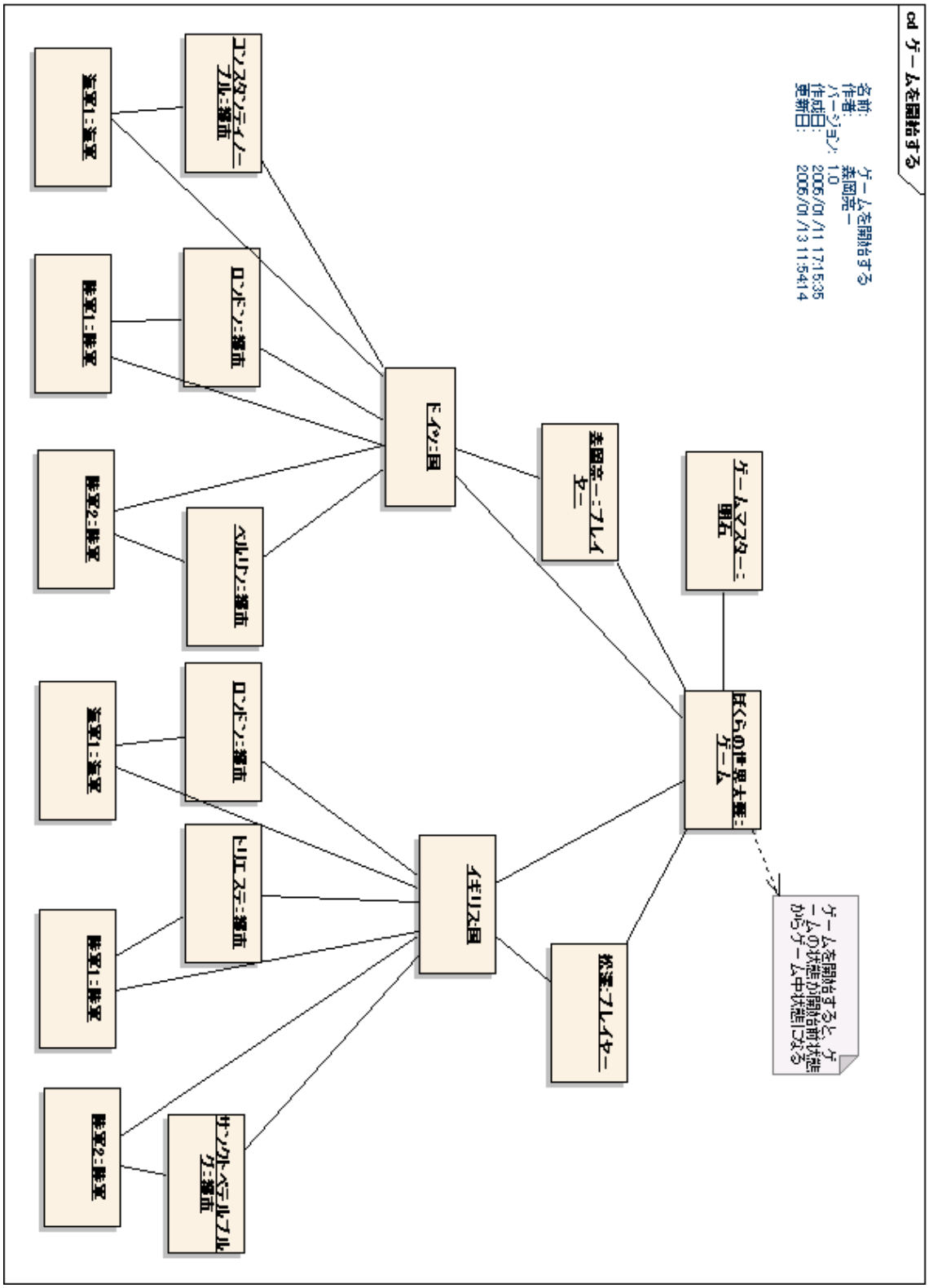


↓　森岡さんがゲームに参加する



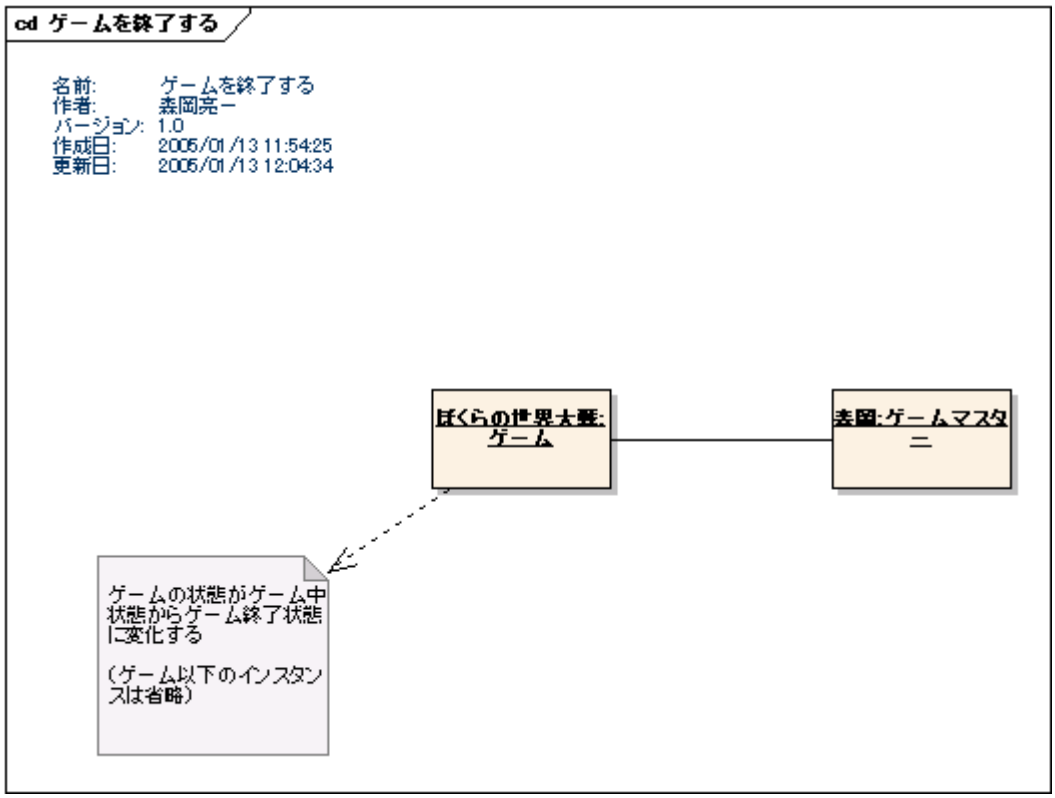
ゲームを開始する

実行前と実行後でインスタンスの変化はない。

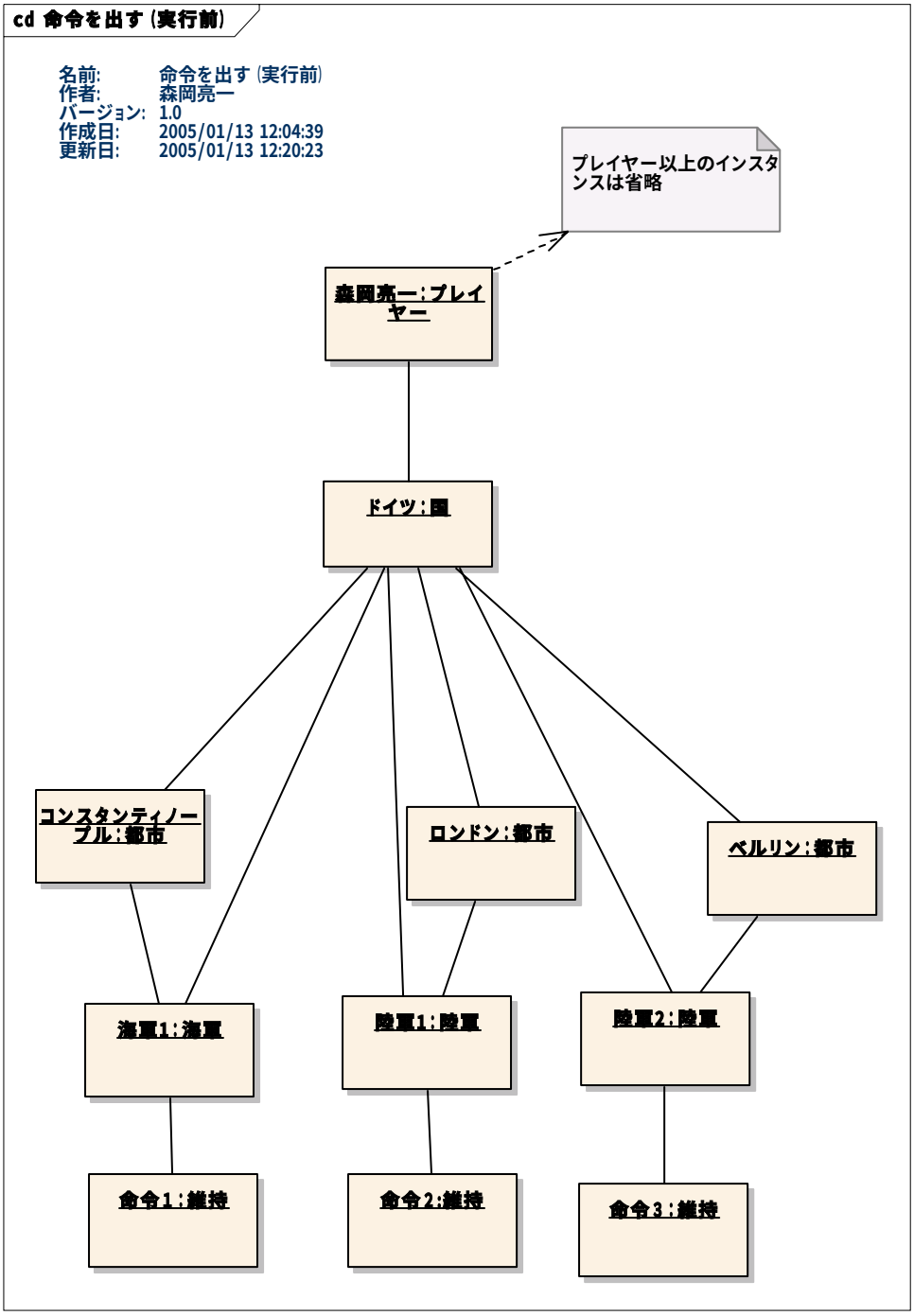


ゲームを終了する

実行前と実行後でインスタンスの変化はない。



命令を出す

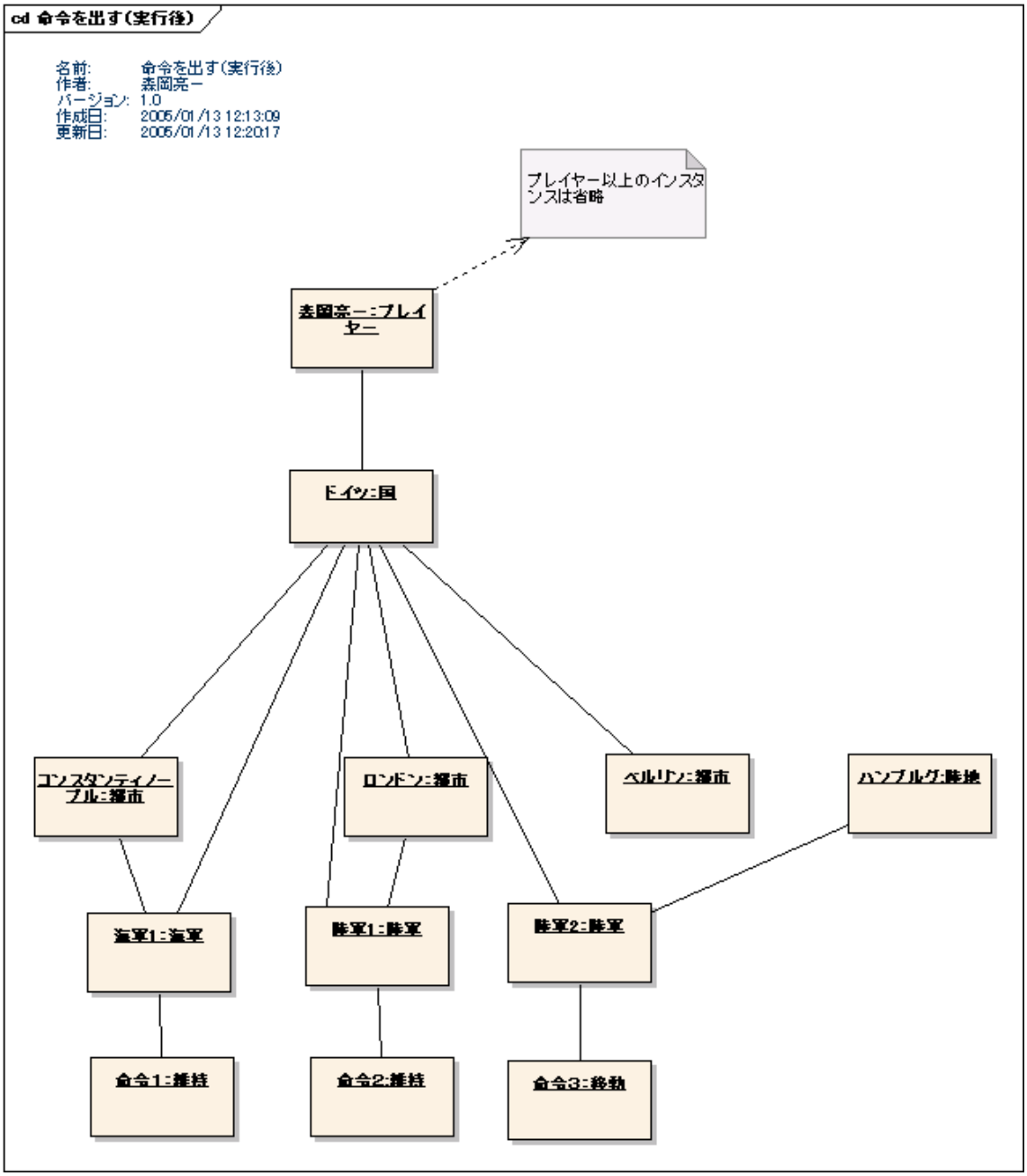


↓　森岡さんが命令を出す（実行後、次ページ）

命令内容　　海軍１：維持

陸軍１：維持

陸軍２：移動　ベルリン→ハンブルグ


cd 命令を出す(実行後)
名前: 命令を出す(実行後)
作者: 森岡亮一
バージョン: 1.0
作成日: 2005/01/13 12:13:09
更新日: 2005/01/13 12:20:17
プレイヤー以上のインスタンスは省略
森岡亮一:プレイヤー
ドイツ:国
コンスタンティノープル:都市
ロンドン:都市
ベルリン:都市
ハンブルグ:陸地
海軍1:海軍
陸軍1:陸軍
陸軍2:陸軍
命令1:維持
命令2:維持
命令3:移動

第二部

第二部の構成

第二部のドキュメントの構成は下記の通りである.

- Moodle 環境構築方法
  - Moodle のモジュール開発環境構築法，Moodle の運用環境構築法を説明する.
- Moodle フレームワークの説明
  - Moodle のモジュール作成に必要不可欠な Moodle フレームワークについて説明する.
- ディプロマシーアーキテクチャ説明書
  - ディプロマシーの全体アーキテクチャを明らかにし，ディプロマシーモジュールのファイル構成，データベース仕様について説明する.
- ディプロマシーfor Moodle マニュアル
  - ディプロマシーfor Moodle の操作方法を説明する.

Moodle **環境構築方法**

目次

1. このドキュメントについて

本書では diplomacy の環境構築方法について説明する．まず，Moodle モジュール開発を行う為の Windows への開発環境構築について説明する．その後，Linux での運用環境構築方法を説明する．

本書を読む前提として Windows, Linux の操作ができるものとする．

2. 開発環境構築方法

本節では, Windows マシンで Moodle を開発するための環境構築方法について説明する．Windows マシンで開発をするためには以下の４つのインストールが必要である．

- Apache2.0.52(Web サーバ)
- MySQL4.0.2(DB サーバ)
- PHP 4.3.9(プログラミング言語)
- Phpeclipse1.1.1(Eclipse プラグイン)
- Moodle(LMS)

以下にそれぞれの詳細な説明とインストール方法を述べる．

2.1. Apache2.0.52 のインストール

○ ダウンロード

Apache2.0 のインストーラを使用してインストールを行う．インストーラは http://httpd.apache.org/からダウンロードするか，noaのCVSからダウンロードする．CVS については，4 で述べている．

○ インストール

インストーラに従いインストールを行う. インストール先は「C:¥Program Files¥Apache Group」とします.

途中でサーバ情報の入力を求められるので，以下を入力してインストールを続ける．

NetworkDomain ： localhost

Server Name ： localhost

Administrator's Email Address ： root@localhost

○ 確認

インストールが完了したら Apache を起動して htto://localhost/にブラウザでアクセスしてみる．ページが表示されていればサーバーインストール成功.

2.2. MySQL4.0.2 のインストール

○ ダウンロード

MySQL4.0.2 のインストーラを使用してインストールを行う．インストーラは http://www.softagency.co.jp/MySQL/downloads/index.html からダウンロードするか，oa の CVS からダウンロードする．CVS については，4 で述べている．

○ インストール

ダウンロードしたファイルを解凍し，setup.exe を実行するとインストーラが起動する．インストーラに従ってインストールを行う．インストール先は「c:¥mysql」とする．また，インストール方法は「typical」を選択する．

○ パスを通す

インストールが終了したら「C:¥mysql¥bin」にパスを通す．

○ 確認

「C:¥mysql¥bin¥winmysqladmin」を実行する．Winmysqladmin を実行するとはじめにユーザー名とパスワードが求められる．はじめは root 以外のユーザーがいないので，以下のように入力する．

User: root

Password:

初期の root のパスワードは空なので空白のまま設定を行う．

winmysqladmin を実行するとタスクバーの常駐に信号のようなアイコンが追加される．この信号が緑色をしているときは mysql は起動中であり，赤色をしている場合は mysql は停止中である．このアイコンを右クリックし「winNT->start the service」とすると mysql は起動する．また，mysql 起動中にアイコンを右クリックし「winNT->stop the service」とすると mysql は停止する．

また，winmysqladmin を使用せず，コマンドプロンプトから mysql サーバを立ち上げる方法がある．

>mysqld-nt –standalone

上記のコマンドを打つとサーバが立ち上がる．

クライアントを立ち上げて動作確認をすることができる．

>mysql

画面が表示されればインストール成功である．

2.3. PHP4.3.9 のインストール

Moodle は PHP で記述されているため，PHP のインストールが必要である．PHP には CGI only 版と CGI plus server API 版の２つがある． CGI only 版だけでは，ファイルアップロードなどの一部の機能がうまく動かないので，CGI plus server API 版を使ってインストールを行う．

○ ダウンロード

http://www.php.net/downloads.php からCGI plus server API版をダウンロード, または, noaのCVSからダウンロードする．CVSについては，4 で述べている．

ダウンロードした zip ファイルを c:¥php に解凍する。

○ 設定ファイルの設置

C:¥PHP¥php.ini-dist を php.ini という名前に変更し，c:¥windows に置く．このファイルが PHP の設定ファイルとなる．

○ PHP の設定

Moodle を動かす為には以下の設定をしなければならない。以下の設定は全て php.ini で行う。

・ メモリの確保

Moodle を動かすためには 16MB のメモリを確保しなければならない。Php.ini で以下の設定を行う。

```
memory_limit = 16M
```

・ GD ライブラリ

画像を扱うためには GD ライブラリが必要である。Php.ini ではデフォルトで GD ライブラリを読み込まないように設定文がコメントアウトされている。以下の文のコメントアウトを外す。

```
extension=php_gd2.dll
```

・ デフォルト言語の設定

Moodle を動かしたときにちゃんと日本語が表示されるように以下の設定をする。

```
magic_quotes_gpc = Off
extension_dir = "C:/php/extensions"
extension=php_mbstring.dll  ;コメントアウトを外す
mbstring.encoding_translation = On
```

○ Apache の設定変更

Apache 上できちんと php が動くように設定を変更する必要がある。設定の変更は「C:¥Program Files¥Apache Group¥Apache2¥conf¥httpd.conf」で行う。以下の設定を追加する

```
LoadModule php4_module d:/php/sapi/php4apache.dll
AddModule mod_php4.c

AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

2.4. phpEclipse1.1.1 のインストール

PHP 言語を書くために Eclipse のプラグインである phpEclipse をインストールする．phpEclipse には以下の機能がある．

- php コード補完
- apache の起動
- mysql の起動
- web ブラウザ
- DB ブラウザ

○ ダウンロード

http://www.phpeclipse.de/tiki-view_articles.phpからプラグインをダウンロードするか，または，noaのCVSからダウンロードする．CVSについては，4 で述べている．インストール eclipse の plugins,features ディレクトリに zip ファイルを解凍し，配置する．

○ 設定

上に挙げた phpEclipse の機能を使用するために以下のように設定する．

- 「ウィンドウ->設定->PHP Web Development->PHP」の設定
  - Apache を Eclipse から起動できるように「Apache Settings->Start Apache」を「-c "DocumentRoot c:/myworkspace/crew/" -k start」のように設定する．「c:¥myworkspace/crew/」は Eclipse のワークスペースのトップディレクトリである．
  - Apache を Eclipse から停止できるように「Apache Settings->Stop Apache」を「-k shutdown」と設定する
  - Apache を Eclipse から再起動できるように「Apache Settings->Restart Apache」を「-k restart」と設定する
  - Apache のパスを設定する。「Apache Settings->Apacche」を「C:¥Program Files¥Apache Group¥Apache2¥bin¥Apache.exe」と設定する．
  - PHP のパスを設定する．「Apache Settings->Run PHP Command」を「C:¥PHP¥php.exe」と設定する．
  - MySQL のパスを設定する．「MySQL Settings->MySQL」を「C:¥mysql¥bin¥mysqld-nt.exe」と設定する．

2.5. Moodle1.41 のインストール

○ ダウンロード

Moodleはhttp://moodle.org/download/からダウンロードするか，または，noaのCVSからダウンロードする．CVSについては，4 で述べている．

○ インストール前準備

インストールをする前にデータベースに Moodle 用のデータベースを作らなくてはならない． MySQL が起動した状態でコマンドプロンプトにて「mysql -uroot」を実行する．「CREATE DATABASE moodle」というＳＱＬ文を実行し，データベースを作成する．ここでは「moodle」という名のデータベースを作成している．

○ インストール

- まず，ダウンロードしたファイルを解凍し，Apache のドキュメントルートに設置する．設置したディレクトリを「moodle」とする．
- 「http://localhost/moodle/」にアクセスするとインストールが始まる．
- インストール言語の選択
  - 「Japanese」を選択し、「Next」をクリック．
- PHP 設定の確認
  - 全てパスしていれば問題なし。警告が表示された場合はヘルプを参照し、対応する。
- インストール先の確認
  - ウェブアドレス，Moodle ディレクトリ，データの場所をそれぞれ設定する．デフォルトでかまわないので「次へ」をクリック．
- データベースの設定
  - データベースが作成できたら，データベースの設定を行う．「タイプ」でデータベースサーバの種類を選択する．ここでは「mysql」を選択する．
  - 「ホストサーバ」にはデータベースサーバ名を設定する．ここではデフォルトの「localhost」とする
  - 「データベース」には先ほど作成したデータベースの名前を入力する．ここでは「moodle」と入力する．
  - 「ユーザー」にはデータベースのユーザーを入力する．ここでは「root」としておく．
  - 「パスワード」にはデータベースのユーザーのパスワードを入力する．ここでは root にパスワードがまだ設定されていないため，空白のままとする．
  - 「テーブル接頭辞」とは moodle のデータベースのテーブル名のはじめにつく

文字列のことである．ここではデフォルトの「mdl_」のままにしておく．

- ✧ すべて入力したら「次へ」をクリックする．

- ➢ 「設定が完了しました」というメッセージが表示されれば，データベースの設定が完了．「続ける」をクリックする．
- ➢ 「著作権表示」が表示されるので，読んでから「Yes」をクリックする．
- ➢ データベーステーブルが作成されます．エラーが起きた場合は設定しなおしである．全て成功した場合「続ける」をクリックする．
- ➢ サイトの詳細設定
  - ✧ サイトで使用する言語などの詳細を設定する．説明を読みながら設定を行う．一度設定をした後でも設定しなおしは可能である．
  - ✧ 「変更を保存」をクリックする．
- ➢ Moodle のバージョンが表示される．「続ける」をクリックする．
- ➢ Moodle のリリース情報が表示される．「続ける」をクリックする．
- ➢ データベースの更新が行われる．全て成功したら「続ける」をクリックする．
- ➢ サイトセッティング
  - ✧ サイトの名前などサイトに関する情報を設定する．設定し終わった後からでも変更できる．
  - ✧ 「変更を保存」をクリックする．
- ➢ 管理ユーザープロフィールの設定．
  - ✧ ユーザー名の設定．デフォルトのまま「admin」としておく．
  - ✧ パスワードの設定．
  - ✧ メールアドレス，都道府県，国，自己紹介の設定を行う．入力されていないと注意されてしまう．
  - ✧ 「プロフィールの更新」をクリックする．

○ 確認

インストール終了後に Moodle トップ画面が表示されれば，インストール完了である．

○ 再インストール，再設定方法

インストールが終了すると config.php というファイルが moodle ディレクトリの直下にできる．このファイルにインストール時に設定した情報が記録されている．もし，Moodle のインストールに失敗し，再インストールを行いたい場合は，この config.php を削除し，「http://localhost/moodle」にアクセスすれば，再びインストールウィザードが表示され，再インストールを行う事ができる．

また，再インストールを行うまでもない場合は，直接 config.php の内容を書き換えればよい．

3. 運用環境構築方法 (Linux)

本章では，Moodle を Linux サーバで動かす為の環境構築方法について説明する．Linux は RedHatLinux9.0 を想定している．Moodle を Linux サーバで動かす為には以下の３つのソフトウェアが必要である．

- Apache2.0.40 （Web サーバ）
- MySQL4.0.2 （DB サーバ）
- PHP （プログラミング言語）
- Moodle(LMS)

以下にそれぞれの詳細な説明とインストール方法を述べる．

3.1. Apache2.0.40

RedHatLinux9.0 には初めから Apache2.0.40 が付属している．これで問題がないため，そのまま使う．

別のバージョンのApacheをインストールしたい場合は，http://www.apache.org/からダウンロードし，インストールする．

3.2. MySQL4.0.2 のインストール

○ ダウンロード

http://dev.mysql.com/downloads/mysql/4.0.html から mysql-server-4.0.21-0.i386.rpm、mysql-client-4.0.21-0.i386.rpm をダウンロードする．（サーバとクライアントで通常は事足りるようだ）noa の CVS からもダウンロードすることができる．CVS については，4 で述べている．

○ インストール

ダウンロードしてきたパッケージをインストールする

```
>rpm -i mysql-server-4.0.21-0.i386.rpm
>rpm -i mysql-client-4.0.21-0.i386.rpm
```

○ ルートの設定

外のサーバからデータベースを見る事がまだできないので，grant all を用いて root に権限を与える．

パスワードが未設定のため，パスワードも設定する．

```
>grant all on *.* to root identified by 'パスワード';
```

○ 動作確認

mysql クライアントを使って，サーバと通信できるか確認する．

```
>mysql -h データベースホストサーバ名-u root –p
Enter:パスワード
```

画面が表示されれば，インストール成功.

3.3. PHP4.3.9 のインストール

○既存の PHP のアンインストール

RedHatLinux にすでにインストールされている PHP は DB と通信をするためのコネクタが入っていない．そのため，まず，すでにインストールされている PHP を削除しなければならない.

```
>rpm -e php-imap
>rpm -e php-ladp
>rpm -e php
```

○ ソースのダウンロード

PHPのソースコードはhttp://www.php.net/downloads.phpからダウンロードするか, noaのCVSからダウンロードする．CVSについては，4CVSを参照してほしい.

ダウンロードしたソースコードを展開する.

○ PHP の make に必要なパッケージのインストール

Configure のオプションで必要なパッケージをインストールする. インストールするのは以下のパッケージである.

zlib-1.1.4-8.8x.i386.rpm	libpng で使用
zlib-devel-1.1.4-8.8x.i386.rpm	libpng で使用
libjpeg-6b-26.i386.rpm	JPEG の画像処理
libjpeg-devel-6b-26.i386.rpm	JPEG の画像処理
libpng-1.2.2-8.i386.rpm	PNG の画像処理
libpng-devel-1.2.2-8.i386.rpm	PNG の画像処理

○ Configure の設定

PHP のパッケージをインストールするための設定を行う.

```
>./configure --with-apxs2filter=/usr/sbin/apxs --with-gd --with-png-dir=/usr --with-zlib
--with-jpeg-dir=/usr --enable-exif --enable-mbstring --enable-mbregex
--enable-zend-multibyte --enable-memory-limit
```

○ インストール

Configure に成功したら make を行い，インストールする．

```
>make
>make install
```

○ Apache の設定変更

/etc/httpd/conf/httpd.conf に以下を追加

```
LoadModule php4_module modules/libphp4.so（make install で追加される）

DirectoryIndex index.html index.html.var index.php (index.php を追加)

AddType application/x-httpd-php .php4 .php3 .phtml .php
AddType application/x-httpd-php-source .phps
```

○ PHP の設定

PHP の設定ファイルを作成する．「php.ini-dist」というファイルを「php.ini」にリネームし、/usr/local/lib にコピーする．作成しなくともデフォルト状態で動作するが、細かい設定を行う場合必要になる．

```
>cp php.ini-dist /usr/local/lib/php.ini
```

3.4. Moodle1.4.1 のインストール

Moodle のインストールは Windows の場合と変わりがないので，Windows 版を参照していただきたい．

Virtualhost の設定のみ以下に示す．

/etc/httpd/conf/httpd.conf に以下の記述を追加する．

```
<VirtualHost IP:ポート番号(例：133.27.173.232:80)>
    ServerName サーバーの名前（例：lms.crew.sfc.keio.ac.jp）
    DocumentRoot ドキュメントルート(例：/home/httpd/moodle1)
</VirtualHost>
```

上記のようにすることで，「http;//lms.crew.sfc.keio.ac.jp」にアクセスするだけで moodle へアクセスできるようになる．

4. CVS

環境構築に必要な全てのソフトウェアは noa の CVS にアップされている．CVS のパスは「lms-dev/moodle-setup」である．「lms-dev/moodle-setup」以下は次のようなディレクトリ構成になっている．

- Common
  - Moodle や phpEclipse など Windows, linux　共通で使うソフトウェアが置いてある．
- windows
  - windows での環境構築に必要なソフトウェアが置いてある
- linux
  - linux での環境構築に必要なソフトウェアが置いてある．

Moodle **フレームワークの説明**

目次

1. このドキュメントについて

本書ではディプロマシーを開発する上で必要不可欠な Moodle フレームワークについて説明する．本書を読む前提として，PHP の知識があるものとする．

2. Moodle の概念

本章で必要な Moodle の概念について説明する．以下が Moodle の概念図である．

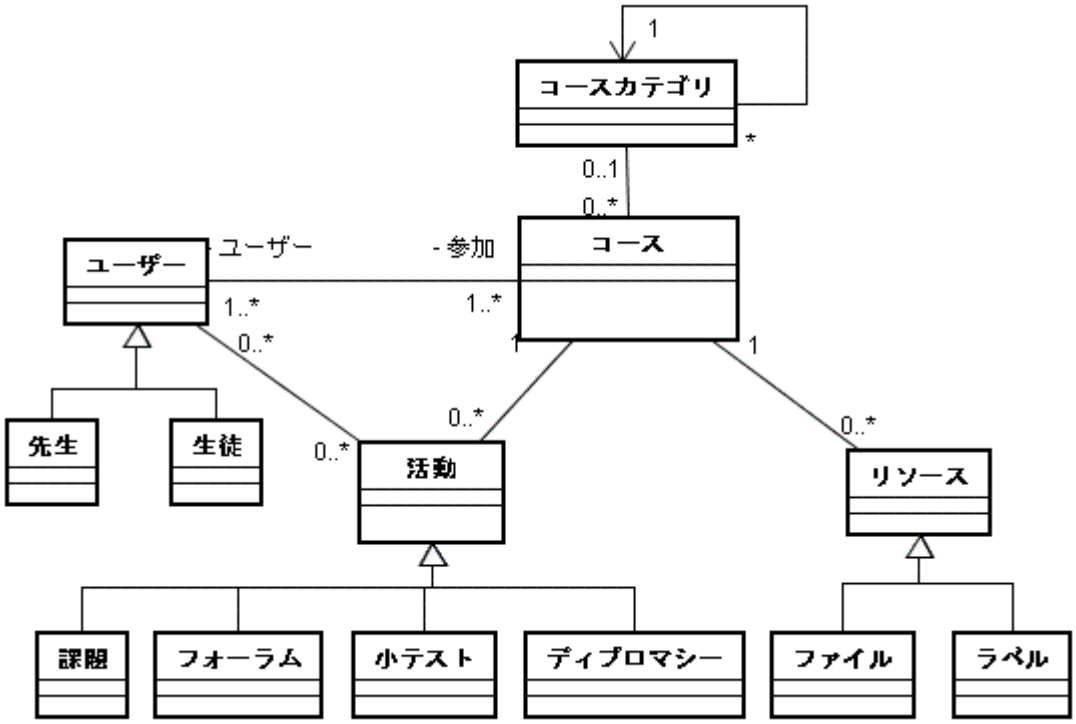


図 1

活動やリソースのサブクラスは他にもいくつか存在するが，説明のための図なので割愛した．

以下にそれぞれの概念の詳細について述べる．

2.1. ユーザー

ユーザーとは Moodle の利用者である．Moodle の利用者には「先生」と「生徒」がいる．

2.2. コース

コースとはいわゆる授業のことである．Moodle にはコースが複数存在する．ユーザーはコースに参加する事ができる．また，先生はコースを作成する事ができるが，生徒はコースを作成することができない．

2.3. コースカテゴリ

コースカテゴリとは各コースを分類する為の概念である．すべてのコースが何らかのコースカテゴリに属さなければならないわけではない．

2.4. 活動

活動とはコースで行われる「課題」や「クイズ」「フォーラム」といったものの抽象概念である．Moodle モジュールを作成することで追加できる活動が増える．

2.5. リソース

リソースとはコースで使用される文書，プレゼンテーション，映像ファイルなどのコースのコンテンツである．ファイルをアップロードしたり，URL を使用したリンクをすること

で，どんなファイルでもリソースにすることができる．

3. Moodle モジュール

Moodle では活動がそれぞれモジュール化されている．

モジュールを新規開発し Moodle にインストールすることで，先生が追加できる活動の種類を増やすことができる．

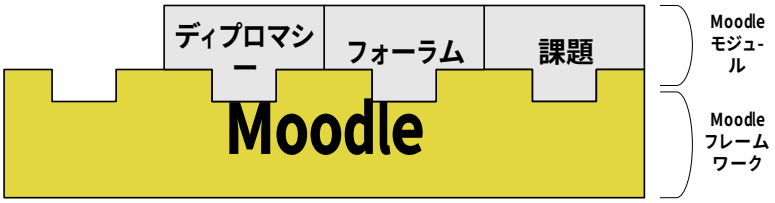


図 2

次節からモジュールの詳細について述べる．

3.1. モジュールのパス

モジュールは moodle のトップディレクトリ直下の「mod」の中に定義されている．

3.2. モジュールのファイル構成

Moodle モジュールは以下のようなファイル構成になっている．これらのファイルは Moodle のフレームワークから呼ばれるので確実にモジュールディレクトリの中に存在しなければならない．

- icon.gif
- index.php
- lib.php
- mod.html
- version.php
- view.php
- db/ mysql.php
- db / mysql.sql

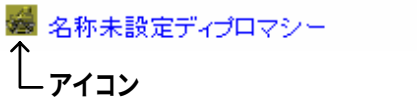
それぞれの詳細については次節で述べる．

3.3. モジュールファイルの説明

前述したモジュールファイルの詳細について述べる．

3.3.1. icon.gif

モジュールのアイコン画像ファイル．活動をトピックに追加した時に表示される．



3.3.2. index.php

モジュールの一覧画面．Moodle 上部のナビゲーションのモジュール名をクリックするとコースに登録されているモジュールが一覧表示される．

図 3

3.3.3. lib.php

Moodle のフレームワークから呼ばれるメソッドを定義したファイル．以下のメソッドを含まなければならない．モジュール名を「NEWMODULE」とした場合のメソッド名を示す．

新しい活動を追加する	NEWMODULE_add_instance()
活動を更新する	NEWMODULE_update_instance()
活動を削除する	NEWMODULE_delete_instance()
ユーザーの行動の概要を取得する	NEWMODULE_user_outline()
ユーザーの行動の詳細を取得する	NEWMODULE_user_complete()
最新の活動を表示する	NEWMODULE_print_recent_activity()
一定の間隔で処理を行う	NEWMODULE_cron()
活動の評価のリストを取得する	NEWMODULE_grades()
活動に参加しているユーザーを取得する	NEWMODULE_get_participants()
活動でスケールを使っているか確認する	NEWMODULE_scale_used()

以下に各メソッドの詳細を述べる．

・ 新しい活動を追加する

モジュールオブジェクト ↓ NEWMODULE_add_instance($NEWMODULE)

コースに新しい活動を追加したときに呼ばれるメソッド．このメソッドの中では追加された活動の作成日時を設定するなどの初期化の処理を記述する必要がある．

・ 活動を更新する

モジュールオブジェクト ↓ NEWMODULE_update_instance($NEWMODULE)

すでにコースに登録されている活動の内容を更新した場合に呼ばれるメソッド．このメソッドの中に更新処理を記述する必要がある．

・ 活動を削除する

モジュール ID ↓ NEWMODULE_delete_instance($NEWMODULE_id)

すでにコースに登録されている活動を削除した場合に呼ばれるメソッド．このメソッドの中で DB からレコードを削除するなどの削除処理を記述する必要がある．

・ ユーザーの行動の概要を取得する

コース ユーザー モジュール モジュールオブジェクト ↓ ↓ ↓ ↓ NEWMODULE_user_outline($course, $user, $mod, $NEWMODULE)

ユーザーの活動レポートに表示する為の情報を取得するメソッド．return false;だけでも構わないが，その場合，活動レポートには表示されない

・ ユーザーの行動の詳細を取得する

```
                                コース ユーザー モジュール モジュールオブジェクト
                                  ↓      ↓    ↓          ↓
NEWMODULE_user_complete($course, $user, $mod, $NEWMODULE)
```

ユーザーの活動レポートに表示する為の情報を取得するメソッド．NEWMODULE_user_outline メソッドよりも詳細な情報を取得する．return false;だけでも構わないが，その場合，活動レポートには表示されない

・ 最近の活動を表示する

```
                                      コース      先生か   活動の開始時刻
                                       ↓          ↓          ↓
NEWMODULE_print_recent_activity($course, $isteacher, $timestart)
```

活動内で行われた最近の活動を表示するメソッド．コースページの最近の活動欄に表示される内容を記述する．

・ 一定の間隔で処理を行う

```
NEWMODULE_cron()
```

crontab によって呼ばれるメソッド．一定の間隔で処理を行いたい場合は，このメソッドに処理を記述すればよい．

・ 活動の評価のリストを取得する

```
                                  モジュール ID
                                       ↓
NEWMODULE_grades($NEWMODULE_id)
```

活動を行ったユーザーの評価のリストを取得する．

・ 活動に参加しているユーザーを取得する

```
                                  モジュール ID
                                       ↓
NEWMODULE_get_participants($NEWMODULE_id)
```

活動に参加しているユーザーを取得するメソッド．どこで利用しているかは調査中である．

・ 活動でスケールを使っているか確認する

```
                                モジュール ID     スケール ID
                                      ↓              ↓
NEWMODULE_scale_used($NEWMODULE_id,$scaleid)
```

活動でスケールを使っているかを確認するメソッド．スケールについては調査中である．

3.3.4. mod.html

コースに活動を新規登録するときや活動を更新するときの活動の設定画面．「/course/mod.php」によって呼ばれているため，拡張子が html であるが，php スクリプトを書くこともできる．

図 4

3.3.5. version.php

モジュールのバージョンと cron の間隔を設定するファイル．以下の２つの変数が定義されている．

・ $module->version
　現在のモジュールのバージョンを表す変数. 日付を YYYYMMDDXX の形式で設定する．

例えば 2005 年 1 月 5 日のモジュールの場合，「20050105」と記述する．

・ $module->cron

cron の間隔を表す変数．秒単位で設定する．

3.3.6. view.php

活動のメイン画面．コースに登録された活動のリンクをクリックした場合に呼び出される．

図 5

3.3.7. db/mysql.php

モジュールをアップグレードするためのメソッドが一つ用意されている．モジュールを更新したときに呼び出される．モジュールを更新したときのテーブル追加，削除，テーブルのカラム追加などの SQL 文をこのメソッドの中に記述する．以下にメソッドを示す．

```
                更新前のバージョン
                        ↓
diplomacy_upgrade($oldversion)
```

3.3.8. db/mysql.sql

モジュールで使用するデータベースを初期化する SQL を記述するファイル．モジュールインストール時に一度だけ呼び出される．

モジュールで使用するデータベーステーブル作成の SQL 文を記述しなければならない．mysql.sql にデータベーステーブル作成の SQL 文を書くときは，以下のようにデータベー

ステーブル名の始めに prefix_を付けなければならない．こうする事で，Moodle インストール時に設定したデータベーステーブル接頭辞をデータベーステーブル名の始めにつけることができる．

```
create table `prefix_diplomacy_city`(
        `id` int(16) unsigned NOT NULL auto_increment,
        `game_id` int(16) unsigned NOT NULL,
        `player_id` int(16) unsigned,
        `space_id` int(16) unsigned NOT NULL,
        PRIMARY KEY  (`id`)
);
```

4. Moodle モジュール開発ポリシー

Moodle のモジュール開発のポリシーでは，lib.php ファイルにモジュールで使うすべてのメソッドを定義する，というものがあるが，これは lib.php を肥大化させわかりづらいコードにしてしまう．

また, Moodle フレームワークから呼ばれる php ファイルとその他のファイルを同じディレクトリで管理するのでどの php ファイルがフレームワークに依存しているかわかりにくくなってしまう．

これらについてはディプロマシーモジュールの説明でどのようにすればわかりやすいモジュール開発が行えるかを説明する．

ディプロマシーアーキテクチャ説明書

1. 目次

2. 本書の構成

本書の構成は以下のようになっている.

- 全体アーキテクチャの説明
- ディプロマシーモジュールの説明
- データベース仕様

3. 全体アーキテクチャの説明

本章では，ディプロマシーの全体アーキテクチャを図で示す．まず，ディプロマシーを動かす為のフレームワークである Moodle の全体アーキテクチャを示し，その後，ディプロマシーについて述べる.

3.1. Moodle アーキテクチャ

まず，全体を把握しやすいようにディプロマシーを動かすフレームワークである Moodle のアーキテクチャ構成図を図 1 に示す.

Moodle は Web アプリケーションななので，動かすためには Web サーバが必要である．今回は Web サーバに Apache を使用している.

Moodle のユーザー情報などの様々な情報を管理するための DB には MySQL を用いた．Moodle は PostgreSQL にも対応しているが，開発用マシンに Windows を利用する為，Windows にも簡単にインストールできる MySQL を選んだ.

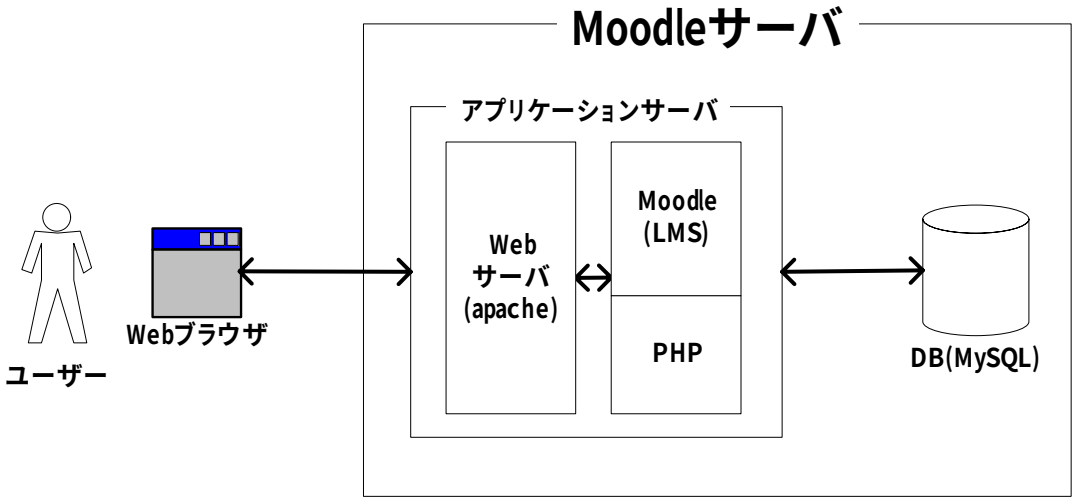


図 1

3.2. ディプロマシーアーキテクチャ

ディプロマシーは LMS(Learning Management System)である Moodle のモジュールとして実装されている．Moodle はサーバサイドプログラムであり，スクリプト言語の PHP

で実装されている．

Moodle のモジュールとして実装する事により，ユーザー管理は既に Moodle に用意されたものを使うことができる．Moodle モジュールの詳細は，Moodle フレームワークの説明を参照してほしい．

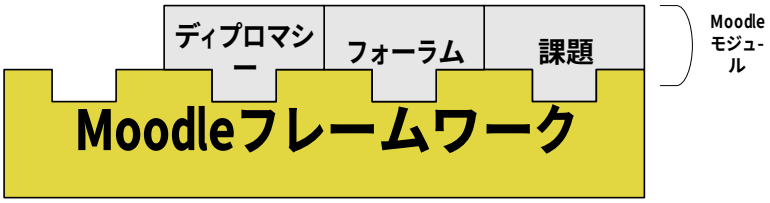


図 2

4. ディプロマシーモジュールの説明

4.1. ファイル構成

ディプロマシーモジュールには，フレームワークに依存した部分と依存していないディプロマシー特有の部分とで構成されている．新しく「diplomacy」というディレクトリをモジュールのトップディレクトリの下に作り，フレームワークに依存していない部分をそのディレクトリの中に置くようにした．以下がディプロマシーモジュールのファイル構成である．

- icon.gif
- mod.html
- mods.html
- index.php
- lib.php
- version.php
- view.php
- db/
  - mysql.php
  - mysql.sql
- diplomacy/
  - action_finish_game.php
  - action_finish_order.php
  - action_finish_turn.php
  - action_join.php
  - action_order.php
  - action_restart_game.php
  - action_start_game.php
  - diplomacy.php
  - viewutil.php
  - gameview_post_processing_of_turn.php
  - gameview_terminated.php
  - gameview_waitingfor_start.php
  - gameview_waitingfor_order.php
  - mapview_map.php
  - orderview_input_target.php
  - img/　(各種画像ファイル)

フレームワークに依存していない，ディプロマシーモジュール特有のファイルについて以下に説明する．フレームワークに依存している部分は Moodle フレームワークの説明を参照してほしい．

ディプロマシーモジュール特有のファイルは以下の４つに分類できる．

・ アクション
・ 画面
・ ライブラリ
・ リソース

それぞれの詳細を以下に述べる．

4.2. 画面

画面とは，ゲーム設定画面や戦況画面を表示させる php ファイルのことである．

4.2.1. view.php との関係

本来，view.php が画面を担当しているが，ディプロマシーモジュールでは，view.php は画面表示の処理は行わず，ゲームの状態によって画面を遷移させている．

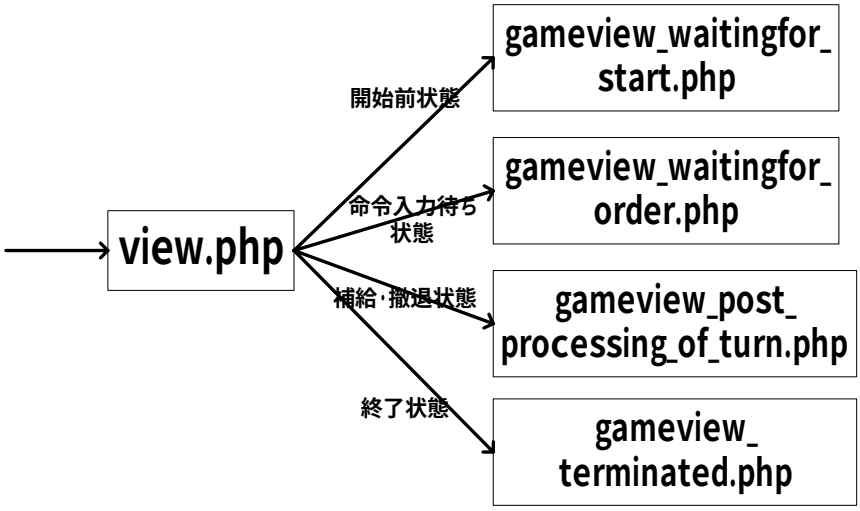


図 3

4.2.2. 基本構造

画面は必ず以下のような hcp チャートの構造になっていなければならない．始めに前処理を行う．その後，状態が正しいかチェックする．ここで状態が正しくない場合は，警告メッセージが表示される．続いて画面を表示する．画面はヘッダー，本体，フッターで構成されている．

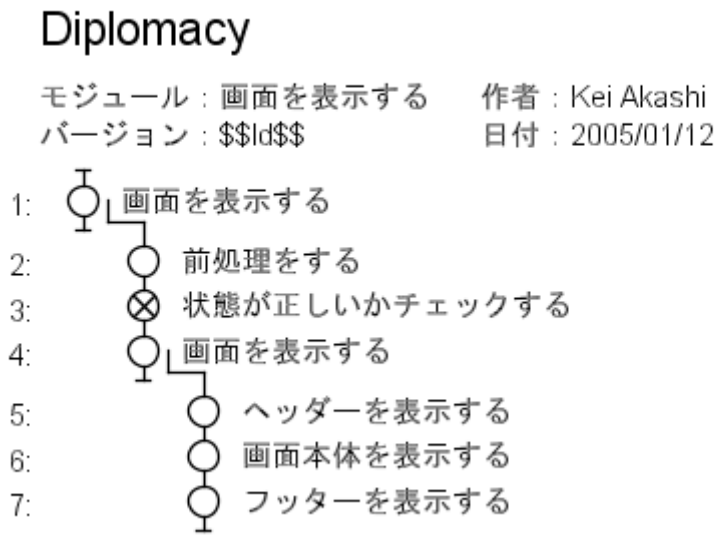


図 4

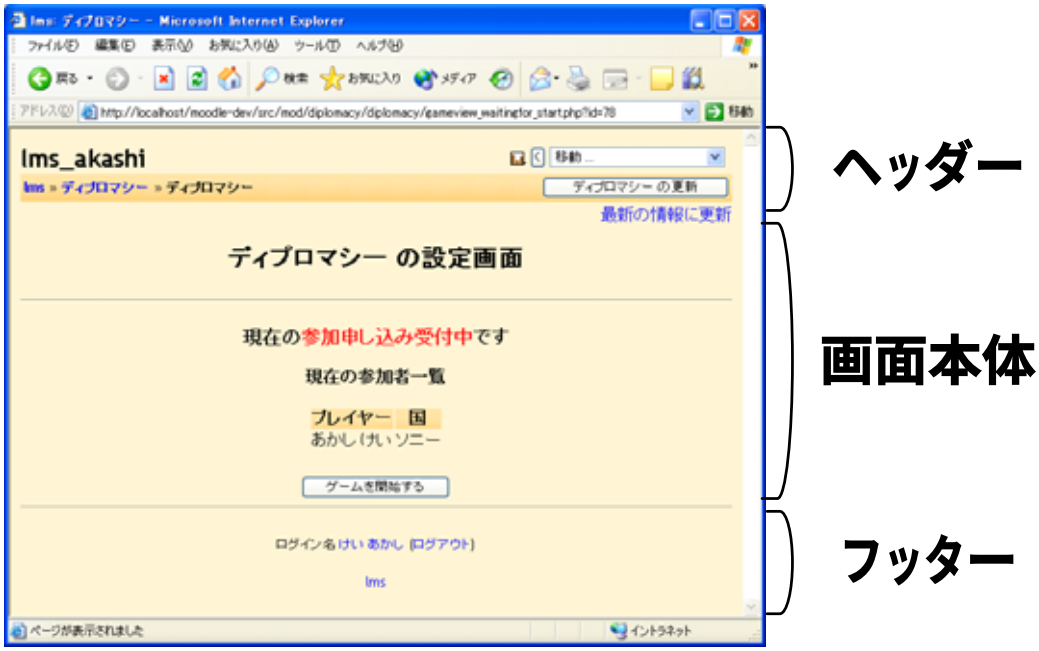


図 5

4.2.3. **画面ファイルとその役割**

画面には以下のファイルが含まれる．それぞれの役割について説明する．

- gameview_waitingfor_start.php
  - ゲーム開始待ち画面

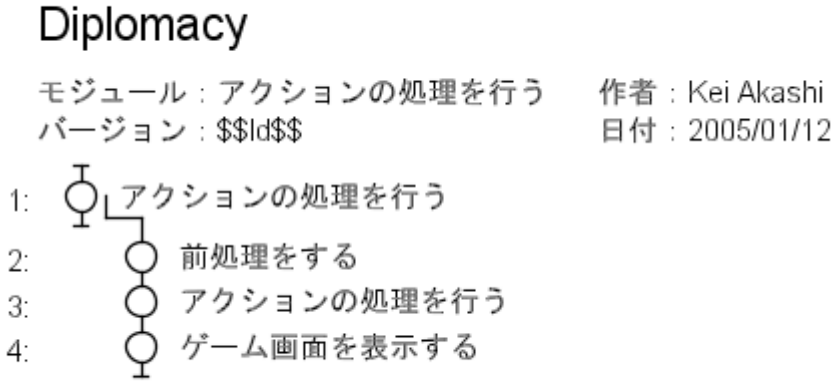
- gameview_post_processing_of_turn.php
  - ターン終了画面．撤退・補給を行う．
- gameview_terminated.php
  - ゲーム終了画面．
- gameview_waitingfor_order.php
  - 命令入力待ち画面
- mapview_map.php
  - マップ表示画面
- orderview_input_target.php
  - 命令ターゲット入力画面

4.3. アクション

アクションとはユーザーの入力などによって発生したイベントによって実行される処理のことである．

4.3.1. 基本構造

アクションの基本構造は以下の hcp チャートのようになっている．必ず前処理を先頭で行い，本処理を記述し，ゲーム画面を表示する処理を行っている．



例としてゲーム開始アクションのソースコードを示す．前処理をする，ゲームを開始する（アクションの処理を行う），ゲーム画面を表示する　という３つのブロックで構成されていることが確認できる．

```
1  <?php
2
3  /*
4   * Created on 2004/12/09
5   * ゲーム開始アクション
6   * @author rocky
7   * @version $Id: action_start_game.php,v 1.3 2005/01/06 03:10:08 rocky Exp $
8   */
9
10 require_once ("../../../config.php");
11 require_once ("diplomacy.php");
12 require_once("viewutil.php");
13
14 //前処理
15 diplomacy_get_diplomacy_instance($id, $a, $course_module, $course, $diplomacy);
16 require_login($course->id);
17 diplomacy_require_gamemaster($diplomacy, $USER);
18
19 //ゲームを開始する
20 diplomacy_start_game($diplomacy);
21
22 //ゲーム画面を表示する
23 diplomacy_show_gameview($diplomacy, $course_module);
24
25 ?>
```

上の例では，10 行目～12 行目の

```
require_once ("../../../config.php");
require_once ("diplomacy.php");
require_once("viewutil.php");
```

で必要なライブラリを読み込んでいる．こうすることで moodle のライブラリやディプロマシーのライブラリを使うことができる．

また，14 行目～17 行目の

```
//前処理
diplomacy_get_diplomacy_instance($id, $a, $course_module, $course, $diplomacy);
require_login($course->id);
diplomacy_require_gamemaster($diplomacy, $USER);
```

では，前処理を行っている．15 行目ではモジュールのインスタンスを取得し，16 行目では，ユーザーがログインしているかどうかを調べている．15 行目、16 行目の処理はどのアクションでも必ず行っている．17 行目の処理はアクションによって異なる．

19 行目～20 行目の

```
//ゲームを開始する
diplomacy_start_game($diplomacy);
```

で，アクションの本処理を行っている．アクションの本処理はメソッドとして diplomacy.php に定義する．

22 行目～23 行目の

```
//ゲーム画面を表示する
diplomacy_show_gameview($diplomacy, $course_module);
```

で，アクション実行後の画面へ遷移する処理を行っている．

4.3.2. **アクションファイルとその役割**

以下にアクションファイルを挙げ，その役割を述べる．

- action_start_game.php
  - ゲームを開始する
- action_join.php
  - ゲームに参加する
- action_order.php

- 軍隊に命令を出す

・ action_finish_order.php
- 命令待ち状態を終了させる

・ action_finish_turn.php
- ターンを終了させる

・ action_finish_game.php
- ゲームを終了させる

・ action_restart_game.php
- 終了したゲームを再開する

4.4. ライブラリ

ライブラリはディプロマシーモジュールで使用するメソッドが定義されたファイルである.

・ diplomacy.php

diplomacy.php には以下に関連する操作を定義している. それぞれについて詳細を述べる.

- 状態関連
  - 状態を操作するためのメソッドが定義されている.
- ユーザーの権限関連
  - ユーザーの権限を取得や設定を行うためのメソッドが定義されている.
- Action 関連
  - アクションの処理に深く関わるメソッドが定義されている.
- Get 関連
  - DB から情報を取得するためのメソッドが定義されている.

ディプロマシーモジュールにある全てのファイルは diplomacy.php を require している. こうすることで全てのファイルで diplomacy.php に定義されている定数やメソッドを使うことができるのである.

・ viewutil.php

viewutil.php は以下に関連する操作を定義している. それぞれについて詳細を述べる.

- 画面遷移関連
  - 画面遷移を行う為のメソッドが定義されている.
- 表示用関連
  - 表示に使うメソッドが定義されている.

画面ファイルは viewutil.php を require している．こうすることで画面ファイルは viewutil.php に定義されている表示用メソッドを利用する事ができる．

また，アクションも viewutil.php を require することで画面遷移関連の操作を行う事ができる．

4.5. リソース

リソースとは画像ファイルなどの部品のことである．画像ファイルは img ディレクトリの中に入れてある．画像ファイルは主にマップ画像と軍隊を表すコマの画像がそれぞれある．

4.6. 命名規則

ディプロマシーモジュールのファイル名の命名規則を示す．

4.6.1. 全体

単語ごとに「_」をいれたファイル名を付ける．大文字は使わない．

4.6.2. 画面

画面ファイルは必ず「???view_」の形ではじまるファイル名が付けられている．

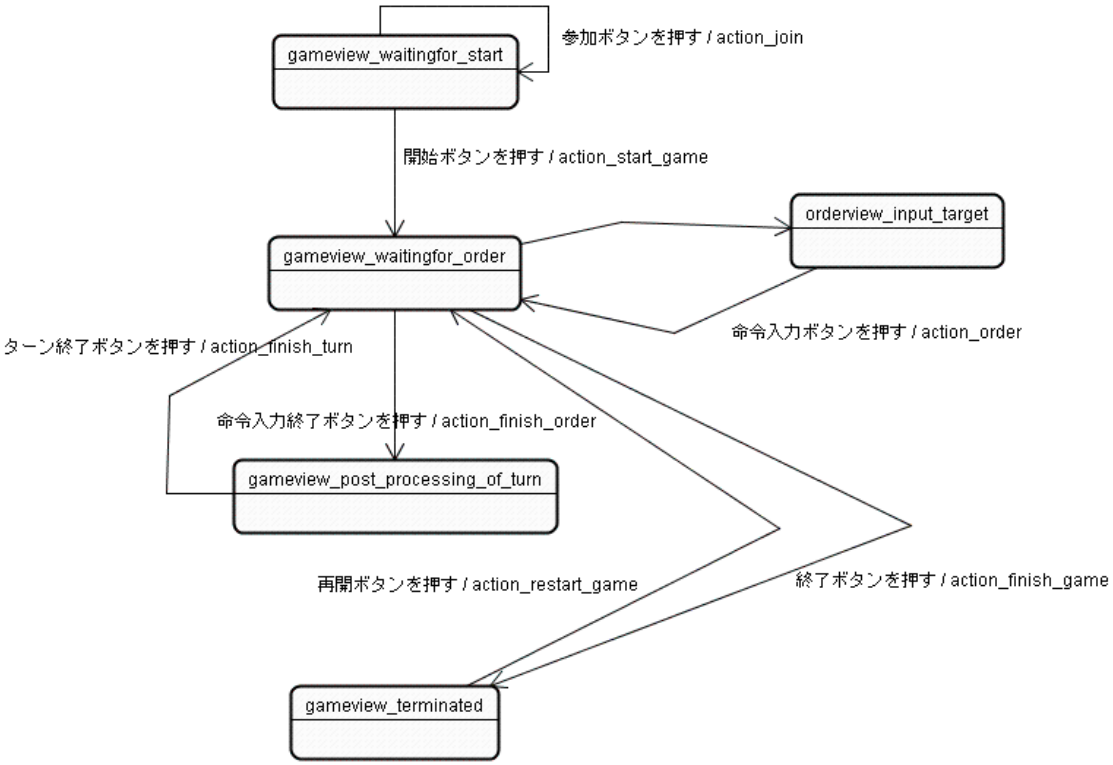
- gameview_
  - ゲームのメイン画面の一部であることを表す
- orderview_
  - 命令入力画面関連であることを表す
- mapview_
  - マップ表示画面関連であることを表す

4.6.3. アクション

これらのファイルは全て「action_」の形ではじまるファイル名になっている．例えば，ゲームを開始するというアクションのファイル名は，「action_start_game.php」となっている．

4.7. 画面遷移図

ディプロマシーの画面とアクションの関係を示すために画面遷移図を示す．



5. データベース仕様

5.1. データベースの ER 図

ディプロマシーのデータベースについて説明する．以下がディプロマシーのデータベースの ER 図である．

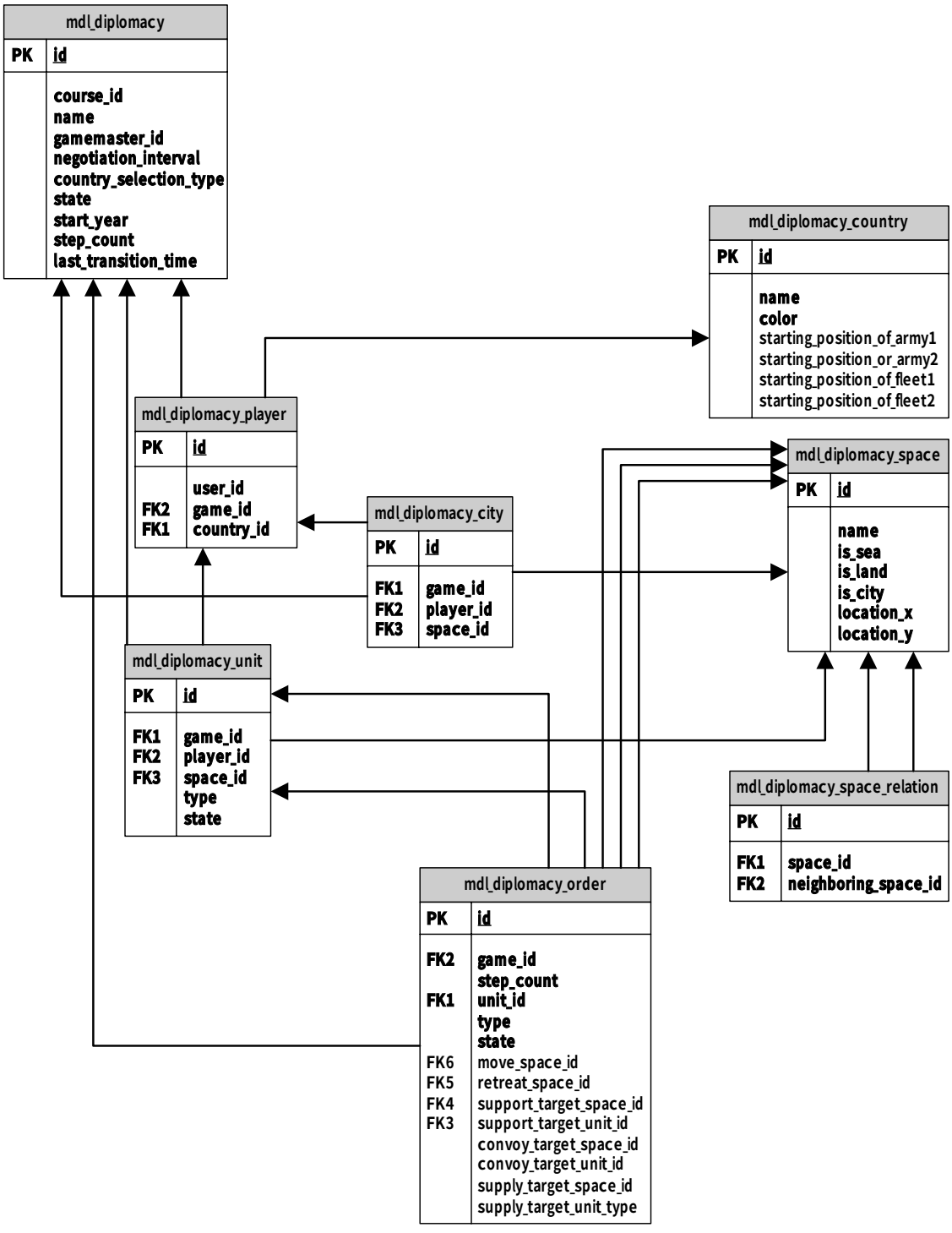


図 1

5.2. 各テーブルの説明

ディプロマシーで使用する各データベーステーブルの詳細を述べる.

5.2.1. **ディプロマシー**(mdl_diplomacy)

ディプロマシーテーブルは，ディプロマシー1 ゲーム分を表すテーブルである．以下のカラムを持つ．course_id はディプロマシーを追加したコースの ID を表す．また，gamemaster_id はゲームを作ったユーザーのユーザーID を表す.

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
course_id	int(16) unsigned		コース ID
Name	varchar(255)	'Unnamed Game'	名前
gamemaster_id	int(16) unsigned		ゲームマスターユーザーID
negotiation_interval	int(16) unsigned	5	交渉時間
country_selection_type	int(4) unsigned	1	国の選択方法(1=選択 2=ランダム)
State	int(4) unsigned		状態(1=開始前 2=命令入力待ち 3=補給撤退 4=終了)
start_year	int(16) unsigned	1900	最初の年
step_count	int(16) unsigned	0	ステップ数
last_transition_time	Datetime		状態変更時刻

表 1

以下にディプロマシーテーブルのデータ例を示す.

id	cource_id	name	game master_id	negotiation_interval	country_selection_type	state	start_year	step_count	last_transition_time
1	10	初心者部屋	2	5	1	4	1900	79	2005-01-13 17:01:23
2	10	初心者部屋	2	10	1	2	1900	2	2005-01-20 23:10:13
3	10	中級・ランダム限定	3	5	2	3	1900	9	2005-01-20 23:09:45
(後略)									

5.2.2. **国**(mdl_diplomacy_country)

国テーブルはプレイヤーが選択する国を表す．国にはそれぞれゲーム開始時に持っている軍隊が設定されている．その初期軍隊の位置を starting_position_of_army1,starting_position_of_army2,starting_position_of_fleet1, starting_position_of_fleet2 で設定している．ほとんどの国の初期軍隊は陸軍２つに海軍１つだが，ロシアの場合は陸軍 2 つに海軍 2 つである．

また，国テーブルは静的なテーブルである．ディプロマシーインストール時にデータを入力し，その後はデータを読み込まれるだけでデータを変更されることがない．

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
Name	varchar(255)	‘Unnamed Country’	名前
Color	varchar(255)	'Black'	色
starting_position_of_army1	int(8) unsigned		初期陸軍 1 の位置
starting_position_of_army2	int(8) unsigned		初期陸軍 2 の位置
starting_position_of_fleet1	int(8) unsigned		初期海軍 1 の位置
starting_position_of_fleet2	int(8) unsigned		初期海軍 2 の位置

表 2

以下に国テーブルのデータ例を示す．イギリスの場合は，初期軍隊が陸軍１つと海軍２つのため，start_position_of_army2 は NULL となる．その他のロシアを除いた国では，陸軍が２つ海軍が１つのため start_position_of_fleet2 が NULL となる．ロシアの場合は，陸軍２つ海軍２つのため，NULL は設定されない．

id	name	color	start_ position_of_ army1	start_ position_of_ army2	start_ position_of_ fleet1	start_ position_of_ fleet2
1	イギリス	Blue	1	NULL	4	6
2	ドイツ	Black	7	8	9	NULL
3	ロシア	Gray	13	14	15	16
4	フランス	Cyan	20	21	22	NULL
5	イタリア	Green	26	27	28	NULL
6	トルコ	Orange	32	33	34	NULL
7	オーストリア	Red	37	38	39	NULL

5.2.3. **マス**(mdl_diplomacy_space)

このテーブルは，「マス」を表す．各マスには「陸」・「海」・「都市」などの属性が付加されている．「海」とは，そのマスに海軍が移動できることを意味する属性である．「陸」とは，そのマスに陸軍が移動できることを意味する属性である．「都市」とは、そのマスが都市であることを意味する属性である．一つのマスが，各属性を重複して持つこともあり得る．

マップ上で軍隊を表示するための座標として location_x、location_y を持つ．

また，マステーブルは静的なテーブルである．ディプロマシーインストール時にデータ

を入力し，その後はデータを読み込まれるだけでデータを変更されることがない.

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
Name	varchar(255)	'Unnamed Space'	名前
is_sea	int(2) unsigned	1	属性(0=海ではない 1=海)
is_land	int(2) unsigned	1	属性(0=陸ではない 1=陸)
is_city	int(2) unsigned	1	属性(0=都市ではない 1=都市)
location_x	int(4) unsigned	1	マップ上の x 座標
location_y	int(16) unsigned	1	マップ上の y 座標

表 3

以下にマステーブルのデータ例を示す.

id	name	is_sea	is_land	is_city	location_x	location_y
1	ロンドン	1	1	1	60	140
2	ウェールズ	1	1	0	52	137
3	ヨークシャー	1	1	0	60	129
(後略)						

5.2.4. **マス関連**(mdl_diplomacy_space_relation)

このテーブルは，マスとマスの隣接関係を表す.

マス関連テーブルは静的なテーブルである．ディプロマシーインストール時にデータを入力し，その後はデータを読み込まれるだけでデータを変更されることがない.

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
space_id	int(16) unsigned		マスの ID
neighboring_space_id	varchar(255)		隣接するマスの ID

表 4

以下にマス関連テーブルのデータ例を示す．隣接関連テーブルでは，隣同士のマスが相互参照するようにしている．ここでは，id が 1 のレコードと id が 3 のレコードが相互参照している.

id	space_id	neighboring_space_id
1	1	2
2	1	3
3	2	1
(後略)		

5.2.5. **プレイヤー**(mdl_diplomacy_player)

ゲームのプレイヤーに関する情報を持つテーブルである．ユーザーID は Moodle のユーザーID を表す.

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
user_id	int(16) unsigned		ユーザーID
game_id	varchar(255)		ゲーム ID
country_id	int(16) unsigned		国 ID

表 5

以下にプレイヤーテーブルのデータ例を示す.

id	user_id	game_id	country_id
1	4	1	3
2	1	1	6
3	3	1	1
(後略)			

5.2.6. 軍隊(mdl_diplomacy_unit)

軍隊を表すテーブルである.

プレイヤーを辿っていけば, game_id を取得することができるが, 実装を楽にするために game_id というカラムを加えた. game_id は一度設定すれば, 変更する事はなく, また, プレイヤーID に一度設定された game_id は変更されることがないので, プレイヤーの game_id と整合性を取る必要がない.

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
game_id	int(16) unsigned		ゲーム ID
player_id	int(16)		プレイヤーID
space_id	int(16) unsigned		マス ID
Type	int(4) unsigned		種類(1=陸軍 2=海軍)
State	int(4) unsigned		状態(1=生存中 2=壊滅)

表 6

以下に軍隊テーブルのデータ例を示す.

id	game_id	player_id	space_id	type	state
10	2	3	1	1	1
11	2	3	4	2	1
12	2	3	48	2	2
13	2	9	7	1	1
(後略)					

5.2.7. **都市**(mdl_diplomacy_city)

都市を表すテーブルである．あるプレイヤーが都市を支配している場合，player_id にプレイヤーの ID が入る．

ゲーム毎に都市のデータがなければならない．そのために，ゲーム開始時に都市データを作成するようにする．

プレイヤーを辿っていけば，game_id を取得することができるが，実装を楽にするために game_id というカラムを加えた. game_id は一度設定すれば, 変更する事はなく, また, プレイヤーID に一度設定された game_id は変更されることがないので，プレイヤーの game_id と整合性を取る必要がない．

カラム名	データ型	デフォルト値	意味
Id	int(16) unsigned		ID
game_id	int(16) unsigned		コース ID
player_id	int(16) unsigned	NULL	プレイヤーID
space_id	int(16) unsigned		マス ID

表 7

id	game_id	player_id	space_id	type	state
10	2	3	1	1	1
11	2	3	4	2	1
12	2	3	48	2	2
13	2	9	7	1	1
(後略)					

5.2.8. **命令**(mdl_diplomacy_order)

各プレイヤーがターン毎に行う命令の情報を持つテーブルである．テーブル内の各レコードは，以下の情報を保持している．

- 命令の内容に関する情報(「どの軍隊に何をさせるか」)
- その命令の状態に関する情報(「その命令が既に実行されたかどうか」または「その命令が成功したかどうか」)

ターンが変わるのと同時に全ての軍隊に対して１つずつ命令が作成される．命令を出す場合は，その都度，データベースのレコードを更新するようにする．1 ターンに同じ軍隊に対する命令は必ず１つでなければならない．

カラム名	データ型	デフォルト値	意味

Id	int(16) unsigned		ID
game_id	int(16) unsigned		ゲーム ID
step_count	int(16) unsigned		ステップ数
unit_id	int(16) unsigned		軍隊 ID
Type	int(4) unsigned	1	種類(1=維持 2=移動 3=サポート 4=輸送 5=撤退 6=補給)
State	int(4) unsigned	1	状態(1=実行前 2=成功 3=失敗)
move_space_id	int(16) unsigned	NULL	移動先マス ID
retreat_space_id	int(16) unsigned	NULL	撤退先マス ID
support_target_space_id	int(16) unsigned	NULL	サポート先マス ID
support_target_unit_id	int(16) unsigned	NULL	サポート先軍隊 ID

表 8

以下に命令テーブルのデータ例を示す．Id が 1 の場合のように維持命令の場合は, move_space_id, retreat_id などの命令のターゲットを表す項目が NULL になる．移動命令の場合は, move_space_id にターゲットのマス ID が入り, その他の項目は NULL になる．同様にサポート命令の場合は，support_target_space_id,support_target_unit_id にターゲットの ID が入る．

id	game_id	step_count	unit_id	type	state	move_space_id	retreat_id	support_target_space_id	support_target_unit_id
1	2	1	10	1	2	NULL	NULL	NULL	NULL
2	2	1	11	2	3	6	NULL	NULL	NULL
3	2	1	12	3	3	NULL	NULL	4	5
(後略)									

ディプロマシーマニュアル

目次

1. このドキュメントについて

このドキュメントではディプロマシーの遊び方を説明する．対象読者は，ディプロマシーを利用しようとするユーザーである．Moodle の操作ができることを前提とする．

ディプロマシーの基本的なルールは割愛する．代わりに，ディプロマシーのルールを解説した Web ページを以下にいくつか紹介する．

- ディプロマシー(Diplomacy) : http://pohwa.adam.ne.jp/31/game/Diplomacy.htm
- ディプロマシー第四版 日本語訳 : http://coffee.vis.ne.jp/diplo/rule/

2. ゲームの進行

状態遷移図

説明

3. ユーザー

ディプロマシーのユーザーは以下の 2 種類である．

- プレイヤー
- ゲームマスター

次にそれぞれの詳細を述べる．

3.1. プレイヤー

プレイヤーとは，ゲームに参加し，ゲームを遊ぶユーザーである．プレイヤーの役割は以下の通りである．

- ゲームに参加する
- 命令を出す

3.2. ゲームマスター

ゲームマスターとは，ゲームを管理するユーザーである．ゲームマスターの役割は以下の 4 種類ある．

- ゲームの作成
- ゲームの開始
- ゲームの終了
- 命令フェーズの終了
- 補給・撤退フェーズの終了

ゲームマスターはプレイヤーが行ってもよい．

4. 操作方法

この章ではディプロマシーの操作方法を説明する.

4.1. ゲームの新規作成(ゲームマスター)

ここではゲームを新しく始めるための新規作成の仕方について説明する．この操作を行うとユーザーは作成したゲームのゲームマスターである.

ゲームを追加したいセクションで「活動の追加」から「ディプロマシー」を選択する

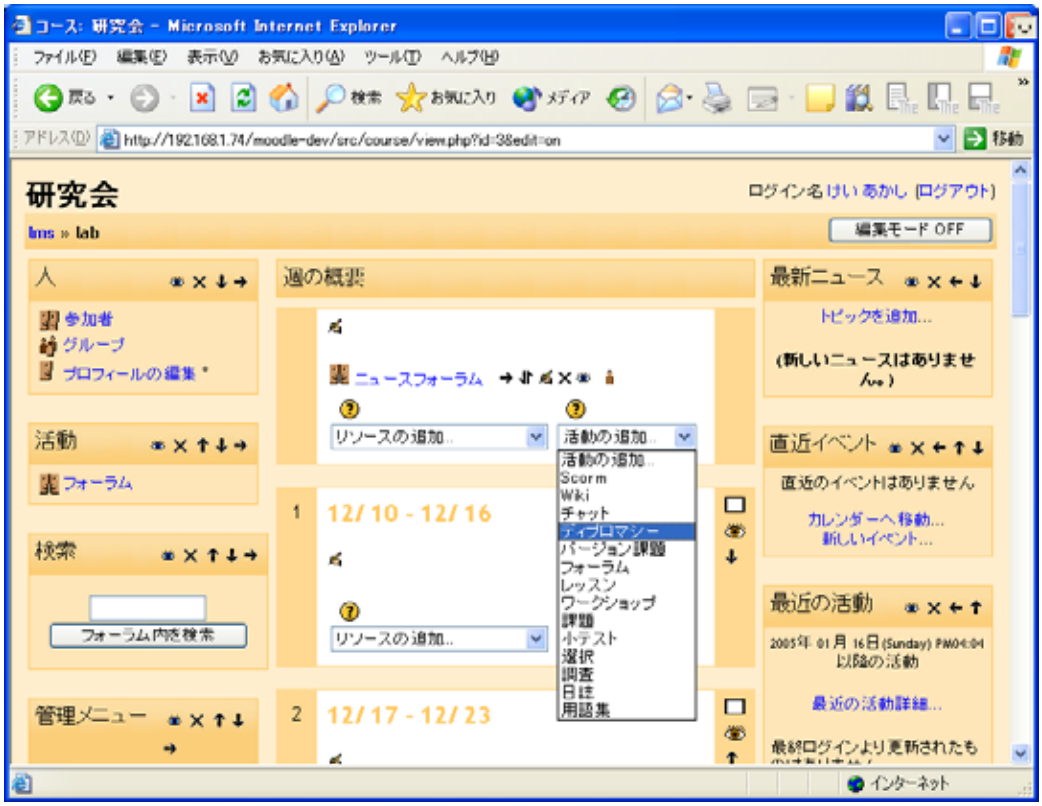


図 1

・ゲーム作成画面が表示される．「名称」「交渉時間」「国の決定方法」を設定し，「変更を保存」ボタンを押す．

図 2

・設定画面が表示されると，ゲームの新規作成が完了する．

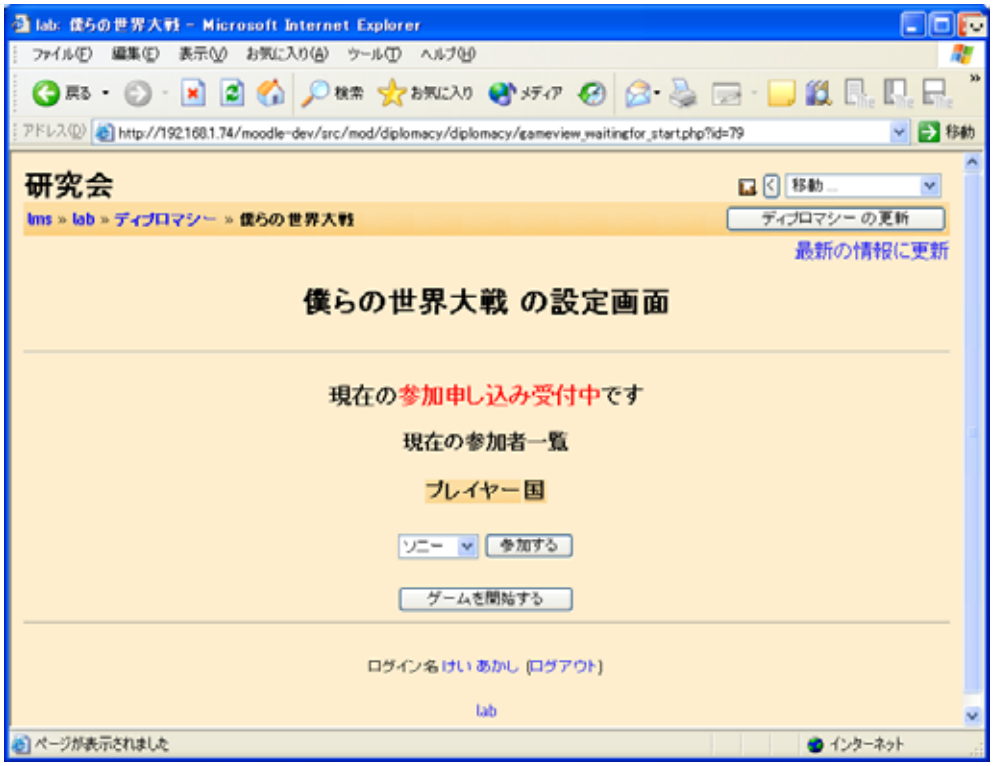
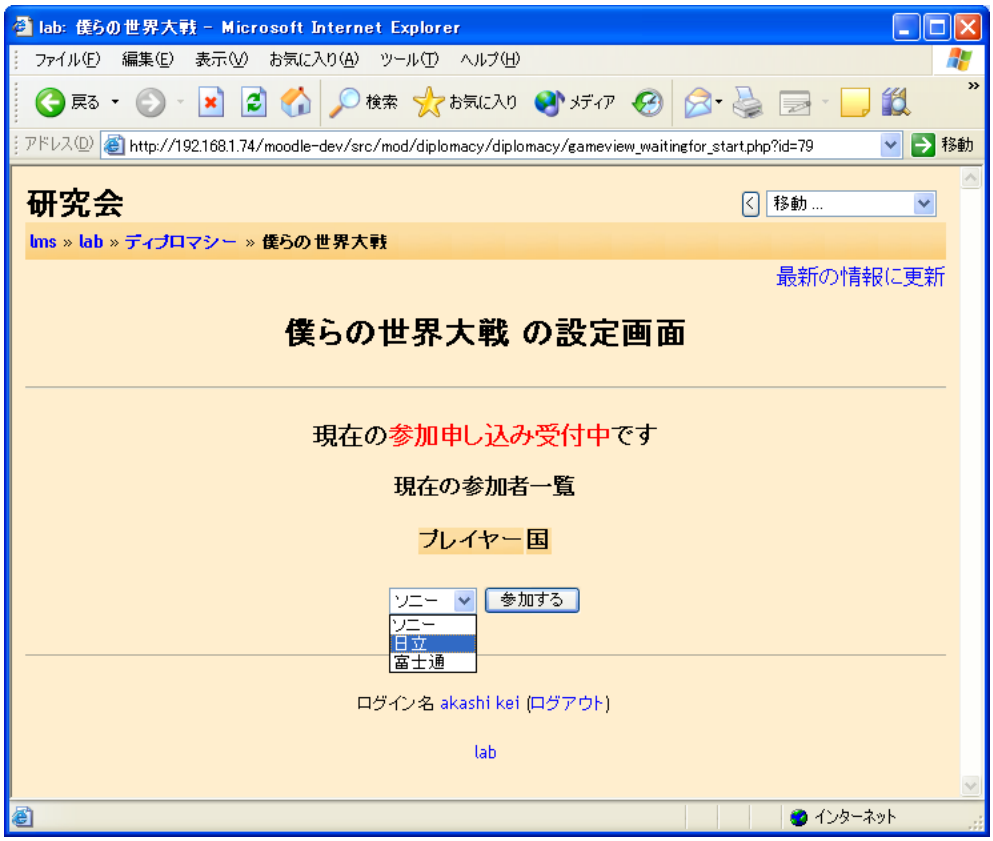


図 3

4.2. ゲームに参加する

ゲームへの参加方法について説明する．ゲームで遊ぶためにはユーザーはゲームに参加しなえければならない．ゲームに参加することでユーザーはそのゲームのプレイヤーとなる．

国を選択し，「参加する」ボタンを押す

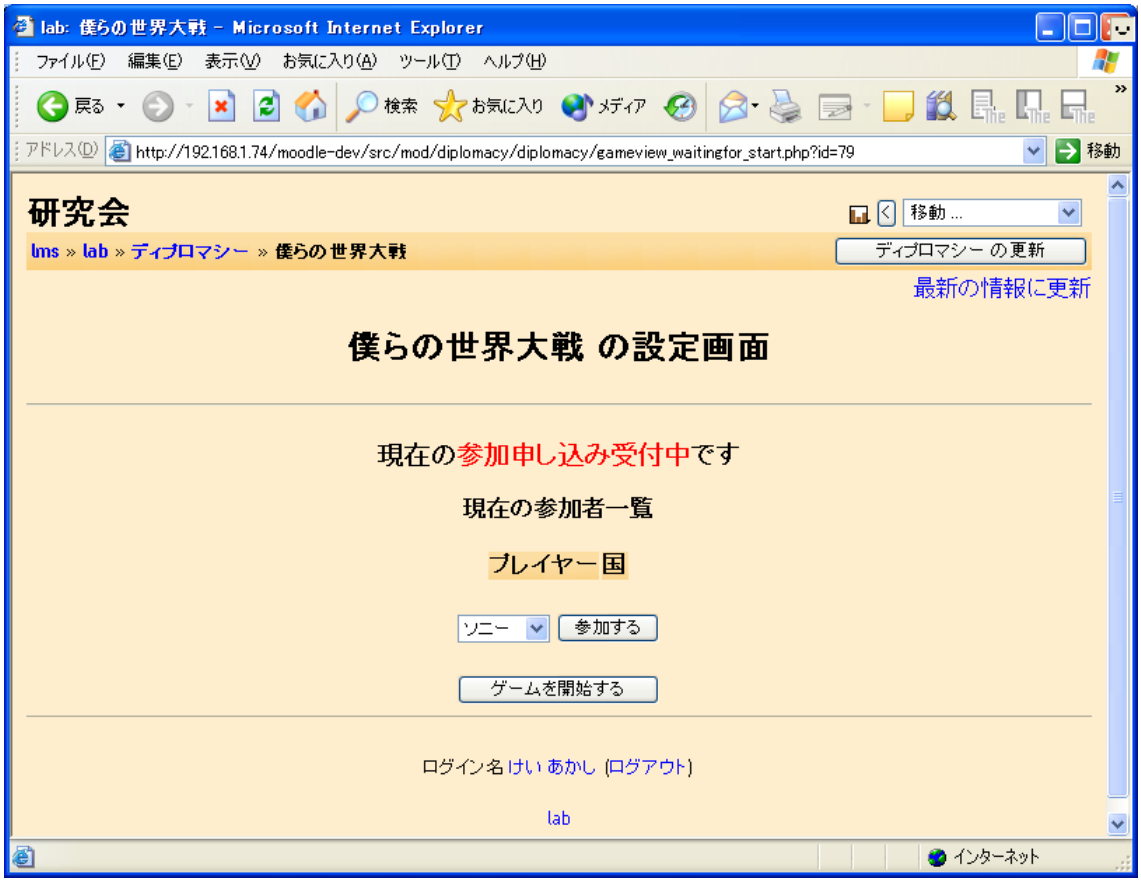


・「現在の参加者一覧」に名前と選択した国が表示されるとゲームに参加するが完了する．

4.3. ゲームを開始する(ゲームマスター)

ゲームの開始方法について説明する．この操作はゲームマスターのみが実行できる．

・「ゲームを開始する」ボタンを押す．



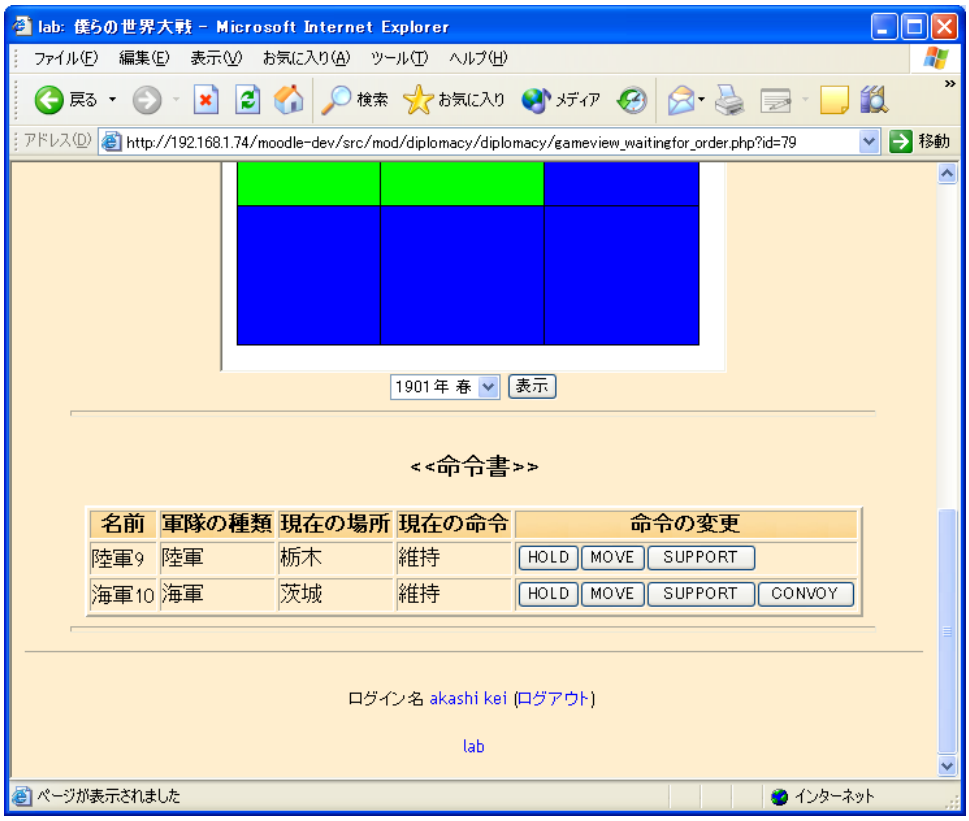
戦況画面が表示されるとゲームが開始される.

4.4. フェーズごとの操作

4.4.1. 命令フェーズ

4.5. 命令する

命令は戦況画面下にある「命令書」から出すことができる.



命令には「維持(HOLD)」「移動(MOVE)」「サポート(SUPPORT)」「輸送(CONVOY)」の 4 種類が存在する.

各命令の内容を以下に載せる.

4.5.1.1. 維持(HOLD)

軍隊を待機させる．誤ったコマンドを指示した場合は，このコマンドでキャンセルすることができる.

4.5.1.2. 移動(MOVE)

軍隊を移動させる．このコマンドを選択すると，移動先の選択画面に遷移する(図 4).

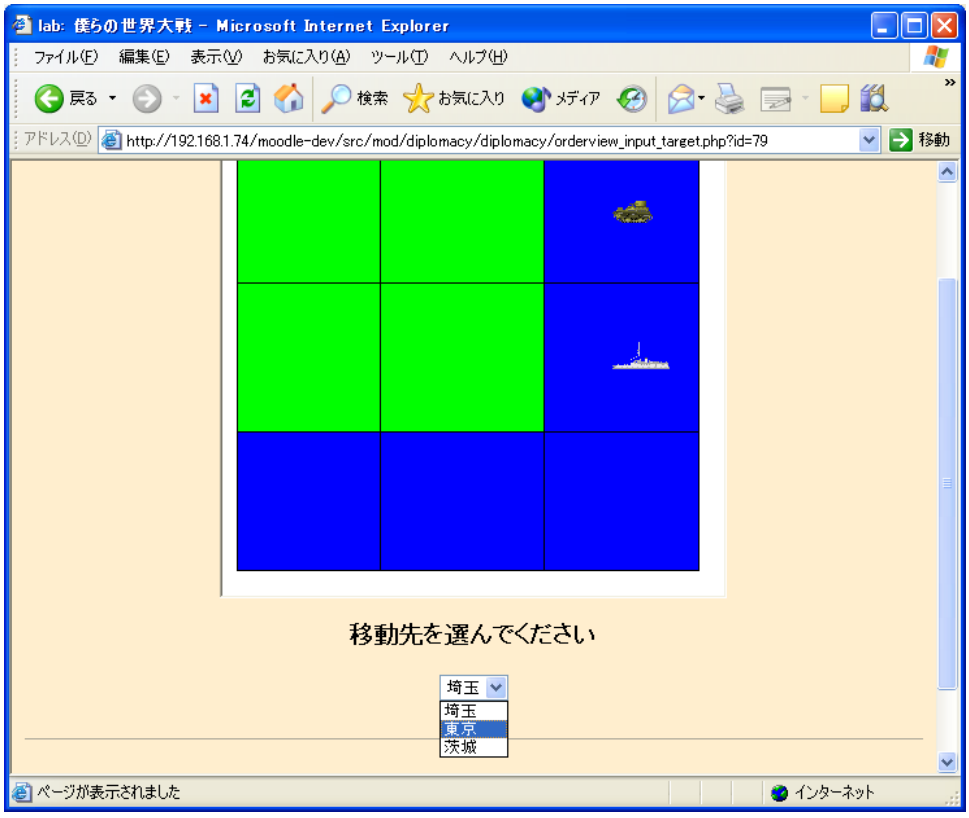


図 4

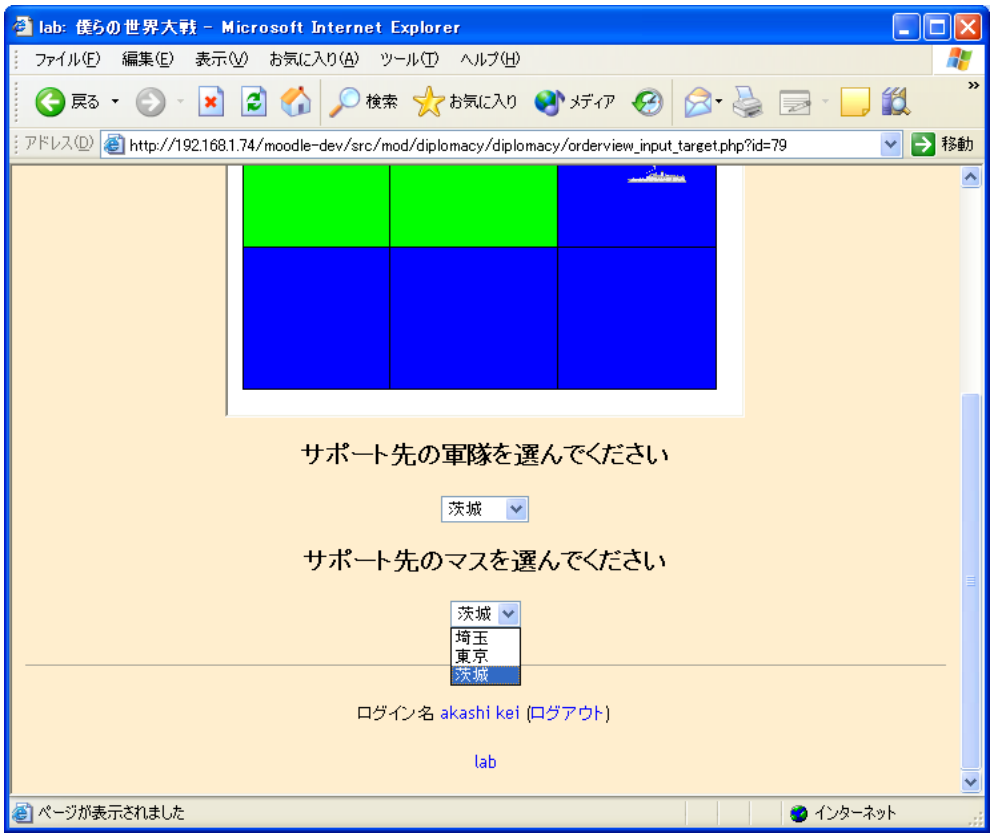
移動先を設定すると，戦況画面に戻る．以下のように命令書が更新される．

<<命令書>>

名前	軍隊の種類	現在の場所	現在の命令	命令の変更
陸軍9	陸軍	栃木	移動 TO 東京	HOLD MOVE SUPPORT
海軍10	海軍	茨城	維持	HOLD MOVE SUPPORT CONVOY

4.5.1.3. サポート(SUPPORT)

他の軍隊をサポートする．未稿.



以下のように命令書が更新される．

<<命令書>>

名前	軍隊の種類	現在の場所	現在の命令	命令の変更
陸軍9	陸軍	栃木	サポート 海軍10 -> 茨城	HOLD MOVE SUPPORT
海軍10	海軍	茨城	維持	HOLD MOVE SUPPORT CONVOY

4.5.1.4. 輸送

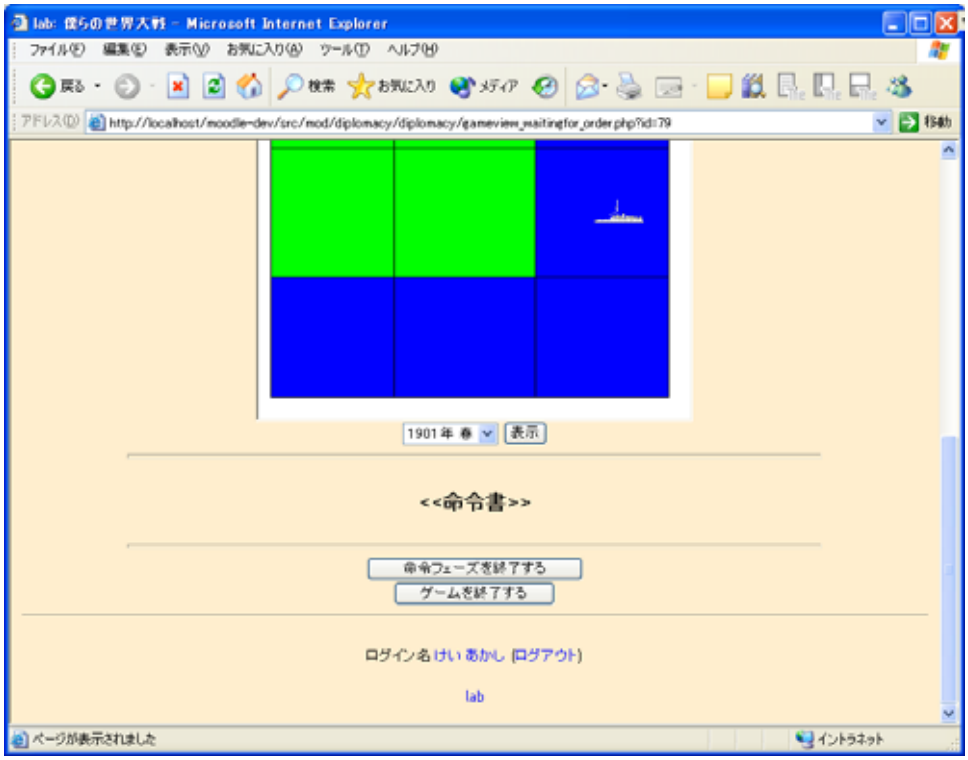
輸送は"CONVOY" コマンドによって行う．このコマンドは海軍しか実行できない．以下未稿．

以下のように命令書が更新される．

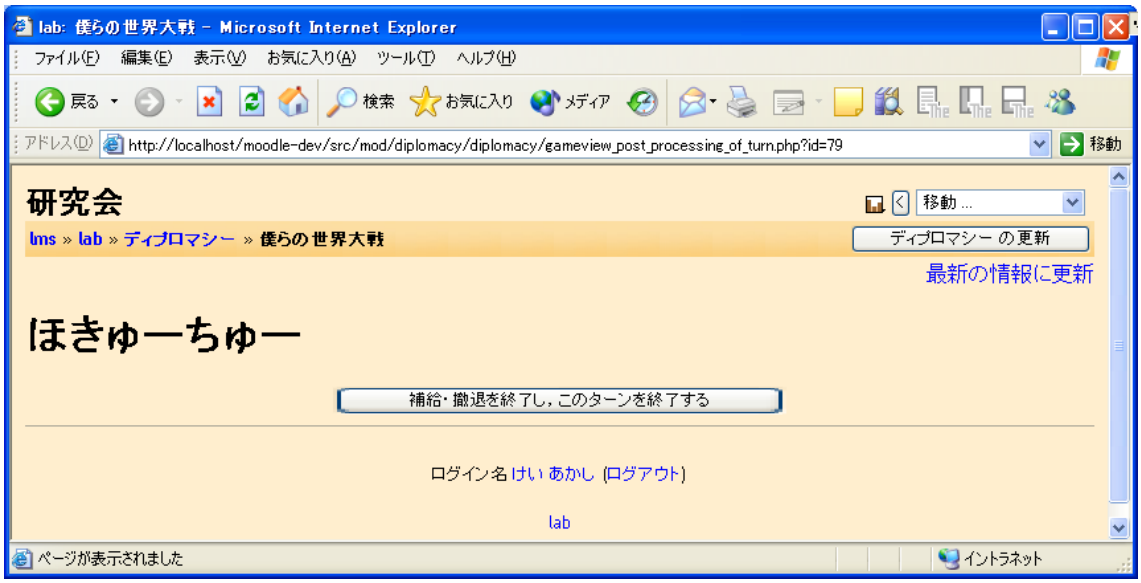
<<命令書>>

名前	軍隊の種類	現在の場所	現在の命令	命令の変更
陸軍9	陸軍	栃木	サポート 海軍10 -> 茨城	HOLD MOVE SUPPORT
海軍10	海軍	茨城	輸送	HOLD MOVE SUPPORT CONVOY

4.5.1.5. 命令フェーズを終了する(ゲームマスター)



補給・撤退画面になる



4.5.2. 撤退・補給フェーズ

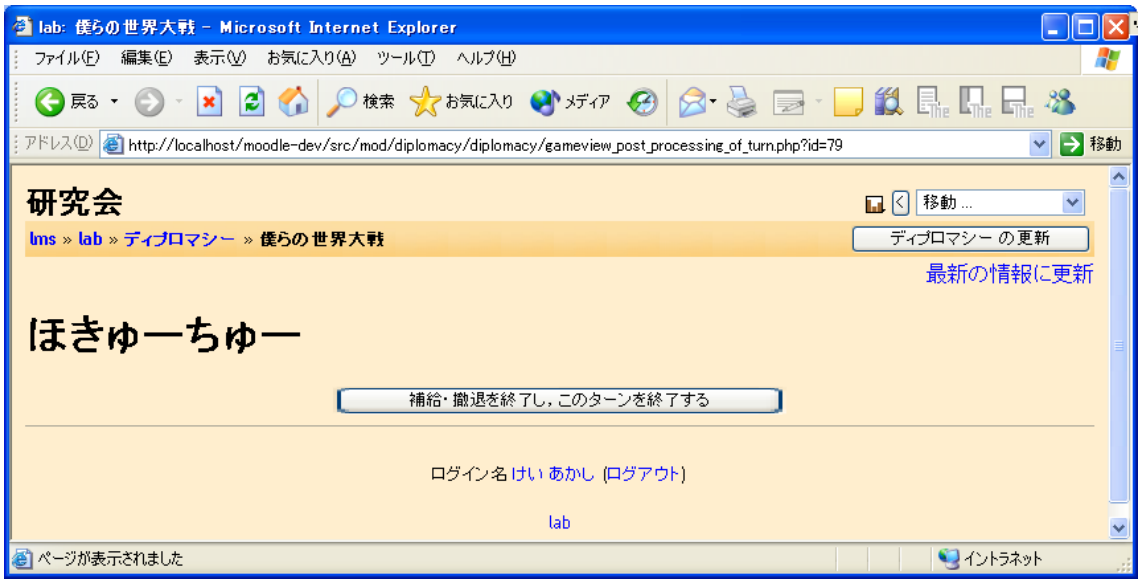
4.5.2.1. 撤退する

4.5.2.2. 補給する

4.5.2.3. 補給・撤退フェーズを終了する(ゲームマスター)

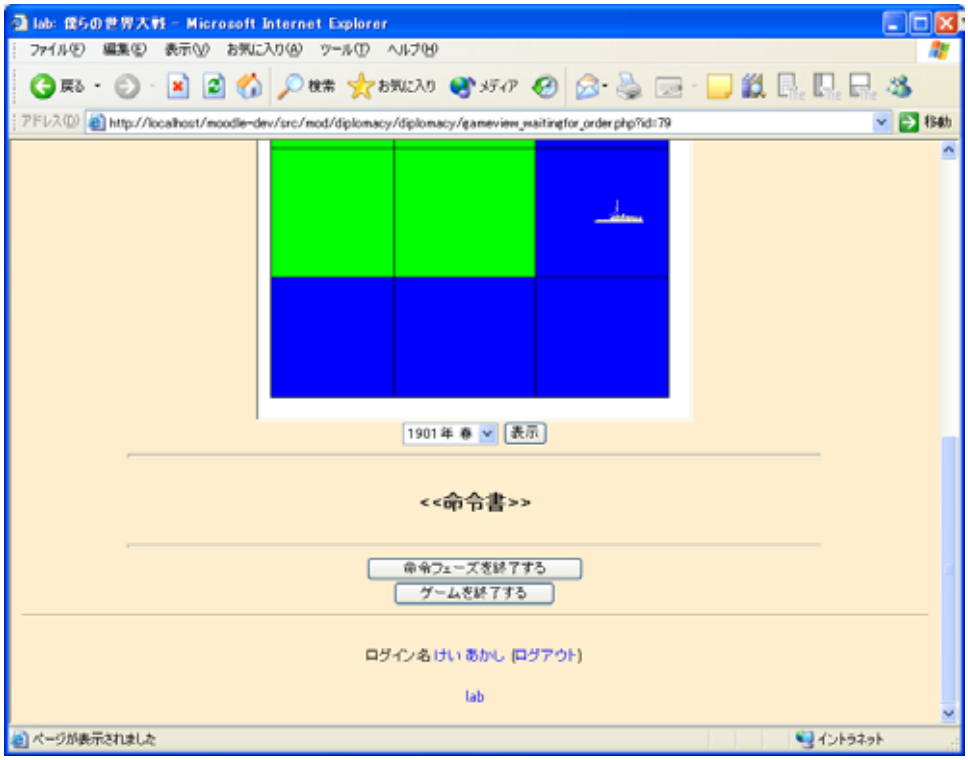
補給・撤退フェーズの終了方法

終了するとターンが１つ進む.

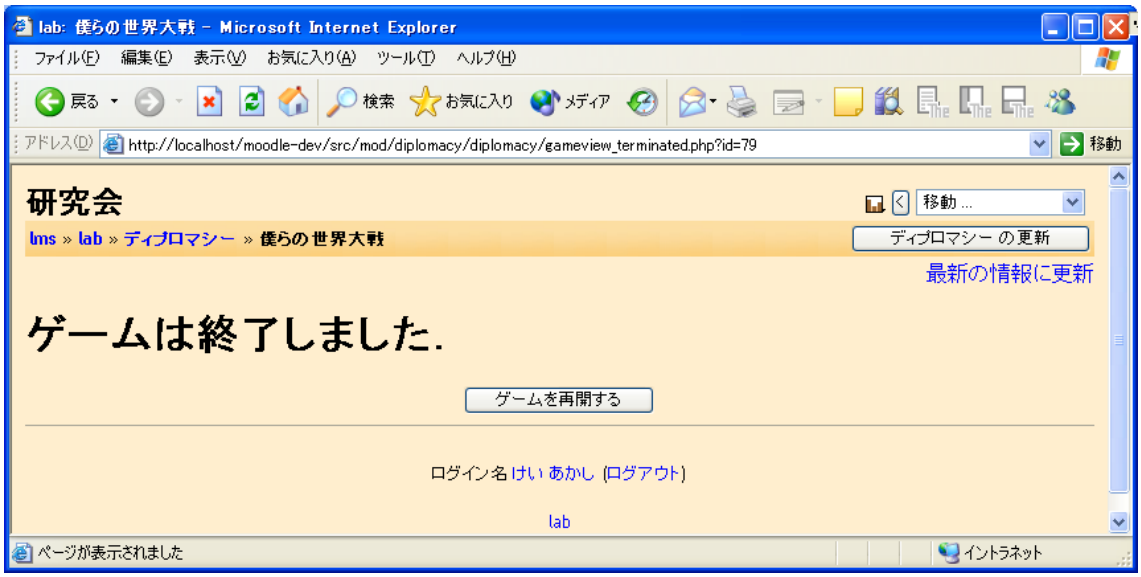



lab: 僕らの世界大戦 - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り メディア
アドレス(D) http://localhost/moodle-dev/src/mod/diplomacy/diplomacy/gameview_waitingfor_order.php?id=79
移動
研究会
移動 ...
lms » lab » ディプロマシー » 僕らの世界大戦
ディプロマシー の更新
最新の情報に更新
僕らの世界大戦
1900秋
次のターンまであと12分5秒
<<戦況>>
国名 プレイヤー 支配都市 陸軍 海軍
日立 kei akashi 2 1 1
ページが表示されました
イントラネット

4.6. ゲームを終了する(ゲームマスター)



ゲーム終了画面．ゲームを再開したい場合は，「ゲームを再開する」ボタンを押す．



第三部

（個人レポート）

1. 個人レポート(明石　敬)

本章では，今期のディプロマシーfor Moodle の開発を通じて筆者が学んだこと、感じたことについて述べる。

1.1. プロジェクトの運営

ディプロマシーfor Moodle プロジェクトは以下のメンバーで開発を行った。

- 明石(PM)
- 森岡
- 鬼頭
- 松澤(レビュアー)

筆者は今期プロジェクトの PM の役割を負った。PM の役割として以下のものが挙げられる。

- 進捗管理
- 仕事のわりふり

この２つの役割について、今期筆者が感じたこと、学んだ事を述べる。

1.1.1. 進捗管理

進捗管理とはプロジェクトの進行具合を管理し、次すべきこと、そのスケジュールを立てることである。PM が進捗を管理する事でプロジェクトメンバーは安心して作業を進めることができる。

筆者は常にプロジェクトの進行具合を把握していたつもりでいた。「大体、プロジェクトが何割くらい進んでいて、あとこういうことをしなければならないな。」ということを常に考えていた。

しかし、そのプロジェクトの進行の把握が甘いことがプロジェクトのスケジュールをたて、実際に作業をすることでわかった。筆者が立てた最初のプロトタイプ作成のスケジュールは以下の通りである。

11 月 4 週	設計
12 月 1 週	実装１
12 月 2 週	実装２
12 月 16 日	プロトタイプ完成
1 月はじめ	完成、テスト、ドキュメント
1 月下旬	ドキュメント完成

表 1

1 月はじめまでの実際のプロジェクトの進行は以下の通りである。プロジェクトがスケジュールどおりに進行していないことがわかる。1 月のはじめにはすでに完成しているはずだが、実装２に手間取ってしまい、1 月はじめにもプロトタイプを終わらせることができなか

った。

11 月 4 週	設計
12 月 1 週	実装 1
12 月 2 週	実装 2
12 月 16 日	実装 2
1 月はじめ	実装 2

表 2

1 月はじめの時点でプロトタイプが終わっていない上にドキュメントも作成しなければならなかったため、スケジュールを立て直す必要があった。以下が私の 1 月 4 日に立てたスケジュールである。

1 月 4 日~11 日	実装 2　プロトタイプ完成
12 日～	テスト
13 日～20 日	作りこみ
21 日～	ドキュメンテーション
1 月 27 日	ドキュメント完成

表 3

ここでレビュアーである松澤氏からスケジューリングのレビューを受けた。松澤氏によるとこのスケジュール通りにこなせるわけがない、ということだった。松澤氏が提案したスケジュールは以下の通りである。1月4日の時点で一時的に開発を打ち切ることになった。

1 月 4 日~11 日	ドキュメンテーション
12 日～20 日	ドキュメント完成。実装 2
21 日～27 日	実装 2　完成

表 4

筆者ははじめ開発を打ち切ることに不満があった。筆者としては筆者が考えた表 3 でうまくいくと考えていたからである。さすがにきっちりとスケジュール通りにうまくいくとは思ってはいなかったが、それなりのスケジュールに近い形でプロジェクトを進められると考えていた。

しかし、松澤氏の提案したスケジュールでいくことになった。なぜならば、ドキュメンテーションを先にすることで現在のプロジェクトの進行状況を把握出来るからである。また、この時点まで開発に参加していなかった鬼頭君と一緒にドキュメンテーションをすることで、最後の実装で鬼頭君が十分な戦力になることが見込まれていたからである。確かに、筆者が考えたスケジュールでは、鬼頭君を十分生かしきれなかっただろう、と思う。

最終的なプロジェクトの進行は以下の通りである。松澤氏が提案したスケジュール通りに進行した事がわかる。

1 月 4 日~11 日	ドキュメンテーション
12 日～20 日	ドキュメント完成。実装２
21 日～27 日	実装２　完成

表 5

これまで述べてきたように筆者にはスケジューリング能力が欠けていた。その原因としてうまく進捗状況が把握できていなかったことが挙げられる。プロジェクトを運営する上で常に進捗状況をうまく把握することが非常に重要なことがわかった。

1.1.2. 仕事のわりふり

今期、筆者が最も苦労したのは、プロジェクトメンバーである森岡君、鬼頭君の 2 人との共同作業であった。今まで行ってきた開発では、PM としてメンバーに指示を出す、ということをして来なかったため、うまく指示を出すことができなかった。結果、逆に 2 人を混乱させてしまう、という場面がいくつかあったように感じられる。

また、プロジェクトメンバーの 2 人の能力を把握しきれてなかいため、無理な指示を出してしまったり、簡単すぎる仕事を頼んでしまっていたりした。これによって筆者が一番えぐい部分を 1 人でやっつけてしまうことがあった。

筆者が 1 人で作業を行ったほうが、プロジェクトメンバーに仕事を振っていく方法よりはやく作業が進む。しかし、それでは、プロジェクトメンバーはいつまでたっても開発に参加することができない。

本来、プロジェクトメンバーの能力を把握した上で的確な仕事を振れば、作業の進行は早いはずである。

1.1.3. ペアプログラミング

今期の最後の開発では主にペアプログラミングを行った。筆者が PC に向かってプログラミングをし、横にプロジェクトメンバーが HCP チャートを見ながらちゃんとプログラムが書けているかを見る。フレームワークや PHP の文法について多少知っている筆者がプログラムを書いたほうがスムーズにプログラミングできる。また、筆者がプログラムを書いていくところをメンバーが見て筆者に質問することで Moodle フレームワークや PHP の文法の知識をメンバーが学ぶ事ができる。

ペアプログラミングの利点は、普段、やりがちな変数のコピペミスや簡単なスペルミスをプロジェクトメンバーがいち早く発見できることである。また、難しいアルゴリズムでは 2 人で議論をしながらプログラムを書く事ができる。

このようにペアプログラミングを行う事でプロジェクトメンバーとうまく共同作業が行えた。残念ながら 1 人で作業をしたときとペアプログラミングをしたときの作業効率をはかるデータが残っていない。しかし、確実に 1 人で作業を行った場合よりもはやく作業が

進んだと感じられる。

　今回の場合に限らず、今後、研究会の他のプロジェクトで開発を行う上でもペアプログラミングは有効である。来学期の研究会では新規履修者と継続履修者でペアプログラミングを行うのもよいかもしれない。

2. 森岡亮一個人レポート

2.1. 今学期の主な作業

・ ディプロマシー分析
  - ディプロマシールール説明作成
・ ディプロマシーシステム分析
  - ユースケース図（ミーティング時に議論しながら作成）
  - クラス図（ミーティング時に議論しながら作成）
  - インスタンス図
・ 開発
  - コマ表示
  - ルール判定ＨＣＰ作成
・ ドキュメント
  - ディプロマシールール説明書作成記
  - システム分析報告書

2.2. 今学期学んだこと

・ 人にものを伝える方法

　ディプロマシーのルール説明を作る過程で多くを学んだ。これに関しての感想はディプロマシールール説明記で詳細を書いている。それ以外にもシステム分析報告書の見せ方など、人のやり方を見ることで、いかにしてプレゼンテーションで見せるかという方法を学ぶことができた。

・ 分析について

　このプロジェクトで議論を通じて分析することで、分析に対する理解を深め、よりよい分析ができるようになったと思う。

・ 今まで学んだことのないプログラミング言語を使って開発することについて

　今まで使ったことのないプログラミング言語で開発することの難しさを知った。このプロジェクトでは開発時間のうち半分くらいはＰＨＰを理解することに費やされてしまった。このことで、ＰＨＰに対する知識を学んだとともに、今まで学んだことのないプログラミング言語をいかにして学び、使うかということも学習することができた。

2.3. 感想

　研究会の履修も三期目になり、プロジェクトを進める上でのかってというものが分かってきた。学習することの種類も増えたとともに、自分自身の学習の成果はともかく、ある程度人を指示できるようになることが期待されているのではないかと思う。

　今回学んだことの中では、人を説明する方法がもっとも重要度と応用性が高いと思う。

単に人に話す方法を学んだというのにとどまらず、その根底には何にでも応用可能な思考法があると考える。実際に何を考えるにも、アウトラインから考えるようになった。

　来学期から自分がどのように研究会での位置を占めるようになるかは分からないが、学ぶべきことを学び、さらに多くのことをできるようにしていきたいと思う。

研究会 2 個人レポート

環境情報学部 3 年
鬼頭 孝行
平成 17 年 1 月 27 日

活動の推移を順番に列挙すると、以下の通りになる。

1. LMS 連句プラグインの考案・設計
2. ディプロマシープロジェクトに途中参加
3. ディプロマシーの設計
4. ドキュメンテーション

1. LMS 連句プラグインの立案・設計

1.1. プロジェクトの推移

研究会の「人を幸せにするコミュニティサイトは何か」という課題を通じて、「連句プラグイン」というシステムを考案した。連句とは、江戸時代に行われた一つの文芸である。このプラグインの考案には、「美しい日本の掲示板—インターネット掲示板の文化論」(鈴木淳史著、洋泉社)にあった「2 ちゃんねると連句には共通した精神がある」という記述がヒントになった。

このプロジェクトの進捗は、システムのユースケース記述を仕上げたところで止まっている。研究会の議論で挙げられた、アーキテクチャが掲示板を意識しすぎているという指摘に対する答えが自分の中で得られなかったことと、このプロジェクトが LMS プラグインに慣れるという名目の上で行われていたためである。プロジェクト一時凍結後、同じく LMS プラグイン開発であるディプロマシープロジェクトに途中参加することになった。

1.2. 感想

「面白そう」という意見を複数いただいたのは、とても嬉しかった。個人的には、このプラグインがどのような面白さを持つかをかなり詳細にイメージしていたために、途中でプロジェクト中止となったのは残念である。機会があれば、ぜひとも完成させたいと思う。

2. ディプロマシープロジェクト

2.1. プロジェクトの推移

このプロジェクトで私がはじめに行ったことは、開発環境の構築とソース読みである。プロジェクトに関わる仕事は、ドキュメンテーションと設計の 2 つである。当初は開発を先に終了させて、残りの時間をドキュメンテーションに充てる予定だったが、松澤さんの判断により、順番を逆にした。理由は以下の 2 点である。

- ドキュメンテーションを先に終わらせることにより、余裕を持って開発できるようにするため
- 資料をまとめることにより、私がスムーズにプロジェクトに関われるようにするため

ドキュメンテーションが終了した後、未実装の機能の設計(軍隊の移動・戦争処理)をチームメンバーと共に HCP チャートを使って行った。

2.2. 感想

2.2.1. ドキュメンテーション

最終的に 100 ページ程度の資料が完成した。1 つのソフトウェアのドキュメントとして、これだけのページの資料を作成したのは初めての経験だった。私が担当したのはそのうちの「ディプロマシーマニュアル」(実際にディプロマシーを遊ぶ人を対象にし、操作方法を説明した資料)で、枚数は 17 ページ程度である。

今回のドキュメンテーションでは、Word の使い方をかなり学ぶことができた。例えば見出しや箇条書きの設定方法や、目次の表示方法などである。過去に後輩から「情報技術基礎で Word について習った」という声を聞いたときに、なぜわざわざ Word なんかをやる必要があるのかと疑問に思っていたが、使いようによっては Word でも構造化された文書を書くことが可能なのである。

2.2.2. 設計

HCP チャートを本番で利用したことは、実は今回のプロジェクトが初めてである。松澤さんからリアルタイムにアドバイスを受けながらのチャート作成は、本当にためになった。HCP チャートの使い方は、「オブジェクト指向プログラミング」や「情報教育論」などを通じて、**頭では分かっていた**のだが、今回の実戦を通じて**腑に落ちた**というのが率直な感想である。教育の場では、このような実戦がなかなか経験させることができないことが辛いところだ。