演習科目におけるモデリング能力育成の取り組み

ー情報工学特別演習ⅠaⅡa（ネットワークプログラミング）においてー

宮崎大学　工学部教育研究支援技術センター
相川　勝

はじめに

宮崎大学工学部情報システム工学科は，日本技術者教育認定機構（JABEE）の認定を目指した教育プログラムを実施するために，平成 16 年度に学科内に「情報システム専修コース」と「情報システム応用コース」の 2 つのコースを設置した．情報システム専修コースは，平成 17 年度に JABEE の認定を受けた教育プログラムである．コース設置に伴い，3 年次に開講されていた情報工学特別演習Ⅰ（前期），Ⅱ（後期）がコースの特色に合わせて，専修コースの科目「情報工学特別演習Ⅰa(前期)，Ⅱa（後期)」と応用コースの科目「情報工学特別演習Ⅰb（前期），Ⅱb（後期)」に再編され，今まで 7 週×2 グループで担当していたテーマである「ネットワークプログラミング」が専修コースのテーマとなり，14 週×1 グループに拡充された．

演習授業のコマ数が倍増したことにより演習内容をより充実させ，科目の教育目標の一つである「モデリング能力の育成」を重視した演習を目指すために，今まで時間の制約によって教えることができなかった技術（パケット観察，マルチスレッド等）や一般的な開発プロセス（分析，設計，実装，テスト）を導入してシステム構築を行う課題演習（グループ演習）を取り入れた演習内容へ拡充を行った．昨年度（平成 20 年度)より新科目の演習が開始されたので改良した演習教材の内容および実施状況等について報告する．

キーワード：ネットワークプログラミング マルチスレッド 開発プロセス

1. 以前までの演習内容

平成 19 年度までは,情報工学特別演習Ⅰ（前期）Ⅱ（後期）として学期に 7 週（14 コマ）×2 グループによる演習であった．7 週の演習の内容は，第 1 回目がネットワークプログラミング（クライアント編）：ネットワーク基礎（IP アドレス，ポート番号，ソケット等），telnet を用いた各種サーバ（SMTP サーバ，POP サーバ等）との通信，TCP クライアントプログラム（echo クライアント），第 2 回目がネットワークプログラミング（サーバ編）：TCP サーバプログラム（echo サーバ），第 3 回目がネットワークプログラミング（複数クライアント対応型サーバ）：select による入出力の多重化, select を用いた TCP 並行型サーバ(echo サーバ)であり，座学によって知識を学び，そして実際にプログラミングを行うことで理解力を高め，それぞれの回で演習課題をレポートとしてまとめることによって知識および技術の定着を図ってきた．

第 1 回から第 3 回までに習得したネットワークプログラミングの知識および技術を発展的に応用できる力を養うために，第 4 回から第 6 回の 3 週にかけて 3 名の小グループに分けて課題演習（グループ演習）を行っていた．課題演習の題材は「マルチサーバ型チャットシステムの設計と実装」である．

マルチサーバ型チャットシステムとは，ユーザが使用するクライアント・アプリケーションと通信をするサーバが複数存在し，そのサーバが統合された広域なチャットシステムである．この題材をもとに，プロトコル設計（手続き，コマンド）を行い，実装するために必要となるデータ構造およびプログラム構造を整理し，実装（プログラミング）を行ってシステムを完成させる．システムが有する機能は，あらかじめ定められた必須機能と各グループで考えたオリジナルな機能を設ける．そして最終回（第 7 回）に，各グループが作成したシステムのプレゼンテーションを行い自己表現能力の向上を図るものであった．

2. 新たに追加した教材

演習のコマ数が 2 倍になったことから，より充実した演習内容にするために新たな教材を追加することにした．追加した教材は，「tcpdump を用いた通信パケットの観察」，「プロトコル設計」，「マルチスレッド」，「排他制御」の 4 つである．

2.1 tcpdump を用いた通信パケット観察

既存のサーバ／クライアントおよび自作したクライアントとサーバ間においてアプリケーション層のデータがどのようにパケット化され，どのようなタイミングでパケット通信が行われているのかを tcpdump を用いて観察させ，理解させることを目的とする．

■ tcpdump によるパケット・キャプチャの手順

① GNOME 端末を 2 つ開く
② 「GNOME 端末 1」で tcpdump を起動する
③ 「GNOME 端末 2」で telnet を用いてサーバに接続する（SMTP サーバ：ポート 25）
④ 「GNOME 端末 2」上で，SMTP プロトコルに従ってコマンドを送信する
⑤ 送信した際，通信パケットのキャプチャ情報が「GNOME 端末 1」にリアルタイムに表示される
⑥ ここで次の情報を観察する
- C→S の通信か，S→C の通信か？（IP アドレス，ポート番号で判断）
- 制御フラグは何か？（SYN, PSH, FIN, RST）
- 送信されたデータサイズの大きさは何オクテッド（バイト）か？
- 受信データは正しいシーケンスの順序になっているか？

【GNOME端末１】
```
% tcpdump -t -x -nn -s 128 \('tcp[13]&8=8 or ip[2:2] > 128'\) and port 25
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 128 bytes
```
【GNOME端末２】
```
% telnet phoenix.cs.miyazaki-u.ac.jp 25
220 phoenix.cs.miyazaki-u.ac.jp ESMTP Postfix
```
【GNOME端末１】
```
IP 133.54.227.20.25 > 133.54.224.220.33242: P 4126549121:4126549168(47)
ack 3414328901 win 1448
        0x0000:  4500 0063 103b 4000 3c06 5dfc 8536 e314  E..c;@.<.]..6..
        0x0010:  8536 e0dc 0019 81da f5f6 2481 cb82 8645  .6........$....E
        0x0020:  8018 05a8 692f 0000 0101 080a 7e23 34ba  ....i/......~#4.
        0x0030:  05b2 3e29 3232 3020 7068 6f65 6e69 782e  ..>)220.phoenix.
        0x0040:  6373 2e6d 6979 617a 616b 692d 752e 6163  cs.miyazaki-u.ac
        0x0050:  2e6a 7020 4553 4d54 5020 506f 7374 6669  .jp.ESMTP.Postfi
        0x0060:  780d 0a                                  x..
```
【GNOME端末２】
```
HELO pc70.cs.miyazaki-u.ac.jp
250 phoenix.cs.miyazaki-u.ac.jp
```

図 1 パケット・キャプチャ

2.2 プロトコル設計

TCP はバイトストリームであるため送信したパケットは確実に受信元に送信される．またその順序も保証されるが TCP が担当するのはそこまでであり，アプリケーションでのデータ表現（メッセージ化）は TCP の上位層であるアプリケーション層のプロトコルが行う仕事である．よって，TCP ソケット通信におけるアプリケーション・プロトコル設計で重要になるのがメッセージ境界の識別である．これはメッセージのフォーマットをどのように定義するかということでもあり，TCP 上のバイトストリームからアプリケーションで使用するデータを正確に取得しなければならない．

メッセージ（メッセージ境界）を識別する代表的な手法として図 2 に示す「デリミタ方式」がある．デリミタ方式とは，メッセージの終端を表現する記号（これをデリミタという）でメッセージ境界を識別する方法である．テキストベースのアプリケーション・プロトコルに使用される方式であり，HTTP, SMTP, POP 等にも使用されている．

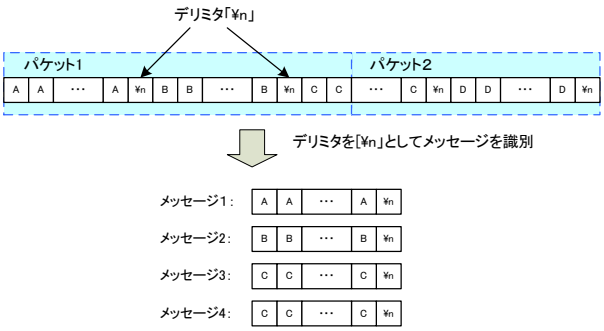


図 2 デリミタ方式によるメッセージの識別

2.3 マルチスレッド

プログラム（プロセス）は，基本的に 1 行ずつコードが実行されながら動作する．通常，分岐やループがあってもプログラム全体は 1 つの流れになっている．このような一連のプログラムの流れを「スレッド」と呼び，1 つのスレッドだけからなるプログラムを「シングルスレッドなプログラム」という．一方，1 つのプログラム（プロセス）で複数のスレッドを並行して実行するプログラムを「マルチスレッド」という．図 3 はマルチスレッドの処理を表現したものであり，処理 A（メインスレッド）からスレッドが 2 つ生成されて処理 B と処理 C を並行的に実行している．基本的にスレッドは対象の処理が終了すると終了となる．

この機構を用いて複数のクライアントを並行的に処理する図 4 に示すようなマルチスレッドサーバを実現する．具体的には，クライアントからの接続要求時にスレッドを生成し，対象クライアントへのサービスを提供する専用のスレッドとして処理を行う．

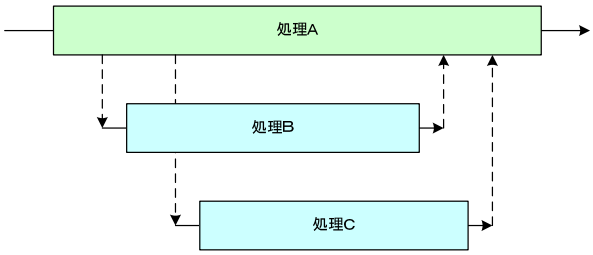


図 3 スレッド処理

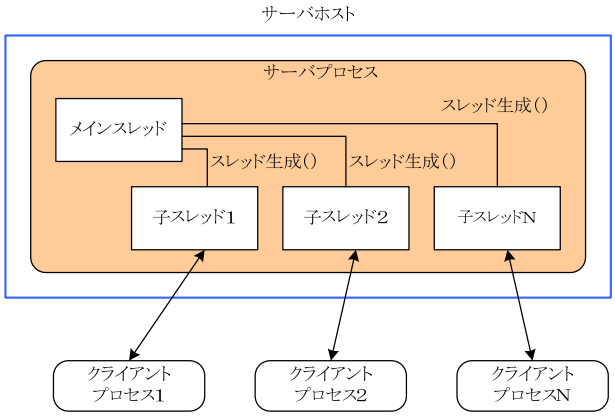


図 4 マルチスレッドサーバ

2.4 排他制御

共通資源のデータを複数のスレッドから安全に更新できるようにするためには，その資源に対する「データの読込み，データの計算，データの更新」のように分割できない部分（クリティカルセクションという）を排他制御する必要がある．この機構を実現する方法として図 5 に示した mutex ロックを用いた．mutex 変数を一種の鍵とし，一つのスレッドが鍵をロックしてクリティカルセクション内に入ると他のスレッドは鍵が解除されるまでその中に入ることができず待ち状態になる．鍵が解除された後，待たされていたスレッドは改めて鍵をロックすることができる．このように共通資源へのアクセスを同時には 1 つのスレッドに限るように制御する．

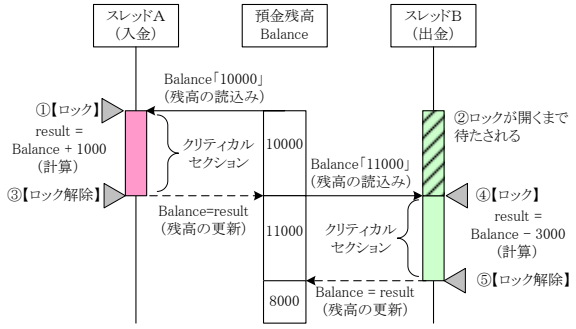


図 5 mutex ロックによる排他制御

3. 課題演習の改良

課題演習は，最終回のプレゼンテーションを含めて 8 週（16 コマ）の演習に改変した．表 1 に挙げた課題演習テーマとして 4 つの題材を取り上げ，1 グループ 3 名でテーマを選択する．各課題テーマに対して「システム要望書」が提示されており，これは利害関係者（ユーザ）がシステムに求める機能を記述した文書という位置付けであり，このシステム要望書をもとに開発プロセスに沿って作業を行う演習形態とした．

開発プロセスの工程には，要求定義・仕様化，分析設計，実装(プログラミング)，テストがあり，そのスケジュールを表 2 のように定め，さらに限られた時間の中で効率的に作業を進めるために各工程において作成する成果物を特定し作業の進捗状況を把握できるようにした．

表 1 課題演習テーマ

≪ Ⅰa 前期≫
ネットショップシステム（商品販売）
航空機チケット予約システム
ホテル客室予約システム
ネットレンタルシステム（レンタル業務）
≪ Ⅱa 後期≫
居酒屋座席予約システム
コンサートチケット販売管理システム
列車座席予約システム
レンタカー予約システム

表 2 課題演習のスケジュールと成果物

開発工程	成果物
要求定義・仕様化，分析設計	要求仕様書，DFD
分析設計	IPO，ファイル設計書， プロトコル設計書
実装（詳細設計）	モジュール構造図
実装	プログラム
実装	プログラム
実装，テスト	プログラム，テスト結果
プレゼン資料作成	プレゼン・ドキュメント
プレゼンテーション，デモ	

3.1 モデリング

今回の演習では，モデリング能力の育成を重視するため，一般的な開発プロセスにおける「要求定義・仕様化」，「分析設計」工程において，標準化されたドキュメントを用いて設計書を作成する．それに先立ちモデリングの進め方および各成果物

の記述方法について教育を行う．

■ モデリングの進め方

○ 要求定義・仕様化

要求定義・仕様化の工程では，ユーザが要望していることが何であるかを「要求」として的確に抽出し表現する．定義された「要求」に対しその要求を満足するための「仕様」を決めていく．「仕様」とは「要求」を満たすべき“具体的な振る舞い”の記述であり，最終的には何らかの形でプログラムのコードに変換されるものである．

- 要求仕様書に「要求」と「仕様」をまとめる

○分析設計

分析設計の工程では，要求仕様をもとにシステムが実現する機能を分析し，その機能をどのように実現するか実現方法を設計する．

- データの流れに着目して，データフローダイアグラム（DFD）を用いてプロセス（処理）とプロセス間のデータの流れ，プロセスとストア（ファイル等）間のデータの流れを分析する
- 要求仕様を実現するための処理手順を考える．その際，その処理に必要な入力とその処理によって生成される出力が何であるかも同時に考え，Input-Process-Output（IPO）としてまとめる
- IPO で使用されるファイルのデータ項目を分析しファイル設計書にまとめる
- IPO をベースにクライアント・サーバ間におけるプロトコルを設計し，プロトコル設計書にまとめる

■ 各成果物の記述方法

○要求仕様書の記述方法

- 要求仕様書にはシステムの目的と要求定義を記述する
- システムへの要求を整理しやすく体系化するために，システムを構成する機能などのカテゴリごとに要求定義を分割する
- 要求定義をカテゴリに分割した後，各カテゴリに含まれる要求を記述し階層化する

  「要求」とは利害関係者が求める，作ってほしいもの（こと）であり適切な範囲を持って記述する

  要求は動詞で表現する
- 要求の定義ができたらその要求を満たすための仕様をその要求の下に記述する

  仕様とは要求を満たすべき具体的な振る舞いである

  仕様は最終的には何らかの形でプログラムのコードに変換されることになる

○DFD の記述方法

DFD は，業務を人や組織からでなくデータの観点から表現するものであり，表 3 の 4 つのシンボル（記号）を用いて作成する．図 6 はネットバンクシステムの DFD 例である．ネットバンクシステムの主たる機能である「口座開設」，「残高照会」，「引き出し」，「預け入れ」，「振込み」がプロセスとして記述されている．

表 3 DFD の記号

記号名	表記法 説明
データフロー	データ名 → • データの流れを示す • 矢印の上または下にデータ名を記入する
プロセス	No プロセス名 • 入力データフローから出力データフローへの変換を行うプロセス（処理）を示す • 参照番号とプロセス名を記入する
データストア	データストア名 • データの蓄積（ファイル）を示す • データストア名を記入する
外部 （源泉／吸収）	発生源または行先名 • 分析対象業務の外部にあるデータの発生源または行先を示す • 外部の発生源または行先の名称を記入する

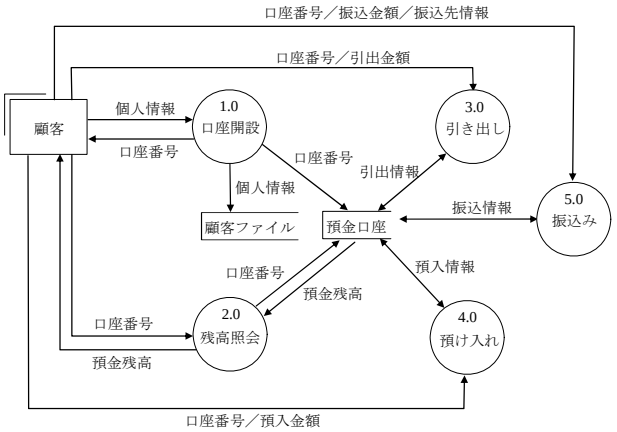


図 6 DFD の例

○IPO の記述方法

DFD の各プロセスに対して仕様分析を行うために，表 4 にまとめた INPUT（入力），PROCESS（処理），OUTPUT（出力）を示し，プロセスの処理内容を明確にする．

表 4 IPO の記述内容

INPUT	処理を実行するために必要となるファイル（データ）を記述する
PROCESS	処理の実行順序を記述する
OUTPUT	処理によって出力されるファイル（データ）を記述する

図 7 は，ネットバンクシステムの機能である「口座開設」に対する IPO の例である．この IPO から口座開設の機能は，4 つの処理で構成されていることがわかる．

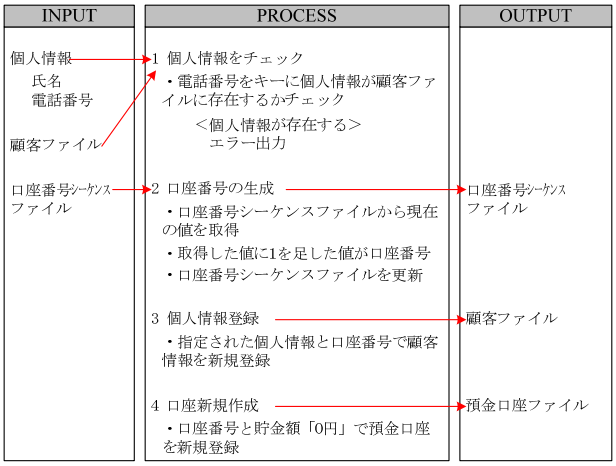


図 7 IPO の例

○ファイル設計書の記述方法

DFD で記述したデータストアを中心に，対象業務システムにおいて必要となるデータを詳細に分析する．

≪データ項目の収集≫

対象業務システムで使用するデータ項目を DFD と IPO から収集する．また，ユーザの要望が記述されているシステム要望書も収集源の一つになる．さらに与えられた情報以外に自分達のアイデアをプラスする．

≪データ属性の決定≫

収集したデータに対してデータ属性（データ項目名，桁数，形式（数値，文字列等），データの意味内容）を決定する．

≪実体とその実体に属するデータ項目の決定≫

対象業務システムにおいて，管理すべき実体（管理対象単位はここではファイルを意味する）を明確にし，それぞれの実体に属するデータ項目を決定する．

○プロトコル設計書の記述方法

プロトコル設計は，各機能においてデータ受け渡しに使用するデータフォーマットとデータ受け渡しの手順を作成する．

≪プロトコル・コマンド≫

クライアントから送信する要求メッセージとそれに対するサーバからの応答メッセージ，またはその逆のデータ受け渡しも含めて次の項目を決定する．

- コマンド：要求メッセージを識別する名称
- コマンド引数：コマンドに必要なデータ
- コマンドの説明：要求メッセージの説明
- レスポンス・ステータス：要求に対する応答
- レスポンス・データ：要求に対する応答データ

≪コマンド・シーケンス≫

要求メッセージ（コマンド，引数）と応答メッセージ（ステータス，データ）の通信手順を時系列に表現したものがコマンド・シーケンスであり，その例を図 8 に示す．

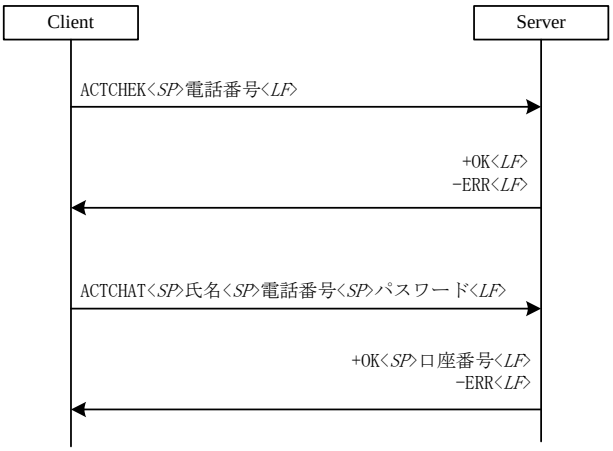


図 8 コマンド・シーケンスの例

3.2 実装

課題演習前半のモデリングによって作成した各種設計書と実装（プログラミング）の間には多少のギャップが存在する．実施したモデリングは広義な分類としては「外部設計」にあたり，それを実装するためには「詳細設計」が必要となる．詳細設計には複数のタスクが存在するが，時間的な制約と演習の特性から「モジュール構造化」のみを取り上げてそれをもとに実装を行うこととした．

■ モジュール構造化

モジュールとは処理を担当する個々の部品であり，そのモジュールが組み合わさって１つの機能を実現する．さらに機能が組み合わさってシステムを実現する．よって，粒度を考慮しながらモジュールを抽出し適切に構造化していくことが重要である．今回はモジュール構造図の作成方法を次のように示した．

- IPO を確認してモジュールの抽出を検討
- モジュールが担当する処理内容と範囲の決定
- 各モジュールの引数と戻り値の決定
- 各モジュールで使用する入力ファイル，出力ファイルの決定
- モジュール構造関係の決定
- サブコマンド呼び出しの明確化

航空機座席予約システムのモジュール構造図例を図 9 に示す．

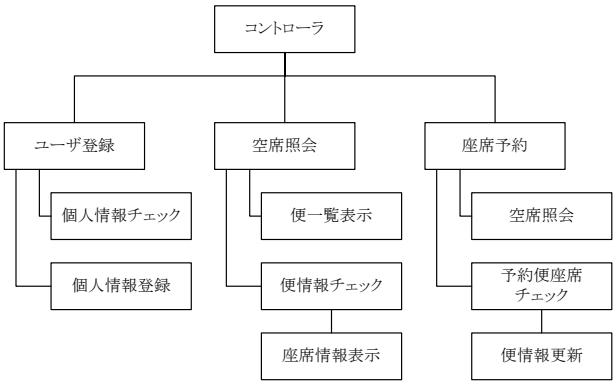


図 9 モジュール構造図

■ 実装工程の進め方

要求仕様書，DFD，IPO，ファイル設計書，プロトコル設計書およびモジュール構造図のドキュメント作成が終了した後，実装（プログラミング）に着手する．効率的に実装を進めていくにあたり実装工程の進め方を表 5 のように指示した．

表 5 実装工程

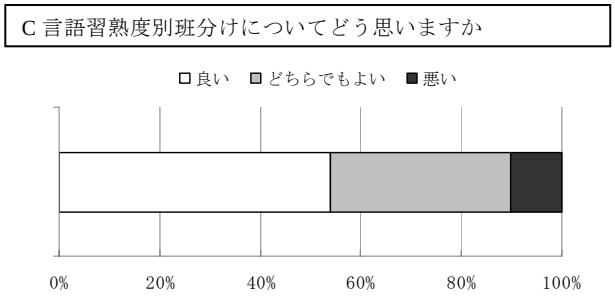
≪STEP1≫
抽象データ型の定義（作成） 抽象データ型を操作するモジュールの定義（作成） 排他制御の検討
≪STEP2≫
各機能モジュールの作成 処理制御モジュール（コントローラ）の検討
≪STEP3≫
コントローラの作成 各機能モジュールの組込み（結合） 排他制御の実装
≪STEP4≫
クライアントーインターフェイスの作成

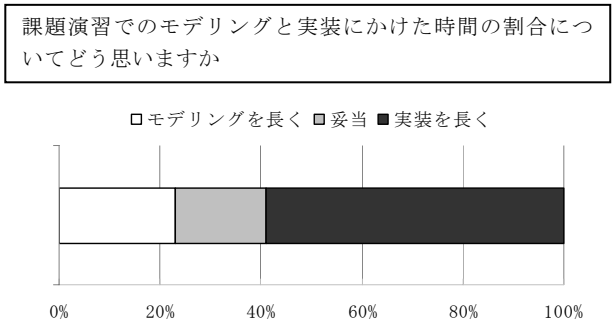
4. 演習の評価

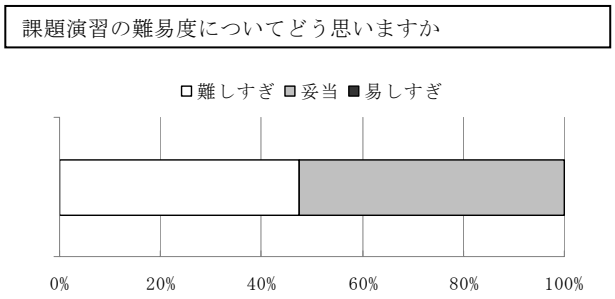
演習内容の評価の一環として，受講学生向けにアンケート調査を実施した．アンケート調査結果を以下に示す．

今回の課題演習の班分け（1 グループ 3 名）は，演習の初回に実施した C プログラミングテストの結果を基に C 言語の習熟度に応じて実施した．これはプログラミングが得意な学生が一人でプログラムを作成してしまうことを避け，習熟度が高い者は高い者同士で，また習熟度が低い者は低い者同士で開発を行えば，人に任せることなく自分が責任を持って担当しなければならない部分がある程度均等化され，協力体制がとれたグループになることを意図した取り組みであった．結果として，どのグループも細かな不具合はあったもののデモを実施できる程度まで作りこむことができ，プログラミングが得意な学生のグループはより高度な機能の実装を試みるなど概ね良好に機能した．このことに対する学生の評価は，「良い」が 54%，「どちらでもよい」が 36%，「悪い」が 10%であり，「良い」と回答した学生が受講者の半数であったことから，概ねこちらの意図したことが伝わった結果であった．

課題演習では，モデリング（分析設計）に重点を置き，着実に成果物を作成しながらモデリング作業を進め，実装（プログラミング）に入るように教育を行った．結果として，どのグループも主要な機能を完成することはできたが，全体的に時間が足りないという印象であった．課題演習でのモデリングと実装にかけた時間の割合についての学生の評価は，「モデリングを長く」が 23%，「妥当」が 18%，「実装を長く」が 59%であり，このことからもシステムを完成させるためには実装に充てる時間が長くほしいという結果であった．効率的に実装を進めるにはしっかりした設計が必要であることを実感した学生も多かったが，期限内にシステムを完成させるためにはやはり実装が優先されてしまうのは，現実のシステム開発の現場でもよくあることであり今後の検討課題としたい．課題演習の難易度については，「難しすぎ」が 48%，「妥当」が 52%，「易しすぎ」が 0%であり，やや設定した課題演習が難しかった結果となった．







5. まとめ

学生の今までのプログラミング(演習も含めて)は，仕様のデザインを頭で描きながら同時にコーディングを行うスタイルであった．しかし，複数人で開発を行うシステムの規模になればこのスタイルでは対応することができず，モデリングというものが非常に重要になってくる．今回は，情報工学特別演習Ⅰa／Ⅱa（ネットワークプログラミング）の演習において，この点を踏まえて演習内容の充実とモデリング能力の育成を目指した演習教材（演習内容）の改良を行った．

課題演習の題材として取り上げたテーマ（ホテル予約システム，航空機座席予約システム，コンサートチケット販売管理システム等）のシステムは，現在の市場においてはWebシステムとして構築されるのが一般的であるが，本演習はC言語によるネットワークプログラミングをベースとしているため，マルチスレッド型のクライアント／サーバモデルをシステムのアーキテクチャとし，その上でシステム要望書をもとに要求定義･仕様化，分析設計，実装，テストの一般的な開発プロセスに沿って，成果物である各種設計書を作成しながら作業を進め効率的な実装ができるようにモデリング面における教育を重点的に行った．

モデリングにおいて，DFDやIPOなどのモデル図を用いて作業を進めたが，学生にとってはこのような図表を作成すること自体が始めてであり，書き方やどのように表現するか等モデリング技法に慣れるまでに時間がかかってしまい課題演習の時間不足の原因となった．学生は時間を割いてまでこれらの設計書をなぜ作成する必要があるのか始めは半信半疑であったが，グループで作業を進めていくなかで，設計書は決定したことを確認できる道具，詳細を検討するときにベースとなる道具，グループ内での認識のズレを防止できる道具であり，作業の効率化と作業の分担を適切に行うことができる十分なメリットがあることを実感できのではないかと思われる．

今後の検討課題は，①課題演習の時間配分，②モデリングで用いる標準化の洗練，③モデリング技法の早期教育である．これらの検討課題をクリアして学生が今まで以上に興味を持って意欲的に取り組める演習に発展させていきたい．

参考文献

1)“平成 20 年度情報工学特別演習ⅠaⅡa テキスト”，宮崎大学工学部情報システム工学科（2008）
2)“アプリケーションエンジニアⅡ（システム分析・7 設計 研修テキスト”，システムプラネット