データベースシステム開発における
形式仕様記述に関する考察

97B619　山本　英貴

指導教員　　野呂 昌満

1　はじめに

ソフトウェア開発において，仕様には，要求仕様と設計仕様がある．現状では要求仕様と設計仕様の関係がドキュメントとして存在しないという問題点が挙げられる．関係が明確でないために要求仕様の変更が起こった場合，設計を変更するには実際にプログラムを見て分析をおこない，変更部分を探さなくてはならず，労力がかかる．仕様と設計の関係が明確になれば，要求仕様の変更を設計の変更に反映させることができることが期待できる．そうすることによって，設計の変更をプログラムレベルで考えずにドキュメントのレベルで議論をすることが期待できる．

本研究の目的は，仕様を形式的に記述することにより，設計と仕様の関係について考察することである．

本研究では，実例として住所録システムを挙げ，仕様を記述し，設計を行う．仕様に変更を加え，それに伴い設計も変更する．その過程での考察をする．

2　形式手法

形式的に記述する方法の一つとして，数学的な理論に基づく手法がある．構文が定義されており，数学的な意味付けがなされている．形式手法の特徴は以下である．

- 意味は明白で，曖昧さを取り除くことができ，矛盾点の早期発見に役立つ．
- 簡潔さを損なうことなく，正確に表現できる．
- 数学の証明技法の応用が可能で，仕様記述の厳密な証明ができる．

2.1　仕様記述言語 Z

数学的表現である集合論，述語論理に基礎をおいている表記法であり，スキーマを用い記述される．

3　住所録システムの仕様と実現

本研究では，住所録システムのついての考察をおこなう際に，はじめは非常に簡単な仕様を記述することとした．はじめの段階では，少ない機能のみをもった住所録システムをを考え，記述する．その後，仕様に変更を加えていく．

3.1　仕様

以下のような仕様について記述する．

- 名前，住所によって構成される．
- 操作として，登録，削除，検索がある．

3.2　形式記述

Z で記述すると以下のようになる．

$$[NAME, ADDRESS]$$

$$REPORT ::= OK \mid AlreadyKnown \mid Error$$

AddressBook

$$known : PNAME$$
$$book : NAME \pfun ADDRESS$$

$$known = \mathrm{dom}\ book$$

3.3　実現方法

近年，インターネットの普及に伴い，ネットワークを介したアプリケーション多数開発されている．そこで本研究では，住所録システムを Web を用いたアプリケーションとして作成することとした．Web とデータベースを連係させ，ネットワークを介してデータベースにアクセスすることによって住所録システムがもつ機能を実現する．

3.4　モデル

本研究では，モデルとして 5 層モデルを考えた．5 層モデルは以下のように構成される．

- ブラウザ層: Web ブラウザが位置する
- コントロール層: HTML への変換等を行う
- アプリケーション層: ユーザからの入力のチェック
- ファンクション層: データベースに対する操作を管理
- データベース層: データベースがおかれる

3.5　クラス図

以下に示すものが，今回設計したクラス図である．

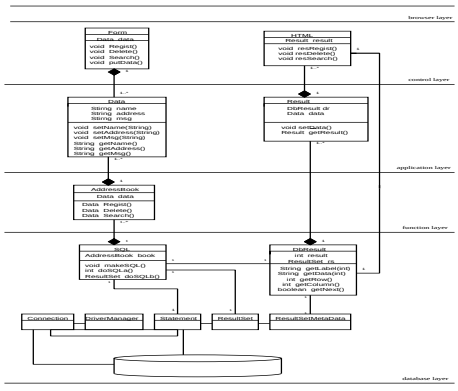

図 1: クラス図

3.6 実現

本研究では，Java を用い実装を行うこととし，設計をおこなった．また，DBMS(DataBase Management System) として PostgreSQL を用いる． コントロール層では，入力のためのフォームの生成や，結果を表示するための HTML を生成するためのクラスを配置した． アプリケーション層では，クライアントからの入力に対するデータのチェックをおこなうクラスを配置した．ファンクション層では，データベースに対する操作を行うクラスを配置し，データベース層では，SQL ステートメントの生成，実行をするクラス，JDBC，データベースを配置した．

4 仕様の変更と分析

今回作成した仕様に変更を加え，変更に伴う設計の変更との関係について分析を行う．仕様に対する変更は，次の種類について考え，分析した結果を以下に示す．

- 住所録システムに機能を追加する．
- 住所録に登録できるデータ項目の変更．
- 同姓同名の人の登録を可能にする．

今回は，システムに機能を追加する例として，一覧の出力を考え，登録するデータに電話番号と電子メールアドレスを追加したデータを考えた．

表 1: 分析結果

仕様	仕様の変更点	設計の変更
v1	なし	なし
v2	操作の追加	ファンクション層
v3	データの追加	アプリケーション層
		データベース層
v4	v2 と v3 の統合	すべて
v5-1	同姓同名を考慮	アプリケーション層
v5-2	同姓同名を考慮	すべて

5 考察

本研究では，はじめに記述した仕様は非常に簡単な機能しかもたないものであった．この仕様を記述する際には，すぐに Z で記述することができた．
この仕様をもとに設計をおこなう際，モデルとして 5 層モデルを考えた． 5 層モデルをもちいることによって，後に変更を行う際に変更しやすいように設計することができた．例えばデータベースを今回使用した DBMS ではなく，テキストファイルをデータベースとして考えた場合には，データベース層のクラスを変更するだけで良く他の層は変更しなくてよいといったことが挙げられる．

今回設計を行う段階で，はじめに設計したものは，仕様の変更をした場合，設計の変更で，すべての層を変更しなくてはならないことになってしまった．仕様を記述したにもかかわらず，設計において問題が発生した．この問題から，仕様からただ設計をおこなっただけでは，しっかりした変更に強い設計はできないことがいえる．
設計の失敗があり，今回の設計をおこなうことができた．

今回仕様に変更を加え，それに対する設計の変更点の分析をおこなった．結果を見ると，データに関することとシステムに対する操作の 2 つが考えられ，それぞれ変更するクラスが配置されている層がことなっている．今後，変更をおこなう際には，この 2 つの場合について考えることにより，変更を加える層がどれであるのか判断することができる．
今回変更を加えたのは数パターンのみであったが，更に変更を加え続けていくことにより，今回分析したレベルよりももっと詳細化された分析がおこなわれることが期待できる．

今回作成した仕様では，Z で記述したものがすべての層について記述してあるわけではない．Z で記述したものは設計での一部分でしかない．各層について仕様を記述することができれば，今まで記述してきた仕様よりも構造が整理されており，変更についても仕様で変更を受けた層がそのまま設計においても変更をおこなう層であると判断することができ，仕様と設計間の関係についても更に明確になることが期待できる．

6 おわりに

本研究では，一つの実例を挙げ，その実例について仕様を記述し，設計を行った．更に，仕様に変更を加えることによる設計の変更について分析を行い，仕様と設計の関係について考察を行うことにより，要求仕様と設計仕様の定義を明らかにし，その両者間の関係についても，明らかにすることができた．
しかし，仕様の変更についてはまだ他の変更についても考える必要がある．また，今回，対象にした実例以外の例を挙げて考察することも必要である．
今後の課題として，以下を挙げる．

- 他の変更による仕様と設計に関する考察
- 他の実例を挙げ考察を行う．

謝辞

本研究を進めるにあたり，一年半御指導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張漢明先生，大学院の熊崎敦司さん，池内仁さん，宗宮健仁さん，青木俊介さんに深く感謝致します．

参 考 文 献

[1] J.M.Spivey,*The Z Notation A Reference Manual Second Edition*, 1992.

[2] Bernard Van Haecke,「*JDBC:Java database Connectivity*」, IDG Book Worldwide,1997.