卒業論文

グラフ文法による構文的プログラム仕様書処理系の実現

An Impremention of Program Speci¯cations Processing System Based on Graph Grammars

泉　博貴
Hirotaka IZUMI

あらまし　本研究では，グラフ文法 [1, 2, 3, 4, 5] と表形式 [1, 3] に関する研究及び Swing[8, 9, 10] を用いた Java プログラミング [6, 7, 8, 9, 10] による HifotmED[3] の作成について扱う．Java は機種への依存がほとんどないため，同じプログラムで様々なハードウェアでの使用が可能である．

本論文では，参考文献 [1, 2, 3, 4, 5] を用いて表形式とグラフ文法についての解説及び本論となる Java を用いた HiformED の作成についての説明を行い，最後に本研究に関する考察と今後の課題について述べる．

キーワード プログラム仕様書, グラフ文法, Hiform Java, Swing, HiformED

1　はじめに

本研究では，グラフ文法 [1, 2, 3, 4, 5] と表形式 [1, 3] に関する研究及び Swing[8, 9, 10] を用いた Java プログラミング [6, 7, 8, 9, 10] による HiformED[3] の作成について扱う．

プログラム仕様書システム Hiform はプロダクト要素の集合（記入項目）として ISO6592-1985(JISX0126-1987：応用システム文書化要項) のガイドラインにある項目を用いている [1, 3].

Java はハードウェアにほとんど依存しないオブジェクト指向言語で，1991 年にサンマイクロシステムズ社が情報家電・携帯端末用のプログラム言語として Oak というプロジェクト名（後に Java に改名される）で開発が行われ，1995 年 5 月に Java が正式発表された．同年にネットスケープコミュニケーションズ社が採用したのを契機に，他社も追従する形で採用するようになった．1996 年には Java 仮想マシン（2.2.2 を参照のこと.）仕様及び Java Language 仕様を出版し同年，JDK(Java Developement Kit) [1]1.0 をリリースした [6, 8]．その後バージョンアップを繰り返し，現在のバージョンは本論文作成時点で 1.3 となっている．

Swing は GUI の構成要素（ウィンドウ、ダイアログ等）を作成するためのツールで，1997 年 3 月に開かれた \Java One" で発表され，翌年 2 月 28 日に Swing1.0.1 として正式リリースされた．Swing が発表される以前は JDK1.0 のリリース時から AWT(Abstruct Window Toolkit) [2]が用いられていた．

本論文では第 2 節で各用語に関する特徴等を説明し，第 3 節で表形式とグラフ文法についての解説を行い，第 4 節で Java を用いた Hiform ED の作成についての説明

[1]サンマイクロシステムズ社が提供する Java プログラミングを行う上での開発環境．作成したプログラムを実行するための実行環境も含まれている．

[2]GUI の構成要素を作成するためのツール．今日の GUI 重視の状況からデザイン面・機能面で不足している等の問題があったが，Swing のリリース後も Java の標準パッケージ内に残されており，その一部は Swing の基盤として利用されている．

を行う．最後に，本研究に関する考察及び今後の課題等を第 5 節で述べる．

2 準備 [1, 3, 6, 7, 8, 9, 10]

2.1 Hiform96 の特徴

\Hiform96" はプログラミング教育を目的として開発された表形式の仕様書システムであり，以下の 17 種類の定型用紙が定められている．

カテゴリ A：プログラム文書
- A1：プログラム概要書
- A2：プログラム管理要綱書
- A3：プログラム契約事項記述書
- A4：プログラム用語解説書
- A5：プログラム仕様書―1
- A6：プログラム仕様書―2

カテゴリ B：データ文書表紙
- B1：データ文章構造図
- B2：データ管理文書
- B3：データ技術文書

カテゴリ C：作業手順文書表紙
- C1：作業手順仕様書構造図
- C2：作業手順管理文書
- C3：作業手順書
- C4：作業手順技術書
- C5：作業手順流れ図
- C6：作業手順例

カテゴリ D：プログラム構造図表紙
- D1：プログラム構造図
- D2：インターフェース仕様書

program name : hanoi_man	A General document
subtitle : hanoi	
library code : cs - 2000 - 01	version : 1.0
author : Tomokazu Arita	original release : 1999/12/22
approver :	current release : 2000/01/28
key words : Hanoi Tower	CR-code :
scope : Fundamental	
variant :	
language : Java	software req. : JDK1.2
operation : interactive batch realtime	hardware req. :
references :	

function : 1. list and explanation of input data or parameter,
2. list and explanation of output data or return value.

1. list and explanation of input data

```
int n;   [ Number of Plates ]
String target;      [ Target Symbol ]
String work;        [ Working Symbol ]
String destination; [ Destination Symbol ]
```

2. list and explanation of output data and return value

```
output data : No. to be moved :Source Symbol -> Destination Symbol
return value : void
```

example :

1. Example of Operation

```
hanoi(5, A, B, C)
```

2. Example of Output

```
1: A -> C
2: A -> B
1: C -> B
3: A -> C
1: B -> A
.....
```

図 1: A1:プログラム概要書

2.2 Java と Swing

2.2.1 Java の特徴

Java 言語の特徴としては以下の通りである．

(1) プログラム全体がオブジェクトという部品によって構成されている \オブジェクト指向言語" である．
(2) \マルチスレッド" を扱える．
(3) プログラム記述上の誤りはコンパイルした時点で可能な限り発見されるため，\誤りを起こしにくい".
(4) 特定のコンピュータ上の Java 言語で開発したプログラムが他の OS や他のハードウェアでも動作できるので，C や C++のように \移植" する必要がほとんどないため，\機種依存性が非常に少ない".
(5) Web ブラウザ上で動作する \アプレット" を作成できる．

2.2.2 Java 仮想マシン

\Java 仮想マシン（Java Virtual Machine）" とは，クラスファイル（拡張子 \.class"）を CPU 上ではなくそのコンピュータ上で実行するための仮想的な機械（つまり本当の機械ではなくソフトウェアの一種）である．この仮想マシンの存在により，クラスファイルはどのハードウェア上でも，どの OS 上でも（すなわち特定のプラットフォームに依存せずに）そのまま実行することができる．例えば，Windows 上の仮想マシンで動作するクラスファイルは UNIX 上の仮想マシンでも動作し，またその逆（つまり，UNIX 上の仮想マシンで動作するクラスファイルは Windows 上で）も動作可能である．

2.2.3 Swing の特徴

\Swing" は JDK1.0 リリース時から用いられてきた AWT が抱えるさまざまな問題を解決し，今の時代に求められる洗練された GUI を持つアプリケーション開発を行うサブシステムである．特徴としては主に，

(1) Swing のパッケージはすべて Java で記述されている．
(2) 各プラットフォーム間で同じ外見と動作をする．
(3) 実行時にルックアンドフィールの切り替えが可能である．
(4) きめ細かい設定やカスタマイズが可能である．

が上げられる．

3 表形式とグラフ文法 [1, 3, 4, 5]

3.1 表形式と複合図

表形式文書の形式化として、複合図を用いる．これにより表形式文書の構造（表形式文書の各項目間の関係）を表現する．以下の図 2 で表形式を表す複合図を示している．

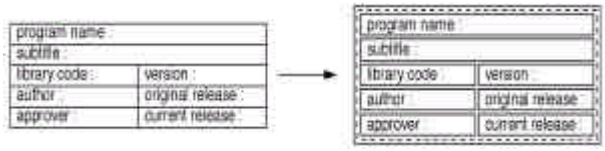

図 2: 表形式とそれに対応している複合図

3.2 複合図とマーク付きグラフ

マーク付きグラフは以下のように決められる．

(1) マーク付きグラフのマークは複合図の項目を示す．
(2) 辺のラベルは各項目間の関係を示す．

ここで，複合図に対するマーク付きグラフを以下の例で示す．ただし，辺のラベルは各項目間の関係を示しており，\lf" は \left of " を \ov" は \over " をそして \in" は \within" を示している．

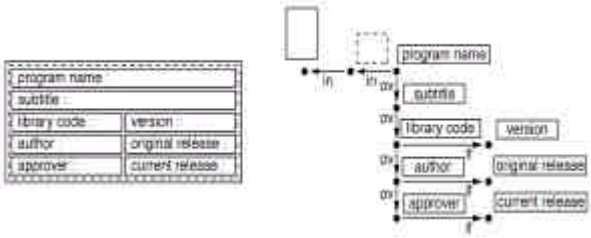

図 3: 図 2 で示した複合図とそれに対応したマーク付きグラフ

3.3 NCE グラフ文法

§ は頂点のラベルのアルファベットで，¡ は辺のラベルのアルファベット，§; ¡ 上のすべてのグラフの集合を $GR_{§;¡}$ で示す．

§; ¡ 上で組み込まれているグラフは 2 組 (H; C) で定められており，H は $GR_{§;¡}$ に属し，C µ §£¡£¡£V_H£fin; outg は (H; C) の接続関係である．また，C の各要素 (®; ¯; °; x; d) は (H; C) の接続指示である．ただし，® 2 §, ¯; ° 2 ¡, x 2 V_H

、d 2 fin;outg である．§;¡ 上で組み込んでいるすべてのグラフの集合は $GRE_{§;¡}$ で示される．

定義 3.3.1

edNCE**グラフ文法**は6つ組G = (§;¢;¡;-;P;S) で定められている．ここで，§ は頂点のラベルの集合，¢ µ § は終端頂点のアルファベット，¡ は辺のラベルの集合、- µ ¡ は終端辺のラベルのアルファベット，P はプロダクションの有限集合，そして S 2 §－¢ は初期非終端記号である．また，プロダクションは X ! (D;C)（ただし、X 2 §－¢;(D;C) 2 $GRE_{\Sigma;\Gamma}$）となる．

□

3.4 表形式と文脈自由グラフ文法

定義 3.4.1

属性 NCE **グラフ文法**は AGG = ħG;Att;Fi で定められ，

(1) G = (§;¢;¡;-;P;S) は AGG の基底（文脈自由）グラフ文法であり，P における各プロダクション p は p = X ! (D;C) で示される．また，Lab(D) はグラフ D のノードにおいてノード記号がラベル付けしているすべての集合を示す．

(2) G の部分集合 V に属するすべてのノード記号 Y は２つの有限集合 Inh(Y);Syn(Y) に分けられる．ただし，Inh は継承属性，Syn は合成属性を意味する．すべての非終端ノード記号 X の属性の集合を Att(Y) = Inh(Y) [Syn(Y) で示す．ここで，$Att(Y) = \bigcup_{Y \in V} Att(Y)$ は AGG の属性の集合である．ただし，Inh(S) = Á とする．Y に属する属性 a を a(Y)、a の値の集合を V(a) と示す．

(3) すべてのプロダクション

$$p = X_0 \ ! \ (D;C) \in P$$

を結びつけたものは，

$$Syn(X_0) \ [\ \bigcup_{Y \in Lab(D)} Inh(Y)$$

のすべての属性の定める意味規則の集合 F_a である．

ここで，意味規則は属性 $a_0(X_{t_i})$ が

$$a_0(X_{i_0}) := f(a_1(X_{i_1});\cdots;a_m(X_{i_m}))$$

$$(0 \cdot l_j \cdot jLab(D)j,\ X_{i_j} \in Lab(D),\ 0 \cdot j \cdot m)$$

の形を持っていることを定めている．ここで，jLab(D)j は集合 Lab(D) の基数を示し，f は $V(a_1(X_{i_1});\cdots;a_m(X_{i_m}))$ から $V(a_0(X_{i_0}))$ への写像である．この状態で $a_0(X_{i_0})$ は $a_j(X_{i_j})(1 \cdot j \cdot m)$ に依存しており，また集合 $F = \bigcup_{p \in P} F_p$ は AGG の意味規則の集合と呼ばれる．

□

3.5 Hiform と属性順位グラフ文法

Hiform 文書を特徴づける属性グラフ文法について考える．このように特徴づけられた形式を Hiform2000 と呼ぶ．Hiform2000 を形式化する文法は Hiform 属性グラフ文法（Hiform Attribute Graph Grammar：HFAGG）と呼ばれる．HFAGG のプロダクションは 280 あり，(プロダクションの一部について図 4 を参照）グラフの開始のマークは \[struct]00 である．各プロダクションは意味規則に関係しており，それらの規則は主に状態や各項目のサイズを評価するのに使われる．各プロダクションの規則においてマークの右下に付けられている数字はプロダクションの認識番号である．

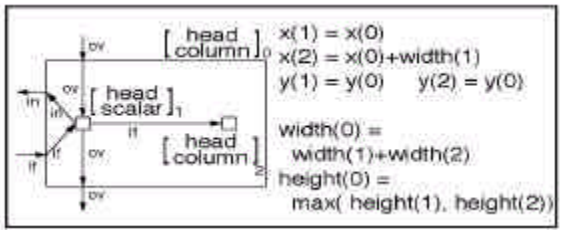

図 4: プロダクションの一部

3.6 Hiform におけるレイアウト問題

複合図におけるレイアウト問題は属性評価によって解かれる．その評価を行うために属性 x;y;width;height

を用いる．ここで，x;y はそれぞれ x 座標，y 座標を計算するのに使われ，width;height はそれぞれ幅と高さを計算するのに使われる．図 5 は属性評価の過程を示したもの（ただし、図 5 において \programname[00] と \subtitle[00] の幅は 2，それ以外の幅は 1，また各項目の高さは 1 でマージン（余白）は 0 となっている.）であり，その導出木でできたものが図 6 に示されるものとなる．

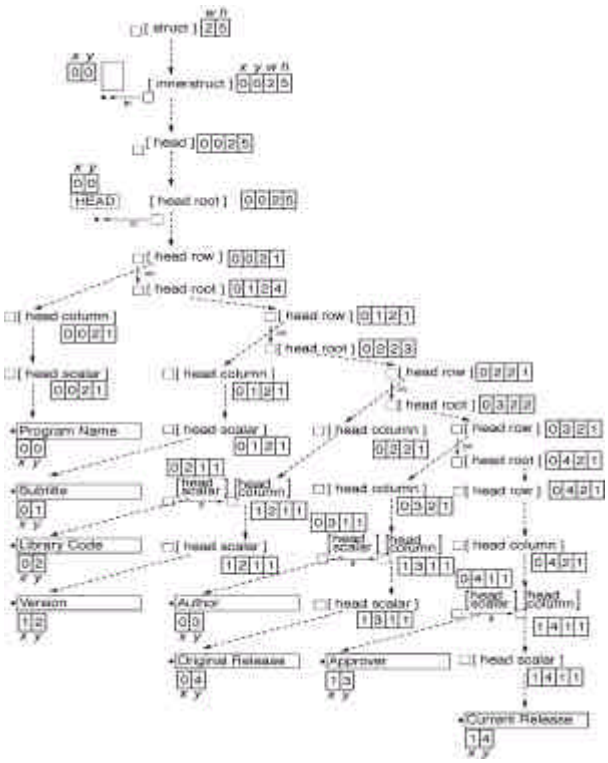

図 5: 属性評価の過程

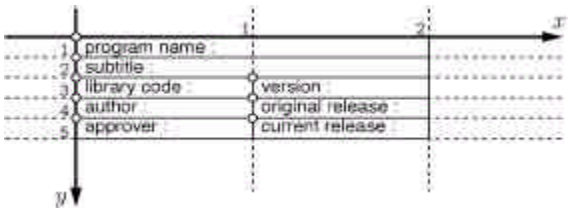

図 6: 導出木でできたもの

4 HiformED

ここでは，HiformED[3] に関するソフトウェア開発について扱う．使用する言語は前述した通り Java を使用する．

本研究における HiformED の開発において，次の 2 点を考慮する．

(a) 用紙選択ダイアログ等から入力されたデータをもとに目的の様式を作成する．
(b) テキストエディッタで直接入力したプログラムから目的の様式を作成する．

4.1 以降で上記の事柄について説明する．

4.1 用紙選択ダイアログからの目的様式の作成

ここでは上記 (a) の事柄つまり，用紙選択ダイアログから各入力項目で各項目を入力し，そのデータをもとに目的の様式を作成することを考える．これはテキストエリアから直接ソースを入力するのが不慣れな人にも簡単に目的の様式の作成を可能としたものである．この一連の流れを下図（図 7）に示す．

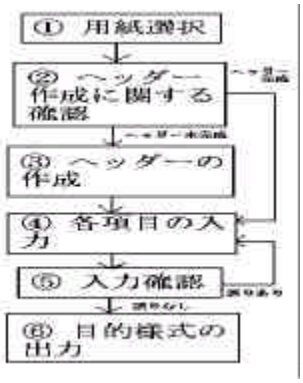

図 7: 用紙選択から目的様式の作成までの流れ

(1) **用紙選択ダイアログ**：ここでは作成したい様式の選択を行う場所である．

(2) **ヘッダー作成に関する確認**：この項目では，各様式に共通であるプロジェクトコード等の入力がすでに完了しているかを問うダイアログである．すでにヘッダーの作成が完了していれば，(3) の項目をスキップできる．

(3) **ヘッダーの作成**：(2) でヘッダーの作成を行っていない場合にこの項目が表示される．ここで，各様式に共通なヘッダーの入力を行う．

(4) **各項目の入力**：ここでは各様式に対応した入力ダイアログが表示される．なお，複数行入力を行う項目についてはスクロール及び自動折り返しが可能になっている．(図 8 参照)

図 8: 各項目の入力ダイアログ

(5) **入力確認画面**：各項目の入力が完了すると，入力された内容がここで表示される（図 9 参照）．もし記入間違い等を発見した時は (4) に戻って入力し直すことが可能である．すべて正しければ，(6) に進む．

(6) **目的様式の出力**：ここで今までに入力されたデータをもとにして，目的の様式が画面に表示される．

現在は (5) の上図のような入力確認画面の表示をする所までを完成しており，今後は入力データを保存し，そのデータから目的の様式を表示すること（つまり，(6) の完成）を目指している．

図 9: 入力項目の確認ダイアログ

4.2 テキストエディッタからの作成

ソースを入力して目的の様式を作成するために，テキストエリアを用いる．これは，アプリケーションの開始時および \MENU" ！ \NEW" で表示される．また，このテキストエディッタにはスクロール機能を追加している．図 10 においてテキストエディッタおよびソース入力時の状態を示す．

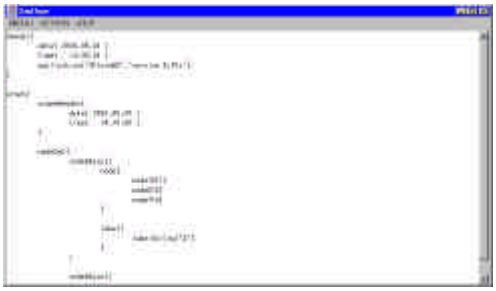

図 10: ソース入力時のテキストエディッタ

このテキストエディッタにおいて，ファイルのセーブおよびロードが使用できる．これらはそれぞれ \MENU" ！ \SAVE"，\MENU" ！ \LOAD" で使用する．ただしロードにおいては，指定された拡張子のみを読み込むようにしている（図 11 を参照）ので，セーブ時に指定外の拡張子でセーブするとそのファイルはこのアプリケーションで開くことができなくなる（図 12 を参照）．これは重要なファイル（特に，Windows においてはシステムファイル等）の読み込みを防ぐためである．

図 11: 指定外の拡張子を持つファイルの選択

図 12: 選択後のエラー表示

次にセーブ機能について説明する．主な内容は上書き保存ができる点である．これはロードした時のファイル名とセーブ時のファイル名が一致した時に上書き保存についてのダイアログが表示され，\はい" を選択することで（上書き）保存が行われる．

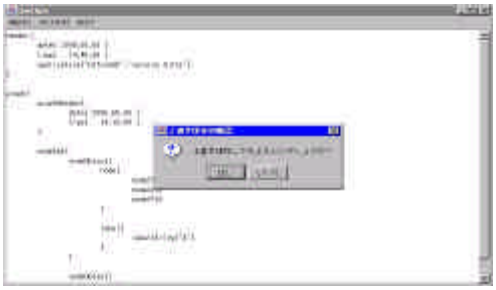

図 13: 上書き保存の確認

ここでは以上のことまでができており，内部構造についての作成を行っていない．したがって今後は内部構造の作成を行い，プログラムの直接入力から目的の様式を表示することを目指している．

5 おわりに

今回の研究では主に Java を用いたアプリケーション開発を行ってきたが，Java アプリケーションの実行に関しては JDK1.3 では確かに以前のバージョンに比べては早くはなってきているが，まだまだ遅く感じる．今後のバージョンアップで実行速度がさらに速くなってほしい．そうすれば，目的の様式を得るのに待たされる時間が少しは解消されるはずである．

また，HiformED の開発はまだ途中であるが，今後完成することになれば誰もが簡単にプログラム仕様書の作成が行え，その余った時間でプログラミングの方に力を注いでくれるであろう．そのためにはグラフ文法に関する知識を多く吸収し，それを応用させて内部構造等の作成を早急に行わねばならない．

参考文献

[1] T. Arita, Attribute Graph Grammars and Tabular Forms, 日本大学大学院総合基礎科学研究科修士論文, 2000, 5-7 ¢ 9-17

[2] Grsegorz Rozenberg(Ed.), Handbook of Graph Grammar and Computing by Graph Transformation, World Scienti¯c Publishing, 1997, 16-23

[3] T. Arita, K. Tomiyama, Y. Miyadera, K. Sugita, K. Tsuchida and T. Yaku, Syntactic Processing of Diagrams by Graph Grammars, Proc. IFIP WCC ICS2000, 2000, 145-151

[4] D. Janssens, G. Rozenberg, Graph Grammars with Neighbourhood-Contorolled Embedding, Theoretical Computer Science 21, 1982, 55-74

[5] Tomokazu ARITA, Kimio SUGITA, Kensei TSUCHIDA and Takeo YAKU, Syntactic Tabular

Form Processing By Precedence Attribute Graph Grammars, Pro. IASTED AI 2001 to appear, 2000

[6] 結城 浩,\ Java 言語プログラミングレッスン（上）", ソフトバンクパブリッシング, 1999, 東京都, 355

[7] 結城 浩,\ Java 言語プログラミングレッスン（下）", ソフトバンクパブリッシング, 1999, 東京都, 320

[8] 大村 忠史,\ Swing による Java GUI プログラミング", カットシステム, 1998, 東京都, 384

[9] 大村 忠史,\ Swing による Java GUI プログラミングII", カットシステム, 1998, 東京都, 354

[10] 大村 忠史,\ Swing による Java GUI プログラミングIII", カットシステム, 1999, 東京都, 348