数理解析研究所講究録
第1744巻 2011年 11-14

UMLパッケージ図に対するグラフ文法

電気通信大学 後藤 隆彰 (Takaaki Goto)
The University of Electro-Communications

電気通信大学 西野 哲朗 (Tetsuro Nishino)
The University of Electro-Communications

東洋大学 土田 賢省 (Kensei Tsuchida)
Toyo University

1 はじめに

図式表現は，その視認性の良さからソフトウェア設計や開発においてよく利用されている．プログラム図式はこれまでに，Hichart (Hierarchical flowchart language)，PAD (Problem Analysis Diagram)，HCP(Hierarchical and Compact Description Chart)，SPD(Structured Programming Diagram)等，様々なプログラム図の記述言語が報告されている．また，それらの言語に基づいた多くのCASEツールが開発されている [1, 2].

一方，ソフトウェア開発におけるモデリングにおいて，近年UML(Unified Modeling Language)が提案され，2005年にはISO/IEC 19501にて標準化が行われている．これまでに，多くのシステムの分析や設計，実装に用いられている．UMLはモデリングのための言語であり，クラス図やシーケンス図などの様々な図を用いてシステム開発の上流工程から下流工程の設計を行うことができる．

これら図式表現をコンピュータ上で自動的に処理するためには，まずプログラム図の構文を定義する必要がある．また，プログラム図などの2次元データを対象として構文解析を行うためには，各要素間の関係が記述される必要がある．それらを実現する手法としてグラフ文法が有効な手段として考えられる．グラフ文法は形式的な手法の1つであり，生成や解析といったメカニズムを厳密に定義できる手法である．グラフ文法に関する研究は，Rozenberg [3] などにより行われている．また，UMLに関するグラフ文法・グラフ変換に関する研究は，[4, 5] などが行われている．

本研究では，UMLのパッケージ図の生成を，グラフ文法に基づいて実現することを目的とする．グラフ文法を用いて対象となる図の仕様を記述し，図式の自動処理を行うための枠組みを提案する．

2 準備

2.1 グラフ文法

定義 2.1. ([3])edNCEグラフ文法 (*edNCE graph grammar*) は，6項組 $GG = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ である．ここで，Σ がノードラベルの有限集合，$\Delta \subseteq \Sigma$ が終端ノードラベルの有限集合，Γ がエッジラベルの有限集合，$\Omega \subseteq \Gamma$ が最終エッジラベルの有限集合，P が生成規則の有限集合，そして $S \in \Sigma - \Delta$ が初期非終端である．生成規則は，$X \to (D, C)$ で表される．ここで，X は非終端ノードラベル，D は Σ，Γ 上のグラフ，$C \subseteq \Sigma \times \Gamma \times \Gamma \times V_D \times \{in, out\}$ は接続命令の集合である接続関係を示している．組 (D, C) は Σ，Γ 上の埋め込み付きグラフである． □

図1に生成規則とその適用例を示す．

2.2 UML

UML(Unified Modeling Language)は，オブジェクト指向のシステム開発の際に用いる図式を用いた，システムモデリングの表記法である．UMLは構造

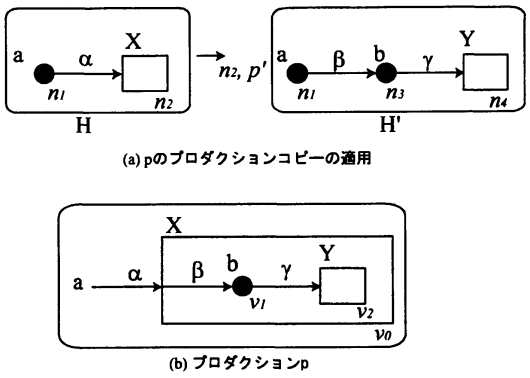

図 1: 生成規則適用の例

図 (structural diagrams) と振る舞い図 (behavioral diagrams) に分けられる．構造図ではモデリング対象の構造を記述し，クラス図，オブジェクト図，パッケージ図等を用いて記述する．また，振る舞い図では，モデリング対象の振る舞いを記述し，ユースケース図，アクティビティ図，状態マシン図等が用いられる．

構造図において，クラス図ではクラス間の静的な関係を記述し，パッケージ図ではクラスをグループ化したパッケージ間の関係やパッケージの入れ子関係をを記述する．

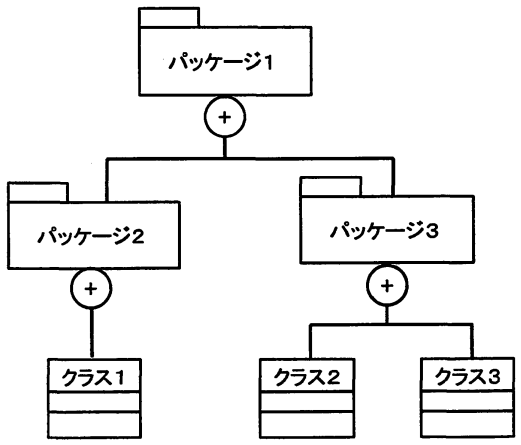

図 2: パッケージ図の例

3 UML パッケージ図に対するグラフ文法

3.1 文法概要

本節では，UML パッケージ図に対するグラフ文法 GGPD (Graph Grammar for Package Diagram) について説明する．

定義 3.1. UML パッケージ図に対するグラフ文法 GGPD は 6 項組 $GGPD = (\Sigma_{PD}, \Delta_{PD}, \Gamma_{PD}, \Omega_{PD}, P_{PD}, S_{PD})$ である．ここで，$\Sigma_{PD} = S, A, T, L, R, M, rop, sp, lep, rip, mip, lec, mic, ric$ はノードラベルの有限集合，$\Delta_{PD} = \{ rop, sp, lep, rip, mip, lec, mic, ric \}$ は終端ノードラベルの有限集合，$\Gamma_{PD} = \{ * \}$，$\Omega_{PD} = \{ * \}$，$P_{PD} = \{ P_1, ..., P_{17} \}$ は生成規則の有限集合，$S_{PD} = \{ S \}$ は初期非終端である． □

GGPD は，パッケージ階層図を生成する．本文法は文脈自由型であり，生成規則は 17 個である．

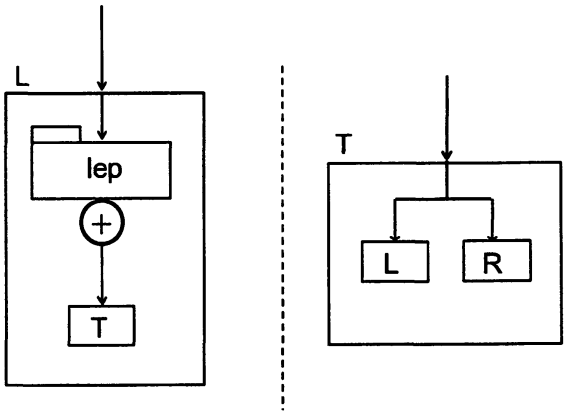

図 3: GGPD の生成規則の例

図 3 に GGPD の生成規則の例を示す．この図では，左側の生成規則では，非終端ノードであるノードラベル L をもつノードに生成規則を適用して，パッケージを表すラベル lep を持つ終端ノードとラベル T を持つ非終端ノードを生成する．

3.2 導出例

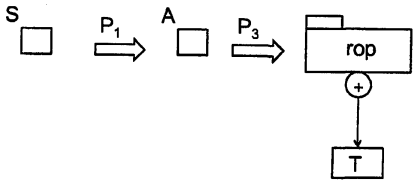

図 4: GGPD の導出の例

図 4 に GGPD の導出の例を示す．この例では，初期非終端ノードであるラベル S を持つ非終端ノードに生成規則 P_1 を適用し，得られたノードラベル A を持つ非終端ノードに，さらに，生成規則 P_3 を適用し，ノードラベル rop をもつ終端ノードと，ノードラベル T を持つ非終端ノードが得られた結果を示している．生成規則を適用した系列は，生成規則の導出木で表すことができる．

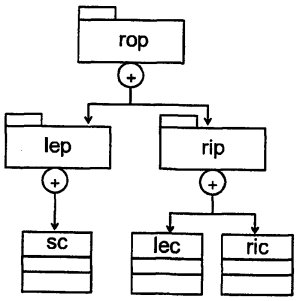

図 5: 導出の結果得られたパッケージ図の例

図 5 に生成規則の適用によって得られたパッケージ図の例を示す．

3.3 Folding / UnFolding

大規模なシステムを対象にパッケージ図を描画することを考慮した場合，扱う図式の規模が大きくなり，図式の視認性に問題が生じることが考えられる．そのため，図式の情報の要約・隠蔽処理が必要である．そこで，文形式の状態で図式を表示することにより，図式の情報隠蔽処理を行う．

図 6 に Folding 前のパッケージ図とその生成規則の導出木の例を，図 7 に Folding 後のパッケージ図とその生成規則の導出木の例を示す．

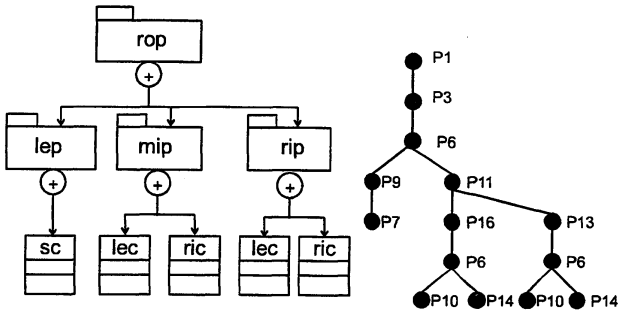

図 6: Folding 前のパッケージ図とその生成規則の導出木

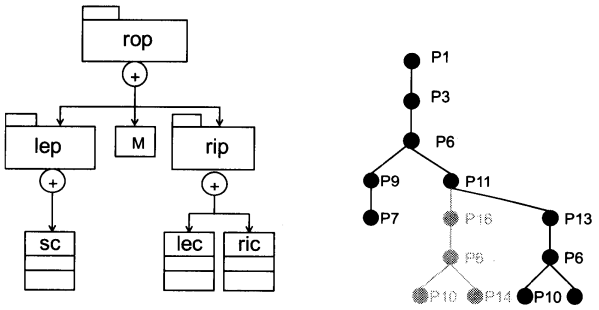

図 7: Folding 後のパッケージ図とその生成規則の導出木

4 UML パッケージ図エディタ

本節では，3 節で述べた文法を基に作成した，UML パッケージ図エディタのプロトタイプについて述べる．本エディタは Java で開発された構文指向エディタである．エディタ画面上に表示されている非終端ノードを選択すると，該当する非終端ノードに適用可能な生成規則の画面が表示される．図 8 は，図中左側のパッケージ図エディタの画面において，ノード ID が 5 のラベル L を持つ非終端ノードをクリッ

クした際に適用可能な生成規則が表示されている状態を示したスクリーンショットである．

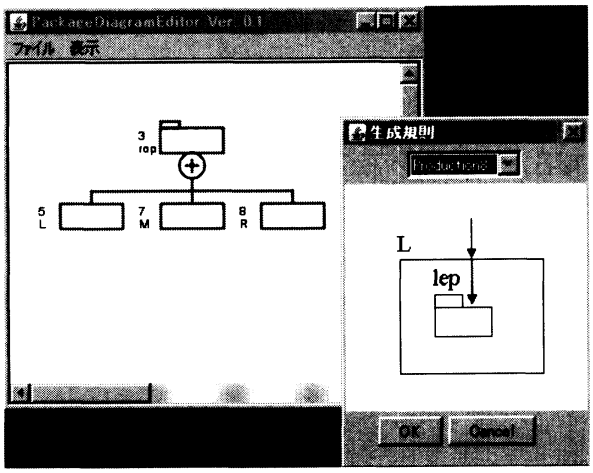

図 8: パッケージ図エディタ生成規則表示画面

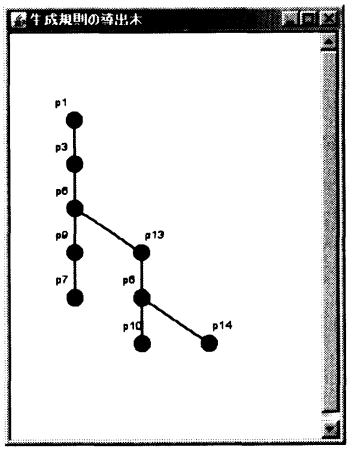

図 9: パッケージ図エディタ生成規則の導出木画面

適用された生成規則は，図 9 で示される，生成規則の導出木で表示することが可能である．

5 まとめ

本稿では，UML のパッケージ図を対象に，パッケージ図の階層構造を生成するグラフ文法の定義を行った．また，定義した文法に対応した構文指向型の図式エディタを試作した．

今後の課題としては，構文解析系の実現が挙げられる．今回開発したエディタでは，文法に従って人手を介して構文に適合した図を生成できるが，任意の入力に対して，図が正しいかどうかの判定を行う事ができない．構文解析系の実現により，任意の入力に対する図式の自動処理が行うことが可能となる．

さらに，ソフトウェアドキュメントの自動生成への応用も行いたいと考えている．

参考文献

[1] 原田賢一. 構造エディタ. 共立出版, 1987.

[2] Yoshihiro Adachi, Youzou Miyadera, Kimio Sugita, Kensei Tsuchida, and Takeo Yaku, A visual programming environment based on graph grammars and tidy graph drawing, *Proceedings of The 20th International Conference on Software Engineering (ICSE '98)*, Vol. 2, pp. 74–79, 1998.

[3] Grzegorz Rozenberg. *Handbook of Graph Grammar and Computing by Graph Transformation Volume 1.* World Scientific Publishing, 1997.

[4] Frank Hermann, Hartmut Ehrig, and Gabriele Taentzer, A typed attributed graph grammar with inheritance for the abstract syntax of uml class and sequence diagrams, *Electron. Notes Theor. Comput. Sci.*, Vol. 211, pp. 261–269, April 2008.

[5] Jun Kong, Kang Zhang, Jing Dong, and Dianxiang Xu, Specifying behavioral semantics of uml diagrams through graph transformations, *J. Syst. Softw.*, Vol. 82, pp. 292–306, February 2009.