3K-9

拡張可能なドメイン特化型言語のエディタの生成手法

市　川　　寛† 佐々木　晃‡

法政大学大学院情報科学研究科† 法政大学情報科学部‡

1. はじめに

本発表ではドメイン特化型言語(DSL)のためのエディタの自動生成手法を提案する．対象とするエディタは，木文法によって定義される抽象構文木の編集を可能とするビジュアルエディタである。本手法では，エディタ作成を支援するための「デザイナ」を用いる．本デザイナでは，言語及びエディタの拡張を GUI によって行うことが可能である．ここでの拡張とは，ドメイン特化型言語の新たな構文を追加することであり，木文法による抽象構文木の導出規則を再定義可能にする仕組みを提供することによって実現される．本発表では提案するデザイナのプロトタイプ実装を行うとともに，ドメイン特化型言語 SOARS[4]に対するエディタの一部を作成した．ドメイン特化型言語[2][3]は問題を解決するプログラム(アプリケーション)をその問題領域に適したモデルの記述方法で表現するための言語である．そのため，汎用言語によるプログラミングの技術を持たないその領域の専門家がドメイン特化型言語を用いることで問題を容易に解決できるようになる可能性も秘めている．しかしドメイン特化型言語の恩恵は無償ではなく，その設計，開発，保守のコストも計算にいれなくてはならない．そこで従来の研究[1]では開発環境の 1 つであるエディタの自動生成を行うことで開発コストの削減を目指した．

本研究では新たにドメイン特化型言語の言語拡張機能とインターフェース設計機能を提案した．ここでの言語拡張とはドメイン特化型言語が本来対象とする問題領域を更に特化させた言語を構築することであり，構築した言語によって，より具象性の高いプログラムがより少ない編集アクションで作成できるようになる．さらに，インターフェース設計機能によってユーザーによる GUI の自由なカスタマイズを可能とし，これにより柔軟なエディタの生成が可能となった．

2. エディタの概要

本エディタは GUI によって抽象構文木を編集するものを指す．そのため，ドメイン特化型言語に対する抽象構文木の導出のための構文規則が必要となる．この構文を木文法と呼び，以下はその一例である．

```
%start AddressBook;
AddressBook => @ADDRESSBOOK Contact*;
Contact => @CONTACT name corporate MailAddress*
                                    PhoneNumber*;
MailAddress => @MAIL text Attribute;
PhoneNumber => @PHONE number Attribute;
Attribute=>@HOME home|@OFFICEoffice|@OTHER other;
```

A Method of Generating Editors for Extensible Domain Specific Languages

†Hiroshi Ichikawa, Graduate School of Computer and Information Sciences, Hosei Universitry

‡Akira Sasaki, Faculty of Computer and Information Sciences, Hosei Universitry

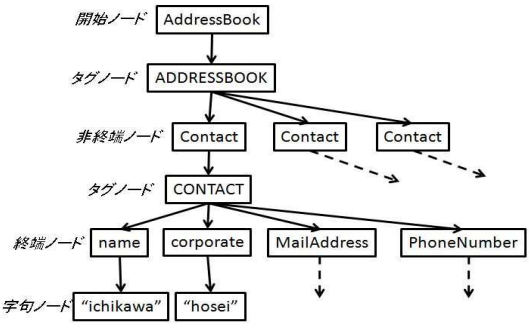

図 1. 木文法から生成される抽象構文木

図 1 は先の木文法によって生成される抽象構文木の一部とノードの種類である．GUI による編集オペレーションは各ノードの導出及び削除である．ノードの導出は木文法に従い，正しく行われるように GUI によって制御される．このため必ず正しい抽象構文木が生成され，字句解析，構文解析は必要なくなり字句エラー，構文エラーは発生しなくなる．抽象構文木の生成はルートからトップダウンで行われ，抽象構文木の葉が全て終端ノードか字句ノードになったらプログラムの完成となる．図 2 は本フレームワークの全体図である．

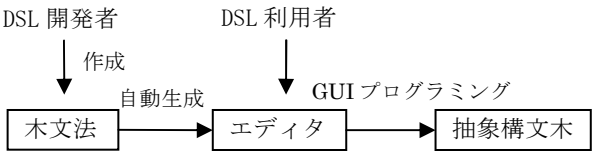

図 2. 全体図

3. デザイナ

デザイナとは自動生成されるエディタのインターフェースの設計と，入力された木文法に対応するドメイン特化型言語の拡張を可能にする GUI ツールである．

3.1 言語拡張機能

言語拡張機能とは木文法に新たな構文を追加し，木文法に対応するドメイン特化型言語が対象とする問題領域を更に特化させる機能のことである．言い換えると，木文法による抽象構文木の生成パターンに制限を加えることに当たる．この言語拡張機能によって木文法の再定義が可能な箇所は 1 つのシンボルからの導出が複数パターン考えられる箇所のみで，木文法においては以下の 4 パターンに一般化できる．

表 1. 木文法で再定義可能な導出箇所

①	1 行に”|”によって複数のタグが存在する場合のタグノードの導出
②	”*”付き終端記号に対応する終端ノードの導出
③	”*”付き非終端記号に対応する非終端ノードの導出
④	字句ノードの導出

表 1 の 4 つの導出箇所では全て導出パターンが複数になる可能性を含んでいる．そのため導出パターンを 1 つでも減らすような構文を再定義すると，生成され得る抽象構文木のパターンが減るため，言語を特化する形になる．

```
Attribute => @HOME home | @OFFICE office
                        | @OTHER other;
```

上の木文法は表 1 におけるケース①に相当する．”|”によって 1 行に 3 つのタグが存在し，タグノードを導出する際には，いずれかのタグ 1 つを選択する必要がある．この時に例えば，導出するタグノードを OFFICE ノード 1 つに固定した場合，導出可能なパターンは 1 つになる．これに合わせた木文法の再定義が以下の木文法であり，これは言語拡張に繋がる．

```
Attribute => @OFFICE office;
```

次に”*”による終端ノード，非終端ノードの導出は 0 個以上の複数のノードの導出が可能であるが，この導出可能個数を制限することで言語拡張が実現する．更に，字句ノードは文字列を値として持つが，その値はデフォルトでは任意であるため，字句ノードに特定の値を持たせることで言語拡張が実現する．実際のデザイナでは，上記，各種言語拡張が可能となるパラメータを入力可能な GUI を備える．

3.2 インターフェースの設計機能

エディタのユーザーが行う抽象構文木の編集アクションは一意に決まらないノードの導出を GUI 操作によって導出するノードを選択し決定すること，及び字句ノードの値を決定することととらえることができる．そのような GUI を導出ウィジェットと呼ぶ．一意に決まらない導出とは 3.1 節の表 1 で示した構造を持つノード部分で発生する．本機能はその際に使用する導出ウィジェットを複数用意し，ユーザーが GUI によって自由に選択できるようにする．

エディタ画面は基本的に複数の導出ウィジェットの集合であるが，特定のウィジェットの集合の占めるエディタ上での面積が大きければタブによって切り替えて表示する方式が有効であると考えられる．しかし占める面積が小さければ 1 つのウィンドウに一括表示し，一覧性を高めるのも有効であり，このような表示方法の切り替え機能も持つ．

4. SOARS シミュレーション言語のためのエディタの生成

本フレームワークを Java Swing GUI ライブラリを持つ Java 言語で実装した．さらに，実装したエディタでドメイン特化型言語の一つであるシミュレーション言語 SOARS のエージェント編集に関する GUI エディタを生成する．SOARS には標準で Visual Shell という GUI エディタが備わっている．本実験では，エージェントの編集パラメータに関する木文法を定義し，この木文法から Visual Shell におけるエージェント編集画面と同等のインターフェースの作成を試みた．図 3 は生成したエディタである．

5. 考察

本研究での言語拡張とは，木文法から生成される抽象構文木のパターンを減らすことで，木文法に対応するドメイン特化型言語が対象とする問題領域をさらに特化させることである．これとは逆に，生成される抽象構文木のパターンを増やすことで問題領域を汎化させることも考えられる．

問題領域を特化させた場合に生成され得る複数の抽象構文木は，元の木文法から生成され得る複数の抽象構文木の部分集合であるため，元のドメイン特化型言語に対するバックエンドを用意さえすれば，そのバックエンドは問題領域を特化させたドメイン特化型言語にも流用可能である．一方，元の言語の構文パターンを増やすことで，生成され得る抽象構文木の生成パターンが増えた場合には元の言語に対するバックエンドはそのまま流用することが出来ない．しかし，新たに加えられた構文に対応する部分に集中して，バックエンドに適切な拡張を施すことで，拡張した言語に対するバックエンドを比較的容易に実装できると考えられる．

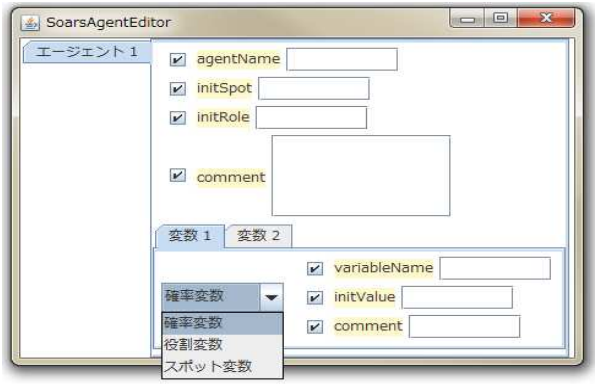

図 3. 生成した SOARS に対するエディタ

6. おわりに

本稿では拡張可能なドメイン特化型言語に対するエディタの生成手法について述べた．本手法ではエディタに言語拡張機能とインターフェース設計機能を持つ「デザイナ」を用いることで元のドメイン特化型言語の問題領域を更に特化させた言語のエディタの生成が可能となり，より少ない抽象構文木の編集アクションで特化されたコードの生成が可能となった．更にインターフェース設計機能によって，より柔軟な表現を持つエディタの生成が可能となった．

謝辞

本研究の一部は科研費(課題番号 21700041)の助成を受けたものである．

文献

[1] 市川寛，”ドメイン特化型言語に対するエディタの生成手法”，“法政大学情報科学部卒業論文”， 2009.

[2] 佐々木晃，須賀康行，”ドメイン特化型言語に対するエディタの自動生成”， 情報処理学会プログラミング研究会，March2008.

[3] 佐々木晃，市川寛，田沼英樹，”GUI コンポーネントに基づく視覚的言語に対するエディタの自動生成”，” 情報処理学会プログラミング研究会”，2010.

[4] 田沼英樹，出口弘，“エージェントベース社会シミュレーション言語 SOARS の開発”，pp.2415-2422，(社)電子情報通信学会，2007.