ソフトウェア工学 119－5
（1998．5．15）

あるキーシーケンス駆動型ソフトウェアの自動生成について

冨岡 学[1] 吉田 勝彦[2] 原田 等[1]

1 同志社大学工学部知識工学科 2 システムセンター・ナノ

ソフトウェアの生産性向上と信頼性向上を目指し，電子式キャッシュレジスタ（ECR）のアプリケーションプログラムを対象としたソフトウェア自動生成ツールの開発を行っている．

本稿は，ソフトウェア部品をファイルで管理し，ソフトウェアの自動生成を行うツールの概要と，状態遷移とキーシーケンスに駆動されるソフトウェアモジュールを，形式的に記述した代数記述式から，キーシーケンス処理部のソフトウェアを自動生成する方法について報告する．

A Study for Automatic Generation of a Key-Sequence driven Software

Manabu TOMIOKA[1] Katsuhiko YOSHIDA[2] Hitoshi HARADA[1]

1 Dept. KE and CS, Doshisha University 2 System Center NANO

A tool, that generates automatically source code for the application software of Electric Cash Register (ECR), is developed to improve productivity and reliability for the software development.

This paper describes an outline of the automotic code generation tool that manages the parts of software, and expresses automatic code generation method for the application software of ECR. To generate the code automatically, the specification ,which describes the state transitions and the parts of key-sequence driven software formally, is used in this method.

1 はじめに

我々は電子式キャッシュレジスタ（ECR）のアプリケーションソフトウェア（以下ECRソフトウェアと略す）を対象にして，キーシーケンス駆動型ソフトウェアの生産性向上と信頼性向上を目指し活動している．

現在ECRソフトウェアの開発は，C言語により行われているが，機能が複雑に絡んでいて，局所的な変更がECRソフトウェア全体に影響が及び，ECRソフトウェアに精通した技術者でないと変更は難しい．この問題を解決するために，C++言語を用いて独立性の高いソフトウェア部品作りを行おうとしている．また，このソフトウェア部品を管理して，ソフトウェアの自動生成を行うツールがあれば，開発者を大いに支援できるものと考える．

そこで我々は現在ECRソフトウェアの部品化と，部品を管理しソフトウェア自動生成を行うツールを開発している．

本稿では2章で，ECRソフトウェアの構成とその部品化について，3章で，部品を管理しソフトウェアの自動生成を行うツールの概要について，4章で，ソフトウェアの自動生成法について説明し，5章はまとめとなる．

2 ECRソフトウェアの構成とその部品化

2.1 ソフトウェアの構成

ECRのソフトウェアは図1に示すような構成をしている．

対象としているECRソフトウェアの部分は，図1上方のアプリケーション部で，4つの部分に大きく分けられる．

- キーシーケンス処理部

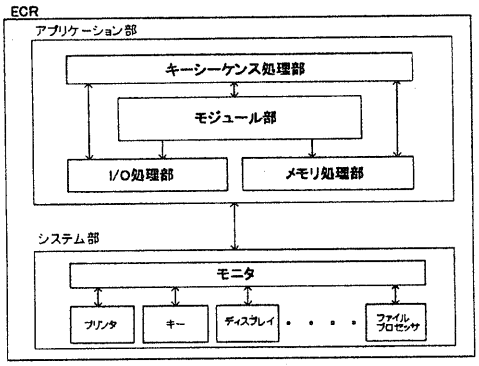

図 1: ECRソフトウェアの構成

ECRのキーイベントを処理して,ECRソフトウェアの状態遷移の処理とモジュール部を駆動する部分である．

- モジュール部

キーイベントに対して実行されるモジュールが存在する部分である．

- I/O処理部およびメモリ処理部

キーシーケンス処理部とモジュール部に呼び出されるライブラリである．

その内I/O処理部およびメモリ処理部は，ハードウェアやOSの変更がないと変更されない部分である．一方，ECRの製品シリーズあるいはECRのユーザ要求により，キーシーケンス処理部およびモジュール部はカスタマイズがよく行われるところである．そこでこの2つの部分を，部品化と自動生成ツールにより開発者を支援する部分とした．

2.2 部品化について

部品化について考える．ただし部品化はC++言語により行うものとする．まず，モジュール部の部品化について考えてみる．

ECRソフトウェアは，状態によって同一のキーイベントに対しても，実行される機能が

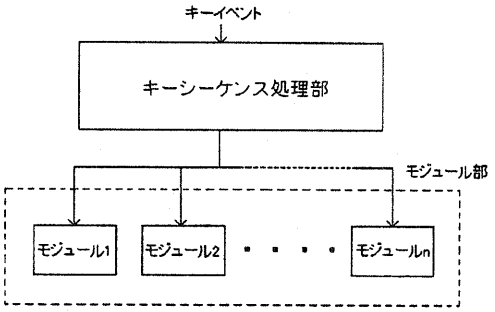

図 2: キーシーケンス処理部とモジュール部の関係

違ってくるが，現状では図 2で示すキーシーケンス処理部に呼び出される個々のモジュールは，状態によらずあるキーイベントに対応した処理がすべて記述されたものである．これでは，あるひとつのモジュール中に，ECR ソフトウェアの機能（「商品を登録する」とか「登録を訂正する」といったもの）が複数個存在することになり，モジュールの理解が難しくなる．この問題を解決するためには，モジュール部に含まれる個々のモジュールが ECR ソフトウェアの機能に一対一で対応すれば良いのではと考えた．つまり ECR ソフトウェアの機能処理の単位で部品化するということである．このように部品化を行えば，開発者にとって部品の内容がはっきりと分かりやすいし，本研究では ECR ソフトウェアの機能の単位で仕様を分けて考えているので，都合が良いのである．また，部品の粒度は，モジュールクラス（モジュール部をクラスと考えたもの）のメソッドの大きさとする．

次に，キーシーケンス処理部の部品化について考える．キーシーケンス処理部は，キーイベントに対して ECR ソフトウェアの状態遷移およびモジュールの呼び出しを行うところで，完結した内容であるので，キーシーケンス処理部全体を部品と考えることにする．またこの部品，キーシーケンス処理部の設計は，Gamma らのデザインパターン [1] から State パターンを適用することにした．ECR ソフトウェアの機能は ECR ソフトウェアの状態に依存し，同一のキーイベントに対して異なった ECR ソフトウェアの機能を実行しなければならない．このことが，オブジェクトの振る舞いが状態に依存し，オブジェクトの振る舞いがその状態により変えなければならない場合は利用することができるという，State パターンの適用可能性に合致したし，その他の ECR ソフトウェアの部分にもデザインパターンを適用して，ECR ソフトウェアのフレームワーク構築することを視野に入れているため，State パターンの適用を決めた．その他に，このキーシーケンス処理部に対する部品は，部品設計の概念だけで実体は 4 章で述べる方法で自動生成することにした．理由は 2 つある．1 つ目は，すべての開発機種に適応できる部品を作った場合，機種によっては無駄になる部分が大きくなり，メモリの無駄遣いにつながる．これは使用可能なメモリ空間に制限の大きい組み込みソフトウェアにとっては致命的である．また，機種ごとに開発する場合は非常に手間がかかるということが挙げられる．2 つ目は，これについては 4 章で詳しく述べる，本研究で使用されている代数記述式から，簡単に生成することが出来るということである．

3 ソフトウェア自動生成ツールの概要

我々が現在開発しているソフトウェア自動生成ツールの概念図を図 3に示す．

このツールは，ツールのユーザが ECR ソフトウェアに必要な機能を，機能の一覧から選択することで，ジェネレータがファイルとして管理している部品を取り出し，仕様書，ソースコード，リソース，テストキーシーケンスデータを自動生成するものである．

この図で自動生成するものの内，本稿で述

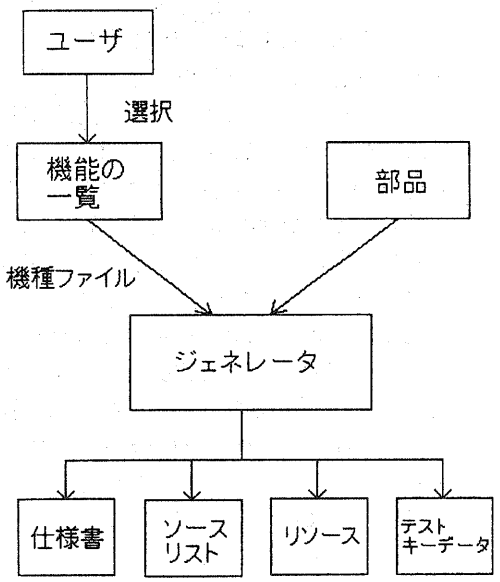

図 3: ツールの概念図

べるのはソースコードについてである．また，テストキーシーケンスデータの自動生成については文献 [2] に，仕様書，リソースの自動生成の部分は開発中である．

3.1 部品の管理方法

ソフトウェア自動生成ツールが管理している部品には次のようなものがある．

1. 仕様

   個々の ECR ソフトウェアの機能に対する仕様を自然言語で記述したもの．

2. キーシーケンスの代数記述式

   このキーシーケンスの代数記述式は，キーシーケンスによる状態遷移と，駆動されるモジュールを記述したもので，新田ら [3] に提案され本研究独自のものである．本論で提案するソフトウェア自動生成において，自動生成の元になるものである．詳しくは，次章で述べることにする．

3. ソースコード

   キーシーケンス処理部により駆動されるモジュールのソースコードで，自動生成のためのソフトウェア部品である．

4. リソース

   プログラムの実行時に必要な印字データ，あるいは ECR 制御設定等のリソースである．特に ECR のディスプレイ表示およびレシートへの印字フォーマットに関しては，呉ら [4] によりプリントフォーマットが提案されている．

5. 部品情報

   部品合成に必要な関数名などの情報と，ユーザの選んだ機能に矛盾がないかどうかを，ジェネレータにかける前にチェックするための情報である．例えば，同時に存在しなければならない機能が存在しているかどうかのチェックが挙げられる．

これらの部品は，ECR ソフトウェアの機能ごとに用意されており，その管理は ECR ソフトウェアの機能に付けられたキーワードに，部品各々の拡張子を付けたファイル名で行う．キーワードは，ECR ソフトウェアの機能が図 4のようなツリーに分類できることを利用してつけている．

このツリーの一番下のノードが ECR ソフトウェアの機能を表しており，キーワードはそのノードにたどり着くまでのノードに付けられている単語を，つなぎあわせたものとなっている．例えば，図 4で丸で囲ったところは DeptRegSingleRegister というふうにつけられる．ここで，キーワードの大文字になっているところは，ノードの区切りを表している．

また，ECR に新たな機能を追加する時は，５つの部品すべてを作成し，キーワードをもとにしたファイルで管理することになる．

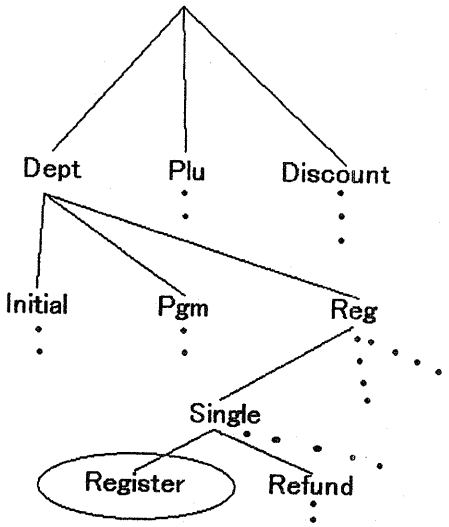

図 4: ECR ソフトウェアの機能分類

3.2 ツールの使用例

本ツールの使用例を図 5に示す.

ユーザは，図左側「ECR ソフトウェア機能一覧」から必要な ECR ソフトウェアの機能を選び真ん中の追加ボタンを押すことで，右側の「開発機種機能一覧」へ必要な機能を追加して行く．また，説明ボタンは ECR ソフトウェア機能の仕様を表示し，削除ボタンは「開発機種機能一覧」から不必要な機能を削除する.

ユーザは開発している機種に必要な ECR ソフトウェアの機能を選択し終わり，その選択した機能のデータを機種ファイルに保存すると，ジェネレータがはたらき自動生成が行われる．ここで機種ファイルとは，選択した ECR ソフトウェアの機能を保存したもので，キーワードが保存されている.

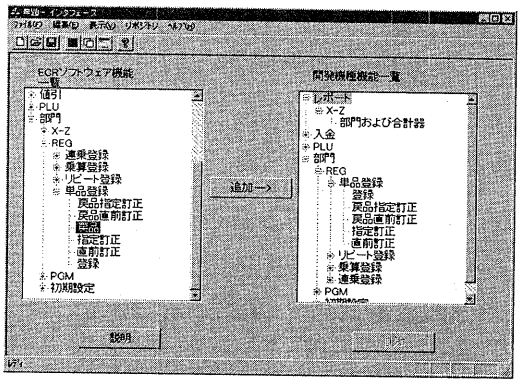

図 5: ツールの使用例

4 ソフトウェアの自動生成について

2 章で述べたように，自動生成する部分はキーシーケンス処理部と，それによって呼び出されるモジュール部である．これらは代数記述式を処理し，ソフトウェアの部品を合成することで自動生成される.

4.1 代数記述式について

本研究独自の代数記述式の基本型は，次のように記述される.

[入力キーレベル](キー名：モジュール名)[出力キーレベル]

ここで,

- レベル：ECR の状態を表す
- 入力キーレベル：キー入力がなされたレベル
- 出力キーレベル：キー名で示されるキーが押された後入るレベル
- キー名：入力キーレベルにおいて，入力可能なキーの名前

- モジュール名：キー名で示されるキーが入力された時実行されるモジュール名

ということを意味している．

```
・[KREQ_NL](KEY_DEPT : REQ_Dept)[KREQ_RL]

・[KREQ_NL](KEY_LVOID : REQ_Last_Void)[KREQ_RL]

・[KREQ_RL](KEY_PVOID : REQ_Past_Void)[KREQ_PVD](KEY_DEPT : REQ_Dept)
    [KREQ_RL]

・[KREQ_NL](KEY_RF : REQ_Refund)[KREQ_NML](KEY_DEPT : REQ_Dept)
    [KREQ_RL](KEY_LVOID : REQ_Last_Void)[KREQ_RL]
```

図 6: 代数記述式の例

実際のECRソフトウェアの仕様から作った代数記述式の例は図6のようなもので代数記述の基本型が連なったものとなっていて，基本型における出力レベルが，次のキーに対する入力レベルとなって連なっている．

4.2 自動生成の方法

今回自動生成することにした部分，すなわちキーシーケンス処理部と，キーシーケンスにより駆動されるモジュール部の設計は，オブジェクト指向により行った．特にキーシーケンス処理部は，Gammaらのデザインパターンから State パターンを選び適用した．

図7に自動生成する部分のクラス図を示す．

それでは，自動生成の手順を述べることにする．ただし，生成されるコードはC++言語で記述されている．

1. 部品の取り出し

   ツールにより選択された機能に対する，キーシーケンスの代数記述式とソースコード，部品情報を取り出してくる．これらは，機種ファイルに保存されているキーワードをたよりにして取り出してくる．

2. 代数記述式の処理

   キーシーケンスの代数記述式を処理し，入力レベルごとにキー名とモジュール名，

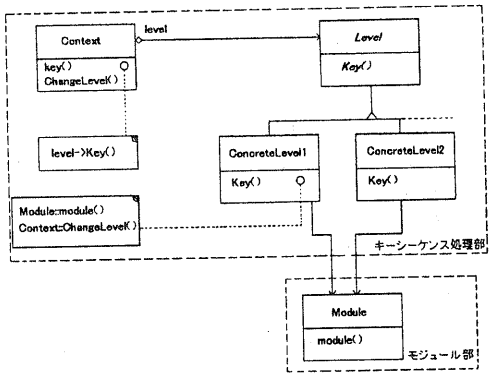

図 7: 自動生成する部分のクラス図

出力レベルの関係を表したテーブル（図8）を作る．また，使用されている入力レベル，キー名のリストも作る．

3. モジュール部の自動生成

   モジュール部の自動生成は，モジュール部に対するクラスを作り，モジュールのソースコードがクラスのメソッドとして機能するようにする．具体的には，クラスの宣言部には部品情報からわかるモジュールの関数名を，メソッドの本体にはモジュールのソースコードを，ジェネレータが生成するモジュール部のソースコードとして書き出すことによって行う．

4. キーシーケンス処理部の自動生成

   入力レベルの名前はConcreteLevelクラスの名前に，キー名は，Levelクラスと Contextクラスのメソッド名にそれぞれマッピングする．入力レベルごとに作った，キー名，モジュール名，出力レベルの関係テーブル（図8）から，入力レベルの名前で作ったConcreteLevelクラスのメソッドの中身にマッピングされる（図9）．このようにしてジェネレータがキーシーケンス処理部のソースコードを自動生成する．

入力レベル
KREG_RL

キー名	モジュール名	出力レベル
KEY_DEPT	CREG:: Single_Register	KREG_RL
KEY_RF	CREG:: Single_Refund	KREG_RML
KEY_PVOID	CREG:: Single_Past_Void	KREG_PVL
⋮	⋮	⋮

図 8: 代数記述式を処理し作成したテーブル例

以上が自動生成の手順で，モジュール部とキーシーケンス処理部の２つのソースコードが自動生成される．ただし生成に必要な，キーシーケンス処理部とモジュール部のクラスのひな形，生成する手順などはジェネレータのプログラムに埋め込んである．

また実行可能なECRソフトウェアは，あらかじめ用意しておくキーイベントをキーシーケンス処理部に渡す部分のソースコード，ライブラリのソースコード，これらを駆動するメインのソースコードと，自動生成した２つのソースコードから作成できる．

5 まとめ

本稿では，開発しているソフトウェア自動生成ツールの概要と，ソフトウェアの自動生成法について報告した．開発したツールは部品のファイルによる管理と，独自な代数記述式の簡単な処理により自動生成を行おうとしたものである．本ツールの実現は簡単に行えると考えるが，開発者を支援するツールとしては弱いと考える．開発者は必要な部品を選ぶだけではなく，部品の改造や，新規に部品を開発するため，部品間の依存関係などの情報が必要となってくるからである．今後は，

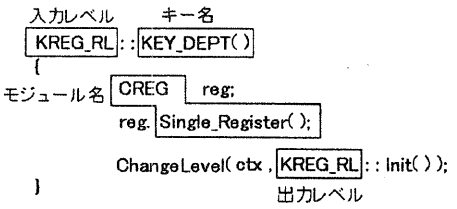

図 9: 自動生成されたConcreteLevelクラスメソッドの例

依存関係など情報を記述する形式的な記法を検討し，ツールの機能を高めようと考える．

また報告した手法を，実験用にパーソナルコンピュータ上で動作するよう簡略化したECRソフトウェアの仕様に対し適用したところ，適用することができたが，ECRハードウェア上で動作するものへの適用は確かめられていない．この点は適用後にまた報告する．

参考文献

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Design Patterns Elements of Reusable Object-Oriented Software, Addison-Wesley,(1995)(本位田 真一，吉田 和樹監訳: デザインパターン, ソフトバンク (1995)).

[2] 冨岡他:あるキーシーケンス駆動型ソフトウェアに対するテストキーシーケンスの自動生成, 情報処理学会第 52 回全国大会 (1996).

[3] 新田他：キーシーケンス処理プログラムの自動生成, 情報処理学会第 44 回全国大会 (1992).

[4] 呉他：組み込み型ソフトウェア用の印字フォーマットジェネレータの構成, 情報処理学会第 54 回全国大会 (1997).