モデル駆動型アーキテクチャを用いたアスペクト指向ソフトウェアアーキテクチャからのコード生成に関する研究
～ コード生成におけるプラットフォーム非依存モデルについて ～

2004MT071 二宮 剛史
指導教員　蜂巣 吉成

1　はじめに

ソフトウェアアーキテクチャに基づいて開発をおこなう際に現れる規則的なプログラムコードを自動生成し，プログラムコード作成の労力を削減することがおこなわれている．プログラムコード自動生成の枠組としてモデル駆動型アーキテクチャ [3] (以下，MDA) がある．MDA はプラットフォーム非依存モデル (以下，PIM) を作成し，プラットフォーム依存モデル (以下，PSM) に変換することで様々なプラットフォームにおけるプログラムコード作成を可能にする．プラットフォームは OS やプログラミング言語などのことである．

本研究室では組込みシステムのためのアスペクト指向ソフトウェアアーキテクチャスタイル [2](以下，E-AoSAS++) が提案されている．E-AoSAS++ は組込みシステムを並行状態遷移機械の集合として規定する．E-AoSAS++ に基づくソフトウェア開発では並行処理などの関心事 (コンサーン) をアスペクトとして適切なモジュール化をおこなうので，結果として得られるプロダクトの再利用性が高くなるという利点がある．

E-AoSAS++ の問題点として，E-AoSAS++ に基づくソフトウェア開発の開発環境が未整備である．E-AoSAS++ に基づくソフトウェアアーキテクチャからプログラムコードを作成する際に規則的な記述があらわれるが，プログラマは全て作成しなければならない．また，組込みソフトウェアは様々なプラットフォームにおいて実現されるので，プラットフォームごとにプログラムコードの記述をおこなわなければいけない．

本研究の目的は，E-AoSAS++ に基づくソフトウェアアーキテクチャからプログラムコードを自動生成するツールを作成することである．ツールを作成することで，アーキテクチャの規則性から定型的なプログラムコードが作成できる．また，ツールを MDA に基づいて作成することで，多様なプラットフォームへの対応が可能となる．ツールにより E-AoSAS++ に基づくソフトウェア開発環境を整備することができる．

本研究では作成するプログラムコード自動生成ツールにおける PIM の設計と作成方法の提案をおこなった．

2　E-AoSAS++

E-AoSAS++ は組込みシステムのソフトウェアアーキテクチャスタイルである．E-AoSAS++ では，組込みシステムを並行状態遷移機械 (以下, CSTM) の集合として規定する．各 CSTM は互いにメッセージを送り協調動作することで組込みシステムの機能を実現する．CSTM の構成を**図 1** に示す．並行処理アスペクトでは CSTM

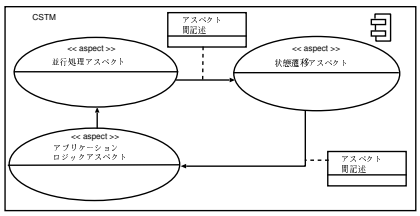

図 1　CSTM の構成

に通知されるイベント管理や並行動作を扱う．状態遷移アスペクトでは CSTM の状態の切替えを扱う．アプリケーションロジックアスペクトでは状態の切替えにともなうアクションを扱う．アスペクト間記述ではアスペクト間の関連を表現する．

3　プログラムコード自動生成ツールとプラットフォーム非依存モデル

3.1　プログラムコード自動生成ツールの概要

プログラムコード自動生成ツールの概要を**図 2** に示す．CSTM 間の関連がコンポーネント図で記述された静的

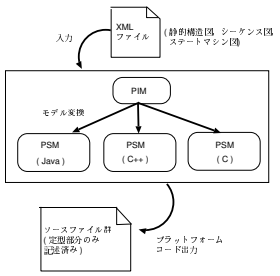

図 2　プログラムコード自動生成ツールの概要

構造図，各 CSTM ごとの状態や遷移にともなう振る舞いが記述されたステートマシン図，シーケンス図を XML 文書形式にして入力する．各図をもとに PIM を作成し，各プラットフォームへの変換論理から PSM を作成する．出力は各プラットフォームコードの定型部分が記述されたソースファイルとする．図に記述される情報をもとに作成できるプログラムコードを定型コードとして扱う．対象プラットフォームは Java，C++，C の各プログラミング言語とする．本研究では入力された図式表現から PIM を作成するまでを扱う．

3.2　プラットフォーム非依存モデルに求められる要件

PIM には CSTM をプログラムコードに変換するのに十分な情報が含まれている必要がある．CSTM は複数のモジュールによって実現される．例として，並行処

理アスペクトを実現するモジュールを**図 3** に示す．並

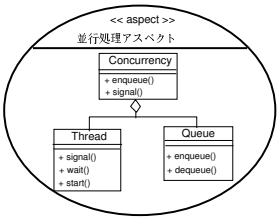

図 3　並行処理アスペクトを実現するモジュール群

行処理アスペクトはイベント管理を扱うモジュールの Queue，並行動作を扱うモジュールの Thread，Queue と Thread を管理するモジュールの Concurrency で実現される．他のアスペクトも同様に複数のモジュールで実現される．CSTM を構成する各アスペクトは CSTM に共通なので入力となる各図には記述されていない．プラットフォームが異なる場合でも CSTM を構成する各アスペクトの実現に必要なモジュールは変わらないので，入力された情報を各モジュールを含んだ表現に変更する必要がある．

3.3　プラットフォーム非依存モデルの設計

プログラムコードで CSTM を実現する際のモジュール構成まで考えた表現の構文規則を考え，構文規則に基づき作成する抽象構文木を PIM とする．構文規則を Interpreter パターン [1] を用いて表現したクラス図を**図 4** に示す．PIM の各ノードには対応するモジュール

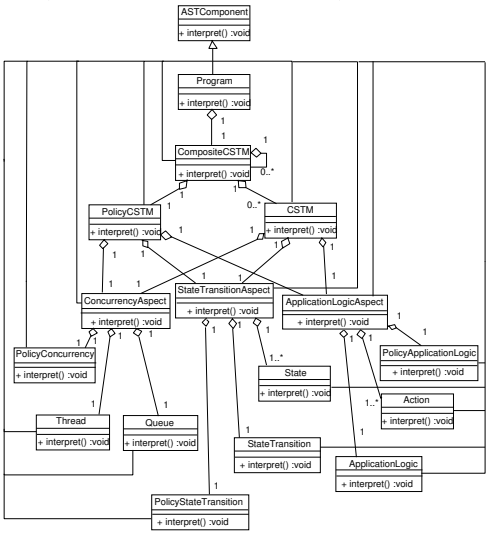

図 4　PIM の表現に使用するクラス図

を実現する際に必要とする情報を持たせる．必要とする情報は図から得られる．例えば並行処理アスペクトを構成するモジュールのひとつである Thread は，プログラムコードで実現する際に図から得られる情報として CSTM 名を使用している．Thread モジュールに対応する PIM の Thread ノードには，CSTM 名を持たせることで定型部分が記述できる．

3.4　コード生成のためのプラットフォーム非依存モデルの作成

PIM 作成手順について述べる．PIM の作成過程の一部を**図 5** に示す．親ノードを作成すると構文規則に従い，

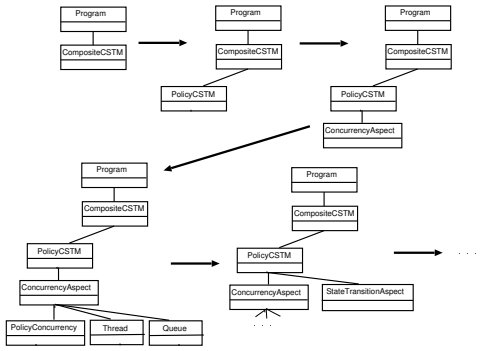

図 5　PIM の作成過程の一部

子ノードのインスタンスを作成していく．同じ子ノードのインスタンスを複数作成する場合は必要な図の情報をもとに作成する．全ての CSTM を構成するモジュールに対応するノードを作成したら PIM が完成する．

4　考察

PIM の妥当性と新たなプラットフォーム追加の考察をおこなう．図をもとに PIM を作成し，各プラットフォームごとの変換論理に基づき PSM に変換し，Java, C++, C の各言語で定型部分が記述されたソースファイルが作成できることを確認した．コード生成をおこなう際に PIM が十分な情報を提供していることから本手法で作成した PIM は妥当であるといえる．PIM が妥当であることから，新たなプラットフォーム追加を考える際は追加するプラットフォームのプログラムコードで各モジュールを作成し，PIM からの変換論理を作成するだけでよい．

5　おわりに

本研究では，E-AoSAS++ に基づくソフトウェアアーキテクチャからプログラムコードを自動生成するツール内における PIM の提案をおこなった．また，PIM の妥当性と新たなプラットフォーム追加時の考察をおこなった．今後の課題として自動生成されるプログラムコードの再利用に関する考察がある．

謝辞

熱心な御指導をいただいた野呂昌満教授，沢田篤史教授，蜂巣吉成講師，有益なアドバイスをいただいた大学院生のみなさまに深く感謝いたします．また，二年間ともに頑張ってきた蜂巣研究室，野呂研究室，沢田研究室のみなさんに感謝します．

参考文献

[1] E. Gamma, R. Helm, R. Johnson, J. Vlissides：Design Patterns Elements of Reusable Object-Oriented Software, Addison Wesley Longman, 1995.

[2] 坂野将秀：組込みソフトウェアのためのアスペクト指向アーキテクチャスタイルの提案，南山大学大学院数理情報研究科 2006 年度修士論文要旨集.

[3] OMG：MDA，http://www.omg.org/mda/.