4V-2

ビジネスプロセスモデルから実行可能なプラットフォームに依存しない UML モデルに変換する手法の提案

波山　智宏† 池田　鈴太郎† 山本　洋平† 石原　純† 高橋　誠† 木村　英明† 片岡　信弘†
東海大学電子情報学部情報メディア学科†

1.はじめに

近年、現状のシステムの改善または新規システムの導入の場面においてビジネスプロセスを分析し把握する事は非常に重要視されている。ビジネスプロセスを可視化しビジネスプロセスモデル（以降 BPM）にすることで分析・把握の精度は上がる。ところが現在、BPM から直接システムを開発する一連の流れは確立されていない。

そこで我々の研究では、モデル駆動型開発を想定し、ARIS を用いてビジネスプロセスを可視化した BPM を利用し実行可能なプラットフォームに依存しないモデル（以降 PIM）を作成することにより、直接開発することを可能にする一連の手法の提案と検証を提供する。

2.ビジネスプロセスモデリング

業務プロセス全体をビジネスプロセスと呼ぶ。システム開発では委託者の要求定義が必要である。そのときに現状の業務プロセスをモデル化することは委託者と共通理解するために有益である。我々の研究ではこのビジネスプロセスをモデル化するために IDS シェアー社の ARIS Toolset を使用する。ARIS はビジネスプロセスを機能ビュー、組織ビュー、データビュー、制御ビューによって記述する。制御ビューはこれらのビューを統合してモデル化する。

eEPC 図は制御ビューを詳細に記述するために使用する。eEPC 図では、イベントとファンクションの２つを主に用いる。ファンクションとはあるオブジェクトに関わる専門的な課題、活動である。ファンクションは、ファンクションを起動させるための何らかのイベントが存在して初めて実行が開始され、機能が要求する目的が新しいイベントとして達成されて初めて終了する。

3.モデル駆動型開発

モデル駆動型開発(MDA)とはモデルを中心として開発を進めていく方法である。中心となるモデルは PIM と PSM の 2 つのモデルから成り立つ。図１に示す。逆に非 MDA 開発ではシステムの厳密なロジック（詳細仕様）は設計モデルで作り込む。

PIM とは Platform Independent Model の略でプラットフォームに依存しないモデルのことである。プラットフォームとは CPU や OS、プログラミング言語などを表している。PSM とは Platform Specific Model の略でプラットフォームに依存したモデルのことである。開発環境、開発言語などに対応したモデルである。PIM から PSM へは、現在ツールを使って自動的に変換することが可能となりつつある。また、PSM 間もブリッジツールによって、変換することが可能である。つまり PIM のモデルを作っておけば、どのようなプラットフォームに対応できる。

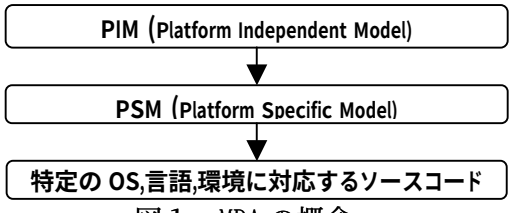

図１　MDA の概念

4.Executable UML

Executable UML は実行可能な PIM を作成する方法のため、要件定義や実装、配置などの方法は提供していない。Executable UML は主に UML のクラス図、ステートチャート図、アクションセマンティックを使用して PIM を作成する。また、PIM の側面を表現するのにユースケース図、コラボレーション図を使用する。

最初に、ユースケースを利用し問題領域を分類し、問題領域をドメインとしてとらえ、それ

A Proposal and Verification of a Technique of Converting the Business Process Model into a Platform Independent Model
†Tomohiro Namiyama,Rintaro Ikeda,Yohei Yamamoto, Jun Ishihara,Makoto Takahashi,Hideaki Kimura, Nobuhiro Kataoka
Tokai University, School of Information Technology and Electronics, Department of Information Media Technology.

ぞれのドメインに対して Executable UML を作成する。各ドメインは独立している。それらのドメインを密接に関係したクラスのかたまりを持った幾つかのサブシステムに分解する。

各クラスのインスタンスはある段階ごとに異なる振る舞いを持つ、ドメインにおけるクラスのライフサイクルをモデル化するためにステートマシンを利用する。ステートマシンは状態を持つ。ドメイン内で何かが起きたときシグナルが発生する。シグナルは１つのイベントは必ず１つのステートマシンによって受け取られる。イベントによって状態変化が起こり、イベントを受けたステートマシンはプロシージャが起こる。プロシージャはアクションによって構成される。プロシージャは新しいクラス間のリンク、新しいオブジェクトを生成、削除を行ったり、クラスの包含範囲を選択したりする。これらによって、実行可能な PIM を作成する事が出来る。

5. BPM から PIM への変換手法

まずビジネスプロセス分析し BPM を作成する。BPM から PIM への変換として本研究では主に eEPC 図を利用する。eEPC 図のファンクションの分解はなるべく細かく、例えば 2 つの動作を 1 つのファンクションに書くべきではない。変換手順は以下のように定義した。

- 一つの eEPC 図をサブシステムの一部として考える。つまり eEPC 図の数だけクラス図が存在するということである。
- システム化を行いたいファンクションを決め、そのファンクションにシステムを関連付ける。これはどのファンクションをシステム化するかを分析する為である。
- システム化を行いたいファンクションにおいて、組織(組織ユニット、役割、要員など)の外部エンティティからの操作や外部エンティティへの出力が必要な場合や外部エンティティをファンクションに関連付ける。
- ファンクションを実行する為に必要なクラス、属性を考え、そのファンクションにクラス、属性を関連付ける。ファンクションの文中の名詞などがクラスや属性の大きな候補になる。
- １つのファンクションは複数のクラスと関連付ける事が出来る事とする。
- 全部のクラスを抽出し終わったら ARIS のモデル生成機能を使用して eEPC 図からクラス図を自動変換する。
- クラス図上で１つのファンクションに複数のクラスが関連付けられている場合、クラス図上のクラス間で関連があるということであるので、クラス図上でそのクラス間を接続する。
- クラス図上で多重度、関連名、ロールを付け加える。
- ライフサイクルをもつクラスのステートマシンを作成する。１つのファンクションに複数のクラスが関連付けられている場合、ステートマシン上であるクラスが他のクラスのオブジェクトを参照するだけか、あるクラスが他のクラスに何らかのシグナルを送信している場合のどちらかである。何らかのシグナルを他のクラスのオブジェクトに送信している場合、それをステートマシン上に記述しそのオブジェクトの状態変化を記述する。
- ファンクション間でクラスの変化がない場合はそのクラスのオブジェクトが自分自身にシグナルを送信しているとし、そのクラスのステートマシン上に記述し状態変化を記述する。
- システム化を行いたい１つのファンクションにおいて、外部エンティティとクラスが存在する場合は外部エンティティがシグナルを送信または受信していると考え、そのクラスのステートマシンに記述する。
- さらにステートマシンに状態変化が必要な場合は記述し、ステートマシンにアクションを記述する。

6.これからの課題

今回の手法に基づいて作成した PIM の検証を数多く行い、変換手法をより正確なものへしていかなければならない。今回提供した変換手法のほぼ全ては手作業であり Executable UML の知識が必要不可欠である。これからはどのファンクションをシステム化するか決めた後は自動的に PIM に変換できることを目指す。

参考文献

[1]ハインリヒ･ザイドルマイヤー著、堀内正博、田中正朗 訳、”ARIS によるビジネスプロセス･モデリング”、ビー･エヌ･エヌ新社,2004
[2]スティーブ J.メラー、マーク J.バルサー著、Executable UML 研究会 訳、”Executable UML”、翔泳社、2003
[3]ANNEKE KLEPPE、JOS WARMER、WIM BAST 著、テクノロジックアート 訳、”MDA(モデル駆動型アーキテクテャ)導入ガイド”、インプレス、2003