産業用制御システムへの OOAD 適用

～eUML／Konesa-RealTime を活用した開発事例紹介～

オムロン株式会社
インダストリアルオートメーションビジネスカンパニー
技術統括センタ　システム技術開発センタ　主事

人見　繁

1．はじめに

近年、FA 分野においてもオブジェクト指向分析・設計（OOAD）を用いた産業用制御システムの開発が試行[1],[2]されつつある。しかしながら、一般的なオブジェクト指向開発手法やモデルの表記法である UML は、ハードウェア制御や並行処理など組込み機器特有の課題や、FA 分野に特徴的な処理が考慮されていない。これらの不足する知見に関しては、個々のプロジェクトで試行錯誤的に補いながら開発を進めている状況である。当社ではこの試行錯誤的な部分（OOAD 導入時の敷居の高さ）を最小化することを目的として、eUML[2],[4]を試験的に導入し、その効果を検証した。

本稿では、eUML と組込み向けの CASE ツールである Konesa - RealTime (以下、KonesaRT)をシーケンス制御トレーニング用検査装置の開発に適用した事例を紹介する。

2．産業用制御システム開発の現状と課題

2．1　開発の現状

FA 分野においても周辺機器のインテリジェント化・IT 化の波がソフトウェア技術者の負担を増大させつつある。本来の目的であるメカ制御に加えて、情報処理やインターネットへの接続などの周辺処理が増えたことが原因である。一般によく言われる「組込み機器開発の複雑化、大規模化」の流れである。従来のような属人的な開発を続けていると、開発の後半に品質が安定しない、工数が予測できないなどの現象に直面する。

このような課題には OOAD が品質向上、開発効率向上の打ち手として期待されている。しかし、FA・組込み分野においては OOAD 導入時の敷居の高さから開発現場に浸透していないのが現状である。

2．2　課題

従来の開発、つまり設計手法を導入していない場合や、属人的な開発を行っている場合に顕著にみられる課題を以下に示す。

(1) 要求仕様分析・設計時における課題
　要求仕様分析が不十分なため、全工程に渡り頻繁な仕様変更が発生し、

小手先での都度対応に追われる傾向がある。また、上流工程を軽視した場合には、小さな改造を行う場合でも「仕様変更がシステムに与える影響範囲が不明確」になるため必要以上に修正、品質確保工数がかかる。

(2) 実装時の課題

詳細設計以降は、担当者の裁量に任されることが多く部品化が進まない、または共通の知的資産として認識されていない。また、ソフトウェア仕様書レベルで品質を確保しても、実装担当者の理解不足、コーディングミスで品質が引き継がれないことがある。

(3) 結合デバッグ時の課題

メカと結合した時点で仕様・ロジックミスが顕在化する。実機を用いた結合デバッグにおいては、ソフトウェアの不具合によりメカ部分を破損することがある。

3. eUML/CASE ツールへの期待と検証ポイント

3．1　期待

eUML への期待は、組込み分野における OOAD 適用のベストプラクティス活用による開発プロジェクトの成功確率向上である。また、CASE ツールへの期待は、ビヘイビアを含めたモデリングによる品質の作りこみ作業の効率化と、その品質の実装への落としこみ（コードの自動生成）の支援である。

3．2　検証ポイント

本試行においては上述した課題の改善と、FA 分野特有の課題を OOAD で克服できるかを検証ポイントとする。本稿では FA 特有の課題として、①様々な運転モードの分析・設計、②インターロックと呼ばれる同期・排他制御機構の設計と事前検証、③プログラムの大半を占める異常系の処理の扱い、の 3 点を取り上げる。

4．開発事例

4．1　装置の概要と開発諸条件

本試行で適用した**図 1** の概観を持つ擬似検査装置の概要と開発諸条件を以下に示す。

- システム規模：入力 40 点、出力 38 点
- 実行環境：SH4、μITRON3.0
- 開発言語：ANSI 準拠 C++言語
- 被験者：入社 3 年目。入社時よりソフトウェア開発に従事
- 当初工数見積もり 3.5 人月

図 1　擬似検査装置の概観

4．2　モデリング

ここでは、eUML 開発プロセスのオーバービュー[4],[5]中にある作業において、分析過程での成果物を一部紹介する。本稿では、ユースケース分析、ドメイン定義、システムビヘイビア分析、オブジェクト構造分析について、「①運転モード」、「②インターロック機構」、「③異常処理の扱い」を取り上げて、その分析モデルを簡単に示す。

4．2．1　ユースケース分析

ユースケース分析で得たユースケース図を図 2 に示す。本装置は大きく分けて自動運転と手動運転とに分れる。どちらの運転も最終的にはユースケースの右側で表現したワークの準備、供給、検査、排出のユースケースを用いる。

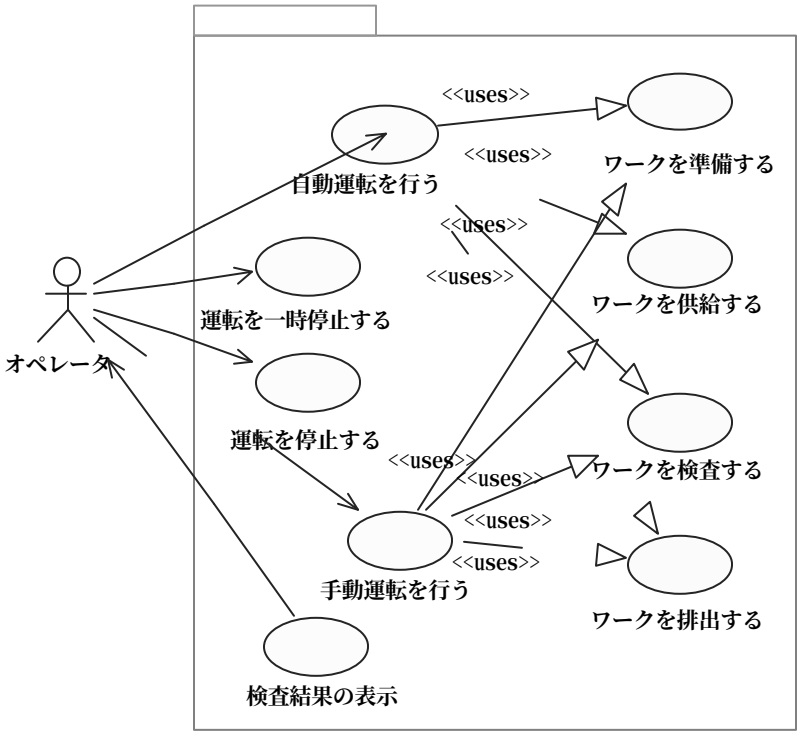

図 2　ユースケース図

4．2．2　ドメイン定義

ドメイン定義は、eUML で典型的なパターンとしてあらかじめ図 3 の左上の構造が推奨されているのでそのまま活用した。次にメカニズムドメイン内のパッケージ分割であるが、ロードアームなどの機能ユニット毎にパッケージを分割したシンプルで理解しやすい構成にした（図 3 右下）。これらのパッケージは独立して動作することを前提としており、自動運転、手動運転ともに同じパッケージを使用する。自動運転においては、上位のコントローラパッケージが全体を統括する構成とした。

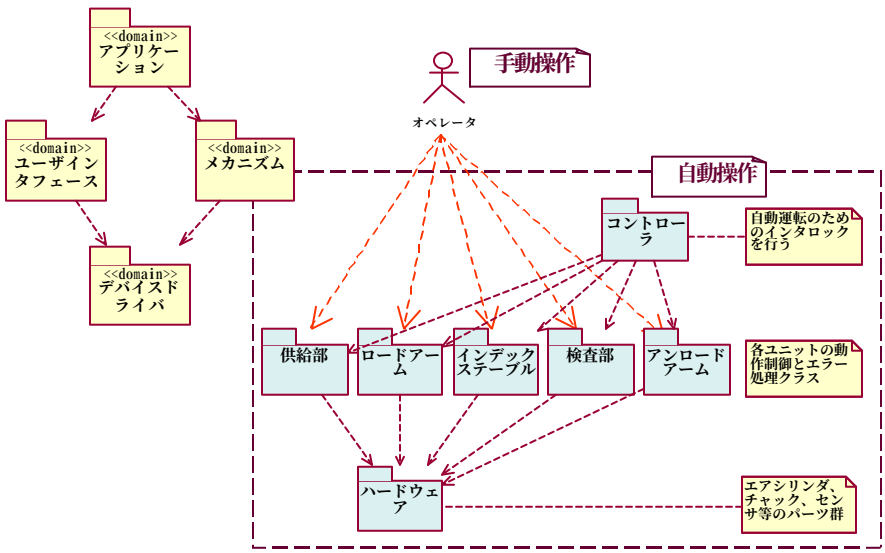

図 3　ドメイン構造とパッケージ構成

4．2．3　システムビヘイビア分析

システムビヘイビア分析においては、装置全体の動作を統括する前述のコントローラの内部処理を決定することになる。

本試行では、コントローラが司る各機能ユニット間のインターロックをステートチャートで表現した（**図４**）。ロードアームとインデックステーブル間の関連は、点線（機能ユニットの区切り）上にある同期状態の記号により結合されている。このように各機能ユニットの単体動作、連携をステートチャートでモデリングすることにより、後工程にて状態遷移表によるヌケ、モレの確認とシミュレーションによるロジックミスの確認が行える。また、ロジック検証済みのモデルは自動コード生成により実機上のプログラムとして実行できる。

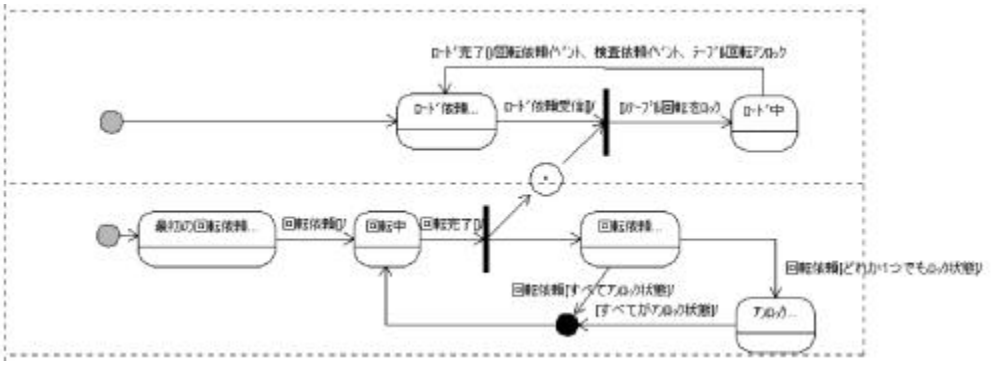

図４　インターロックの表記例（抜粋）

4．2．4　オブジェクト構造分析

オブジェクト構造分析においては、大まかなソフトウェア部品を抽出する。ここではメカニズムドメインに属するロードアーム内のソフトウェア部品の構成を部品の特性に合わせて階層化した（**図５**）。また、異常処理については、正常系の処理と混在し見通しが悪くならないように分離して配置した。

5．開発実績と評価

5．1　開発工数

本試行においては、当初見積もりの3.5 人月に対して、1.7 倍の工数を要した。ただし、開発全体の工数の内訳としてeUML の作業以外（管理・資料作成、技術習得、技術トラブル対応）がほぼ50%を占める結果となった（**図６**）。このオーバヘッド時間は、その当時の担当者のスキルや社内の事情、さらにはトラブルにより発生したオーバヘッドである。こ

れらのオーバヘッドのうち eUML/KonesaRT の技術習得（約 10%）を除く大部分は、工数分析の結果、今回の特殊事情であることがわかった。これを評価から除外して考えると、eUML ガイドラインに沿った作業は、当初計画の 3.5 人月内で吸収できる範囲であった。つまり、OOAD を導入したときの迷いによるオーバヘッドは eUML の導入により最小限にすることができた。

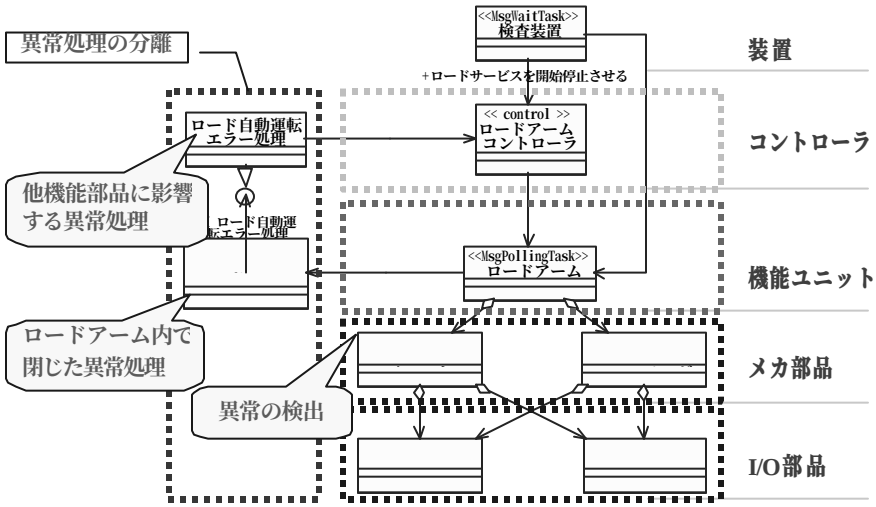

図５　制御部品の階層化

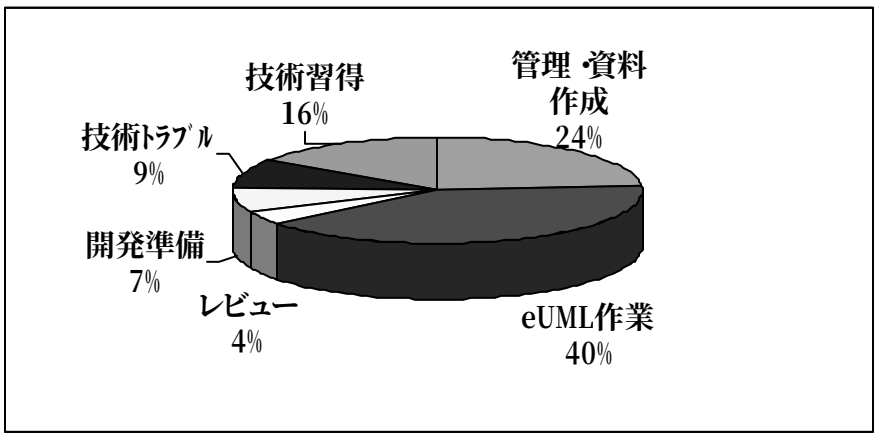

図６　開発工数の内訳

5．2　自動コード生成

本試作で開発したソースコードの実ステップ数（コメント抜き）は表１に示すとおりである。当初想定していたよりも自動生成されたコード比率が高い結果になった。これは制御要素が状態遷移モデルで記述しやすいことが大きな理由である。

表１　実装ステップ数比率

コードの種類	ステップ数	比率
① KonesaRT によって自動生成されたコード	12.1 [K Step]	75.6 [%]
② 手書きで実装したコード(※)	1.4 [K Step]	8.8 [%]
③ KonesaRT 提供の RTOS ラッパ	2.5 [K Step]	15.6 [%]
合計	16.0 [K Step]	

※　状態遷移表に埋め込むガード条件やアクション、および状態遷移を持たないクラスのメソッドが該当する。

6．検証結果

本試行の結論としては、産業用制御システム開発に eUML/CASE ツールを適用した場合、開発効率、および品質の向上が期待できると考える。若干、改善すべき点はあるものの、次の２点がその大きな理由である。

- ✧ 部品化の概念、状態遷移ベースのモデリングは FA 向きであることを実感した。
- ✧ 予想以上に自動生成したコード比率が高かった。

以下に本試行のねらいに対する評価を列挙する。

(1) 上流工程の設計品質向上に関しては、並行性およびインターロックの基本方針の検討、各ハードウェア制御の状態モデルまで踏み込んだ設計イメージの共有、レビューが設計者間でできる状況になった。

(2) 実装の効率化については、メカと対になったソフトウェア部品が提供できた。またメカとの組み合わせによる再利用は、装置の部品化として現実的な解であると考える。しかしながら、自動コード生成については自動コード生成比率が高いもののツールに不完全な部分があり、手修正が必要であった。

(3) 結合デバッグ期間の短縮については、状態遷移モデルのシミュレーションによるインターロックの事前検証が効果であった。実機を用いた結合デバッグにおいてはバグに起因する予想外のメカ動作によりメカが破損することがある。シミュレーションの事前検証はデバッグ時のメカの保護、開発者の安全確保という観点でも有効な手段となる。

全体を通して、多少改善の余地があるものの、当初の狙いどおりの効果が確認できた。自動コード生成についてもCASEツールの進化により解消されることを期待する。

7．おわりに

本試行を通して、eUMLとCASEツールの組み合わせによる分析、モデリングから実装へのシームレスな流れを体感することができた。特に、インターロックなど装置の全体制御に絡む部分においては、モデリング、シミュレーションによりメカ動作の事前検証が行えた点、品質を確保したモデルをもとに実行コードを生成できた点が評価できる。今後は本試行を踏まえて、FA分野においても「実行可能なモデルと実行環境」を提供することにより、CASEツールでモデリングしたソフトウェアがコントローラで即実行できるような環境づくりを加速したい。

8．参考文献

[1] オムロン(株)恵木守、(株)永和システムマネジメント平鍋健児、UMLとIEC1131-3を利用した産業用制御システムの設計と実装、情報処理学会第61回全国大会

[2] オムロン(株)妹尾吉紳、西川信孝、鶴岡正敏、UMLを活用したオブジェクト指向開発プロセスの実践、OMRON TECHNICS,Vol.41 No.4,2001,pp371-pp378

[3] オージス総研 渡辺博之、ソニー堀松和人、UML モデリングの極意 前編/後編、JavaWorld,Oct,2001,pp113-121, Nov, 2001,pp121-129

[4] キャッツ(株)渡辺政彦、UMLの組込みシステム向け拡張「eUML」の全ぼう、DesignWave,Dec,2001,pp41-49

[5] オージス総研 渡辺博之、組込み・リアルタイム向オブジェクト指向分析・設計トレーニング Ver.1.03、オージス総研トレーニング資料