1Q−06

UMLとデザインパターンを利用したFA用分散制御システムのシミュレーションと実装の統合化

— 状態遷移仕様とイベント連鎖からの分散制御ソフトウェアの自動生成 — *

戸村 豊明 †　金井 理　岸浪 建史 ‡　伊深 和浩　上広 清　山元 進 ¶
旭川工業高等専門学校 ††　北海道大学大学院工学研究科 ‡‡　モトローラ株式会社 ¶¶

1. はじめに

近年，FA（ファクトリオートメーション），BA（ビルオートメーション）の分野において，多数の制御ノードを相互接続したオープンネットワークを持つ分散制御システム（DCS）が導入され始めている．

図1のように，DCSは多数のデバイスから構成される事から，通信の遅れやトラフィックといった制御特性を事前解析できるDCSシミュレータが必要とされている．各制御ノードで実行される制御ソフトウェアの挙動は有限状態機械として仕様化できる事が多く，この仕様をDCSシミュレータ用モデルだけでなく，各制御ノード上へ実装される制御ソフトウェアの開発にも，シームレスに利用できる統合的開発ツールが必要とされている．

本研究では，DCSのシミュレーションと実装の統合を目的として，オブジェクト指向モデリングに基づき図1のstatechart図[1]とそれらの間のイベント連鎖をDCSシミュレータ用モデルとなるJavaコードとして実装し[2]，さらにそのJavaコードから各制御ノードの実装用ソフトウェアコードを自動生成可能な[3]デザインパターンに基づくDCS開発方法論と開発ツールを提案してきた．しかし，具体的な制御対象の開発を対象とした，提案方法論とツールの有用性の検証が不充分であった．

そこで本報では，FA用ピック&プレースユニットの分散制御ソフトウェア開発に対し，上記の3つのパターンを適用する事により，これらのパターンの有用性を示す．

2. 本研究で提案されたデザインパターン

これまで我々が提案してきた3パターンを以下に示す．

- Statechartパターン[2]：図1の各statechart図[1]を状態機械，状態，状態遷移，イベント，ガード条件，アクションのオブジェクトの集合として実装する．
- Event-Chainパターン[2]：statechart図のアクションとイベントの間をバインディングオブジェクトで連鎖的に接続し，各デバイスのイベント送信オブジェクトがデバイスの内外へイベント送信するよう実装する．
- Statechart-Compilerパターン[3]：上記の2パターンのオブジェクト記述に基づいた状態の階層構造を持つオブジェクトの構文解析木を作成し，各オブジェクトへ変数名を割り当て，各制御ノード上の制御ソフトウェア開発用言語（Neuron C）のコードを自動生成する．

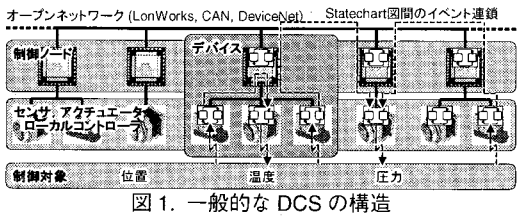

図1. 一般的なDCSの構造

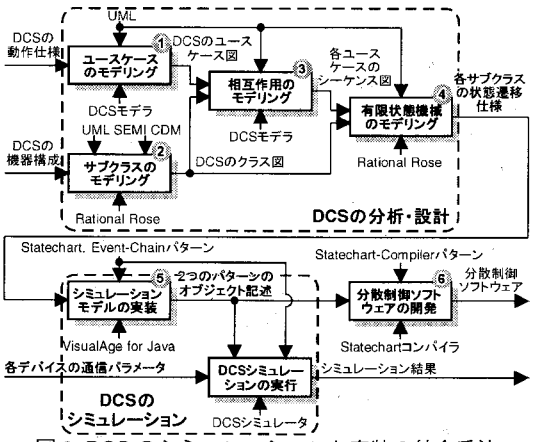

図2. DCSのシミュレーションと実装の統合手法

3. DCSのシミュレーションと実装の統合手法

本研究で提案するデザインパターンを用いたDCSのシミュレーションと実装の統合手法を図2に示す．この手法は，以下に示す3つのプロセスへ分解できる．

(1) DCSの分析・設計: DCSの機器構成と動作仕様をもとに，DCSに要求される機能（ユースケース），DCSを構成するサブクラスを明らかにした後，最終的に各サブクラスの有限状態機械とイベント連鎖を，各サブクラスの状態遷移仕様として得る．

(2) DCSのシミュレーション[2]：(1)で得られた各サブクラスの状態遷移仕様をもとに，Statechartパターンと Event-Chainパターンを用いてDCSシミュレーションモデルを実装した後，各デバイスの通信パラメータを与えて，DCSシミュレーションを実行する．

(3) 分散制御ソフトウェアの開発（DCSの実装）[3]：Statechart-Compilerパターンを用いて，(2)から得られた2つのパターンのオブジェクト記述をもとに，制御ノード用分散制御ソフトウェアを生成する．

上記のうち，(2)，(3)については前報[2),3)]で報告しているので，本報では，(1)のDCSの分析・設計の具体的適用例を以下の節で述べる．

---

* An integration of simulating and implementing distributed control systems for FA using UML and design patterns - automatic generation of distributed control software from state transition specifications and event chains -

† Toyoaki Tomura
†† Asahikawa National College of Technology
‡ Satoshi Kanai, Takeshi Kishinami
‡‡ Graduate School of Engineering, Hokkaido University
¶ Kazuhiro Ibuka, Kiyoshi Uehiro, Susumu Yamamoto
¶¶ Motorola Japan

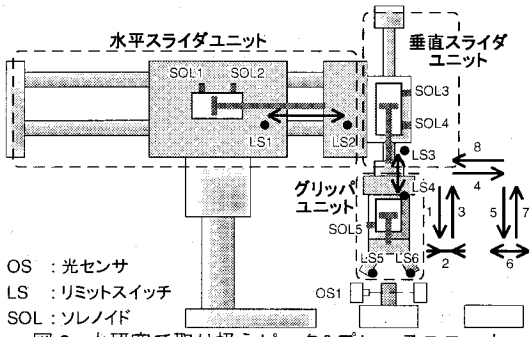

図 3. 本研究で取り扱うピック&プレースユニット

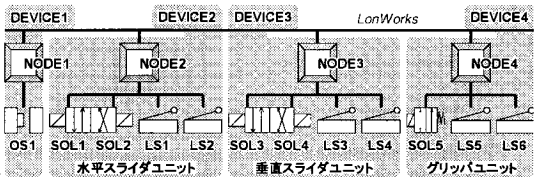

図 4. ピックプレースユニット用 DCS の構造

4. 本研究で取り扱う FA システム

本研究では，図 3 に示すピック&プレースユニットの制御ソフトウェア開発に対し，図 2 の開発方法論を用いて DCS を分析・設計する．このユニットは水平・垂直スライダ，グリッパからなり，以下の動作を繰り返す．

(1) Going Down1: ワークがテーブル上にあれば，垂直スライダユニットが下降．
(2) Gripping: グリッパユニットがワークを掴む．
(3) Going Up1: 垂直スライダユニットが上昇．
(4) Moving Forward: 水平スライダユニットが前進．
(5) Going Down2: 垂直スライダユニットが下降．
(6) Releasing: グリッパユニットがワークを放す．
(7) Going Up2: 垂直スライダユニットが上昇．
(8) Moving Backward: 水平スライダユニットが後退．

図 4 は，このユニットを制御する DCS の構造を示したものである．この DCS は，光センサと各小ユニットを 1 つのデバイスとする小規模の DCS であり，オープンネットワークとして LonWorks を用いている．

5. FA システムの分散制御ソフトウェア開発

図 2 に示す開発方法論に従って，この FA システムの制御用 DCS の分析･設計を下記の手順により実施する．

①DCS の動作仕様は，前節で示した動作のシーケンスである．ゆえに，前節の動作(1)〜(8)は，それぞれ 1 つのユースケース [1] として定義できる．
②DCS の機器構成とは，図 4 に示した DCS における各デバイスとそのセンサ・アクチュエータを指している．ゆえに，図 4 におけるデバイス・センサ・アクチュエータは，それぞれ 1 つのサブクラスとして定義できる．
③①で得られた各ユースケースを，②で得られたサブクラスのオブジェクト間の相互作用を表すシーケンス図として定義する．ここで，Going Down1 ユースケースを実現するためのシーケンス図 [1] を図 5 に示す．

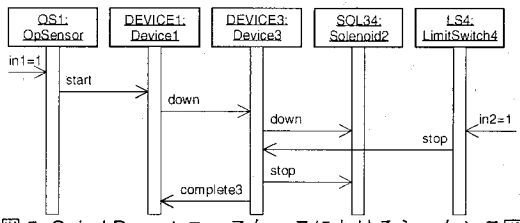

図 5. Goind Down1 ユースケースにおけるシーケンス図

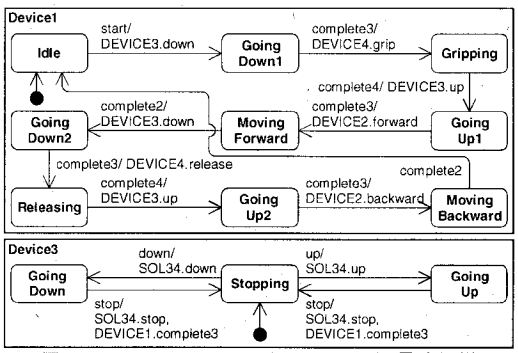

図 6. シーケンス図から得られた状態遷移仕様

④③で得られた各シーケンス図（図 5）から，同一のオブジェクトの挙動を抽出・合成する事により，statechart 図で記述された各サブクラスの状態遷移仕様を定義する．図 5 のシーケンス図に関係する Device1, Device3 サブクラスの状態遷移仕様を図 6 に示す．状態機械間のイベント連鎖は，statechart 図におけるイベント・アクションの中で定義されている．
⑤④で得られた各サブクラスの状態遷移仕様(図 6)は，Statechart パターンと Event-Chain パターンを用いて，実行可能 Java コードとして実装される．
⑥⑤で得られた Statechart パターンと Event-Chain パターンのオブジェクト記述をもとに，Statechart-Compiler パターンを用いて，各制御ノード上へ実装される分散制御ソフトウェアの Neuron C コードを自動生成する．

6. まとめと今後の課題

本報では，我々が提案してきたデザインパターンに基づく DCS 開発方法論と開発ツールを，FA 用ピック&プレースユニットの分散制御ソフトウェア開発に対して適用する事で，その方法論とツールの有用性を検証した．

今後は，これまで提案された 3 つのパターンを完全に統合したパターンを新たに提案してゆく予定である．

参考文献

[1] OMG: “OMG Unified Modeling Language Specification Version 1.3”, OMG, 2000.
[2] T. Tomura et al.: “Developing Simulation Models of Open Distributed Control System by Using Object-Oriented Structual and Behavioral Patterns”, Proceedings of ISORC 2001, pp.428-437, 2001.
[3] 戸村他: “デザインパターンによる DCS シミュレーションと実装の統合化— イベント連鎖からの DCS ネットワーク構成情報の抽出・実装 —”, 情報処理学会第 63 回全国大会講演論文集, pp.459-460, 2001.