オブジェクト指向デザインパターンに基づく超分散システム制御ネットワークシミュレータ用コンポジットデバイスモデルの迅速構築

旭川工業高等専門学校　○戸村 豊明
北海道大学　金井 理，岸浪 建史
モトローラ　上広 清，山元 進

要 旨

FA・BA の分野で，制御ノードの数が 1 万を超える超分散システム制御ネットワークが導入されている．ネットワークシミュレータ上での制御特性の解析を目的として，複数のデバイスモデルからなる CompositeDevice モデルを迅速構築するための手法が現在必要とされている．

本報では，オブジェクト指向に基づき，予め定義されたデバイスのオブジェクトとその個数をパラメータとして，CompositeDevice モデルを迅速に構築するためのデザインパターンとそれを用いた CompositeDevice モデル構築手順を提案する．さらに，モデル構築の具体例を示す事で，このパターンとモデル構築手順の有効性を示す．

1. はじめに

近年，FA・BA（ビルオートメーション）分野で，制御ノード数が 1 万を超える LON や CAN といった超分散システム制御ネットワークが導入されており，通信時間遅れやトラフィックの頻度に応じた制御特性をネットワークシミュレータにより事前解析したいという要求が高まっている．

図 1 のように，この制御ネットワークにおいて、デバイスは制御ノードに相当し，複数のセンサやアクチュエータの集合体となっている．この制御ノード数が増加した場合，シミュレーションを効率化するには，デバイスモデルの迅速構築手法が必須となる．このため，本研究では，オブジェクト指向とデザインパターン[1]に基づき，デバイスモデルの Java コードを迅速開発する手法を[2]を提案してきた．

しかし，実際のシステムでは，CompositeDevice と呼ばれる再帰的に定義されたデバイスの集合体に特定の制御機能を持たせ，この集合体をシステムの区画・フロアごとに繰り返し配置して設計を行う場合が多い．従って，大規模なシミュレーションモデルの迅速構築には，この Composite Device 単位でのモデル迅速構築手法が必要不可欠となる．この CompositeDevice には，デバイスの種類や個数が類似したパターンを持つものが多い．従って，デバイスの種類と個数をパラメータとして，CompositeDevice のシミュレーションモデルを柔軟かつ効率的に構築可能なデザインパターンと，モデル実装コードの開発手順が求められる．

本報では，オブジェクト指向に基づき，予め定義されたデバイスのオブジェクトとその個数をパラメータとして，CompositeDevice モデルを迅速に構築するためのデザインパターンとして Composite Device Constructor パターン，およびこのパターンを用いた CompositeDevice モデルの構築手順を新たに提案する．さらに，CompositeDevice モデルの構築の具体例を示す事で，パターンと手順の有効性を示す．

2. CompositeDevice モデルの迅速構築に対する要求条件

本研究で取り扱う CompositeDevice モデルは，図 2 のように，SEMI の Common Device Model[3]を一部拡張したものである．図 2 の場合，Device の具象クラスを追加すると，CompositeDevice の機器構成が不明確になるので，図 3 のように，CompositeDevice が Device の具象クラスを直接管理する手法が考えられる．表 1 はこの手法と本研究で提案する手法を比較したものであるが，(2)と(3)の項目において，図 3 の手法は大きな問題点を持っている．

表 1 より，CompositeDevice モデルの迅速構築に対する要求条件として，以下のような条件が挙げられる．

(1) 具象クラスを定義せずに，特定の機器構成を持つ Device オブジェクトを一種のクラスとして取り扱う事で，CompositeDevice クラスとの関連構造は不変．

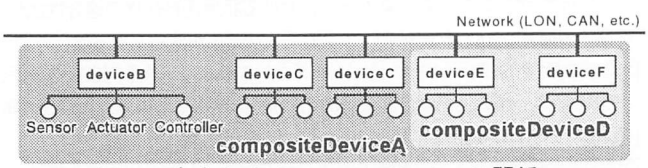

図 1. デバイスと CompositeDevice の関係

表 1. 構成要素を直接管理する手法と本研究の手法との比較

比較項目 ＼ 静的構造の定義方法	CompositeDeviceクラスによる構成要素の直接管理	DeviceContainerクラスによる構成要素の管理（本研究の手法）
(1) 構成要素の具象クラス	具象クラスを定義可能	具象クラスの定義は不要
(2) CompositeDevice—構成要素間の関連構造	具象クラスの追加ごとに，関連を定義し直す必要あり	具象クラスは定義されないので，関連は不変
(3) 構成要素オブジェクトの生成の責任	全具象クラスのコンストラクタを呼び出す具体的な手続きをCompositeDeviceへ実装	構成要素はDeviceContainerクラスが生成するので，コンストラクタ呼び出し手続きは常に一定

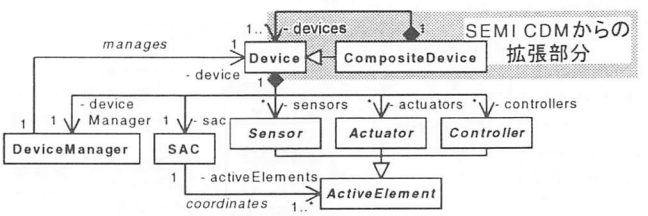

図 2. 本研究で取り扱う CompositeDevice モデル

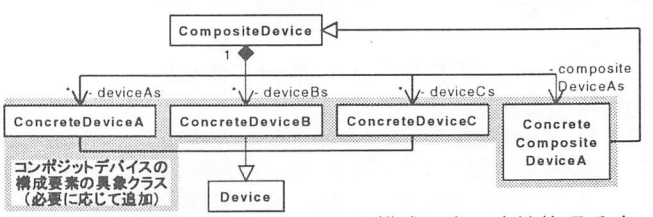

図 3. CompositeDevice クラスによる構成要素の直接管理手法

(2) 構成要素となる Device オブジェクトの個数に関わらず，CompositeDevice クラスにおける構成要素のコンストラクタ呼び出し手続きは不変．

(3) Device オブジェクトの追加・削除が容易．

上記の要求条件を満たすために，各条件に対して，以下の機能を持つデザインパターンがそれぞれ必要となる．

(1) 関連構造を保持するために，Device オブジェクトに対し，ユニークな名前を付け，それを記憶しておく．

(2) Device オブジェクト群を格納するためのクラスが，コンストラクタ呼び出しを代行する．

(3) (1)で述べたユニークな名前と，オブジェクトの個数を受け取るインターフェイスを提供する．

また，上記のデザインパターンのみでは，実際のモデル実装コードを開発する方法が不明確ため，このデザインパターンに適合する詳細なモデル構築手順が別途必要になる．

3. Composite Device Constructor パターン

上記の要求条件を満たすため，本研究では，図 4 に示す Composite Device Constructor パターンを提案する．このパターンは，[2]で提案した Device Constructor パターンに対し，CompositeDevice，DeviceContainer，DeviceFactory を追加したものである．このパターンの利点は明示的な具象クラス定義が不要で，かつ Device と CompositeDevice オブジェクトの区別なしに，これらを動的に組み合わせるだけで，新たな CompositeDevice モデルを定義可能な点にある．

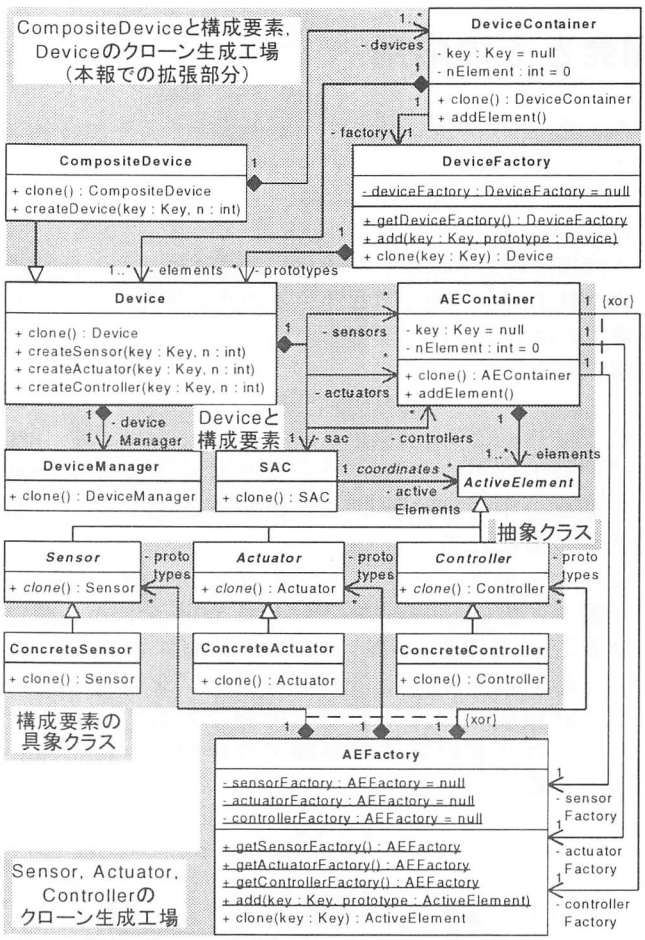

図 4. Composite Device Constructor パターン

CompositeDeviceモデルに関するクラスの役割を次に述べる．

- **Device**：デバイスモデルを表す．
- **CompositeDevice**：CompositeDevice モデルを表す．構成要素のキーとその個数をもとに，Device（CompositeDevice）オブジェクトを生成できる．
- **DeviceContainer**：特定の種類の Device (Composite Device) オブジェクトを格納する．種類を表すキーとオブジェクトの個数を属性として持つ．
- **DeviceFactory**：各種の Device (CompositeDevice) オブジェクトを登録しておき，そのキーを参照する事で，オブジェクトのコピーを生成する．

4. Composite Device Constructor パターンを用いた CompositeDevice モデルの構築手順

Composite Device Constructor パターンを用いた Composite Device モデルの構築手順を図 5 に示す．この手順における各プロセスについて，以下に述べる．

① デバイスの仕様記述を参照し，Device Constructor パターン[2]に従い，Device オブジェクトをモデリングする．その動的挙動は UML の状態図として記述され，Statechart パターン[4]に従って実装される．

② ①で得られたオブジェクトを，シミュレータ上の DeviceFactory へ登録する手続きを Java 言語で記述する．

③ 機器構成と，Device オブジェクトのキーをもとに，Device オブジェクトを組み合わせて，Composite Device モデルをオブジェクト図として記述する．

④ オブジェクト図をもとに，CompositeDevvice モデルと構成要素を生成する手続きを Java 言語で記述する．

5. CompositeDevice モデルの構築の具体例

Composite Device Constructor パターンを用いた Composite Device モデル構築の具体例として，図 1 に示す 1 個の deviceB，compositeDeviceD，2 個の deviceC からなる compositeDeviceA を考える．この時の各プロセスについて以下に述べる．

表 1. Device オブジェクト登録手続きと CompositeDevice オブジェクト構築手続き

Deviceオブジェクト登録手続き	CompositeDeviceオブジェクト構築手続き

`DeviceFactory f = DeviceFactory.getDeviceFactory();`
`Device deviceB = new Device();`
`/* deviceBの機器構成を設定する */`
`f.add(new Key("deviceB"), deviceB);`
`Device deviceC = new Device();`
`/* deviceCの機器構成を設定する */`
`f.add(new Key("deviceC"), deviceC);`
`Device deviceD = new CompositeDevice();`
`/* deviceDの機器構成を設定する */`
`f.add(new Key("deviceD"), deviceD);`
	`CompositeDevice deviceA = new CompositeDevice();`
	`deviceA.createDevice( new Key("deviceB"), 1);`
	`deviceA.createDevice( new Key("deviceC"), 2);`
	`deviceA.createDevice( new Key("deviceD"), 1);`

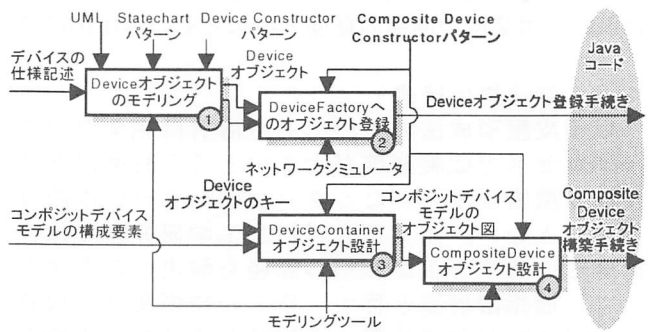

図 5. Composite Device Constructor パターンを用いた CompositeDevice モデルの構築手順

① 3 種類の構成要素が compositeDeviceA のモデリング以前に定義済なら各構成要素の構築手続きが得られる．

② ①のオブジェクト構築手続きとオブジェクトのキーを用いて，DeviceFactory オブジェクトへ①のオブジェクトを登録する手続きを記述する（表 2 左側参照）．

③ オブジェクトのキーと compositeDeviceA の機器構成より，DeviceContainer オブジェクトを組み合わせる事で，オブジェクト図が得られる．

④ ③のオブジェクト図より，CompositeDevice クラスの createDevice()呼び出しからなる compositeDeviceA の構築手続きを記述する（表 1 右側参照）．

Composite Device Constructor パターンとモデル構築手順に基づくモデリングツールとネットワークシミュレータ[5]は，モトローラで現在開発されている．

6. まとめと今後の課題

本報では，複数のデバイスモデルからなる Composite Device モデルの迅速構築を可能とする Composite Device Constructor パターンとモデル構築手順を提案した．

今後は，デバイス（CompositeDevice）モデル間のイベント連鎖を記述するためのパターンを提案する予定である．

参考文献

[1] Gamma, Helm, Johnson, Vlissides : "オブジェクト指向における再利用のためのデザインパターン", ソフトバンク, 1999.

[2] 戸村, 金井, 岸浪, 上広, 山元 : "オブジェクト指向デザインパターンを活用した超分散システム制御ネットワークシミュレータ用デバイスモデルの迅速構築", 2000 年度精密工学会秋季大会学術講演会講演論文集, 2000 (発表予定).

[3] SEMI : "SEMI E54.1-0298 Standard for Sensor/Actuator Network Common Device Model", SEMI, 1998.

[4] 戸村, 金井, 岸浪, 上広, 山元 : "超分散システム制御ネットワークシミュレータの開発 (第 2 報)", 2000 年度精密工学会春季大会学術講演会講演論文集, 2000.

[5] 山元, 上広 : "超分散システム制御ネットワークシミュレータの開発(第 1 報)", 2000 年度精密工学会春季大会学術講演会講演論文集, 2000.