情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2010-EMB-18 No.9
2010/8/10

システム・バリエーションのアスペクト指向による部品化

上野高元[†1]　野口雅司[†2]　植木雄一[†2]
松本倫子[†1]　吉田紀彦[†3]

本論文では多様なバリエーションを持つシステムを，アスペクト指向を用いて効率的に設計する手法を提案する．アスペクト指向技術により，システム共通部分とバリエーション機能部分を分離することが可能であり，またバリエーション機能間に相互独立性を確保することにより，機能管理，設計効率が向上する．本研究の成果として，異なるバリエーション機能間でも相互独立性が保てることを示した．

Aspect-Oriented Component Reuse of System Variations

Takamoto Ueno,[†1] Masashi Noguchi,[†2]
Yuichi Ueki,[†2] Noriko Matsumoto[†1]
and Norihiko Yoshida[†3]

In this paper, we propose an aspect-oriented technique to improve design process of systems with many functional variations. This technique enables us to separate functional variation parts from common parts, and also ensures mutual independence between different functional variation parts. Consequently, it improves design productivity.

1. はじめに

近年，情報通信技術の発展と共に，システムに課される要求は多様化・複雑化の一途を辿っている．特に，組込みシステムは汎用システムと異なり，リソース制約，リアルタイム性，高信頼性と非常に厳しい要求が課されている．それためにシステムの開発はより複雑化し，設計期間の長期化を招いている．

システムに課される要求はそれだけに留まらず，国，地域毎に異なる法規，環境，部品メーカの違い，ユーザの嗜好の違いなど，多種多様である．これにより，同一用途のシステムでありながら，それら異なる要求を満足する為のバリエーション機能を持つシステムを生み出しており，その機能数は最早無視することの出来ないレベルまで膨れ上がっている．このようなシステムに対して開発現場では，設計時に全ての機能を組み込み，工場・販売拠点において必要な機能を選択，決定している．しかしながらこの方法では，すべての機能を網羅した試験の組合せ，バリエーション機能管理が難しく，非効率な開発を余儀なくされている．

そこで本研究では，アスペクト指向に基づいてバリエーション機能を部品化することを提案する．アスペクト指向とは，複数のコンポーネント，あるいはモジュールに跨る横断的関心事を分離，独立させることにより，システムの開発効率，保守性を向上させる技術である．

異なるバリエーション機能は，その順序を意識せずに機能追加が可能であることが望ましい．その性質を相互独立性と呼び，その性質を持たせることで，バリエーション機能数と部品数が等しくなり，バリエーション機能管理が容易になる．

本論文では，異なるバリエーション機能間でも，順序に捉われずに機能追加が出来る部品の作成が可能であることを示す．以下，2 節では，複数のバリエーション機能を有するシステムの発生と，そのシステムを効率的に設計する為に解決しなければならない課題，及び本論文で提案する解決方法について概説する．3 節では，本研究で利用する技術であるアスペクト指向技術について概説する．4 節では，アスペクトによるバリエーション機能の部品化について述べ，バリエーション機能管理を容易にするためには相互独立性を保つことの重要性を述べる．5 節では，本研究で用いたモデリング言語である MATLAB/Simulink/Stateflow を簡単に紹介する．6 節では，提案した手法を，自動ライト制御システムへ適用した結果を述べる．最後に，7 節をまとめとする．

2. バリエーションの発生と解決策

多様化・複雑化した要求は，同一用途の目的として設計されたシステムでありながら，複数のバリエーション機能が存在するシステムを発生させるに至った．その顕著な例を 1 つ挙げれば，2006 年に発売された Mercedes Benz C class が取り得るオプションは非常に広範

†1 埼玉大学大学院理工学研究科
Graduate School of Science and Engineering, Saitama University
†2 カルソニックカンセイ株式会社
Calsonic Kansei Corp.
†3 埼玉大学情報メディア基盤センター
Information Technology Center, Saitama University

Vol.2010-EMB-18 No.9
2010/8/10

であり，全く同じものが 2 つと存在しないと言われている[1]．その背景には，異なる地域毎の法令順守，顧客指向による幅の広い選択肢，電子機器によってもたらされる機能オプションなど，様々な要因があり，バリエーション機能はシステムの至るところに散在している．

多くの開発現場では，このような同一用途のシステムでありながら複数のバリエーション機能を有するシステムを設計する際，予め起こり得るすべての機能を組み込んでおき，コンパイラスイッチ，コンフィグレーション・パラメータに基づいた場合分けによる機能選択などの方法でバリエーション機能に対処しているのが現状である．しかしながら，すべてのバリエーション機能組合せを網羅したテストの難しさ，バリエーション機能の管理の難しさなどから，非効率な設計を余儀なくされている．

本研究では，アスペクト指向技術に基づいてバリエーション機能を部品化することを提案する．すなわち，システムのバリエーション機能を横断的関心事として扱い，システム共通部分から分離・独立したアスペクトとして部品化する．これにより，バリエーション機能を体系的に扱うことができ，システムの保守性，設計効率が向上することが期待できる．

3. アスペクト指向技術

システムが大規模化・複雑化するにつれ，新しいシステムの開発には既存の部品を再利用することが重要となってきた．オブジェクト指向技術では，システム全体の機能をデータと手続きを持ったオブジェクトの集まりとして捉えることにより，個々の機能の独立性が重視され，システムの開発効率，保守性，個々の機能の再利用性が向上した．

しかし一方で，オブジェクト指向技術は，システムの全ての機能を分離，独立できる訳ではない．複数のシステムに跨る機能は分離，独立させることが出来ず，その機能の追加，変更，修正は，関連するコードを内在する全てのコンポーネントに及び，システムの開発効率，保守性が十分に向上しない．アスペクト指向技術[2] は，オブジェクト指向技術だけでは不十分な点を補い，システムの開発効率，保守性をさらに向上させるものである．以下，アスペクト指向技術の目標，および要素について述べる．

3.1 アスペクト指向の目標

オブジェクト指向に基づいてモデリングを行うときには，階層化を用いるのが一般的である．すなわち，所望の動作を行うシステムをトップレベルに配置し，そのシステムを実現するためのコンポーネント，あるいはサブコンポーネントをその下層に配置するといったように，システムの機能を階層的に位置付ける．そして，最下層にはある動作を行う機能が配置され，この最小単位がオブジェクトと呼ばれる．

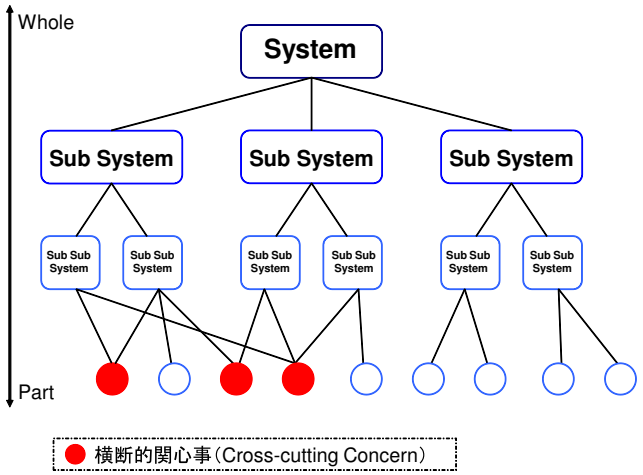

図 1 横断的関心事
Fig. 1 Cross-Cutting Concerns.

しかしながら，オブジェクト指向に基づく階層化原理が，必ずしもすべてのシステムをモデリング出来るとは限らない．システムを設計するとき，あるコンポーネント，あるいはオブジェクトが，別の機能の実現に関連があることはむしろ自然であり，現実のシステムの多くは複数の機能が複雑に絡み合った姿をしている．従って，ある機能をどのコンポーネント，あるいはオブジェクトに担当させるべきかを決定することは極めて重要，かつ困難な問題である．

ここまで「機能」と呼んできたものは，システム開発者がある観点に基づいてモデリング・設計を行った際に,「個別に着目することができ」なおかつ「ひとまとめに扱うことのできる」ものである．アスペクト指向ではこれを関心事 (concerns) と呼び，先に述べたような複数のコンポーネントに跨る関心事を横断的関心事 (Cross-Cutting Concerns) と呼ぶ（図 1）．

アスペクト指向の目標は，横断的関心事をコンポーネントから分離・独立させることにある．すなわち，システムを分析した際に複数のコンポーネントの下層に位置してしまう関心事を，システムの至る所に散在させてしまうのではなく，その関心事に対応する専用の部品として分離，独立させることを目指す．それにより，システムの開発効率，保守性の向上が

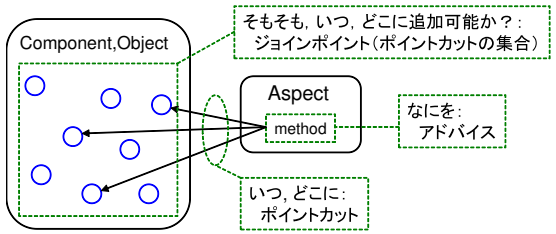

図 2 アスペクト指向の要素
Fig. 2 Elements of aspect-oriented.

期待できる．

ここで，アスペクト指向技術はオブジェクト指向技術に取って変わるものでないことに注意されたい．アスペクト指向技術は，あくまでもオブジェクト指向技術では扱うことが困難な部分を補うものである．従ってシステム開発者は，オブジェクト指向に基づいてシステムを設計し，それでは不十分な点をアスペクト指向により補うことを念頭に置かなければならない．

3.2 アスペクト指向の要素

アスペクト指向に基づくプログラミング言語は Aspect J[2]，Aspect C++[3] などがある．それら言語には共通してジョインポイントモデルが採用されており，ここでもそれを紹介する．

横断的関心事を分離・独立させ，そのシステムに組み込むためには,「何の」機能を組み込むのか，また,「いつ」「どこに」目的とする機能を組み込むのかということを指定しなければならない．そのうち,「何の」機能に当たるものをアドバイス,「いつ」「どこに」組み込むのかに当たるものをポイントカットと呼ぶ．一方で，システムに機能を追加する際には，そもそもいつ，どこに機能を追加することが可能であるのか，ということを厳密に定めなければならない．この機能を追加可能な場所のことをジョインポイントと呼び，ポイントカットは予め定義されたジョインポイントから指定される．

アスペクト記述を定義した次の段階として，アスペクトにより記述された機能を元のプログラムに適用する為の仕組みが必要となる．アスペクトに記述された機能をシステムに追加することを織り込み (Weaving) と呼び，それを実現するプログラムをウィーバー (Weaver) と呼ぶ．

以上より，ある言語にアスペクト指向技術を導入するためには，ポイントカット，アドバイスを含んだアスペクト記述法の確立，およびアスペクトを対象とするシステムに織り込む為のウィーバーの作成が必要となる．

4. アスペクト指向によるシステム・バリエーション機能の部品化

アスペクト指向をモデル変換に応用する試みは既に行われている．文献 4) では，モデル駆動アーキテクチャ(Model-Driven Architecture，MDA) に基づく設計手法において，ある特定のプラットホームや実装技術に依存しないプラットフォーム独立モデル（Platform Independent Model，PIM）から依存するプラットフォーム特化モデル（Platform Specific Model，PSM）への変換情報をアスペクトを用いて与えることを提案している．これにより，システム設計者はモデリングの一環としてアスペクトを定義することができ，モデルコンパイラの機能を拡張することができる．また，文献 5) では，実行可能 UML を用いた通信探索における具体化作業をアスペクト指向技術により自動化出来ることを確認している．

本研究では，バリエーションのあるシステムをアスペクト指向に基づいて設計する．最初に，その設計方針について述べる．

4.1 方　　針

複数のバリエーション機能を有するシステムにおいて，バリエーション機能に捉われない共通機能が存在することは自然であり，本研究ではバリエーション機能から分離，独立させたシステム共通部分が設計可能であることを前提としている．このとき，システム共通部分とは，例えば共通な入出力機構など，簡素なものであって構わない．このシステム共通部分に対し，必要なバリエーション機能を選択し，追加していく．

本研究では，バリエーション機能の部品化をアスペクト指向に基づいて行う．しかしながら，その部品化は単純に一つ一つのバリエーション機能をアスペクトとしてシステムから分離・独立させることではないことに注意されたい．あるバリエーション機能が，他のバリエーション機能と全く関連性を持たないのであれば，システム共通部分からバリエーション機能を分離・独立させることは難しくない．それは，バリエーション機能を実装するものがコンポーネントであろうと，オブジェクトであろうと変わりは無い．より困難な状況は，図 3 のようにあるバリエーション機能に当たるコンポーネントが，他のバリエーション機能であるサブコンポーネント，あるいはオブジェクトと関連性を持っている場合である．このような場合は，異なるバリエーション機能を追加する順番を考慮しなければならないが，それらの順序を意識することは好ましくない．

各バリエーション機能は，他のバリエーション機能と独立して存在していることが望まし

Vol.2010-EMB-18 No.9
2010/8/10

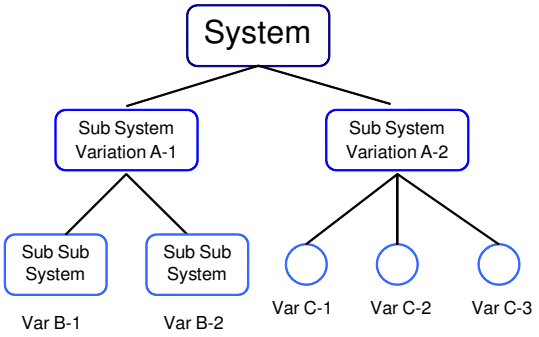

図 3 関連性のあるバリエーション機能
Fig. 3 Relative functional variation.

い．システム共通部分に，複数のバリエーション機能を追加する際，バリエーション機能間に独立性を持たせることで，追加順序を意識せずにシステムへの機能追加が可能である．この性質を部品間の相互独立性と呼び，この性質を持たせることでバリエーション機能数と部品数が同一になることが期待できる．本研究において，部品間の相互独立性を確保することは，バリエーション機能部品を減少させるための非常に重要な課題であり，次節ではそれを満足するための方法を述べる．

4.2 相互独立性の確保

既述のように，異なるバリエーション機能が互いに関係性を持たないのであれば，異なる部品間で相互独立性を確保することは容易である．また，互いに関連性のあるバリエーション機能間でも，アスペクト指向による分離・独立は困難な課題ではない．問題は，図 3 のようにバリエーション機能であるコンポーネントの下層に，異なるバリエーション機能が配置されている場合であり，この場合は相互独立性を確保するための仕組みが必要である．

ここでは，異なるバリエーション機能として，一方がコンポーネント，他方がパラメータである場合について述べる．

我々は，図 4 のようにバリエーション機能（VarB1_val）が他のバリエーション機能（ComponentA1）内に配置されていても，それらの機能追加先をシステム共通部分とすることを考えた．

そのために，ComponentA1 をアスペクトとして部品化する際に，アドバイス内に VarB1_val を参照する為の関数 getter_VarB を挿入する．そして，VarB1_val を部品化

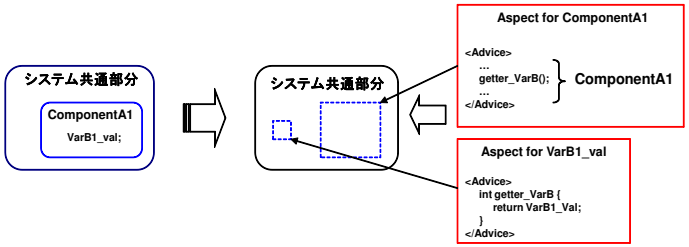

図 4 関連性のある機能間の相互独立性の確保
Fig. 4 Ensuring mutual independency between relative variations.

する際には，その機能を値としてではなく，VarB1_val を返り値とする関数とする．これらを共通仕様モデルに追加することにより，順序に依存することなく異なるバリエーション機能が追加可能となり，相互独立性を保つことができる．

異なるバリエーション機能がより複雑なものになったとしても，本質的な解決策は異なるバリエーション機能間において，その機能の追加先を共通のものとすることである．

5. MATLAB/Simulink/Stateflow

本研究では，システムのモデリング言語として，Mathworks 社[6] が開発した MATLAB/Simulink，およびその機能拡張ツールである Stateflow を用いている．

MATLAB は元来，数値計算の為の豊富なライブラリを持つ数値解析用ソフトウェアであり，その中で用いられるプログラミング言語の名称でもある．MATLAB を利用することにより，汎用プログラミング言語を用いるよりも短時間で科学技術計算の問題を解決することが可能となる．

また，Mathworks 社は特定の分野の問題を解決することに特化した MATLAB 関数を集めた様々な Toolbox を用意している．MATLAB にこれら Toolbox を追加し，機能拡張することにより，信号・画像処理，通信システム，制御系設計，金融工学など，様々な分野での利用を可能としている．

Simulink は MATLAB 環境と統合して，システムをグラフィカルにモデリングするためのプラットフォームである．特定領域の問題を解決するための様々なブロックライブラリを持ち，それらブロックを結線することで，システムの直感的なモデリングを可能とする．また，設計されたモデルはシミュレーションが可能であり，Simulink の機能拡張ツールであ

る Real-Time Workshop を利用すれば，作成した Simulink モデルからソースコードを生成可能である．

Stateflow は Simulink を更に拡張し，ステートマシンとフローチャートを開発する環境を提供する．Stateflow チャートは状態・遷移を表す図形とそのアクション（動作）・条件式を記述するテキストによって構成され，設計者はこれらを用いてイベント駆動によるシステムの設計・構築が可能となる．Stateflow は Simulink 同様，シミュレーションが可能であり，また機能拡張ツールである Stateflow Coder を導入することにより，自動的にソースコードを生成することができる．

ここで，Simulink，および Stateflow により作成される .mdl ファイルは構造化文書であることを述べておく．すなわち，機能追加は MATLAB/Simulink 上で行うのではなく，ウィーバーにより .mdl ファイルを直接変更・修正する．そのため，アスペクトにより部品化した機能を追加するウィーバーは，.mdl ファイルを変更・修正するものとなる．そのプログラムの作成は今後の課題とし，本論文での機能追加は .mdl ファイルに対して手作業により行っている．

6. 研究例題：自動ライト制御システム

本研究では，様々なバリエーション機能を持つシステムの例題として，車載 BCU（Body Control Unit）の 1 つである自動ライト制御システムを取り扱う．仕様書からシステムを分析し，バリエーション機能に捉われない共通仕様モデルを MATLAB/Simulink/Stateflow により作成する．一方，バリエーション機能部分はアスペクトとして部品化を行い，共通仕様モデルへ機能追加を行う．

6.1 自動ライト制御システムの仕様とバリエーション機能

自動ライト制御システムのアーキテクチャ構成は図 5 のようになっており，主要なコンポーネントとして SensorInputConditioning，LampControl，ModeSelection から構成される．以下，それらコンポーネントの機能と，内在するバリエーション機能について概説する

6.1.1 SensorInputConditioning

このコンポーネントは光学センサから A/D 変換されたデジタル値を入力とし，正規化する．Sensor Input Control はアクセサリ・スイッチ，イグニッションスイッチの状態を入力として受け取ることにより光学センサの ON/OFF を決定し，コンポーネント Lamp Control へその状態を出力する．Input Normalization は入力された光学センサのデジタル

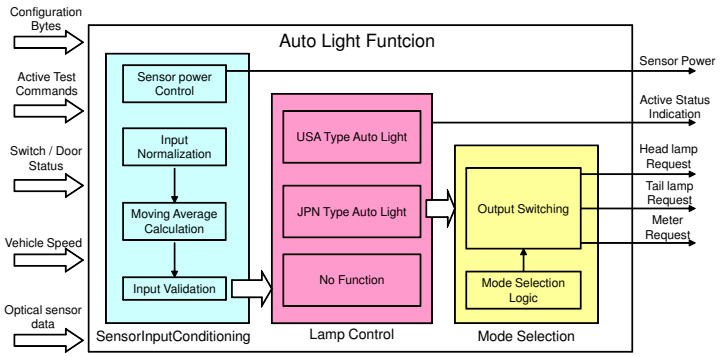

図 5 自動ライト制御システムのアーキテクチャ構成
Fig. 5 Architecture of Auto Light System.

値を関数により正規化し，Moving Average Calculation はその平均値を計算する．Input Validation は正規化・平均化された光学センサ値を，光学センサが ON の状態になってから数ミリ秒経過した値のみを適切な値として採用する．ここで正規化，および平均化された光学センサ値はコンポーネント Lamp Control へ出力される．

Input Normalization で用いられる正規化関数は，部品メーカにより決定されるバリエーション機能である．このバリエーション機能は，関数が異なるだけであり，かつシステム内では 1 度しか扱われない．

6.1.2 Lamp Control

このコンポーネントは図 5 にあるように，モジュール USA Type Auto Light，JPN Type Auto Light，No Function から構成される．しかしながら，これらモジュールの本質的な機能は同一であり，ヘッドランプ，テイルランプの ON/OFF を決定する．USA Type Auto Light は北米向け，JPN Type Auto Light は国内向けのモジュールであり，環境の違いからその機能にはバリエーションが発生している．機能の違いを簡単に述べれば，USA Type モジュールは，街灯が少ない環境から，エンジン停止から数分後にヘッドライトを消灯するためのタイマ機能が付されている．一方で，JPN Type モジュールにはトンネルが多い環境から，ライトの制御はより複雑なものとなっている．なお，No Function は，USA Type Auto Light モジュール，JPN Type Auto Light モジュールのいずれの機能も不要な場合に選択される．

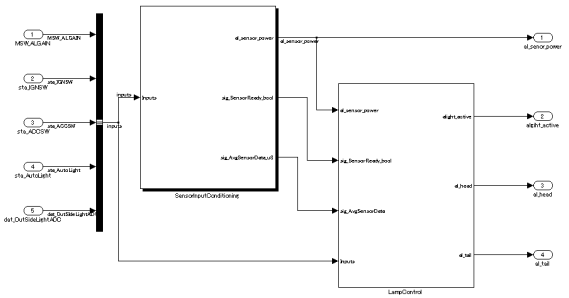

図 6 自動ライト制御システムの共通仕様モデル
Fig. 6 Auto Light Common System Model.

また，Lamp Control 内には輸出先の違いによるバリエーション機能とは他に，ユーザの嗜好により決定されるバリエーション機能が存在する．その機能は，ライトを制御するための閾値であり，USA・JPN Type Auto Light モジュール内の至るところに存在する．

6.1.3 Mode Selection

Mode Selection はコンポーネント Lamp Control において，JPN Type Auto Light を選択した場合にのみ有効となる．このコンポーネントはテストのために配置されている．

6.2 共通仕様モデル

自動ライト制御システムの，バリエーション機能に捉われない共通仕様モデルを構築する（図 6）．この共通仕様内に組み込んだ入出力シグナル，および機能は，次の基準に従っている．

- 入出力シグナル：共通仕様モデルには，バリエーション機能に関わらずシステムで利用される入出力シグナルのみを与える．
- 共通機能：共通仕様モデルには，バリエーション機能に依存しない部分の機能を与える．

まず入出力シグナルでは，現存するすべてのバリエーション機能で利用されるシグナルを与えることは好ましくない．それはすなわち，あるバリエーション機能では利用されるが，他のバリエーション機能では利用されないシグナルを与えてしまうことを意味する．不必要なシグナルを与えることは，システムを煩雑にしてしまう．よって，本研究では，バリエーション機能に関わらず利用される入出力シグナルを共通仕様モデルに与えている．

次に，共通仕様モデルに与えられる機能は，入出力シグナルと同様に，バリエーション機能に関わらず必要な機能のみを追加している．具体的には，コンポーネント Sensor Input Conditioning 内のモジュール Sensor Control，Moving Average Calculation，Input Validation である．コンポーネント Lamp Control については，その入出力関係のみを定義し，その機能は具体的に定めていない．なお，現段階でシステムに共通の機能として与えている部分にバリエーション機能が発生した場合，共通仕様モデルを変更・修正しなければならないことに注意しなければならない．

```
<aspect_name = "aspect-Macker1">
  <pointcut name = "pointcut-Macker1">
    <Model = "Stateflow" blockname = "transition"
             Tag = "Pointcut Input Formula" >
  </pointcut>

  <advaice name = "advaice-2Macker1"  Type = "add-Parameter">
    <String>
      Lvar_TempSensorData_s16 =  (CONST_NORM_OFFSET_33
          + (CONST_NORM_CORR_222 – dat_OutSideLightADC)
          * CONST_NORM_MUL_100 / Lvar_AL_Gain_u8)
    </String>
  </advice>
</aspect>
```

図 7 アスペクト記述例
Fig. 7 Example of Aspect description.

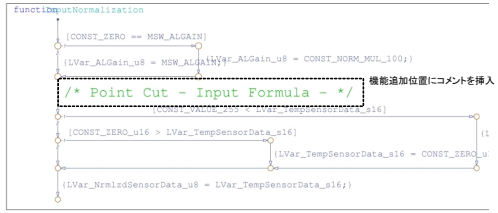

図 8 Stateflow モデル：コメント “Point Cut - Input Formula -” を付すことにより機能追加位置を特定
Fig. 8 State flow Model：Specifing pointcut by adding comment “Point Cut - Input Formula -”.

6.3 バリエーション機能の部品化

バリエーション機能の部品化の例として，以下の 2 つのバリエーション機能の部品化について述べる．

- メーカ依存部分：Input Normalization モジュールの正規化関数
- 地域特有部分：USA Type モジュール

メーカ依存部分のバリエーション機能は，コンポーネント Input Normalization の，光学センサにより得られた値を正規化するための関数であり，他のバリエーション機能と比較すると小さな機能である．図 7 は，正規化関数を追加するためのアスペクト記述例を表しており，図 8 は機能を追加するコンポーネント Sensor Input Conditioning の Stateflow モデルの一部分である．Stateflow モデルはアスペクトの要素であるポイントカットの指定が困難である為，現状では，図 8 中に示すように，機能を追加する場所にコメントを付すことで対応している．

次に，地域特有部分のバリエーション機能として USA Type モジュールを挙げる．地域特有部分のバリエーション機能としては 2 つのモジュール USA Type，JPN Type があり，これらは共通の機能としてヘッドランプ，テイルランプの制御を行う．従って，USA Type

Vol.2010-EMB-18 No.9
2010/8/10

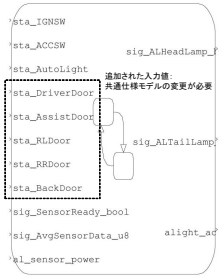

図 9 USA Type モジュール
Fig. 9 USA Type Module.

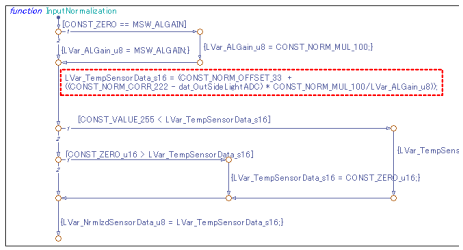

図 10 機能追加後の Stateflow モデル
Fig. 10 Stateflow Model after adding function.

モジュール，JPN Type モジュール に共通の機能を抽出したモジュールを作成し，必要に応じて機能追加を行うことも可能である．しかし，将来的に異なるバリエーション機能が発生し，共通機能が変更となった場合，既に作成した部品を変更する必要となる可能性がある．これは，既に作成した部品を再利用することが出来なくなることを意味するので，好ましいことではない．以上のことから，USA Type モジュール，JPN Type モジュールに共通な機能を抽出して共通仕様モデルに追加することはせず，USA Type モジュール，JPN Type モジュールそのものを部品化することを選択した．

USA Type モジュールの部品化は，MATLAB/Simulink を用いて USA Type モジュールを作成し，モデルから必要な部分を抽出することにより行っている．図 9 は MATLAB/Simulink により構築した USA Type モジュールであり，共通仕様モデルには存在しない入力シグナルが存在している．これはすなわち，機能追加の際には，USA Type モジュールを追加するだけでなく，共通仕様モデルの入力関係を修正しなければならないことを意味する．

6.4 バリエーション機能の追加

6.3 節で述べたバリエーション機能部品を共通仕様モデルに追加する．ただし，共通仕様モデルにアスペクトを追加する為のウィーバーの作成は今後の課題であり，今回は定められたポイントカット，およびアドバイスに従って，手作業による機能追加を行う．さらに，以下に述べるバリエーション機能は，異なるコンポーネントに存在するので，どの順序で機能追加を行ってもアスペクト，およびアスペクト数に変わりはない．すなわち，これらの部品には相互独立性が保たれている．

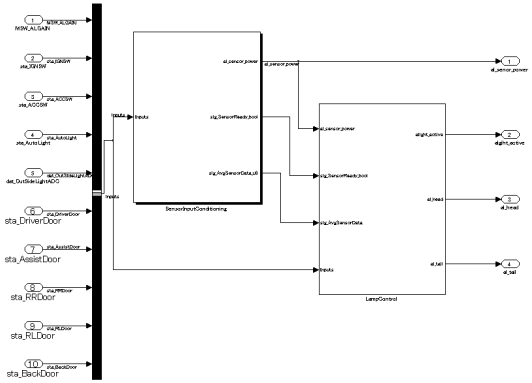

図 11 機能追加後の自動ライト制御システムモデル
Fig. 11 A Simulink model of Auto Light System after adding functions.

まず，Input Normalization モジュールの正規化関数の追加を行った結果を述べる．ポイントカット，アドバイスは図 7 に記したものである．それらを追加して再度読み込んだ結果，関数が追加されていることが分かる (図 10).

次に，USA type モジュールの機能追加を行った結果を述べる．USA Type モジュールを追加するためには，6.3 節で述べたように，必要な入力シグナルを共通仕様モデルに追加しなければならない．具体的にはシステム全体に対する入力シグナルの追加，入力シグナル数の変更，および追加した入力シグナルとそれらを入力値とするブロックへの結線，およびブロックのパラメータの変更である．これら入力シグナルの追加に伴う共通仕様モデルの変更，および USA Type モジュールの追加を含めて，USA Type モジュールを追加する為に必要なアスペクトの数は 43 個 である．

USA Type モジュールを追加した Simulink モデルを図 11，図 12 に示す．図 11 は USA Type モジュールを追加した後の自動ライト制御システム全体の Simulink モデル，図 12 は，サブシステム Lamp Control の Simulink モデルである．

6.5 相互独立性を有する部品化アスペクトの存在証明

図 13 は，Input Normalization モジュールの正規化関数，USA Type モジュールの追加を異なる順序で行った 2 つのモデル AutoLight_Formula_USA.mdl，および AutoLight_USA_Formula.mdl を diff コマンドを利用して差分をとった結果である．コマンドラ

Vol.2010-EMB-18 No.9
2010/8/10

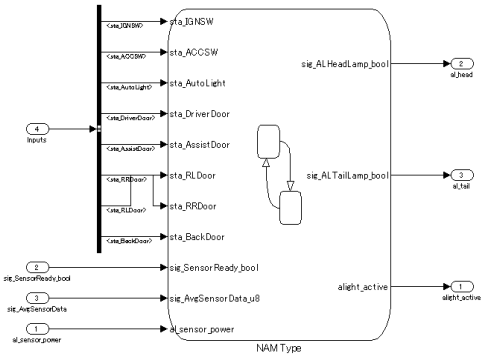

図 12 機能追加後のモジュール Lamp Control モデル
Fig. 12 A Simulink model of Lamp Control after adding functions.

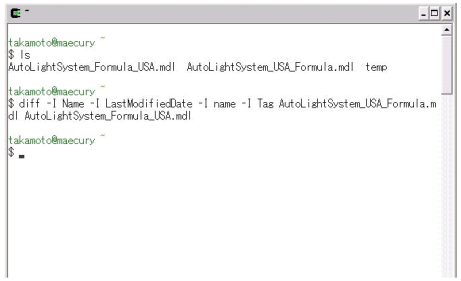

図 13 異なる順序で機能追加を施したモデル間の差分
Fig. 13 Difference between models adding functions in different order.

インには何も表示されないことから，これらのモデルが同一であることが分かる．これにより，本研究の成果として，順序を意識せずに異なるバリエーション機能を追加することが可能であることが示された．

7. 終わりに

本研究では，複雑化・多様化した要求から発生した複数のバリエーション機能を持つシステムの設計効率化を図るため，アスペクト指向に基づいてバリエーション機能を部品化すること提案した．その方針はシステム共通部分とバリエーション機能を分離，独立させ，バリエーション機能をアスペクトにより部品化し，共通仕様モデルに対して必要な機能を追加するものである．本研究の成果として，異なるバリエーション機能部品間で，順序に依存せず機能追加を可能とする相互独立性を保つことを示した．以下に，今後の課題を述べる．

7.1 アスペクト指向記述法の確立，ウィーバーの作成

現段階ではバリエーション機能部品，すなわちアスペクトの追加は手作業で行っているが，将来的には自動化を行う．そのために，ジョインポイントを洗出し，ポイントカット，アドバイスの記述法を確立させ，ウィーバーの作成を行う．

7.2 より複雑なシステムへの対応

本研究で述べた方法論が真に実用的であるのかを検証しなければならない．そのために，より複雑なバリエーション機能を持つシステムへの適用を行う．

参 考 文 献

1) Cristian Dziobek et al.: Functional Variants Handling in Simulink Models, Proc. of MathWorks Automotive Conference (2008).
2) The AspectJ Project: http://www.eclipse.org/aspectj/
3) Olaf Spinczyk et al.: AspectC++: an aspect-oriented extension to the C++ programming language, ACM International Conference Proceeding Series, Vol.21, pp. 53-60 (2002).
4) 鵜林尚靖ほか：アスペクト指向に基づく拡張可能な MDA モデルコンパイラ，第 3 回 SPA サマーワークショップ (2004).
5) Tatsuya Hoshino et al.: Communication Model Exploration in Aspect-Oriented Executable UML, Proceedings of International Conference on Applied Computing 2009, Vol.2, pp.129-134 (2009).
6) Mathworks: http://www.mathworks.com/