アスペクト指向アーキテクチャ設計のための横断的関心事抽出に関する研究
− ロバストネス分析を用いた要求とアーキテクチャの対応付け −

2007MI018 江坂 篤侍　　2007MI040 服部 裕介　　2007MI116 熊崎 陽介
指導教員　野呂 昌満　　沢田 篤史　　蜂巣 吉成

1　はじめに

膨大で複雑なソフトウェアシステムが増加するにつれ，ソフトウェア開発において，アーキテクチャ設計はより重要な分野となった [2]．アーキテクチャは仕様モデルなどから設計され，システムの骨格や作り方を定める．アーキテクチャ構築方法の中でも，アスペクト指向技術が注目されている．アスペクト指向技術はシステムを複数の視点で捉え，これまでのソフトウェア開発技術では解決し辛かった問題に対処することが出来る [6]．

横断的関心事とアスペクト指向アーキテクチャの関係は必ずしも明確ではない．このことから，アスペクト指向アーキテクチャ設計において，横断的関心事からのモジュール分割の決定や，モジュール構造からの横断的関心事の識別は困難である．

本研究の目的は仕様モデルと横断的関心事の対応関係を明確にすることである．これにより，アスペクト指向アーキテクチャ設計を支援する．

本研究ではロバストネス分析 [1] の結果を仕様モデルとし，アーキテクチャスタイル [2] を手がかりに横断的関心事との対応関係を明確にする．ロバストネス分析は仕様記述であるユースケース記述を分析し，ロバストネス図を用いてシステムの基本的な構造を表す．このロバストネス分析の結果の構文からモジュール分割を決定し，アーキテクチャを設計する．我々は，アーキテクチャスタイルは非機能要求に関する関心事の設計解であると認識している．そこで，アーキテクチャスタイルを手がかりにロバストネス分析の結果とアーキテクチャスタイルの実現する関心事を対応付ける．関心事との対応関係によりロバストネス分析の結果からアスペクト指向アーキテクチャ設計が可能になり，アスペクト指向アーキテクチャ設計を支援出来ると考えた．

ロバストネス分析の結果とアーキテクチャスタイルに構文的な関係があるとの仮説を立てた．仮説よりロバストネス分析の結果の構文から，アーキテクチャスタイルの実現する関心事の識別や，関心事からロバストネス分析の結果の構文を決定出来ると考えた．

ロバストネス分析の結果の構文の要素に対して拡張を行う．本来のロバストネス分析の結果の構文のみではアーキテクチャスタイルと対応付けることは困難であった．そこで，構文の要素を拡張することで，構文の要素に意味を対応付けた．これにより，意味も考慮してロバストネス分析の結果の構文とアーキテクチャスタイルを対応付けることが可能となった．

研究手順は次の通りである．まず，ロバストネス分析の結果の構文の要素に対しての拡張を定義する．次に仮説からロバストネス分析の結果とアーキテクチャスタイルとの対応関係を提案する．最後に提案する対応関係を事例を用いて検証する．

本研究の結果として，仕様モデルから横断的関心事を識別し，それによるモジュール分割を決定することで，アスペクト指向アーキテクチャ設計が可能となった．このことから，仕様モデルと横断的関心事の対応関係が明確になり，アスペクト指向アーキテクチャ設計を支援出来たと考える．

本研究室は組込みソフトウェアのためのアスペクト指向ソフトウェアアーキテクチャスタイル (以下，E-AoSAS++)[5] を提案している．本研究のアスペクト指向アーキテクチャの記述には E-AoSAS++ を用いた．

2　背景技術

ロバストネス分析，アーキテクチャスタイルについて説明を行う．

2.1　ロバストネス分析

ロバストネス分析とは，オブジェクト指向の分析方法である．ユースケースに記述された機能を実現するために必要な要素とその役割を分析し，ロバストネス図を用いて表現する．ロバストネス図は以下の 3 種類のコンポーネントとユースケースに登場するアクタとその関連による構文で構成される．図 1 はロバストネス図で用いるコンポーネントである．

- バウンダリ
  システムとそのアクタ間の作用をモデル化
- エンティティ
  長時間存在する，永続性のある情報をモデル化
- コントロール
  ほかのコンポーネントの操作をモデル化

図 1　ロバストネス図のコンポーネント

2.2　アーキテクチャスタイル

アーキテクチャスタイルは，よくあるシステムの構造を分類したものであり，コンポーネントとコネクタの種類とセマンティクスモデルで表現される．アーキテクチャスタイルはその構造により，非機能特性を実現する．代表的なものに Shaw と Garlan の提唱する 7 つのアーキテクチャスタイル [2] がある．

- コンポーネント
  計算主体

- コネクタ
  コンポーネント間の相互作用の記述
- セマンティクスモデル
  全体の特性を決定付ける，部分的な特性

3 仕様モデルと横断的関心事との対応関係

本研究は仕様モデルと横断的関心事の対応関係を明確にする．これにより，アスペクト指向アーキテクチャ設計を支援する．我々は，仕様モデルのロバストネス分析の結果とアーキテクチャスタイルに構文的な関係があるとの仮説を立てた．仮説から，ロバストネス分析の結果の構文とアーキテクチャスタイルの構文，アーキテクチャスタイルの持つ意味との対応関係を明確にする．我々は，アーキテクチャスタイルの持つ意味が存在するシステムには，そのアーキテクチャスタイルの実現する関心事が存在すると考える．また，アーキテクチャスタイルは，その構文により関心事を実現していると考える．ロバストネス分析の結果に対して構文の拡張を行うことで，ロバストネス分析の結果とアーキテクチャスタイルとの対応関係を明確にする．対応関係を明確にすることで，我々はロバストネス分析の結果の構文から，横断的関心事の識別と横断的関心事によるモジュール分割の決定が出来ると考えた．

3.1 ロバストネス分析の結果の構文の拡張

ロバストネス分析の結果の構文の要素に対する拡張を定義する．本来のロバストネス分析の結果の構文のみではアーキテクチャスタイルとの対応付けは困難であった．そこで，構文の要素に対して拡張を行うことで，構文の要素に意味を対応付ける．これにより，意味も考慮して構文をアーキテクチャスタイルと対応付けることが可能となった．我々は，定義や事例からロバストネス分析の結果の構文の要素に対応しうる意味を分析し整理した．ステレオタイプと 4 種類の矢印を用いてこの整理した意味を構文の要素に対応付ける．

対応付ける意味とその分類方法を表 1 に示す．

表 1　ロバストネス分析の結果に対応する意味と拡張案の一覧

構文要素	分析した意味	拡張案
コントロール	データの流れ	<<データ通信>>
	データの入力	
	データ出力	
	データ保存	
	データの変換	<<データ処理>>
	データの更新	
	データの生成	
	データに依存する判断	<<データ制御>>
	データの監視	
	イベントの流れ	<<イベント通信>>
	イベントに依存する判断	<<イベント制御>>
	イベントの監視	
バウンダリ	ユーザからのデータ入力	<<ユーザ>>
	ユーザからのイベント入力	
	外部システムからの情報入力	<<ハードウェア / ソフトウェア>>
	外部システムへの情報出力	
	外部システムからのイベント入力	
	外部システムへのイベント出力	
エンティティ	処理感の一時的なデータ	<<一時データ>>
	システムに存在し続けるデータ	<<永続データ>>
関連線	一方向のデータフロー	実線片方向矢印
	双方向のデータフロー	実線双方向矢印
	一方向のイベントフロー	破線片方向矢印
	双方向のイベントフロー	破線双方向矢印

3.2 アーキテクチャスタイルの持つ意味との対応関係

アーキテクチャスタイルの持つ意味をロバストネス分析の結果の構文と対応付ける．我々はアーキテクチャスタイルの持つ意味が存在するシステムには，そのアーキテクチャスタイルの実現する関心事があると考えている．そこで，アーキテクチャスタイルの持つ意味をロバストネス分析の結果の構文のパターンとして定義した．アーキテクチャスタイルのコンポーネントとコネクタの持つ責務や，それらの関連をロバストネス分析の結果の構文と対応付けた．これにより，アーキテクチャスタイルの持つ意味とロバストネス分析の結果の構文が対応付くと考えた．アーキテクチャスタイルの持つ意味をロバストネス分析の結果の構文と完全に対応付けることは困難であった．よって，既存のロバストネス分析の結果の構文に対するパターンの照合により導き出せるのは関心事が存在する箇所の候補である．このことから，パターンと一致する構文が現れた時，アーキテクチャスタイルを適用するシステムであるかを設計者が判断する．このように，対応関係を用いて，ロバストネス分析の結果から横断的関心事の識別が出来ると考えた．

各アーキテクチャスタイルの定義からコンポーネントやコネクタの持つ責務を整理し，その責務はロバストネス分析の結果の構文にどのように現れるか整理した．対応関係を表 2 に示す．

表 2　アーキテクチャスタイルの実現する関心事との対応関係

Architecture style	Component and Conector	Feature
Pipes and Filters	Filter	データ処理，データ制御，一時データ(但しデータ処理，データ制御いずれか必須)
	Pipe	データ通信，一時データ(但しデータ通信必須)
Layerd	Layer	何でもあり(但し異なるステレオタイプのバウンダリが同じレイヤになることはない)
	Protocol	データ通信，イベント通信
Object Oriented	Object	何でもあり(但しデータ構造もしくはバウンダリ必須)
	Massage Passing	データ通信，イベント通信
Event-Based	Module	他のアーキテクチャスタイルのコンポーネント
	Implicit Invocation	イベント通信，イベント制御
Interpreter	Program being interpreted	データ構造，外部ソフトウェア境界，外部ハードウェア境界，ユーザインタフェース
	Program state	一時データ
	Internal interpreter state	データ制御，データ処理，一時データ，データ構造(但しデータ制御必須)
	Simulated interpretation engine	コントロール，バウンダリ
	Data access	データ通信
Blackboard	Shared Data	データ構造，データ通信(但しデータ構造必須)
	Knowledge Source	何でもあり
	Controller	データ制御，イベント制御(但しデータ制御必須)
	Various connection	データ通信，イベント通信
Process Control	Control	永続データ，データ制御，イベント制御(但しデータ制御とデータ構造の関連必須)
	Process	何でもあり
	Control loop paradims	データ通信，イベント通信

Feature	Syntax
データ処理	<<データ処理>>のコントロールとデータ通信の連続
データ制御	<<データ制御>>のコントロールとデータ通信の連続
イベント制御	<<イベント制御>>のコントロールとイベント通信の連続
データ構造，永続データ	<<永続データ>>のエンティティ
一時データ	<<一時データ>>のエンティティ
データ通信	<<データ通信>>のコントロールと実線関連線の連続
イベント通信	<<イベント通信>>のコントロールと破線の連続
外部ソフトウェア境界	<<ソフトウェア>>のバウンダリ
外部ハードウェア境界	<<ハードウェア>>のバウンダリ
ユーザインタフェース	<<ユーザ>>のバウンダリ
コンポーネント	アーキテクチャスタイルのコンポーネント

3.3 アーキテクチャスタイルの構文との対応関係

ロバストネス分析の結果の構文とアーキテクチャスタイルの構文を対応付ける．我々は，アーキテクチャスタイルはその構文により関心事を実現していると考えた．よって，アーキテクチャスタイルの構文をロバストネス分析の結果の構文のパターンとして定義した．アーキテクチャスタイルのコンポーネントとコネクタ，それらの関連をロバストネス分析の結果の構文と対応付けた．これにより，アーキテクチャスタイルの構文とロバストネス分析の結果の構文が対応付くと考えた．アーキテクチャスタイルの実現する関心事を識別した時，ロバストネス分析の結果の構文を，そのアーキテクチャスタイル

の構文に対応付く構文にする．これにより，ロバストネス分析の結果の構文から，その関心事を実現するアーキテクチャスタイルのモジュール分割を決定することが出来ると考えた．複数の関心事を識別した時，可能であれば複合アーキテクチャとして実現することで，より適したアーキテクチャ設計を行うことが出来る．

各アーキテクチャスタイルの定義から，コンポーネントやコネクタはロバストネス分析の結果の構文にどのように現れるか整理した．対応関係を表 3 に示す．

表 3　アーキテクチャスタイルの構造との対応関係

Architecture style	Component and Connector	Structural element
Pipes and Filters	Filter	データ入力，データ処理，データ制御，データ出力
	Pipe	データ入力，バッファリング，データ出力
Layerd	Layer	同じ抽象度の層で分割出来れば何でも良い
	Protocol	通信
Object Oriented	Object	データ構造，手続き
	Massage Passing	通信
Event-Based	Module	他のアーキテクチャスタイルのコンポーネント
	Implicit Invocation	イベント制御
Interpreter	Program being interpreted	データ構造
	Program state	手続き
	Internal interpreter state	データ構造
	Simulated interpretation engine	データ入力，データ処理，データ出力，制御
	Data access	データ通信
Blackboard	Shared Data	共有データ
	Knowledge Source	手続き
	Controller	制御
	Various connection	通信
Process Control	Process	データ/外部入力，プロセス変数，手続き，データ/外部出力
	Control	データ入力，制御，設定値，データ出力
	Control loop paradims	データ通信

Structural element	Syntax
手続き	データ処理，インタフェース，データ構造，制御
データ処理	複数の<<データ処理>>のコントロールと<<一時データ>>のエンティティと実線矢印
データ構造/設定値/プロセス変数 / 共有データ	<<永続データ>>のエンティティ
通信	データ通信，イベント通信
データ入力/出力	データ通信
外部入力/出力	<<ハードウェア>>のバウンダリ
バッファリング	データ通信
データ通信	<<一時データ>>のエンティティ，<<データ通信>>のコントロール，実線矢印のいずれか又は組み 合わせ
イベント通信	<<イベント通信>>のコントロール，破線矢印のいずれか 又は 組み 合わせ
制御	データ制御とそこから出る実線矢印，又はイベント制御とそこから出る破線矢印
イベント制御	<<イベント制御>>のコントロールと，そこから出る破線矢印
インターフェース	バウンダリ

これらの要素は外部に委任することもあり，その時は<<ハードウェア>>のバウンダリとして現れる．

4　事例検証

提案する対応関係を，MDD ロボットチャレンジ 2010 の仕様 [3](以下，競技仕様書) に基づく自動航行飛行船制御ソフトウェア (以下，飛行船) を事例として検証した．

4.1　アスペクト指向アーキテクチャ設計手順

事例に対して行ったアスペクト指向アーキテクチャ設計手順を示す．まず，仕様モデルに対しロバストネス分析を行った．次に，分析結果から横断的関心事の識別と，関心事を実現するアーキテクチャスタイルのモジュール分割を決定し，アスペクト指向アーキテクチャの設計を行った．

ロバストネス分析の際に我々は 3 つの工程を可能なものから繰り返し行う．

工程 1. 構文に対する拡張
　ロバストネス分析の結果の構文の要素に対応する意味は，仕様から分かる場合と，アーキテクチャスタイルを適用する際に決まる場合がある．ロバストネス分析の結果の構文の要素に対応する意味が分かり次第，拡張を行う．

工程 2. ロバストネス分析の結果の構文からの関心事の識別
　アーキテクチャスタイルの持つ意味との対応関係を用いて，横断的関心事を識別する．

工程 3. ロバストネス分析の結果の構文の決定
　アーキテクチャスタイルの構文との対応関係を用いて，ロバストネス分析の結果の構文を決め，モジュール分割を決定する．

これらにより，ロバストネス分析の結果の構文から横断的関心事の識別とモジュール分割の決定を行い，アスペクト指向アーキテクチャの設計が可能となると考えた．

4.2　横断的関心事の識別とモジュール分割の決定

3 つの工程を用いてロバストネス分析の結果から横断的関心事の識別とモジュール分割の決定を行う．ロバストネス分析の結果の構文から Blackboard，Process Control，Pipes and Filters，Event-Based の実現する関心事を横断的関心事として識別した．さらにロバストネス分析の結果の構文を変更し，その構文からアーキテクチャスタイルのモジュール分割を決定することでこれらの関心事を複合アーキテクチャとして実現した．例として Blackboard を用いて説明する．

■Blackboard　工程 2 の結果，ロバストネス分析の結果の構文から Blackboard の関心事を識別した．さらに工程 3 の結果，ロバストネス分析の結果の構文から Blackboard のモジュール分割を決定した．永続データのステレオタイプのエンティティが Shard Data，データ制御のステレオタイプのコントローラが Control，Control が活性化する箇所が KnowledgeSource となる．

最終的にロバストネス分析の結果から識別した関心事を図 2 に示す．識別した関心事を実現するモジュール分割を決定した．このモジュール分割は Process Control, Event-Based, Blackboard, , Pipes and Filters の複合アーキテクチャとなっている．

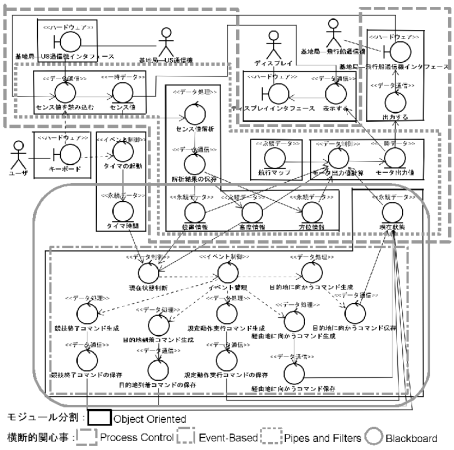

図 2　横断的関心事を識別したロバストネス分析の結果

4.3　E-AoSAS++ に基づくアーキテクチャ設計

E-AoSAS++ に基づいてアーキテクチャ設計を行った．識別した横断的関心事によって分割されたモジュールをそのアーキテクチャスタイルのアスペクトとし，アスペクト指向アーキテクチャ設計を行った．その結果の一部を図 3 に示す．

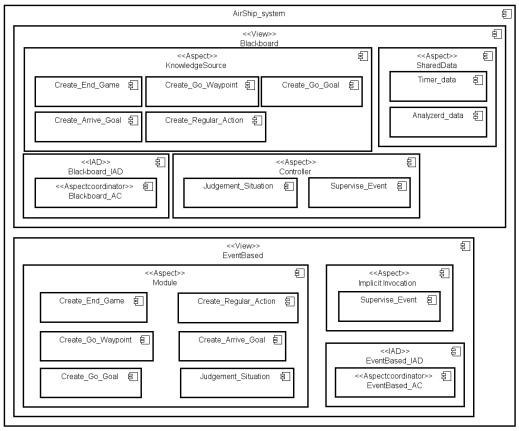

図 3　アスペクト指向アーキテクチャ設計結果
(コンポーネント図一部抜粋)

5　考察

5.1　ロバストネス分析の結果の構文の拡張の考察

アーキテクチャスタイルとロバストネス分析の結果を対応付けるためにロバストネス分析の結果の構文の拡張を定義した．拡張表現はステレオタイプや関連線の種類分けにより行う．拡張表現は複数の事例から構文の要素に対応する意味を分析しそれを基に定義した．よって定義した拡張表現は，ロバストネス分析の結果のコンポーネントや関連線に対応する意味を構文として付加するものである．以上のことから，ロバストネス分析の本来の意義が拡張表現により失われることなくアーキテクチャスタイルと対応付けることが出来た．

5.2　識別した関心事に関する考察

本事例において識別した Process Control, Event-Based, Blackboard, Pipes and Filters の実現する関心事が妥当であるか以下に考察する．

■Process Control　飛行船は外乱の影響を受ける問題がある．Process Control を適用することで，飛行船のモータへ制御値を送り，飛行船を制御し，この問題を解決する．

■Blackboard　目的地までの自動航行で飛行船が実行する動作は複数存在し，実行する動作が現在の状態に依存する問題がある．Blackboard を適用することで，現在の状態から実行する動作を決定し，目的地まで試行錯誤しながら辿り着かせ，この問題を解決する．

■Event-Based　飛行船の動作の実行する順序関係は現在の飛行船の状態に複雑に関係する．Event-Based を適用することで，動作の実行を一括に管理する．

■Pipes and Filters　飛行船はセンサの入力からモータへの出力までの間に，センス値の解析やモータ出力値の計算などの複数の処理を行う．Pipes and Filters を適用することで，これらの処理毎の独立性を実現する．

以上より，各アーキテクチャスタイルの適用は妥当であると考えた．識別したアーキテクチャスタイルの実現する関心事は妥当であることから，アーキテクチャスタイルの実現する関心事との対応関係は妥当である．

5.3　仕様モデルと横断的関心事の対応関係の考察

ロバストネス分析の結果とアーキテクチャスタイルに構文的な関係があるとの仮説を立て，対応付けを行った．これにより，仕様モデルと横断的関心事の対応関係が明確となった．対応関係により，ロバストネス分析の結果から横断的関心事を識別し，アスペクト指向アーキテクチャ設計が可能となった．よって，アスペクト指向アーキテクチャ設計を支援することが出来たと言える．

5.4　関連研究と比較した本研究の有用性

本研究室では過去に関連研究 [4] が行われている．これはロバストネス分析の結果と横断的関心事を対応付け，仕様モデルとアスペクト指向アーキテクチャの対応関係の明確化を試みている．本研究とは違い，横断的関心事の基準や，関心事から構造を決定する指針がない．本研究は横断的関心事の基準をアーキテクチャスタイルとし，関心事の内容が明確であることから，過去の研究に比べ一般性が高いと考える．また，関心事から構造を決定する指針は，より洗練されたアスペクト指向アーキテクチャの設計が可能となる．以上のことから過去の関連研究に比べ本研究は有用性が高い．

6　おわりに

本研究では，アスペクト指向アーキテクチャ設計における問題点を挙げ，仕様モデルと横断的関心事との対応関係を明確にすることを目的とした．仕様モデルと横断的関心事の関係の仮説を立て，仮説から対応関係を提案し，飛行船制御ソフトウェアを用いて事例検証と考察を行った．その結果，仕様モデルと横断的関心事の対応関係が明確となった．

今後の課題として，本研究で提案する対応関係の一般性の確認と，洗練を他ドメインの事例を用いて行うことが挙げられる．

参考文献

[1] D. Rosenberg and K. Scott, *Use Case Driven Object Modeling with UML: A Practical Approach*, Addison-Wesley, 2001.

[2] M. Shaw, D. Garlan, *Software Architecture, Perspectives on Emerging. Discipline*, Prentice-Hall, Inc. , Upper Saddle River, New Jersey, 1996.

[3] MDD ロボットチャレンジ 2010 実行委員会 , “ MDD ロボットチャレンジ 2010 競技仕様書，” 2010; http://sdlab.sys.wakayama-u.ac.jp/mdd2010/MDD2010 regulation.pdf.

[4] 江場智文，大石早織, “ アスペクト指向に基づく飛行船制御ソフトウェア開発，” 2008 年南山大学卒業論文，2008

[5] 加藤大地, 蜂巣吉成, 沢田篤史, 野呂昌満, “ アスペクト指向に基づくソフトウェアアーキテクチャの文書化方式，知能ソフトウェア工学研究会 (KBSE), vol.108， no.449，pp55-60，2009.

[6] 野呂昌満, “ アスペクト指向プログラミング概観，” Seamail，vol.13，no11.