Vol.2010-SE-169 No.3
2010/7/22

ジョインポイント写像による ドメイン特化 AO 機構の開発手法

外村 慶二[†1] 鵜林 尚靖[†1]
中島 震[†2] 岩井 明史[†3]

ドメイン特化の AO 機構は，アスペクト記述の可読性，メンテナンス性，再利用性を高める上で有効である事が知られているが，しかしながら，その開発手法についてはあまり知られていない．本稿では，Web アプリケーション開発を対象としたドメイン特化の AO 機構である AOWP の開発をケーススタディとして，ドメイン特化の AO 機構の開発手順について述べる．本稿で提案する開発手法では，最初に，ドメインに特徴的な横断的関心事の振る舞いに基づいてドメイン特化のジョインポイントを設計する．そして，そのジョインポイントに基づいて汎用的な AO 機構の拡張としてドメイン特化の AO 機構を開発する．

A Domain-Specific AO Mechanism Development Method Based on Join Point Mapping

Keiji Hokamura,[†1] Naoyasu Ubayashi,[†1]
Shin Nakajima[†2] and Akihito Iwai[†3]

Domain-specific aspect-oriented (AO) mechanism is effective for improving readability, maintainability, and reusability of aspects. However, the development process of the mechanism is not clear. This paper proposes a new development process through a case study of development of AOWP which is a domain specific AO framework for Web application development. In the proposed process, we design a set of domain-specific join points by analyzing the behavior of domain-specific cross-cutting concerns. Then, we design a domain-specific AO mechanism based on the join points.

1. はじめに

特定のアプリケーション，及びドメインに特化したアスペクト指向 (AO) 機構は，アスペクト記述の可読性，メンテナンス性，再利用性を高める上で有効である[6),9),15)]．AspectJ 等のアスペクト指向プログラミング (AOP) 言語は，ロギング等の横断的関心事をアスペクトとしてモジュール化できる[10)]．アスペクトは，織り込み (weaving) という処理により，対象システムのプログラム構造に基づいて適用され，開発者が対象システムを変更する事なく新たな処理を対象システムに追加できる．しかし，その一方で，アスペクトが，深く対象システムのプログラム構造に依存してしまい，記述が複雑になったり，対象システムの変更で壊れたり，複数のシステムで再利用する事が難しいという問題点があった．幾つかの研究では，この問題を解決する為に，横断的関心事の振る舞いを適切に抽象化し，その振る舞いに作用するアスペクトを記述する事で，アスペクトと対象システムのプログラム構造の依存度を軽減する手法を提案している．

しかしながら，ドメイン特化の AO 機構を開発する為の一連の分析/設計/実装方法についてはあまり知られていない．6)，15) では，ドメイン特化の AO 機構と汎用的な AO 機構を適切に使い分ける事の重要性について述べられており，ドメイン特化の AO 機構を開発する上で，その必要性と，何に着目してドメイン特化の AO 機構を作るのかを検討する事が必要である．また，どのようにドメイン特化の AO 機構を設計するか，設計したドメイン特化の AO 機構をどのように実装するか，についても指針がある事が望ましい．

そこで，本稿では，Web アプリケーション開発を対象としたドメイン特化の AO 機構 AOWP[7)] の開発をケーススタディとして，ドメイン特化の AO 機構の設計から実装までの一連の開発手法について説明する．我々は，最初に Web アプリケーションに含まれる横断的関心事の振る舞いを分析し，それらの多くに共通する Web アプリケーションに固有なイベントとして HTTP リクエスト/レスポンスの授受 に着目して，ドメイン特化のジョインポイントと，それを取り扱う為のポイントカット&アドバイス機構とアスペクトのインスタンス生成機構の設計を行った．そして，それらの AO 機構を，汎用的な AO 機構 (AOWP

†1 九州大学
Kyushu University
†2 国立情報研究所
National Institute of Informatics
†3 株式会社デンソー
DENSO CORPORATION

では PHP の言語仕様に基づいた AO 機構) と対応付け，その拡張としてドメイン特化の AO 機構を実装した．また，本稿では，ケーススタディに基づいて，ドメイン特化の AO 機構の開発基盤としての，拡張可能な AO 機構に必要な性質について考察する．

以降，2 節で，ドメイン特化の AO 機構の必要性について，Web アプリケーション開発を例にして説明する．3 節で，我々が行った Web アプリケーション開発を対象としたドメイン特化の AO 機構の設計と実装について説明する．4 節で，ケーススタディに基づいて考察を行い，5 節で本稿をまとめる．

2. 問題意識

本節では，Web アプリケーションを例として，汎用的な AO 機構のみを利用したアスペクト記述の問題点について説明する．

2.1 ドメイン特化の AO 機構の必要性

図 1 に，汎用的な AO 機構のみを用いたアスペクト記述例を示す．このアスペクトは，PHP で実装された Web アプリケーションの全てのページアクセスに横断的に作用するアスペクトであり，一定時間以上 (ここでは 60 秒) Web サイトに滞在したユーザに，Web サイトの使いやすさに関するアンケートを表示する機能を実装している．このアスペクトは，PHP を対象とした AO フレームワークである AOWP[7] を用いて記述している．AOWP では，アスペクトを PHP のクラスとして記述し (1 行)，ポイントカットとアドバイスをフレームワークが提供するクラスを用いて定義する (3-21 行)．このアスペクトは，ユーザのページアクセスを取り扱うサーバ側の一連のプログラム (以降，リクエスト処理と呼ぶ) に対して作用し，(1) ページアクセス処理の前にユーザの滞在時間をチェックし，必要に応じて PHP の出力制御機能を有効にするアドバイス (7-10，24-34 行)，(2) 必要に応じて出力にアンケート表示を行う為の記述を追加する 2 つのアドバイス (12-15，17-21，35-41 行)，の計 3 つのアドバイスを定義している．図 1 の記述から，汎用的な AO 機構のみを用いてアスペクトを記述する時に，以下の 2 つの理由から記述が複雑になり，アスペクトの可読性やメンテナンス性が低下すると考えられる．

1 つ目のアスペクト記述が複雑になる理由は，汎用的な AO 機構ではドメインに特徴的なイベントを直接取り扱うことができない点である．汎用的な AO 機構では，特定のプログラミング言語の実行モデルに基づいて AO 機構作られており，ドメインに特徴的なイベントを特定のプログラミング言語の実行モデルに対応付けて間接的に取り扱う必要がある．例えば，図 1 では，ユーザが行うページアクセスという Web アプリケーションに特有のイベントを，PHP の言語仕様に基づく実行モデルに対応付けてポイントカットを記述している (3-5，18 行)．PHP では，ページアクセスを，ページアクセスに含まれる URL に対応したファイル名を持つスクリプトファイルを実行して処理する．その為，ページアクセスの前後に適用する処理を，スクリプトファイルの実行前後に作用する before アドバイスと

```
1  class QuestionnaireAspect extends AOWP_PerJoinPointAspect {
2    public function __construct() {
3      $scrExePC = new AOWP_ScriptExecutionPointcut('.*');
4      $ifFstScrExePC = new AOWP_IfPointcut('isFstScrExe()');
5      $allReqPC = $scrExePC->opAnd($ifFstScrExePC);
6
7      $startBufAdv = new AOWP_BeforeAdvice();
8      $startBufAdv->setPointcut($allReqPC);
9      $startBufAdv->setAdviceBody('_startOutputBuf');
10     $this->addAdvice($startBufAdv);
11
12     $endBufAdv1 = new AOWP_AfterAdvice();
13     $endBufAdv1->setPointcut($allReqPC);
14     $endBufAdv1->setAdviceBody('_addQues');
15     $this->addAdvice($endBufAdv1);
16
17     $endBufAdv2 = new AOWP_BeforeAdvice();
18     $exitCallPC = new AOWP_FunctionCallPointcut('^exit|die$');
19     $endBufAdv2->setPointcut($exitCallPC);
20     $endBufAdv2->setAdviceBody('_addQues');
21     $this->addAdvice($endBufAdv2);
22   }
23
24   protected function _startOutputBuf(AOWP_JoinPoint  $jp) {
25     if (session_id() == null)            start_session();
26     if (!isset($_SESSION['asked']))      $_SESSION['asked'] = false;
27     if (!isset($_SESSION['asking']))     $_SESSION['asking'] = false;
28     if (!isset($_SESSION['visitTime'])) $_SESSION['visitTime'] = time();
29     if (!$_SESSION['asked'] && time() - $_SESSION['visitTime'] > 60) {
30       $_SESSION['asked'] = true;
31       $_SESSION['asking'] = true;
32       ob_start();
33     }
34   }
35   protected function _addQues(AOWP_JoinPoint $jp) {
36     if ($_SESSION['asking']) {
37       $_SESSION['asking'] = false;
38       $_SESSION['asking'] = false;
39       $outputSource = ob_get_clean();
40       /* レスポンスにアンケート表示を追加 (省略) */
41 } } }
```

図 1 汎用的な AOP 機構を用いたアンケート表示アスペクト
Fig. 1 Questionnaire aspect using general-purpose AOP mechanisms

after アドバイスとして記述している (3-15 行)．また，PHP では，exit や die 関数を用いてリクエスト処理をプログラムの任意の箇所で終了できる為，ページアクセスの後に作用する処理を，それらの関数呼び出しの前に実行される before アドバイス (17-21 行) としても定義している．このようにドメイン特化のイベントに作用する横断的関心事を，特定のプログラミング言語の実行モデルに対応付けて記述する事で，アスペクト記述が複雑になり，可読性やメンテナンス性の低下の原因となる．

また，もう 1 つの複雑化の原因は，ドメインに特徴的な状態管理をアスペクト記述で取り扱うのが容易では無い点である．AspectJ 等の汎用的な AO 機構では，アスペクトのインスタンスを，特定のオブジェクトのインスタンス (AspectJ の perthis，pertarget) や，制御フロー (AspectJ の percflow，percflowbelow) 等に関連づける事ができ，目的の状態管理に応じてアスペクトのインスタンス生成方法を使い分ける事ができる．しかし，ドメイン特有の状態管理の中には，このようなプログラム実行に基づくアスペクトのインスタンス生成機構では容易に取り扱えないものがある．例えば，多くの Web アプリケーションは，負荷分散のような Web アプリケーション全体としての状態管理が必要な機能や，図 1 で示したユーザの滞在時間等のユーザごとの状態管理 (以降，ユーザセッションと呼ぶ) が必要な機能が数多く存在する[8]．これらの状態管理は，サーバ/クライアント間のメッセージ授受のプロトコルである HTTP に基づく状態管理である為，上で示したような汎用的なアスペクトのインスタンスの中で取り扱う事ができず，アドバイス記述の中でそれらを記述する必要があり，結果としてアドバイス記述が複雑になる．例えば，図 1 では，PHP のユーザセッションを取り扱う為の API を利用し，ユーザの最初の訪問時間，アンケートを表示したか，アンケートを表示する必要があるか，という 3 つの状態をユーザごとに管理している (25-29，36-38 行)．

2.2 本稿の目的

上で述べたような，ドメイン特化のコンセプトの取扱いに起因するアスペクト記述の複雑さを軽減する方法として，ドメイン特化の AO 機構を利用する事ができる[6),9),15)]．ドメイン特化の AO 機構では，アプリケーションやドメインに特徴的なイベントを直接取り扱う為のジョインポイントやポイントカット機構を定義しており，それらを用いる事でアスペクト記述の可読性や再利用性を高める事ができる．

本稿では，Web アプリケーション開発に特化した AO 機構である AOWP[7)] の開発をケーススタディとして，ドメイン特化の AO 機構の設計から実装までの一連の開発手順について説明する．本稿では，最初にドメインに特徴的なイベントを直接表すジョインポイントを設計し，そのジョインポイントに基づいてドメイン特化のポイントカット&アドバイス機構及びアスペクトのインスタンス生成機構を設計する．6)，9)，15)，16) では，特に，ドメイン特化のポイントカット機構の有効性について述べられているが，本稿では，上で述べたドメイン特化の状態管理の難しさを解決する為に，ドメイン特化のアスペクトのインスタンス生成機構についても開発し，その有効性を示す．

また，本稿では，ドメイン特化のジョインポイントを汎用的なジョインポイントと対応付け (本稿では，ジョインポイント写像と呼ぶ)，汎用的な AO 機構の拡張としてドメイン特化の AO 機構を実現する．なお，本稿で説明する開発手順は，AO 機構を用いた Web アプリケーション開発を対象とした言語指向プログラミング (Language-oriented programming)[17)] と考える事ができる．言語指向プログラミングでは，ソフトウェアの生産性，メンテナンス性，可読性，再利用性等を高める為に，(1) ドメイン特化のプログラミング言語を設計し，それを用いてシステムを記述する，(2) ドメイン特化のプログラム記述を，既存のプログラミング言語の記述に変換するコンパイラを作成する，という手順でシステムを構築する．本稿では，汎用的な AO 機構の織り込み機構 (weaver) を拡張して，織り込み時にドメイン特化のアスペクト記述を汎用的な AO 機構と対応付ける事で，ドメイン特化の AO 機構を実現する．

3. ケーススタディ

本節では，ドメイン特化の AO 機構の開発手順を説明する為のケーススタディとして，Web アプリケーションを対象としたドメイン特化の AO 機構 AOWP[7)] の開発について述べる．最初に，ドメイン特化のジョインポイントとなる Web アプリケーションに特徴的なイベントを決定する為のドメイン分析について述べる．そして，そのイベントに基づく Web アプリケーションに特化した AO 機構の設計について述べる．最後に，設計したドメイン特化の AO 機構を実装する為に行った汎用的な AO 機構の拡張について述べる．

3.1 ドメイン分析

最初に，ドメイン特化の AO 機構で取り扱うイベントを決定する為に，Web アプリケーションに含まれる横断的関心事の振る舞いについて分析を行った．ここでは，前節で述べた 2 つの問題点と対応する形で，(1) 多くの横断的関心事の作用点となる共通のイベントは何か，(2) 多くの横断的関心事で必要になる状態管理はどのようなものか，という 2 つの観点に基づいて分析を行った．分析対象となる Web アプリケーションに特徴的な横断的関心事としては，ユーザ認証[1),12)]，アクセス制御[1),12)]，バージョニング[12)]，動的画面生成[11)]，ア

表 1 リクエストジョインポイントのコンテキスト情報
Table 1 Context information of request join points

リクエストジョインポイントのコンテキスト情報	汎用的なジョインポイントのコンテキスト情報
リクエストURL	$_SERVER['REQUEST_URI']
フォームデータ	$_GET ($_POST)
クッキー	$_COOKIE
HTTPリクエストのヘッダ情報	$_SERVER

表 2 リクエスト処理のジョインポイント写像
Table 2 Join point mapping for request transaction

	in/out	汎用的なAO機構のジョインポイント
M1	in	in_script_execution(url)
	out	out_script_execution(url)
M2	in	in_script_execution(url)
	out	in_function_call(exit|die)

クセス解析[1]，負荷分散，入力値検証[12)–14)] 等が考えられる．

1 つ目の観点については，上の横断的関心事の多くがユーザのページアクセスに作用する処理である為，リクエスト処理を横断的関心事の共通の作用点として考える事ができる．例えば，動的画面生成はリクエスト処理の結果であるレスポンスに作用する関心事であり，また，入力値検証についてもユーザがページリクエスト時に送信したデータに対して行われる事が多い．また，2 つ目の観点については，上の横断的関心事の幾つかは，一連のリクエスト処理にまたがる状態管理を必要とする．例えば，ユーザ認証や 2 節で述べたアンケート機能等は，各ユーザの一連のページアクセスに作用する機能であり，負荷分散等は Web サーバ全体としての一連のページアクセスに作用する機能である．

上の分析に基づいて，我々は，ドメイン特化の AO 機構で取り扱うイベントとしてリクエスト処理に着目する事とした．以降，リクエスト処理を直接表すジョインポイント，それらを取り扱う為のポイントカット&アドバイス機構，一連のジョインポイントと関連付けたアスペクトのインスタンス機構，の順で AO 機構の設計を行う．

3.2 ドメイン特化の AO 機構の設計

(1) リクエスト処理を表すジョインポイント

最初に，リクエスト処理を表すジョインポイント (以降，リクエストジョインポイントと呼ぶ) を設計した．ドメイン特化のジョインポイントの設計では，(a) ドメイン特化のジョインポイントに含まれるコンテキスト情報，(b) ドメイン特化のイベントと汎用的な AO 機構のジョインポイントの対応関係 (ジョインポイント写像と呼ぶ)，の 2 つを定義する．

リクエスト処理を表すジョインポイントのコンテキスト情報は，Web におけるサーバ/クライアント間のメッセージ授受に使われる HTTP リクエストの仕様に基づいて，表 1 のように定義した．このコンテキスト情報は，以降のドメイン特化の AO 機構の設計全般で利用される．

ジョインポイント写像の定義では，ジョインポイントとなるドメイン特化のイベントと対応するプログラムコード上の区間 (開始点と終了点) を，汎用的な AO 機構のジョインポイントの開始点と終了点を用いて定義する．本稿では，リクエスト処理のジョインポイント写像を，表 2 のように定義した．表 2 では，リクエスト処理のイベントを 2 つのイベント区間 M1，M2 に対応付けて定義している．表 2 の (in/out)_script_execution は，スクリプトファイルの実行を表すジョインポイントの前後を表現しており，in_function_call(exit|die) は，exit もしくは die 関数の呼び出しを表すジョインポイントの前を表現している．2 節で述べたように，PHP では，ページアクセスに対して URL に対応したファイル名を持つスクリプトファイルが実行される為，M1 は，スクリプトファイルが最後まで実行された場合，M2 は，exit もしくは die 関数の呼び出しにより，リクエスト処理がスクリプトファイルの実行途中で中断される場合のイベント区間を表している．

(2) ドメイン特化のポイントカット記述子

次に，リクエストジョインポイントを取り扱う為のポイントカット記述子を定義する．ここでは，以下のシグネチャを持つ request ポイントカットを定義した．

```
1 request(url_pattern, form_data_patterns, cookie_patterns, header_patterns)
```

ここで，ポイントカット記述子の引き数は，表 2 のコンテキスト情報に基づいて定義しており，url_pattern はリクエスト URL を指定する文字列パターン，form_data_patterns，cookie_patterns，header_patterns はフォームデータ，クッキー，ヘッダ情報をそれぞれ指定するデータ名と文字列パターンの対を要素とする配列である．

(3) ドメイン特化のアスペクトのインスタンス生成機構

最後に，Web アプリケーションにおける状態管理に適したアスペクトのインスタンス生成方法を，表 3 のように定義した．per-application は，負荷分散等の Web サーバ全体としての状態管理を行う為のアスペクトインスタンスであり，Web アプリケーションの起動時にインスタンス化され，そのインスタンスが全てのプログラムで共用される．per-session

表 3 リクエストジョインポイントに基づいたアスペクトのインスタンス生成機構
Table 3 Aspect instantiation mechanism based on request join points

アスペクトインスタンスの種類	説明
per-application	Webアプリケーション全体で1つのインスタンスを生成
per-session	各ユーザごとにインスタンスを生成
per-request	1つのリクエスト処理につき1つのインスタンスを生成

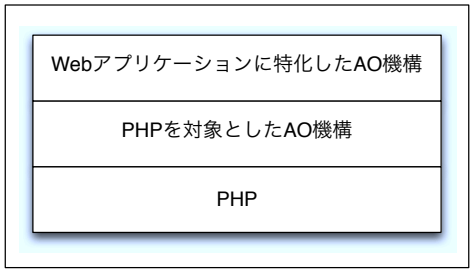

表 4 言語拡張のレイヤー
Table 4 Layer of language extension

は，各ユーザごとの一連のページアクセスにまたがる状態管理を行う為のアスペクトインスタンスであり，2 節で述べた各ユーザの滞在時間やユーザ認証結果の保持等をアスペクトのインスタンスの中で取り扱う事ができる．per-request は，ページリクエスト処理ごとにインスタンス化されるアスペクトである．

3.3 汎用的な AO 機構の拡張によるドメイン特化 AO 機構の実装

設計したドメイン特化の AO 機構は，図 4 に示すようなレイヤ構造の言語拡張に基づいて実現しており，PHP を対象とした汎用的な AO 機構の織り込み機構を拡張する事で実装した．AOWP は，織り込み機構自体も PHP を用いたオブジェクト指向設計に基づくプログラムとして実装されており，ポイントカットの評価，アドバイスの適用，アスペクトのインスタンス生成に関わる織り込み機構内のクラスを修正する事で，上で設計したドメイン特化の AO 機構を汎用的な AO 機構に組み込んだ．以降，順に，この織り込み機構の拡張について説明する．

(1) リクエスト処理を取り扱う為のポイントカット&アドバイス機構の実装

リクエスト処理を取り扱う為のポイントカット&アドバイス機構は，(a) request のポイントカット記述子を表すクラスの定義，(b) アドバイスの適用方法の変更，(c) proceed メソッドを実現する為の変更，の大きく分けて 3 つの拡張を AOWP の汎用的な織り込み機構に対して行った．

(a) の request ポイントカットは，AOWP_RequestPointcut クラスとして定義した．AOWP でのポイントカット記述子の定義は，AOWP_Pointcut クラスを継承したクラスを作成し，主に，ポイントカット記述子の引き数を受け取るコンストラクタの記述と，_isJoinPointShadow と _runtimeMatch の 2 つのメソッドのオーバーライドによって行う．

_isJoinPointShadow は，静的なコード解析から得られる情報に基づいて，織り込みの際にジョインポイントと対応するプログラムに対してコード変換を行うかを判断するメソッドである．ここでは，表 2 の情報に基づいて，以下のように _isJoinPointShadow を記述した．

```
1 protected function _isJoinPointShadow(AOWP_JoinPoint $jp) {
2   if ($jp instanceof AOWP_ScriptExecutionJoinPoint)
3     return $this->requestURLMatch($jp->getFileName());
4   else
5     return $this->_advice instanceof AOWP_AfterAdvice &&
6         $jp instanceof AOWP_ExitJoinPoint;
7 }
```

_isJoinPointShadow メソッドは，織り込み機構が対象システムのコード解析を行いプログラム中のジョインポイントを生成した後，生成された全てのジョインポイントを引き数として順に呼び出される．上の記述では，表 2 のジョインポイント写像の定義に基づいて，ポイントカット記述子に指定された URL パターンと適合するファイル名のスクリプトファイルの実行を表すジョインポイント (1，2 行) と，適用されるアドバイスが after アドバイスの時の exit もしくは die 関数の呼び出しを表すジョインポイント (4-6 行) を選択している．

また，_runtimeMatch メソッドは，以下のように記述した．

```
1 protected function _runtimeMatch(AOWP_JoinPoint $jp) {
2   if ($jp instanceof AOWP_ScriptExecutionJoinPoint)
3     return AOWP_RequestPointcut::_checkTopLevelScriptExecution() &&
4         $this->_formDataMatch($jp) && $this->_cookieMatch($jp) && $this->_headerMatch($jp)
5   else
6     return $jp instanceof AOWP_ExitJoinPoint && $this->_runtimeURLMatch() &&
7         $this->_formDataMatch($jp) && $this->_cookieMatch($jp) && $this->_headerMatch($jp)
8 }
```

_runtimeMatch メソッドは，実行時の情報に基づくポイントカットの評価を行うメソッドである．上の記述では，4，7 行目で，フォームデータ，クッキー，リクエストヘッダ情報の評価を行っている．また，スクリプトファイルの実行を表すジョインポイントの場合は，そ

れがリクエスト処理に直接対応する実行であるかを評価している (3 行)．さらに，exit もしくは die 関数の呼び出しを表すジョインポイントの場合は，リクエスト URL がポイントカットに指定されたパターンと適合するかを評価している (6 行)．

(b) のアドバイスの適用方法の変更については，織り込みの際のコード変換を制御する AOWP_WeaveCommand クラスの変更を行った．AOWP_WeaveCommand クラスでは，上で述べた _isJoinPointShadow メソッドの返り値が正の時のジョインポイントについて，ジョインポイントの種類とアドバイスのタイプ (before，after，もしくは around) に基づいてコード変換を行う．ここでは，AOWP_WeabeCommand クラスの after アドバイスを対象としたコード変換について，以下のように変更を行った．

```
1 // 省略
2 else if ($advice instanceof AOWP_AfterAdvice) {
3   if ($advice->isHavePointcut(new AOWP_RequestPointcut(null)) &&
4       $joinPoint instanceof AOWP_ExitJoinPoint)
5     $weaveCommand->weaveBeforeAdvice($advice, $joinPoint);
6   else
7     $weaveCommand->weaveAfterAdvice($advice, $joinPoint);
8 }
9 // 省略
```

上の記述では，表 2 に基づいて，request ポイントカットで選択した exit もしくは die 関数の呼び出しに対する after アドバイスについて，ジョインポイントの前にコード変換を行うようにしている (2-5 行目)．

(c) の変更は，リクエスト処理を表すジョインポイントに対して around アドバイスを適用した時の，proceed メソッドを実現する為のものである．around アドバイスは，ポイントカットで選択したジョインポイントが表す処理の代わりに実行されるアドバイスである．また，proceed メソッドは，around アドバイスの中で元々のジョインポイントの処理を実行する為のメソッドである．表 2 のリクエストジョインポイントの終了点を表す exit もしくは die 関数の呼び出し (M2 の out) は，必ず対応するスクリプトファイル実行の制御フロー (M1 の処理区間) に含まれる．そこで，リクエストジョインポイントに around アドバイスを適用する為の織り込み処理では，対応するスクリプトファイルの実行全体を proceed メソッドで呼び出せる形で変換し，それらの処理の代わりにアドバイスが実行されるようにコード変換を行っている．そして，proceed メソッドの実行中に exit もしくは die 関数が呼び出された時に，proceed メソッドの呼び出し元であるアドバイス処理に実行を戻す為に，図 3.3 の定義済みのアスペクトを対象システムに織り込むように織り込み機構を変更した．また，リクエストジョインポイントを対象とした around アドバイス記述に含まれる

```
1  class AOWP_RequestAroundManageAspect extends AOWP_PerJoinPointAspect {
2    private static $_IN_PROCEED = false;
3
4    public function __construct() {
5      $throwRequestAroundEndExeptionAdvice = new AOWP_BeforeAdvice();
6      $exitOrDieFunctionCallPC = new AOWP_FunctionCallPointcut('^exit$|^die$');
7      $throwRequestAroundEndExeptionAdvice->setPointcut($exitOrDieFunctionCallPC);
8      $throwRequestAroundEndExeptionAdvice->setAdviceBody('_throwRequestAroundEndException');
9      $this->addAdvice($throwRequestAroundEndExeptionAdvice);
10   }
11
12   protected function _throwRequestAroundEndException() {
13     if (AOWP_RequestAroundManageAspect::$_IN_PROCEED) {
14       throw new AOWP_RequestAroundEndException();
15     }
16   }
17   public static function inRequestProceed() {
18     AOWP_RequestAroundManageAspect::$_IN_PROCEED = true;
19   }
20   public static function outRequestProceed() {
21     AOWP_RequestAroundManageAspect::$_IN_PROCEED = false;
22   }
23 }
```

図 2 リクエストジョインポイントを対象とした proceed を実現する為の定義済みアスペクト
Fig. 2 An pre-defined aspect for proceed method for request join points

proceed メソッドの呼び出しに対して，以下のようなコード変換を行うように織り込み機構を変更した [*1]．

```
1 AOWP_RequestAroundManageAspect::inRequestProceed();
2 try {
3   $joinPoint->proceed();
4 } catch (AOWP_RequestAroundEndException $exception) {
5 }
6 AOWP_RequestAroundManageAspect::outRequestProceed();
```

このように PHP の例外モデルを利用して，リクエストジョインポイントを対象とした proceed メソッドを実現している．

(2) Web アプリケーションに特化したアスペクトのインスタンス生成機構の実装

表 3 に示すアスペクトのインスタンス生成機構は，実行時の全てのアスペクトのインスタンス生成を管理する AOWP_AspectInstanceManager クラスを変更して実現した．

*1 説明の為に，実際の変換されたコードを簡略化して記述している．

AOWP_AspectInstanceManager の getInstance メソッドは，アスペクトを表すクラスの名前を引き数に取り，そのクラスの基底クラスに基づいてアスペクトのインスタンスを生成する．そこで，表 3 の per-application，per-session，per-request に対応するアスペクト定義の為の基底クラスとして，AOWP_PerApplicationAspect，AOWP_PerSessionAspect，AOWP_PerRequestAspect を定義した．また，AOWP_AspectInstanceManager の中では，以下のようにアスペクトの基底クラスに基づいて，そのインスタンスを管理している．

- *AOWP_PerApplicationAspect*： 1 つのアスペクトインスタンスをシリアライズしてファイル上で管理．
- *AOWP_PerSessionAspect*：シリアライズしたアスペクトインスタンスをクッキーに保存するユーザの識別子と関連付けて管理．
- *AOWP_PerRequestAspect*：AOWP_AspectInstanceManager の静的変数の中で管理．

なお，AOWP_PerApplicationAspect のインスタンスは，PHP のファイルロックの API を利用して，複数のユーザからの同時アクセスに対して排他制御を行っている．

4. 考　　察

本節では，最初に，前節で定義したドメイン特化の AO 機構の有効性ついて考察する．そして，拡張可能な AO 機構[2)–5)] を用いたドメイン特化の AO 機構の開発について，前節で述べたケーススタディに基づいて考察する．

4.1 ドメイン特化のアスペクト記述の有効性

2 節で述べたアンケート機能について，前節で開発したドメイン特化の AO 機構を用いて記述したアスペクトを図 4.1 に示す．図 1 の汎用的な AO 機構のみを用いて記述したアスペクトと比較すると，記述量が少なくなり，可読性が高くなっていると言える．ポイントカットは，RequestPointcut を用いてアスペクトが作用するページリクエストを適切に表現できており，アドバイスの定義についても，リクエスト処理に作用する 1 つの proceed メソッドとして簡潔に記述できている．また，アスペクトを各ユーザに関連付けてインスタンス化する事で，ユーザの滞在時間，アンケート表示の有無をアスペクトのインスタンス変数として管理でき，ユーザごとの状態管理を容易に記述できた．

4.2 拡張可能な AO 機構

本稿では，設計したドメイン特化の AO 機構を，汎用的な AO 機構を直接拡張する事で実装した．AOWP では，織り込み機構の自体も PHP を用いて実装している為，PHP の言語仕様に基づいて比較的容易にドメイン特化の AO 機構を実装できた．しかしながら，一般的に AO 機構を直接拡張する事は容易ではない．そこで，拡張可能な AO 機構[2)–5)] を用いる方法が考えられる．

```
1  class QuestionnaireAspect extends AOWP_PerSessionAspect {
2    public $fstAccTime;
3    public $asked = false;
4
5    public function __construct() {
6      $this->_fstAccTime = time();
7      $allReqPC = new AOWP_RequestPointcut('.*');
8
9      $quesAdv = new AOWP_AroundAdvice();
10     $quesAdv->setPointcut($allReqPC);
11     $quesAdv->setAdviceBody('ques');
12     $this->addAdvice($quesAdv);
13   }
14
15   public function ques(AOWP_JoinPoint $jp) {
16     if (!$this->asked && time() - $this->fstAccTime > 60) {
17       $this->asked = true;
18       ob_start();
19       $jp->proceed();
20       /* 出力にアンケート表示のJavaScriptを追加 (省略) */
21 } } }
```

図 3 ドメイン特化の AO 機構を用いて記述したアンケートアスペクト
Fig. 3 Questionnaire aspect described in the domain-specific AO mechanism

Josh[4)] や SCoPE[2)] を用いる事で，複雑なポイントカット記述を簡潔にするドメイン特化のポイントカット記述子を容易に定義できる．しかしながら，3 節で述べたような，ドメイン分析に基づくアドバイスの適用方法やアスペクトのインスタンス生成機構の拡張についてはサポートしていない．

abc[3)] は，拡張可能な AspectJ コンパイラであり，その柔軟性の高さから多くの AO 言語の拡張の実装に利用されている．しかしながら，abc の利用にはコード解析やコード変換等について理解する必要があり，その利用は，一般のプログラマにとって必ずしも容易なものではない．

プログラマにとって比較的容易に拡張可能な AO 機構として，POPART[5)] がある．POPART は，AO 機構を拡張する為のクラス群がメタ・アスペクト・プロトコル (meta aspect protocol) として定義されており，これらのクラス群を継承して拡張する事で，一般的なプログラミングと同様の手法で AO 機構を拡張できる．例えば，JoinPoint や Pointcut クラスをカスタマイズしてドメイン特化のジョインポイントやポイントカットを定義した

り，MetaAspect クラスを拡張する事でアドバイスの適用方法等をカスタマイズできる．また，MetaAspectManager を拡張する事で，アスペクトインスタンスの振る舞いについても柔軟にカスタマイズでき，ドメイン特化の AO 機構の実装基盤として有効だと考えられる．

5. ま と め

本稿では，Web アプリケーションに特化した AO 機構 AOWP の開発をケーススタディとして，ドメイン特化の AO 機構の設計から実装までの一連の開発手順を示した．本稿で示したドメイン特化の AO 機構の開発手順では，ドメイン特化の AO 機構のジョインポイントを設計し，そのジョインポイントに基づいてポイントカット&アドバイス機構及びアスペクトのインスタンス生成機構を設計した．また，設計したドメイン特化の AO 機構を，汎用的な AO 機構の拡張として実現する方法を示した．

参 考 文 献

1) Ahroum, R., Hokamura, K., Balouek, D., Nakajima, S. and Ubayashi, N.: Aspectual encapsulation of Web application features, Vol.109, No.231, 電子情報通信学会, pp.13–18 (2009).
2) Aotani, T. and Masuhara, H.: SCoPE: an AspectJ compiler for supporting user-defined analysis-based pointcuts, *AOSD '07: Proceedings of the 6th international conference on Aspect-oriented software development*, New York, NY, USA, ACM Press, pp.161–172 (2007).
3) Avgustinov, P., Christensen, A.S., Hendren, L., Kuzins, S., Lhoták, J., Lhoták, O., de Moor, O., Sereni, D., Sittampalam, G. and Tibble, J.: abc: an Extensible AspectJ Compiler, *Proceedings of the 4th International Conference on Aspect-oriented Software Development (AOSD '05)*, New York, NY, USA, ACM, pp.87–98 (2005).
4) Chiba, S. and Nakagawa, K.: Josh: an Open AspectJ-like Language, *Proceedings of the 3rd International Conference on Aspect-oriented Software Development (AOSD '04)*, New York, NY, USA, ACM Press, pp.102–111 (2004).
5) Dinkelaker, T., Mezini, M. and Bockisch, C.: The art of the meta-aspect protocol, *AOSD '09: Proceedings of the 8th ACM international conference on Aspect-oriented software development*, New York, NY, USA, ACM, pp.51–62 (2009).
6) Hoffman, K. and Eugster, P.: Towards reusable components with aspects: an empirical study on modularity and obliviousness, *ICSE '08: Proceedings of the 30th international conference on Software engineering*, New York, NY, USA, ACM, pp. 91–100 (2008).
7) Hokamura, K., Naruse, R., Shiozuka, M., Ubayashi, N., Nakajima, S. and Iwai, A.: AOWP: Web-specific AOP framework for PHP, *ASE '09: Proceedings of the twenty-fourth IEEE/ACM international conference on Automated software engineering* (2009).
8) Jazayeri, M.: Some Trends in Web Application Development, *FOSE '07: 2007 Future of Software Engineering*, Washington, DC, USA, IEEE Computer Society, pp. 199–213 (2007).
9) Kellens, A., Mens, K., Brichau, J. and Gybels, K.: Managing the Evolution of Aspect-Oriented Software with Model-based Pointcuts, *In Proceedings of the European Conference on Object-Oriented Programming (ECOOP*, Spring-Verlag, pp. 501–525 (2006).
10) Kiczales, G., Lamping, J., Menhdhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M. and Irwin, J.: Aspect-Oriented Programming, *Proceeding of the 11th European Conference on Object-Oriented Programming (ECOOP '97)*, pp.220–242 (1997).
11) Kojarski, S. and Lorenz, D.H.: Domain driven web development with WebJinn, *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications (OOPSLA '03)*, New York, NY, USA, ACM Press, pp.53–65 (2003).
12) Lemos, O. A.L., Junqueira, D.C., Silva, M. A.G., de Mattos Fortes, R.P. and Stamey, J.: Using Aspect-oriented PHP to Implement Crosscutting Concerns in a Collaborative Web System, *Proceedings of the 24th Annual Conference on Design of Communication (SIGDOC '06)*, New York, NY, USA, pp.134–141 (2006).
13) Masuhara, H. and Kawauchi, K.: Dataflow Pointcut in Aspect-Oriented Programming, *Proceedings of the First Asian Symposium on Programming Languages and Systems (APLAS '03)*, pp.105–121 (2003).
14) Niu, N., Yu, Y., Gonzlez-Baixauli, B., Ernst, N.A., do Prado Leite, J. C.S. and Mylopoulos, J.: Aspects across Software Life Cycle: A Goal-Driven Approach., *T. Aspect-Oriented Software Development VI*, Vol.6, pp.83–110 (2009).
15) Sullivan, K., Griswold, W.G., Song, Y., Cai, Y., Shonle, M., Tewari, N. and Rajan, H.: Information hiding interfaces for aspect-oriented design, *ESEC/FSE-13: Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering*, New York, NY, USA, ACM, pp.166–175 (2005).
16) VanLanduyt, D., Opde beeck, S., Truyen, E. and Joosen, W.: Domain-driven discovery of stable abstractions for pointcut interfaces, *AOSD '09: Proceedings of the 8th ACM international conference on Aspect-oriented software development*, New York, NY, USA, ACM, pp.75–86 (2009).
17) Ward, M.: Language Oriented Programming, `http://www.cse.dmu.ac.uk/~mward/martin/papers/middle-out-t.pdf` (2003).