社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－ＳＥ－156 (4)
2007－EMB－ 5 (4)
2007／5／28

「ウィンターワークショップ 2007・イン・那覇」開催報告

松塚 貴英(富士通研)
沢田 篤史(南山大)
青木 利晃(JAIST)
福安 直樹(和歌山大)
妻木 俊彦(日本ユニシス)
中村 友昭(新日鉄ソリューションズ)
浦本 直彦(日本 IBM)
羽生田 栄一(豆蔵)
鷲崎 弘宜(国立情報学研)

ソフトウェア工学研究会では，本年もテーマを絞った集中的な議論の場として，ワークショップを開催したので報告する．

A report of "Winter Workshop 2007 in Naha"

Takahide Matsutsuka (Fujitsu Labs.)
Atsushi Sawada (Nanzan Univ.)
Toshikatsu Aoki (JAIST)
Naoki Fukuyasu (Wakayama Univ.)
Toshihiko Tsumaki (Nihon Unisys)
Tomoaki Nakamura (Shin-Nittetsu Solutions)
Naohiko Uramoto (IBM Japan)
Eiichi Hanyuda (Mamezo)
Hironori Washizaki (NII)

This paper reports "Winter Workshop 2007 in Naha", held at Jan. 25-26 in Naha city, Okinawa pref.

1. はじめに

情報処理学会ソフトウェア工学研究会では，1997 年より，テーマを絞った集中的な議論の場として，ワークショップを毎年開催している．本年度も，下記5つのテーマを取り上げ，各分野に造詣の深い討論リーダーのもと，それぞれの分野で，いま何をやるべきか，そのために解決すべき研究・技術課題は何か等を議論した．

- 組込みソフトウェア
  セッションリーダ: 青木 利晃(北陸先端大)，沢田 篤史(京大)
- プログラム解析
  セッションリーダ:福安 直樹(和歌山大学)
- 要求工学
  セッションリーダ:妻木 俊彦(日本ユニシス株式会社)，中村 友昭(新日鉄ソリューションズ株式会社)
- サービス指向
  セッションリーダ:浦本 直彦(日本 IBM)
- アーキテクチャとパターン
  セッションリーダ:羽生田 栄一(豆蔵)，鷲崎 弘宜(国立情報学研究所)

今回の会場は，那覇の国際通りに面したホテルとカンファレンスホールで，当初予定を上まわる 54 名の参加者があり，盛況であった．5 つの討論グループに分かれ，各分野に造詣の深い討論リーダの元で，大学および企業の参加者それぞれの立場から，研究活動や技術課題について議論を行った．討論グループによっては前もってメールでの議論を行い，当日持ち寄ることにより，内容の濃い議論を行うことができた．また，今回は全体セッションの時間を少なくして討論グループごとの議論時間を多めに確保し，それぞれのグループでの議論の内容を密度の高いものにした．全体セッションでは各討論グループで議論され，整理された内容や今後の課題について報告が行われた．

次節より，各討論グループで行われた議論の内容をそれぞれ述べ，その後運営上の課題について議論し，まとめを行う．

2. 組み込みソフトウェア

本ウィンターワークショップの組込みソフトウェアセッションは，今回で7回目の

開催となり，「組込みシステムのためのモデリング」というタイトルで参加者

の募集を行った．その結果，以下の参加者がセッションに参加し，討論を行った．

青木 利晃(北陸先端科学技術大学院大学)
沢田 篤史(南山大学)
新屋敷 泰史(松下電工/九州工業大学)
中島 震(国立情報学研究所)
中村 宏明(日本 IBM)
平山 雅之(情報処理推進機構)

本セッションでは，まず，参加者全員がそれぞれのポジションを発表し，議論を行った．参加者のポジションは，要求モデリングの際の非正常系の取り扱い，設計モデリングにおける性能や実時間性の取り扱い，モデリングを支援するプロセスとツール，複数の設計パラダイムの衝突，および，現状分析に関するものであった．その後，「組込みシステムのためのモデリング」というキーワードで，その特徴や課題について討論を行った．

最初に「モデリング」について確認を行った．UML などの記法で文書化を行うだけではモデリングではない．モデリングは，性能や実時間性の解析といった特定の目的を伴うものであり，作成されるモデルは対象システムを抽象化したものである．

次に，ポジション発表に関連する設計モデリングと要求モデリングについて討論を行った．設計モデリングでは，開発対象システムの特徴をすべて反映した設計モデルを作成することは不可能である．対象システムの特性をすべてとらえることができたモデルを，仮に，「神様(oracle)モデル」と呼ぶことにする．組込みシステムの場合，自然界や機械を対象とするため，この神様モデルは，(神様以外の) 誰にもわからないものである．そこで，この神様モデルを様々な視点から，それぞれの特性を推測するモデルを作成することになる．この際，1つの設計モデルですべての特性を取り扱うのは困難である．よって，それぞれの特性毎に，それを反映させるのに適した記法や視点で，対象となる部分を記述する．MDA では，プログラムを自動生成できるモデルを作成するが，これはプログラムを自動生成するためのモデルである．性能や実時間性を解析するためには，やはり，それぞれの視点から，関係する部分を抽出する必要がある．このようにして作成されたモデルは，神様モデルの，それぞれの視点や特性における近似であり，それを繰り返す，すなわち，多視点からモデリングすることにより，より神様モデルに近づけ，妥当なシステムを設計するのである．以上のことは，一般のシステム開発でも同様であるが，組込みシステムでは，センサやアクチュエータといったハードウェア，さらには，それらが接する自然界を考慮しなければならないため，特に複雑である．

組込みシステムを対象とした場合は，要求モデリングに関しても自然界が大きく関係する．非正常系に関するモデリングでは正常ではない出来事を把握しなければならないが，自然界や物理的に発生するそれらを抽出するのは困難である．また，非正常な事態を抽出するだけでなく，そのような事態が発生した時の対処法を決めるのも重要である．組込みシステムは，社会システムや人間生活と深く関係しているため，法律や基準に基づいた対処が必要な場合が多い．よって，自然界だけでなく，法律や基準といった社会的な決めごとを考慮したモデリングも必要となる．

さらに，組込みシステムのモデリングでは，様々な異なる計算モデルが混在している．例えば，車載制御システムを対象とすると，フィードバック制御などを行うための連続モデル，車内ネットワークなどを用いた接続状況を捉える分散モデル，イベントや状態に基づいてソフトウェアの状態変化を記述する離散モデルといったものを作成することになるであろう．これらのモデルは，相互に関係があるが，直接的には相互に対応づけることが困難であり，全体として考察や解析を行うことは難しい．それらを統一的に扱えたり，橋渡しができるアプローチが必要である．

以上のように，組込みシステムは，制御工学やソフトウェア工学など複数の分野が関係している．現状では，システム制御の分野に携わっていた開発者がソフトウェアの部分も開発したり，一般のソフトウェア開発に携わっていた開発者が制御の部分も開発したりしている．前者の開発では，制御中心の思考になりがちであり，その結果，ソフトウェアの部分について十分に検討が行われない場合がある．後者の開発は，その逆である．組込みシステムでは，どちらも重要であり，バランスのとれたモデリングをする必要がある．

このように，本セッションでは，組込みシステムのモデリングに関して議論を行い，問題点の分析と課題の抽出を行った．モデリングはソフトウェア工学の代表的な手法であり，爆発的な規模の増大と多用性の増大により混沌としている組込みシステムを，整然と開発するには必須の手法である．モデリングは，対象システムの仕様や設計を単に文書化するだけではなく，特定の目的を持って試行錯誤しながら対象システムの抽象イメージを定めるものであり，そのためには，対象システムの本質的な難しさを認識する必要がある．今回は，このようなモデリングの議論を組込みシステムを対象として行ったことにより，組込みシステムの本質的な難しさが少し明らかになったような気がした．モデリングという活動の見直しと，本質的な問題点と課題の議論ができ，非常に有益であったと思う．

3. プログラム解析

3.1. 背景と目的

ソフトウェア開発を支援するツールの中には，その基盤技術としてコールグラフの生成やフロー解析といったプログラム解析技術を利用しているものが多く存在する．最近ではその解析対象となるシステムが，複数の実装言語を用いて構築されるようになり，開発支援ツールも複数の言語を同時に対象とする必要がでてきた．また，解析対

象が単一の言語で実装されている場合でも，ツールが要求する解析が高度になり，今後は複数の解析技術を組み合わせて利用することがより重要になると予想される．一方，Eclipse などの統合開発環境が利用されるようになり，一つのツールの中で，複数の言語が同時に扱える環境が整いつつある．それらのツールの多くは，各々の言語を独自にサポートするにとどまり，言語と言語の間の関係を積極的にサポートするものは少ない．例えば，HTML の文法と JavaScript の文法をそれぞれ理解し，予約語を強調表示したり，それぞれの言語においてクロスリファレンスを提供するような機能は実装されているが，HTML と JavaScript を双方向にたどれるような機能はほとんど実装されていない．そこで本セッションでは，複数のプログラム解析技術を組み合わせる上での現状の問題点について整理するとともに，今後の方向性について議論を行なった．

本討論グループには，14 件の論文が提出され，参加者は 16 名であった．論文は，技術や解析対象の組み合わせ方によって以下の通り 5 種類に分類でき，プログラム解析に関する様々な技術や問題点について広範に議論することができた．

- ヘテロジニアスなシステムの解析(5 件)
- プログラム解析の利用例(4 件)
- ソースプログラムの変更履歴(3 件)
- 多言語に対応した解析ツール(1 件)
- アノテーションの利用(1 件)

3.2. 討論の結果

(1) 解析対象

ヘテロジニアスな解析対象として，Web アプリケーションを扱うものが 4 件と最も多かった．Web アプリケーションは Java などのサーバサイドの技術，JavaScript などのクライアントサイドスクリプト，XML などのデータ表現形式，HTML などのページ記述が互いに影響し合う．ヘテロジニアスな対象を解析するために，「言語非依存のソースコードモデルが必要である」という意見が出された．これにより異なる言語で記述されたコンポーネント間でのデータフローを捉えることが容易になる．

また，アプリケーション開発にはさまざまなフレームワークが利用されるが，フレームワークが用意しているライブラリの中身を解析するのは複雑なので，「フレームワークに応じて，それぞれのライブラリが持つ意味を考慮した解析が必要である」という意見が出された．ライブラリの意味は，なんらかの形式で外から与える必要があるが，ライブラリの設計に関する情報を利用することにより，ライブラリの中身をプログラム解析して得られる結果より有益な解析結果を得られる場合が多い．

(2) 解析の粒度

プログラム解析においては，解析の粒度をどの程度に設定するかが重要であるとの意見が複数の論文において議論された．リファクタリングの対象候補の検出などのためにソースプログラムの変更履歴の解析が行われるが，その際には「変更を取得するタイミングが重要」であり最も細粒度には，「すべての編集操作を取得すべきである」との意見が出された．一方で，すべての編集操作を扱うには計算資源に限界があり，「細粒度のデータの中からリファクタリングの対象候補を見つけるための探索空間を絞る工夫が必要である」との意見が出された．具体的には，特徴的な変更のパターンを編集操作の履歴から見つけるなどの方法が提案された．

(3) プログラム解析以外の情報の利用

議論を進める中で，プログラムの解析には，「プログラム解析以外の情報を活用することが重要である」との結論が得られた．例えばフレームワークなどで用意されるライブラリは，ソースコードが手に入る環境にあれば，それらを解析することによってある程度の情報は得られる．しかしながら，ライブラリを開発した際の意図を積極的に活用することによって，そのライブラリを利用したプログラムの理解のためにより有益な情報が提供できる．また，プログラム開発においては，開発者にしか張ることのできない追跡性も存在する．例えば，実装時における開発者の意思決定や選択は，解析だけでは正確に取得できない場合が多い．プログラム開発者によるアノテーションを利用することによって，プログラム解析だけでは精度の出ない問題を補完することができる．

プログラムの記述の容易さと解析の容易さとは，一般にトレードオフの関係にあるが，今後は双方においてよりメリットの得られる「プログラムの記述方法の提言が重要である」との意見が出た．

4. 要求工学セッション

要求工学は扱う問題も工程も広い範囲を対象にしている．そこで本セッションの目的では，事例・最新動向の把握，および，研究者・実務者間の相互交流とコミュニティの形成を重視することとした．これは今後の要求工学の発展のために，実プロジェクトにおいて要求工学をより役立つものにする事を意図したものである．

このため参加者一人一人による個別発表を中心にし，批評者を指定しての批評を組み合わせて実施した．これは REFSQ[3]のスタイルを真似たものである．なお具体的な進め方は，本セッションの Web ページ[4]で紹介している．

4.1. 参加者

11 機関から 14 名が参加した．14 名の内，企業からは六名，大学からは八名(院生二名)である．前回(2005 年)の参加者からは全体で二名減ったが，企業からの参加者は倍増しており，実プロジェクトでの要求工学への関心の高まりがうかがわれる．

4.2. 個別発表の内容

13 件の発表がおこなわれた．前回と比べ手法や実証的な研究の発表が少なくなり，ビジネスや環境の理解に

関する発表が多くなった．実際，前回ビジネスや環境の理解に関する発表は一件であったが，今回は五件に増加している．また要求工学に対する問題提起的な発表もみられた．

4.2.1. ビジネスや環境の理解

言語行為展望論にもとづいた会話構造のモデル化，ソフトウェアにユーザーの意図を理解させるための手法，顧客分析の重要性を主張する提案がなされた．また，ビジネスの分析に要求工学的な手法を適用するという提案や，ビジネスプロセスモデルを作成した事例も紹介された．

4.2.2. 要求の獲得・検証・活用

ログや自然言語の分析による要求抽出・検証，要求や仕様変更の分析，制限言語を用いた要求仕様の活用支援についての手法が発表された．また，最近注目されているセキュリティ要求に関する手法の提案もおこなわれた．

4.2.3. 問題提起・教育

要求工学の研究成果がなかなか実プロジェクトに浸透しないことについて，原因の分析と問題提起がおこなわれた．また，要求工学の教育に関する取り組みが紹介された．

4.2.4. 個別発表の評価

ビジネスや環境に関する発表では五件中四件が手法の提案であり，今後の進展が期待できる．また事例はi*[5]による分析を扱っており，ゴール分析の実プロジェクトへの浸透が進みつつある現状がうかがわれる．

要求の活用に関する発表では，自然言語やログの分析・利用が紹介された．これも今後の進展が期待される分野である．なお要求工学に対する問題提起については，残念ながら時間がなく議論が深まらなかった．

4.3. セッションの運営

様々な立場の人が議論に参加することによる議論の多様性と深まりの確保，および，研究者・実務者の枠をこえたコミュニティの形成を目的として，運営上の工夫をいくつかおこなった．

4.3.1. 批評

発表ごとに批評者を一人決め，発表内容に対する批評を依頼した．また批評者と発表者の間で事前に打ち合わせすることを推奨した．批評者は経験や立場(企業か大学か)の点で，なるべく発表者と異なる人を選ぶことで，コミュニティの拡大，議論の多様性確保を意図した．

4.3.2. 事前活動

セッションの Web ページ[4]を随時更新して参加者への働きかけを継続するとともに，簡単なアクションや役割を依頼することで参加意識を高めることを狙った．

4.3.3. 懇親会

ワークショップ全体の懇親会の後にセッション独自の懇親会をおこない，コミュニティの形成をはかった．

4.3.4. 運営の評価

批評者になることによって参加意識が高まり，様々な立場の人が活発に議論に参加することができた．ただし，実務者と研究者の意見対立のような，意見の多様性はあまり見られなかった．

4.4. セッションの評価

要求工学の動向の把握，参加者の交流という目的は達成できたと考えている．また，コミュニティの形成については，長いスパンで評価するべきであろう．

なお，今回は時間に対して発表数が多く，残念ながら発表時間，質問時間とも10分という短い時間となった．議論の多様性に悪影響を与えたことは否定できないため，今後はより長い時間を議論にあてる工夫が必要と考えている．

5. サービス指向セッション

2007 年のサービス指向セッションでは，SOA やモデル駆動開発といった従来からのトピックに加え，Software as Services(SaaS)マッシュアップ，REST，Ajax といった Web 2.0 を支える技術，SOA と Web 2.0 の統合など に関するキーワードで論文を募集した．

ソフトウェア開発におけるサービス指向，モデル駆動の流れは，SOA およびその周辺技術の進化と共に普及しつつあり，企業においても，SOA に基づくソフトウェア＆サービス基盤構築を導入・検討する事例が増えてきている．これは，2006 年のワークショップでの合意であった，企業内のサービス連携が現実になってきたことの証左であろう．また，SOA の普及につれ，家電の連携など多様なプラットフォームへの適用も始まっている．これらの状況を踏まえ，現実的な企業システムやプラットフォームに SOA を導入していく上で見えてきた課題は何であろうか?

一方で，新しい Web 技術およびビジネスモデルの潮流として Web 2.0 が注目を浴びている．例えば，SOA の基盤技術である Web サービスは，その仕様の複雑さが阻害要因の一つとして指摘されており，シンプルなコミュニケーションプロトコルである REST は，Web サービスの対抗馬として語られることも多い．Web 2.0 と SOA は今後どのように統合，あるいは淘汰されていくのか?

このような問題意識のもと，このセッションに投稿された 6 件の発表を3つのトピックに分け，8 名の参加者による活発な議論が行われた．トピックは以下のとおりである．

[トピック 1] サービス指向を支える基盤技術
[トピック 2] サービス指向と Web 2.0
[トピック 3] サービス指向における新しい開発方法論

「サービス指向を支える基盤技術」トピックでは，以下の 3 件が発表された．

(1) 動的再構成の観点からのソフトシステムバスとエンタープライズサービスバスの比較
染谷雅美, Mohammad Reza Selim, 後藤祐一, 程京徳 (埼玉大学)

(2) SOA システムの動作検証用 UML シミュレータ

に関する研究
倉畑宏行 (大阪大学), 藤井拓 (オージス総研), 宮本俊幸 (大阪大学)

(3) オブジェクト指向モデルを利用した SOA のためのモデル指向開発手法
井垣 宏, 青山 幹雄 (南山大学)

染谷は, システム全体の自己監視, 自己計測, 自己制御をおこなうソフトウエアシステムバスと, SOA の基盤技術である Enterprise Service Bus (ESB)を動的再構成の観点からの比較を行った. 倉畑は, システムのモデルを視覚的にわかりやすく表現するための SOA システムの動作検証用 UML シミュレータの重要性について述べた. 井垣は, 家電機器を Web サービスを用いて連携するホームネットワークシステムの開発において, 検証プロセスを組み込んだモデル駆動型設計(MDD)手法を提案した. 全体討論では, モデル駆動における検証技術の重要性, コンポジットサービスの検証技術, ユーザビリティの問題(実際の現場では複雑なツールは使いこなせない)などの指摘があった.

「サービス指向と Web 2.0」トピックでは, 以下の 2 件の発表をもとに, 議論を行った,

(4) Web 2.0 時代のサービスアーキテクチャに関する一考察
松塚 貴英 (富士通研)

(5) Web 2.0 は SOA にとって有益か?
浦本直彦 (IBM)

松塚は, Web2.0 技術とサービスを概観し, 仕様記述とテスト, レガシーマイグレーション, セキュリティなどの課題を指摘した. 浦本は, Web 2.0 技術をエンタープライズで用いる際のセキュリティに関するプロジェクトを紹介した.

「サービス指向における新しい開発方法論」トピックでは, 南山大学の青山が様々なプラットフォームにおける開発から実装までの流れをサポートするユニバーサル SOA 構想について述べた.

(6) ユニバーサルサービス指向アーキテクチャ (uniSOA)試論
青山 幹雄 (南山大学)

現在の SOA を取り巻く環境の変化の中で, 原理であるアーキテクチャ設計に立ち返ることの重要性を指摘するとともに, 複雑化する標準仕様の問題, SOA と Web2.0 技術におけるサービスのコントロールポイントの問題などについて議論が行われた.

参加者が少なかった分, 一つ一つのテーマに対して, 中身の濃い議論ができたと思う. サービス指向のコンセプトは, 普及が進む SOA と, 簡単で使いやすい Web2.0 の 2 つの流れの中で, ますます重要になってくることが認識された. 次段階として, 企業を超えたサービス連携や, 様々な分野での応用が考えられるが, 整合性を持ち, 上流から下流, ユーザまでをカバーする技術や方法論が必要である. また, セキュリティやレガシーとの統合など, エンタープライズ SOA (+Web 2.0)の実現に必要な技術の発展が今後の鍵の一つであろう.

6. 討論報告:アーキテクチャとパターン

6.1. 目的と経緯

本セッションでは, ソフトウェア開発において密接に関係するアーキテクチャ技術とソフトウェアパターン技術それぞれの特性や課題, および, 両者間や周辺技術との関係について議論した. 参加者は, 羽生田栄一(豆蔵),鷲崎弘宜(国立情報学研究所)の両名を討論リーダとして, 下滝亜里(南山大), 鹿糠秀行(日立製作所), 久保淳人(早稲田大), 中山弘之(早稲田大), 深澤良彰(早稲田大), 田中裕一(東京工業大), 廣瀬康行(琉球大), 河野真治(琉球大)の全 10 名であった.

ソフトウェア工学研究会パターンワーキンググループは, これまでウィンターワークショップに毎年パターンセッションを設置し, 継続的にパターンおよびパターン技術の本質を明らかにしてきた. しかし, 周辺領域との関係について議論し尽くされていなかった. 特に, パターンがソフトウェアの構造上の決定指針を与えることを考えるとき, 機能/非機能要求と構造設計の関係を扱うアーキテクチャ領域との関係について, さらなる議論が必要である.

そこで本セッションでは, 参加者のポジションペーパ発表を起点として各技術に関連する経験や提案を概観し, 続いて, アーキテクチャとパターンそれぞれの課題や展望, 関係や周辺技術について幅広く議論した.

6.2. 議論成果

(1) 要素技術とトピックマップ

各参加者のポジションペーパ発表と議論によって両技術における種々の要素技術への理解を深め, その特性を明らかとした. 概要を以下に示す.

(1-1) 設計におけるパターン

- ソフトウェア進化の指針としてのデザイン進化オペレータ(下滝): 安全な進化の仕組みを詳細に明らかにするうえでデザインパターンやリファクタリングよりも細粒度な操作単位および周辺との関連モデルが必要なことを明らかにした.
- ソフトウェアパターン適用における逸脱パターンについて(鹿糠): パターンの背景には, パターンの典型的な誤適用と回避策としての逸脱パターンが潜み, その顕在化が学習と適切な適用を促すことを明らかとした.

(1-2) パターンの広がりと応用

- 観と場と(医療における類似ノウハウ)(廣瀬): 病院情報システムの基本的なアーキテクチャの特性, および, 医療におけるパターンに似た類似ノウハウの可能性を明らかとした.
- 文書類似度によるパターン間関連分析(久保): 文書間類似度の算出によりパターン間の連続適用といった関連を抽出できることを明らかとした.
- パターン間の関係による重要度(中山): パターン間の参照関係から各パターンの重要度を算出し

検索に役立てることが学習等に有効なことを明らかとした.

(1-3)　アーキテクチャ設計の具体的技術

- Continuous C によるリファクタリング(河野): 継続の仕組みを持つ C の下位言語プログラムが状態遷移設計に有効なこと, および, リファクタリングの可能性を明らかにした.
- 一級継続にもとづく動的更新可能な Web アプリケーションのアーキテクチャ(田中): 継続の仕組みにより状態遷移を保存することで停止させずに変更可能な Web アプリケーションアーキテクチャを提案し, その有効性を明らかにした.

(2)　アーキテクチャとパターンの融合

上述の各要素技術および周辺技術の関係を整理し, ソフトウェア開発におけるアーキテクチャとパターンの関係を詳細に整理し, 図 1 のようにまとめた.

特に, アーキテクチャ設計に注目した場合に, アーキテクチャとパターンの関係の 1 つの見方として, パターンがコンテキスト(機能要求および非機能要求を含むものと位置づけた)と実現手段を結びつけるものであり, かつ, プロセスであると位置づけ可能なことを整理した. さらに, 上述の要素技術を出発点として, 各部分の内部構造を検討し明らかとした.

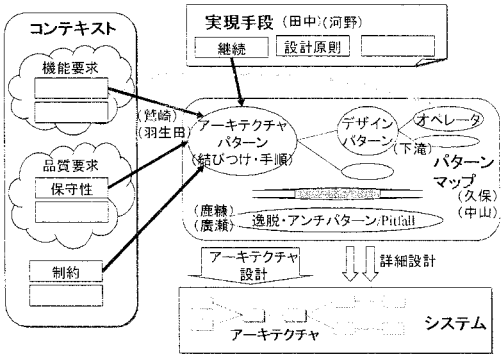

図 1: 開発におけるアーキテクチャ設計とパターン

6.3. 3.　課題と展望

以上の議論を通じて, 主にソフトウェアの設計におけるアーキテクチャとパターンの関係を捉えることに成功した. 一方で, 関係および個々の詳細をより明らかにするためには, 下記の課題への継続的取り組みが不可欠であるという気づきを得た.

- 「粒度」「階」を意識した関係整理: ソフトウェアの設計活動や利用知識・手段には階層・粒度があり, グラウンド(0 階)の把握が重要である. さらに, 階をまたぐ繋がりが確実にあり, その発見に向けた努力が必要である. パターンマップとしてのノウハウ間関連の可視化はその1つの手段として有効である. また, 粒度を定めることが, パターン/パターン間関係の発見と整理に繋がる.
- 対として存在する逸脱パターンの顕在化: 医療ドメインにおける同種概念の存在より, 逸脱パターンの必然性の裏づけを得た. 失敗の疑似体験をもたらす逸脱パターンの教育効果は大きく, 今後, 種々のパターンについて顕在化が求められる.
- アーキテクチャ・パターンにおける概念/用語の厳密化: これらの領域における概念や用語には曖昧なものが多く, 精密な議論と関係整理の妨げの一員となっている. 例えばフォース, 制約, コンテキスト等の概念は, その定義について曖昧な部分がある.

成果の全ては[9]において公開している. 今後は, これらの課題について, パターンWG内外における議論と実践を通じて, 開発・組織活動におけるアーキテクチャおよびパターンの活用と支援技術の発展を目指す.

7. 運営についての課題

今回, ワークショップの運営方法についていくつかの課題が出た. 以下は, 参加者やグループリーダーなどから出された要望や課題をまとめたものである.

- 宿泊費が高いとのクレームが出た
  これは, 宿泊代として計上した金額に懇親会での飲食代が含まれていたためである. 飲食代は会議費に含めるべきで, 宿泊費に含めるべきではない.
- 当日不泊者がいた
  主催側が当然宿泊を提供する前提で募集を行っていたが, 当日になってから別の宿泊施設に泊まった参加者がいた. 募集する際の Web ページなどに, 宿泊が提供されることが前提であることを明記すべきである.
- 登録間違いが多い
  学生なのに学会会員, 研究会会員なのに学会会員で登録するなど, 当日になって種別変更をした方が何名かいた. アナウンスの方法に改善の余地がある.
- 会議時間が短い
  会期が2日間とはいっても, 初日の午後から夕方まで, 2 日目は朝から昼までとなるため, 実際の分科会の時間は少なくなってしまう. うまく時間を配分して分科会時間をもっととる, または 3 日間に延長することなどを検討する余地がある.
- ポジションペーパの受付から更新締切までの期間が短い
  余裕を持ったスケジュールを組む必要がある. 今回, 準備の開始自体は早かったのだが, 会場の手配などが遅れたため, CFP を公開するのが 11 月末になってしまった. 遅くとも 11 月初頭には CFP を出す必要があると感じた.
- 予稿のページ数が少ない
  これについては, ワークショップなのである程度仕方ない. ただ, 予備議論や当日議論を長く取

る方法もあるため，セッションごとに工夫する余地があると感じた．

- 予稿集の掲載順序
  事務上の都合で予稿集へのポジションペーパの掲載は投稿順にしていた．分科会によっては指定された順に記載することも必要である．
- 受付で作業が混乱し，開始が40分遅れた
  領収書などあらかじめ準備していた書類がソートされておらず，受付スペースも狭かったため，受付処理に時間がかかってしまい，議論の時間を圧迫してしまった．休み時間に受付するなど，工夫する必要がある．
- ナイトセッション
  当日，ナイトセッションが可能であることがわかったのだが，希望していた分科会は既に他の場所を予約済みであり，結果的には行わなかった．可能であるならあらかじめアナウンスしておくべきと感じた．

これらの課題については，今年度以降の委員長に引継ぎ，今後のワークショップの運営に反映させたい．

8. おわりに

運営に関して課題も残ったが，全体としてはテーマの設定もよく，多数の参加者を得ることができた．また，それぞれの参加者が議論をすることにより，分科会ごと，または全体としての技術課題の共有と理解が深まったと考えている．

参加者間によるこれらの議論とその共有により，ソフトウェア工学の今後の研究・開発と実践の推進に寄与することを期待する．

参考文献

[1] Gomaa, H.:Designing Concurrent, Distributed, and Real-Time Applications With Uml, Addison-Wesley, 2000

[2] 白川 洋充, 竹垣 盛一, システム制御情報学会編: リアルタイムシステムとその応用, システム制御情報ライブラリー,朝倉書店, 2001

[3] International Workshop on Requirements Engineering: Foundation for Software Quality, http://www.refsq.org/

[4] http://kaiya.cs.shinshu-u.ac.jp/rewg/wws2007/

[5] i*: an agent-oriented modelling framework, http://www.cs.toronto.edu/km/istar/

[6] 松下誠 他: ウィンターワークショップ・イン・石垣島参加報告, 情処研報, 2004-SE-145, 2004.

[7] 柴合治 他: ウィンターワークショップ 2005・イン・伊豆参加報告, 情処研報, 2005-SE-148, 2005.

[8] 満田成紀 他: ウィンターワークショップ 2006・イン・鴨川参加報告, 情処研報, 2006-SE-152, 2006

[9] http://patterns-wg.fuka.info.waseda.ac.jp/