第2回アジア太平洋ソフトウエア工学国際会議 (APSEC'95) 報告

渡辺 晴美 † 三浦 信幸‡ 劉 少英 †† 山田 宏之‡‡

† 東京工業大学 情報理工学研究科 計算工学専攻
‡ 日本電信電話 (株) ソフトウェア研究所
†† 広島市立大学 情報科学部 情報数理学科
‡‡ 愛媛大学工学部情報工学科

1995 年 12 月 6 日 ~ 9 日に、オーストラリア、ブリスベーンで第 2 回アジア太平洋ソフトウエア国際会議 (APSEC'95) が開催された。本稿では、同会議の概要について報告する。

Report on the Second Asia-Pacific Software Engineering Conference(APSEC'95)

Harumi Watanabe † Nobuyuki Miura‡
Shaoying Liu†† Hiroyuki Yamada‡‡

† Department of Computer Science, Tokyo Institute of Technology
‡ NTT Software Laboratories
†† Department of Computer Science, Hiroshima City University
‡‡ Department of Computer Science, Ehime University

The Second Asia-Pacific Software Engineering Conference was held on December 6 − 9, 1995 at Brisbane, Australia. We briefly summarize the conference.

1 はじめに

第 2 回アジア・太平洋ソフトウエア国際会議 (Asia Pacific Software Engineering Conference, 略称 APSEC) は，1995 年 12 月 6 日から 9 日までオーストラリアのブリスベーンで開催された．主催は the Australian Computer Society(ACS) である．前回は，1994 年 12 月 7 日 ~ 9 日に東京の早稲田大学国際会議場で，情報処理学会ソフトウエア工学研究会主催で開催された．次回は，1996 年 12 月に韓国ソウルで韓国ソフトウエア科学会ソフトウエア工学研究会 (KISS/SIGSE) 主催で開催される予定である．

本会議はアジア・太平洋ソフトウエア地域を主対象とするソフトウエア工学に関する国際会議である．幹事国・地域は，日本，韓国，台湾，香港，シンガポール，オーストラリアの 6 つであるが，参加者をこれらの国・地域に限定しているわけではない．

参加者は，14 の国・地域 (日本，韓国，オーストラリア，シンガポール，香港，中国，台湾，ニュージーランド，イギリス，カナダ，マカオ，USA，フランス，ドイツ) から 137 名であった．

投稿論文は 160 件であり，上記参加国・地域に加え，インドとアルジェリアからの投稿があった．昨年の投稿が 92 件であったのに比べると非常に盛んになったことが伺える．採択論文は 53 件であり，日本の発表は 14 件あった．なお昨年の採択論文は 43 件であった．

本会議では，以上の採択論文の発表が 7 日 ~ 9 日に行なわれ，各々の午前のはじめに基調講演が，午後のはじめにパネル討論が行なわれた．そして，初日の 6 日はチュートリアルのみが開催された．以下，これらの各々について述べる．

2 基調講演

2.1 Prof. Dines Bjørner

本会議の初日には，国連大学 Bjørner 教授による，

> Software Systems Engineering – >From Domain Analysis via Requirements Capture to Software Architectures

が講演された．

Domain Analysis に始まり，Requiremets Capture，Software Architecture へととつながる，一連の流れについて講演が行われた．飛行機の航空管制という事例が用いられ，実際に分析・記述を行いながら，紹介された．特に，Domain Analysis では，形式的手法の一つ，RAISE を用いた分析について詳しく述べられ，domain analysis の形式的手法による取り扱いが聴衆に良く理解されたと思われる．

2.2 Prof. Motoshi Saeki

2 日目は，東工大 佐伯助教授による，

> Communication, Collaboration and Cooperation in Software Development – How Should We Support Group Work in Software Development ?

が講演された．

ソフトウェア開発は，様々な人達による，種々のツールを用いた協調作業であり，開発環境をより効率良く，より快適にするためには，開発プロセス中の協調作業を明らかにして，どのように支援できるかを考えていく必要があると説いた．これに基づき，ソフトウェア工学の分野で行われたいくつかの Case Study や CSCW(Computer Supported Cooperative Work) の分野で研究されてきた Groupware などの研究成果が紹介され，Groupware をソフトウェア開発の種々の段階に適用した事例が紹介された．今後は，従来，別々の community で研究されてきた，ソフトウェア工学と CSCW の研究成果をうまく融合していく必要性があると主張した．

3 パネル討論

パネル討論は，7 日，8 日，9 日の 3 日間行なわれた．以下，8 日に行なわれた “are Formal Methods Relevant” のパネル討論について報告する．オーストラリアのクイーズランド大学の Ian Hayes 教授が議長を勤め，奈良先端大の荒木啓二郎教授，イギリスのヨーク大学の David Duke 博士，そして Murdoch 大学の Val Veraart 教授がパネルメンバーであった．最初に，各パネルメンバーが，幾つか異なった側面からフォーマルメソッドを紹介し，その後，質疑応答に移り，どのフォーマルメソッドがシステム開発に

適当かということについて興味深い討論が行なわれた．

本討論は，Ian Hayes 教授がフォーマルメソッドに関して彼の立場から意見を述べることにより始まった．彼によれば，フォーマルメソッドはソフトウエア開発に関する幾つかの目的のために用いることができる．まず第一に，フォーマルメソッドはプログラミングのために以下の 2 つの意味で利用されている．

1. プログラミング言語を利用すること．すなわち，プログラムすることが形式言語を書くことであるということ．
2. 適当な変数間の関係を理解することにより，プログラムがどのように働くかを理解すること．

次に，フォーマルメソッドは，次の 2 つの側面の仕様のために用いることができる．

1. フォーマルメソッドは，多くのプログラミング言語でフォーマルメソッドとして利用可能なものよりも，プログラムの状態を高い抽象化レベルを用いて正確に記述することが可能である．そのため，プログラムの詳細を調べることなしに，プログラムの機能を簡単に理解することができる．
2. 形式仕様は，実装の正しさを証明 (verify) する基礎として，また，さらに形式仕様から実装を導出する基礎として利用することができる．

第三に，フォーマルメソッドは正確にシステムをモデル化する手助けができ，そして要求獲得中に実装の詳細を抽象化することができる．最後に，彼はフォーマルメソッドの教育に関する 2 つの問題点を指摘し，既存のフォーマルメソッドの成熟度について述べた．

荒木啓二郎教授は，フォーマルメソッドの 7 つの神話について日本ではどのように考えられているかについて紹介した．7 つの神話とは以下である．

Myth 1. フォーマルメソッドはソフトウエアが完全であることを保証できる．

Myth 2. フォーマルメソッドはプログラムが提供することについての全てである．

Myth 3. フォーマルメソッドは安全性に厳密なシステムにのみ役に立つ．

Myth 4. フォーマルメソッドは高い教育を浮けた数学者を必要とする．

Myth 5. フォーマルメソッドは開発費を増大させる．

Myth 6. フォーマルメソッドはユーザーに受け入れられない．

Myth 7. フォーマルメソッドは現実的で大規模なソフトウエアには使わない．

Anthony Hall が挑んだ上記 7 つの神話を列挙した後，彼は，日本のほとんどの人々が同じような神話を持っているであろうが，一方で，その多くの人々がフォーマルメソッドを全く認めていないようであると指摘した．日本の研究者はフォーマルメソッドには興味があるが，彼らは，小規模なプログラムに対しての練習または実験的応用を単に楽しんでいるにすぎない．そして，大学や企業内にフォーマルメソッドのチュートリアルコースがほんの少ししかないと述べた．

David Duke 博士は，フォーマルメソッドが safety-critical または high-integrity システムの開発に役割を果たしている形跡はあるが，定量化と解析に通じているにすぎないと述べている．

彼は，フォーマルメソッドをヒューマン–コンピュータインタラクションへ適用する最近の開発に伴って起きた問題が，ソフトウエア工学でフォーマルメソッドが適当かどうかという疑問を，ますます一般的な問題として扱うことを可能にしたことを指摘した．最後に，フォーマルメソッドの利点は，複雑なシステムの挙動の適当な側面を，正確にそして扱い易く捕らえることを可能にするばかりでなく，フォーマルメソッドが，人材や時間の費用や設計内容などの制限に合うようにする方法を開発者に依存できることにある．

Val Veraart 教授は，最初に，インフォーマル，セミフォーマル，そしてフォーマルを区別することにより問題を述べ，そしてそれから，彼の経験と研究をもとに，フォーマルメソッドが捕捉的なものとして方法論を支援するのに適しているということを述べた．彼は，フォーマルメソッドはより高い開発費を必要とし，少なくとも開発立ち上げ初期の費用は高くなることを指摘した．従って，フォーマルメソッドが加え

た最終プロダクトの価値が，投資した分に匹敵することを確実にしなければならないと述べた．

以上のパネルメンバーの講演の後，質疑応答に移り，聴衆とパネルメンバーとの間で，興味深い討論が行なわれた．

産業界の一般参加者の何人かが，産業用環境で，しかも競争市場で，複雑なシステムを開発するためのフォーマルメソッドの有用性と能力について語った．その主旨は，フォーマルメソッドは利益を達成する手助けになるが，財源が必要であり，実際的な制約に耐えるのは難しいということであった。

一方，オーストラリアとニュージーランドの大学の教授達の何人かは，フォーマルメソッドを使うことは大きな困難ではなく，フォーマルメソッドに用いられている関数，集合理論，論理の概念や記法はプログラミング言語に良く使われているものであると述べてた．クイーズランド大学の Roger Duke 教授は，さらに，フォーマルメソッドの教育の欠如が，フォーマルメソッドの普及において最大の障害であると述べた．一方，広島市立大学の Shaoying Liu 教授は，この障害はフォーマルメソッドの教育の欠如ばかりでなく，既存のフォーマルメソッドは本質的に難しさと未熟さを持っており，そしてそれゆえに，適当な構造化手法，オブジェクト指向法そして既存のフォーマルメソッドを統合することで，より強力で，ユーザーとの親和性が高く，そして実際的なフォーマルメソッドを開発することが，フォーマルメソッドを広く普及させる重要な一歩であると述べた．

他のフォーマルメソッドの重要な話題としては，仕様変更，システム管理，そしてシステム保守が議論された．

4 論文発表

論文の話題は，以下に示したセッションからわかるように、ソフトウエア工学に関する様々な分野に広くわたっていた。

1A Testing 1

1B Requirements Modelling

1C Concurrency

2A Object-Oriented Development

2B Software Process

2C Distributed Systems

3A Testing 2

3B Development Environments 1

3C Formal Methods

4A Quality Assurance and Reliability

4B Reuse

4C The Specification Process; Maintenance

5A Development Environments 2

5B Information Systems Design and Management

5C Reasoning and Verification

以上のセッションについて 3 ~ 4 つの発表が行なわれた．特に，Testing と Development Environments は 2 セッション設けられ，目だっていた．以下に発表件数が多かった Testing，Development Environment，そして Requirements Modelling と Object-Oriented Development についての感想を述べる．

■ **Testing**

テストに関するセッションは，7 日と 8 日の午前に 2 つ設けられ，合計 7 件の発表があった．その内 3 件が日本の発表であり，日本のテストの分野への関心の高さを感じた．テストケースの生成に関する話題が多く，特にオブジェクト指向と並行性に関するテストの話題が 2 件づつあった．

オブジェクト指向システムのテストに関しては，7 日と 8 日各々に以下の 2 件の発表があった．

1A-1 “Dynamic Data Flow Analysis for C++”

T. Y. Chen, C. K. Low

C++プログラムをテスト可能にするために，既存のデータフロー解析法を拡張する方法について提案している．その拡張法として，既存のデータフロー解析で用いている状態遷移図を拡張し，新しい状態遷移図を提案し，状態をクラスと変数で，挙動をクラスと型で示せるようにしている．この方法により，変数やクラスメンバーのメモリ位置に関するエラーの発見を可能にし，さらに，メモリー位置に基づく挙動のシーケンスの追跡を可能にしている．

3A-3 "Testing of Object-Oriented Programs Based on Finite State Machine"
H. S. Hong, Y. R. Kwon, S. D. Cha
オブジェクト指向プログラムのためのテストケースを自動生成するために，Class State Machine(CSM) を提案している．CSM は，拡張有限状態機械の一種であり，状態，遷移そしてクラスのデータメンバからなる．CSM によりクラスのテストを可能にしている．

並行システムためのテストケース生成に関しても，7 日と 8 日各々に以下の 2 件の発表があった．

1A-2 "Event Interactions Graph for Test-case Generation of Concurrent Programs"
T. Katayama, Z. Furukawa, K. Ushijima,
並行プログラムの挙動を表現する Event InterActions Graph(EIAG) と，テストケースとして EIAG 上の協調パス (cooperated path) を提案している．協調パスは可達でない状態や通信エラーの発見ができる．

3A-4 "Test Suite Generation Methods for Concurrent Systems based on Coloured Petri Nets"
H. Watanabe, T. kudoh
カラーペトリネットにより並行システムをモデル化し，カラーペトリネットの可達グラフのパスからテストケースを生成する方法を提案している．この方法は，可達グラフを求める際，同値マーキングという方法を使い，テストケースの数の削減に勤めている．

オブジェクト指向を扱った発表者は，今後，並行性について取り組んでいきたいと述べ，逆に，並行性を扱った発表者はオブジェクト指向に取り組んでいきたいと述べていた．近い将来，両方の性質を網羅するテストケースの生成法が可能になり，より実際的な複雑で大規模なシステムのテストが可能になることであろう．

ところで，実際的なシステムの場合，生成したテストケースは非常に大きくなり，従って，実際の開発では，テストを終了する時期を決定することは大きな問題となる．本会議ではテストを終了する時期を決定する方法についても以下の提案があった．

1A-3 "A criticism on the capture-and-recapture method for software reliability assurance"
S. Isoda
Capture-and-Recapture 法 (CR 法) は，テスト工程の最中に残りのソフトウエアのバグの数を見積もることを可能にする方法，すなわち，テストを終了する時期を決定することを可能にする方法である．埋め込まれたバグの割合いのための信頼間隔見積りを与える式を導入している．

他に，以下の発表があった．

3A-1 "On the analysis of subdomain testing strategies"
T.Y. Chen, H. Leung and Y.T. Yu
partition testing を拡張し，重複するサブドメインのテストを可能にしている．また，partition testing をランダムテストと比べた場合の有効性について述べ，そして，このテストの有効性について示すために，平均故障率を求めている．

3A-2 "An enhanced zero-one optimal path set selection method"
C.G. Chung and J.G. Lee
zero-one optimal path set selection 法の短所は，プログラムが大きくなるにつれて，計算量が指数関数的に増加するため，大規模なプログラムに適用できないことであった．彼らは，この問題解決のために，この方法を拡張し，5 つの縮約則を定義し，選択したパス集合を再利用できるようにした．

■ **Requirements Modelling**

この session では，3 件の発表があった．

B-1 "From Scenarios to Timed Automata: Building Specifications from User Requirements"
S. Some, R. Dssouli, J. Vaucher
Requirements の表現手段として良く用いられる，システムを部分的に記述したシナリオ片を時間付きオートマトンに自動的に変換するアルゴリズムが提案された．ATM(Auto Teller Machine) のシナリオの例に対する適用例も紹介された．

B-2 “Expressing Inter-Perspective Reletionships: A Logical Approach”
L. Perrussel
異なる人々・異なる表現方法で記述された，Multi-Viewpoint の仕様に対し，viewpoint 内の fragment 間の関係を規定する為の枠組として Multi-agent system を利用することが提案され，その formal な取扱いについて論じられた．

B-3 “Building the Structure of Specification Docuements from Utterances of Requirements Elication Meetings”
N. Miura, H. Kaiya, and M. Saeki
要求仕様作成会議中にとりあげられた話題の時間的隣接関係を用いて，仕様書を構造化する方法が提案された．Case Study と作成された仕様書の評価実験についても触れられた．

■ **Object-Oriented Development**

このセッションは，オブジェクト指向開発のための CASE ツールに関する論文，クラス階層の再構成に関する論文，オブジェクト指向プログラムに対するメトリクスに関する論文，および，オブジェクト指向の実務経験者へのアンケート調査結果に関する論文の４編から構成されていた．ここでは，アンケート調査結果に関する以下の論文について概説する．

“A Survey of Experiences amongst Object-Oriented Practioners”, J. Daly, J. Miller, A. Brooks, M. Roper, and M. Wood (U. of Strathclyde)

この論文はオブジェクト指向パラダイムで叫ばれている主張の妥当性を検証する研究の一部として，１９９４年の上半期にオブジェクト指向の実務経験者を対象になされたアンケート調査の結果を報告している．有効な回答は２７５と決して多いとは言えないが，全体の約４６％はオブジェクト指向のソフトウェア技術者であり，全体の６５％以上が３年以上の実務経験者であったと報告された．また，使用言語の上位３つは，C++ （５７．８％），Objective-C （３６．６％），Smalltalk（３３．８％）と圧倒的にＣ系の言語であった． アンケートはオブジェクト指向パラダイムの利点，継承，オブジェクト指向プログラムの理解の困難さ，C++ プログラミング言語等，６つの項目から構成され，それぞれについてその結果が報告された．以下，結果の一部を概説する． まず，オブジェクト指向パラダイムは従来のパラダイムに比べ，解析と設計の容易性，再利用性，生産性，保守容易性で優れているとの結果が出ていた．オブジェクト指向の実務経験者への調査であることを考えると妥当な結果に思える．次に，オブジェクト指向ソフトウェアの理解を困難にする要因の一つとして継承が挙げられた．その他に挙げられた要因は設計文書の欠如あるいは不備，および，貧弱あるいは不適切な設計があった．しかしながら，これら２つの要因は従来のパラダイムにも存在している問題であり，オブジェクト指向のみの問題ではない．また，回答者の多くが C++言語を実務で用いているのに，C++ は標準的なオブジェクト指向言語でないと回答していること（約６０％）は，興味深い．

■ **Development Environments**

このセッションは，分散ソフトウェア開発の統合支援環境に関する論文，形式プログラム開発のための支援環境に関する論文，プログラミング言語とその環境機構とを記述するための形式に関する論文，および，コンフィギュレーション管理の支援環境に関する論文の４編から構成されていた．ここでは，以下の論文について概説する．

Modelling systems that integrate programming language and environment mechanisms, K. J. Ransom and C. D. Marlin (Flinders U. of South Australia)

この論文は，GUI を備えたＷＳ上のソフトウェアの統合開発環境において，プログラムコードの入力や表示等の表現を柔軟にするために，プログラミング言語と環境機構とを統合して扱うための形式について述べている． しかしながら，内容的はまだ概念レベルの段階であり，実際の開発環境へのこの形式の適用，および，既存のビジュアルプログラミング環境との差異が明確にされていないこと残念であり，今後に期待したい．

5 チュートリアル

8 種類のチュートリアルが 4 本づつ並行に初日の午前と午後行なわれた．

T1 “A Holistic Approach to Software Measurement”
Ray Offen, Ross Jeffery

T2 “Integrated Computer-Aided Software Engineering Environments”
Chris Martin

T3 “Software Architecture and Iterative Development Process”
Philippe Kruchten

T4 “Introduction to Process Management Concepts and the Capability”
Judy Bamberger

T5 “Knowledge-Based Software Maintainance”
Paul A. Bailes

T6 “Enterprise Object Modeling”
Richard T. Due

T7 “Software Quality Evaluation and Certification”
Hans-Ludwig Hausen

T8 “Components of an Effective Process Improvement Program”
Judy Bamberger

著者らが受講したチュートリアルは，T1 ，T2 ，T6 ，T8 である．以下に著者らが受講したチュートリアルの感想をまとめる．

■ **T1: A Holistic Approach to Software Measurement**

このチュートリアルでは，前半を Ray Offen 教授が measurment について，モデルの役割と，基礎理論について述べられ，後半を，Jeffery 教授が実際的な理論と幾つかの例について述べられた．実際的な理論としては，Jeffery-Offen Model, Measure, Manage Paradigm (M3P) について紹介があった．例として，ソフトウエア会社，電気通信会社，政治組織，保険会社を上げた．

■ **T2: Integrated Computer-Aided Software Engineering Environments**

従来，各ソフトウェア開発工程別に開発されてきた CASE tool を紹介し，それらの問題点と Case tool の統合の必要性を説いた．さらに，いくつかの統合化 Case Tool と Case tool 統合化の枠組の一つとして PCTE(Post Common Tool Enviroment) が紹介された．さらに，ソフトウェアプロセスモデリングの話にも触れ，統合 Case Tool の研究との関わりについて述べた．

■ **T6: Enterprise Object Modeling**

オブジェクト指向パラダイムの一つである Enterprise Object Modeling について紹介があった．Enterprise Object Modeling は，Enterprise Modeling，Business Process Engineering そして Workflow Analysis が結びついたものであり，組織の計画，制御，通信機能，そして情報システムやデータプロセシングシステムの経済的，効率的な管理をして生産性を増加させることに役立つ．

■ **T8: Componets of an Effective Process Improvement Program**

いわゆる講演の形式ではなく，参加者を少人数のグループに分け，ソフトウェアプロセス改善について実際に議論を行うという形で行われた．まず，効率的な会議の進め方ということで，会議の agenda・目的・時間・各参加者の役割を明確にすべきことを述べ，この後の議論はこの指針に従って行われた．続いて，問題解決のための諸技術 (brain storming，KJ 法のようなカードに書き留めた内容のグルーピング・構造化など)，意志決定を定量的に行うためのツール，実際のプロセス改善活動の計画方法などを実践し，模擬的なプロセス改善活動を行った．

6 その他のイベント

レセプションは 6 日，7 日，8 日の 3 回行なわれた．6 日 7 日は立食形式の簡単なものであったが，クリスマスの時期であったため，8 日はクリスマスパーティーの形式で行なわれ，余興なども披露された．そのため，楽しい雰囲気の中，参加者同士の親好を深めることができ好評であったと思う．また，昼食も立食形式で準備されていたため，昼食時にも多くの方々と議論や歓談を行なうことができた．

7　おわりに

本稿では，第 2 回アジア・太平洋ソフトウエア国際会議の内容を概説した．前回と比べると，最終日 1 回のみであったパネルが 3 回になったこと，チュートリアル日が 1 日加わり，盛況であったこと，そして，何よりも投稿論文数が 92 件から 160 件へと大幅に増えたことから 2 回目は，初回よりも充実した会議になったと思われる．

残念であったことは，前回より多少は増えたものの，産業界からの参加がやはり非常に少なかったことである．そのため，基礎的な内容や理論的な内容に傾いていたように思われる．また，今回は，ツールの展示がなかったことが，少し寂しく感じられた．今後，産業界からもさらに多くの参加を期待し，実際的な内容の論文の数が増え，ツールの展示も盛んに行なわれるようになることを願っている．

本会議のプログラムは，
http://www.cs.uq.oz.au/conferences/apsec95/から得ることができる．

また，アブストラクトも
http://www.cs.uq.edu.au/conferences/apsec95/abstracts/で得ることができる．また，チュートリアル，一部の基調講演，パネル討論以外についての各講演の詳細な内容については以下の会議録に掲載されている．

Proceedings of Asia-Pacific
Software Engineering Conference.
IEEE computer Society Press, 1995
ISBN 0-8186-7171-8

次回は，韓国ソウルで開催が予定されている．次回の会議がさらに盛況で充実したものになることを期待したい．