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

2005－SE－148（7）
2005／5／31

ソフトウェア・アーキテクチャ研究動向の調査報告概要

立堀 道昭　　丸山 宏　　小林 真　　Daniel Yellin
IBM
吉田 尚志　　川井 奈央
IPA/SEC

概要

IPA/SEC の活動の一環として行なった、最近のソフトウェア・アーキテクチャ研究動向調査の内容と結果について概要を発表する。ソフトウェアシステムを開発する際、その抽象的なアーキテクチャの設計が、システム開発全体に及ぼす影響が大きいことは広く知られている。ソフトウェア・アーキテクチャの研究は、その意味で、ソフトウェア工学上重要なものである。本研究動向調査では、1999 年以降の学術文献を広く調査してその動向をみた。さらに、この研究領域で活躍している研究者 10 名へのインタビューを通して、文献調査結果の検証を行い、また、文献調査からは得られない知見を得た。本発表では、IPA/SEC へ行なった調査報告から抜粋し、調査の内容・方法と調査結果について簡単に報告する。

A Survey Report Digest on Research Trends in Software Architecture

Michiaki Tatsubori　Hiroshi Maruyama　Makoto Kobayashi　Daniel Yellin
IBM
Hisashi Yoshida　Nao Kawai
IPA/SEC

Abstract

This presentation reports a digest of our survey report on research trends in Software Architecture, which has been done as an activity of IPA/SEC. It is well-known that in developing a software system the abstract architectural design of the system affects much on the whole software development. In this sense, the research on software architecture is very important as a software engineering research. In this survey, we extensively investigated academic papers published since 1999 to observe their trends. Additionally, we conducted interviews with 10 researchers in the area of Software Architecture to further validate and deepen our analysis. In this presentation, extracting from our report for IPA/SEC, we briefly explain our methodology and the results of our investigation.

1. はじめに

ソフトウェアシステムを設計・構築する際には、システムの利害関係者の要求を満たすようにする必要がある。ここでいう利害関係者には、エンドユーザ、システムの影響を受ける組織的なプロセスに関わる管理者やその他の人々、システムの開発や保守に携わる技術者、顧客、ビ

ジネス上の協業や法規上の規程などに絡んでシステムに影響を与えるような外部組織などが含まれる。

ソフトウェア開発過程において、実際にシステムを実装する前に、よいアーキテクチャを設計することは非常に重要である。1980 年初頭、ソフトウェア開発の問題に直面していた現場では、ソフトウェア・アーキテクチャの概念の重要さが認められつつあったものの、当時、人々は様々なアドホックな解決方法を用いるのみであった。1990 年代に入り、研究者は、より洗練された手法を確立し始めた。また、同時期、ソフトウェア開発の現場では、複雑なソフトウェアシステムを設計するために、方式、技法、パターンやイディオムなどのレポジトリを共有しながら利用することの重要さが認識されはじめてきた。

本稿では、我々の行なった、ソフトウェア・アーキテクチャ領域における最近の研究動向調査について IPA/SEC に報告したもの [1] の概要を述べる。近年、ソフトウェア・アーキテクチャに関して、既にいくつかの有名な書籍が出版されてはいるものの、この領域における研究はまだまだ活発に行なわれている。そのため、今日、当領域のどういった分野が活発に研究されているのか、または、されていないのかはよくわからない。本動向調査は、これを幾分でも明らかにすべく行なわれたものである。

本調査は、IPA/SEC の活動の一環として行なわれた。SEC (Software Engineering Center) は、独立行政法人情報処理推進機構 (IPA) のもと、日本におけるソフトウェア工学発展の産官学共同推進の拠点となるべく、昨年設立された機関である。SEC は、ソフトウェア工学における重要な領域の１つとしてソフトウェア・アーキテクチャに着目している。

本調査の目的は、ソフトウェア・アーキテクチャ研究領域における最近の活動を眺めて解析することにより、近い将来の研究動向を掴むことにある。研究動向への影響は、古いものより、最近の研究成果のほうが大きいと考えたため、我々は、直近（過去６年間）の研究活動に着目することにした。

典型的なサーベイ論文は、著者の知識や経験を基に書かれるものであるが、本調査では、そのような方法によると生じがちな偏りをなるべく避けたかった。そのため、我々は、調査の基盤として、定量的な手法により、研究動向に大きく寄与していると思われる研究成果の評価を行なうことから始めた。この定量的評価と評価結果の解析に基づき、我々は、ソフトウェア・アーキテクチャ領域で活躍し、産学のコミュニティに多大な影響を及ぼしていると思われる研究者の一部にインタビューを試みた。このインタビューを通して、我々の文献調査結果の検証し、また、文献調査だけからでは得られにくい、研究背景の詳細や実際に関する知見を得た。

2. 調査方法

本節では、研究動向を分析する上で我々がとった方法を説明する。我々は、定量的な研究動向評価のために、我々の用意した分類方式により研究成果公表文献を分類した。そこで以降、まず、我々の分類方式について述べ、それから、我々が用いた研究動向の推定方法について述べる。

分類の枠組み

本調査上、ソフトウェア・アーキテクチャの研究の切り口として次の３つの軸を考慮した。

- ソフトウェア開発プロセスにおいてソフトウェア・アーキテクチャの果たす役割を考慮した研究であるかという観点（役割による分類）
- 産業界におけるどのプラクティスに最も関連しているかという観点（プラクティスによる分類）
- ソフトウェア開発上必要な作業をどの程度機械化するのかという、研究の目的とする解決策の観点（機械化の程度による分類）

１つめの、役割による分類では、次の５つの役割のうちのどれに着目しているかにより研究を分けることにした。

- [R] アーキテクチャへの要件（Requirement）– 大抵のソフトウェア開発は、様々な利害関係者によるシステムへの要件をアーキテクチャに反映することのできるような形で定義することから始まる。
- [M] アーキテクチャのメタモデル（Metamodel）– アーキテクチャの設計は、

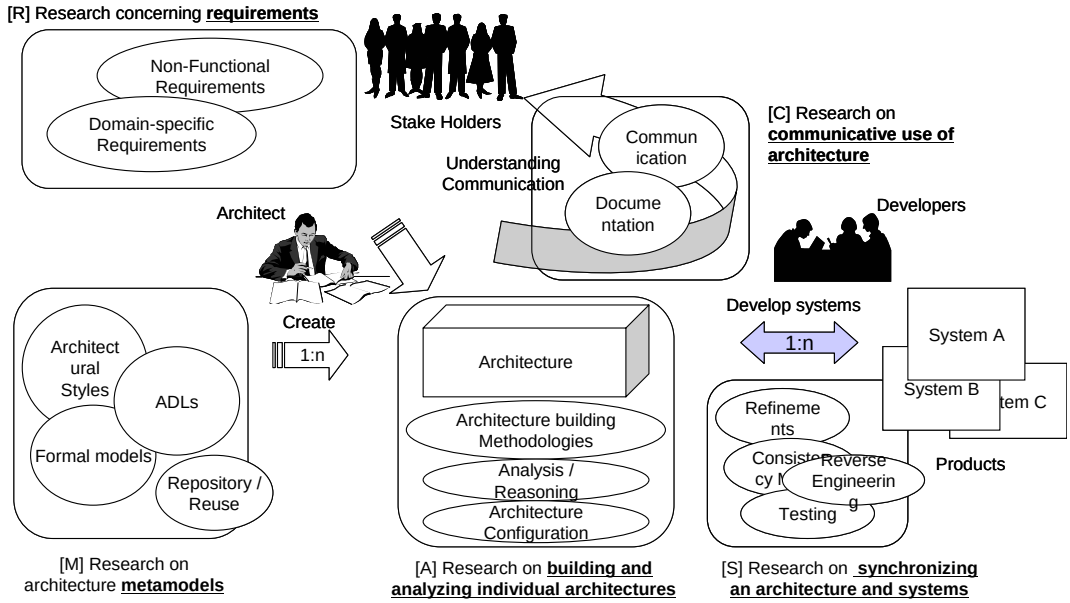

図 1　ソフトウェア・アーキテクチャの役割による分類

記述言語や参照アーキテクチャなど、アーキテクチャを規定するためのメタモデル的な何かに基づいてアーキテクトが行なう。

- [A] アーキテクチャの構築・解析（Analysis）– アーキテクチャを実際に構築・修正する。その際、構築したものが要件を満たしているかどうか解析する必要がある。
- [C] アーキテクチャを用いた利害関係者とのコミュニケーション（Communication）– 全ての要件を満たすことができない場合、要件を調整するために、アーキテクチャを用いて利害関係者と交渉する。
- [S] アーキテクチャとシステム間の同期（Synchronization）– 抽象的なアーキテクチャは、実際に動くシステムの実装に落とさなければならない。また、逆に、システムに変更があった場合、それはアーキテクチャにも反映されるべきである。

図 1 に、これらの役割を示した。

２つめの、プラクティスによる分類では、それぞれの研究がその動機として、ソフトウェア開発現場や産業界で重んじられているどのような事柄を問題領域としてとらえているかにより、研究を分けることにした。我々は、そのような事柄として次のようなものを考えた。

- [RUP] Rational Unified Process (RUP) など、ソフトウェア開発プロセスの（事実上の）標準 – ソフトウェア・アーキテクチャを用いる現場で、一種の制約である。
- [PL] ソフトウェア・プロダクトライン – 様々なバリエーションの製品の開発を容易にする概念。
- [Pat] パターン – 分析、設計、実装など、様々な局面で、典型的に用いられる解法を抽象化し、解法が用いられる文脈や目的とともにカタログ化したもの。
- [UML] Unified Modeling Language (UML) – ソフトウェア設計記法の（事実上の）標準。
- [MDA] Model-Driven Architecture (MDA) – モデルありきでシステム開発を行なっていくパラダイム。
- [CS] ケース・スタディ– 現実のソフトウェア開発過程や結果を観測・解析する。
- [DSA] Domain-Specific Architecture (DSA) – アプリケーション領域を特定し、狭まった問題領域特有の解法を考える。

DSA では、さらに、どのアプリケーション領域を対象としているかにより分類できる。例えば、組み込み領域や分散計算領域などである。

本調査では、DSA に属すると判断された研究について、この分類も行なった。

最後の、機械化の程度による分類では、それぞれの研究がその目的として、ソフトウェア・アーキテクチャを用いたソフトウェア開発上の作業をどのように簡易化しているかにより、研究を分けることにした。作業の簡易化には様々なアプローチがありえるが、我々は特に、何もなければ人間が行なわなければならないような作業をどの程度機械にさせるかという観点に着目した。そのような機械化の程度として次の３つの程度を考えた。

- [level 0] マニュアル・レベル – ほとんどの作業を人間が行なう。
- [level 1] 半自動化レベル – 機械による大きな支援を得ながら人間が行なう。
- [level 2] 全自動化レベル – 人間はわずかな指示や選択を行なうのみであり、後は機械が自動的に行なう。

これら３つの分類軸において、実際には、複数の役割、複数のプラクティス、複数の機械化の程度にまたがる研究もあるが、本調査では、それぞれどれか 1 つに属するものとした。その裁定は、研究の重心が主にどちらにあるかを、著者らが研究の内容をみることにより判断して行なった。

研究動向の推定方法

本調査では、研究動向の推定上調査対象とする文献の抽出範囲を、特定の国際会議における 1999 年以降のフル・ペーパーに絞った。直近の研究動向を調べる上で、ある程度の内容の新鮮さと品質を担保するために、ジャーナルやワークショップの論文は除外した。国際会議は、我々の内部の国際会議データベースを利用し、学際的産業的なインパクトが大きく重要であると考えられる国際会議の中で、ソフトウェア工学やソフトウェア・アーキテクチャに関連する 19 の国際会議に絞った。これにより約 3,400 の論文が対象となった。

注目すべき論文の特定のために、我々は、年あたりの引用数を用いた。（引用数の計算には Google Scholar を利用した。）年あたりの引用数が最も多い上位 40 論文を注目すべき論文であるとして抽出した。ただし、2004 年の論文は引用が期待できなかったため、19 国際会議の中で、我々が特に重要と考えた国際会議（主要国際会議）に絞った上で、ソフトウェア・アーキテクチャに関する論文 11 個を抽出した。

本調査上、特に重要な国際会議（主要国際会議）として用いたのは、次に示す４つの国際会議である。

- ICSE – International Conference on Software Engineering
- FSE – ACM SIGSOFT Symposium on the Foundations of Software Engineering
- OOPSLA – International Conference on Object-Oriented Programming, Systems, Languages, and Applications
- ECOOP – European Conference on Object-Oriented Programming
- また、残りの 15 は、次に示す国際会議である。
- WICSA – Working IEEE/IFIP Conference on Software Architecture
- SPLC – Software Product Line Conference (including International Conference on Product Family Engineering)
- AOSD – Aspect-Oriented Software Development Conference
- BPM – International Conference on Business Process Management
- CD – International Working Conference on Component Deployment
- EDOC – International Enterprise Distributed Object Computing Conference
- FASE – International Conference on Fundamental Approaches to Software Engineering
- GPCE – International Conference on Generative Programming and Component Engineering
- ICCBSS – International Conference on COTS-Based Software Systems
- ICECCS – International Conference on Complex Computer Systems
- ICSOC – IEEE International Conference on Service-Oriented Computing
- ICWS – IEEE International Conference on Web Services
- RE – International Conference on Requirements Engineering
- UML – International Conference on the Unified Modeling Language

表 1　年あたりの引用数（ave./year）により抽出された上位 40 論文のリスト

Conf.	Year	Authors	Title	citations	ave./year
WICSA	2002	Joao Pedro Sousa, David	Aura: an Architectural Framework for User Mobility in Ubiquitous Co	61	20.33
ICSE	2002	Jonathan Aldrich, Craig C	ArchJava: connecting software architecture to implementation	60	20.00
ICSE	2000	Nikunj R. Mehta, Nenad M	Towards a taxonomy of software connectors	91	18.20
ICSE	2003	Don S. Batory, Jacob Nea	Scaling Step-Wise Refinement	33	16.50
ICSE	1999	Nenad Medvidovic, David	A Language and Environment for Architecture-Based Software Deve	97	16.17
ICSE	2002	Eric M. Dashofy, Andre va	An infrastructure for the rapid development of XML-based architectu	37	12.33
ICSE	1999	Rick Kazman, Mario Barb	Experience with Performing Architecture Tradeoff Analysis	70	11.67
ICSE	1999	Ivan T. Bowman, Richard	Linux as a Case Study: Its Extracted Software Architecture	67	11.17
ICSE	1999	Rudolf K. Keller, Reinhard	Pattern-Based Reverse-Engineering of Design Components	66	11.00
UML	2000	David Garlan, Andrew Koi	Reconciling the Needs of Architectural Description with Object-Mod	52	10.40
WICSA	1999	Christine Hofmeister, Rob	Describing Software Architecture with UML	61	10.17
ECOOP	2002	Jonathan Aldrich, Craig C	Architectural Reasoning in ArchJava	29	9.67
WICSA	2001	Eric M. Dashofy, Andre va	A Highly-Extensible, XML-Based Architecture Description Language	36	9.00
ICSE	1999	Jean-Marc DeBaud, Klau	A Systematic Approach to Derive the Scope of Software Product Lir	50	8.33
ICSE	1999	Jan Bosch	Product-Line Architectures in Industry: A Case Study	48	8.00
ICSE	2003	Anita Sarma, Zahra Noroo	Palantir: Raising Awareness among Configuration Management Works	16	8.00
WICSA	2002	Shang-Wen Cheng, David	Using Architectural Style as a Basis for System Self-repair	23	7.67
UML	1999	Luis Filipe Andrade, Jose	Interconnecting Objects Via Contracts	45	7.50
PFE	2001	Jan Bosch, Gert Florijn, D	Variability Issues in Software Product Lines	28	7.00
ICSE	1999	Eric M. Dashofy, Nenad M	Using Off-the-Shelf Middleware to Implement Connectors in Distribu	41	6.83
SPLC	2000	Martin L. Griss	Implementing Product-Line Features by Composing Component Aspe	33	6.60
FSE	2001	Sebastian Uchitel, Jeff Kı	Detecting implied scenarios in message sequence chart specification	25	6.25
ICSE	2000	Roy T. Fielding, Richard N	Principled design of the modern Web architecture	31	6.20
WICSA	1999	Nenad Medvidovic, David	Assessing the Suitability of a Standard Design Method for Modeling	37	6.17
ICSE	1999	Elisabetta Di Nitto, David	Exploiting ADLs to Specify Architectural Styles Induced by Middlewa	36	6.00
OOPSLA	2001	Dirk Riehle, Steven Fralei	The Architecture of a UML Virtual Machine	23	5.75
WICSA	1999	Mark H. Klein, Rick Kazm	Attribute-Based Architecture Styles	34	5.67
WICSA	1999	Jeff Magee, Jeff Kramer,	Behaviour Analysis of Software Architectures	33	5.50
ECOOP	2003	Jonathan Aldrich, Vibha S	Language Support for Connector Abstractions	11	5.50
GPCE	2002	Sandeep Neema, Ted Bap	Generators for Synthesis of QoS Adaptation in Distributed Real-Tim	16	5.33
ICSE	2001	Rick Kazman, Jai Asundi,	Quantifying the Costs and Benefits of Architectural Decisions	21	5.25
WICSA	2001	Bridget Spitznagel, David	A Compositional Approach for Constructing Connectors	21	5.25
ECOOP	2000	Marcus Fontoura, Wolfgar	UML-F: A Modeling Language for Object-Oriented Frameworks	26	5.20
FSE	1999	Pascal Fradet, Daniel Le	Consistency Checking for Multiple View Software Architectures	31	5.17
RE	2002	Jon G. Hall, Michael Jack	Relating Software Requirements and Architectures Using Problem Fr	15	5.00
SPLC	2002	Jan Bosch	Maturity and Evolution in Software Product Lines: Approaches, Artef	15	5.00
RE	1999	John C. Grundy	Aspect-Oriented Requirements Engineering for Component-Based S	29	4.83
FSE	2001	Nima Kaveh, Wolfgang Em	Deadlock detection in distribution object systems	19	4.75
FSE	1999	Michel Wermelinger, Jose	Algebraic Software Architecture Reconfiguration	27	4.50
ICSE	2003	Bridget Spitznagel, David	A Compositional Formalization of Connector Wrappers	9	4.50

表 2　主要国際会議から抽出された 11 論文のリスト

Conf.	Year	Authors	Title
FSE	2004	Sebastian Uchitel, Robert Chatley, Jeff Kran	System architecture: the context for scenario-based model synthe
FSE	2004	Xiaofang Zhang, Michal Young, John Howard	Refining code-design mapping with flow analysis
ICSE	2004	Wilhelm Hasselbring, Ralf Reussner, Holger J	The Dublo Architecture Pattern for Smooth Migration of Business
ICSE	2004	Mari Matinlassi	Comparison of Software Product Line Architecture Design Method
ICSE	2004	Mauro Caporuscio, Paola Inverardi, Patrizio I	Compositional Verification of Middleware-Based Software Architec
ICSE	2004	Mark Grechanik, Don S. Batory, Dewayne E.	Design of Large-Scale Polylingual Systems
ICSE	2004	Alexandre R. J. Francois	A Hybrid Architectural Style for Distributed Parallel Processing of
ICSE	2004	Rohit Khare, Richard N. Taylor	Extending the Representational State Transfer (REST) Architectur
ICSE	2004	Hong Yan, David Garlan, Bradley R. Schmerl	DiscoTect: A System for Discovering Architectures from Running S
ICSE	2004	Bas van der Raadt, Jasper Soetendal, Michie	Polyphony in Architecture
ICSE	2004	Ian Gorton, Jereme Haack	Architecting in the Face of Uncertainty: An Experience Report

- WWW – International World-Wide Web Conference

本調査では、以上の過程により抽出された全 51 論文について、先に述べた分類の枠組みを用いて分類してその傾向をみた。ここで、これら 4 主要国際会議や、全 19 国際会議の妥当性をここに示すことはしないが、調査対象として十分適切であったと我々は思っている。抽出された論文のリストを表 1 と表 2 に示しておく。

以降、本稿では、抽出された文献の分類・調査から得られた結果とインタビュー結果をまとめたものから、特に、役割による分類からみられた研究動向を詳しく紹介する。

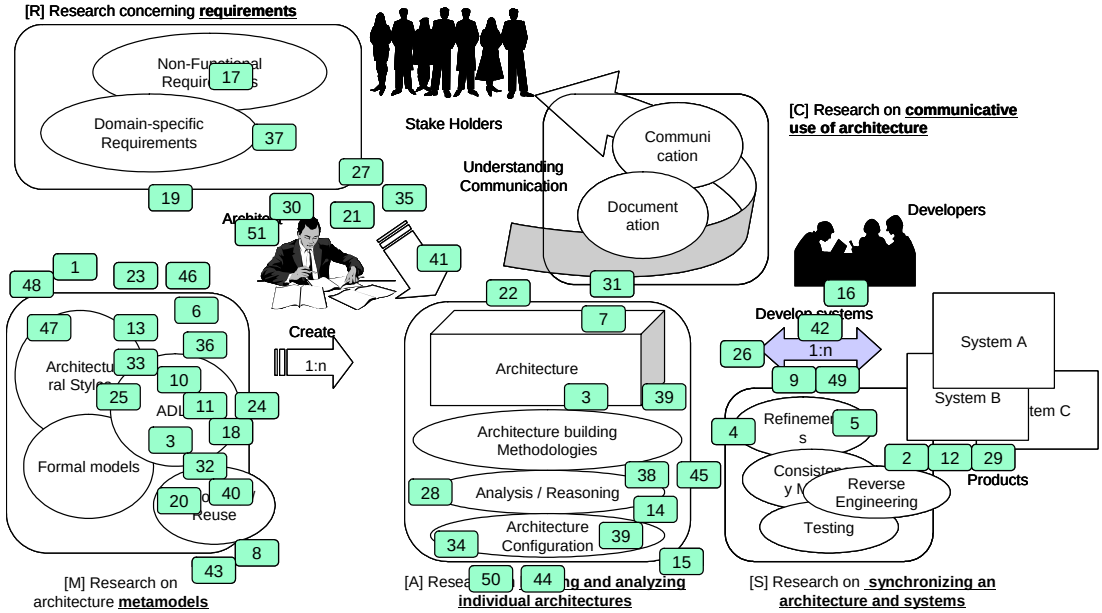

図 2　図 1 に抽出論文をマップしたもの

3. 役割による分類からみられる研究動向

抽出された 51 論文を役割に分類した結果を図 1 にマップしたものは図 2 のようになった。各番号は、リストにおける各論文の出現順についているが、ここでは無視してよい。[R]（アーキテクチャへの要件に関する研究）には 7 文献、[M]（アーキテクチャのメタモデルに関する研究）には 23 文献、[A]（アーキテクチャの構築・分析に関する研究）には 11 文献、[S]（アーキテクチャとシステム間の同期に関する研究）には 12 文献が分類され、[C]（アーキテクチャを用いた利害関係者とのコミュニケーションに関する研究）に分類された文献はなかった。

全体的な研究動向

[C]の研究成果がみられないことは、インタビューにおいても多くの研究者からも賛同を得られた。挙げられた理由として、この分野では、研究の成果を測ることが難しく、評価がしにくいという面が大きいようである。その一方で、この分野の研究の重要性は誰しもが認めるところである。そもそも、ソフトウェア・アーキテクチャの概念が導入された主要な動機の 1 つは、ソフトウェア・アーキテクチャをシステムの利害関係者とのコミュニケーションに用いて、要件の調整を図ることにある。評価のための枠組みの確立などには、経済学、人間学などの社会科学的な要素が必要とされるという意見もインタビューにおいていくつか出てきている。要求工学における取り組みに見られるような、社会科学的な要素を取り入れた研究が、今後この分野の発展には欠かせないようである。

アーキテクチャへの要件 [R]

この分野では、要件の中でも特に、非機能要件が注目されているようである。アーキテクチャでどのように捉えるかを扱った研究として、Klein らによる Attribute-Based Architectural Style (ABAS) と Grundy による、Aspect-Oriented Component Requirements Engineering がある。また、Cheng らは、新たな非機能要件として、動的に変わり行く環境に対応できるシステムに着目した。

アーキテクチャのメタモデル [M]

ソフトウェア・アーキテクチャ研究の分野で、アーキテクチャのメタモデルを決める代表は、アーキテクチャ記述言語 (Architecture Description Language, ADL) であるが、その統合の動きがみられた。過去に、非機能要求ごと、ドメインごとなどに、様々な ADL が提案されてきたが、結局のところ、決定版は難しいようである。非機能要求やドメインは次々に新しいも

のが生まれてくるためだ。Dashofy らは、拡張可能な ADL やそれに対応したツールのフレームワークを提供し、新たな非機能要求やドメインの出現に追随していけるような仕組みを提案している。

一方、ADL に現れる重要な言語要素として、コネクタが着目されて研究されている。コネクタは、システムのコンポーネントをつなぐ要素であり、その能力や制約が ADL の記述力や使いやすさを決めるといってよい。Mehta らは様々なコネクタの分類法を提案している。また、Spitznagel らは、様々なコネクタを設計する上で、既存のコネクタを再利用して拡張できるようなしくみを提案している。

アーキテクチャの構築・解析 [A]

従来アーキテクチャを検証する上で着目されていたのはその構造であったが、そのふるまい（または意味）に踏み込んだ検証に関する研究が Magee らや Uchitel らによりなされている。一方、解析の結果として得られるものの抽象度を、システムの利害関係者レベルに近づけようとする動きも盛んである。Kazman らの Architecture Tradeoff Analysis Method (ATAM) やその発展系ともいえる Cost Benefit Analysis Method (CBAM) はそのための方法論であり、また、Uchitel は具体的に解析結果をシナリオレベルまで反映させるような技術を提案している。

アーキテクチャの構築支援としては、アーキテクチャは頻繁に変わりゆくものとしてとらえた Medvidovic らや Wermelinger らの研究がある。また、Fradet らは、複数の人々により安全にアーキテクチャの変更ができるような環境のための形式的なアプローチを提案している。

アーキテクチャとシステム間の同期 [S]

アーキテクチャとシステム間のギャップを埋めようとする研究が見受けられた。Batory らの提案する AHEAD モデルとそのツールは、アーキテクチャレベルで反映された非機能要件への対応を実装レベルまで対応させるために、Feature 指向や Aspect 指向の技術を用いようというものだ。一方で、Aldrich らは、プログラミング言語の言語要素として、アーキテクチャ的な要素を取り入れることにより、このギャップを埋めようとしている。

アーキテクチャからシステムを導出する流れがある一方、現実的には、アーキテクチャを無視したシステムの変更が起こりうる。Sarma らの提案する環境では、様々な作業者によるシステムの変更が他に及ぼす影響が明示されるような支援を試みている。Zan ら、Keller ら、Yan らは、様々なパターンを検出することによって、システムからアーキテクチャ上の要素を復元する技術を提案している。

4. その他の調査結果

役割による分類に基づく調査結果については既に述べた。ここでは、その他の調査結果の概要として、[M]（アーキテクチャのメタモデルに関する研究）、[A]（アーキテクチャの構築・分析に関する研究）、[S]（アーキテクチャとシステム間の同期に関する研究）の分野で目立った知見を紹介する。

文献調査から直接は分からなかったことであるが、インタビューを通し、[R][C]の研究コミュニティと、[M][A][C]の研究コミュニティは隔離されている傾向があることがわかった。[R][C]の研究コミュニティは、システムの利害関係者からの要件の特定や調整の目的にソフトウェア・アーキテクチャを用いる立場であり、[M][A][C]の研究コミュニティは、システムを導出するための抽象成果物としてソフトウェア・アーキテクチャを捕らえる、いわゆる MDA（Model-Driven Architecture）的な立場である。これらは独立した研究分野であると考えることもできるが、その連携により独立には解決できない問題を解決していくような研究機会があるかもしれない。

その他、プラクティスに関して得られた知見には以下のようなものがあった。

- ソフトウェアプロダクトラインへのソフトウェア・アーキテクチャの適用が盛んになされている。
- UML (Unified Modeling Language) を用いたソフトウェア・アーキテクチャの検証や拡張の提案など、UML との融合が図られつつある。

また、アプリケーション領域に関して得られた知見には以下のようなものがあった。

- 組み込みシステム領域に特化した研究は枯れつつあるようだ。
- 分散領域が盛んに研究され、成果が出されている。
- 非集中化システムやユビキタス領域での研究の成果が出始めている。
- ビジネス・プロセス領域へのアプローチはまだまだこれからであるようだ。

5. まとめに代えて

本発表では、IPA/SEC から公表される予定の報告を部分的に抜粋して紹介したに過ぎない。詳細は、その報告をお待ちいただきたい。本発表から、ソフトウェア・アーキテクチャの研究やSECの活動に興味を持っていただければ幸いである。

参考文献

[1] M. Tatsubori, H. Maruyama, M. Kobayashi, D. Yellin, H. Yoshida, and N. Kawai, A Survey Report on Research Trends in Software Architecture, Software Engineering Center Report, IPA/SEC, 2005. (To appear)