Vol. 36 No. 1　　情　報　処　理　　Jan. 1995

解　説

アジア・太平洋におけるソフトウェア技術

# 3. オーストラリアにおけるソフトウェア工学†

Karl Reed††
(翻訳編集：井上　克郎†††)

本稿では，オーストラリアのソフトウェア工学の概要について述べる．まず，その歴史について述べ，今後の方向についても述べる．また，実際に製品を作っている産業界とソフトウェア工学の研究コミュニティとの関係も示す．さらに他では見られないような独自の研究と実践についても述べ，企業における研究及び大学の研究についても示す．

## 1. まえがき

オーストラリアのソフトウェア工学の分野は，著者が1988年に文献[28]で述べたレベル以上の発展を遂げている．多くの大学では，チーム単位のプロジェクトとして，また授業としてソフトウェア工学が入ってきている．さらに，いくつかの大学の課程は，技術者協会（Institution of Engineers）によって認定されたものになっており，その多くは，オブジェクト指向技術に影響されている．

オーストラリアのソフトウェア産業は世界的にみても大きなものであり，約17億ドルの売上がある[14]．これらの産業は，1960年代初期からソフトウェアパッケージやツールを生産しているが，それらは設計やコードを再利用して作られる場合が多く，ソフトウェア工学の考えを実践したものと言えよう．かつては，太平洋諸国の中では非常に進んでいたこれらの産業も，近年，情報産業を育成している周辺諸国の挑戦を受けるようになってきており，ようやく政府も競争力の強化に動き始めた．

オーストラリアでは，形式的手法からCASE，ソフトウェアの品質，ソフトウェア開発に必要なリソースの予測など，幅広いソフトウェア工学の研究が行われている．また，実践的な研究がこの分野の発展に必要不可欠なものと重要視されており，事実，拡大している．ソフトウェア工学を実践する人も研究する人も，他の地域の人と同じように何が重要かを認識していたが，国をあげてそれらの問題に取り組むことはできなかった．

## 2. 歴史の概要

オーストラリアの最初のソフトウェア工学の活動が何であるかよく分からないが，少なくとも，1960年代の終わりにはモジュール化設計法の概念が認識され，それに基づいてソフトウェアパッケージが作られるようになって，ソフトウェア産業が発展していった．移植が容易なようなプログラムの設計法[22]，自動フローチャート作成[22]，フロー構造からプログラムの生成とテストカバレッジ[13]などの考えは1970年代の初頭にはよく理解されていた．SNOBOLなどの高級言語をプロトタイピングに使うことはContol Data Australiaというコンサルト会社によって1970年代中期には行われていた．

1978年の第8回オーストラリアコンピュータ会議（Australian Computer Conference）では，ソフトウェア工学が主要なテーマになった．そこでは，アプリケーションの生成系（4GL）や決定表の処理系など15本のソフトウェア工学関連の論文が発表された．また，HalsteadのSoftware Scienceメトリクスの批判[25),34)]などをはじめとして，しだいにオーストラリアコンピュータ誌（Australian Computer Journal ACJ）*にソフトウェア工学の論文が現れるようになった．特に，

† Software Engineering in Australia by Karl REED (Amadahl Australian Intelligent Tools Program, Department of Computer Science and Computer Engineering, La Trobe University).
†† La Trobe大学コンピュータ科学・工学科
††† 大阪大学基礎工学部情報工学科

* Australian Computer Society（ACS）から約14000部発行されている．

プロセスの標準化に関する論文が現れてきており，形式化されたプログラミング，部品化を考えた開発，分類の支援，関数手続きとインタフェースの標準化などがすでに認識されている[31]．

1976年から1988年まで行われたMonadsプロジェクトではソフトウェア工学がハードウェアの研究の領域まで達した[30]．ここでは情報隠蔽やモジュール化を支援するハードウェアの実現を行っている．また，システムアーキテクチャの抽出法[21]やソフトウェアと既存のアーキテクチャとの対比などの研究も行われた．1982年の5月号のACJはソフトウェア工学の特集であり，また，1986年には第1回オーストラリアソフトウェア工学会議(Australian Software Engineering Conference ASWEC) が開かれ，Dave Parnasが基調講演を行った．

1980年代初期，多くの工学系の大学ではソフトウェア工学が学部の授業として導入されたり研究プログラムの対象となったりし，すそ野を広げる役割を果たした．初期の研究グループとしては，University of Queensland(UQ) のGordon RoseとAndrew ListerらのZを用いた大規模形式的仕様の生成手法とその支援ツールの研究[37]があげられる．移植心臓機能回復装置 (Implantable Cardiac Defibrillator) の形式的仕様の記述など[27]，Zの商業的な応用も連邦科学研究機構(Commonwealth Scientific Research Organization CSIRO) の情報工学部門 (Division Information Technology DIT) によって行われた．またそのころUniversity of New South WalesのRoss Jeffery, Mike Lawrence, Graham Lowらがソフトウェアの生産性の研究を行っている．

その当時，通信の分野でも，Numerical Petri Net(NPN) やプロトコルなどに関して多くの研究が行われた．PROTEANと呼ばれるNPNのためのツールがTelecom Australiaの研究所で開発され，La Trobe UniversityのTharam Dillonによってさらに Procedural Petri Netへ拡張された[24]．Royal Melbourne Institute of Technology(RMIT) のLindsay Jacksonらのグループは MELBA というCCITT仕様のための環境及びSDLという記述言語を開発した[6]．

1980年代の終わりにはソフトウェア工学は確固とした研究分野となり，オーストラリアソフトウェア工学会議は毎年開かれるようになった．Computer Sciences Australia(CSA) やTelecom Australiaなどの企業も研究に参加するようになり，メトリックスやオブジェクト指向技術が広まった．国防科学技術機構 (Defence Science and Technology Organization DSTO) では，Stefan Landherrが予算規模約110万ドルで12人の研究員からなるソフトウェア工学グループ (SEG) を作った．これはたぶん産業界の中では一番大きなソフトウェア工学の集団であろう．

ここ数年，連邦政府の支援のもとでUniversity of QueenslandにソフトウェアEE術教育センタ (Key Centres for Software Technology and Teaching) やソフトウェア検証研究センタ (Software Verification Research Centre) などの大きな研究グループが作られた．また，CSIROとMacQuarie Universityの高度システム工学共同研究センタ (Joint Research Centre for Advanced Systems Engineering JRCASE) ではシステム工学と伝統的なソフトウェア工学の境界に着目して研究を行っている．1989年，Amdahl Australiaは，ハイパテキストに基づいたCASE環境の研究のための資金をLa Trobe Universityに提供した．1994年，Melbourne UniversityとRoyal Melbourne Institute of Tcchnologyによって共同運営されいている協調情報技術研究所 (Collaborative Information Technology Research Institute CITRI) は，電話装置会社のL. M. Ericssonからインテリジェントネットワークアーキテクチャの設計について研究を行う資金を得た．多くの政府機関がソフトウェアの供給者にAS 35633 (オーストラリアソフトウェア品質管理標準[2]) の証明を要求するなど，ソフトウェアの品質保証も大きな課題になってきた．また，Griffith Universityではオーストラリアソフトウェア品質研究所 (Australian Software Quality Institute SQI) を設立し，ソフトウェアの品質の問題について研究や普及活動を行っている．

オーストラリアコンピュータ協会 (ACS) の下にソフトウェア工学に関するいろいろな分野の分科会が作られている．たとえば，Paul Farrowが率いるACSの高信頼システム技術委員会(National Technical Committee on Safety Critical Systems SCS TC) は，州政府やACSのソ

フトウェア品質分科会（Software Quality Association SQA），いろいろな大学などと協調して，ソフトウェアの品質に関して各種の教育を行っている．その他，オーストラリアソフトウェアメトリックス協会（Australian Software Metrics Association ASMA)(4. 参照)，オーストラリアソフトウェア保守協会（Australian Software Maintenance Association)，最近結成されたオーストラリアソフトウェア工学協会（Australian Software Engineering Association）などが全国規模で活動を行っている．

米国やヨーロッパと同様に学界とソフトウェアの産業界との交流は多くはなく，形だけの支援が電気通信会社などから行われている．

## 3. オーストラリアのソフトウェア産業，ソフトウェア工学の適用

現在，ソフトウェア産業全体では約17億ドルの売り上げがあり[14]，輸出は5億ドルを越えている．これらは1960年代前期より主にパッケージソフトウェアを生産してきており，再利用中心の文化ができあがっている．また，あまりはっきりした記録は残されていないが，画期的なものも作られている．たとえば，高階な商用言語（4 GL）や画面ベースのアプリケーションソフトの生成系(Screen Based Application Generators SBAG)，そのほかミニコンピュータ用コンパイラやオペレーティングシステムなどが1970年代中期に作られている．

オブジェクト指向技術を得意とするコンサルタントやソフトウェアハウスがたくさん現れてきており，オーストラリアで作られた商用オブジェクト指向言語 Ochre などもある．毎年，オブジェクト指向技術のための非学術的で商業的な会議が開催されており，オブジェクト指向法が広く受け入れられていることが分かる．

残念ながら産業界は学界との交流があまりない．各企業は，自分たちの方法がまねされるのではないかと，あまり公表したがらないため，大学などでは研究情報が得られない．したがって大学が企業のもつ情報を用いて研究しその研究成果を産業界に技術移転して国全体の経済に貢献する，ということができない．

次の章では研究活動についてより詳しく述べる．今述べたような制限はあるが，産業界と学界との交流はいくつかの分野では行われている．特に生産性の研究は産業界の支援によって行われており，また，ソフトウェアの品質の分野でも協調が行われている．紙面の関係でその一部だけを紹介する．

## 4. 研究分野—主なプロジェクトとグループ

### 4.1 推定および実験研究

実験研究は推定モデルを作るということで，Ross Jeffery と Mike Lawrence によって，1970年代の終わりに始められ[17]，University of New South Wales にソフトウェア工学研究グループ(Software Engineering Research Group SERG)が設立された．このグループは産業界との交流をもつものの数少ない例である．

SERG の成果はめざましく，たとえば Jeffery, Lawrence, Low らはあるアプリケーションの領域では，プロジェクトのサイズが大きくなってもプログラムの生産性があがる場合があることを報告している[18]．さらに，ファンクションポイントを用いた計測，推定方法の改良もこのグループが行っている[20]．SERG はオーストラリアソフトウェアメトリックス協会の設立にも重要な役割をし，全国的なファンクションポイントに基づく推定法の研究を広めた．最近，Jeffery らはバックエンド CASE ツールの有効性も報告している[19]．

ソフトウェア開発の実験研究も増えてきている．たとえば，University of Newcastle ではオブジェクト指向設計に基づいた再利用の有効性の検証を行っている．この研究は，CASE ツールの有効性，産業界での再利用方法の解析に基づいている．

ソフトウェア工学の中でも，システム性能をモデル化する技術は遅れている分野であろう．University of South Australia の Jim Warren は注釈付きデータフロー図を用いた性能予測モデルを開発している[23]．

### 4.2 メトリックス

ソフトウェアメトリックスの分野の研究はあちこちで行われている．La Trobe University の Tharam Dillon らのグループは，システムの信頼性を推定するメトリックスを提案している．これは情報理論に基づいた複合メトリックスで，モジ

ュールとメッセージ渡しの複雑度を測る．

CSIROとMacquarie Universityの高度システム技術共同研究センタ（CSIRO-Macqurie University Joint Research Centre for Advanced Systems Engineering JRCASE）はM3Pと呼ぶ方法でソフトウェアメトリックスを商業的な面から研究している．また，SQUATerというソフトウェアの品質，計測のための統合ツールを作成している[26]．

### 4.3 ツール

このJRCASEのグループは，図の形式化，抽象的なグラフエディタなどのツールに関する研究も行っている．プロセスの生産物の視覚化，操作法の基礎的な形式化に興味をもっており，CASEツールをこれを用いて設計することを考えている．

University of South AustraliaのグループはCASEや統合ソフトウェア工学環境（Integrated Software Engineering Environments ISEE）の研究を行っている．ここではユーザとISEEとのやりとりの抽象化の度合いを研究している．Flinders UniversityのJenny HarveyとChris Marlinはプロセスモデルに基づいて図を用いてプロセスの定義ができるようなソフトウェア開発環境の研究を行っている．

University of Queenslandではソフトウェア開発環境について数多くの研究が行われており，すべてを述べることはできない．形式的手法を支援する研究も行われており[37),38]，これについては他の節で述べる．前に述べたように，ここでは既存の形式的手法のためのツールの統合も試みている．

Chris Marlinをはじめとする Flinders Universityのチームでは，一つの標準表現で表された生産物を，複数の視点で表現するためのツールの研究を行っている[15]．開発者に分配したタスクをモニタするツールなどプロセスに基づいた開発環境や，半自動文書作成に基づいて，差分的な意味解析を行い変更部分の追跡を行う再利用支援ツールなども研究されている．

Amdahl AustraliaはLa Trobe Universityでのハイパテキストと CASEとの統合に基づいた知的ツールの研究を支援している[7]．この研究の特徴としては，グラフ部品の定義法に基づいてグラフィックエディタをカスタマイズすることができることである．グラフィックエディタの中のどの部品もハイパテキストのボタンとして用いることができる．また，設計理論の記録ツール，自動プロジェクト追跡ツール（ツールの利用度や文書の状態から進捗を判断する），NLP要求仕様の獲得のためのツールなども研究されている．

1980年代の中期から終わりにかけて，Royal Melbourne Institute of Technologyでは，CCITTのSDLやCHILLの開発を支援する環境のMELBAプロジェクトが行われた[6]．現在，そのグループではSDLを用いた設計理論の記録ツールSOFTEAMの研究を行っている[16]．

### 4.4 ソフトウェアプロセス

University of Queenslandのソフトウェア技術センタが開発している高信頼性システムのアーキテクチャの枠組みTARDISでは，アーキテクチャ以外のたとえばシステムのタイミングなどの制約が開発プロセス全体を通じて記述できる．TARDISはAndrew Listerが1992年の第14回ソフトウェア工学国際会議（International Conference on Software Engineering）で行った基調講演のテーマにもなった[4]．

University of QueenslandのKBSEグループは，実際の開発プロセスと予想されるプロセスとの関係を調査し，それに基づいて，分散チームワークのための方法論とそのツールを開発した．これにはUniversity of Dortmundが開発したMerlinシステムを用いており，同期の取れた協調作業について研究を行っている[39]．

ソフトウェアプロセス改良ネットワーク（Software Process Improvement Network SPIN）は，共同技術研究所（CITRI）のLin Zucconiによって設立され，オーストラリアコンピュータ協会（ACS）の技術委員会の一つとなっている．ACSのソフトウェア品質分科会（SQA），オーストラリアソフトウェア品質研究所（Australian Software Quality Institute ASQI[2)]），そして連邦政府は，SPICEプログラムに多くのソフトウェア開発会社が参加するように支援している．高度システム技術共同研究センタ（JRCASE）のツールはプロセスの評価や改善など，難しい領域を対象としている．

### 4.5 オブジェクト指向

オブジェクト指向法については数多くの研究が

ある．

La Trobe UniversityのTharam DillonとElizabeth Changはオブジェクト指向の考えを，既存の手続き的言語のソフトウェアや，データベース，知識ベースをもとにしたアプリケーションの開発に使えるように概念モデルの拡張を行った[9]．彼らはそれを設計法，ユーザインタフェース設計法，データ処理システム開発法などに用い[8]，その結果は1994年7月パリで開かれたAI'94会議の基調講演として発表された．

Melbourne Universityのソフトウェアグループはオブジェクト指向法による設計書の解析方法について研究している．COTAR(Center for Object Technology Applications and Research)と呼ばれるオブジェクト指向法の研究センタがUniveristy of Technology SydneyにBrian Henderson-Sellarsによって設立されている．

Monash UniversityのClaytonキャンパスでは，オブジェクト指向法のそれ自身から関連分野まで幅広く研究を行っている．また，Queensland Univeristy of TechnologyではWirthが開発した言語Oberon-2[39]のコンパイラを作成し，多くの大学で使われている．

### 4.6 形式的手法

John Staplesが率いるUniversity of Queenslandのソフトウェア検証研究センタ（Software Verification Research Centre SVRC)[33]では，ポインタを含む詳細化法の拡張[3]，Zのオブジェクト指向への拡張（OBJECT Z[11]）など，この分野で多くの業績をあげている．Zの統合的な開発ができるようモジュール化を許すような拡張（仕様記述言語SL 1）もCOGNITOプロジェクトの中で行われている．また，SL 1からAdaへの自動変換の研究も行われている．

2.で述べたように，連邦科学研究機構(CSIRO)では形式的手法を実際のシステムに適用することも試みられている[27]．

### 4.7 ソフトウェアの品質

ソフトウェアの品質に関して多くの活動がみられるようになってきた．多くの研究活動がこれに関連し，また，ソフトウェア品質協会の認証コースのような種々の講習が行われるようになった．Queensland州のGeoff Dromeyが率いるオーストラリアソフトウェア品質研究所（Australian Software Quality Institute ASQI）は，この分野で多くの研究をしている．たとえば，ASQIの最近のレポートの中で，ソフトウェアの品質の戦略的な問題点について述べている[10]．また，ASQIではPASS-Cというシステムを開発している．このシステムは，品質の低さと相関があると思われるプログラム上の欠点を解析する[1]．

### 4.8 テスト

Queensland University of Technologyでは性能低下を最小に抑えることができるデバッグ手法を開発した．この方法では，中間コードのコンパイル，実行を並行して行う．Melbourne UniversityのT. Y. Chenはブラックボックステストのためのテストベッドを開発しつつある．他に，データフロー解析に基づいた自動テスト手法の研究もある．

## 5. ソフトウェア工学教育

ソフトウェア工学の教育がいつ始まったか定かではない．New South Wales Institute of TechnologyのJohn Leaney(現，University of Technology Sydney）は，1975年にその授業を開始した．Royal Melbourne Institute of Technologyをはじめとして他の大学でも1980年までには同様の授業を開始した．

いくつかのコンピュータシステム工学の学士号及び一つのソフトウェア工学の学士号の課程は，工学協会によって認定されいている（たとえばLa Trobe UniversityやUniversity of Melbourne)．そしてそれらは多くのソフトウェア工学専攻の学部学生を擁している[29]．多くのコンピュータサイエンスの学科では三つ以上，そして，いくつかの学科では非常にたくさんのソフトウェア工学の授業が用意されている．すくなくとも一つのコンピュータサイエンスの学科（Swinburne University of Technology[12]）では，その課程の中心がソフトウェア工学になっており，また，他の大学でも同様のことを検討しているようである．また，多くの大学ではソフトウェア工学の考えを最初の学年で教えている．

ときどき，はじめに教える言語は何が良いかについて激しい議論が起こる．このことはいくつかの大学ではソフトウェア工学の教育とも絡んでいる．関数的な考えをプログラム開発に用いるのが

良いと考えるいくつかの学科ではMirandaを用いている．ソフトウェア工学の考えを教える前にちゃんとした基礎を教えたほうが良いと考える学科ではPascalを用いている．最初からソフトウェア工学の考えを浸透させていったほうが良いと考える学科ではAdaを使い始めている．また初年度にオブジェクト指向法を教えるのが良いというところもあり，C++がそれにふさわしいか議論になることがある．しかし言語が本質的に複雑なうえ，基本的なプログラミングのスタイルがないので見直そうとする学科が出てきている．

このように，ソフトウェア工学の教育については世界的にみても非常に良いレベルにある．1980年代の初頭の斬新な試みとして始まり，今や教育の中の大きな地位を占めるまでになった．また，ソフトウェア工学に基づいた課程を育成したという点で他の主要な先進国以上に成功したと言えよう．

## 6. 将　　来

オーストラリアのソフトウェア工学の研究の将来について議論するにあたって次の二つのことを考える必要がある．まず，ソフトウェア産業の技術的な基礎を確立するために国の主導が不可欠である．次に，今後の主流となる流れを認識することである．前者については，政府がこのことを認識し，手を打つことが望まれている．現在，少なくとも一つはそういう例がある．

流れの認識は容易ではない．オーストラリアの研究者は個人主義で，この論文で示すようにかなりいろいろなことをやっている．一方，先端領域では優秀な研究グループが大規模な研究をやっている．オーストラリアでは研究資金はピアレビュや学界の重鎮によって分配が決められるが，これを利用してそのようなグループを組織すれば，一個人で研究するよりも，学界全体にとってメリットは大きい．研究の歴史や現状をみると，開発方法論，開発現状の分析，開発全体を支援するツールなどの研究への潮流が窺えよう．これらが将来のオーストラリアのソフトウェア工学の研究全体の枠組みとして発展することが期待される．また，数年後にはソフトウェア工学を学んだ多くの学生が卒業し，研究者となって，より多くの研究がされるようになろう．

ここ数年に限ってみれば，独自の方法論に基づいたツール支援の研究が増えるであろう．また，再利用や（プロセスなど）規範的な方法論の需要が高まることが予想される．後者は，ここ12カ月のうちに，ACS SPIN, SPICE計画，ASMAサーベイなどの発展にともなって飛躍的に増えることになるであろう．また数年以内には，政府がソフトウェア工学は国内のソフトウェア産業の育成に貢献していることに認識するであろう．そしてオーストラリア国立ソフトウェア工学研究所（Australian National Software Engineering Institute）の設立につながることを期待する．この研究所ではとりあえず，ソフトウェア産業の技術的な支えとなろう．最近，首都キャンベラにある連邦科学研究機構（CSIRO）はソフトウェア工学関係の活動を拡大することを決定したが，これは，将来を占う材料となろう．

最後にイギリスの偉大な詩人であり科学小説の作家でもあったH. G. Wellsの詩を引用する．歴史というものは気まぐれなもので，将来を予想しようとする人は恥をかくであろう．

**謝辞**　この論文を書くにあたって題材を提供したすべての人に感謝する．グループや個人それぞれに謝意を表すことはできない．この論文がオーストラリアのソフトウェア工学に関係する活動全体をうまく表していることを期待する．また，何度も草稿を読んでもらった閲読者と妻に感謝する．

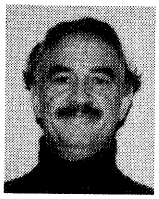
## 参 考 文 献

1) Australian Software Quality Institute PASS-C User Manual (version 1.1), Technical Report, SQI-93-04 School of Computing and Information Technology, Griffith University.
2) Australian Software Quality Institute SPICE-Project Overview, School of Computing and Information Technology, Griffith University (May 1994).
3) Bancroft, P : Pointers in Refinement Calculus (A Case Study), Australian Software Engineering Conference 1993, pp. 11-19.
4) Burn, S., Lister, A. M. and McDermid, J. A. : TARDIS : An Architectural Framework for Timely and Reliable Distributed Information Systems, Australian Software Engineering Conference 1991, pp. 1-16.
5) Cheng, A. S. K., Han, J., Welsh, J. and Wood, A. : Incorporating Constructive Tools into a Generic Language-Based Editor, Australian

Software Engineering Conference 1993, pp. 197-203.
6) Cheng, K. E., Pascoe, R. S. V. P., Coong, T. S. and Jackson, L. N. : An Integrated Approach to Automated Telecommunications Software Development : Design Experience from the Melba Project, Proc. First Pan Pacific Computer Conference, pp. 360-378 (Sep. 1985).
7) Cybulski, J. and Reed, K. : A Hypertext Based Software Engineering Environment, IEEE Software, Vol. 9, No. 2, pp. 62-68 (Mar. 1992).
8) Dillon, T. S. and Chang, E. : Object-Oriented Technologies and Technologies, Proc. Inaugural Conference on Object Oriented Representations at AI94, Paris, pp. 11-30 (June 1994).
9) Dillon, T. S. and Tan, P. : Object-Oriented Conceptual Modeling, Prentice-Hall, 1993.
10) Dromey, R. G. and Green, I. : Strategic Directions for Software Quality in Australia, Australian Software Quality Institute and Coopers and Lybrand, ASQI-94-01, Griffith University.
11) Duke, R., King, G., Rose, G. and Smith, G. : The Object-Z Specification Language Technical Report 91-1, University of Queensland, Department of Computer Science, 1991.
12) Grant, D. D. and Smith, R. : Undergraduate Software Engineering—An Innovative Degree at Swinbume, Australian Software Engineering Conference 1991, pp. 149-162.
13) Herman, P. M. : A Dataflow Analysis Approach to Program Testing, Australian Computer Journal, Vol. 8, No. 3, pp. 92-96 (Nov. 1976).
14) IDC "Directions, 1993", International Data Corporation, 1993.
15) Jacobs, D. A. and Marlin, C. D. : Software Process Representation to Support Multiple Views of the Enacted Process, Proc. First Asian-Pacific Software Engineering Conference (Dec. 1994), to appear.
16) Jayaputura, G. T. and Cheng, K. E. : SoftEMA : A Design History and Justification Maintenance Tool, Australian Software Engineering Conference 1993, pp. 185-196.
17) Jeffery, D. R. and Lawrence, M. J. : An Inter-Organizational Comparison of Programming Productivity—An Empirical Look at Intuition, Proc. 4th International Conference on Software Engineering, Munich (Sep. 1979).
18) Jeffery, D. R. and Lawrence, M. J. : Commercial Programmer Productivity—An Empirical Look at Intuition, Australian Computer Journal, Vol. 15 (Feb. 1983).
19) Jeffery, D. R. and Low, G. : Software Development and Back-end Case Tools, Information and Software Technology, Butterworth-Heinemann, Oxford, U.K. Vol. 33, No. 9 (Nov. 1991).
20) Jeffery, D. R. and Low, G. : A Comparison of Function Point Counting Techniques, IEEE Trans. on Software Engineering (May 1993).
21) Kaunitz, J. : A Descriptive Model for the Architectural Design of Information Systems, Australian Computer Journal, Vol. 14, No. 3, pp. 91-98 (Aug. 1982).
22) King-Smith, P. : Packages, Availability and the Specific Characteristics of Commercial Packages, Proc. Soft-ware, Soft-why, Soft-how, Australian Computer Society Software Industry Committee (July 1976).
23) Lew, K. S., Dillon, T. S. and Forward, K. E. : Software Complexity and Its Impact on Software Reliability, IEEE Trans. on Software Engineering, Vol. 14, No. 11, pp. 1645-1655 (Nov. 1988).
24) Li, P., Dillon, T. S. and von Thum, M. A. : Investigation of Developing Protocol Implementations from a Petri Net Description, Australian Software Confernce, 1988.
25) Lister, A. M. : Software Science — the Emperors New Clothes?, Australian Computer Journal, Vol. 10, No. 2, pp. 66-70 (May 1982).
26) Offen, R. J. and Aronov, V. A. : Model-Based Approach to Software Measurement, Proc. First Australian Conference on Software Metrics, pp. 118-127 (Nov. 1983).
27) Parle, A. : Application of Formal Methods to a Real Time Software Development, Proc. Australian Software Engineering Conference, pp. 11-20 (May 1987).
28) Reed, K. : Software Engineering in Australia —A Report to the Panel Session Software Engineering Development in the Asia Pacific, Proc. 10th International Conference on Software Engineering, Singapore, supl. pp. 14-19, (1988).
29) Reed, K. and Dillon, T. S. : An Undergraduate Software Engineering Major Embedded in a Computer Systems Engineering Degree, Proc. Software Engineering Education SEI Conference 1990 Pittsburgh Penn. USA, L. E. Deimel (ed) LNCS 423, pp. 49-66 (Apr. 1990).
30) Rosenberg, J. and Abramson, D. : The Monads Architecture : Motivation and Implementation, Proc. First Pan Pacific Computer Conference, pp. 410-423 (Sep, 1985).
31) Smiley, D. : H. Process Standardization in Information Systems Engineering, Australian Computer Journal, Vol. 11, No. 2, pp. 42-47 (May 1979).
32) Standards Australia, Australian Standards 3563.1 & 3563.2, Software Quality Management System Part 1 : Requirements, Part 2 : Implementation Guide.
33) SVRC 1993 Annual Report, Software Verification Research Centre Department of Computer Science, University of Queensland.

34) van de Kniff, D. J. J. : Software Physics and Program Analysis, Australian Computer Journal, Vol. 10, No. 3, pp. 82-86 (Aug. 1978).
35) Verschoor, R. and Low, G. : Software Reusability in Australia, Proc. Australian Software Engineering Conference, Software Engineering 1993, pp. 141-150.
36) Watkins, R. P. : The Design of Flog, an Automatic Flowchart Generator, Australian Computer Journal, Vol. 6, No. 7, pp. 124-128 (Nov. 1974).
37) Welsh, J. and Yang, Y. : Tool Integration Techniques, Australian Software Engineering Conference 1991, pp. 405-418.
38) Welsh, J., Spanevello, D. and Yang, Y. : A Generic Tool for Document Review by Distributed Team Work, Proc, 17th Australian Computer Science Conference, 1994.
39) Wirth, N. : The Programming Language Oberon, Software Practice and Experience, Vol. 18, No. 7, pp. 671-690.

(平成6年12月5日受付)



**Karl Reed**

Royal Melbourne Institute of Technology 卒業. Monash 大学大学院修士. 現在, La Trobe 大学コンピュータ科学・工学科準教授. オーストラリアコンピュータソサイアティ(ACS)フェロー, 終身名誉会員. 1986年米国メリーランド大学客員として TAME プロジェクトに参画. 豪州におけるソフトウェア工学教育の草分けである. ICSE プログラム委員, CASE ワークショッププログラム委員長, ACS のコンピュータシステムとソフトウェア工学に関する技術委員会委員長, Australian Computer World 誌コンサルタントエディタ.