ソフトウェア工学 113－10
（1997．3．7）

「ワークショップ・イン・松山」報告

大西　淳[1)]，鯵坂恒夫[2)]，上原三八[3)]，津田道夫[4)]，
蓬莱尚幸[3)]，山田宏之[5)]

[1)] 立命館大学，[2)] 京都大学，[3)] 富士通研究所，
[4)] 日立製作所，[5)] 愛媛大学

1997年1月23日と24日に渡って愛媛県松山市で開催されたソフトウェア工学研究会主催の『ウィンターワークショップ・イン・松山』について報告する．本ワークショップでは「要求工学」，「新工法」，「保守(と2000年問題)」の3つのテーマを設け，テーマごとに約15名のサブグループを形成し，グループ単位で活発な討論を進め，成果を得た．本稿ではワークショップの企画，運営，それぞれのサブグループの討論内容，およびワークショップでの成果について紹介する．

Report of the Winter Workshop in Matsuyama

Atsushi OHNISHI[1)], Tsuneo AJISAKA[2)], Sanya UEHARA[3)],
Michio Tsuda[4)], Hisayuki HORAI[3)], Hiroyuki YAMADA[5)]

[1)]Ritsumeikan University, [2)]Kyoto University, [3)]Fujitsu Laboratories Ltd., [4)]Hitachi Ltd., [5)]Ehime University

"Winter Workshop in Matsuyama" was held in Dougo, Matsuyama, Ehime, from January 23 to 24. In this article, the authors briefly summarize the workshop. Three themes were provided in this workshop. These are "Requirements Engineering," "the New Deal in software engineering," and "the Year 2000 problem and Software Maintenance." Attendants were divided into three groups in accordance with the three themes and they made hot discussions. The authors describe the workshop including plans, discussions and the workshop results.

1 はじめに

ソフトウェア工学研究会では，93年度に「サマーワークショップ・イン・大雪」を，94年度に「ウインターワークショップ・イン・沖縄」を，95年度には「サマーワークショップ・イン・立山」を開催してきたが，これまでのワークショップでは研究発表が中心であった．そこで現時点でソフトウェア工学が直面しているさまざまな問題点と新しいコンセプトや今後に向けての方策などについて，今後の研究や実践に役立つような活発な議論を展開することを目標として，今回のワークショップではテーマを絞り，テーマごとに密度の高い討論の場を設けることにした．

今回は，今後特に重点的展開の必要性が見込まれるライフサイクルの両端部分，すなわち「要求工学」と「保守」，および古典的ライフサイクルを打破しプラットフォームとコンポーネントの組み立てをベースとする「新工法」の3つのテーマを設け，参加を募ったところ，幸いなことに産学のさまざまな分野から45名の方から参加申し込みがあった．参加者のこれまでの成果・経験と今後への展望期待を元に約1日半ではあるが，活発な討論がなされ，成果が得られた．すでにソフトウェア工学研究会からワークショップ論文集[1]が刊行されているが，これは参加者のポジションペーパーとワークショップで取り上げたテーマの紹介を中心にまとめたものである．本稿ではワークショップでの討論内容と得られた成果を中心にまとめている．

運営に当たっては，各テーマごとに担当実行委員を定め，討論内容や進行は実行委員に任せた．要求工学グループは蓬莱委員をまとめ役に，佐伯元司(東工大)，山田，大西の各委員が，新工法グループは鯵坂委員をまとめ役に，井上克郎(阪大)，野呂昌満(南山大)の各委員が，保守グループは上原委員をまとめ役に，玉井哲雄(東大)，津田の各委員が担当し，ローカルアレンジメントを山田委員が，実行委員長を大西が担当した．

また，今回は参加者全員に2頁のポジションペーパーの提出をお願いした．原則としてLaTeX形式としたが，青山幹雄研究会幹事からスタイルファイルを戴き，編集作業が大いに捗ると共に，ワークショップ論文集の仕上がりが均一で奇麗なものとなった．一方，非LaTeX利用者には不便を感じた方もおられた．また電子

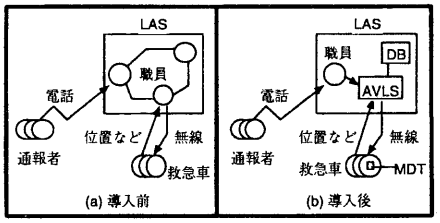

図 1: LAS システム導入前後

メールを実行委員の打合せや参加者との打合せに積極的に利用した。

会場は道後温泉大和屋本館とし、2日間の合宿形式で開催された．ワークショップの全体会を大広間で，テーマごとの部会を3つの小広間で行った．大広間には計算機の画面を映せるプロジェクタが設置してあり，PowerPointを用いて発表が行われた．設備，サービスを含め会場は大変立派で好評であった．

2 要求工学

要求工学における問題点、および、それらの問題点の解決のために寄与するテクノロジーについて活発な議論を行った。その際、London Ambulance Service(LAS)システム開発[2]を事例として用いた。

ワークショップにおける議論は以下の順に行われた: 1. ワークショップ参加者の自己紹介。2. LASシステムの概要把握。3. LASシステム開発を成功させるために必要な要求の抽出。4. 必要な要求を獲得できなかった問題点の究明。5. 問題点を解決するテクノロジーに関する議論。6. 各参加者の研究紹介。7. 「理想的な」LASシステム開発像の構築。

2.1 事例

LASは、市民からの電話による通報を受け、救急車の適切な配車を行うサービスである。LASシステム導入前(図1(a))は，複数の職員による協調的な作業で、各職員の作業にはかなりの知性が必要とされていた。

LASシステム導入後(図1(b))は，通報や救急車や道路などに関する情報を蓄えるデータベースを利用したAuto Vehicle Locating System (AVLS)により自動的に最適配車を行う。市民からの電話による通報を受け

た職員はLASシステムにその通報に関するデータを入力し、救急車からは現在位置などの情報が送られる。AVLSシステムが決定した最適配車は、無線により各救急車に伝えられ、救急車に装備されたMobile Data Terminal (MDT) に表示される。

LASシステム導入後のある日、1. 通報のトラフィックの増加、2. AVLSによる救急車の位置などの情報の正確な把握困難、3. データベースが正確なデータの保持不能、4. 1つの通報に対する複数の配車、5. 最適ではない配車、6. 救急車への大量のメッセージ送付、7. 乗務員によるMDT操作不能、8. システムの処理速度の悪化、などの事象が悪循環し、最終的にはシステム全体がダウンした。

2.2 LASシステムへの要求

LASシステム開発の要求プロセスにおいて獲得すべきであったと思われる要求項目には以下のものが含まれることが判明した。

非機能要求: 1. データベース容量などの性能 (特に、ピーク時性能)。2. システム強度 (robustness)。3. 応答速度。4. 各部の信頼性 (特に、無線通信)。5. コスト、開発期間に関する見積り。6. システム範囲。

機能要求: 1. システムのインテリジェンス (特に、最適配車の定義)。2. 起こり得る事象 (特に、例外事象) の洗い出し。3. ユーザインタフェース。

2.3 LASシステム開発における問題点

LASシステム開発の失敗の原因となったと思われる要求プロセスでの獲得・分析の活動の問題点には、以下のものが含まれると判明した: 1. ピーク時のデータなどの調査・収集・分析・利用が行われていない。2. stakeholder(救急車乗務員・職員など) や専門家 (最適化 (AI) 専門家・ユーザインタフェース専門家・類似システム開発経験者など) が参加していない。3. 障害が発生した日までは正常に動作していたことから、事象の洗い出しはある程度成功していたと思われるが、例外事象の分析が不徹底であった。4. robustnessを高めるためのFail-safeな設計のための基準がない。5. 非機能要求と機能の関連付けが行なわれていない。6. 信頼性に関する検討の不足。7. 精度や誤差 (例えば、移動している救急車の位置の把握の誤差) に関する解析を行っていない。8. 最適配車の定義があいまいである。9. 適切なユーザインタフェースの設計が行われていない。10. 多視点・対立要求・要求間ズレの検出とネゴシエーションが不徹底である。

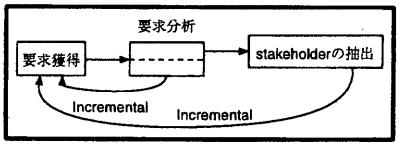

図 2: stakeholder・専門家・要求の獲得

2.4 解決のためのテクノロジー

上記の問題点の中から、「stakeholderや専門家をどのようにみつけるか、見つけた後にどのように情報や要求を獲得するか、獲得した情報や要求に含まれる多視点や対立要求をどのように見つけネゴシエートするか」に焦点を当てて解決のためのテクノロジーに関する議論を行った。

2.4.1 stakeholder・専門家・要求の獲得

我々はstakeholderや専門家 (以下、両者を併せてstakeholderと呼ぶ) の発見および要求の獲得のプロセスの全体的な枠組を構築した (図2)。まず、stakeholderからインタビューなどの手段を用いて要求を獲得する。この枠組では、stakeholderや獲得された要求は不完全なものと考え、漸進的に、要求の獲得およびstakeholderの追加を行う。漸進的な要求の獲得では、stakeholderを固定し、獲得した要求を分析し、その結果をstakeholderへフィードバックし、新たな要求の獲得を行う。反対に、漸進的なstakeholderの追加では、獲得した要求を固定し、獲得した要求を分析し、その結果をもとにstakeholderの抽出を行う。

新しい要求を漸進的に得るための要求分析技法としては、プロトタイピングやシミュレーションなどがあることが分かっている。新しいstakeholderを得るための技法としては、以下のようなものが提案された: 1. 関与する人を全部見つけて、その中から絞り込む。2. 既存システムの入出力を扱う人を対象とする。3. システム範

囲の内側にいる人。すなわち、システムの影響を被る人を対象とする。4. システムを分割し、部分システムの領域ごとの専門家を集める。5. シナリオ分析を行い、関与する人を洗い出す。6. 類似システムやドメインモデルを参照する。7. 熟練者や経験者は重要である。

2.4.2 対立とネゴシエーション

我々は、獲得した要求間に存在する対立の検出とネゴシエーションのプロセスの全体的な枠組を構築した(図3)。我々は、要求間の対立の検出やネゴシエーションに先立ち、知識や立場などの違うstakeholder同士の相互理解も重要なプロセスであると考える。

stakeholderの相互理解のためには、stakeholder間の語彙や表現に統一性を与えることが重要となる。そのための手法として以下のものが提案された: 1. 言葉の対応付けや共通語彙の辞書を作る。2. 共通の「第3の表現」を用いて記述する。第3の表現の候補としては、シナリオやプロトタイプがあげられる。

本ワークショップでは、特に、「第3の表現」の利用についてさらに深い議論を展開し、第3の表現について、以下のような知見を得られた: 1. 第3の表現の利用方法としては、相互理解したい人が全員で協調して記述する方法と個別に記述し結果を比較する方法が考えられる。2. stakeholderが持っている知識をすべて相互理解する必要はなく、第3の表現は相互理解すべき範囲を記述できるだけの能力を持つように設計されていればよい。3. 要求分析者は、各stakeholderと第3の表現を介して要求獲得を行う。stakeholderごとに用いる第3の表現は異なっても構わない。4. 要求分析者は、あるstakeholderから第3の表現をとおして得た事柄を他のstakeholder用の第3の表現に翻訳して伝達することでstakeholder間の相互理解を助けなければならない。5. 要求分析者は、第3の表現をとおしてなるべく多くの事柄をstakeholderから得ようと努力する。これ以上得られないという点に達したところはシステム範囲と密接な関係があると考えられる。

対立する要求の検出において利用可能な技術には、マルチビューポイントに基づく分析手法がある。stakeholder間の違いによる対立の検出のみならず、同じ対象に対する異なるモデル間の対立や異なる記法間の対立の検出も重要である。

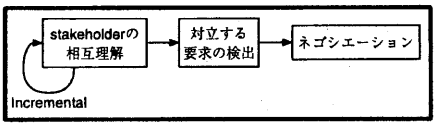

図3: 対立とネゴシエーション

ネゴシエーションのプロセスでは、共通の尺度を設定し、取捨選択を行う。ネゴシエーションで利用可能な技術としては、トレードオフ分析、prioritization手法、要求間の依存関係分析などが重要である。また、後で問題点が現れたときの対処を可能にするために、捨てられた要求とその理由を記録したり、採択された要求の開発プロセス後段における利用に関するトレーサビリティを記録することが重要である。

2.5 理想的なLASシステム開発

上記の枠組に沿ったLASシステムのユーザインタフェース(特に、MDT)に関する理想的な要求獲得について考察した。MDTに関して実際に発生した主な問題点は、以下のとおりである: 1. 大量のデータを受信したときのスクロールあふれ。2. 救急車乗務員が操作を行なわない。3. 救急車乗務員が別の人のIDでloginする。

上記の問題点を残さないための要求プロセス技術として、我々が議論した枠組や技法が有効であることを確認した: 1. 救急車乗務員がstakeholderの一人であることは容易に発見できると考えられる。2. 問題点の回避のためには、例外のチェックを徹底的に行う必要があり、シナリオの利用(第3の表現)、ユーザインタフェースの専門家の参加、実機を用いた現場での目的別のプロトタイプやシミュレーションが有効である。3. 既存システムからの変更点を洗い出し、集中的に分析することが有効である。4. 要求の理由をstakeholderの言語へ翻訳して理解してもらうことは有効である(例えば、なぜボタンを押さなければならないかをstakeholderに理解させる)。5. 要求の取捨選択に関するトレーサビリティを保持することは、次期システムの要求を固めるときに有効である。

3 ソフトウェア新工法

ソフトウェア新工法とは，ここ数年急激な勢いで伸びている新種のソフトウェアに対応するための工法である．パーソナルユースのコンピュータとネットワークが日常の業務や生活に行き渡り，インターネットのブームを経て，生産的作業環境として使えるイントラネットの導入，改善がいまさかんに行なわれている．新種のソフトウェアとは，このような新種の環境を形成する比較的軽いシステム要素である．その開発には，要求の定義，仕様に基づいて設計を行ない，そのすべてをプログラミングするという工程ではなく，システム形態の基礎を与えるプラットフォームとその上でのコンポーネントの組み立てによる，いわばソフトウェアシステムのプレハブ工法が適している．

今回のワークショップでは，ソフトウェア新工法が従来のソフトウェアエンジニアリング技術といかに差別化されるかを明らかにし，ひいては新工法推進のために必要な新技術課題を的確に認識することを目標とした．本研究会における新工法関係のイベントは，96年7月の大阪研究会でのパネルに続いてまだ2度目であり，新工法といっても，コンポーネントウェアやパッケージソフトといった用語から類推される漠然としたイメージがあるのみである．実際，寄せられたポジションペーパーにも，アーキテクチャ，コンポーネント（部品）やオブジェクトといった対象を中心に見たものや，コンポーネントベースのプロセス，方法論，環境など，工法そのものの考察を中心にしたものなど，新工法に対する参加者のさまざまな角度からの視線が交錯していた．

参加者全員のポジションステートメントを受けた議論でさまざまなキーワードが飛び交った後，視点の拡散を防ぐため，まずソフトウェア新工法の性質と対象を絞り込む作業を行なった．在来工法が重厚長大ソフトウェアに対するものであり，これを単純に裏返せば，新工法は軽薄短小ソフトウェアを相手にするものということになる．そこでこの軽薄短小をさらに解析してみた．

［軽］ データ量が軽い，トランザクションが軽い，セキュリティが軽い，…．そこそこの機能，性能，信頼性，…，が満たされればよい．安い．Good enough for me.

［薄］ ドメインが薄い（狭い）．だれにでもわかる．簡単に使える．ドキュメント指向．

［短］ ライフサイクルが短い．すぐに使える．早くできる．

［小］ サイズが小さい．オフィスシステム，グループワークシステム，…．

このような性格に加え，新工法の対象に共通すべきものとして，WWWの応用と拡張などを典型例とする

ネットワークシステム，分散システム

があるというコンセンサスも得られている．

性質と対象が決まったところで次の作業は，これまでのセッションでふんだんに吐露されたさまざまな技術的手法の整理である．コンポーネントとコンテナに対する認識や，シナリオをもとにしたスクリプト言語の体系などに関する討論を経て，試行錯誤の末，表1に示すような枠組で整理された技術項目一覧表をまとめた．

行見出しは新工法を施工する役割である．第1行はコンポーネント自体を「作る」コンポーネントベンダの技術課題である．第2行の「選ぶ」という役割は，在来工法に対応するものがない新工法に最も特徴的な部分であって，ソフトウェアソムリエ[1]といわれるような新職種を生む可能性もある．第3行は，選ばれたコンポーネントを「組立てる」際の技術課題で，ここにも従来の再利用技術とは異なる，エンドユーザ対応の技術項目が含まれる．

列見出しは，ソフトウェアエンジニアリングに共通ともいえる視点である．工法の対象物としてのプロダクト，工法を進める方法論やプロセス，そしてそれを支援する環境に区分している．

以下，各技術領域を簡単に説明する．

作る・プロダクト 新工法の場合，プロダクトは選ばれ，組立てられる対象としてのコンポーネントである．適用の場合に応じて，特殊高機能な性質と一般汎用的な性質を計測，調整できるような，少なくともその性質を系統的に整理できるような技術が求められる．これが動的，自律的に調整できれば，

[1]NTT・忠海 均 氏，シナジーインキュベート・菊田昌弘氏らにより，日本規格協会のソフトウェアCALS委員会で紹介された．

表 1: ソフトウェア新工法の技術課題

	プロダクト	プロセス	環境
作る	特殊化 / 一般化機構 適応 / 進化機構 インタフェース定義 プロトコル定義	?	?
選ぶ	カタログ技術 (分類学)	検索自動化 (エージェント) 流通機構 評価法 コンテクスト記述 / 解析	コンポーネントブローカ
組立てる	アーキテクチャ カスタマイジング / テーラリング	設計制約 スクリプティング 中粒度対応	ビジュアル化

適応 / 進化機構を備えたコンポーネントとなる．一方，コンポーネントは他との協調によって動作するものであるから，インタフェースやプロトコルの定義 / 解釈法も重要な要素技術である．

選ぶ・プロダクト この領域は「プロダクトを選ぶ」と読むのではなくて，「選ぶためのプロダクト技術」である．（前項，次項も同様．）対象ドメインの「薄い」コンポーネントが森羅万象に渡り出回ったとき，しかも，ワインを料理に合わせなければならないように，さまざまな適用対象に合わせてコンポーネントを選ぶためには，カタログを形成 / 利用する技術が求められる．情報の抽象化，統合化一般論，ないし分類学という途方もない領域ともいえる．

組立てる・プロダクト コンポーネントの納まるコンテナの構成を決めるのが，プロダクト技術としてのアーキテクチャの役割である．ここに示したカスタマイジング / テーラリングは，それが構造的に容易なプロダクトの性質を求めることを意図しており，例えばメニュー項目の同定やスクリプト記述の言語要素などに反映される．

選ぶプロセス 検索の問題，流通の問題とともに，コンポーネントの機能性，性能や信頼性，さらには適用性や接続性などの品質を評価する規準が必要である．そしておそらく最も手に負えないのがコンテクストであろう．コンポーネントが周りの環境に合うかどうか，コンポーネント同士に「食い合わせ」がないか，といった整合性を調べる技術である．

組立てるプロセス アーキテクチャのプロセス的意義は，設計に（物理的）制約を与え，選択肢を減らして「割り切る」ことにより，工程進捗を加速させることにある．どのような制約が有効か，それがどのように要求に影響するか，などが研究課題となる．スクリプティング方法論は構造的化テーラリングのプロセスを与える．適用領域依存性をいかに乗り切るかがひとつのかぎとなろう．中粒度対応では，従来の「拡散→収拾型」のプロセスから「決め打ち型」プロセスへの移行が課題となる．

「作るプロセス / 環境」の技術項目は不明のまま終った．「作る」ところでは，モジュール設計やプログラミングも含まれることから，在来工法がある程度そのまま生きるとも考えられる．しかし，ここでもプロダクト技術には新工法特有のものが多く，それらに対する方法論や支援技術が今後項目としてあがってくることも期待される．とくに，ドメイン依存のプロダクトに対して，ドメイン独立の環境が目標とされる．環境については総じて閑散としているが，プロダクトやプロセスについても

項目を同定したまでなので，具体的に何をどう支援すればいいのか必ずしも明らかになっておらず，今後の研究テーマが豊富に残されているということである．

このように研究開発メニューが充実してくると，新工法の対象は必ずしも軽薄短小ソフトウェアに限らないのではないかという欲も出てくる．「短」のメリットを保ちながら，「軽」を補うことは，コンポーネント自体の高度化，重装備化が組立てに悪影響を及ぼすことなく進められれば，あるいは可能かもしれない．

4 西暦2000年問題とソフトウェア保守

西暦2000年問題を中心に，状況，解決手法とツール機能について討論した．その中で，今後取り組むべき問題点を明らかにした．

4.1 西暦2000年問題

2000年問題は同時期に発生し，しかもシステムダウンなどにより社会的影響を起こす可能性が高い．今回，この2000年問題を取り上げ，現場で実際に対応している方々に参加を呼び掛けて，オープンな情報交換と討議を行った．参加者は，白杉（新日鉄情報通信システム）本村（ソフトウエアジェネレーション）寺崎（日本IBM）藤田（野村システムサービス）佐野，小川（富士通）滝（NEC）中野，小林，津田（日立）のＳＩベンダを中心とした方々である．各参加者が対応方法や対応事例を持ちより，いくつかの課題に分けて議論した．

4.1.1 2000年問題の認識

今回のワークショップでも，アカデミック分野の参加者の関心は低く，「危機感をあおっているのではないか？」という意見もあった．一般企業においても，まだ認識が不十分で，JISA/JUSAの調査では未検討の企業が23.3%ある．また，ガートナグループの米国企業CEO調査では，34%のCEOが「関係ない」「知らない」と回答している．各参加者は，「もっと危機感をもたないとあぶない」との意見が大勢で，その理由は以下の通りである．

- 「いつまでに対応しないとビジネスに影響があるのか」が分からない
- 多くの装置にソフトが組込まれており，影響が大きい．例えば，自動倉庫の商品保管期限管理で間違って商品が破棄されるてしまう．
- バブル崩壊後，多くのプログラマが業界からいなくなり，人手不足が起こる．

課題は，経営者への説得と予算化であり，2000年問題を情報システム部門の問題ではなく経営的な問題と捉えて全社横断の対応組織の設置が重要である．

4.1.2 2000年問題対応シナリオ

効率的に解決するために，見積や対応計画立案，対応方式について議論した．見積の課題は，まだ見積技法が確立されていない点である．これは，実施経験が蓄積されておらず学習効果がでていない理由による．また，「対象ソフト資産が整理されているか，」「担当者がシステムを熟知しているか」，などのパラメタの影響も大きい．海外の実績報告も出ているが，「和暦がない」「国外への安価な発注先がある」「リスク管理のコストが入っている」こともあり，ストレートに参考にするのは問題がある．対応計画では，優先順位の付けかた，パイロット調査による見極め，短期間に逐次対応していくショートサイクルの作業計画がキーになる．また，習熟メンバを集中化して対応するセントラルキッチン方式が有効である．2000年対応ガイドが各社から出ており，これはWWWで読む事が出来る．2000年問題の説明と影響，問題への取組み方と対処方法が解説してある．代表的なガイドを紹介する．

- 「IBM西暦2000年対応計画・導入ガイド第5版」（日本IBM），
  http://www.ibm.co.jp/ad2000/1doc.html
- 「西暦2000年対応ガイドブック」（NEC），
  http://www.sw.nec.co.jp/ad2000

対応方式では，2桁不変方式と4桁拡張方式について議論した．一般的には4桁方式の方が，データ移行の発生と修正モジュールが増えることによりコスト高になると認識されているが，事例報告の中に，2桁方式よりも少ないコストで対策した事例があり，方式の選定基準が求められている．また，企業間データの扱いでは，「現行通り2桁のままで並び順で交換」する方式がベストとい

う結論になった．しかし，企業間の連結テスト方式については課題が残った．

4.1.3 2000年対策特有の技術的課題

2000年対策作業では，日付データ項目の検出，影響波及解析，ソース修正，移行，テストの作業を行う．プログラムに書かれている日付項目の検出率向上には，ファイルレイアウトの確定がキーになる．他には，ルールによる検索方法もあり，起点設定による実データ解析も有効である．影響波及解析では，過剰波及と見落しへの対策がいる．波及解析は支援ツールで行うため，ツールの解析確度の分析をおこなう．過剰波及にたいしては，ツールに波及限度値機能（打ち切り機能）を持たせ，対話的に精度を上げる作業が必要となる．いずれにせよ，日付項目の検出と影響波及解析の確認は人手作業になる．テストの工程は支援技術が少ないため，多くの課題がでた．テスト検証項目の設定と検証方法，テストデータ作成（特に4桁対応テストデータの作成）．日付という連続したデータであるために，過去のデータや2000年をまたがる連続データとの組合わせテストが発生する．2000年対策は，長期間に逐次的に行われるので，システム全体の一貫テストも課題である．また，膨大なソフト資産を対象に対策作業をするため，マシン環境の議論も出た．各企業のコンピュータ資源が不足する．対応のひとつとして，PC上でホストコンピュータのソフトウエアを稼動させてソース修正からテストする方式がある（製品名：ENI390 IBMのMVSが稼動）．これらの作業を支援するツールの議論も行った．ツールの限界点や効果などの検証が必要．今回，紹介された支援ツールは，悟（さとる），MAINTENANCE/2000（日本IBM），TASKAL2000, RWB/2000（富士通），PRO-AD2000（NEC），THINKAID/2000（日立），SCAN DATE（野村総研），RESCUE/2000, SOFTAUDITO, REVOLVE, HIDOCなどである．

2000年問題に関して，各社の課題やアイデア，有効方法などを自由に議論できたのでワークショップの意義はあった．各社共，多くの事例を発表したことで，議論がより深まったのも成果であった．今後は，より情報の共有化を図り効率的な問題解決を推進したい．

4.2 保守一般

保守テーマ一般の参加者は，玉井（東京大学），松尾谷（NEC），四野見，井上（日本IBM），上原，長橋（富士通研究所）の方々である．保守テーマでは，プログラムを対象とした保守には限界があり業務レベルの仕様を対象に保守すべきである，ソフトウェア保守の要求がどのような目的により生ずるのかを明らかにすることにより保守作業のあり方を改善できるのでは，といった議論を行った．また，ソフトウェア進化プロセスを定量的に測定し分析することにより，ソフトウェアの開発保守を捉えていく研究が紹介された．リバースエンジニアリングやプログラム解析ツールに関しても多くの議論を行った．特に保守ツール開発基盤の要件として，

- 構文解析，変数辞書，制御フロー，データフローなどの基本となる解析情報を容易に作成しアクセスできること（APIの整備も重要）
- 様々な解析ツールから利用し易いよう，各種の解析機能を階層型に構成し提供すること

等を議論した．2000年問題を契機として，資産を分析・理解したりデータフローを追跡する機能が今までになく強く求められている．この機会に有効な保守ツールをユーザに提供し広めていくべきである．

5 おわりに

ウインターワークショップ・イン・松山での討論内容と成果を中心に紹介した．本ワークショップがソフトウェア工学の進展に寄与することを願っている．

最後になるが，本ワークショップの企画・運営に御尽力戴いた情報処理学会事務局，ソフト工学研究会連絡委員の皆様，ワークショップ実行委員の皆様，そしてワークショップに参加し討論戴いた皆様に深謝する．

参考文献

[1] 「ウィンターワークショップ・イン・松山」論文集，情報処理学会，1997年1月．

[2] Report of the Inquiry Into The London Ambulance Service, South West Thames Regional Health Authority, ISBN 0-905133-70-6, 1993.