解　説

分散開発環境

3. グループウェアのソフトウェア開発への応用†

垂　水　浩　幸††

1. はじめに

グループの協調作業を支援するグループウェア (groupware) が注目されている.

ソフトウェア開発はグループ協調作業の代表的なものである. たとえインハウス分散[1]の場合でも, グループの作業を支援し, グループ内コミュニケーションを円滑にするグループウェアはソフトウェア開発に有益である. 広域分散開発環境では対面会議の開催が困難になるなど既存のコミュニケーション手段に制限が出てくるので, なおさら新しい媒体としてのグループウェアへの期待が大きい.

その期待の大きさは, たとえば研究所だけではなく企業現場からもグループウェア導入についての検討報告が数多く発表されている[23),46),54),56)]ことからもうかがえる. また, 教育の現場である大学からも導入検討事例が発表されている[27)].

本稿では, まずグループウェアについて簡単に解説した後, ソフトウェア開発作業へのグループウェアの導入に関する研究事例を紹介し, 最後に問題点を示し, 今後の課題・動向について述べる. 特に仕様書作成時の議論 (Design Rationale) の蓄積技術についてはホットな話題でもあるので詳しく述べる.

2. グループウェア

グループウェアに関する一般的な解説文や調査はいくつか発表されている[7),19),20),38),41)]ので, ここでは簡単に概観するにとどめ, 分散開発と関係の強い研究動向について次章で重点的に述べることにする.

まず, グループウェアの定義であるが, ここでは**「共通の作業（または目的）に従事する人々からなるグループを支援し, 共有環境へのインタフェースを提供するコンピュータベースのシステム」**という Ellis の定義をあげておく[7)].

次に, グループウェアを大きく分けると, リアルタイム（同期）対面型, リアルタイム（同期）分散型, 非リアルタイム（非同期）型の三者にカテゴライズされる[19)]. リアルタイム型とは, 参加者がリアルタイムに対話をしながら同時に作業を進行するもの, 非リアルタイム型とは参加者が蓄積型通信を用いたメッセージ交換を行いながら非同期に作業を進めていくものである. また, 対面型とは全員が同じ部屋で肉声や直視による情報交換とともに利用するものであり, 分散型とはそうでないものである. 非リアルタイム型で対面型のものはありえない.

2.1 リアルタイム対面型のグループウェア

リアルタイム対面型の代表は Colab[53)] のような電子会議室である. Colab は, 会議室の各参加者の席にコンピュータが設置され LAN で接続し

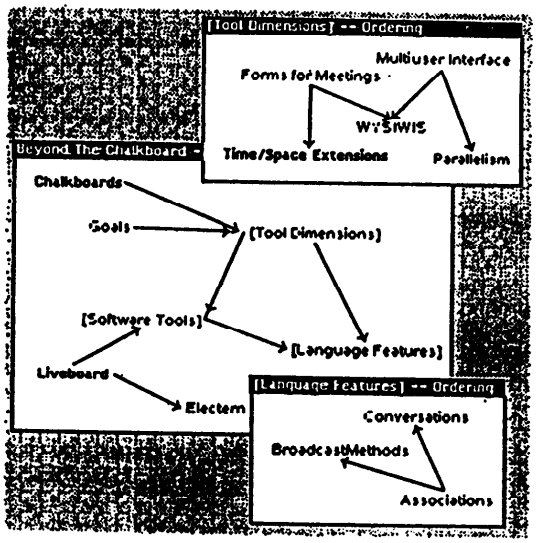

図-1　Cognoter の画面例

† Groupware for Software Development by Hiroyuki TARUMI (NEC Corporation, Kansai C&C Research Laboratory). Internet: tarumi@obp. cl. nec. co. jp

†† 日本電気(株)関西 C&C 研究所

たようなものであり，参加者は共用のウィンドウと個人用のウィンドウを利用することができる．共用のウィンドウで編集中のものは，それを共用している他の参加者も同じように見ることができ，共有編集が可能となっている．さらに，個人用画面以外に全員から見える位置（黒板に相当する）に共用スクリーンがあり，共用ウィンドウはそこにも表示される．

また，Colab の中で用いられる Cognoter[9]（図-1）では，アイデアを共用ウィンドウで木構造状に編集することができ，ブレーンストーミングなどの支援が可能である．

2.2 リアルタイム分散型のグループウェア

リアルタイム分散型のグループウェアには，ワークステーションを用いて在席したままで会議を可能にした MERMAID[59),60)] や TeamWorkStation[18)]，分散エディタの GROVE[7)] や DistEdit[25)]，意志決定を支援するためにハイパテキストの共有編集を可能にした rIBIS[49)]，臨場感通信によって非公式の対話をサポートしようとした CRUISER[52)] などがある．

2.3 非リアルタイム型のグループウェア

非リアルタイム型のグループウェアの研究分野から電子メールシステム，ハイパテキスト共有，共同執筆の三つについて述べる．

電子メールシステム

電子メールシステムがそれ自体グループウェアと言えるかどうかは議論の余地があるが，非リアルタイムグループウェアを実現するための基盤技術として重要であることは間違いない．現在ソフトウェア技術者に広く使われている UNIX 上の電子メールの場合，読まれたことの確認がとれない，数多く送られてくるメッセージのフィルタリングが困難であるなどの問題点が多い．それを改善する試みとして 1)マルチメディアのメールを実現したほか，投票やメールを読んだことの確認が容易に行える Active Message 機能や，メッセージのフィルタリング，プライベートな掲示板などの機能を有する Andrew Message System[2)]，2)半構造化したメッセージ (Semi-Structured Message) を利用し，ルールベースでのメッセージフィルタリングを可能にした Information Lens[34)] とそれを発展させた Object Lens[28)] などがある．特に Information Lens でその効果が強調された半構造化手法は支持を得ており，日本国内でも研究開発が盛んである[17),37),39),45)]．

一方，OA 分野では，パーソナルコンピュータ上での電子メールシステムが欧米で比較的よく普及している[4),41)]．これには日本と比較して企業内コンピュータネットワークがよく発達しているという背景がある．OA 向けのメールシステムにはスケジュール管理機能と連携するものや受信の確認がとれるものなどがあり，実用面で UNIX 上のものよりも勝っている．

ハイパテキスト共有

共有されたハイパテキストの非同期編集を可能にすることにより，意志決定を支援するハイパテキスト共有型のグループウェアも一つのカテゴリをなしている．代表的なものは gIBIS[5)] で，これは先に述べた rIBIS の非リアルタイム版である．

gIBIS は IBIS (Issue Based Information Systems) という問題解決手法をグラフィカルなインタフェースでサポートしている．IBIS モデルでは問題点 (Issue)，問題に対する解答案 (Position)，それに対する議論 (Argument) の三種のノードを使用する．これらの関係を図-2[5)] に示す．gIBIS のユーザはグラフィカルなユーザインタフェースによってこれら三種類のノードを木構造に整理し，ノードの追加，ブラウジング，検索などを行うことができる．

共同執筆

共同執筆システムとして著名なものには Quilt[32)] がある．Quilt では，遠隔地にいる共同執筆者との間で文書添削などの共同作業を可能にした．添削コメントなどは電子メールで転送される．文書に対する編集権の割り当てを指定することができるので，共著，通常の添削，編集などの用途に応じて使い分けることができる．

他には文献16)，55) など日本での共同執筆研究例もある．

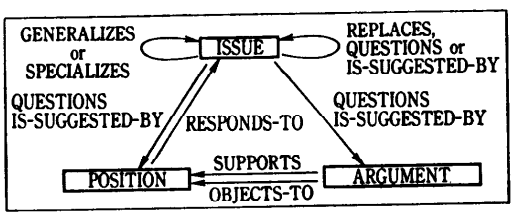

図-2 IBIS モデル

3. 分散開発とグループウェア

3.1 分散開発におけるグループウェアの役割

通常，ソフトウェア開発はグループ共同作業である．たとえば開発作業者の作業時間配分のうち半分以上が他の作業者への質問や情報提供であるという調査例[24]をみても分かるように，グループ内部でのコミュニケーションは開発作業のうちの非常に重要な部分を占めている．たとえばグループ内での意思疎通ができていなければ，インテグレーション時まで問題点が露見せず，後戻り工程を生じてソフトウェアの生産性を著しく低下させるが，このようなことはソフトウェア開発の常識であるといってもよいだろう．

今後の CASE に期待される Coordination, Navigation, Communication, Control, Adaptation の機能[44]のうち最初の4つは広い意味での開発者間のコミュニケーションに関連している．しかし現状では，ソフトウェア開発におけるコミュニケーションを本格的に支援するツールはまだ普及していない．作業者のコミュニケーションの効率を上げることに関しては方法論（たとえばレビュー方法論[10]など）に頼っている．

このように，CASE 分野へグループウェア技術を吸収する試みはまだ十分行われておらず，今後技術統合が大いに進展していく可能性が高い．以下に示す事例はそのような試みの一端である．

3.2 ソフトウェア開発向けグループウェアの研究事例

3.2.1 ICICLE

Bellcore で開発された ICICLE[3] は，ソースコード検査を支援するリアルタイム対面型* のグループウェアである．X Window 上で動作し，ソースコードを表示するウィンドウ，コメントを表示するウィンドウなど，複数のウィンドウを利用する．

ICICLE はコーディングエラーの一部を発見したり，コードの知識を提供するなど，CASE ツールとしてみても面白いシステムであるが，グループウェアとしての機能に注目すると，次のような特徴がある．

- コード検査会議における司会者，検査者，書記などの役割を支援する．
- ペーパーレスの検査作業を行う．ソースリストのハードコピーも紙への記録も行わない．
- 検査者は自分のワークステーションで検査作業ができる．
- コードに対するコメントは共有される．

ICICLE の適用実験の結果，利用者の評価は良好であったとされている．また手作業によるコード検査会議と比較すると，参加者の役割に変化が生じたと報告されている．たとえば，書記の役割は大幅に変化し，単なる記録係ではなく決定権をもち始めた．

3.2.2 KyotoDB

KyotoDB[36] はデータベース中心の総合的な CASE 環境であるが，その中に作業者間の対話機能（協調活動支援ツール）が含まれている[62),63]．

KyotoDB では，ソフトウェア構成要素とそれらの相互関係をデータベースに格納している．協調活動支援ツールは，仕様などの修正時にデータベース内の一貫性を保つため，作業者間でのコラボレーション（話し合い）が必要になったとき呼び出される．たとえばある仕様が変更になったとき，KyotoDB は変更の波及する部分の担当者のリストを自動的に作成し，コラボレーションの仲立ちをする．実際のコラボレーションは既存の対話ツール (UNIX の phone など) を利用しての話し合いとなる．

3.3 Design Rationale の記録

3.3.1 Design Rationale とは

1987年に Curtis ら[6] は，ソフトウェア開発の重要な部分はアプリケーション要求内容の把握や技術的コミュニケーションなどのソフトウェア成果物 (artifact) を得る過程にあり，それらに対する視点が従来のソフトウェア工学では欠けているということを示した．このことはグループウェアのソフトウェア開発への応用と深く関係している．

Design Rationale すなわち**設計理由**を成果物とともに記録していくという考えは Curtis らの指摘に対する解決策の一つである．ソフトウェア設計の成果として出てくる仕様書はユーザのなんらかの（明文化されていない）目的や，他の仕様書から導出されるものである．その導出過程にはさまざまな議論，比較検討が技術的あるいは非技術的な面から行われているはずである．そのような

* 文献 3) によれば分散化の計画あり．

議論・検討を記録しておくことにより，以下のような利点が出てくる．

- 仕様変更などの保守の場面で，仕様決定の経緯が検索できれば，容易に変更できる部分や変更不可能な部分などをより正確に同定できる．
- 過去のシステム事例や他人の作ったモジュールを再利用するようなとき，カスタマイズのための資料となる．
- 仕様の決定理由を忘れて，過去の議論を蒸し返すような無駄がなくなる．
- 特に深い理由もなく決められた仕様を必要以上に絶対視して，見直しの機会を逸するような不合理が減少する．
- 議論を記録するという行為を意識することにより，問題点の積み残しに気づかずに下流に進むようなことを避けられる．

これらの利点は集中開発環境においてすら重要である．分散組織環境では日常のコミュニケーションが疎になりがちであるため，Design Rationale を記録することは，ノウハウなど開発情報の共有のためにより一層有益なはずである．

3.3.2 モデル

設計理由を記録するモデルのアプローチとしては IBIS 系のものと知識処理指向のものの二つがある．前者は比較的非形式的に記録ができて分かりやすいが，自動推論機構との相性は良くない．後者は自動推論との相性は良いが，モデルはやや複雑になる[33]．

IBIS 系のものの代表として Potts のモデル (図-3[48]) がある[47),48)]．このモデルは，成果物とそれを書き換えるステップと，IBIS 議論モデルの三要素 Position, Issue, Argument との関係を図示している．議論の結果選択された Position が次のステップに反映される．

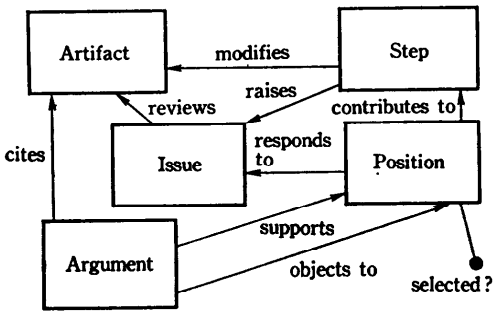

図-3 Potts のモデル

知識処理指向のものの代表として，後述する SIBYL で用いられている DRL (Decision Representation Language) がある[29),31)]．DRL では，IBIS と異なり 5 種類のノード Goal, Alternative, Claim, Question, Procedure とそれらの間の関係 (約 14 種類) によるグラフで議論の過程を表現する．

DRL のモデルを利用して議論の過程を図示した例を図-4[31)]に示す．この例では，「Zeus システムの製作に利用する最適な知識表現言語を決定する」という Goal に対して，KEE, LOOPS, STROBE の三つの Alternative を比較している．この Goal にはいくつかのサブゴール (楕円で示され，is-a-subgoal-of という関係で結ばれている) がある．たとえば LOOPS と STROBE という Alternative は，「開発コストを最小にする」というサブゴールを満足させる (facilitate 関係)．これらの関係やノードに対し Claim, Question, Procedure ノードが次々と張り付けられていく．

このモデルでは，関係も Claim の一種であるため，たとえば Claim A の内容に対する反論と Claim A が Claim B を支持しているという関係に対する反論を区別して記録できる．Claim に対する議論は長方形のノードに対して矢印が引かれ，関係に対する議論は矢印に対して矢印が引かれる．たとえば，STROBE が開発コストを最小にするという facilitate 関係について「STROBE には IMPLUSE が附属しており，それが多くのインタフェースを提供している」という Claim (図-4 中央下部) があり，これは先の facilitate 関係を支持している (support 関係)．この関係に対し，「そのインタフェースが制約になる」という反論がある (denies 関係) が，さらにその反論に対して「ソースコードを修正できる」という反論がある．この二つの反論は，関係に対する反論と Claim の内容に対する反論として区別されている．

さらに，Claim は確からしさ (plausibility) を値としてもっており，値の変化がグラフ上で波及していくような仕掛けになっているため，議論の状況変化に応じた Alternative の比較評価が数値的に行える．また，あるサブゴールを諦めるなどの仮想的な条件での推論を支援する Viewpoint という機能もある．

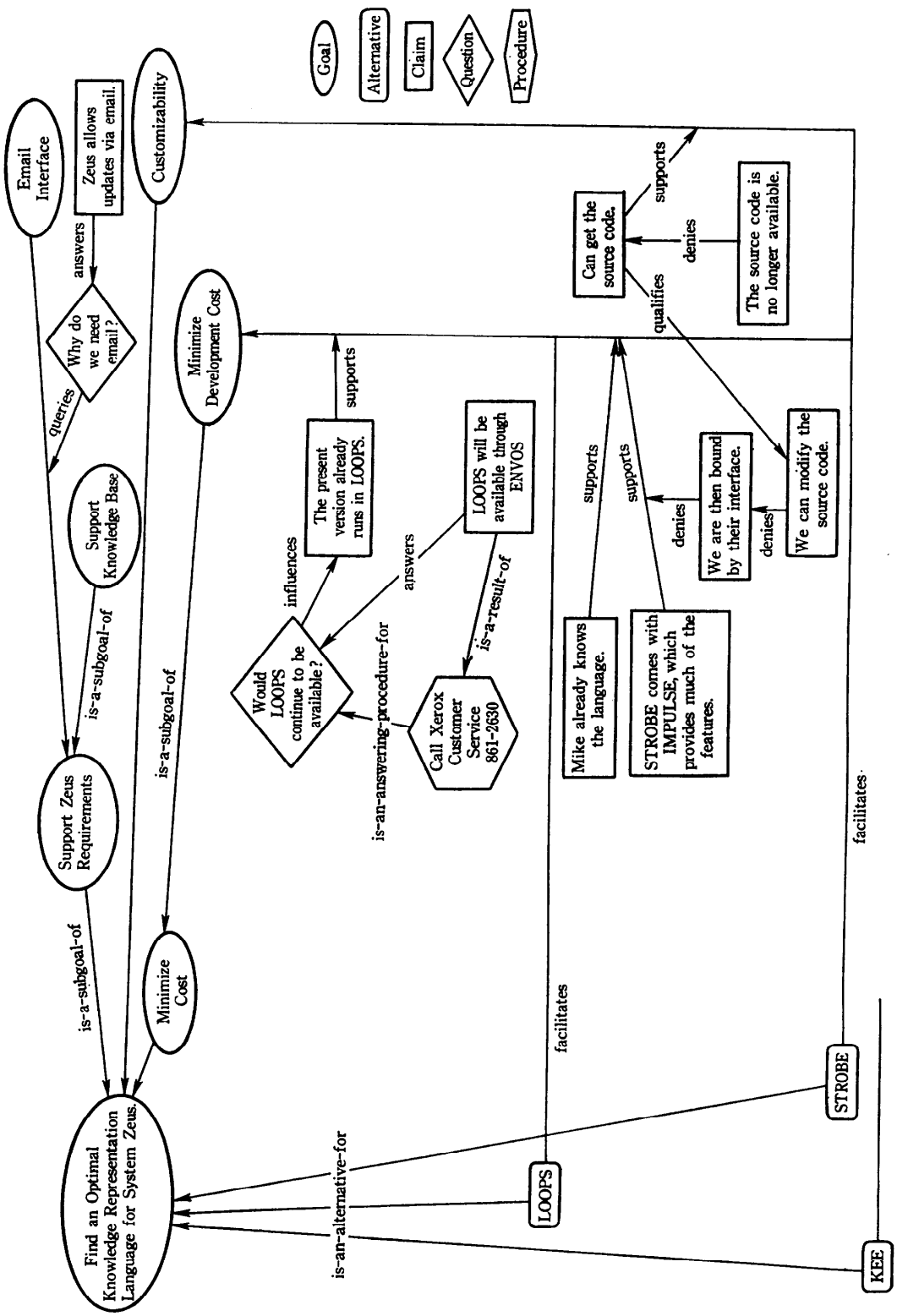

図-4 DRL のモデルによる議論記録の例

設計理由の記録モデルは記録が簡単でも検索や処理ができなければ価値はなく，逆に表現力に富み検索や処理が容易でも特殊な記録技術と記録工数を必要とするようでは一般の実用にならないので，まだ課題は多く残されている．両者の折衷を狙った例としては文献33)などがある．

3.3.3 記録方法

設計理由を記録する方法には，議論中の発言と同時に行う方法と，議論の後で行う方法がある[31]．

発言と同時に記録する方法の場合，時間的余裕がないので比較的簡単なモデルでフリーテキストを用いて記録していくことになる．したがって自動推論などの処理がしにくいが，記録工数はさほど発生しない．この場合，リアルタイムの場合と電子メールのような非同期型の場合とではおのずと記録方法が異なるはずである．

議論の後で記録する場合は，記録作業という手間が発生するが，表現能力のある言語で簡潔かつ正確に記録することが可能であり，後からの検索や種々の処理が行いやすい．これには，賛成反対意見をすべて記録する方法と，結論のみを記録する方法がある．

3.3.4 Design Rationale 研究事例

a) SIBYL

Lee は前述の DRL モデルによる設計理由処理システム SIBYL を開発した[30),31)]．これは Object Lens を利用したシステムの上で実装されている．DRL モデルの特徴で分かるように，SIBYL はグループウェアというよりは意志決定支援システムといったほうがふさわしい．

ユーザは SIBYL に対してたとえば 1)問題に対するいくつかの対案と各案の評価の表，2)評価理由または評価が未了の理由，3)ある仮定を置いたときの評価の変化，4)過去の事例などを問い合わせることができる．

SIBYL はどちらかというと記録された設計理由の利用技術についての研究であり，記録技術の研究ではない．

b) gIBIS

Yakemovic らは先に説明した gIBIS をソフトウェア開発現場で実際に応用し，その成果を報告している[61)]．彼女らは，18 カ月以上にわたり IBIS 手法を適用してソフトウェア設計の議論の経過を記録した．まず itIBIS (Indented Text の文字インタフェースによる IBIS ツール) で 8000 ノードの蓄積があり，次いでそれを gIBIS に移植し，このとき 800 ノードほどが追加された．彼女らの経験によれば，以下のような利点があったとされている．

- 問題の理解が早く，問題点が早期に発見されるため，開発コストが削減される．
- 議論されたこと，議論の未熟な部分が明らかで理解しやすい．
- 従来と視点の異なるレビュが行えた．
- 会議の議事進行予定（いわゆる Agenda）の作成に役立つ．そのため会議が生産的になり，議論がそれるとすぐに分かる．
- 作業グループ内外のコミュニケーションに役立つ．

ただし，だれの意見なのかが分かりにくい，何が議論されたのかは分かるが何が決まったのかが分かりにくい，などの問題点も指摘されている．

4. グループウェアの導入に関する問題点

グループウェアの導入には困難がともなうことが多く報告されている．これは，グループウェアが個人の好みで導入されるものではなくあるグループ全員が強制的に使用させられるところに大きな原因がある．

たとえば，Grudin[14)]は，グループウェアの導入によって利益を得る人と，仕事が増える人の不一致があるとき，導入がうまくいかないと指摘している．特に管理者は部下の仕事を増やしてでも自分に利益のあるようなツールは導入したがるが逆のものは導入したがらないと述べている．

また，Markus[35)]らはグループ構成員の役割が対称的（たとえば単純な電子メール）であっても，ある一定数の構成員（クリティカルマス）がグループウェアを受け入れないとグループ全体の利益にならないという解析を行っている．

以上の報告は特にソフトウェア開発作業を対象にしたものではない．ソフトウェア開発作業は一般にグループ構成員の数が比較的多く，作業の役割が対称的ではない複雑なものなので，Grudin や Markus の視点からいえばグループウェア導入への障壁はより高くなる可能性がある．

さらにツールの使い勝手についての問題も指摘

されている．たとえば Tatar らは Cognoter の使用経験について以下のような問題点を指摘している[58]．

● 同じものを見ていない．だから視線や身振りが役に立たない．また，「これ」などの代名詞が役に立たない．

● 変更が匿名で行われる．だれが変更したのか分からない．

● 項目の編集は個人ウィンドウで行われる．その結果は一度に送られ，急激な変化をもたらす．

● 他人の変更が伝わるまでに予期できない遅れがある．

● アイコンの移動は個人的に行われる．よってオブジェクトを識別するための場所の情報が共有できない．

● だれでもがタイプできるシステムなのに実際にはだれがタイプするか相談して決めている．

以上述べたような問題点や失敗例は，これまでのグループウェア研究があまりにも技術先導的で，グループ作業に関する社会心理学的な考察やグループ作業時のユーザインタフェースに対する経験が不足しているところから出てきている．このような問題を解消するためには地道なフィールドリサーチが必要である．これには会議などのグループ作業に関する心理学サイドからの報告[8),22)]や以下のような報告例などが役に立つであろう．

報 告 例 1

Galegher ら[12)]は，電子メールのみ使用するグループと非使用グループの間に同じ課題を与えて，学生を被験者とした比較実験を行っている．その結果，電子メールではなく対面会議を開くほうが，特にプロジェクトの立ちあげの場面で進捗が速いとしている．ただし成果の質はほとんど差がないということである．プロジェクトの初期の段階での対面会議の重要性については市村ら[15)]も述べているし，またソフトウェア開発の経験ともよく合致する．

報 告 例 2

Bullen らは，25 の組織グループウェア使用者 223 名にアンケート調査を行い，以下のような結論を出している[4)]．

● 日常の仕事を，日常より便利で，覚えやすく忘れにくい形でサポートしたものだけが普及する．電子メールがその例．

● 電子メールでは，複数のメッセージを関連づけて整理する機能が重要．

● ツール間の連絡が大変重要．孤立したツールは生産性を下げる．

● ツールの利用のために投資する努力と，ツールの使用によって得られる利益を比較して，後者が勝っていなければならない．

● ツール導入時の動機づけ，丁寧でリーダーシップのある導入指導，機械的ではなく，背後にある考え方にまで踏み込んだ教育，継続的発展的な指導が重要である．

● ただ導入するだけでなく，仕事の仕方を考え直さないと，生産性は向上しない．

報 告 例 3

小塚ら[26)]は電子メールの日常使用状況のデータをもとに，電子メール使用形態や発信数の偏りについて報告している（他に文献 57) など）．また，貫井ら[43)]は，組織に電子メールを導入する実験を行い，新しい環境に慣れるコストや非同期通信による遅延は大きな問題にならないとしている．

報 告 例 4

福田ら[11)]は，遠隔地で作業するソフトウェア開発者の間での電話利用時間を制限するという実験を行った．その結果，予想に反してコミュニケーションの回数と総時間はほとんど変化せず，むしろ短時間に集中してコミュニケーションが行われることで効率は良くなった．これは電話時の在席率の上昇や，電話による割り込みの減少の効果である．

報 告 例 5

仲谷ら[40)]は，ソフトウェア開発における問題発生時の臨時のコミュニケーションに着目し，モデル化した．このようなコミュニケーションでは，問題の発見者，解決策の考案者，解決策決定の責任者，解の実行者が一般に別であることが特徴だとしている．

5. その他の議論

5.1 要求分析用グループウェア

ソフトウェアの要求分析フェーズに専用のグループウェアに関する研究は今のところ事例が知られていない．これは，要求分析作業においては KJ 法など汎用の問題整理技法や，それらの技法に基づくたとえば Cognoter などの汎用ディス

カッションツールが使用可能であり，特にソフトウェア開発のためにチューニングしたグループウェアを必要とする積極的な理由が明らかでないためと推測される．しかし今後，要求分析における対話のモデル化の研究[51]などを基礎にして，要求分析専用のグループウェアが開発される可能性もある．

5.2 日本的グループウェア？

欧米と日本では意志決定の仕方が異なる[22]のだから，日本には日本流のグループウェアが必要なはずだという話は，専門家の間でもよく話題になる（たとえば文献19)）．実際に稟議や根回しのためのグループウェアが開発されたという話はまだ聞いていないが，そのようなものが出てくる可能性はあるだろう．

しかしソフトウェア開発用のグループウェアについて考えてみると，上流部分を除けば技術的検討，質問と回答，作業連絡が主になるので，稟議や根回しのシステムが必要になるわけではない．欧米との違いよりもむしろ作業グループごとの作業標準や習慣の違いが問題になる．同じ会社でも部所が異なればまったく違う作業標準を用いている場合もあり，カスタマイズ機能が重要である．

ただ，日本の場合キーボードに対する抵抗が（ワードプロセッサの普及にもかかわらず）まだ払拭されたとは言えないので，グループ全員にキーボードの使用を義務づけるようなグループウェアは欧米に比較すると失敗する確率が高くなる．ソフトウェア開発現場の場合でも，全員が積極的に電子メールを使用するほどキーボードに慣れ親しんではいないといってよい[57]．

また，日本語の場合仮名漢字変換があるので慣れた人でも欧米なみの打鍵速度は期待できず，たとえば Cognoter のような会議中にタイプをするようなシステムの導入はより困難であろう．

6. 今後の課題

4.で述べたように，グループウェアの現場導入にはまだまだ問題が多く，グループウェアの研究コミュニティでもこの点が現在の課題となっている．

ソフトウェアの分散開発にグループウェアを導入する場合，さらに問題が多い．CASE ツールを導入するだけでも，開発方法から変化させなくては意味がなく，仕事のやり方の変化には抵抗が大きいという経験がある（文献42)など）．この上グループウェア導入の問題点が重なるわけであるから，CASE とグループウェアの統合したシステムが実用になるためにはメンバ全員が享受できる明らかなメリットがなければならない．この際，ユーザなどソフトウェアの専門家以外も開発に参加することも考慮しなければならない．

今後は心理学者・社会学者なども含めた学際的な研究を活発にしていき[21]，現場で使えるグループウェアを模索していかなくてはならない．

また，技術面では，グループウェアのツールキット化が重要である．グループエディタ，電子メール，リアルタイム会議などのグループウェアの個々の機能要素を部品とし，部品の組合せとカスタマイズによって現場の事情に合わせたグループウェアアプリケーションを自由に構成できるようにしなければならない．また，それにより現場で繰り返し評価しながら改善していくことも容易になる．ツールキット的な意識は DistEdit[25]，LIZA[13]，鼎談[50]などにみられる．

7. おわりに

ソフトウェア開発への応用を意識してグループウェアの技術動向と問題点について述べた．参考文献リストにあげたものの多くは 1988 年以後の文献であり，この分野がまだ若く，現在活発に研究が進行していることが分かる．本稿の作成に当たってはできるだけ最新の情報を集めるよう努力したが，おそらく執筆後にもさまざまな関連研究発表がなされるであろう．本稿で述べたようにグループウェアのソフトウェア開発への応用には問題点が山積しているが今後の研究によって一層の進展が期待される．

謝辞 本解説をまとめるに当たり，富士通(株)の青山氏，三菱電機(株)の坂下氏，日本電信電話(株)の長野氏，京都大学の鯵坂助教授，および査読者の方より貴重な助言をいただきました．ここに深謝いたします．

参考文献

1) 青山幹雄：分散開発環境：新しい開発環境像を求めて，情報処理，Vol. 33, No. 1, pp. 2-13 (1992).
2) Borenstein, N. S. and Thyberg, C. A.: Cooperative Work in the Andrew Message System,

CSCW '88 Proceedings, ACM, pp. 306-323 (1988).
3) Brothers, L. et al.: ICICLE: Groupware for Code Inspection, *CSCW '90 Proceedings*, ACM, pp. 169-181 (1990).
4) Bullen, C. V. and Bennet, J. L.: Learning from User Experience with Groupware, *CSCW '90 Proceedings*, ACM, pp. 291-302 (1990).
5) Conklin, J. and Begeman, M. L.: gIBIS: A Hypertext Tool for Exploratory Policy Discussion, *CSCW '88 Proceedings*, ACM, pp. 140-152 (1988).
6) Curtis, B. et al.: On Building Software Process Models Under the Lamppost, *Proc. of the 9th International Conference on Software Engineering*, IEEE, pp. 96-103 (1987).
7) Ellis, C. A. et al.; GROUPWARE: Some Issues and Experiences, *Comm. ACM*, Vol. 34, No. 1, pp. 39-58 (1991).
8) Forsyth, D. R.: *An Introduction to Group Dynamics*, Brooks/Cole (1983).
9) Foster, G. and Stefik, M.: Cognoter: Theory and Practice of a Colaborative Tool, *CSCW '86 Proceedings*, ACM, pp. 7-15 (1986).
10) Freedman, D. P. and Weinberg, G. M.: *HANDBOOK OF WALKTHROUGHS, INSPECTIONS, AND TECHNICAL REVIEWS, Evaluating Programs, Projects, and Products (Third Edition)*, Little, Brown, and Company, Inc. (1977) (岡田他監訳, TBS 出版会).
11) 福田由起雄, 井上敦子, 津田純一郎: 遠隔地ソフトウェア開発の実験, 情報処理学会ソフトウェア工学研究会, 71-7 (1990).
12) Galegher, J. and Krout, R. E.: Computer-Mediated Communication for Intellectual Teamwork: A Field Experiment in Group Writing, *CSCW '90 Proceedings*, ACM, pp. 65-78 (1990).
13) Gibbs, S. J.: LIZA: An Extensible Groupware Toolkit, *CHI '89 Proceedings*, ACM, pp. 29-35 (1989).
14) Grudin, J.: Why CSCW Applications Fail: Problems in the Design and Evaluation of Organizational Interfaces, *CSCW '88 Proceedings*, ACM, pp. 85-93 (1988).
15) 市村 哲他: チーム協調作業のための支援環境モデルの提案, 人工知能学会第4回全国大会, 15-3, pp. 449-452 (1990).
16) 市村 哲他: チーム協調作業支援環境—共同執筆支援への適用, 情報処理学会マルチメディア通信と分散処理研究会, 47-3 (1990).
17) 市村 哲, 松下 温: 誤解の発生を抑制するメイルシステム, 人工知能学会第5回全国大会, 13-4 (1991).
18) Ishii, H.: Team WorkStation: Towards a Seamless Shared Workspace, *CSCW '90 Proceedings*, ACM, pp. 13-26 (1990).
19) 石井 裕: CSCW: コンピュータを用いたグループワーク支援の研究動向, コンピュータソフトウェア, Vol. 8, No. 2, pp. 14-26 (1991).
20) 石井 裕: グループウェアのデザイン, bit, Vol. 23, No. 3, pp. 273-283 (1991).
21) 石井 裕: コミュニケーションからコラボレーションへ, 電子情報通信学会ヒューマンコミュニケーション研究会, HC 91-15, (1991).
22) 石川弘義: 会議の心理学, 筑摩書房 (1983, 文庫版 1986).
23) 川尻信哉他: リアルタイムシステム分散並行開発環境: ICAROS, 情報処理学会ソフトウェア工学研究会, 73-15 (1990).
24) Kedzierski, B. J.: Communication and Management Support in System Development Environments, in *Computer Supported Cooperative Work: A Book of Reading*, Greif, I., Ed., Morgan Kaufmann (1988).
25) Knister, M. and Prakash, A.: DistEdit: A Distributed Toolkit for Supporting Multiple Group Editors, *CSCW '90 Proceedings*, ACM, pp. 343-355 (1990).
26) 小塚 宏, 辻純一郎, 坂下善彦: 分散環境における業務情報交換の試み, 情報処理学会マルチメディア通信と分散処理研究会, 47-10 (1990).
27) 凍田和美, 宇津宮孝一, 吉田和幸: 統合コミュニケーションシステムを用いたグループプログラミング支援の試み, 情報処理学会ソフトウェア工学研究会, 77-6 (1991).
28) Lai, Kum-Yew and Malone, T. W.: Object Lens: A "Spreadsheet" for Cooperative Work, *CSCW '88 Proceedings*, ACM, pp. 115-124 (1988).
29) Lee, J.: SIBYL: A Qualitative Decision Management System, in *Arificial Intelligence at MIT*, Winston, P. H. and Shellard, S. A. Eds., pp. 104-133, MIT Press (1990).
30) Lee, J.: SIBYL: A Tool for Managing Group Decision Rationale, *CSCW '90 Proceedings*, ACM, pp. 79-92 (1990).
31) Lee, J.: Extending the Potts and Bruns Model for Recording Design Rationale, *Proc. of the 13th International Conference on Software Engineering*, IEEE, pp. 114-125 (1991).
32) Leland, M. D. P. et al.: Collaborative Document Production Using Quilt, *CSCW '88 Proceedings*, ACM, pp. 206-215 (1988).
33) Lubars, M. D.: Representing Design Dependencies in an Issue-Based Style, *IEEE Software*, Vol. 18, No. 4, pp. 81-89 (1991).
34) Malone, T. W. et al.: Semi-Structured Messages are Surprisingly Useful for Computer-Supported Coordination, *CSCW '86 Proceedings*, ACM, pp. 102-114 (1989).
35) Markus, M. L. and Connolly, T.: Why CSCW Applications Fail: Problems in the Adoption of Interdependent Work Tools, *CSCW '90 Proceedings*, ACM, pp. 371-380 (1990).
36) Matsumoto, Y. and Ajisaka, T.: A Data Model in the Software Project Database KyotoDB, *Advances in Software Science and Technology*, 日本ソフトウェア科学会編, Vol. 2, pp. 103-121 (1990).

37) 松尾 朗他：電子メールにおけるエージェントシステム，情報処理学会第 43 回全国大会，2J-13 (1991).
38) 松下 温 (編著)：グループウェア入門，オーム社 (1991).
39) 松浦宣彦，平岩真一，松下 温：情報の効果的活用に基づいたグループウェア，情報処理学会マルチメディア通信と分散処理研究会，50-14 (1991).
40) 仲谷美江，西田正吾：ソフトウェア開発プロジェクトにおけるトラブルコミュニケーションモデル―ソフトウェア開発プロジェクトにおける問題解決プロセス，第7回ヒューマンインタフェースシンポジウム論文集，計測制御学会，pp. 379-384 (1991).
41) 日本電子工業振興協会：ニューオフィスシステム (NOS) に関する調査研究報告書，91-シ-1 (1991).
42) 進化する CASE，日経コンピュータ，1990年11月5日号，pp. 78-101.
43) 貫井春美，栗原美佐，三原幸博：グループウェア支援機能の実験的考察，情報処理学会ソフトウェア工学研究会，71-8 (1990).
44) 落水浩一郎：CASE とは，日本ソフトウェア科学会サマーチュートリアル「90年代の CASE」資料，日本ソフトウェア科学会，pp. 1-13 (1991).
45) 奥村晃弘，北 英彦：電子メールに基づくグループウェア Brownie 他，情報処理学会第43回全国大会，3N-5～6.
46) 大江美和他：ソフトウェア開発におけるグループウェア利用の一考察，電子情報通信学会 1991 年春季全国大会，D-258, 259 (1991).
47) Potts, C. and Bruns, G.: Recording the Reasons for Design Decisions, *Proc. of the 10 th International Conference on Software Engineering*, IEEE, pp. 418-427 (1988).
48) Potts, C.: A Generic Model for Representing Design Methods, *Proc. of the 11 th International Conference on Software Engineering*, IEEE, pp. 217-226 (1989).
49) Rein, G.L. and Ellis, C.A.: rIBIS: A Real-time Group Hypertext System, *Int. J. Man-Machine Studies*, Vol. 34, No. 3, pp. 349-367 (1991).
50) 暦本純一：共有型作業空間における柔軟な同時実行制御方式，日本ソフトウェア科学会第8回大会，E3-3 (1991).
51) Robinson, W.N.: Negotiation Behavior During Requirement Sepcification, *Proc. of the 12 th International Conference on Software Engineering*, IEEE, pp. 268-276 (1990).
52) Root, R.W.: Design of a Multi-Media Vechicle for Social Browsing, *CSCW '89 Proceedings*, ACM, pp. 25-38 (1988).
53) Stefik, M. et al.: Beyond the Chalkboard: Computer Support For Collaboration and Problem Solving in Meetings, *Comm. ACM*, Vol. 30, No. 1, pp. 32-47 (1987).
54) 鈴木健造他：遠隔地・分散開発における開発管理についての一考察，情報処理学会全国大会，第41回大会 7G-2～4 (1990)，第 42 回大会 5S-3～4 (1991)，第 43 回大会 5J-7～10 (1991).
55) 田淵 篤：ハイパー・アノテーション―添削を中心とした共同執筆環境の提案，情報処理学会第43回全国大会，2F-11 (1991).
56) 高橋克也他：分散並行開発における協調作業支援環境，情報処理学会第 42 回全国大会，5S-1 (1991).
57) 垂水浩幸：電子メールによるディスカッション，「コンピュータネットワークのヒューマンウェア」シンポジウム報告集，情報処理学会 (1989).
58) Tatar, D.G. et al.: Design for Conversation: Lessons from Cognoter, *Int. J. Man-Machine Studies*, Vol. 34, No. 2, pp. 185-209 (1991).
59) Watabe, K. et al.: Distributed Multiparty Desktop Conferencing System: MERMAID, *CSCW '90 Proceedings*, ACM, pp. 27-38 (1990).
60) 渡部和雄他：マルチメディア分散在席会議システム MERMAID，情報処理学会論文誌，Vol. 32, No. 9, pp. 1200-1209 (1991).
61) Yakemovic, K.C.B. and Conklin, E.J.: Report on a Development Project Use of an Issue-Based Information System, *CSCW '90 Proceedings*, ACM, pp. 105-118 (1990).
62) 山下 薫，沢田篤史，鎌坂恒夫，松本吉弘：CASE 環境における協調活動支援ツールの試作，日本ソフトウェア科学会第7回ソフトウェア研究会「分散環境におけるシステムと応用一般」資料 (1990).
63) 山下 薫，鎌坂恒夫，松本吉弘：ソフトウェアエンジニアリングデータベース KyotoDB におけるユーザインタフェースと協調活動支援機能の実現，情報処理学会第 41 回全国大会，5H-2 (1990).

(平成3年9月27日受付)

垂水 浩幸 (正会員)

1960年生．1988年京都大学大学院工学研究科博士後期課程情報工学専攻研究指導認定退学．同年日本電気(株)入社．ソフトウェア生産技術開発本部を経て，現在同社関西C&C研究所勤務．京都大学工学博士．ソフトウェア工学とヒューマンインタフェース，特にソフトウェア開発環境に興味を持つ．ACM, IEEE Computer Sciety, 日本ソフトウェア科学会各会員．