SEAMAIL

Newsletter from Software Engineers Association

Volume 4, Number 6-7-8 1989

目　次

第４回　ＳＥＡ環境ワークショップ報告

ソフトウェア技術者協会（SEA）は，ソフトウェアハウス，コンピュータメーカ，計算センタ，エンドユーザ，大学，研究所など，それぞれ異なった環境に置かれているソフトウェア技術者または研究者が，そうした社会組織の壁を越えて，各自の経験や技術を自由に交流しあうための「場」として，1985年 12月に設立されました．

その主な活動は，機関誌 SEAMAIL の発行，支部および研究分科会の運営，セミナー／ワークショップ／シンポジウムなどのイベントの開催，および内外の関係諸団体との交流です．発足当初約 200人にすぎなかった会員数もその後飛躍的に増加し，現在，北は北海道から南は沖縄まで，1650 名を越えるメンバーを擁するにいたりました．法人賛助会員も約 60 社を数えます．支部は，東京以外に，関西，横浜，長野，名古屋，九州の各地区で設立されており，その他の地域でも設立準備をしています．分科会は，東京，関西，名古屋で，それぞれいくつかが活動しており，その他の支部でも，月例会やフォーラムが定期的に開催されています．

「現在のソフトウェア界における最大の課題は，技術移転の促進である」といわれています．これまでわが国には，そのための適切な社会的メカニズムが欠けていたように思われます． SEA は，そうした欠落を補うべく，これからますます活発な活動を展開して行きたいと考えています．いままで日本にはなかったこの新しいプロフェッショナル・ソサイエティの発展のために，ぜひとも，あなたのお力を貸してください．

ＳＥＡＭＡＩＬ　Ｖｏｌ．４，　Ｎｏ．６－８　　平成元年８月２８日発行

編集人　岸田孝一

発行人　ソフトウェア技術者協会（ＳＥＡ）

〒１０２　東京都千代田区隼町２－１２　藤和半蔵門コープビル５０５

印刷所　サンビルト印刷株式会社　　〒１６２　東京都新宿区築地町８番地

特価　１，５００円　（禁転載）

第４回　ＳＥＡ環境ワークショップ

ご　挨　拶

1. 実行委員長、プログラム委員長より

中野秀男

やっと第４回環境ワークショップの報告書がＳＥＡｍａｉｌに掲載できた。編集担当が他の仕事を優先にしたためである。テープ起こしが終わったのが３月で、セッションサマリは最後までかかり、最後の１人はＥｍａｉｌで送ると言いながら本人の環境の変化のためファイルまで出来ながら電送できず、もらったＡ４で２枚の原稿を改めて打ち込む事になってしまった。担当自身も昨年の年末に張り切ってアンケートをパソコンに向い家で深夜に入力した結果、年初から２月にかけて胃をいため、好きな酒も飲めずつらい思いをした。

次回の釧路は提案書を作るワークショップになる予定であり、このようにすべてを網羅する報告書は今回限りにするか、折角HyperMediaのセッションがあるから欲しい人にはHyperCardに整理してフロッピーにするべきだろう。（電脳机の発表を読んで欲しい）

やっと本論にはいる。第４回ソフトウェア開発環境ワークショップは神戸ポートピアの国際会議場で開かれた。例年、この環境ワークショップは全員が同じホテルに泊まり、そのホテルでセッションが行われたが、今回はやや都会の神戸のためか、新しくオープンした新神戸駅近くのオリエンタル・ホテルが一応のオフィッシャルホテルとなり、ホテルと会場の間は地下鉄とポートライナまたはタクシーで通勤することとなった。

今回のセッションは

「HyperText/Media」

「ManMachineInterface」

「MicroMainframeLink」

「ComputerAidedSoftwareEngineering」

であるが、特別に中国から３人の先生を招待し「中国セッション」を行った。夜のＢＯＦとして本音を語る（酔っぱらいのザレ言か？）セッションとして「理想と現実」のセッションがあった。今回はテープを完全に起こしたので、それらの発表と討論はあますことなく再現されていると思う。

本来、ＢＯＦはオフレコであるが、一部に危ない発言もあったようだが、あまりにも楽しかったので、すべての発言者に原稿のゲラを送り検閲して貰った上で記録を載せている。ワークステーション派のいじめとメインフレーム派の開き直りの対決となった「ＭＭＬ」のセッションから１時間程度の休憩の後でアルコール入りのＢＯＦとなったため異常に盛り上がった感があった。

この報告書では、各セッションごとの３つの発表レジメ、発表・討論記録、座長によるセッション・サマリ、発表と討論に対するアンケートからなる。全体に対するアンケートと何人かのポジション・ペーパも最後に添えてある。各発表の記録では示されたＯＨＰとの関連も出したかったが、そこまでする余力がなくあきらめた。それを期待された発表者の方にはあやまります。

中国セッションを除けば、記録はすべて話し口調のままである。中国セッションは、おそらくこのＳＥＡｍａｉｌが中国に送られるので、間違いと失礼のないように岸田代表幹事に最終チェックをお願いした。それ以外の記録のアンカーマンとアンケートは中野が担当した。各記録のテープ起こしは、各セッションのレポータとして登場しているが大変な仕事を頼んだと思う。私の担当の学生にはＳＯＮＹのＩＣリピートなる最大８秒間のリピート機能をもつテープレコーダを貸したが、このようなテープ起こしにもＳＥＡからアイデアが出ないかと思う。

要約した報告になるとどうしてもまとめた人の主観と偏見が入る。意見の違う人が読むと歯がゆくなることがママある。それもあって、今回は記録は省略しない方針と、発表者には記録の修正の機会を与えた。それにしてもＥｍａｉｌで連絡出来る人と出来ない人との最後の編集の仕事量の違いには大きな差があった。週末の楽しみのテニスを何度か我慢して校正した。

各セッションについての感想はアンケート等に反映されているのであえて書かないが、今回の反省は次回の釧路の環境ワークショップに出て来るだろう。釧路では今回までの流れを大きく変えて、より攻撃的な、愚痴のこぼしあいでない環境ワークショップになるだろう。スペクトルの広い、レベル差の大きいＳＥＡの中で、いくつかの切り口からソフトウェア技術者の先頭を走るためのワークショップになる可能性をひめている。逆に、普通（？）のＳＥＡ会員から遊離する可能性もひめているかもしれない。

各セッションの座長は、一部を除けば、各テーマをリードする人達であった。ＢＯＦの深瀬さん以外は、あまり自分の意見を途中で出さなかった。むしろ、冗長な、間違った発表や意見は切り捨て、たとえ文句が出ようとも流れを正しくする、座長好みにする姿勢も出して欲しかった。

―昨年、昨年のＳＥＡの中国訪問、今回の中国セッション、そして今年のソフトウェアシンポジウムと連なるＳＥＡと中国とのおつきあいが、今秋、日本で開催されるはずだった中日シンポジウムにまで持ち込めなかったのは残念である。しかし、中国は５０年や１００年の単位で考える国なので、丁度良いかもしれない。今年のソフトウェアシンポジウムの唐先生の基調講演もＳＥＡｍａｉｌに載る予定と聞いているが、今回の「中国セッション」も文字の形で残るのは良いことである。討論の記録にあるように、中国を新聞でしか知らない人には、すべて日本人の感覚で物を捉えようとする。

この原稿の草稿は、夏休みの休暇のためいる信州岩岳で書いている。もうすぐバーベキューが始まる。ビールが呼んでいる。もう少しでやめよう。もともと、この報告書の巻頭言は実行委員長のすべてがうまくいったと言う意味の「おことば」の予定であった。想像通り、待てど暮らせど原稿は飛んでこない。６月の時点で、実行委員長と２人のプログラム委員長の連名の「ことば」に変更になり、最終的にはプログラム委員長の１人「裏ジャック」の中野の独断と偏見に満ちた文から報告書が始まる事となった。来年のソフトウェア・シンポジウムの運営が暗示されるものである。やだ、やだ。

それはともかく、今回のワークショップにはＫＣＳの皆さんには大変お世話になった。盛田さんは言うに及ばず、岡本さん、平山さんはじめ若手の皆さんに世話になった。私のサイトのｏｕｃｏｍ２がＫＣＳとＳＲＡにネットワークで直結していることも大変助かった点である。

私もＳＥＡに入ってから、色々言われたが、その１つが「Ｏ型人間」だと言うことである。このＳＥＡｍａｉｌの裏表紙にあるように、ＳＥＡでは血液型に拘る。Ｏ型の「Ｏ」は大雑把の「大」らしい。それで、この文も大雑把のＯをそのままにした文になった。本来なら、大阪標準語で書くべきであるが、それは記録の所で読んで欲しい。大雑把の後は繊細な事務局の中島さんの「事務局から」である。宜しく。

2. 事務局から

事務局の中島です。今回のＳＥＡｍａｉｌは、昨年１１月に神戸で行われた環境ワークショップの報告書でまとめました。環境ワークショップはＰＣがいつも危ない人達ばかりなので、毎回ハラハラしますが、今回は伝説となった「神戸シーバス事件」の、あの神戸で行われるので心配でしたが、実行委員長はじめＳＥＡ関西の温かい人達のおかげで無事終了しました。それらはあますことなく報告書に現われていることでしょう。ただし、夜の部はいろいろあったみたいです。

ＳＥＡｍａｉｌは、これ以外にもプロセス・ワークショップ等の報告書がめじろおしです。お楽しみにして下さい。それにしても胃潰瘍寸前まで行きながら報告書をまとめられた中野先生、ご苦労さまでした。これからも、お酒をすごされないように。

第４回　実践的ソフトウェア開発環境に関する集中討論　プログラム

11月 24日（木）

Panel-1:　Hyper Media (10:00 ～ 13:00)

－　Hyper Media のソフトウェア開発への応用を探る

座　長：　熊谷　章（PFU）

話題提供者：　佐原　伸（SRA）　Hyper Card とソフトウェア開発

本田克巳（YHP）　電脳机の概念と展開

藤野晃延（FXIS）　CSCW の技術動向

Special Session:　中国軟件事情 (14:30 ～ 16:00)

－　いま中国のソフトウェア界では何が起こっているか？

座　長：　臼井義美（JIP）

話題提供者：　鐘　鍚昌（中国軟件技術開発中心）

朱　三元（上海軟件技術開発中心）

何　克清（武漢大学）

Tool Presentation (16:30 ～ 17:00)　　座長：林　香（SRA）

Tool Demonstration (17:00 ～ 19:00)

展示予定のツール（順不同）：

電脳机，HyperCard，Prototyper's Workbench，Xme，ERAS，StP，Mapper，Linc，Saber，Algorithm Animation，SAL，Visual MMI

情報交換パーティ (19:00 ～ 21:00)　　司会：平山伸一（KCS）

11月 25日（金）

Panel-2:　Man Machine Interface (10:00 ～ 13:00)

－　ソフトウェア（プロセス／プロダクト）の可視化はどこまで可能か？

座　長：　野村行憲（ICS）

話題提供者：　古閑幸一（HST）　AP 開発におけるシミュレーションの活用

塩谷和範（SRA）　PWB におけるマルチ・ビューの効用

柳瀬健一（KCS）　Xme による環境インタフェイスの改善

Panel-3:　Micro Mainframe Link (14:30 ～ 17:30)

－　現状における MML の問題点とその改善の方向を探る．

座　長：　久保宏志（富士通）

話題提供者：　坂下　秀（Astec）　理想の MML とは？！

関口純一（JIP）　MML 構築の一事例

平山伸一（KCS）　MML 構築上の諸問題

BOF:　開発環境の理想と現実 (18:00 ～ 21:00)

座　長：　深瀬弘恭（Ascii）

スピーカ：　岸田孝一（SRA）　上妻健一郎（熊本電応研）

高野　豊（松下電器）　平尾一浩（HST）

11月 26日（土）

Panel-4:　CASE (10:00 ～ 13:00)

－　いまの CASE ツールはどの程度有用か？　どう使えばよいか？

座　長：　野村敏次（JIP）

話題提供者：　小野康生（三菱電機）　制御分野での CASE

佐藤千明（長野県協同電算）　Mainframe環境の CASE

桜井麻里（SRA）　StP の利用経験と評価

第４回　実践的ソフトウェア開発環境に関する集中討論　参加者名簿（最終版）

No	氏　名	所　属	TEL	備考
1	盛田　政敏	ケーシーエス　ソフトウェア事業部	078-391-8291	実行委員長
2	臼井　義美	日本電子計算　大阪支店	06-448-6022	プログラム委員長
3	中野　秀男	大阪大学　工学部通信工学科	06-877-5111	プログラム委員長
4	青木　　淳	富士ゼロックス情報システム　技術推進室	03-378-8010	プログラム委員
5	井川　裕基	日本電子計算　大阪支店	06-448-6022	プログラム委員
6	久保　宏志	富士通　パッケージ企画統括部	03-437-5111	プログラム委員
7	熊谷　　章	ＰＦＵ　研究開発部	0427-96-5211	プログラム委員
8	坂下　　秀	アステック　カスタマ・サポート・グループ	03-477-1541	プログラム委員
9	佐藤　千明	長野県協同電算　業務部開発課	0262-28-3215	プログラム委員
10	塩谷　和範	ＳＲＡ　環境開発部	03-234-2611	プログラム委員
11	高野　　豊	松下電器産業　特別プロジェクト室	06-908-1151	プログラム委員
12	野村　行憲	岩手電子計算センター　医療システム部	0196-51-2626	プログラム委員
13	平尾　一浩	ヒラタ・ソフトウェア・テクノロジー	096-344-2611	プログラム委員
14	平山　伸一	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	プログラム委員
15	深瀬　弘恭	アスキー　技術企画室	03-486-1207	プログラム委員
16	松浦　敏雄	大阪大学　基礎工学部情報工学科	06-844-1151	プログラム委員
17	赤間　　保	富士通ビーエスシー　商品企画開発部	03-544-0506	
18	飯塚　正樹	横河ヒューレットパッカード　ＣＳＯレスポンスセンタ	03-331-6111	
19	池谷　　寧	富士通静岡エンジニアリング　第２開発部	0534-53-5513	
20	石元　繁一	三菱電機コントロールソフトウェア　技術統括部開発課	078-651-7361	
21	稲村　　浩	カノープス電子　第１技術部	078-411-5292	ツール展示
22	今別府芳暢	ＮＴＴ九州ＳＥ総合センタ　技術開発室	096-321-3611	
23	岩井田浩章	電力計算センター　大手町事業部	03-215-6481	
24	岡村　　博	三菱電機コントロールソフトウェア　技術第２部伝送課	078-651-7361	
25	小野　康生	三菱電機制御製作所　工業部制御計装システム設計課	078-652-2121	話題提供者
26	勝山　道朗	日本オリベッティ　システム開発部技術支援課	03-711-1246	
27	岸田　孝一	ＳＲＡ	03-234-2610	話題提供者
28	栗原　正利	ＳＲＡ　環境開発部	03-234-2615	
29	黒坂　靖子	日本電子計算　科学技術事業部技術営業部	03-668-6171	
30	桑名　栄二	ＮＴＴソフトウェア研究所　ソフトウェア開発技術研究部	03-740-5666	
31	上妻健一郎	熊本テクノポリス財団　電子応用機械技術研究所	096-286-3300	話題提供者
32	河本　重遠	日立エンジニアリング　システムエンジニアリング部	0294-53-2211	
33	古閑　幸一	ヒラタ・ソフトウェア・テクノロジー	096-344-2611	話題提供者
34	小前　俊哉	三菱電機コントロールソフトウェア　技術統括部開発課	078-651-7361	
35	坂本　　治	情報処理振興事業協会　シグマシステム開発本部　事務システム開発室	03-255-0421	
36	桜井　麻里	ＳＲＡ　環境開発部	03-234-2611	話題提供者
37	佐藤　勝雄	東電ソフトウェア　技術開発部	03-592-1312	
38	佐原　　伸	ＳＲＡ　環境開発部	03-234-2611	話題提供者
39	鐘　　友良	富士ゼロックス情報システム　技術推進室	03-378-8010	
40	杉田　義明	ＳＲＡ　教育企画部	03-239-5473	
41	関口　純一	日本電子計算　官公システム事業部システム部　開発課	03-668-7261	話題提供者
42	高橋　哲也	神戸製鋼所　電子技術センター　システム制御研究室	078-991-5611	
43	田中　正則	ＳＲＡ　開発推進室	03-234-2611	
44	棚田　真司	ＳＲＡ　環境開発部	03-234-2611	ツール展示
45	辻本　恵一	情報処理振興事業協会　シグマシステム開発本部　技術システム開発室	03-255-0421	

第４回　実践的ソフトウェア開発環境に関する集中討論　参加者名簿（最終版）

No	氏　名	所　属	TEL	備考
46	中田　修二	情報処理振興事業協会　シグマシステム開発本部　事務システム開発室	03-255-0421	
47	新田　稔	ＳＲＡ　ソフトウェア工学研究所	06-344-2611	ツール展示
48	野見山和則	三井銀ソフトウェアサービス　システムエンジニアリング事業部	03-440-6286	
49	野村　敏次	日本電子計算　情報システム事業部システム部	03-668-6171	
50	濱田　勉	ＮＴＴ九州ＳＥ総合センタ　技術開発室	096-321-3611	
51	浜野　剛至	ＳＲＡ　関西支社	06-344-2611	
52	林　香	ＳＲＡ　環境開発部	03-234-2611	座長
53	広瀬　隆夫	日本電子計算　情報機器事業部営業部	03-668-6171	
54	藤岡　卓	三菱電機　情報電子研究所　システムソフトウェア開発部	0467-44-9084	
55	藤野　晃延	富士ゼロックス情報システム　技術推進室	03-378-8010	
56	古田とおる	富士通ビーエスシー　商品企画開発部	03-544-0506	
57	本田　克巳	横河ヒューレットパッカード　ＣＳＯプロジェクト統括室	03-331-6111	話題提供者
58	丸山　伸一	日本電子計算　開発本部ＡＩ開発部	03-668-6171	
59	三浦あさ子	ＳＲＡ　関西支社	06-344-2611	
60	柳瀬　健一	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	話題提供者
61	山口　郁生	電力計算センター　横須賀事業部	0486-56-7595	
62	楊　啓廷	ＳＲＡ　システム開発第２部	03-234-2622	
63	吉本　伸彦	日本経済新聞社　システム本部	03-242-9925	
64	渡辺　雄一	電力計算センター　大手町事業部	03-215-6481	
65	海尻　賢二	信州大学　工学部情報工学科	0262-26-4101	招待
66	篠田　陽一	東京工業大学　工学部情報工学科	03-726-1111	招待
67	方　学芬	静岡大学　工学部情報知識工学科	0534-71-1171	招待
68	安間　文彦	富士通エフアイピー（第１回環境ＷＳ実行委員長）	03-481-4334	招待
69	鐘　錫昌	中国国家科学技術委員会　中国軟件技術開発中心		招待
70	朱　三元	上海電子計算機軟件技術開発中心		招待
71	何　克清	武漢大学　軟件工程研究所　軟件工程実験室		招待
72	中島千代子	ソフトウェア技術者協会　事務局	02-234-9455	事務局
73	野中　哲	日本電気　マイクロ波衛星通信事業部	045-932-1111	
74	北野　義明	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
75	山口　敬介	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
76	安達　久人	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
77	岡本　隆一	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
78	大熊千佳子	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
79	魚田　昌孝	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
80	原田　二郎	ケーシーエス　ソフトウェア事業部　ソフトウェア技術部	078-391-8291	会場／記録係
81	馬場　始三	大阪大学　工学部	06-877-5111	会場／記録係
82	澤田　宜己	大阪大学　工学部	06-877-5111	会場／記録係
83	久下　真司	大阪大学　工学部	06-877-5111	会場／記録係
84				
85				
86				
87				
88				
89				
90				

第4回　ＳＥＡ環境ワークショップ

セッション1　ＨｙｐｅｒＭｅｄｉａのソフトウェア開発への応用を探る

チェアマン　：熊谷　章　（ＰＦＵ）
プレゼンテータ：本田　克巳　（ＹＨＰ）
：佐原　伸　（ＳＲＡ）
：藤野　晃延（ＦＸＩＳ）
レポータ　：久下　真司　（阪大）

1. はじめに

熊谷（ＰＦＵ）：　１人３０分ぐらい喋って貰って、その後、ディスカッションしたいと思います。それでは、まず最初の本田さん、お願いします。

2.　プレゼンテーション

2.1 電脳机の概念と展開

本田（ＹＨＰ）：　神戸製鋼さんと１年前から一応電脳机という名前をつけて、ちょっと変わったものを共同開発ということでやらしていただきました。これは、スモールトークを使ったハイパーテキスト・メディアシステムをつくろうと１年前にベクトラというパソコンの上で仕事を始めましてそれなりの成果はできてきたのですが、途中でアップルのハイパーカードが発表されましてずいぶん先を越されたと思いましてそのときはショックだったのですが、よく考えてみますといろんなところが違います。そういうところも含めましてご紹介させていただきます。

今、わたしのＹＨＰの方では、ＥＷＳの方にＸｗｉｎｄｏｗで移植しましてスモールトークからＣに書き直していますが、Ｘウィジットというオブジェクトオリエントの部品で使ったものを作っています。あとで展示の方でご覧いただけると思います。そういうわけで、今はこれを複合技術情報システムといっていますが、ネットワーク環境のもとでどういうような環境というのが、エンジニアにとって必要かと考えるのがねらいであります。ただ一番初めは、「ＡＩ利用のソフトウエア開発環境」というのがテーマでした。ただソフトウエア環境について何か作りたいというのが初めでした。ご承知の通りソフトウエア環境というのは、いろんな人がいろんなところでお金をかけてやっていまして例えばシグマ計画とか、相当お金をかけて４、５人のチームではとてもまともなものができないということで途中で変えましてペーパーレスオフィスみたいなものをやろうというのがこの目的です。私たちの考え方がポジションが違うと思いますのでそれをご紹介したいと思います。

１つの認識といいますのは、われわれ会社といいますか製造業がお客様多いんですが、電機にしましても自動車にしましてもとりあえず製造大国ということで日本が非常に生産能力がいい。ところがソフトウエアを初めとしまして研究というところでは開発力が非常に弱い。例えば、ＴＱＣですとかカンバン方式ですとかジャストインタイムですとかそういうものを改良するメソドロジーというのが製造業の改良にはあるのですが、開発力を強化する、特にソフトウエアですね、それに関しては、メソドロジーがない。唯一の方法は精神論的に開発力を強化していくしかない。それは、どうも違うのではないかというのがもともとのスタートポイントでした。最終的には、それを使ってわれわれのねらいとするところでは、エンジニアリング環境でのハイパーメディアというのがどういうインパクトを与えるかというところを考えていきたいということでやってきました。

そういうことは実は企業のトップマネージメントの方にも非常に興味をもっていただきまして、特にソフトウエアだけがそういう事をいえると思うのです。一般的に生産性をあげる、開発効率をあげるというのは、一応２つの方法があります。１つは、個々のエンジニアの生産性をあげる、効率をあげる。もう１つは、今の仕事は特にソフトウエアはそうですけども、グループでしている。そうするとそのグループの中で、重複した作業はやめて、なるべく、個々の持っているノウハウをうまくつかう。いわゆる共同の作業を効率よくするには、どうしたら良いかと２つあります。とくに後者の方にいま非常に注目されつつあると思います。例えば、ある例ですけども、日経メカニカルの記事ですが、これにちょっとかいてありますように営業部門とか製造部門とかほとんど、コン

ビュータを使って仕事をしているのです。たとえば、経理の伝票ですとか給与計算を紙でやって伝票を渡している所というのはないわけです。そういう共同で情報を蓄積して共同で情報を利用する態勢がコンビュータを使ってできているのです。営業の人とか製造の人が自分の紙をおこしてそれをファイリングしている人はいないわけです。とくに、ソフトウエアを始めとして開発環境の人というのは、未だに紙でしか仕事をしていない。特に紙も使っていない人も結構いる。そういう意味でこれはどう考えてみてもおかしいじゃないかというのがもともとの初めです。

私たちは、もともと、ＷＳを使っているところですが、最近ＷＳを使って少しましな環境を使いたいということでお客さんも工場を立てるときは、ＬＡＮのケーブルを張ると言うことはどこでもやっている。あるいはＷＳを１人１台というのはまだないですが、２、３人に１台全体で３０－４０台導入している会社も結構現実にあります。ところが、それで何が変わったかといいますと環境としては、変わっていません。例えば、ＬＡＮケーブルをひいて何をしているかといえば、ときたまファイル転送をしてちょっと興味のある人がメールをおくると、フロッピーでわたしてもいいんですけど、わざわざ使い方のわからないＦＴＰを使ってマニュアルを見るのに３時間ということもありますけども、そういうことをやる。また、１人１台ＷＳを使っているからといって何をやっているかというと私たちの会社だからかもしれませんが、ほとんどやっぱりＣＡＤなんですね。ネットワークされた環境の良さは、なにもかわっていない。逆にいうと非常に性能のいいパソコンが入ったに過ぎないのです。そういう意味では、ちょっと間違っているのではないかというのが本音です。

われわれは、１つ仮定をおいたのですが、もちろんクリエイティブな仕事というのもありますが、かなり多くの開発の作業というのは、かなりの組織的に効率があげられると考える。いわゆる、製造業がＴＱＣとかカンバン方式を使ったように必ず効率をあげられるという前提を考えました。つまり、例えば製造業は、材料があって在庫があって製品がある。この流れを、どれくらい効率よくするかということをＴＱＣとかカンバン方式とかでやっていたわけなのですけれども、それと同じように開発という作業はソフトウエアを含めてなにか材料がなければエンジニアは、出力が出せない。例えばその材料というのは、新聞であり論文であり社内報告であり実験でありほかのプログラムである。それから在庫もなければならない。これは今のところ、紙ファイルとか本棚とかキャビネットとかそういうもの、あるいは、頭の中が一番多い。で、いろんな仕事がありますけども、エンジニアの最終的な出力というのは、成果物ということでは、新しい報告書であり機械屋さんであれば、図面でありソフトウエアやさんでは、ソースプログラムです。この流れをどういう風にうまく、いわゆるツールとかコンピュータの環境をサポートできるかを考える。

われわれも良く、テクニカルＯＡとか言っていますが、ＤＴＰとかワープロとか、あるいは、ＣＡＳＥの一部もそうかもしれませんが、全体の作業のごく一部しかサポートしていないのです。ワープロとかＤＴＰは清書するというところしか、光ファイルは、ここのところのごく一部のいわゆる検索的な仕事をしている情報の在庫にしか役に立たない。そういう意味で、これでは将来そういう環境でいいソフトウエアができるとは、ちょっと思えない。もう少し具体的にいいますと、たとえば、入力情報のことを考えましても、メディアの質が違うのです。コンピュータでは、扱い易いのは、テキスト情報とかグラフィック情報ですから、ワープロを使えばいいということでやっていますけれども、実はほとんどのエンジニアは、ソフト屋も含めて、一番多くの情報は、イメージ情報、画像情報。例えば、手書きの議事録であり、論文であり、専門誌なのです。もちろん、新聞とか雑誌とかがすべて電子出版されるのが前提であれば、また違うかも知れませんが、現在は、そうではありません。たとえば、「私の会社では、議事録は全部ワープロでやっています。」というところがあるのですが、ワープロでも誰か手書きのコメントをいれた、私が興味を持ったからここでアンダーラインを引いたら、もうこの情報は、イメージでしか扱えないわけです。そういうわけで、イメージはたいへん重要であるにも関わらずコンピュータでまともにイメージを扱える方法がなかなかなかった。この点が、大きな問題だろうと認識した。

我々の材料、在庫製品の流れを具体的にある人のデータをとったところなのですけども、ソフトウエアの開発のプロジェクトリーダーの例です。開発計画を、決めたりするのですがそのとき彼がどのくらい入力があるか材料があるかを月当りでみますと、日経コンピュータ、日経ビジネス、マーケティングニュース等をページにして合計１０００－２０００ページを見ています。それが入力です。情報の在庫は、彼の場合ですと片袖の机と本棚

２つ持っています。ファイルの厚さにして３．３ｍ、ページ数にして１３０００ページ。ただ、非常におもしろいのは、彼が作った議事録ですとか、報告書ですとか工程図ですとかいわゆるオリジナルの情報は、その内の４０ｃｍにしかない。残りは、誰かが作った議事録ですとか仕様書のコピーである。いわゆる重複した情報つまりネットワークされた環境ではいらないものが圧倒的に多いのです。何がメリットがあるかというと１万ページのコピーが節約できます。いま、コピー代が１枚１０円ですから１０万円の節約になります。そういう話ではない。つまり、１万ページの重複した情報を集めるためにファイリングしてメンテナンスをしてアップデートをするという作業をエンジニアの貴重な時間を使ってやっているわけです。本当は、必要でないかも知れない作業をやっているわけです。それと成果物、彼は開発リーダーですからプロジェクトを決めます。従って、あるプロジェクトの検討報告書を作ったり、仕様書を作ったりします。これが合計だいたい７０ページくらいの仕事ですが１．５人月と見ています。だいたい４０ページを月に作成をしています。基本設計をして詳細設計をしてこれも3.5人月ですから、月４０ページです。ということで、だいたい月当り５０ページ以上新しい情報を出せない。多分他のエンジニアもそうだと思います。もっとクリエィティブな仕事をしている人はもっと少ないです。たぶん、月２０ページとか１０ページとかそういうものです。所が月５０ページということは年間６００ページ、その人が年収６００万の人でしたら、１ページ１万円かかることになるわけです。そうやって作った情報がどれくらい他の人に流通しているかというとそれは、非常に少ないです。

そういう意味でまだまだいろんな意味で改良できるところがあるのではないか、ということでこのプロジェクトを始めたときに一応３つの点を決めました。１つは、１人１台を前提として、すぐには無理かも知れませんがいわゆる紙をなくす方法でいく。我々が調査したのですが、例えばこのように合理化するとファイリングの共有化なんて事をします。１つの課でいままで個人で持っていたファイルを共有しよう。しかし、余り遠くにあると見に行かないでまた自分でファイルを作ってしまう。つまりある情報を取り出すのに１分以上かかると必ずまた紙のコピーを作ってしまう。とにかく１分以内で必ずどんな情報でも出せなくなければならない。それを考えると、稼動率を無視してとにかくＷＳを１人１台持たなくてはいけない。それでないと文化は変えられない。もう１つ、さっき話しましたイメージの情報が非常に多いです。ですから、Ａ４の大きさのイメージが絶対扱えないと話にならない。これが、もう一つの重要なポイントです。

それからそのとき手書きでやるかあるいは、ワープロみたいのでやるかはいろいろ議論がありました。我々も実は、１番最初に手書きでいれますとそれを平仮名認識しましてそれを漢字変換してやるのをやりましたが見事に失敗しました。それは、平仮名をいれますといま、認識効率がかなりいいものですからそれでいれますと９９％位認識できるはずが、実際やってみますと６０％ぐらいしか読めない。どうしてかといろいろ調べてみますと今の認識では筆順で認識している。４８文字中誰でも１文字か２文字筆順を間違えて覚えている。そのために認識できない。これは、日商のワープロ試験問題なのです。これは、２級のクラスの試験問題です。時間は、３０分です。専用のワープロで、２級ですから社内ではトレーナーになる位能力を持つ人が３０分かけてもいいということです。

仕事の中でこれが、まだ使えるという状況ではない。アメリカのように小学生や中学生がテストでキーボードを使う状態でしたら、期待が持てるかも知れませんが、少なくとも１０年は、イメージを使わなければならないというのが我々の結論です。イメージベースのものをするというのは決まったのですが、もう一つの大きな問題がありました。日頃利用している情報は、キーワード検索をしていません。そういう意味で別の方法でやる上で探したのがハイパーテキストです。ハイパーテキストは、ＸｅｒｏｘのＰＡＲＣの研究所でＴｅｄＮｅｌｓｏｎが、１９６０年代からいろいろやっていましてアメリカでは非常に普及している考え方です。考え方は、簡単ですが、かなり重要なインパクトがあったと思います。それを使うことに依って情報の検索を誰でも使えるようにしたというのがねらいでした。

ハイパーテキストの定義は、従来のワープロの文書が、情報を探すのに頭からみていかなければならなっかった。その意味で、リニアーなテキストであるといっています。それに対していわゆる電子ファイルシステムですとキャビネットがあって引出しがあってホルダーがある。これは、いわゆるハイアラキーなテキストということになっています。それに対してハイパーテキストは、ネットワークのリンクが自由にできる。例えば、１つのカードの

中のハイパーテキストという言葉がわからないとしますと、現在、人間はどういう風にするかといいますと注をつけてハイパーテキストについては何頁を見なさいと書いてあります。このような情報の検索をしています。

ハイパーテキストでは、電子的にリンクというエリアを決めましてそこの中でマウスのボタンをおす。すると、それに関連したものがでてくる。それだけで物事を検索していこうという方法がハイパーテキストです。最近では、画像ですとか、音声ですとかもできるのでハイパーメディアと呼んでいます。アメリカでは、ハイパーカードなんかで商用になっています。これは、Ｍａｃ上でやった、アメリカの本田の修理工用のカタログです。アメリカの修理工はレベルがあまり高くありませんから、マニュアルもよく引けない。例えば、エアクリーナーの修理をしなさいとするとガスケットについての情報が知りたいとすれば、ここでボタンを押せば良いという方法で情報をやりとりする。この方法を、特定のものでなくてふだんソフトウエア開発環境で使っている文書でやろうとするためにＡ４のイメージを扱えなくては、絶対駄目だということになります。

そういうことでノウハウの共有化とかネットワーク環境に依って仕事をするという習慣をつけようとした。ハードウエア屋のエンジニアとエンジニアのあいだのオフィシャルのノウハウの交換の方法は回路図しかない。ハイパーテキストを使いますとここのところにボタンがついていて手書きの設計メモがあるかも知れません。ＩＣにつかいますと、データ仕様書がでてくる。よってネットワーク上でこれができるとほかの人が検索できる。特に重要なことですが、熟練のエンジニアでしたら、回路図を作るときには、何を調べて何を考えてという考え方のプロセスを、例えば、新人のエンジニアがみることができるのです。つまり、ノウハウを個人の上でなくネットワーク上のＷＳにもデータとしてためていこう、このようなことを改良した環境を与えれば何か文化が変わるかも知れません。

ソフトウエアの問題は、いろいろありますが、我々が一番認識したのは、文書作成とかの更新・再利用が非常に難しい点です。それをどうするかといいますとイメージでいれるとなんでも入り、非常に安易です。それを共有するということにメリットをもたせたいと思いました。ソフトウエアの関係では、何かプロジェクトがありますとプロジェクトごとにタイトルをマウスビックしますと仕様書がでてきます。モジュールをマウスビックすると内部仕様書がでてきます。入力というところをマウスビックしますとモジュールのダイアグラムがでてきます。そうすることによって環境として効率をあげようとする。机の上のごちゃごちゃは少なくともウインドウの中だけにしようとする。そうすることによってうまくすると在宅勤務もできますし、リゾート地で仕事をすることもできます。これが目的です。

我々が提供するのは、ネットワークされた環境です。ペーパーレスといいますが、これは間違いだと思います。オリジナルのペーパーはいるが、コピーして他人に配ったりすることはしなくて良くなる。よって、コピーをなくすオフィス、コピーレスオフィスを考えたい。私どもの会社は、３０００人位の会社ですが、ＨＰが世界最大の電子メールを使っています。社員が、３０００人中２０００人がＨＰデスクという電子メールシステムを使っています。例えばペーパーレスオフィスというと電子メールをやれと、確かにドキュメントを作っていると東京、アメリカ、ドイツにおくるのに非常に簡単です。ｃｃ：でおくれば、１秒か２秒で全部送れます。我々の会社では、２、３人に１台しかないですから、端末に人が座っていれば、１分以上待ってば、情報はとれないわけです。そうすると紙のコピーをとるしかない。我々のコンピュータは、そんなに容量がありませんから、メールですと２、３カ月位メールを貯めているともう消してくださいときます。それにより１年前２年前のものを電子的に保存しておくのはできない。１０人の人に送るのは簡単ですが１０人が１０人ともプリンターに出すので電子メールをやったために紙が非常に増えたというのは我々の会社でも経験があります。

これは、間違いです。我々が作ろうとしているのは、センターのネットワークファイルにいれておいてそれをみんなで参照する。たとえば、議事録では、議事録を送るのではなくて個人的な掲示板を作っておいて議事録ができましたというメッセージだけやります。そこで、議事録にリンクをつけておけば、良いわけです。もし、議事録を見たければ、そこをボタンで押せば、本物の議事録がでてくるわけです。ある調査では、エンジニアの持っているノウハウの内５％だけしか公式の情報として会社の中で使われていない。それを解決する唯一の方法は、各個人は分散しているＷＳのうえにあたかも自分のファイリングシステムのように紙のファイルの代わりにファイルする。こうすると他の人からも使うことは、できるわけです。

これは、どこまで見せるかというプライバシの問題はありますが、少なくとも会社でやっている仕事は、なんらかの基準を設けて他の人が見れるようにするのはエチケットではないかと思っています。そういう環境でのみ、新しい変革ができる。会社でワープロを使っているとか、コンピュータを使っているとかいっている人でワープロを良く使っている人でも週に１回必ず使っている人は、かなり使っているほうです。ところが、コピーを週に１回しか使わない、コピーを月に１回しか使わない人はいないわけです。そういう意味で、コンピュータ化されていないコピーとか紙ファイルとかファックスとか非常に大きな作業を改良できる余地が残っている。コンピュータをあまりインテリジェントなものでなくてそういうくだらないものから使っていこうというのが我々のアプローチです。

よって我々は、ハイパーメディアをつかって今までよりも多くの人に情報を使ってもらいたいということ、イメージやマルチメディアが、ＷＳとかを使ってフィジブルになってきたそういうことによって文化を変えるチャンスがでてくるのではないか。このままでは、高級なパソコンの代わりとしてしかＥＷＳは、つかえない。それをなんとか早く変えなければならないのではないかと思っています。

佐原（ＳＲＡ）：今の話の中で、日本語入力が手書きの方が遅いというところがありましたが、私がＸＥＲＯＸのＪ－ＳＴＡＲで実験をしたら、かなり入力速度が遅いですが、手書きより２倍は早いという結果がでて、いまＰＣの日本語入力ではもっと、Ｊ－ＳＴＡＲでなれている人では、３倍４倍早いのではないかと思います。唯一速いのは、読めないなぐり書きはさすがに手の方が速い。

それから、コピーレスという話ですが、先ほど理由の中では我々がコピーをとるのはたぶん画面が小さいので全体でみたいときではコピーをとってしまうのであって、先ほどの理由とは違うと思います。

本田：入力にはいろんな意見がありまして最終的には、ワープロ入力みたいなことになると思います。ただ、我々がやったのは、ワープロが好きな人が全体の人数の１／１０位の人数で使うシステムではない。全員が使うという条件でないと紙はなくせない。一部の人が、紙でしかわからないのではいけない。ワープロは、百万台か２百万台世の中にでています。とても大型機の入力とかの日本語変換に比べれば、５万か１０万のワープロの方がずっと賢いです。

しかし、ワープロには、入門教室があるけれども、コピーにはそのようなものはありません。教室があるうちは、まだまだ自然な入力であるとはいえません。パーセンテージは、違ってくると思いますが、必ずおちこぼれてくる人がいるだろうと思います。僕らがターゲットにしている人は、ドシロウトの人です。それでないと紙はなくならないと思っています。

それでコピーをとる理由ですが、確かに今は画面もとてもきついです。ようやくＷＳでなんとかＡ４が１枚見れる程度です。１つはコピーの使い方は変わると思います。今のコピーのとり方というのは、コピーを保存をするためです。これからは、コピーは保存しなくていいのです。少なくともそのコピーのファイリングはしなくていいです。

2.2 ＨｙｐｅｒＣａｒｄとソフトウェア開発

佐原（ＳＲＡ）：ハイパーメディアと関係があるのかどうかわかりませんが、ハイパーカードとソフトウエア開発ということで話をするようにという指定がありました。

ハイパーカードというのは、アップルのマックの上で動くハイパーメディアと呼べるかどうかは別として一応ハイパーメディア的な機能を持ったソフトウエアです。それを普通に使っている分には、個人用の情報管理システムですが、ソフトウエア開発で使えるのではないか。たとえば、プロトタイピングツールとして使えると思います。これからは、実験してみた成果の報告です。

こういう新しいことをやるときは御利益がないといけないので、生産性というあまり好きでない言葉を使いますが、生産性がいいよという話をして、それから銀行のＡＴＭのモデルをハイパーカードでやってみたのとＣＡＳＥツールを使ったものと比較です。

いままでいろんなソフトウエアを作ってきた中の、あまり仕事でない部分のソフトウエアのいろんな生産性とかの比較をしてみました。それでいちばん左の欄が開発年度の次がＯＳ、次が言語、次が開発の時間です。次は、機能です。次は、注釈をのぞいたステップ数をかいています。次の欄が、どういうアプリケイションかということです。いちばん右は、開発の時の方法論もしくは知っていた方法論です。ここでおわかりいただけるように、まず住所録のシステムをいくつか作っていますので、比較するとハイパーカードで作ったものは３時間でできて機能が６あってステップ数が３２２で、いちばん上のフ

ォートランで作ったものは、開発時間６００時間というのは実は、１日１０時間として６０日、３カ月かかって３０００行のプログラムを作って住所録を作ったのと比べると、ここでまず２００倍違います。で、機能で6倍も違うと１２００倍も開発の生産性が違う。これはすごい。住所録に関しては、ＯＳのちがいは、あんまり影響しないと思いますので言語としての違いがでてきているということです。

もう１つ銀行のＡＴＭの問題がありますが、これは、ハイパーカードを使ってそのモデルを作ったものの画面であります。これを、ＣＡＳＥツールでもって、そのモデル、データフロー図っていうか、これ制御フロー図なんですが一応リアルタイムのシステムを記述できるもので書いたものです。ここにいたるまでに苦戦しました。その１つの理由は、私がこのアプリケーションに対して知識がなかったということです。これを私が、専門家にみせたら「こんなものはなかなかいいですね。私のところでは、実はいきなりコーディングなんですよ」といわれました。それで実際問題が起こっているのです。

だから、このモデル化というのはやはり必要で、素人がやるとイメージがないのでどこかデータフローとか制御フローを忘れていて、ああしようこうしようで頻繁に線が消えたりついたりする。それですごい時間がかかってしまう。それからもう一つは、作るときにある程度の機能を全部考えないとシステムモデルはできない。

それに対して、ハイパーカードで作った方のモデルは、何をやったかというと最初は残高確認ボタンというオブジェクトがあるだけ、カードがあるだけ、あと全体もカードなんですが、そのカードの方が一応残高の口座の管理をしているということになっています。その３つのオブジェクトしかなくて、まずはじめて一応動いたということで徐々にほかのボタンとか機能をつけてきた。そうすると実は、これを作ったときには、３０分ぐらいで最初の状態ではちゃんと動いて、「こういうものが表示されていいなぁ」と確認できるわけです。

ですから、プロトタイピングの中では、ＣＡＳＥツールよりもこちらの方がよいかなという気がしています。で、こちらの方がいいのは、先ほどの方があまりにも具体的すぎて目に見えるモデルだから作り易い。だけど、そもそもあれでいいのかという、実は銀行にいけば見えるモデルだからこれをまず簡単に作ってみてそういうシステムのモデルを作るというのはいいのですが、ホントにこれで銀行で現金をおろそすときのもっと簡単にお金を下ろせる方法を考えるためには、これは逆に具体的すぎて良くなくて、この場合だけこういうＣＡＳＥツールでもってモデルをじっくり考えるというのは意味があるのかなぁという気がしています。ハイパーカードを使うとなんとプロトタイプ作りが楽なのでしょうというお話でした。

臼井（ＪＩＰ）：ハイパートークのいいところをいろいろご紹介いただけたのですが、使っておられてここがまずいなというところを教えてください。

佐原：メモリの制約があるので、アプリケーションとしては問題がないのですが、開発環境としては、エディタがあるだけですが、多少構造化的なエディタがあるだけですが、それがアプリケーションのウインドウが開いているときにこの辺にダイアログボックスというものがあって、こことエディタの間の両方を見ながらプログラムを作っていくということが今の開発環境ではできない。エディタを閉じないと戻れないことになります。これは、明らかにメモリの制約でそのような仕様に対しているとしか思えません。

こういった開発環境が比較的貧弱で、ただ貧弱といっても言語自体は結構しっかりしているので、かつ３２０行で結構凝った住所録ができるという位で、個人ツールとしてはそれくらいでいいだろう。

困ったのは、結構複雑なスタックがいっぱいできてきてそれを分析しようとするときに、ＳｍａｌｌＴａｌｋのようないろいろなリファレンス的な機能がない。どこに何が書いてあるかというのは、実にさっきのように出してみないとどこにあるのかわからない。だから開発するための環境がちょっと苦しい。

しかし、アプリケーションとしてはほとんど何も問題がないというのはそこです。あとで、音楽が２分くらい流れるアプリケーションを見せますが、２分の音楽のためにだいたい２Ｍバイト位メモリを食います。だからそういう今のメディアは、ハイパーメディア対応になっていないのでそういうマルチメディア的な情報を入れるとディスクをがばっと食うという副作用がある。

海尻（信州大）：私どもでは、学生に絵をかかすと結構時間がかかるのですが、その辺はどうなっているのですか。

佐原：トランプの絵はもともとどっかのスタックにあったのをコピーしてきたというものです。これは、影をつ

けるのはボタンの機能の中に影をつけるというのがありますからシャドウボックスとかをクリックするだけです。

海尻：例えば私どもでは、いまちょっと文法の関係をやっているので、構文図を学生が作っているのですが、１つの構文の構文図を書くのに１時間以上かかっています。

佐原：今のハイパーカードは、いわゆるお絵描き機能を拡張した機能があるのですが図形をかく機能は、別のツールがあってさっきのＣＡＳＥツールについていた設計図を書くツールでつくってそれからこちらにコピーする。つまり図形を画くプログラムで図形として書いといてデータもそれで管理しているから、たとえば、Ｍａｃの上ではそれは絵として切りとって来れますから図形として書いて絵として切りとってくると結構速くできます。

2.3 ＣＳＣＷの技術動向

藤野晃延（ＦＸＩＳ）：今日、前の人がいろいろ関連することを全部いってくれたのでだいたいもう話さなくていいのではないかと思っています。

というわけで、私がしゃべることはこういうことです。ＣＳＣＷというのはどういうことかといいますと、主な内容としてはこの４つだと思います。だいたいＣＳＣＷという言葉があまりまだメジャーでないのでなんとか覚えていただきたいということとそれが実際、どういうことを考えないといけないかのかということだけを話したいと思います。

私のお送りした紙の中にだいたい書いてあります。けれども、アメリカのアクティビティとしてはだいたいこんな感じです。この中で大切なのは、いままでやっていることはアナロジーなのですけども、ＣＳＣＷでやろうとすることは、たとえば車のことだけを考えていくのではなくて、車が実際動くハイウェイのシステムをどうやって設計すればいいのかを考えるということを、コンピュータでも必要なのでないか。つまりただ単にパーソナルなＷＳを１人に１台与えて、その中に使えるソフトがこんなにあるのかといっているのでは駄目で、そこら辺にハイウェイのようなものをそろそろ必要になっているのではないかというのが、大きな流れです。

そのヒストリカルなアクティビティは、次のようなものがあります。まず、ＤＥＣとかＭＩＴとかが最初にワークショップみたいなのを開いていわゆるチームでや、ある程度まとまったグループでのクリエィティブな活動でサポートしようではないかとか、コンピュータでサポートするにはどうしたら良いかというものです。

今年のリフレクションとしては、今年もＣＳＣＷ’８８というのがあったのですが、それは’８６とあまりかわったものはなかったようですがリフレクションとしては、コンピュータサポートのＣＳＣＷがないというのが実感です。で、ＣＳＣＷがある程度考えられるようになったというのもさきほど本田さんがおっしゃってましたが、コンピュータのソフトにしろハードにしろ非常にぜいたくに使えるようになってきたのが１つの流れです。それと同時に、個人的レベルでやるっていうことだけでなく、協調してやるサポートは、いままで全然考えられてないからきっちり考えよう。そうしないといわゆるドラスティックなレベルでの変革みたいなものが生まれてこないのではないかという危機感がある。

ちょっと古いのですが、ＣＳＣＷのアクティビティを紹介します。これは、ＣＳＣＷ’８６のやつですけどもだいたいアクティビティは、４つあります。ワークグループ間をサポートするのと、組織的なコミュニケーションをどうサポートするかということと、コラボレーションデータベース。

先ほどハイパーメディアでのこの形のサポートは、最近注目されている。あとは、リアルタイムミーティングのサポートがあります。その中では、’８８ではＰＡＲＣの中では残念ながら新しくでていません。残念ながらＸＥＲＯＸでは、研究のレベルでは、特にＰＡＲＣではいま、ＳＵＮにだいたいマシンが１対ずつあって、その自分たちの環境、いままでＭｅｓａ、Ｍｅｓａの上のＣｅｄａｒっていう環境をＳＵＮのＵＮＩＸベースの環境に移していて、とても研究を続けている状態ではありません。そのためにｏｕｔｐｕｔとしては、めぼしいものがなかったというのが残念です。

で、最近ではハイパーメディア関連では非常に新しいものがでてきています。ＣＳＣＷの中で考えつくサポートするエリアっていうものはここにある４つです。？とか、プロジェクトマネージメントとか、グループによるデザインアクティビティとか、デザイン時のディシジョンメイキングです。

あと大きいのは、エレクトリックメールのサポートです。とくに、メールを論理的な意味であるサブジェクトについてのメールのヒストリーをとっておくとか、あるメールをあるサブジェクトに大してスクーリングするとかっていうのがずいぶん要求されてまた効果的であるということをデザインだとかディスカッションとかに有効であることがわかっています。

あとは、ＣＳＣＷっていう実現している現時点でのベースとなっているテクノロジーっていうのは、データベースです。それも従来考えられているデータベースでなくてハイパーメディアみたいに様々な種類のメディアっていうかオブジェクトがネットワークでつながっていて、リンクがダイナミックに張れるようになっている。それとリニアーでない文章とハイパーメディアそれとあとは、ネットワークサービス全体っていうのが考えられてきた。

とくに、アメリカのキャンパスネットからきているのですけれども、教育なんかで使った場合、ネットワークで１０００人が使った場合、ネットワークサービスっていうものはどうあるべきかという問題がでてきました。そこら辺では、実はうちでもかなり問題で多いときには、１日に１０台とかＳＵＮが新しく追加されるのです。するとうちは、ＸＥＲＯＸマシンをもともと使っていてそれでたぶん５００台とか１０００台とかあるビル内でつながっていたりしますので、そこにＵＮＩＸという、それまでＸＮＳでやっていた所に、ＵＮＩＸという新しい異端児がやってきたのでさまざまなトラブルがあって、とくにスループットが悪くなったりとかいう問題があって、やはり自分たちも苦労している。

一応、テクニカルベースとしてさきほどの本田さんの方がハイパーメディアという話で、わりとしっかり話していただいたのですけども、実は我々のところでも先ほどノートカードというのがありましたが一応あれはＬＩＳＰシステム上で動いていたのですが、今はＪ－ＳＴＡＲ上で動いているＶＩＥＷＣＡＲＤというものが動いていましてやはりそれを先ほど本田さんがおしゃったように、ある程度アプリケーション上に使えるようにと、たとえばオーサリングシステムですとかそれから新しいスタイルのデータベースとかがその上でできるのではないかと現時点いろいろとやっています。残念ながら、ＸＮＳのプロトコルではＸですとかＴＣＰ／ＩＰ上でのＸですとかＳＵＮなんかが、サポートしているような形でのネットワークオリエンティッドな協調すなわちプロセス間通信をネットワーク間でやるようなことを十分にサポートしていなくてネットワークオリエンティッドな形で共有できるデータベースも個人環境と同じようなイメージで実現するというのが残念ながらできていません。今後の問題点はおそらくその辺になってくると思います。で、実際に最近ハイパーテキストという形で、うちはスモールトークを出していることもあってその上で、かなりいろいろなものを作っていましてあとのプレゼンテーションでご覧いただけると思います。これなんかもそうですね。これなんかもオブジェクト指向プログラミングを勉強するＣＡＩシステムともいえるし、オブジェクト指向プログラミングのプログラムをビジュアルにしたという形です。実は、ＡＲＫというアメリカのＸＥＲＯＸＰＡＲＣにいた人が研究していたものを、外部仕様だけいただいてインプリしなおしたという形のものです。一応、Ｍａｃ　の上で動いていますのでアニメーションができるというので、オブジェクト指向プログラミングを理解するオーサリングシステムとしてみていただけるとかなりおもしろいと思います。

同様に、ペトリネットのアブストラクトマシンを青木純がインプリしていまして、これもハイパーテキストライクでして各モードで自分のもう一つ別のペトリネットをバインディングできるというハイパーテキストでいうリンクをつけるというものです。

あと最近の動きとしましては、ＣＳＣＷというものは、より一般の人がコンピュータというものをより使いやすいメディアとして利用していく形で、これが必ずしもペーパーレスになるかどうかというのも良くわかりませんが、そういう方向でトライしてみる必要があるというのは確かです。

最近では新しいユーザインターフェイスとして提案されています、ＳＵＮとＡＴＴから出ています、Ｏｐｅｎ Ｌｏｏｋというものでも一応ハイパーテキストライクなユーザインターフェイスというものをそのような形でやはり持っています。残念ながら、ＣＳＣＷっていうのはまだどういうアクティビティをすればいいのかが、アプリケーションのレベルまでまだ完全に達しているとは言い切れません。少なくとも、これからはコンピュータを使う人をパーソナルなレベルでサポートするのでなくてグループで使うように、サポートしなくてはいけないのではないかということがたぶん、今年くらいからこの方向での研究がようやくスタートさせることができるかも知れないなというのでありますが、残念ながら明確なテクニカルディレクションみたいなものがはっきりできないのが残念です。

久保（富士通）：ＣＳＣＷのインクルージョンというのを提示してください。これは、何を書いたのですか。
藤野：いま現時点でターゲットにあげたもの
久保：だれが
藤野：ＣＳＣＷ’８６というコンファレンスの中でカテ

ゴライズです。
久保：するとエンブリカルスタディをやっていう人は結構多い。
藤野：そうです。なぜかというとＭＩＴとかＰＡＲＣとかあと大学関係のところがかなりあって、
久保：そういうのは、どの様な人がやっているのですか。
藤野：エデュケーション関係です。
久保：２番目の環境というのは環境を作るということですか、使おうとすることですか。
藤野：残念ながら環境を作る人だけだとか使うだけだというのははっきりしていなくて、自分たちでつくりながら使いながらたとえば、自分たちでこういうところをコンピュータでサポートできる。たとえば、このような会議をしているときにその会議に対するサポートというのをどういうかたちでやってくれたらいいのかということ。
久保：それでなくて、だまされていいかということです。
藤野：かなりだまされる必要性がある。たとえば、これもそうです。これは、ＸＥＲＯＸのＰＡＲＣでやっているものですが、会議をしているメモをリアルタイムにヒストリーをとっていく。なおかつ互いの意見交換みたいなものをつまり、コピーとかするのでなくリアルタイムでサポートしようとするものです。

岸田（ＳＲＡ）：ＣＳＣＷでは、うちの栗原君が一度出たので、彼に聞けば会議の席上でどうだったかは、わかると思います。

ＣＳＣＷの第１回の会議を主催したのはＭＣＣです。ＭＣＣでは、ソフトウエア環境の上流工程、とくに設計とかです。グループとして、設計のエンバイアラメントのグループが、ある設計のプロセスのグループと設計のインフォメーションのグループというようなグループに分かれていて、エンブリカルスタディのところでは、たとえば実験的なデザイナーのミーティングルームみたいなもので壁に大きなエレクトリックボードをおいてみんなはそれぞれ端末でもっていろいろやる。それをモニタールームでモニターしてモニターしたやつを、あとで分析するという研究をしています。

それから心理学の専門家の人が、ＭＣＣはご承知の通りいろいろなスポンサー企業から成り立っていますが、スポンサー企業を訪ねて各スポンサーのなかのスーパーデザイナーという人にインタビューして頭の中でどういうことをデザイン的に考えているか、それをまた一部のものにどうやって伝達するかを考えている。それは、プロセスとかインフォメーションの問題です。それからエンバイアラメントのグループでは、そういうふうにデザインが共同作業として進として、どういうエンバイアラメントがあればいいかというものをプロトタイプをつくって、ＭＣＣの中でも使い、ある程度使えるようになったプロトタイプは、スポンサー企業にばらまいて使ってもらう。

久保：もう１つ、最後のインクルージョンはなんですか。
藤野：これは、さっきも言ったように、まずコグニティブなアクティビティの研究があるわけです。人間のディシジョンメイキングをコンピュータでサポートするときにはどう言ったことが必要か。ましては、それがコーポレィティブなものの時には、どうするかをサポートする。

熊谷（ＰＦＵ）：ハイパーメディアをつかうとＣＳＣＷがうまくいくのではないかというものですね。

3. 討論

野中（ＮＥＣ）：たぶんＹＨＰの方に答えてもらうのがいいのではないかと思いますけども、まずハイパーテキストは今の技術の段階で見ると、例えば専門家のデザイナーが一生懸命構図を作ってそれでプレゼンテーションツールみたいなものを作れば見栄えがすごくいいものができそうなきがしますが、もし各個人が勝手なものをバラバラいれて、作った人がリンクを張って、みんなが勝手に作ってそういうものを構築していくと、だんだんグチャグチャになっていったりとかメンテが大変だったりすると思いますが、そのへんは何か実際の経験でお話があれば、お聞かせ願いたいです。
本田：おっしゃる通りだと思います。われわれは、電脳机はネットワークバージョンは来月出荷なので、本当の意味でのお客様の声は聞いておりませんが、実際に使っていただこうという話には、必ずそれはあります。それは、次のように考えています。

１つは、今の情報のまとめ方はツール以前に、ＵＮＩＸのシンポジウムでは、しつけの問題としましたが、しつけの問題ですとか会社の仕事のルールの問題です。たとえば、開発プロジェクトがあったときに、どういうドキュメントを作らなければいけないか、どういう構造をしなければいけないか、それは逆に言うとツールをいれるいれないに関わらず、共同作業でやる部分であればルールとして決めておかなければならない。あとでデモン

ストレーションで見ていただければわかりますが、例えばある開発プロジェクトフェニックスというプロジェクトのところには、工程図、議事録、作業依頼書、バグレポート、マニュアルそれが次のサブメニュウにあります。それは、どのプロジェクトは共通で、それをもとにしてやっていかなければならないというのは共同の場合の作業の方法です。

それから、もう１つは個人的なノウハウの問題、これはむしろ強制されたら困るのです。さっき、設計図の話をしましたが、設計図に例えば手書きのノウハウをつけておく、ノウハウをつけるつけないは強制できないのです。で、どういうものにノウハウをつけるか、日経エレクトロニクスのコピーをつけるか、データシートを張りつけるか、これもわかりません。ですが、少なくともアイコンがあったり文字があってボタンがあれば、リンクしているのがわかればブラウジングができるだろう。僕らは、別に汚くても良いただし、誰でも使えることを目的にしています。これは、ツールをつかう問題だけでなくて、会社の情報の管理の方法が決まっているからです。最終的にはユーザが自分で決めることだと思います。

海尻：ハイパーテキストのリンクの問題ですが、リンクがスタティックで要するに、前もってこれにはこれのリンクがあるって言うのはどうなっているのですか。それともう１つ、ついつい手でいれた方が速いのでいれてしまいますが、今の話では、ＷＳ１台どころかファックス１台をいれておかなければいけないという気がしますがどうですか。

本田：今の両方ともそうだと思います。いまのハイパーテキストの１つの限界は、ハイパーテキストは基本的に個人的な情報管理には非常に役立つ。いま、おしゃったようにスタティックです。実際には、そうじゃないものもあります。たとえば、ドキュメントの管理は、ハイパーテキストでもできます。ですけれども、実際にはある設計図面を見たときにこの部品を使っている設計をしている人は、会社の中で誰がいるだろう。これは、本人に聞けない。ですから、今までで言うとデータベース的な検索をしなければならない。

で、我々は電脳机は、オブションでハイパーテキストの中にＵＮＩＸのプロセスを動かすようにしようとしています。１つの使い方は、リレーショナルデータベースを検索しまして、例えばあるものに相当するもののダイナミックなリンク、その時点でデータベースに入っている関連しているものを出して、それをハイパーテキストにリンクする。あともう１つ、あるお客様に話しているのは、リンクの所をｐｒｏｌｏｇを使って推論しようとしています。それは、ユーザインターフェィスとしてみれば、それがダイナミックについていたか、スタティックについていたかは、わかりません。ただ、それに関する情報が出てくる。そういう意味でユーザインターフェィスとしては、見かけ上、ハイパーテキストでやっているように見えるということをやろうとしています。

熊谷：リンクはどういう風になっていますか。例えば、ダイナミックなリンクだとソースとディスティネーションの間に名前があってそれで名前のあいたのを名前空間のリレーションを張らないと一義的に決まりませんね。たとえば、あと具体的にものを出してものとものどうしビジュアルな形で見ていて線で結ぶのかとかリレーションの張り方は具体的にどうされるのですか。

本田：例えば、この場合で言いますと、実際にはこのリレーションには、カードのＩＤがついています。あるエリアで押すとあるカードが出てくる。ところが、この場合にはそれがＳＱＬのコマンドが書いてある。それは、検索時に実行します。そのときにリレーションデータベースのプロセスをやって、次に目次カーソルみたいなやつでそれに該当したリンクが自動的に作るとしています。ですから、質問文的なものは、スタティックなリンクで、それを実行時に実際の物理的なＩＤを検索する。

熊谷：そうすると、内部ではポインターがあるだけと言うことになりますか。

本田：そうです。

熊谷：ネーム空間とかは、ないのですね。

本田：基本的にはそうです。

深瀬（アスキー）：たぶんＵＮＩＸの上でハイパーカードとかを動かすということになりますと問題となることがいくつかありまして、１つは、誰がリレーションを張るかというアクセスのコントロールや、整合性を誰が保証するのか、特にできあがったデータベースとしては、使う側からすればあまりいじることはないのですが、ネットワークで分散していれば、自分用のＶｉｅｗを作りたいとかでいろいろいじるわけです。それをどの様に管理するのですか。いわゆるブラウジングの機能の配慮をしているかが、個人で使うか、プロジェクトで使えるかの分かれ目だと思うのですが

本田：その辺は問題でありまして、真のネットワーク対応のデータベースがあるかというとないわけです。ネットワーク上の１つのところにデータベースマシンをおいて、データベースサーバーを走らせるというのはいくらでもありますが、あくまで分散してできるかという意味ではありません。じつは、電脳机は基本的に言うとファイルシステム、ネットワークファイルシステムですから、その意味ではトランスペアレントでありますが、基本的には、ファイルシステムです。

たとえば、今のセキュリティとかアップデートですとかオーサリングとかはファイルシステムの機能を使っています。たとえば、個人レベル、グループレベル、それ以上のレベルで管理しています。その辺は考え方ですが、１つ例えば、議事録というものを会社で登録するときにそれを誰がオリジナルを削れるかといいますと、やっぱり議事録を発行する責任のある人しか削れないのです。ところが、見ることができるのはグループの人はみてもいいと決められます。そうすると、リンクの張り方とか管理の方法も基本的にそうです。我々はカードと呼んでいますが、それはオーナーがデリートする権利やアップデートする権利があります。ある人は、それに対してリンクをはる。つまり、リードオンリーの権利があります。しかし、今はオープンなネットワークですが、それを将来的には暗号化して、それを耐えられるようなものにしなくてはいけませんが、そのときにはさっきお話したリレーショナルデータベースがＵＮＩＸのネットワーク上である場所に集中し合いますので現在では不可能です。今の段階では、フィージブルなパフォーマンスをえられない。

深瀬：ただ、たとえばカーネギメロンのキャメロットというプロジェクトは、キャンパスワイドに１つのツリー構造をもったファイルシステムでしか持っていないデータベースです。それから、バークレイのグループも分散型のＲＤＢのプロトタイプを動かして実験しています。それが、何年も研究されていて、使えるかどうかというところまで来ていますが、そういうのを組み合わさざるをえないと思うのですが、個人的にはＲＤＢでロックマネージャーを動かしてイエローページのサービスでなんとかするというのを社内でやっているのですが、限界があります。ネットワーク上のプロトコルとして、サービスとして欠落している部分がずいぶんありますから、そこらへんうにＸの中だけでなんとかしようというのは難しいのではないでしょうか。

本田：ぼくもそう思います。ですから逆に言うと、ＲＤＢはオプション機能で本当にそれが必要な人だけが使えばいいとそれ以外の人はスタティックなリンクでやろうと思っています。

熊谷：そういう意味では、個人ベースと言うことになるのですかねぇ。

岸田：最初の質問は、佐原さんに対するものですが、ハイパーカードを使ってアプリケーションの例を作られたわけですが、うかがっていると、ハイパーカードを使ってアプリケーションを作るということは、オブジェクト指向だけなのですか。ぼくは、ハイパーカードをプログラムを使うだけで、ソフトウエアのパラダイム開発では、オブジェクト指向でやるというだけの話なのかという質問です。

あとの話は、３人全員にかかるわけですが、ハイパーメディアはソフトウエア開発環境の中でどうやって使っていくかという話は、結局いままでいくつかの質問の中であったように開発環境の中のデータベースをどうするという話なのか。そうすると、いろいろ問題があると思います。

１つは私は現実にマネージャーですが、マネージャーの観点からすると、非常に古典的なウオーターフォール形のソフトウエア開発があります。あれは、管理の指標としてなんステップできたとか何人月予定していたのなんパーセント終わったかというふうに言っているわけですが、あれは、結局時間軸があって、そこに仕事が１次元的に流れていって、最終的にここで仕事が終わるはずで、今はここまで来ているから７５％終わったというふうな管理の仕方です。それを、若干改良して繰り返し型の開発環境を平行して走らせるという、この線がマルチに走るというだけです。で、ハイパーメディアでやっていくということは、どうも基本的にポインターがあってリンクが張られていくということは、情報はツリー構造にできている。これは、いったい管理の指標として何をするのですか。枝の本数で管理するのですか。マネージメントからするとハイパーメディアの定量的な指標はなんですか。

それから、２番目のもう一つの問題は、開発環境がオブジェクトベースでやはりなんらかのモデルが必要です。モデル無しでやっていくと、結局いま机の上は、ちらっかっているが、それをコンピュータ化して、コンピュータの中でやっていくだけなのでないか。だから、モデル

を決めて、モデルにしたがってみんながやっていくのでなければならないのでないか。

例えばわたしのポジションペーパーに書きましたが、5年くらい前リドルのＪＯＳＥＰＨこれは、単にスペックをどんどんコンピュータにためていくわけですが、項目ごとにキーワードをつけて適当にキーワードでもってリンクすると言うもの。それから、ＭＣＣでやっているｇＩＢＩＳというやつ。これは、設計するということは設計上のいろいろな問題点を解決すると言うことだと考えて、いろいろとみんなが提案してそれぞれの解決方法に対して、賛成反対というリンクを張ってデータベースかしていくという考え方です。このようなモデルをハイパーメディアの仕掛のうえでのせて何かやらないと、たぶん実際やっていけないのでないかと思います。

佐原：今のところ、ハイパーメディアはオブジェクト指向とわりと相性がいいと思います。いろいろなイメージとかテキストを含めて、それをオブジェクトと考えて作っていくとやりやすそうだというところはあります。もっといいのはないというのは、ハイパーメディアにべつにかんけいない、もっと別に方法論を考えるという話です。だから、今のところオブジェクト指向＋今までの方法論をかき集めてやるという段階です。

そのあとの質問ですが、岸田さんと反対でモデルがないと駄目ではという話ですが、設計をやるときに何かモデルがあってそれにしたがってやるというのは、たとえばサッカーでみるとボールを受けると必ずある方向に動き出すというろくなものができないと思いますが。

本田：たぶん２番と３番の答えだと思いますが、ハイパーメディアの定量的な指針は、非常に難しいです。最後の方にありますが、それによるソフトウエアの開発モデルっていうのも絶対に必要とおもいます。ただし、１番最初にスライドでおみせしましたが、僕らがソフトウエア開発のツールを作ろうとしたとき、一番問題になったのは、モデルが会社によって違うのです。まだ、確定的に、このモデルを使えば必ず成功するというモデルはまだわからないわけです。そういう意味で、このモデルを選べば必ず成功しますというものは言えないような気がします。これからやりたいのは、１年か、２年かかると思いますが、実際に使ってもらうとモデリングはどこまでやらなければいかないかということは出てくると思います。僕は、使えば見えてくると思います。

１つ参考になるかも知れないのでスライドをお見せしたいと思います。電子ファイルは僕らの電脳机の比較の対象になるのですが、そのときどうやるかやどうやってまとめるかというのが文章の場合あったのですが、そのときに１番大きな問題点と言うのは構造と言うのは一義的な点です。たとえば、電子ファイルでいいますとノウハウとかドキュメントのまとめ方もそうですが、プロジェクトA、プロジェクトB、プロジェクトCというふうに分けて、その中で、議事録、図面と分けておいてそこをまたプロジェクトAというふうに分けるとかこういうことを決めておかなければいけない。

それは、今のモデリングという問題にもありますが、これはいわゆるツリー構造とかハイアラキーなものでもなんですが、これだけで本当にいいかわからないのです。たとえば、セクレタリーから見たときには、ツリー構造の方がいいかも知れない。で、それはハイパーテキストはいい加減なツールですからなんでもできるのですが、ではなんでもやってやっていいですよと言うとさっき深瀬さんが言われたように混乱するかも知れません。それをどこまでどうやったらいいかというとそれはわからないのでのってくれそうな（だませそうな）お客さんを見つけて一緒にやろうというのが今の段階です。

岸田：佐原さんの意見に、モデルとマニュアルとは違います。つまりこうやったら必ずこうやりなさいというのはそれはマニュアルであってモデルとは言えない。モデルと言うのは、本田さんがおしゃったようにあるものをどう見るかという話です。しかし、ハイパーメディアは一種のインプリメンテーション上のモデルだと思いますが、それを使う側からすればいろいろなモデルとしてみていかなければいけない。そのとき、モデルという概念がないと困るのは再利用ができなくなる。確かに設計と言うのは、ひらめいてやっていきますからその都度、特に精度タイプでないシステムの設計は、かなりひらめきによっていろいろなものがアトランダムに出来てくる。ただしできあがったものがコンピュータの中にはいるときには、ある個人的なフレームワークのどこかに位置づけられているはずであってそれを他人がみたときにそういう話かというような格好でみられないといけないのではないか。コンピュータの中に蓄積されたものを見る眼鏡としてモデルがないと、ごみくずかごが、コンピュータの中にたくさんできて誰も使わないということにしかならないと思います。

佐原：熊谷さんの前にまとめられた資料の中にモデルとかそのたぐいのものは、抽象的なものをやるのが今までの情報処理で、ハイパーメディアはもっと具体的なものをそのまま目でみられるからよいのだという話があったのですが、私もそれに同感でしてモデルという抽象的なものをださない方がいいのではないか。

で、机の上が混乱するのが、そのまま反映されるという言うのはその人の設計者の能力がそのまま反映されると思います。したがって、モデルはいらない。いらないのではなくてある個人の中にいろんなモデルを知っていてそれを自分たちが使えるように引き出してくる。だから、ハイパーメディアだからといって、新しいモデルが、できているわけではありませんが今私たちがやっているのはともかくいろいろなものを作ってみないと何がこれから必要ないのかはわからない。

熊谷：私は、ハイパーメディアはツリーだというのは結構あるのですが、ネットワークだと思います。ツリーは１つの形態であって実際はネットワークでさっき本田さんがおしゃったように、マルチビューで見れることがたぶんいいところであって、ツリーだとビューをちょっと変えても大した変化はないです。サブセットで見れるのしかないのですが、ハイパーメディアはデータとデータの間がぐちゃぐちゃにリンクがはられていまして、リンクに何種類かのいろいろな種類がありまして、自分がそのとき必要なリンクをたとえばＡならＡとかグリーンならグリーンそれだけをつまみ出せというと自分の中心から子供孫とかその指定をすると、たとえば３世代まで引っ張りだせというとグリーンのリンクが３世代だけ取り出してくれるとか、そのところがすごくいいのであって、たぶんツリーは全然違うのではないかと私は基本的に思います。

岸田：ツリーでもネットワークでもいいのですが、つまりウォータフォールとか古典的なモデルではたとえば、見積ステップ数みたいなものを考えるわけです。それに合わせて管理しているわけです。たとえば、１００人月かかる分の９０人月いれたから９０人月分で来ているというふうに大づかみな信用です。では、ハイパーメディアみたいにネットワーク的にものを作っていく、できあがったものは、ネットワークである。すると最初に見積のネットワークをコンプレクシティーメジャーみたいなものがあって、今、ネットワークをまだ非常に単純なものしかできていないから完成度が１０％かなーみたいなそういうことで考えていくのではどうでしょうか。

熊谷：その次に私が言おうとしたのは、ここにこのようにマッピングするからおかしいのであって、この２次元の平面にはハイパーメディアはないのです。

岸田：つまり、このような新しいツールを世の中に持ち込もうという場合には、買ってくれないといけないのです。お金を持っている人はだいたいマネージャーです。ですから、マネージャーに生産性１００倍という言い方はウォータフォールのモデルの考え方でステップ数を時間軸で言うともっと短くなることができますよと言うだけの話です。この様に、マッピングして説明していいのかそれともネットワークでできあがったものに対してなんらかのメジャーをだしてこれだけ複雑なものがたとえば、１人月でできますよというふうな言い方がいいのではないかという単なるアドバイスです。

久保：僕、さっきの１２００、結構気に入っています。１つは、文字の数です。単語の数でもいいかも知れませんが、ただそれは生産性という点からは、結構悪いものではないと思います。やはり、言語が高度になれば１つの言葉に集約できるものが大きくなりますので、同じことをやるのに非常に明快になると思っています。ただ、岸田さんのどの様にマネージメントするかというのは非常にいい問題提議だと思います。答えはありませんが、古典的にはモジュールという概念があります。モジュールの大きさは人間がコントロールできる大きさと言うものがありますが、それに相当する概念でオブジェクトの大きさと言うのがあるのかなと思っています。ただ、オブジェクトをどのように量化するのかというのは、おもしろい問題です。

臼井：だいたいソフトをうまく作っていくとか個人のひらめきをどのようにうまく整理していこうと言うのはパーソナルな話であって、産業的に成功させようとするには管理の問題などがあります。たぶん、今現在あるものは情報がネットワークされていて、関連しあっていて非常に複雑になっているからそのままでは管理できないのでいろいろな制約をもうけているのです。

古い話では、構造化プログラムでは制限することによって正しいものを作りませんかという話をしていたと思います。どのように制約を加えていけばよりよいものができるかということでソフトウエアはやってきたので、

その制約をどんどん取り除いて自由にやりましょうというのはそこからある意味ではバックしようとしています。いろんな人がいろいろな立場でうまく使えるような制限条件と言うのは絶対に決めないとやっぱりカオスの状況に陥ると思います。

佐原：臼井さんは、構造化プログラムを間違った解釈をしていると思います。構造化プログラムと言うのは、データを抽象化してそれを素直にプログラムしようとするのだと思います。

篠田（東工大）：モデルｆｏｒマネージメントのところでディベロップメントｏｆハイパーメディアと書いてあります。これにツリーモデルと書いてありますが、これがいまいち僕には理解できない。マネージメントのモデルとディベロップメントのモデルを分離して考えないといけないのに、あそこでは、いっしょにしている気がしています。佐原さんの発表では１つのプロトタイピングの例としてハイパーメディアがあります。マネージメントのモデルとしては、ハイパーメディアベースのとして開発したとしてもそれはそれでおいといて実際出来てくるもので管理するわけです。

佐原：ハイパーカードでマニュアルを作れといわれましたが、それをマニュアル５０ページで抑えてねといわれてもそれはマシンの中にはいているからページと言うもので表現できません。

篠田：ハイパーテキストといっていますが、僕はそんなにハイパーでないと思っています。商用になっているハイパーメディアがありますが、ハイパーハイパーといっていますが、ちゃんと構造を持ったものが扱われています。たとえば、ＫＭＳでは階層構造です。ハイパーカードでもフォルダーありますがあれは、ディレクトリーに移っているだけです。だから、あまりハイパーでないと思うのですが。

佐原：構造が複雑でも見ればわかるものってあります。ハイパーカードなんか見ていていいのは、結構スタックの作り方が複雑ですがみるとそれなりにわかるという利点があります。今までのソフトウエアのようにごちゃごちゃに作るとできたものもごちゃごちゃしていますが、ハイパーメディアを使うと中身はごちゃごちゃしているが、外はすっきりしているものができる気がするのですが。

篠田：今まで見てきたハイパーメディアのカード間のつながり方とかスタックの行き来と言うのはちゃんとしたものです。あまり複雑になっていない。あまりハイパーでないという意見です。

藤野：今の意見は、非常に正しいと思います。さっきＣＳＣＷのところでハイパーテキストは構造化されたデータだといったと思いますが、篠田さんのおしゃっているのは、Ｃで言うところのストラクチャーのポインターみたいなもので、つまりデザインしている人には、ストレートフォワードに見えなければ良くなくて、そこら辺のモデルというフレームワークみたいなものがきちんとなくてはならない。

熊谷：フレームがあってフレームがあった上で作っていかないとすっきりしたいいものができないという話でしょ。

篠田：そうじゃないです。それもありますが、もとからフレームを作ってそれに基づいて作っていかないといけないのでなくて、ネットワーク構造でもいいんです。ただ、それを見やすい形に写映する。

熊谷：わかりました。しかし、それはいままでやってきた計算機の使われ方です。ハイパーメディア系がハイパーだといえる点は、ものが最初で、それを意味ある人がみてどういう意味かをリンクか何かで自由にはっていくのです。それを自分で最初に他人にわかるようにフレームを作ってもいいのですが、ベースは自分でめちゃくちゃなものにリンクをはっていって固まりを作ってそれを最後に今のようなデータベースのフレームにするのです。

岸田：それは反対で、佐原さんのあげた例で言うと僕がいったものは定石とかそういったものでなくて碁盤と言うのは１９ｘ１９のマス目でするものですよと言うのが僕のいったモデルで、今熊谷さんの意見は碁盤の上で将棋をさしてもいいですよという話になります。それでは、めちゃくちゃになってしまいます。

本田：電脳机でお客様と今と同じ話し合いをするのですが、先ほどこの図をおみせしました。これは１つの設計者のモデルですが、回路図しか残っていません。回路図というのはこの様な手書きの設計メモがあるかもしれないが、これは設計者の電気回路を作る上でのモデルです。あるいは、ソフトウエアのドキュメントっていうのはこういうふうになるかも知れません。プロジェクトがあって仕様書があってモジュールダイアグラムがあってデータフローダイアグラムがある。これもそのソフトウエア開発のモデルです。僕らはそういった意味でハイパーメディアと言うのは自由にできるという言い方があるし、

めちゃくちゃになるという言い方もあると思いますが自由にもできるというところも１ついっておきます。そこで、どういう制限をするかでこの文書体形を決めるというのは会社のしくみです。ツールだけでは決まりません。機械を設計するときのノウハウの伝達の方法、これもそのエンジニアとかで決まっている問題でツールだけでは決まりません。その時で、まず１つ自由度がなくては困るだろう。だから、きまりは決めなければいけないものですが、僕が決めるものではありません。使う人が決めるものです。

野村（ＩＣＳ）：ハイパーカードを使っていてちょっと感じるのですが、自分考えているパターンをそのまま表現できる。今までは、機械に制限された形でデータを整理していた。しかし、ハイパーカードを使ってくると自分にあったものでデータを整理できる。できたものを判断する人が自分と同じ発想パターンをしていれば、非常に使いやすいものができるが、波長の違う人がみると非常に使いにくくなる。だから、ある程度人間を研究してそれに合わせたやり方をしないと駄目であります。結局いいたいことは、モデルで管理しようとすれば、人間に立ち帰らなければなりません。ハイパーメディアも管理できるだろうと思います。

渡辺（電力計算センタ）：ハイパーメディアで２、３わからないところがあります。１つは、マンマシンインターフェイスのことにも関するのですが、ハイパーメディアと言うのは、ユーザにとって全然ハイパーでないと思います。さっきわざわざ設計仕様書だとか要求書だとかそれぞれのテキストがネットワークをはるだとかリレーションをはるだとかの話をしていましたが、あれを何で１つのビューにみせないのかなと思います。だからそれらをシートと呼べば、シートにアトリビュートをもたせておいて単一のビューとしてみせといて、その中のいくつかのアトリビュートのついているところだけを引き出してきてたとえばコンパイルするだとかがちゃんとできていいのではないのでしょうか。そういうことによってかなり現状の家内製手工業的なソフトウエアの生産としては寄与できるんではないかと私個人としては思っています。

というのは、私個人としては経験的な判断からするとまだそう言うところに関しては、経験的にモデリングができているからその中のボトルネックになっているところに関して、そういうふうなマンマシンインターフェイスを改善することによって生産性に寄与できるのではないかと思っています。具体的にはどういうことかといいますと、非常に不満はありますが、ＣＯＢＯＬという言語を想定しましてたとえば、ＣＯＢＯＬのｃｏｐｙｌｉｂで普通書いているところを何であれをダイレクトにハイパーテキストのたとえば、普通の設計仕様書にかくレイアウトの絵を何であそこに書いてコンパイラに通らないのかというのが非常に不満なのですが、あそこをプレコンパイルしてその情報ををＣＯＢＯＬのコードに落としてやって出すという仕掛けが、ちゃんとシートにアトリビュートがついていてひとつの単一ビューとして見れれば良い。そういう意味での理論的な背景とかもう少し泥臭い話になるのかわかりませんが、そういうところの議論ができていないからハイパーテキストはまだちょっとそのもの自体に議論をかける必要があると思います。

本田：おっしゃる通りだと思います。そういう意味で僕らの電脳机のソフトウエア開発と言うことでやってきたのですが、電脳机の購入を真剣に考えているお客様はソフトウエア開発の人は一人もいません。電気屋さんと機械屋さんです。そういう意味では、ソフトウエア開発にはこういうものをいれる前にやらなければいけないことが非常に多いので当分売るのは恐いなと僕は思っています。だから、それまでにやらないといけないメソドロジーとかモデリングとかがあると思います。ただ一つの見方と言うのは、ソフトウエアにしましても、どこにしましても、いまハイパーメディアという見方とハイパーメディアを使うことによって再利用性と言うところがポイントだと思います。それはあくまでもハイパーメディアの環境を提供するに過ぎない。そういう意味では当たっていると思います。

青木（ＦＸＩＳ）：昔からハイパーカードとかがでたときに、このシステムを話していたのですが、そのときにぐちゃぐちゃになってネットワークになるのかツリーになるのかいろいろありましたけれども、そういうものを使って来ている段階でどういうことがでてきたかをちょっと話してみます。

たとえば僕なんかは、グラフをいっぱいかいてネットをはる。そうしたときに、自分がいろいろなところにネットをはって、複雑になってわからなくなってしまったときにいざ今度は何をし出すかといいますと、トポロジカルな処理をしだします。つまり隣接行列とかを作ったりして、この情報量をどれだけ複雑なんだろうかという

ことをはかる測定装置を自分で作っていきます。

ハイパーカードは、リンク付けばかりが先行していまして、それをはかる道具が全然整備されていませんので、はかる道具に重点をおいていきますと、また違ったビューが見えてきますので、非常に最近そういうことで楽しいです。ですから、ハイパーテキストはぜひみなさんやっていただきたいなと思います。

本田：今のお話は、たいへんおもしろいですが、こないだのシンポジュームでもそういう話がでたのですが、1つは電子化するということでそういうことは自動的にできる可能性があります。いまリンクと言うのもスタティックとお話しましたけれども、我々が考えていますのはリンクをたくさんはっていきますと文書でたくさんできます。ですけども、ＵＮＩＸの上でやっていますから、たとえばそのリンクをつかったかどうかはコンピュータで判断できます。そうするとたとえば、リンクははったけれども1年間使わなかったリンクがあったそうすると自動的に削る。使用頻度に応じて優先順位をあげていくこともできます。そういうことをコンピュータでやっていくことによってそういうことも可能になってくるという気がしています。そういう意味でそれがほんとに必須の機能なのか、あるいは単なる遊びなのか。だから、一人でも多くの人に使ってほしいと思います。

久保：本田さんの話ですが、本田さんのやっていることに僕はまったく異議を唱えるつもりはまったくありませんが、神戸製鋼の方に発言願いたいのですが、だいたいにおいて、つかうほうは作る方にだまされ聞いていると言うことがありますよね。本田さんの方はある意味で世直しのようなことを考えておられる。作る方がそんなことをいったって、使う方がその気にならなければいけないと思うのですがどの程度神戸製鋼さん　・・・（テープの切れ目）・・・その時にいろいろな障害があると思います。いろいろな障害をできるだけ具体的にあげていただきたい。それがどの様に克服されていくのかの見通しを、その中で技術で解決されていければどちらかといえばやさしいとおもいます。テクノロジーで解決できる要素とそうでない要素とを織りまぜながら神戸製鋼の立場で、あまりＨＰに遠慮しないでしゃべっていただきたいと思います。

高橋（神戸製鋼）：この共同研究開発、お互いだまし合ってやったという点も多いにあると思いますが、わたしも神戸製鋼の中では研究所の人間で社内ではだます方の立場ではないかなと思っていますが、大きなブレイクスルーというのは今もちょっと問題になっていますが、誰かをだまして使わせることじゃないかというふうに僕自身は思っています。

いま社内で、ある程度プロトタイプ的なものができまして、いまはいかに社内でだまして使わせようかなと、だまして使ってもらいますとハイパーメディアに関わらず、まず現場のひとをだまして使ってもらう。そうしますと、はじめはなんだかんだいってましても、ああ言う機能はないかとか逆に向こうから問いかけがきてまたいいものになっていくというような繰り返しだったと思います。そういう意味でいまはだましているところですといっておこうかなというところです。

久保：だます人間の見つけ方を。いろいろな方法があると思います、上から責める、下から責めるあるいは、自分の回りからせめていくとか、のそこいらを

高橋：これがいいやり方かどうかわかりませんが、私どものやり方はまず身近な今までの仕事でおつきあいのある人からまずだましていこうかなというのが、実際問題としてはいいのではないかなと思っているのですが

杉田（ＳＲＡ）：ハイパーリンク関しては、さきほどの野村さんの意見にまったく同じでありまして人間のことをもっと研究しないといけない。で、そのためにはいま行なわれているいろんな議論については、だいたいスタティックなものをみんなに押しつけようとしています。仕事と言うのは、ネットワークベースで仕事をしてそれで、マルチユーザで使っていくのだと、それぞれみんなには仕事のスタイルというのがあるわけですからそれぞれの人たちが、与えられているのと違う状態でさらにリンクをはっていくスタイルをかんがえていかなくてはいけません。一応、言葉で言うとマルチユーザのあたりのインプリというねらいがないとこれは成功しないと思います。

佐原：みなさんは固定的なリンクがハイパーテキストだと考えているようですが、ハイパーテキストの定義はそもそもリンクがあってそこにスクリプトみたいのがあって動的なリンクができるような仕掛けが当然あるのです。だから、固定的リンクと言うのは、まだハイパーテキストとして完全にできていないのです。

中野（阪大）：ネットワークとかツリーとかハイパーメ

ディアの話があったのですが、もともとわたしは、ソフトウエアでなくてグラフ理論だとかが専門ですが、青木さんの話に関連して言うと、インターメディアのはなしをいろいろ聞いていますと、どの様に表示しようかという話が非常に多いのです。そこをものすごく苦労なさって規模が大きくなると全然わけのわからないものになります。それは、みなさん一生懸命どのように表示しようかと考えておられますが、グラフ理論の世界では、いろいろたくさんやっていて、ある分野ではＣＡＤのレイアウトなんかはまさにその世界です。だから、わたしは表示の世界ではいろいろやりたいなと思っています。今までの過去のグラフ理論とかのをもう少し見てくださいねということがいいたい。

佐原：私は、ソフトウエアやる前にグラフ理論をやっていたのですが、だから当然、ＣＡＤの世界とこっちも全部しっていますが、まだグラフ理論もあんまり解決していないと思いますが。

中野：解決はしていません。しかし、いろいろな成果物があるのだからそれを使ったらどうか。ただ、もう１つ私の意見は卒業研究で１つやってくれといっているのですが、単にネットワークとしてとらえてはいけないのであって、ツリーが中に埋まってますよと、それともうひとつクロスリンクです。ツリーでなくてフォーレストというのですが、木がたくさん埋まってますよと、いくつかの話があってそのあいだにクロスリンクがあるという見方でかけばいいのではないか。それが、違う見方すると違うフォーレストになりますよと、そういう話にしとかないと、ネットワークをどう描きますかということをいくら議論しても、仕方がないのではという気がします。

で、アプローチとしてはツリーがありますという世界をやって表示するとわりかししっかりするのではないかという気がします。で、整理をよくしている人は、きっとツリーがあるのだろうと、整理の悪い人はツリーは１本かそこらで、クロスリンクばかりで酒を飲んだら消えてしまうという気がします。

渡辺：中野先生にちょっとお伺いしたいのですが、そうしたときにある程度、たとえば基本的な設計とかですと要求提議、設計、サブ設計、コーディング、テストとかがありまして、いまいったツリーとかフォーレストとかだいたいこういうふうに作ればいいんだよといったときに、そこを固めちゃってばハイパーメディアとかの使い方とかがある程度パターン化されていって、それをうまいアルゴリズムに乗せれば、うまく使えそうだというふうに解釈してよろしいですか。

中野：基本的にはそうですけども、ハイパーメディアが狙っていることは、クロスリンクの話でクロスリンクを今度は木の部分だとみなせば、ネットワークになります。というのが、ハイパーメディアの一番いいたいことなのでしょ。水平発想だとかというのが使えるのじゃないですか。

渡辺：その時にはもっと、さっき僕がいったようなマンマシンインターフェイスのビューと言うのをつかって、そういうことを意識しないでさせるということをすれば、アルゴリズム的にそういうことがもうほぼできているからというふうに解釈してよろしでしょうか。

中野：アルゴリズム的には、ある程度できていると思います。木だとかクロスリンクだとかでもたかだか１０００ｘ１０００だとか２０００ｘ２０００の世界でそれをやろうと言うのが無理じゃないかとおもいますが。

FB佐原：最後に中野先生がおしゃったことがよくわからない。

中野：ある程度あるんじゃないですかという話？たかだか１０００ｘ１０００のところにウインドウをはりまくってももうわからないのじゃないですか。解像度の話です。

4. さいごに

本田：たいへん楽しい議論だったのですが、もっと気楽につかってしまってください。

佐原：ハイパーカードを使っているとモデルとかなによりも楽しい。楽しいというのが原動力で、それでいろんな人がモデルみたいなものを考えないとモデルというものはでてこないと思います。やっぱり使ってください。

藤野：とにかく使って新しい議論とか使い方とかをどんどん探っていくのがいいと思います。

熊谷：私が、座長として一言いいますと我々がいままで議論したのはハイパーテキストがベースだと思うのです。具体的にいいますとハイパーメディアというのはアニメーションだとか音だとかが入ってくると全部頭が吹っ飛ぶんです。ＰＣだとかＥＷＳでちょこまかと動くのはいいのですが、たぶん本質を表現しているとは私は思わない。ほんとは、ＮＥＸＴなんかはそうかもしれませんが、あのようなハイパーメディア系のマシンがでてきたときにあっと驚いてしまうのですが、そのようなマシンを作

るようにしたいとおもいます。

5. チィアパーソン・サマリー

（株）ＰＦＵ　熊谷　章

このセッションは、最近話題を集めている HyperCard や HyperText をまとめてHyper系システムとして特集したものです。Hyper系システムは、何を狙いとして、どのようなアプローチで開発されており、一体どのようなもので、何に役に立つのか、という辺りに関して議論するのが主目的である。議論のとっかかりとして、最初に三人の講師からお話を戴いた。

最初の本田さんからのHyperTextに関するコンパクトにまとまった内容とソフトウェア開発における品質管理と電脳机の関係に関した面白いお話が聞けた。本田さんは、実際に作成したシステムを会場に持ち込んで実演も行う熱演であった。彼の結論は、HyperMediaを使用することによりソフトウェア開発の技術を変革させ、文化を変えようというものである。

二番目の佐原さんは、HyperCardとＣＡＳＥツールを実際のソフトウェア開発に応用したらどうなったかという事例研究とそれの分析をして戴いた。結論としては、HyperCardを使用したら、従来のプログラミング方法と比較して１２００倍の生産性の向上があった、というものである。彼の結論は、一部の事務分野を除けばHyper系のシステムを用いたシステム開発が可能であるから、便利で生産性の高いこのパラダイムを使わない手はない、というものだ。

三番目の藤野さんの話は、コンピュータを用いたグループ作業における協調支援システムとそのときにおけるHyperText系システムとの関連についてである。彼の話は、ＣＳＣＷ’８６のプロシーディングに基づき、ＣＳＣＷのアクティビティとしては、ワークグループ間のサポート、組織的なコミュニケーションのやり方、コラボレーションなデータベース、リアルタイムミーティングのサポートの四つがある、というものだ。結論は、ＣＳＣＷは一人一台のＷＳが実現された後で本格的に議論されるものでこれからが重要なテーマになり、HyperMediaはそのとき重要な役割を果たすという。

基本的な考え方とその開発技術、その応用事例と分析、これから将来の方向と申し分のない講師陣に恵まれたというのが実感であった。

発表の後の議論の内容を以下にまとめておき、次回以降への指針の一助としたい。最初の議論は、HyperText中のリレーションのリンクに関するものだ。個人が勝手にリンクを張れば、リレーションはメチャクチャにならないか、予めリンクを張っているスタティックなものと手で自由に張るダイナミックなリンクとそれらのリレーションの検索が話題となった。ＲＤＢ，ＳＱＬ，Ｐｒｏｌｏｇ等を用いた解決策が一つの解として出された。

二番目の議論は、マルチビューとブラウジング機能に関するものだ。分散環境下においてプロジェクトで実際に運用するためには、色々な問題があり現在ＵＮＩＸにある機能の組合せだけでは限界があり、うまくいかないのではないか。

三番目の議論は、HyperCardはオブジェクト指向のパラダイムでソフトウェアを開発するというだけのことか、というものだ。これに対しては、オブジェクト指向と従来の方法論をかき集めて適用すべきである、という解答があった。

四番目は、HyperMediaのようにツリーやネットワーク構造になった場合マネージメントを考えると定量的な指標は何になるか、そしてそのときの設計モデルは何になるか、というものだ。この議論は伯仲した。古典的なソフトウェア開発に心身とも浸りきった人々と流行に乗って軽やかに本質に迫っている人々の対決となった。前者が岸田、久保、臼井さんであり、後者が佐原、熊谷さんであった。真理はどちらにあるかは殆ど自明である。このテーマは、今後も続けなければソフトウェア開発は変革できないと感じた。

五番目は、HyperMediaはハイパーでないという議論、これも中々奥が深い議論になった。総じて、方法論やモデルが存在しないとソフトウェア開発はカオスな状況に陥ってしまうが、Hyperはこの状況を解決する所か、場合によってはカオス状態を助長させるに過ぎない、という意見が強かった。これに対して、リレーションの定量的測定とその分析、シートをコンパイルしてＣＯＢＯＬ言語に変換させる方法、グラフ理論の応用、新しいHyperMediaマシンの必要性等が考えられた。

以上で議論が終了したが、後で考えてみればHyper-Mediaは我々にもっとも基本的な問題を提示していることが分かる。それは、今までのソフトウェア開発は既に知られていることを対象としていかに早く効率的に作成するか、しか考えていなかった点である。知っている人々がモデルや方法論を用いて知らざる人を導く方法である。これはこれでよいが、ソフトウェア開発にはもう一

つ別の類いの者があることが分かった。それは、様々な情報を介して新しいノウハウや方法論やモデルを作り出すという行動である。これは、特に設計や分析のプロセスで重要なもので人間の知的な活動に深く結びついていると考えられる。今まで我々は前者しかコンピュータでうまく扱うことが出来なかったが、HyperMediaのパラダイムの獲得によって後者も実現できる環境を手にいれた。そういう意味で、HyperMediaをソフトウェア開発にいかに使用するかは我々ソフトウェア技術者の知性そのものが問われている、と考えられる。

6. 感想

6.1 ＨｙｐｅｒＭｅｄｉａの発表に関する感想

熊谷章（ＰＦＵ）：
本田：分かりやすい平易な発表で良かった。イメージスキャナーがキーになるあたり、まったく同感である。もう少し電子ファイルとの相違を強調した方がよい。
佐原：１２００倍の生産性は快適である。モデルや方法論を用いた方がかったるいには同感。ＨｙｐｅｒＣａｒｄとＳｔＰの比較が面白い。現実はＨｙｐｅｒで、RevolutionはＳｔＰという表現は当たっている。
藤野：ＣＳＣＷの主要研究テーマは分かったが、個々の技術をどのように研究開発しているか、という当りが知りたい。

本田克己（ＹＨＰ）：ObjectOrientedProgrammingとＨｙｐｅｒＭｅｄｉａを区別して議論しなければならないのではないか！？

岡村博（三菱）：
電脳机のお話：特になし
ＨｙｐｅｒＣａｒｄのお話：マンマシン関係のプロトタイプツールとして使うなら良いが、本格的なソフト開発にはどうか？
ＣＳＣＷのお話：何の話か全く分からなかった。

野村行憲（岩手電子計算）：
ＣＳＣＷとＨｙｐｅｒＭｅｄｉａの関係は？
ＣＳＣＷ？
ＨｙｐｅｒＭｅｄｉａはアプリケーションの操作マニュアルに応用すると良いものができそう。

盛田政敏（ＫＣＳ）：本田さん、しっかりプレゼンテーションしてくれてありがとう。“組織的に考える”という点は同感です。我々もいろいろアプローチしていますが、組織内の情報構造がなかなかうまく整理できないで困っています。

藤野（ＦＸＩＳ）：
本田さん：なかなか、良く考えられていると思います。ベースに哲学があって、非常に面白いし、実験としていろいろな情報を得て、更にこのシステムを拡張していって欲しいと思います。
佐原さん：Ｏｂｊｅｃｔ（Ｍｅｄｉａでも可）間のリンクには、静的と動的なものがあると思う。例えば、関係情報と必要なアクションが典型的だ。この意味でアクションをしめす「言語」をユーザが意識するとしないに拘らず、そこに存在する事が要求される。ＨｙｐｅｒＣａｒｄとＨｙｐｅｒＴｅｘｔの違いは、前者にはＨｙｐｅｒｔａｌｋという、制限はされているが、アクションを示す「言語」が密接している処が、その強みの原因とみられる。

臼井義美（ＪＩＰ）：
本田さん：全員が使える文書ファイルは素敵だけれど、文書が増えてきたときの管理・検索方法はどうなっているのだろう。

佐藤勝雄（東電ソフトウェア）：ＨｙｐｅｒＴｅｘｔの実例として大変興味があった。特に実務のコピー量との対比が面白かった。電子メイルを行うとＤｉｓｃがいっぱいになるというのは、当社でも生じている問題点であった。

今別府芳暢（ＮＴＴ）：
電脳机：コピーレスオフィスが実現すればすばらしい。開発業務における共用化を総合的に解決できたら良い。
ハイパーカード：プロトタイピングに使ってみたい。
ＣＳＣＷ：？？

濱田勉（ＮＴＴ）：
電脳机：面白いけど、文化を変えるには大変でしょう。今のディスプレイでは受け入れられない。共有ファイルは信用していない。自分流の重み付け。
ハイパーカード：帰ってから住所録を作ろう。

ＣＳＣＷ：初めて聞きました。勉強がたらん。

上妻健一郎（電応研）：

1．電脳机の話で出てきた“コピーレス”の概念には共感を覚える。現在、文献のコピーをファイルしているが、徐々にスペースを食い、整理も大変になっている。これをＨｙｐｅｒＣａｒｄで何とかと考えているところである。イメージで文書を扱うことでユーザの間口をひろげることは出来るが、情報量の爆発的増大と情報の再生産の難しさがネックになると思う。

2．ＨｙｐｅｒＣａｒｄの開発環境が貧弱という話が佐原さんからあったが、それはあくまでも標準としての機能が弱いからで、サードパーティからかなりの数のツールが出ていることを付加しておく。それよりも、処理系の遅さ、カードサイズの固定されていること、マルチカードが実現されていないことが問題で、一応それらの点についての改善は推奨されているので期待している。

安間文彦（富士通）：ネットワークを有効に活用する技術は必要だと思う。その解決策の１つとしてハイパーメディアを利用して情報の共有化を図る環境を提供しようということに興味がある。こういう環境の管理技術がどういう状況か知りたい。

野中哲（日本電気）：入力された情報にリンクをはっていく手間はどうなのか？

人が勝手に入力したものに対して、

いつのまにかリンクがはれてしまうのじゃないか？

イメージから意味が抽出するような技術に対する展望は？電脳机をパソコンに乗せたことの意義は？　ｏｒ動機？ＣＳＣＷというのは、別にソフトウェア開発に限った話ではないわけですね。

井川裕基（ＪＩＰ）：

ＹＨＰ：イメージ取扱のコスト／パフォーマンス（リソース消費量とデータの　　質）に疑問。実物を見ないと。

ＨｙｐｅｒＴａｌｋ：これも動いているものを見たいですね。

辻本恵一（シグマ本部）：

本田氏：ソフトウェア開発の生産性向上＝共同作業上の問題が重要というアプローチは共感。ノウハウの共有のための情報の構成管理を行う意義は大きい。ペーパレス化については、これまでの習慣から紙の上でものを考える（でしか考えられない）のをやめられるか？共同作業の支援として、それ以外のサポートはコンピュータ上で出来るか？　メイルは有効か？

ＣＳＣＷ：共同作業サポートできるならすばらしい。夢。

坂本治（シグマ本部）：ＣＳＣＷとＨｙｐｅｒＭｅｄｉａの関連がよく見えない。

高橋哲也（神戸製鋼所）：ＹＨＰ：現状分析等が具体的で、提案内容も具体的であった。

ＳＲＡ佐原氏：実際のＨｙｐｅｒＴａｌｋのプログラム例があって面白かった。

Ｘｅｒｏｘ：実際のシステムの具体例をもっと見せて欲しかった。

飯塚正樹（横河ＨＰ）：ハイパーカードの目でみながらプロトタイピングができるという機能の重要性を改めて認識した。ＵＮｉＸ上の開発をするようになると、机の上で頭の中だけで考えてしまうことが多くなり、実際の動きを追うのがむずかしくなる。

三浦あさ子（ＳＲＡ）：ミーティングの議事録は声で入るというのはどうだろう。チームでの開発にＨｙｐｅｒＭｅｄｉａが役にたつかもしれないということが分かった。リンクがあまり複雑化すると、そこから情報を引き出してまとめる（？）、解析する、作業がとてもむつかしくなるのではないか。

林　香（ＳＲＡ）：

本田：電脳机：めざしている所は面白い。実現された内容の実用化はまだまだか？イメージ・スキャナーの実用性は？

田中正則（ＳＲＡ）：本田さん：コピーレスの考え方には賛成です。ただ、データの共同利用を行うときに管理はだれがやるか等、討論の時に聞いてみたいと思います。

佐原さん：生産性１２００倍はすごい。ただ、大規模システムの開発に適応することは可能か？

藤野さん：よくわからなかった。

野見山和則（三井銀ソフトウェアサービス）：ＨｙｐｅｒＣａｒｄについてますます興味がわいてきました。

佐藤千明（長野県協同電算）：ＨｙｐｅｒＣａｒｄによるプロトタイピングは、要求仕様の検証という点では有効であろう。但し、それを実機上でどう実現するかについては橋渡しがあるのでしょうか。ユーザに夢を見させておいて、できあがったシステムとそれとの間に大きなギャップが出るのは問題だ。

杉田義明（ＳＲＡ）：ネットワーク・ペーパレスの適用を考えることで、企業の文化を変えたいとの話がＹＨＰの本田さんからあった。その例として議事録をペーパレス化することのメリットを強調されていた。私は、この件に関して、企業文化を変えるパワーとしては弱いように思う。むしろ、会議そのものをなくしてしまうような試み、それらを支援するソフトウェア等の開発や実験が必要だと感じた。

岸田孝一（ＳＲＡ）：「ＨｙｐｅｒＭｅｄｉａをどう使うか」について、もうそろそろ体系的に整理した議論が必要かなという気がする。

藤岡　卓（三菱電機）：（初参加の第１セッションなので）独特の雰囲気に戸惑っています。ＨｙｐｅｒＭｅｄｉａについては、まだまだこれからだと思います。

小前俊哉（三菱電機コントロールソフトウェア）：同じ事を考えていますが、私は手書きよりワープロ（エディタ）の方が早いと思います。でも、うちではこんな事あまりやらしてくれないんだヨナー！

青木　淳（ＦＸＩＳ）：ＨｙｐｅｒＭｅｄｉａを実際に使用するためには、現在のＷＳのアーキテクチュアではダメで、ハードウェアからしっかり考え直さないといけないようだ！

海尻賢二（信州大）：
ＨｙｐｅｒＭｅｄｉａ（Ｔｅｘｔ）のリンクはｓｔａｔｉｃ？
Ａｕｔｈｏｒとｒｅａｄｅｒの別が長短両面
イメージの入力も１人１台？

久下真司（阪大）：本田さんの電脳机には、非常に興味をもちました。特に、コピーレスという所に。

山口都生（電力計算センター）：ＨｙｐｅｒＭｅｄｉａの有効性がよく分かった。ただ、ＯＨＰ（？）がちょっと見にくかったのが残念。

中田修二（シグマシステム開発本部）：個々のプレゼンテーション自身は理解しやすいものだと思います。これからこの方面の進歩が急速に進歩すると期待できると思っています。

澤田宣己（阪大）：現在の研究の方向を知ることができ、参考になった。中でも、本田さんの話は興味がもてた。

北野義明（ＫＣＳ）：ソフトウェアの開発上で必要十分なメディアは、オブジェクト指向でないとＷＳ上にのらないのかなあ。

楊啓廷（ＳＲＡ）：あまり予備知識がなかったので、よく理解できなかった。

鐘　友良（ＦＸＩＳ）：佐原さんの紹介したＨｙｐｅｒＣａｒｄとソフトウェア開発はおもしろいと思います。聞くと、今までのＨｙｐｅｒＣａｒｄの開発環境はまだ不十分だと思います。

桑名栄二（ＮＴＴソフトウェア研究所）：もう少し、本質的な話が欲しかった。例えば、岸田さんがＣＳＣＷの補足説明をされたが、ソフト開発の各工程、作業の中でどのようにＨｙｐｅｒＭｅｄｉａが利用されようとしているのか等。

浜野剛至（ＳＲＡ）：
・コンピュータがＭｅｄｉａとして色々な情報を扱って行くとは思いますが、そうなるとＳｏｆｔｗａｒｅがもっと面白くなるだろうと思います。
・ＣＳＣＷをもっと理解したいなと強く思います。
・ＨｙｐｅｒＣａｒｄでプログラムする時のデバッグって大変なのでしょうか。

古閑幸一（ＨＳＴ）：
（ＨｙｐｅｒＣａｒｄ）　ＨｙｐｅｒＣａｒｄを使ったことがないので？？？しかし面白そうである。
（電脳机）　確かにペーパ管理は余計な作業かもしれない。しかし、ペーパ上での自由な作業がオンラインで出

来るか！？

平尾一浩（ＨＳＴ）：ＨｙｐｅｒＣａｒｄでの生産性１０００倍はすごい。しかし、あくまでもプロトタイプレベルとかんがえなくてはならないのか。

関口純一（ＪＩＰ）：「電脳机の概念と展開」でのコピーレスという考えは面白い。ペーパレスはいまのところ、紙の方が優位に立っているので、理想としては良いが、現実的でないが、コピーレスは考慮する価値はある。「ＣＳＣＷ」という言葉を初めて聞いた。

稲村　浩（カノーブス）：コピーレス大歓迎。ＨｙｐｅｒＭｅｄｉａ対応の良いエディタ（思考と同じ速度で入力できる）が望まれる。

石元繁一（三菱電機コントロールソフトウェア）：本田氏：ペーパレスに関する同氏の考え方に共感を覚えました。しかしながら現実性はどうか？　ＥＷＳ１台／１人もさることながら、ネットワーク等技術的問題、今までの文化（‘紙’が原紙である。上司の承認印が重要）、紙はやはり圧倒的に見やすい（ちらつく画面にくらべて・・）。実現できると素晴らしいと思います。

中野秀男（阪大）：３人の発表については、良く聞いているので別に感想はありません。もう少し、ＨｙｐｅｒＭｅｄｉａの解説の話が入れば良かったかと思っています。

6.2 ＨｙｐｅｒＭｅｄｉａの討論に関する感想

棚田眞司（ＳＲＡ）：情報の分類や検索には良い環境を提供していると思う。感想は、それだけのものかといったところです。

田中正則（ＳＲＡ）：ＨｙｐｅｒＭｅｄｉａを使ってみたくなった。まず、どう使うか悩む。開発環境として使うのは難しそうなので、とりあえず日常の情報整理に使ってみたい。また、データの共同利用（コピーレス）の考え方には賛成ですが、どのように共同利用していくかは今後考えていきたい。

熊谷　章（ＰＦＵ）：議論は、カオス対論理構造という図式に終始したようだ。トリーは論理構造の代表であり、ネットワークはカオスの代表である。生物の分類はトリーの典型であるが、あれは対象物がＲｅｅｄＯｎｌｙで調べるだけだから、そして分類し区別するのが目的だから意味がある。我々が開発するオブジェクトは、従来、世の中にない物を作り出したり、考え出すのだからカテゴライズは二次的なものだ。一次的なものは、いかに考えることを支援するかである。思考と表現とそれらの評価がうまくやれる仕組みとしてコンピュータを使用したいのだ。ＨｙｐｅｒＭｅｄｉａシステムは、コンピュータの全く新しい一面を引き出すキーになると確信している。

野中　哲（ＮＥＣ）：現在のＨｙｐｅｒＴｅｘｔは、これから出てくるいろいろなシステムの基礎となるべきものなのでしょうね。いままでの単純なｂｙｔｅストリームのファイルに、ＨｙｐｅｒＴｅｘｔがどってかわり、その上にいろいろなツールを構築することになるだろうな。つまり、アセンブラのようなものでしょう。

山口郁生（電力計算センター）：非常におもしろい議論だった。その中で印象に残ったのは、

・ＨｙｐｅｒＭｅｄｉａは頭の中の発想をそのまま計算機に移したものである。今までの計算機の発想でとらえてはいけない。

・オブジェクト間のリンクがダイナミックに生成、消滅するようなシステム。まるで、ニューロとシナプスの関係のようだ。

林　　香（ＳＲＡ）：

ＨｙｐｅｒＭｅｄｉａ：これから５年は研究できる。

←いろいろな利用方法がありそう。

←完成度が低い。ｅｘ．データの複雑度の計算

←へたなＣＡＳＥツールより良いかも

←変なＭｏｄｅｌＢａｓｅｄ

←私はＭｏｄｅｌＢａｓｅｄツールが嫌いです。

←ｌｅａｒｎしなければ使えない。

←具体化が悪いだけかもしれない。

関口純一（ＪＩＰ）：

やはり使ってみないと討論に参加できないので使ってみたい。

藤野晃延（ＦＸＩＳ）：なかなか有意義な議論だったように思う。ＨｙｐｅｒＭｅｄｉａはもっともっと広く使われる必要がある。そこから新しい使い方や、その使う上での追加されるべき機能がアイデアとして出てくるはず。つまり、新しい「ＤｒｅａｍＭａｃｈｉｎｅ」というものを試行錯誤しながら形作って行くのではないかと思う。まあ、いろいろとやってみれば。

桑名栄二（ＮＴＴ）：岸田氏の問題提起に関して感謝。しかし、議論は表面的なものに傾いていたように思う。もう少し、設計作業、デバッグ作業の中にｏｂｊｅｃｔは何で、どのようにリンクを張るかを見つけるために、現在の開発作業、作業者のふるまいの分析・研究の必要性について議論出来ればと良かったように思う。Ｈｙｐｅｒｍｅｄｉａは単なる道具。ＳＡ／ＳＤツールも単なる道具であり、本当に使おうと思えば、設計に対する知識が必要。開発に関する知識を整理し、ｋｎｏｗ－ｈｏｗがなくても開発できるようにしたい。

藤岡　卓（三菱電機）：早く使ってみたいとは思っているのですが・・

渡邊雄一（電力計算センター）：ＨｙｐｅｒＭｅｄｉａって本当にＨｙｐｅｒなの？ＨｙｐｅｒＭｅｄｉａのインプリメント、もしくは管理方法と、より好ましいＭＭＩを加味した利用方法は今後の課題。まず、お金を確保してＨｙｐｅｒＭｅｄｉａに接して行くことから始めたいが。

北野義明（ＫＣＳ）：ＨｙｐｅｒＭｅｄｉａの良さはリンクの張り易さ。それ以外の事が駄目だといって、ＨｙｐｅｒＭｅｄｉａはダメというのは早い。もっとリンクの張り方について議論したかった。

浜野剛至（ＳＲＡ）：出始めのものにＭａｎａｇｅｍｅｎｔのＳｃａｌｅを当てはめようとするといびつになりはしないかと心配です。でもモデルは必要だと思います。カオスもいいですけどね。

岸田孝一（ＳＲＡ）：ソフトウェアエンジニアは、あまり「モデル」が好きではないように感じられる。これは問題だ！

杉田義明（ＳＲＡ）：リンクのつけすぎた場合のさまざまな問題について、より深く議論したい。さらにダイナミック・リンクをマルチユーザの環境で、どのようにつけて活用していくのか。使えば楽しいと思うので、これから楽しみだ。

髙橋哲也（神戸製鋼所）：いろいろな立場からみた、自由な意見が飛び交って大変面白かったです。

本田克己（ＹＨＰ）：大変面白かった。もっと気楽にＨｙｐｅｒＭｅｄｉａを使って欲しい。

飯塚正樹（ＹＨＰ）：ハイパーメディアシステムを作ろうとしている人、使おうとしている人、それをマネージメントしようとしている人の間での自分のもくろみを聞くことができた。それを使う上での問題ていぎを聞くことができたが、身がまえてしまっている部分が多いと思われる。もっと気軽にだまされてみるのもいいのではないか？

三浦あさ子（ＳＲＡ）：

・ＨｙｐｅｒＭｅｄｉａをｔｅａｍでの開発に使うにはまだまだ時間がかかりそうだ。

・いろいろな意見があって面白かった。将来があるものだと思うのでみんながいろいろな使い方をして、いろんな方向に発展していった方がいいと思う。

野村行憲（ＩＣＳ）：

ＨｙｐｅｒＭｅｄｉａのｓｏｆｔｗａｒｅへの応用は次の３つのパターンがある。

・ＨｙｐｅｒＭｅｄｉａの世界を構築する。

・ＨｙｐｅｒＭｅｄｉａそのもののアプリケーションを作る。

電脳机　スタックウェア

・ＨｙｐｅｒＭｅｄｉａでソースプログラムを表現する。

このあたりが整理されずに議論されているような感じがする。

野見山和則（三井銀ソフトウェアサービス）：

・リンクの管理はＲＤＢ等の機能を使うしかないと思うが、誰でも簡単に扱えるようにするにはしんどい感じがする。

・パーソナルな世界からグループワークの世界へ移って

きているのだからＭｏｄｅｌ必要派の岸田さんの意見に賛成。

中田修二（シグマ）：カオス対モデルと人間との議論が有益だったと思います。ＨｙｐｅｒＭｅｄｉａの使い方とツールとしての限界がある程度浮きぼりにされたと思います。

辻本憲一（シグマ）：ハイパーカードのメリットは、これまでの設計書がまとまるまでにいたる過程で発生するメモの記述の整理することにあるような気がする。これまでの定型的なドキュメントでは、ノウハウは記述しきれない。

安間文彦（富士通エフアイビー）：ＨｙｐｅｒＭｅｄｉａが５年後どのように普及しているか興味がわいてきた。

今別府茂暢（ＮＴＴ）：電脳机を使ってみて、次のハイパーメディアを考えたい。

濱田　勉（ＮＴＴ）：
・感覚的にはよさそうネと思います。
・集めるだけ集めたデータの中から、どうやって自分の必要なところを見つけだして、意味づけ出来るのか

古閑幸一（ＨＳＴ）：ＨｙｐｅｒＭｅｄｉａはやっと人間の考えを道具を使ってうまく機械の中に取り込めるようになったと感じた。

久下真司（阪大）：グラフ理論を使ったハイパーメディアの表示についてもっと議論して欲しかった。

澤田宣己（阪大）：今までの机の上の混乱を単にコンピュータに持ち込むのではなくいかにハイパーメディアを用い、それを表示するかの話はとても興味深く聞くことができました。

久保宏志（富士通）：「ハイパーテキストをimplementation languageとするソフトウェア開発プロセスを管理するためのquatitve measureとhow to manage」という岸田さんの問題提起は大事。「ネットワーク・オブジェクト、トリー・オブジェクトを扱う技術はハードウェアの世界の方が進んでいる。ソフトウェア人はもっとハードウェア技術を学べ」という中野さんの忠告は耳を傾けるに値する。

上妻健一郎（電応研）：本田さんの話でリンクポインタの代わりにＳＱＬのコマンドを与えられることで動的なリンクを張ることができるということだったが、これではリレーショナルデータベースのフロントエンドとして使えるということと同じではないだろうか。岸田さんの話でハイパーメディアを利用したソフト開発を管理者からみて何か定量的な工程評価ができるメジャーを用意する必要があるという提言は現場で管理者に受け入れてもらうためには非常に重要だと思う。中野先生のお話でハイパーメディアはネットワーク型だけれども、そのなかに木構造が埋まっていると思うという意見には同感だ。

佐藤千明（長野県協同電算）：モデルが必要でない場合には簡単にリンクして管理できるＨｙｐｅｒＣａｒｄは有効でしょうが、大規模システム開発には何らかのモデル、手順が管理上必要でしょう。個人で使う場合には、Ｅａｓｙなツールでもいいでしょうが、ＨｙｐｅｒＣａｒｄをうまく活用した大規模プロジェクトの実績がみたい。どこかでやっているでしょうか？

海尻賢二（信州大）：
・ＨｙｐｅｒＣａｒｄは結局ｓｔａｔｉｃ（？）
・Ｘｗｉｎｄｏｗの．Ｘｄｅｆａｕｌｔのようなｉｎｉｔｉａｌファイルを用意して、カストマイズ可なＨｙｐｅｒＴｅｘｔを作りたい。
・イメージをＨｙｐｅｒＭｅｄｉａに載せて回線の速度の速度として可能か？数百台の分散環境で。
・手書きの図を入力して、ｃｈａｒｔ　ｅｄｉｔｏｒで編集して他のｒｅａｄｅｒに提供したい。

平尾一浩（ＨＳＴ）：ＨｙｐｅｒＭｅｄｉａで、開発の考え方自体が変わってきていると思う。討論の中で、モデルの必要性が挙げられたが、私はその必要性を強く感じている。今までいろいろなモデルが出されたが、実際の開発に「ピッタリ」あうのは出ていない。Ｈｙｐｅｒで開発自体が変わっているので、まったく新しいモデルを考えるチャンスではないかと思う。佐原さんが、サッカーの動きを例に挙げてモデルは不必要と言われたが、それはサッカーをやっている一人一人の動きに大きな信頼がなければならない。モデルはモデルなので、自由度

のある基本部分のみをモデル化することで、人の動きに関して、信頼度０から信頼度４０～５０にすることでできるのではないかと思う。だって、サッカーも基本的な動きはあるわけでしょう。

中野秀男（阪大）：予想されたことだが、質問者が固定されるのはしかたがないのか。最後のｐｒｅｓｅｎｔａｔｏｒが言った「とにかくＨｙｐｅｒ＊は使ってみて下さい、楽しいですから」という言葉は面白かった。

日本語　ハイパーテキストシステム　「電脳机」
技術資料

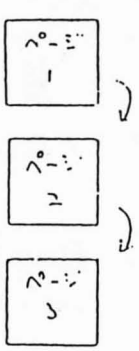
05/02/88 RevE

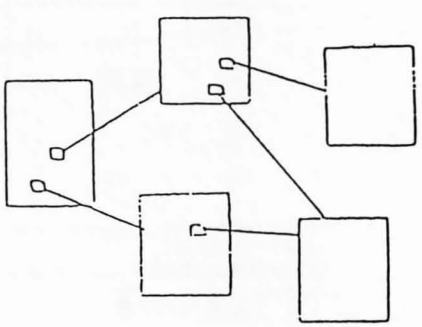
YHP CSO　本田克己

ハイパーテキストとは何か

ハイパーテキストとは、アメリカのTed Nelsonによって提唱された、新しいコンピュータの利用環境のコンセプトで、アメリカで、大学／研究機関で近年研究が盛んになっている。実用化されつつあるものとしては、XEROX社のNoteCards、OWL社のGuide などが発表されはじめた。日本でもBTRONプロジェクトの実身／仮身のコンセプトが全く同じものである。ハイパーテキストのアイデアは非常に単純である。従来のワープロやデータベースの利用方法が、情報を頭から順にみていく"線形な文書"であるのに対して、情報中のキーワード／アイコンなどをマウスなどでピックすることによって意味的に関係した別の文書／情報を次々に検索していく"非線形な文書"、あるいは動的な検索(Dynamic browsing)が可能な文書のことをハイパーテキストと呼ぶ。初期の研究時には"文章情報"だけで、図形情報などは扱えなかったので、ハイパー"テキスト"という名前が残っているが、実際のシステムでは図形、画像、あるいは音声動画像も含めているものもあり、ハイパーメディアと呼ぶべきという説もある。

線形文書　　非線形文書





ハイパーテキストの基本的なコンセプトは、もう我々のまわりにある本に見つけることができる。たとえば、脚注、注などの注記である。本文の中で補足的な説明が必要であり、かつ、本文中にその情報を埋め込むのが適当でないとき"注"として補足情報を別のところにおき、注１、注２などのように情報が関係していることを示す記号をつけ（ハイパーテキストではこれをリンクと呼ぶ）、本文を見やすく、かつ、詳細な情報が必要な人にはそれが取り出せるようにしている。これはハイパーテキストそのものであり、コンピュータでは注１、注２などが文書上の特定の単語であったり、アイコンであったり(NoteCardsの"リンクアイコン")、特定の四角形のエリアであったりするだけである。

しかし、利用するときにはコンピュータによる効果は非常に大きく、注をさがすためにいちいちページをめくらなくても、"リンクアイコン"なり"ボタン"をマウスでピックするだけで、そのページが、情報が、画面上に表れてくるので高速で使いやすい。

もう一つの本の中でのハイパーテキストのコンセプトは、「目次と本文」の関係のようなものである。本の目次をざっと見わたすだけで本全体の内容の概説を知ることができる。あとは必要な部分のみ何ページにあるかを調べて、そこを見ればよい。

このように、情報を階層化して取扱いやすいようにできる。ハイパーテキストでは、目次のページの内でページ数を示すところに、実際のページ番号のかわりに"リンクアイコン"をつければよい。

"注"が関係情報や参照のような横方向のリンクであるのに対して、"目次"などは情報を階層化する縦方向のリンクである。もちろんハイパーテキストではこの両者を区別することなく、統一的に扱うことができる。"目次"的な利用方法だけの実例としては、最近はやりだしているアイデアプロセッサなるソフトとしてみることができる。

3) オープンなアーキテクチャであること

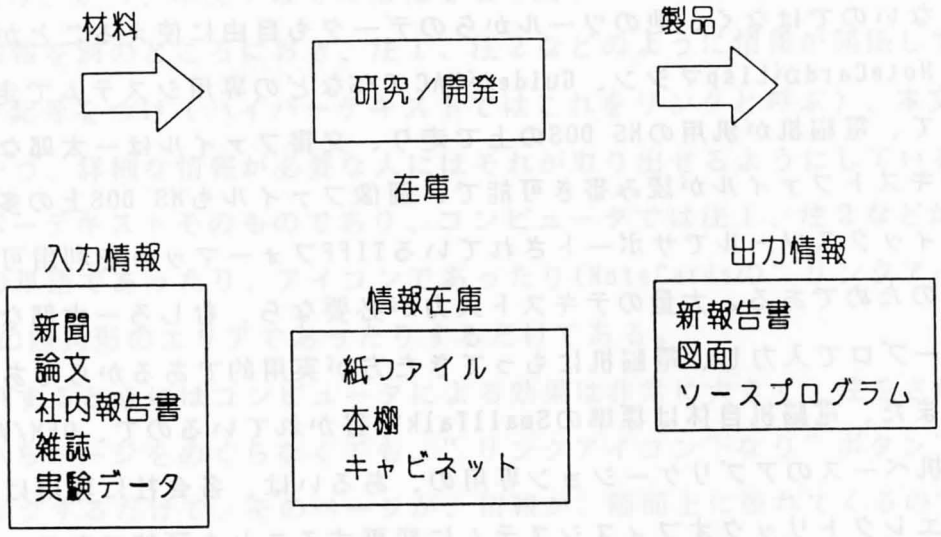
ハイパーテキストは"情報をみるため"のツールであるから、その情報をつくるためには、すべてハイパーテキストのソフトウェアの上だけでしかできないのではなく、他のツールからのデータも自由に使えることが望ましい。NoteCardがLispマシン、GuideがMAC O/Sなどの専用システムで走るのに対して、電脳机が汎用のMS DOSの上で走り、文書ファイルは一太郎などの標準テキストファイルが読み書き可能で、画像ファイルもMS DOS上の多くのグラフィックスツールでサポートされているTIFFフォーマットを利用可能なのもこのためである。大量のテキスト入力が必要なら、むしろ一太郎などの専用ワープロで入力し、電脳机にもってきた方が実用的であるからである。

また、電脳机自体は標準のSmallTalkで書かれているので、OEM/VEUには電脳机ベースのアプリケーション専用の、あるいは、各会社に専用に設計されたエレクトリックオフィスシステムに変更することも可能である。

以上に加えて、文節変換を含む日本語機能などは、必要最小限のハイパーテキストシステム機能と考えているが、電脳机を除いて、市販／研究中のものも含めて、このレベルのものはまだないようである。

電脳机は、このように"新しいコンピュータの応用"を考える実用的なコンセプトマシンである。

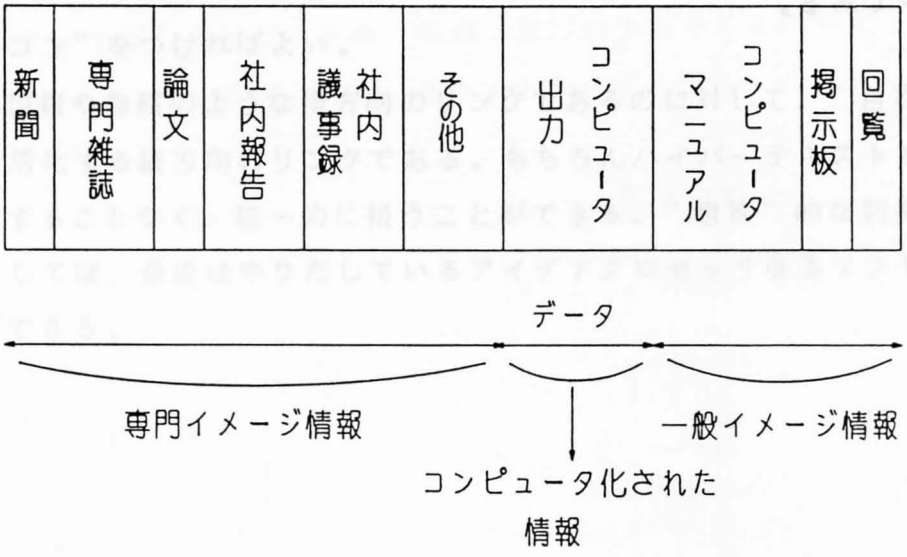
開発業務を" 製造" ととらえる



CSO KY PROJECT
CSOKY0088K.Honda 6/20/88)

YOKOGAWA HEWLETT・PACKARD

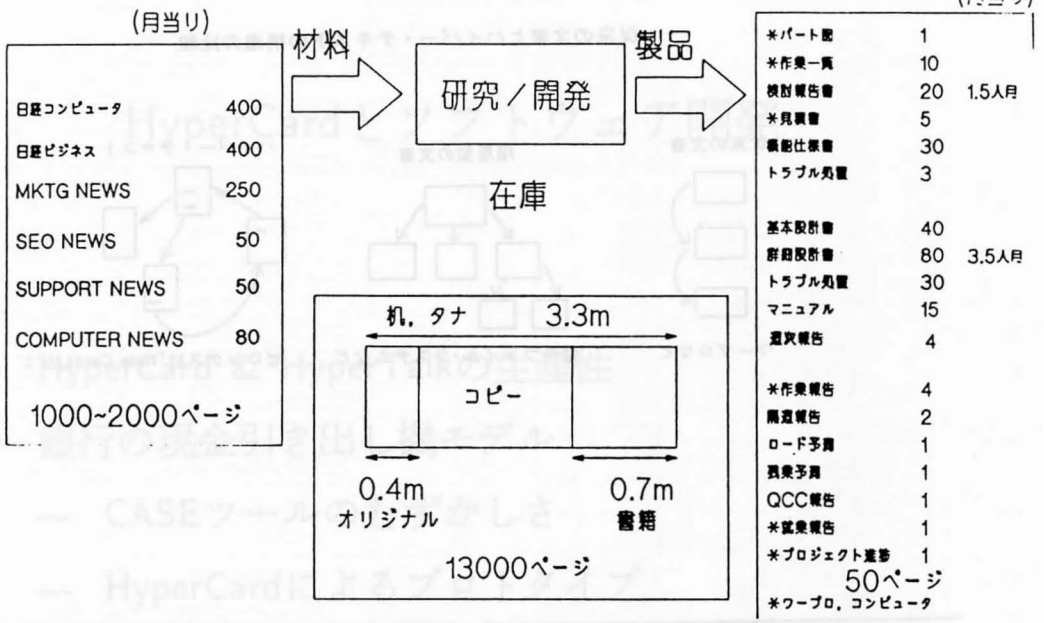
エンジニアへの入力情報



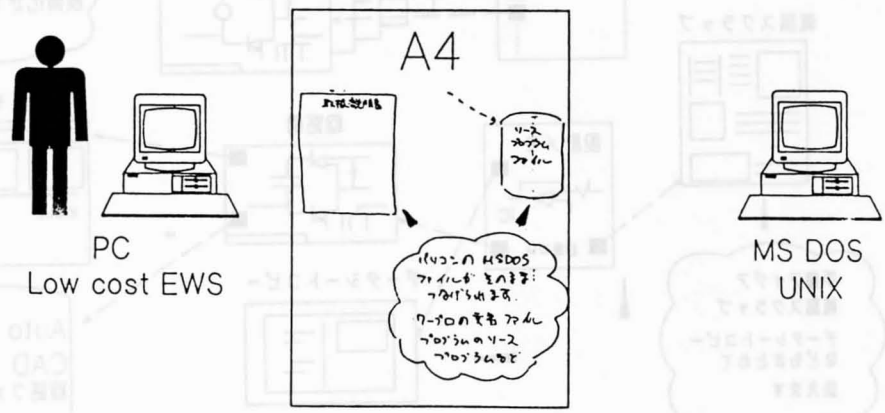
CSO KY PROJECT
CSOKY0090K.Honda 6/20/88)

YOKOGAWA HEWLETT・PACKARD

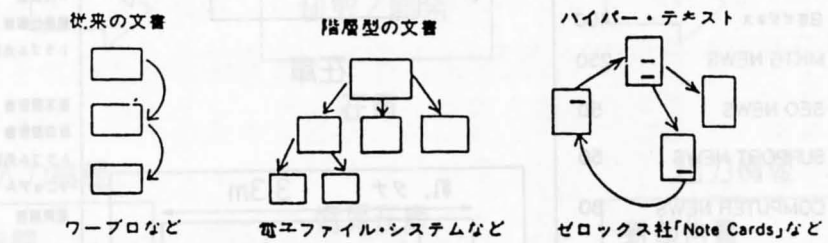
ソフトウェア開発プロジェクトリーダの例



1人1台　A4イメージベース　オープンアーキテクチャ



従来の文書とハイパー・テキストの構造の比較

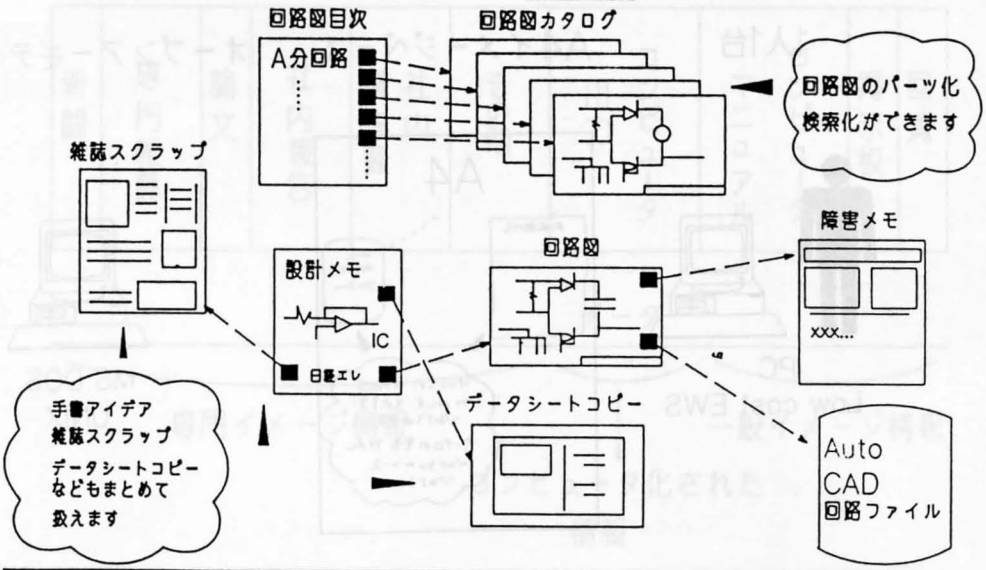


CSO KY project
Phoenix OHP (K.Honda

YOKOGAWA HEWLETT・PACKARD

ハードウェアエンジニアのBさん

あなたの設計図がすべて 電脳机 でさがせます



CSO HONDA
SEPC003(1988/06/08)

HEWLETT PACKARD

cut ↓ ↓ cut ↓

88年11月7日
SRA
ホイル 1

HyperCardとソフトウェア開発

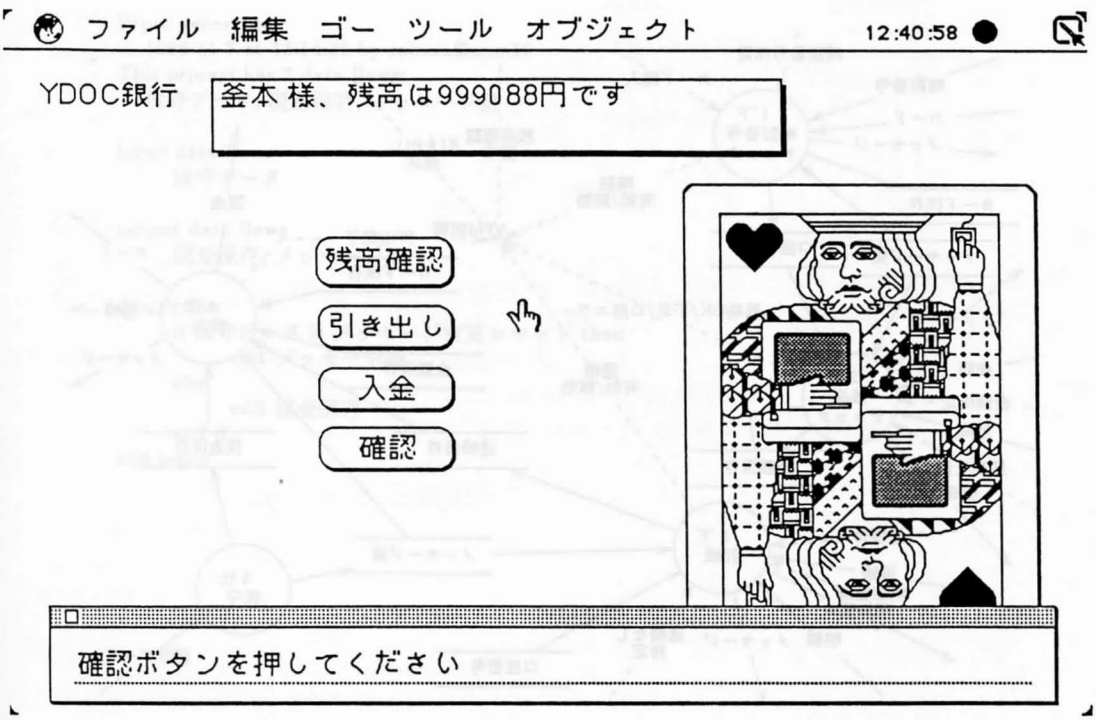
- HyperCard & HyperTalkの生産性
- 銀行の現金引き出し機モデル
  - — CASEツールのむずかしさ
  - — HyperCardによるプロトタイプ

佐原伸

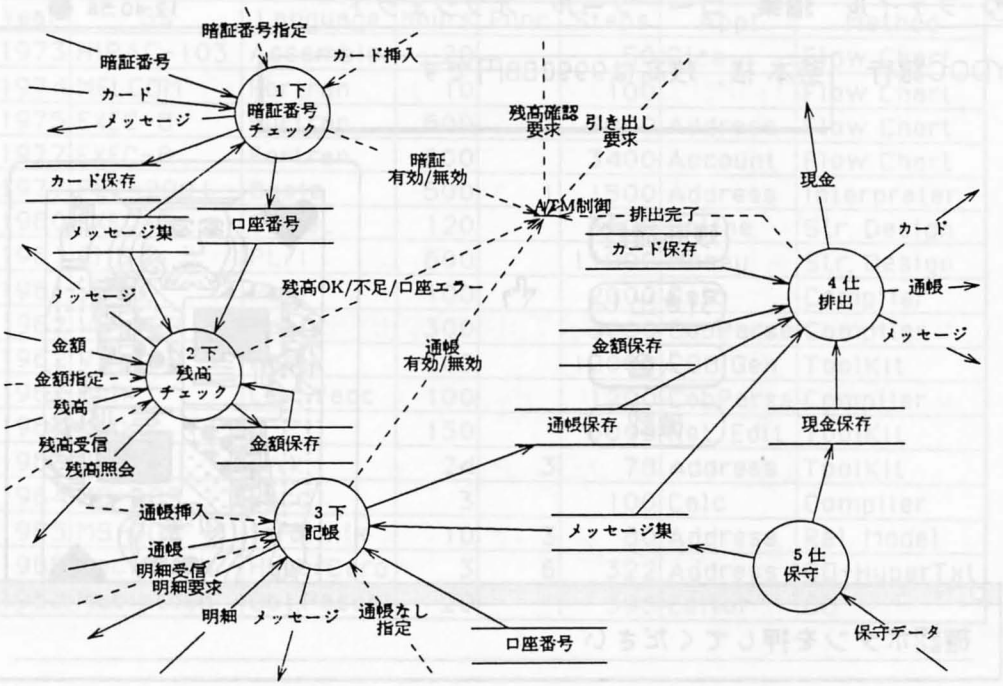
SRA

Table-1 Productibity

	A	B	C	D	E	F	G	H
1	Year	OS	Language	hours	Func	Steps	Appl	Method
2	1973	HIPAC-103	Assembler	20		50	Dice	Flow Chart
3	1974	MELCOM	Fortran	10		100		Flow Chart
4	1975	EXEC-8	Fortran	600	1	3000	Address	Flow Chart
5	1977	EXEC-8	Fortran	600		3400	Account	Flow Chart
6	1977	PET-2001	Basic	500	1	1500	Address	Interpreter
7	1980	EXEC-8	COBOL	120		3000	Online	Str. Design
8	1981	CMS	PL/I	600		11000	Money	Str. Design
9	1981	VOS3	PL/I	100		2000	Calc	Compiler
10	1982	VS-100	Basic	300		3000	CobParse	Compiler
11	1982	VS-100	Basic	600		12000	COB Gen.	ToolKit
12	1982	Unix	Lex, Yacc	100		1200	CobParse	Compiler
13	1983	Unix	C	150		3000	Net. Edit	ToolKit
14	1983	Unix	awk	24	3	78	Address	ToolKit
15	1984	MS-DOS	Yacc	3		100	Calc	Compiler
16	1985	MS-DOS	Informix	10	3	80	Address	Rel. Model
17	1988	Macintosh	HyperCard	3	6	322	Address	OO+HyperTxt
18	1988	Macintosh	Obj.Pascal	20		396	Editor	OO


ファイル 編集 ゴー ツール オブジェクト
12:40:58
YDOC銀行
釜本 様、残高は999088円です
残高確認
引き出し
入金
確認
確認ボタンを押してください

atm1



現金自動支払機

プロセス

Process 4: 排出

Pspec generated
1988 11 7 at 13:14:20 by sahara@sran15
This process has 9 data flows:
現金, カード, メッセージ, 通帳, カード保存, 通帳保存, 現金保存, 金額保存, メッセージ集

input data flows
カード保存, 通帳保存, 現金保存, 金額保存, メッセージ集

output data flows
現金, カード, メッセージ, 通帳

description
put カード保存 into カード
put 通帳保存 into 通帳
put 返却メッセージ into メッセージ
if 金額保存 > 現金保存 then
put 現金不足メッセージ into メッセージ
else
put 金額保存 from 現金保存 into 現金
end if
send 排出完了
end pspec

Process 5: 保守

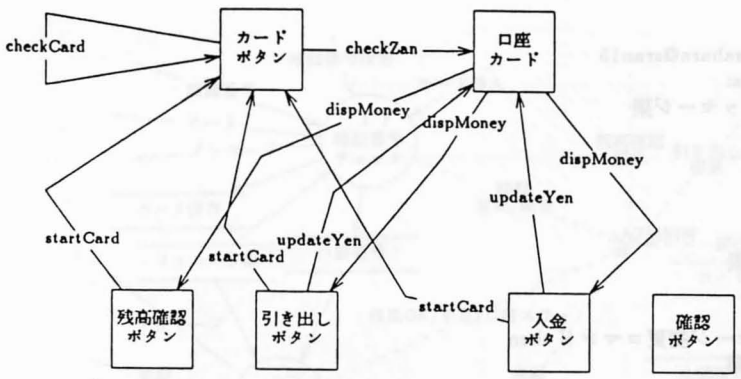
Pspec generated
  1988 11 7 at 13:14:21 by sahara@sran15
This process has 3 data flows:
  保守データ, 現金保存, メッセージ集

input data flows
    保守データ

output data flows
    現金保存, メッセージ集

description
    if 保守データ is メッセージ変更コマンド then
        edit メッセージ集
    else
        edit 現金保存
    end if
end pspec

atm1


checkCard
カード
ボタン
checkZan
口座
カード
dispMoney
dispMoney
dispMoney
updateYen
startCard
updateYen
startCard
startCard
残高確認
ボタン
引き出し
ボタン
入金
ボタン
確認
ボタン

atm1

```
      atm    88.11.07     12:57     1

Script = *** atm ***
on openStack
 hide message box
 show menuBar
 pass openStack
end openStack

on idle
 global youkyu
 if youkyu <> "残高確認" and youkyu <> "引き出し" ツ
 and youkyu <> "入金" then
  put card field "金額表示" into work
  put "皆様のYDOC銀行です" into card field "金額表示"
  put "ボタンを押してください" into line 2 of card field "金額表示"
  wait 30
  put work into card field "金額表示"
  wait 10
 end if
end idle

*** Background = bkgnd id 2692 ***
*** Card =card "口座" ***
on checkZan
 global  youkyu
 push this card
 set the lockScreen to true
 go to stack "口座"
 go to any card
 put background field "残高" into money
 put background field "顧客名" into namae
 pop  card
 set the lockScreen to false
 if youkyu = "残高確認" then
  send "dispMoney" && money  & "," & quote & namae & quote ツ
  to card button "残高確認"
 else if youkyu = "引き出し" then
  send "dispMoney" && money  & "," & quote & namae & quote ツ
  to card button "引き出し"
 else
  send "dispMoney" && money  & "," & quote & namae & quote ツ
  to card button "入金"
 end if
end checkZan

on updateYen
 put param of 1 into money
 put param of 2 into namae
```

```
        atm    88.11.07     12:57    2
  push this card
  set the lockScreen to true
  go to stack "口座"
  find namae in field "顧客名"
  put money into background field "残高"
  pop  card
end updateYen

*** card button "残高確認" ***
*** Location: 182,122 ***
*** Rect: 150,110,215,135 ***
*** Visible: true ***
*** Style: roundRect ***
*** Script: ***
on mouseUp
  global  youkyu
  put "残高確認" into youkyu
  send "startCard" to card button "カード"
  set hilite of card button "カード" to true
end mouseUp

on dispMoney
  put the param of 1 into money
  put the param of 2 into namae
  put namae && "様、残高は" & money & "円です"
  into card field "金額表示"
  put "確認ボタンを押してください"
end dispMoney
*** card button "カード" ***
*** Location: 405,199 ***
*** Rect: 322,84,489,315 ***
*** Visible: true ***
*** Style: transparent ***
*** Script: ***
on mouseUp
  send "checkCard"
end mouseUp

on startCard
  put "カードを入れてください"
end startCard

on checkCard
  ask "暗証番号を入れてください"
  if it is 1919 or it is 9999 then
    put "暗証番号が正しくありません"
  else
```

```
      atm    88.11.07     12:57    3

   send "checkZan"
  end if
end checkCard*** card button " 確認" ***
*** Location: 183,213 ***
*** Rect: 151,202,215,224 ***
*** Visible: true ***
*** Style: roundRect ***
*** Script: ***
on mouseUp
  global youkyu
  put "" into youkyu
  hide message box
  put space into card field "金額表示"
end mouseUp
*** card button "引き出し" ***
*** Location: 183,154 ***
*** Rect: 151,143,215,165 ***
*** Visible: true ***
*** Style: roundRect ***
*** Script: ***
on mouseUp
  global youkyu
  put "引き出し" into youkyu
  send "startCard" to card button "カード"
  set hilite of card button "カード" to true
end mouseUp

on dispMoney
  put the param of 1 into money
  put the param of 2 into namae
  put namae && "様、残高は" & money & "円です"
  into card field "金額表示"
  ask "引き出し金額を入力してください"
  put it into getMoney
  put getMoney & "円引き出します" into card field "金額表示"
  put money - getMoney into i
  if i < 0 then
    beep
    put "残高が" & i & "円足りません" into card field "金額表示"
  else
    put namae && "様、残高は" & i & "円です"
    into card field "金額表示"
    send "updateYen" && i  & "," & namae  to card口座"
  end if
  put "確認ボタンを押してください"
end dispMoney
*** card button "入金" ***
```

```
        atm    88.11.07      12:57     4

 *** Location: 183,184 ***
 *** Rect: 151,173,215,195 ***
 *** Visible: true ***
 *** Style: roundRect ***
 *** Script: ***
 on mouseUp
   global youkyu
   put "入金" into youkyu
   send "startCard" to card button "カード"
   set hilite of card button "カード" to true
 end mouseUp

 on dispMoney
   put the param of 1 into money
   put the param of 2 into namae
   put namae && "様、残高は" & money & "円です"
   into card field "金額表示"
   ask "入金額を入力してください"
   put it into inMoney
   put inMoney & "円入金" into card field "金額表示"
   put money + inMoney into i
   put namae && "様、残高は" & i & "円です"
   into card field "金額表示"
   send "updateYen" && i  & "," & namae  to card口座"
   put "確認ボタンを押してください"
 end dispMoney
```

SEA 3rd Software Development Environment Workshop

The Time for Computer-Supported Cooperative Work

Terunobu Fujino

Fuji Xerox Information Systems Co., Ltd.

Abstract

Since to achieve the higher software productivity variety of tools and development environments have been developed and introduced and somehow becoming in real-use. Actually the progress of micro-processors and its recent MIPS war have been changing the style of software development. What conventional software tools and environments support is mainly focusing on individual work and little are taken into account for group work. However in large-scale software development the ratio of the collaboration work time against whole development time is extremely high, so that without computer supports for collaboration to achieve more efficient productivity seems to be difficult.

In this paper current CSCW activities and basic concept of CSCW are reported, and applicability of the CSCW technologies for Software Engineering is also discussed.

1. はじめに

ソフトウェア開発に纏わる様々な問題、例えば生産性であるとか品質であるとか、これを克服しようとすると既に素手で立ち向かうことは難しい。何故なら対象とするソフトウェアが余りにも巨大、且つ複雑になってきているからである。従ってツール、あるいは開発環境という一種の「武装」を行うことでこれに立ち向かうことになる。このとき、今迄のツールあるいは開発環境というものは概ね、個人当たりの「能力」の向上を目指した観点からの「装備」であった。つまりソフトウェア開発に携わる者一人一人の能力増幅器といったものであり、個人レベルの能力の増幅がその使命であるといえる。

しかし個人レベルの能力増幅では、現実のソフトウェア開発に対し充分な効果を上げえるとは言い難い。実際のソフトウェア開発では、多くの人が相互に関連して作業を行うことが普通であり、この傾向は開発するソフトウェアの規模が大きくなるほど顕著である。従来の支援ツールや開発環境では、この協同作業、あるいは協調作業を対象として、それを効果的に支援しようとするものは殆どなかった。

しかし、近年のマイクロ・プロセッサの処理能力向上、およびメモリ・チップの高集積化やCD-ROM等の新しい周辺機器の出現により、今迄は扱うことが難しかった様々なデータ、それは画像や音声といったマルチ・メディアである、がコンピュータで扱えるようになったこと、更にネットワークの拡充により、複数のコンピュータ間の情報交換、通信が飛躍的に向上したこと等により、協調作業を支援する為の基盤が漸く整うこととなった。

ここでは「コンピュータによる協調作業支援」の現状と動向について紹介する。

2. CSCWの概要

「コンピュータによる協調作業支援」を略してCSCWと呼ぶ。これはこの分野でのパイオニア的な活動を行っていたコンファレンスに由来している。以下に簡単にその歴史を示す。

◆DEC/MIT Workshop on Computer-Supported Cooperative Work : Aug., 1984

◆MCC Interdisciplinary Design Symposium : May, 1985

◆*CSCW '86* : Dec., 1986 @ Austin, Texas

◆*CSCW '88* : Sep., 1988 @ Portland, Oregon

CSCWの目的は、「コンピュータにより、如何に協調作業を支援するか」ということであるが、その目標とする処は次のように考えられている。

- CSCW Objectives

◆Addresses the **processes** of **people working together** in organizations.

◆Actually an **extention** of **OA** that could fix the **OA problem**.

◆like **designing highway systems** for cars rather than **people in cars**.

3. CSCWの4つのカテゴリ

一口にCSCWといってもその守備範囲は広い。その性格からCSCWを更に4つのカテゴリに分けることができる。

- 4 major Categories from CSCW '86

◆Workgroup Communication
·small teams of people
·face-to-face interaction

◆Organizational Communication
·Electronic mail
·Computer Conferencing

◆Collaboration Data bases
·HyperText/Media
·captures the interpersonal transactions

◆Real-time Meetings
·whiteboard (analogy of corkboard)
·electronic Brainstorming

またこのカテゴリ分けに従ってCSCW'88の論文を分類すると次のようになる。

Type of Effort*

Major Area	System Development	Evaluations	Survey	Theory
I. Workgroup Communication	9, 10, 22, 27**	19, 28	29	
II. Organizational Communication		5, 6, 7, 8, 15, 24, 26		25
III. Collaboration Data Bases (hypermedia)	11, 12, 13, 14	16		17
IV. Real-time Meetings	1, 2, 3, 21, 23, 30	4, 18, 20		

*Most reports fall into multiple categories; this is the primary focus
**Numbers correspond to those used in the conference proceedings.

T.F. FXIS, Nov.24,'88

同様にこれらの中核をなす基盤技術として、以下のようなものが揚げられる。

- Underlying Technologies:

  ·data bases
  ·structured documents and hypertext
  ·access controles
  ·privacy
  ·networks and network services

就中、Hypertext/mediaに関するテクノロジは、今後のソフトウェア開発および支援環境の重要な要素となりうる。以下ではHypertext/mediaにスポットをあてる。

4. Hypertext/mediaのソフトウェア開発への応用

一言で言うと、ハイパーテキスト/メディアとは、ネットワーク状のリンクで結合されたノードに基づいてデータが蓄積されるような情報管理手段を指す。その基本的な機能として、次の3項目が考えられる。

- network of ***textual/graphical*** nodes (≃ OODB)
- *pointers* to other nodes are represented as ***link icons***
- easy to ***create/append*** new *nodes* and new *links* ***dynamically***

またブラウズ(拾い読み)できるよう、以下の機能を備えている。

- can be *browsed* in 3 way :

  ① by **following links** and opening windows successively to examine their contents

  ② by *searching* the network for some **string/key-word/attribute-value**

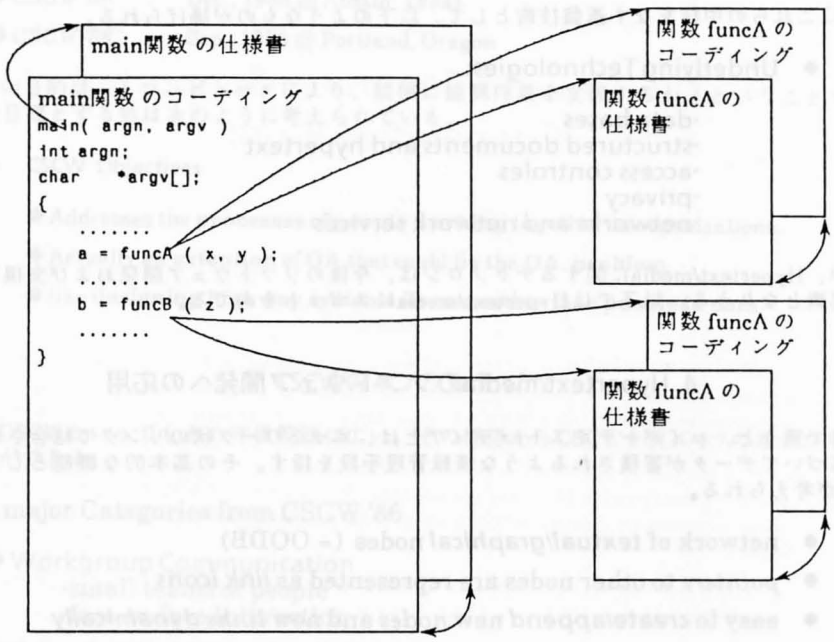
  ③ by *navigating* around the nodes in the network *graphically* diplayed

このようなHypertextの性質、機能は、ソフトウェア開発ツールとしても応用可能である。

- **HyperText as a programming tool**
  - several logical relations can be expressed
  - ***caller/callee*** relationship between modules
  - *revision/history* ***Links*** can be easily provided
    (*graphically* and *logically*、by *browsing/traversing* links)

次に示す図のように、モジュール間の呼び出し関係を用いて動的に他のモジュールを参照、更新したり、あるいはモジュールのコードに対応する仕様書等のドキュメントを関連付けておくことにより、コードとドキュメントの一貫性を保持することができる。
Hypertextを応用することの大きな特徴は、上記のようなリンク(関係)が視覚的に理解しやすい形で提供されることにある。これにより利用者の操作性、認識容易性、さらにはcognitive-overheadと呼ばれる対象の切り替え(context-switch)時の心理的負荷も軽減される。

最近ではMac上のHyper-Cardが同様の機能を提供している。Hyper-Cardでは更にこの概念を推し進めて「**Programming by Example**」というソフトウェア作成の新しいアプローチを提唱しており、ユーザ自身が例題に基づいてソフトウェア(スタック·ウェア)を作成したり、カストマイズしたりすることが可能となっている。



更にユーザ･インタフェースの標準として提案されているOpen-Look®等では、ヘルプ機能をHypertext関連の技術を応用して実現している。

Acknowledgements

Author would like to express much appreciations to our colleagues, especially Jun Aoki, Youliang Zhong for their valuable discussions with me and for their excellent works to implement and evaluate the several tools and environments. Sincere thanks also to our director N.Tatsuta for giving me the chances to explore this study.

Refferences

[1] Irene Greif, ed. "Computer-Supported Cooperative Work: A Book of Readings", Morgan Kaufmann Pub., 1988

[2] Edward Barret, ed., "Text, ConText, and HyperText", MIT Press, 1988

[3] David Alex Lamb, "Software Engineering: Planning for Change", Prentice Hall, 1988

[6] Charles F. Martin, "User-Centered Requirements Analysis", Prentice Hall, 1988

第4回 ＳＥＡ環境ワークショップ

セッション２ ＭＭＩ：ソフトウェア（プロセス／プロダクト）の可視化はどこまで可能か？

チェアマン ： 野村 行憲（ＩＣＳ）
プレゼンテータ： 古閑 幸一（ＨＳＴ）
： 塩谷 和範（ＳＲＡ）
： 柳瀬 健一（ＫＣＳ）
レポータ ： 森本 雅子（ＫＣＳ）

1. はじめに

野村（ＩＣＳ）： 岩手電子計算センターの野村です。ＭＭＩのセッションは一人３０分程度喋ってもらって、それに関して討論し、最後に全体についても討論したいと思っています。それでは最初のＨＳＴの古閑さん、お願いします。

2. プレゼンテーション

2.1 ＡＰ開発におけるシミュレーションの活用

古閑（ＨＳＴ）： おはようございます。私は熊本から来ましたヒラタ・ソフトウェア・テクノロジーの古閑と言います。ここでは、ビジュアルなプリベーションに関する考えについてお話したいと思います。

最近、私たちの会社では CRT ライン用のアプリケーションソフトウェアを開発いたしました．これは上位アプリケーション．．．．アプリケーション上位がプログラマブル・コントローラ、私共のほうではシーケーサーと呼んでますが、そういう部分で、データを受け取り、画面に情報を表示するというものです．ここでは、それを作成する過程で色々感じたことについてお話したいと思います。　なお私が考えるビジュアルな画面というのはですね、画面上に線ですとか、円ですとかを使って絵が描かれているものですとか、画面上に任意の位置に数字データとかを表示するものを言います。

これが実際に作成したアプリケーションのＭＭＩの部分なんですけど、これはＸライブラリ関数を使って作成されたものです。ちょと簡単に画面の方を説明させていただきますと、この部分がグラフになっていて、いろんなデータをグラフ上にして表すという部分、ここもグラフなんですけど、ここのところにいろんな文字データとか表示されています。

ここにも、いろんな機械データというか、実際にはＣＲＴのバッキングラインだったんですが、ＣＲＴのシーケンシャル No とかがここに入っているわけです。私達がこの画面を作成する上で、プロジェクト上では画面制御支援グループというのを作りまして、あまりＸライブラリ関数とかを意識せずに、いいように、ユーティリティ関数というのを作って、この画面を作ったわけですけど、このユーティリティ関数を作った場合のメリットというのは、あまりＸライブラリ関数とかを意識せずにすむ、もう一つは、グラフィック・パッケージが変更された場合にも、上位モジュールにはあまり影響しないということがありました。

次に、この画面を作るまでの簡単な作業を説明しておきますと、一応画面イメージというのを頭の中で描いて、それを紙の上で数値データに直して、あとはユーティリティがあって、画面上で確認するという作業ですね。これは皆さんのやってらっしゃることですので、簡単に想像はつくと思うんですけど、それを何回も何回も繰り返して画面を作っていきました。この画面を作るっていう作業は、結構面倒くさいことでして、画面制御支援グループでは画面のレイアウトですとか確認だとかに、結構時間をとられてしまいました。このＯＨＰは、今に言ったことですけど、画面設計が面倒であり、時間を費やしてしまったという問題がおこりました。

これは、数値データをおくことによって画面レウアウトの微調整ですとか、全体の調整だとかを数値で行うことによって、面倒くさい作業になったんだということです。このプロジェクトでは、平行してドキュメントを作成していたんですけど、そのドキュメント上にのせる画面データと実際のプログラムとが何回もプログラム上の画面の変更されるうちに、食い違ってきたという事態が発生しました。これは非常に困ったことで、実際にお客さんにドキュメントを渡すんですけど、実際のプログラムとドキュメントの画面が違うと。ここらへんの問題をどうにかしてアプリケーションの開発過程で改善できな

いかなということになりました。私が感じたことに、こういうものがあればそれはなくなるんじゃないかなということです。

というのは、まずは視覚的画面設計が行えるということです。これは一口で言ってしまえば、お絵書きツールですね。そういうもので画面設計すれば、数値データをおくことよりもずっと作業の効率がはかれるんじゃないかな。画面上に機能の設定ができるというのは、このような画面が必要とされる場合には画面上にここに何を出すかとかグラフを使うとか、そういう機能が要求されるわけですから、画面、ビクチャー・エディタとか、お絵書きツールの中にそういうものができればいいんじゃないだろうかということです。

もう一つは、そういう定義した機能というのとアプリケーション上位とのインターフェイスがうまくおこなえるようなインターフェイス部分もほしいなと、そのときは感じたんです。中間言語でジェネレートされるというのは、ドキュメントなんかのデータをプログラムのデータからドキュメントのデータへ移すためには、なんかジェネレーターみたいなのを用意して、そいつを通すことによってプログラムのデータもドキュメントのデータもはきだすみたいな感じでやりたいと思ったわけです．これは言語ジェネレータを用意されていて、この中間言語で作ったものをいろんな用途に使用するために言語ジェネレータがあれば便利だなと感じました。

今までの話っていうのは実際にやったことではなくて、こういうものがあれば、画面の設計とかビジュアルなアプリケーションを作る場合は、作業の効率化がはかれるんじゃないかなと思ってやったことなんですけど、これからやらなくちゃいけないのは、ビクチャー・エディタ、これは今まで言いましたれど、それについて、いかに簡単にきれいにできるビクチャー・エディタを作っていくかということを考えなきゃいれないなっていうことです。

どうしても、数値データというか、数値データで作ったほうが視覚的に作ったものよりも精度なんか良くてきれいにできるという場合がありますので、そういう何かきれいにでてくるみたいなビクチャー・エディタを作っていきたいなということ、あとアプリケーション上位とのインターフェイスなんかも、あまり情報量が大きくなくって面倒くさくないというようなものを作っていきたいなと思ったんです。

あと、ウィンドウ毎の動作定義というのもありますけど、これは例えば一つウィンドウがＯＰＥＮされた、ある数個のウィンドウがあって、一つウィンドウがＯＰＥＮしたら他の今までＯＰＥＮしていたウィンドウは閉じるですとか、ある一定の大きさに一つのウィンドウがなったら、他のウィンドウもそれに比例して小さくなるですとか、そういうものの定義ができればいいんじゃないかなということです。

あと、言語ジェネレータ、これも今まで言いましたけど、例えばXライブラリ用に作ったデータというのはそれを加工して使うよりそういうジェネレータがあれば便利だし、もっとほかの、例えば TeXですとか、あすこらへんのジェネレータも作れればいいなということです。

以上、私が考えたことはMMＩ．．．MMＩというよりもMMＩを作る上での作業の効率化ということだったんですけども、いいMMＩを作るにはこういうツールがあっても、それ、MMＩを作る人の思想みたいなのがありますので、例えばこういうツールができましてもいいMMＩができるとはあまり思ってもいません。ここでは、こういうこともまじえて、一緒に考えていきたいと感じています。

野村（ＩＣＳ）： 大変、予定時間を随分早く終わってしまいましたので、今の発表に対する質問か、もし、ありましたらお受けしたいと思いますが、今の発表をお伺いになってて、意見か質問のある方ございませんか。

三浦（ＳＲＡ）： すいません。まず、ＳＲＡの三浦と申しますけども、さっきのＯＨＰをちょっと見せていただきたいんですけども、まず最初に、ウィンドウ上でのMMＩというのをまず作られたというので、これは、どの部分をウィンドウとしてとらえて、ウィンドウ毎の動作を規定したいんで、どの部分をウィンドウととらえているかっていうことなんですけど。例えば、そのグラフの一個がウィンドウのなか、それともレイアウト全体がウィンドウなのかということをまず聞きたいんです。

古関： レイアウト全体です。

三浦： 全体。それは、ウィンドウ一個についての動作を？？？に書きたいということですか。

古関： はい、そうです。

三浦： それとですね、そのウィンドウに対して何か操作を示すということも定義できないといけないということでしょうか。例えば、今そのウィンドウに何ができるかということが実はよく分からないんですね。表示しかできないのか、だから、一回そういうデータを表示して

しまったらそれで終わりなのか、それとも何かあって、何かが起きる度に、二度や三度、何かが起きる度にそこに何かを表示するんだと思うんですけど、表示するとか、表示が変わるとか、そういうことが起こるとか、キーボードの入力とか、そういうのを受け付けたりするのかというなというのもよく分からないんですね。

古閑： この画面ではですね、キーボードの入力ですとか、そういうのは受け付けませんけど、これについて言わせていただければ、これは、数値データをリアルタイムに表示するという機能しかもっていません。

三浦： ということは、どこかからデータを読んできてそのたびに表示していかないといけないんですか。

古閑： いや、プログラマブル・コントローラですか、それにデータを読み込んできてここに表示しています。

三浦： ということは、リアルタイムに読んできて表示するということがそのどこかに書いておけるんであって、ただ画面のレイアウトだけを他に書いておけるということが必要なんですか。

古閑： いや、これを作っている過程ではそういうことはあまりやって．．．私が今いったツールでは全然できてなくって．．．。

三浦： そういうのが、欲しいって思ったから、そういうレイアウトだけが必要なんですか。

古閑： いや、レイアウトだけじゃありません。こういう数値データをどこに出すですとか。

三浦： でも、それがレイアウトでしょ。

古閑： ああ、そうです。というか、こういうグラフとかはですね、もっと道具化してですね、結局は、レイアウトになってしますわけです。

三浦： ちょっと何かはっきりしないっていうか、どこを捉えてこうしたいって思っているのか、ちょっとよく分からないんですが。ウィンドウっていうのを全体だとすると、何をそれから、ウィンドウ毎に何かをしたいっていうのを、どこにどうやって反映されるのかなっていう．．．．．。

古閑： 要は、こういう絵を書くとか、こういう線を書くとかいうのがありますね。そういうのを数値データではなくって、お絵かきみたいにしてやりたいというのが一つですね。もう一つは、例えば、こういう文字データがありますね。これは文字データなんですけど、この文字データを画面上のどこに出すようにするとか、グラフをどこに表示するですとか、グラフですとか文字データを出すというのは、どこかに機能としての道具が用意されていて、お絵かきツール上ではそれをレイアウトするというだけになればいいんじゃないかなっていうことだったですけど。

三浦： それは、やっぱり例えば何か入力する、報告するとか、そういうのを一つのオブジェクトとしてどこかグラフ上におきたい。だから、グラフの基軸をどこかにおきたい。そういう何か一つの機能を果たすオブジェクトとみたいなのがあって、それをどこかにおけるというレイアウトを？？？中間言語みたいなのが欲しいと考えたわけですか。

古閑： はい、そういうツールとそれによってできたものが、中間言語でできたらいいということです。

三浦： そのあと、そのグラフに表示する数値をどこから読んでくるかとかそういうことについては、またあとから考えるということですか。

古閑： それはインターフェス部分で考えたい。

三浦： インターフェス部分というのは、このペーパーでいうと３ページ目にある上位ＡＰとのインターフェスがなりたつことによってというのですか。

古閑： はい、そうです。

三浦： そのへんの？？？は全然考えていない。

古閑： 全然、．．．具体的には考えていないですけど、全然考えてないっていうわけじゃないんで、例えばお絵かきツールありますね。例えば、これがボタンの場合とか、ここにボタンを置きたいとかする場合ですね、その時はボタンを置いたら自動的にウィンドウが開いて、そこでそのボタンの動作ですね、例えば押さえたら何か返すですとか、テキストのボタン中に何を置くですとか、そういうのをお絵かきツールの中でできればいいなっていうふうには考えてますれど、まだ具体的にはやっていません。

三浦： はい、わかりました。どうもありがとうございました。

浜野： すいません。ＳＲＡの浜野と申します。中間言語の位置づけをちょと教えて欲しいんですけど、もっと

具体的に。作図作業フローの中にＯＫとか書いてありますが、あの中に？？？？？？？？。

古閑： とにかく中間言語の位置付けですか。どの変換からの中間言語をどのようにもたしたいかということです。

浜野： どのようにもたしたいか、ということですか。

古閑： ええ。

浜野： 絵だけだったら、その絵だけのそのデータをバシャーと？？？のメモリー・ディスクとかディスクに落とせばいいんですよね。中間言語に展開するというのは何か理由があるからですよね。次に？？？に展開したらいいか。例えばプログラミング・ファイルが接続するのが面倒くさいからオブジェクト化してパーソナル化して、レイアウトする。そして、それをＳａｖｅする。そして、それを中間言語でもって表示する。それは、何とかしたいから？？ということですか。

古閑： はい。

浜野： それがちょっと見えなかったんで教えて欲しいのですが。

古閑： 何をしたいかということですか。

浜野： 例えば、？？？？の？？？？とは逆に？？？考えたら．．．。

古閑： 実はまだあまり考えてないんです。中間言語になぜやったかというのは、さきほど言いましたように、このプロジェクトではですね、実際にウィンドウを作った時のデータというのはですね、そのままドキュメントのデータとしては使えなかったんですよ。Ｘライブラリ関数のデータと具体的にはｐｉｃのデータですね。そのデータというのはそのまま使えなくって、ちょっと困ったなっていう話があったんですよ。まあ二度手間っていう話もありますけど、もうドキュメントとプログラムのあれが違ってくると。それだったら、お絵かきツールを作ったところまでを中間言語にしといて、あとはジェネレータ用意することによって、中間言語からｐｉｃのデータですとか、ｐｉｃのｓｏｒｃｅですね、ｓｏｒｃｅはきだしてしまえばいいんじゃないかとか、Ｘライブラリ関数を作ってしまえばいいんじゃないかとか、いうふうに考えて中間言語に落とした方がいいんじゃないかと思ったんですけど。

野村： 今ので回答になっていますか。

浜野： ？？？？

三浦： すいません。ちょっと質問なんですけど。今のでちょっとわからなかったから質問。　出来上がったプログラムとドキュメントの画面ですね、それが違うから非常に不便だったから、一つのものから？？？の方にジェネレートしたいと。そういうことから、単に静的なものであればいいんですね。ｓｔａｔｉｃなものであればいいんですね。それだけの理由であれば、ｓｔａｔｉｃなものであればいい。

浜野さんの質問したのは、要するにｓｔａｔｉｃではなくて、あとでプログラムを作る時に使いたい。要するにプログラムを簡単に作るれるようにしたいとか、そういう何か理由があるのかということ。

古閑： いえ、そういう理由はありません。

三浦： ないんですか。わかりました。

野村： 他にご質問、ご意見ございましたら。ちょっと他にないようなんで僕が伺ってみたいんですが。さっきちょっと、話あったと思うんですが、ＭＭＩの部分の中間言語のジェネレータとしての位置づけで、処理ツールが欲しいというお話だったんですが、それはもっとアプリケーションとＭＭＩを独立させるというような意味で、中間言語ではなくてオブジェクトそのものを出してしまうというようなことは考えなかったんですか。

古閑： いや、そういうことは考えていません。考えませんでした。オブジェクトとかそういう中では。

野村： オブジェクトが出来てしまってアプリケーションから呼び出せる関数みたいな形で使えるものになるとすれば、画面のやりとりは画面のやりとりとして独立して使えるし、アプリケーションはアプリケーションとして独立しているという環境の方がもっといいんじゃないかなって今ふと思ったんですが。

古閑： そうですね。

野村： 何か質問とかありましたら。

会場： ちょと教えていただきたいんですが。ウィンドウがでてるＯＨＰお願いしたいんですが。いちよう、モニタと書いてありますんで数値を何か見にいって、それが反映して出てくるという形だと思われるんですけども。

それが見に行くっていう行き方というみたいなそのへんはどうなっているんですか。もうちょって詳しく。

私自身も、今現在、モニタみたいなのを作ってまして、実際にはＥｖｅｎｔＤｒｉｖｅｎで作ってるんですけども、例えばそういうグラフがガーッと変わると思うんですけど、数値が？？？？することによって、それがいつも自分でグルグル、グルグル回りながら見ているのか、それとも、そこの数値だけが変わったから何かがきてそれだけが変わるのかとか、モニタの作り方にもそうとう色々なものがあって、ちょっとユーザー・インターフェースとはかけ離れるかもしれないんですけど、そのへん制御系なんかどのようにして作っておられるのかなという興味がありまして。

古閑： このプロジェクトでは私はドキュメントを担当しているんですよ。ここらへんの話、どういうふうにしてデータをとってくるのかということは私の方では分かり辛いので、うちの平尾っていうのが来てますので、そちらに振ってみたいと思います。

平尾（ＨＳＴ）： ＨＳＴの平尾ですけども、さっきでたプログラム・オブ・コントローラも十台ぐらいありまして、それと通信をやっているわけですけども、同時プロセスが十個くらい通信計算と通信かけますので、？？？そこらへん使いまして、データをとる言語をもう一つ走らせてそれが振り分ける。あとはこっちの方は、？？？とかで待ち状態にされるので、そういう通信プロセスがもう一つ個あって、それがうまくプロセスを振り分けてデータをとりにいっているということなんです。

会場： 今の質問でもう一つきいていることがあったと思うんですけど、とったあと表示をするのは一個の数値が変わったら一個だけ変えるのか、あるいは全部また書き直すのかというあたりがあったような気がするんですが。

平尾： 一個だけです。

会場： 一個だけ。

野村： 他にご質問ありませんか。だいたい３０分くらいなんでひとまず、後で、全体の討論でやりたいと思いますので。おつかれさまでした。引き続きまして、ＫＣＳの柳瀬さんにＸｍｅによる環境インタフェースの改善ということで話題を提供していただきます。

2.2 Ｘｍｅによる環境インタフェイスの改善

柳瀬： おはようございます。ＫＣＳの柳瀬と申します。今、前に座ってまして考えてたんですけども、やはり、私のようなものがですね、発表する場ではないかなあとおもっているんですけども、なぜか知りませんけども、臼井さんですとか中野先生ですとか岸田さんのかげのミーティングで、「柳瀬、Ｘｍｅをやりなさい」と決まってしまいまして、大変光栄なんですけども、不安がいっぱいで何を話そうかなと前からさんざん考えていたんですけれども、みなさんの発表の内容を聞いてますと、理論的なことを主に言われているんですれども、そもそもＸｍｅというのは酒匂さんが開発されたものですから、私がそのような理論的なことを言うのはおかしんじゃないかなと思いまして、Ｘｍｅの理論といいますか、アーキテクチャといいますか、そのようなお話は今日はいたしません。

それと、ＭＭＩということなんですけれども、それに関しましても私はあまり知識をもってない。そうしますと、実際ですね、ここに立つにあたって、理論武装をしなければいけないなと思ったんですけども、私のような青二才が理論武装をしましても、皆さまには太刀打ちできませんし、へまをやりますのは？？？ですので、そういうのはやりません。そうしますと、今日は何をお話するかといいますと、本当にＸｍｅで本当に簡単にこのようなツールが作成できますよということと、それに対する私のしよかん的なことをお話させていただきます。

その前にですね、ひとり４０分ということなんですが、どう考えても場がもちませんので、簡単に自己紹介をさせていただきます。名前はいいとしまして、年齢は２４歳ですんでこの中では一番若いんじゃないかなあとおもいます。出身地は田舎です。最終学歴は、理工学部を卒業したということですけども、入社は昨年の四月、まだ二年目です。そうしますと、今までに大学時代に何をやってかといいますと、丁度、私のいた学科にですね、ＦＡＣＯＭのＭ－１７０というものがありまして、それをだれも使わなかったんですね。そうしまして、それは幸いと思いまして、？？？から、その１７０を独り占めにして使っていたというそういうことをやってました。

あとは、今の仕事なんですけど、入社当時はＵＮＩＸの運用と管理ということです。そもそもＫＣＳにＵＮＩＸが導入されたのは、丁度この私の入社した時と重なりまして、ＵＮＩＸを知らなければならないということで、運用とか管理とか環境のセットアップＰとかその辺りか

ら攻めないとだめだなあということで、なぜか、私もＵＮＩＸをやりたかったので、このようなところに、仕事をさせてもらっているといいますか、仕事をしているといいますか、そのようなことをしています。それが祟りまして、現在ではＵＮＩＸと名のつくものは全て私のところに回ってきまして、とうていわたしひとりではどうにもならないと、そのような現状になってます。

それで、使用マシンですれども、基本的にはうちはＮＥＷＳが２台とＳＵＮが１台あります。今日、Ｘｍｅの事例ということでお話しさせていただきますけれども、それはＮＥＷＳに対する、ＮＥＷＳに適用したお話です。それと、知っている言語を言いますね。ＵＮＩＸをやっているんですから、Ｃはまあ、当たり前で、十月頃からＬＩＳＰをやりまして、あとＣＯＢＯＬとかＦＯＲＴＲＡＮも学生時代からゴロゴロ、ゴシゴシやってましたんで、まあ学生時代にやっていたということは、あまり役に立たないということなんですけど、知っているということで。あと、ＢＡＳＩＣはパソコンを趣味でやってましたので、それなりに分かります。あと趣味はＪＵＮＥＴということで、これ以上言う必要はないと思います。特技としましては、季節がらですね、松茸狩り、これはだれにも負けないと確信しています。ひとりが一本見つける間に、二、三十本は見つけます。まあ、そのようの私ですけれどもよろしくお願い致します。

さて、本題の方なんですけれども、Ｘｍｅということですけれども、このなかは、参加されている方の中にもたくさんＸｍｅを使われているような方もいらっしゃると思うんですが、そもそも、馴れ初めとしましては、昨年の二月、ＳＲＡの酒匂さんですけども、こういうツールがありますよと、欲しい人は言ってください、さしあげますよと、言ってくださいまして、そういう言葉は私は見逃さないんですね。すかさず酒匂さんにおねだりしまして、「くださーい」と言っていただいたということです。そうしまして、第一印象としましては、すごい。感動したわけです。そのわけは、あとでちょっといわせていただきます。Ｘｍｅって、結局なんだろうということなんですけども、酒匂さんがですね、マニュアルとか前回のＳＥの長野ワークショプなんかにおきましてはですね、Ｘウィンドウ環境下で作動する簡易ユーザー・インターフェイス構築ツールといわれているんですね。確かにそうなんですけど、初めての方には一体何だろうというようなこともあると思います。そうしますと、酒匂さんのもう一つのアプローチとしまして、ツールの統合化という観点からも作成されているんじゃないかなと思いまして、私がＸｍｅって何だろうときかれた場合、このように言うことにしています。それは既存のツールゾーンに対してマルチウィンドウを使ってインターフェイスを容易に構築できるツール。確かにこれだけでしたら、Ｘｍｅの全てを物語ってないんですけれど、私の今日お話しする内容はこれくらいの前提知識でいいんじゃないかなと思っています。

それと、Ｘｍｅでどういうことができるのということですけども、これは実際ですね、パネル制御とありますけど、これは実際ウィンドウ、Ｘｍｅウィンドウ配下のウィンドウ、パネルというものが実際、目に見えているわけなんですけど、このパネルですね、目に見えている部分、ですからマウスでクリックしましたよというものをＸｍｅのプロセスにおくってやたり、そのようなことを管理するのがパネル制御といいます。そして手続制御。この簡単な手続、演算ですとか、条件判断ですとかを制御する部分。ですから、あるボタンとかってですね、どういう定義付けするかというのも、こういうところで定義されている。あとプロセス制御ですけれども、これは既存のツールですとかコマンドを実行させたり、その制御、実行にあたっての制御をするところです。あと通信制御でありますけども、ソケット、これは言うまでもありません。Ｘｍｅ同士が通信できるというようなことです。このようなことができるんですけども、実際どうすればできるのということになりますが、それは定義ファイルというのがありまして、それはＸｍｅ独自の難解な記述言語で、ゴリゴリゴリゴリと書いてやるんですね。そうしまして、そのファイルをＸｍｅのインタープリタで実行することによって、何かのツールが作動すると、そのような仕組みになってます。

それでですね、定義ファイルといいましたけども、その簡単な例をあげてみます。例えばですね、下からいきますけども、これを実際の画面とします。そうしますと、Ｘｍｅでですね、このパネルを作りましてここにテキスト、Ｘｍｅテキストを表示させる。そしてボタンを二つ作りまして、こちらにはａｃｔｉｏｎという文字を書いて、こちらにはｅｎｄと。このボタンをａｃｔｉｏｎのボタンをクリックすると、ただ単に、Ｘｍｅウィンドウの Ｘｔｅｒｍが開くと、ただ単にこれだけの動作なんですけれども。それから、ｅｎｄボタンを押しますとＸｍｅのプロセスが終了すると。要するに、これがみんな消えきゃうという、そのような簡単なものなんですけども、

それをこのような形で実施するわけなんですね。ＵＮＩＸをやっておられる方でしたら、もう、簡略型のファイルなんですけども、まず一番上にＳｅｌｆというのがありますけども、そのＳｅｌｆでですね、この実際の裏の、後側のパネルですけれども、これの大きさですとか位置ですとかを定義する。次に、Ｂｕｔｔｏｎとあります。Ｂｕｔｔｏｎと言いますと、実際にこの、このようなですね、このようなて言ったらあれですけども、まあＢｕｔｔｏｎを作成するところなんですけども、これでＸ座標、Ｙ座標と幅と高さですね。そしてこのＢｕｔｔｏｎの定義です。ですから、ＴｅｘｔとしてＥｎｄというＴｅｘｔを中に書きなさいよと。そして、使用するｆｏｎｔは何なのということで、ａｃｃのどうのこうのというｆｏｎｔ指定している。そうしますて、このボタンが押された時に、どのような動作をするかということをｔｖでｑｕｉｔということでＸＭＥを終わりなさと、そのような感じで書くようになってます。そしてＴｅｘｔ欄は、文字を表すための記述をしてるところです。あともう一つ、ＸＭＥがですね、既存のＵＮＩＸコマンド等を呼びたすということに大変優れていまして、今の例で言いますと、ａｃｔｉｏｎボタンを押しますとｊｔｅｒｍ、ｘｔｅｒｍが作動するという。このボタンの定義はですね、ただ単にｕｘでｊｔｅｒｍ　－　ｎ　ｔｅｓｔ　と書いてあれば、もうそのボタン、ａｃｔｉｏｎボタンガ押されれば勝手にウィンドウが作られるというようなものです。定義ファイルというのは、一見、難しいんですけども、慣れてしまえば簡単だなというような気もします。ですから、ＸＭＥで何かものを、アプリケーションを作ろうとしますと、このようなものをズラズラズラーッと書くわけですね。このような繰り返しを。そうしまして、一つのアプリケーションをアプリケーションとして作るということです。ですから、ボタンの関係ですとか、このボタンが押されたら何をするという定義付けですとか、ボタンどうしの関連付けですとか、もちろんテキスト処理もできますし、グラフィック処理もできると。そのようなものです。そうしますと、実際に．．．．。今日、お話している内容は私の原稿にほぼあってますんで、そちらを参照して下さい。

実際、私が簡単にＸＭＥを用いまして、ちょこちょこと作ったものはそこは説明させていただきます。これは、酒匂さんのＸＭＥをいただいた時にいっしょについてきたもので、私が作ったものじゃないんですけれども、まずこれは、ネットワークで接続されているリモート・マシンへのアクセスを管理する一つのツールと言っていいんじゃないかなと思います。こんどは、ここにｋｃｓｎｅ１とかずらずらずらと書いてありますけど、これは実際に相手のマシンなんでですね。そうしまして、このボタン、例えば今ですと、ｋｃｓｎｅ２というボタンをクッとクリックしますと、ウィンドウが生成されて、実際にアグロウンした形でででてくる。これはもう、どのようなコマンドを発行したかというのは、言うまでもないことなんで省略させていただきます。もう一つですね、大阪大学というボタンがございますけども、このをクリックしますと、ｔｉｐが呼び出されていまして、自動ダイアリングされて、阪大の方のサイトにつながるというようなものです。

あとのこちらの定義はですね、実際、ウィンドウを生成する場合のｊｔｅｒｍとかｘｔｅｒｍの大きさですとかを定義する部分です。これは大変便利でして、これは私は必ず出してます。ウィンドウ上で。次にですね、私が実際におこなっている作業といいますと、さきほど言いましたように、色々あるんですが、プログラミングも多少やってますんで、そのあたりの支援するようなツールが何かないかなあと、むりやりプログラミングに必要なコンパイラですとか、デバッガですとか、マニュアル参照ですとか、エディタをまとめたものです。今ここで、ファイル名を指定するわけなんですね。そうしまして、ｅｄｉｔボタンを押した時、そうしますとウィンドウが開いて実際にｅｄｉｔ　できる。そのような場面です。それに、あとｐｒｉｎｔでしたら、このファイル名のものが実際にプリントされまして、ｌｅｓｓでしたらこのファイルがｌｅｓｓで参照できる。ｍａｎとありますけど、ｍａｎ、マニュアル参照ですけども、ソケットというものに対してｍａｎを見たければ、ｍａｎボタンをクリックしてもらえば、ウィンドウが切られてソケットに対するマニュアルが出てくるというような感じです。それとですね、じゃあここでＬＩＢＲＡＲＹとありますけども、実際にコンパイルするにあたって、仕様書ＬＩＢＲＡＲＹを表示していまして、利用者はですね、必要なＬＩＢＲＡＲＹをちょんちょんちょんとクリックしてやればそれが勝手にコンパイルの時にくっつくという感じです。ですから、今選択しているものはといいますと、ＸとＸｍｅｎｕとｔｅｒｍｃａｐとｓｊ２ｌｉｂですね。これがコンパイルにあたって使用されるＬＩＢＲＡＲＹということを指定しています。そうしまして、これだけのことを設定しましてあと、Ｏｂｊｅｃｔですね。どの

ようなものを生成するか。そう、ただ単にファイル名ですけれども、それをここに指定してやりますとｄｂｘもある。？？にでていますね。選択していくと。そしてｃｃとやりますと、ｂａｃｋ ｇｒｏｕｎｄでコンパイルされまして、もしエラーがあればウィンドウを開いてエラーメッセージを出すと。何もエラーがなければ、そのまま終わると。それで、コンパイルをしている時は、このボタンは網がけのような部分、網がけになっています。あとＲＵＮとあるんですけども、これはちょっとあまり使えません。あとｄｂｘというボタンがあるんですけども、実際にこれでコンパイルなんかしてますと、ｄｂｘなんかも使いたくなるんで付け加えたんてすけれど。例えば、ｄｂｘ、先程のコンパイルでｃｃの時に－ｇ（ｇフラッグ）がでましたので、このｏｂｊｅｃｔに対するｄｂｘということになるんですが、このボタンを押してやるとウィンドウが開いてｄｂｘの枠になる。あともう一つ何か欲しいなあというとで、ｌｅｓｓで別のウィンドウで表示してやると。ですから、これのこちらと、いまちょっとエラーメッセージを表示していませんけど、エラーメッセージの画面とｄｂｘの画面、この三つを使ってデバックすることができるというような簡単な仕組みです。実際、これを私も使っていますけど、テスト、プログラミング・テストぐらいにしか使えません。正直なところ。実際のところ、アプリケーション組む段階になりますと、分割コンパイルどすとか、そのようなことを致しますので実際あまり使えないなと思ったりしています。

次にＸｍｅｊｕｎｅｔということで、ｊｕｎｅｔが提供している機能、要するにメールとニュースを読むような、読む機能を統合化、まとめてみたものです。

まずですね、このウィンドウ、これは普通のウィンドウ何ですけども、このなかでこれから自分がしたいようなことを選ぶんですね、マウスで。この場合ですと、ｓｅｎｄ ｍａｉｌとありますから、メールを今から送信したい。そうしますと、この画面が出てくる。そうしまして、この画面でメールを送信することができるというわけなんですけども。まずここでで、ファイル名というのがありますが、そのまえに、ここにｓｅｌｅｃｔ ｆｉｌｅがありますけども、Ｘｍｅのですね、ファイルのディレクリ構造を可視的に表示することができるんです。丁度、ｓｍａｌｌ－ｔａｌｋのブラウザーのような感じでディレクリとかファイルを表示することができまして、それによってマウスでクリックするとぴゅっとこちらに入ってくる。そのようなこともできます。もし、ファイルが、既存のファイルがなかければ、新しいファイル名を勝手にここに入力してやればよいわけですけども、そうしまして、ここでｅｄｉｔを押してやりますと、ただ単にこのファイルに対するｅｄｉｔの画面が生成される。そして、どうたらこうたらとゆう感じでｅｄｉｔして終了しまして、まあそのですね、今から送るＳｕｂｊｅｃｔ、主題はなんなの、ということでここで何か記述しているわけですね。ＳＥＡｉｎＫＯＢＥの原稿とゆうことで。そうしまして、誰に送るのということで、相手の名前を選択してやればいいということなんですね。ですから、今の段階でしたら阪大の中野先生ですとか、岸田さんですとか、墨田さんと、あと桜井さんと臼井さんにメールが届くようにセットアップしている。最終的に、Ｓｅｎｄ ｍａｉｌというボタンを押すと、勝手にメールが送信されます。実際、どのようなコマンドを呼び出しているかということは、言うまでもないのでやめておきます。

次に、ニュースを読みたいよと、？？？です。ｒｅａｄｍというボタンをクリックしてやりますと、これはただ単にデータ文を動かしてやって、ｍａｉｌコマンドを発行してやるだけ。それで丁度、桜井さんからメールがきていてよろこんでいたところです。それともう一つｊｕｎｅｔで、ニュース機能が提供されてましてますんで、それを何らかの形でとりまとめたいなと思って、それをこういう形で表してみました。それでｊｎｅｗｓと選択しますとこのパネルがｏｐｅｎされまして、実際自分の読みたいところだけを選択していればいいんですね。そうしてれば、ここに反転といいますか黒くなりますんで、実際それだけ、そのニュースグループを選択したと。もしこの中になければ、ここに実際書き込んでやれば、そのニュースグループを読むことができると。そうして、セットアップしまして、ｊｎｅｗｓというやつを押しますとただ単にウィンドウが開いてｊｎｅｗｓを実行しているだけと、このように簡単な仕組みです。あとこれは、先程ｊｕｎｅｔ のｍａｉｌ送信のところで言ったのと同じですけれども、これは社内用のｍａｉｌ送受信用のものです。これは全く同じです。あとアドレスの指定です。部長・次長のところにクリックしましたんで、この四人に対してはメールが届くと。あと、ｏｎになっている、上向いるのがｏｎで下がｏｆｆですから、クリックして上を向いている人にメールがとどくという感じです。

それで先程言いました、ディレクトリの可視化という

ことで、Ｘｍｅ、先程言いましたけども、このような感じで表示してくれます。ですから、ユーザーはですね、今どこのディレクトリにいるのか、その中にどういうファイルがあるのかということをいちいちコマンドをたたかかなくてもすむ。そうしまして、どこかを選択してやりますと、それがちゃんとセットされるという、そのような感じになっています。そうしまして、このように簡単にいくつかのツール、これ以外にもちょこちょこありますけど、作ったんですけれども、それの仕様版的なことをですね、ちょっとお話させていただこうかなと思います。

まず、Ｘｍｅの長所、いいところなんですけども、まず、言語を知らなくても作成が容易だということ。先程、ｔｅｒｍｃａｐ型の定義技術ファイル、なんかわけのわからない文字列が並んでいたように思いますけども、その作成を支援するＸｍｅｄなるツールが付随していると。これによりまして、ユーザーはあるファイルの中身をいちいちＶＩか何かで作成しなくてもいい。これが実際の画面なんですけども、これがＸＭＥＤのメインの画面でして、今はこの画面に対して編集をおこなっているという感じです。そうしまして、編集ボタンを押して、今ですとｎａｋａｎｏ@ｏｓａｋａという中野先生のボタンですね、ボタンを定義付けをおこなうのに編集しているところなんですけども、これを選択しますと下のようなパネルが作成できる。そうしましてここに、順番に合うように定義してやればいい。Ｘ座標、Ｙ座標と高さと幅、幅と高さですか、そのように定義してやってもいいですし、実際ですね、ここをマウスでクリックしながら動かすこともできますので、どちらでもいいです。たとえば、マウスをクリックして動かしますと、その結果がすぐこちらに反映されますので。

それとあと、内部的なことをちょっと申しますと、ｔｉｍｒｏｍ１２というＦｏｎｔを使います。そしてＴｅｘｔ、このボタンに対するＴｅｘｔ表示はどうしますかということで、ｎａｋａｎｏ@ｏｓａｋａというそのようなものを表示させます。そうしまして、そのボタンが選択されたときにどうするかという定義なんですが、ここは実際はアドレスが入っているわけなんです。そうしまして、Ｘｍｅのｓｈｅｌｌ変数のような概念がございまして、Ｘｍｅのａ１という変数にもし選択されたならば、このボタンが選択されたならばこのアドレスをセットしなさい。選択されなければ何もしない。このような定義付けをおこなっているわけです。それど、このツール、ＸＭＥＤの大変いいところはここで変更加えますと、直ぐさそまこちらに反映されます。実際、目で見ながら、確認しながらユーザーは定義付けをおこなうことができますので、大変便利じゃないかなとおもいます。ですけれども、実際、これで作成するといいますか、複雑なことをしようと思いますととうていこれでは追い付きませんで、ＶＩでやらなければならないという結果になはなりますけれども。

次にまた一つの長所としまして、対話型のインターフェイスが構築できるということですけれども、先程お見せしましたもの、パネル、ボタンですとかが無造作に表示されていましてユーザーは何を押したらいの、どうすればいいのということになりかねないんですけども、実際に対話型のインターフェイスの構築は可能です。といいますのは、ＸＭＥ自身がですね、内部プログラム という手続制御ですけれども、そういう仕組みをもっていますし、あと？？と連動することもできますのでそのボタンに対するチェック機能をですね、例えばこのボタンが押されたけれども何々がセットされていなければキャンセルしなさいとか、順序立てのようなこともできますし、その結果ですね、変な動作や変なクリックなどを設定なりをしますと何らかの形でユーザーに知らせるというような、テキストで知らせるというような事が可能です。ですから、本当のメニュー形式のですね、次何しなさい、次あれしなさい、次こうしなさいというようなツールも作成可能かと思います。

あと、今日のテーマで可視化ということで無理やりじゃないですけども、つけてみたんですけれども、これは先程の私の作成例でも明らかなようにコマンドですとかツールに対する呼び出し、これには絶大な効果を発揮できます。実際、目で見ながら確認できますので。それとあと、これらのオプション指定ですね。これも大変優れているんじゃないかなあと思います。それともう一つは先程言いましたファイルのシステム構造ですが、これもちゃんと、グラフ化といっていいのか分かりませんけども、このように視覚的にＵＮＩＸのファイル・システムをとらえることかできるというこのようなところじゃないかなと思います。あと、いいところもあればちょっと不満だなという部分もございまして、それはまずですね、パネル・デザイン、ウィンドウ設計ですれけども、画面設計ですれけども、そのバリエーションが不足という、ですからボタンとかスイッチの数がちょっと少ないんじゃないかなとい思ったりしています。

例えばその他ですね、ポップアップメニューの機能がついてないんですね。ですから、ポップアップメニューの機能を是非ともつけていただきたいと思います。あとロータリー・スイッチなんですれども、無理やり作ろうと思えば作れるんですけれども、ちょっと簡単に実現できないんで、簡単に実現できるようなロータリー・スイッチに、何かもつけて欲しいなあと思ったり致します。それとあと、複数のパネル定義ができないということなんですけども、先程のｊｕｎｅｔのどうのこうのというのかございましたけれども、何か一見、複数、こいつからこいつを呼び出しているような感じ、二枚のパネルを制御しているような感じなんですけれども、実際これはどしているかといいますと、このボタンを押されますと、もう一度ＸＭＥを呼び出してやっているんですね。どうことかといいますと、ＸＭＥの定義ファイルというのがございましたが、あれでは基本的に一枚のパネルしか制御できないんですね。そうしまして、定義ファイルというものがありますけども、ＸＭＥの一つのプロセスは一つの定義ファイルしか管理できない。ですから、ＸＭＥの一つのプロセスでは一枚のパネルしかできません。現状では。ですからここでこさえた場合、もう一度ＸＭＥのプロセスを起こしてやって、それでこのパネルを作っているということをしています。ですから、実際裏でＸＭＥのプロセスが二つ走っているということです。あとパネル・デザインは大変難しいとありますけども、上で容易であるといいながら下で、矛盾しているようとお思いかと思いますけれども、確かに容易なんですけども、単にＸＭＥで移動とか拡大、縮小、容易にできるんです。しかしながら、ドット単位で指定できますものに、とのような大きさにしよう、そのボタンをどこに配置しよう、ましてフォントの数も２００種類ほどありますし、ボタンも少ないと言いながら４，５種類ありますんでどれを使おうかということで、どのようにデザインするかということが、大変難しくて困っております。ですから、何らかのかたちでこのあたりを支援できるようなものがあればなあと思ったりもします。

次に、無理やりわがままに、まとめにはいります。先程から見ていただきましたように、ＸＭＥの魅力といいますのは既存のツールに対してマルチ・ウィンドウですとか、マルチ・フォントですとか、マウスを使ったインターフェイスが容易に構築できるという、これに尽きるんじゃないかと思ったり致します。そうしまして、実際の操作環境の改善といたしまして、いままでＵＮＩＸのインターフェイスはｓｈｅｌｌというコマンド型のインターフェイスだったんですけども、まあ、だったんですから、実際にはユーザーが頭で何かごちゃごちゃごちゃごちゃ考えながら、いいのか悪いのか入力して、それからいいなと思って実行する。そのような感じだったんですけど、ＸＭＥをちょっと利用することによりまして、目で見ながら考えて実際にパネルにおとして目で分かりますので、目で見ながら考えて選択して、そして実行するというそのようなユーザーに対してはコマンドですとかオプションですとかそのようなところが可視化されているんじゃないかなと思います。それとＸＭＥのここで提供インターフェイスとありますけれども、ＸＭＥで作成したアプリケーションが提供するインターフェイスなんですけれども、これは基本的にはＸＭＥは万人に使いやすいインターフェイスを提供するというものじゃないなと思います。といいますのはですね、アプリケーションのユーザー・インターフェイスというのは実際、それを用いる、利用する人が考えるのが一番なんですよね。ですから、ユーザーそれぞれによって、そのユーザー・インターフェイスの仕様も異なりますし、例えばですね、私でしたら、先程のボタンのこんなボタンの形式で何の説明書きもいらないようなこんなインターフェイスでもいいんですけども、うちの某部長はこんなものはだめだと。要するに、マシンに座ったらマシンが何でも教えてくれる。ですから、マニュアルがいらない。次、ああせい、こうせいとシステムが親切丁寧に教えてくれると、そんなわがままを申しました。まあ、ユーザー、使うユーザーにとって色々様々ですので、ましてＸＭＥは先程言いましたように、このＸＭＥＤというツールを使いまして自分の好きなように設定できますんで、自分の使い易いように作ったわけすから、自分なりには大変使い易い。私のこれはｘｉｎｔとやった後のデフォルトの画面なんですけれども、まずこ最初のネットワークのあれが必ず出でいます。そうしまして、このアイコンですけれども、これは先程言いましたｊｕｎｅｔで、ｊｕｎｅｔが管理しているあのツールですね。あれのアイコンです。このような感じで私の生活にはＸＭＥは欠かせない。それともう一つですね、デモと言いますか、社内ではＵＮＩＸ　はまだまだマイナーなものでして、説明する、ＵＮＩＸを説明したりする場合が大変多いですけれども、そうした場合ですね、そけで私があそこに座ってですね、マシンの前に座って、コマンドをかちゃかちゃたたいいて、ユーザーは何のことか、ユーザーといいますか聞い

ている方は何のことやらさっぱり分からないという現状に陥るんですけれども、こういうかたちで目に見えたものを、実際、マウスのクリックですとかそのようなかたちで説明しますと、こちらも説明しやすいですし、聞いている方も理解できるんじゃないかなと思ったりいたします。そんな感じで私は大変気に入って、XMEを気に入って使っています。まだ表面的にしか使っていませんけれど。

それと、もう一つ最後にですけれども、XMEの期待ということで、酒匂さんと色々メールのやりとりをさせていただいているんですけれども、その中でですね、汎用的なX11版のベータバージョンがもう今週には完成する、入ってきますよということで、またおねだりしようかなと思ったりしています。そうしまして、X11になりまして色々と改善とか拡張とかされていますので、そのあたりも大変期待しています。実際、そのようなどこがどうなったとか、どこかどのように改善されたとか前もって分かっているんですけれども、私が言う必要はないと思いますんで、酒匂さんがもうすぐどこかで発表されると思いますんで、今回はやめさせていただきます。そうしまして、大変内容がMMIということと比べますと大変薄いんですけど、とりあえず、終わらせていただきます。

野村： 有り難うございました。それでは、質疑応答いはいりますが、とりあえず松茸とり以外のところでご質問、ご意見お願いします。

田中正則（SRA）： SRAの田中です。正則の方です。私もXMEをちょっと使わせてもらっているんですけれども、MMIにとって結構、レスポンスというのが重要な位置に入ってくると思うんですけども、その点XMEではどうでしょうか。

柳瀬： 私が使っている場合ですけれども、確かにですね、コマンドを叩いてからXMEが起動するまでには若干遅いな、レスポンスが悪いなという気もいたします。といいましても、私は基本的に一台のマシンを占有していますので、あまりそのような、遅いとかそんなことは考えたことはないです。

田中： だんだん人というのは、欲求が高くなってきてきっと柳瀬さんも満足いかなくなると思うんですけども、そんなこと柳瀬さんに言うようなことじゃないと思うんですけど、酒匂に言うことだと思うんですが、要望としてXMEのコンパイラ、今インタプリタなんで、コンパイルして、何か中間ファイルにはいって、それをRUNと実行かけるようなかたちになればいいなというふうに個人的には思っているんですが、その点、どうでしょう。

柳瀬： そうですね、私自身はあまり深く考えてないんですけども、既存のインタプリタ形式でもいいんじゃないかなと思ったりいたします。それから、私がですね、日頃、不満なりですね、要望のあったことはすかさず、酒匂さんにメールで出すようにいたします。

田中： わかりました。どうもありがとうございました。

栗原： SRAの栗原です。補足が一つと、意見が一つあります。今の柳瀬さんの説明の中でXMEもっている機能というのが言われてましたけど、ディレクトリを見せるとか、スクロール、ボタンというのは、実はマップの世界で有名なマイケル、夜の名前のマイケル、昼の名前は違う名前なんですがけど、そういう人がいまして、そういう人が何でもマップでないといやだといって、ライブラリを作って、それを組み合せてXMEをつくった。ですから、XMEのああいう見せかけの不可なる部分というのは、もう一人有座という人間が絡んでますので、酒匂と並んで有座というのは変なやつだと理解してます。

もう一つ、意見なんですけど、私はXMEを使ってません。主にというのはあれだけど、わたしの場合はどっちかというとウィンドウ・マネージャーの方に機能をおいて使っているんですけども、私が理解しているXMEというのは画面、Xのプログラムを作りましょう、画面をどのように定義しましょうか、てやるとにかくXMEのレベルではXLIBしかないから、そこのレベルだけはCでコーディングしなきゃならない。出してみてあんまり良くないな、このボタンを押さずに？？？いくのはやだな、これまたコーディングを書き直す。これはすごい手間なんですね。それを何回かやっているうちにこれではいけないというので、画面を楽に定義というか、確認、動作を確認して設計できるような何か仕掛けを作りましょうというので、僕は始まったと思うんですね。そういう意味ではあるシステムのマン・マシン・インタフェースのプロトタイプを作るためにはどういうプログラムが、ツールがあればいいかなと考えて作られたプロトタイプだと思うんです。

ですから、定義がしにくいとか、本当は画面の定義は

もっと楽にできたほうがいいんですけど、そのへんはたぶん、最初の計画ではメインではなくて、動きを確認したいということ思うんです。ですから、XMEは確認するための手段だったと思うんです。しかし、それができてみると、実は、結構うけたりして、いつのまにかそのXMEの定義ファイルを作るのが目的になっちゃって、なんか？？？したかたちで、画面定義のためのツールを作ったのに、その画面定義をいかに楽にして作るかという変な話にもなっていると思うんですけども、一つXMEDというのでたぶん、詳しいことは僕、よく知らないんですけど、XMED もXMEで書いてあるんだと思います。同様に繰り返していけば、先程言ったいろんな人のユーザー・インターフェイス、いろんな好みがあると思うんですけど、とにかく一種類では絶対できないと思うんですけども、そうすると自分に合わせたユーザー・インターフェイスを構築していくようなXMEの定義ファイルをジェネレイトするXMEの定義というのが作れるはずなんですね。常に効率は悪いかもしれませんけど。そのへんについては、どういうふうな、あるいは、そういうことをなさっていますか。

柳瀬： まだそこまでは、いたしておりません。先程、有座さんのお名前がでてきたりしましたけども、私もそういうところちょっと、内緒ですね、SRAさんの。これは有座さんの作られたものだということくらいしか分からなかったんです。すいませんでした。

桑名（NTTソフト研）： NTTソフトウェア研の桑名です。実は私も同じようなツールを作ったことがありまして、ユーザーに対するイージー・ツールというようなイメージなんですけど、確かに、画面定義ツールとしましては非常に似ていましていいんですが、私の作ったツールでも、当初ですね、？？？におけるやっている手順とか作業がですね、そういうものを自動化できるものが支援がないかな、それからマン・マシン・インターフェイスというものがありえないかなというのを当初考えまして、作ったんですけども、結局、うまくいかなかったという点があります。今も例えば、言われたんですけど、テストの過程においてはですね、エディタをOPENして、DBXをOPENして、実際は使えるようだけども使わないということとして言われたんだと思うんですけども、実際、私もそのように部会？？の終わったあと、作りまして、エディタをOPENしてDBXをOPEN、そのいったかたちのものをポップアップ・メニュー形式で作ったんです。ところが、いざ使ってみようかとしますと、いろんな人に提供したんですけども、結局、あまり使っていただけなかった。それも何か原因があるんじゃないかなと思いましてお聞きしたい点がい一点あるんですけども。もう一つ対話型インターフェイスをこれから作るようなこと、そういうふうな機能もあると言われとましたけども、実は対話型インターフェイスも作りました。作ったんですけども、これもうまくいきませんでした。どうしてかなと色々よく考えてみたところですね、何かソフトウェア・パックにおける機能の話なんかもでたんですけど、プロセスとか実際の設計作業なんかにおけるオブジェクトは何なの、そのプロセスの断片のリレーショナルは何なのかということをきちっとデータをとってですね、考えないで、かなりエキスパートとかいろんな人の話を単に聞いて、思いつきのようなかたちで、スクリプトを書いたりなんかしたためですね、結局、みんなに本当にうまく使えるようなものにならなかったといっているんですけども、この今のXMEとしては確かに画面定義ツール、マン・マシン・インターフェイス定義・ツールとしては非常にすばらしいものだと思うんですが、本当に実際のデバッグとか、テスト支援のツールですとか、あと対話型インターフェイスを構築していくにはこれから何が必要なのかなというところを、今まで作った経験で意見があったら教えて欲しいんですけども。

柳瀬： そうですね、私自身アプリケーションを作った経験があまりございませんので、一概に言えないのと、もう一つXMEでどこまでできるのかというのがまだ見えてませんね、申し訳ありませんけど、一概にはいえないんですけど、ただ単に先程からプロトタイプですとか、画面の定義という観点で見れば大変優れているだろうと思います。答えになっていません。

野村： じゃあ、次の方どうぞ。

野中（日本電気）： 日本電気の野中です。まえにSRAの方が、どっちかって言うとユーザー・インターフェイスよりも画面の動作の確実だときいて少し安心したというか、そういう気がしたんですけど、基本的な考え方としてshellが動いてて、そしXMEがshellに対して何かくわせるというかたちですよね。結局、僕も今までのアプリケーションて感じなんですね。というか、今までのアプリケーションを起動して、単にそれにどう

いう引数を与えるかという制御だけしかやってないわけなんですね。簡単にできれば、それはそれでいいと思うんです。やっぱり、ずっとだんだん話を聞いていると落ち込んでくるというか、悲しくなってくるというか、そういう気がしますね。これはこれでいいと思うんですが、みんな悲しいでしょっていうのが僕の意見で、もっとすごいものを考えましょうよとか、誰か考えてくださいよとかそういう気がしました。

柳瀬： そうですね、確かにその通りだと思います。すいません。ちょっと、最初にひとこと言い忘れていたんですけれども、第一印象が感動ということを書いていたんですけども、なぜ感動したかといいますと、例えば、ちょうど二月ごろ頂いたんですけど、そのころ私が何をしていたかといいますと、ちょうどC言語でXをやりはじめていた頃だったんですね。ですから、C言語でゴリゴリ組むんではなくて、このようなかたちで簡単に作成できるということで、大変すばらしいなと思いました。もちろんですね、これ自身をC言語で書いても１０００ステップくらいですから、簡単にできますからいいんですけども、実際、これをCで書こうとしたら、たかがこれくらいのものわざわざCで書かなくても、邪魔くさいなあという感じで、そしたらXMEでやろうかなと思ったりしています。そんな感じです。

海尻（信州大学）： 信州大学の海尻といいます。三つほど。いやに簡単なんですけども。ウィンドウ・マネージャーの上で動くんですね。？？した上でというか、ウィンドウ・マネージャーを起動させた上で動くということ。

柳瀬： そうです。

海尻： 次の点は、ＳＵＮＶＩＥＷですとウィンドウにある機能はだいぶん、絶対低い、低いといいますか、単にウィンドウ・マネージャーですけども、ツール？？？？、その書いたあとの状態を保存してそれで起動できると。ああいうふうなものがあれば、できれば、ありのカストマイズがかなり簡単ではないかなと、別に柳瀬さんに言うようなことじゃないですけど、という気がしました。　　もう一点は、今後もしももっと広範囲に広げることができれば、何らかのかたちで公に、Ｘウィンドウにつけるというわけにはいかないと思いますけど、捜していただきたいという希望です。

柳瀬： まあ一点目はいいとしまして、二点目ですけれども、早速、酒匂さんにメールでもうっておきます。三点目なんですけども、私自身、もう少しXMEに頭を突っ込みたいなあとも思ったりしていますし、そこから得られたものを何らかの形で別のものに反映したいなあと思ったりもしています。ですから、いつになるかわかりませんけども、また、お機会に発表させていただければなあと思っています。

野村： ちょっと、職権乱用して、些細な質問をしたんですか、さっきの発表の中でパネル・デザインのバリエーションか不足だということがあったと思うんですが、具体的に、今欲しいものはどんなものかということですね、それとロータリー・スイッチというのを初めて聞くんで、それがどういうものなのかを伺いたかったんですけど。

柳瀬： はい、ロータリー・スイッチですけれども、ただ単なる私の言葉なんです。申し訳ないんですけど。今のボタンを見ていますと、ＯＮ，ＯＦＦ、基本的には、ＯＮ，ＯＦＦの二種類しか制御できないわけなんですよね。それでも、何回かクリックすることによって、ある一定の順序に従ってロータリーしていくようなかたちのスイッチということてす。それでもう一つ、欲しい機能ということなんですけども、そういうものも欲しいです、あと先程言いましたポップアップメニューも欲しいですし、まだ、グラフィク機能が大変貧弱と言ったら失礼なんですけれとも、弱いなあという思いますので、そのあたりをもう少しサポートしていただければなあと思ったりしています。

野村： それでは、コーヒーもきているようですので、ここで１５分くらい休憩したいと思います。あすこの時計で１１時３５分まで休憩したいと思います。

野村： それでは、戻って来ていない人たちもちょっといるようですけども、その人たちは参加費をなげていると思って、引続き進めさせていただきます。三番目の話題としてですね、プロトタイパーズ・ワークベンチでいいんですか。

塩谷： はい。

野村： 塩谷さんの方から発表していただきます。

2.3 PWB

塩谷（ＳＲＡ）： 初めに、最初から謝っておきます!このような議題を与えられた、私は、つもりでした。そ

れで、どうしようかなあと思って、どんなふうに話をしていったらいいのかなあと思って色々考えまして、昨日、おとついですね、ここまで神戸に来ても今一つ分からなくって、野村さんたちとか、小前さんとかに話して、出たとこ勝負でだいこうということと思ったんですけど、なんとかＯＨＰを使わなきゃいけない。それで、きのう情報交換パーティの帰りに、ＯＨＰとペンを手に入れまして、今日は風邪もひいているし、調子も悪いし、早く帰って書こうと思って帰りかけたんですけど、ついついふらふらといってしまいまして、ホテルに帰るのが遅くなって、それから勢いにまかせて書きましたんでただでさえきたない字がますますきたなくなって、ただでさえ下手な絵かますます下手になってますので、そこらへんのところは適当に説明をおりまぜながらいきたいと思います。

一応ですね、Prototyper's WorkBenchをもとに私がいいたいことを言うということにしたいと思います。それで、マルチ・ビュー、 ＭＭＩの中のマルチ・ビューということでしたので、それは恐らく、ＰＷＢの中で６つ、一応６つといっているんですけど、６つのモデルを並列に動かして、評価していくんだというようなうたい文句で作っているんですけども、その６つのモデルということでマルチ・ビューというものがあったんだと思います。それで、私の考えるマルチ・ビューということなんですけども、どっかの会社の藤田さんのような方がいらっしゃるときれいな絵が書けるんですけども、なかなかそういう人が身じかにいませんでしたので、しかも、夜書いたんでこんなになって．．．、一応ですね、マルチ・ウィンドウの絵のつもりなんですね。ここらへんにウィンドウが出ているつもりなんです。ここらへんは例えば電話の絵ですとか、これは何ですか、熱気球ですね、これはディスプレイとしてのつもりです。これはうさぎの絵が書いてあるつもりです。それで、書いてあること自体、全然、意味はないんですけど。要するに、言いたいことはマルチ・ウィンドウはこんなふうにいっぱいウィンドウが出てきていて、いろんなものが出てきているというのが、ひと言でマルチ・ウィンドウといえば、そんなんじゃないかなと思うわけです。それでですね、ここに私の考えというのを書いているんで、私はどう考えるかということなんですけども、マルチ・ウィンドウと言うのはああなっています。マルチ・ビューというのはマルチ・ウィンドウというのの特殊形、見え方の点では特殊形なんじゃないかなと思います。それで、マルチ・ビューなんですけども、これ、見え方はこういうふうに見えていると思って下さい。見え方は、いちよう、これはマウスのつもりなんですね。それで、これはブタの絵を書こうと思ったんですけど、書けないんでブタと書いた。それで、いちようこれはうさぎの絵を書いたんだけど、わからないだろうと思って、ウサギと書いた。それで、これは供え餅のできそこないみたいなんですけど、いちようオレンジということにしました。別にアップルでもよかったんですけど。それで、マルチ・ビュー、見え方はこれと同じでなんでマルチ・ビューなんだよということなんですけど、私は、対象が一つありまして、何か対象があって、色々に見る、あるいは色々に解釈するとか、色々に見えちゃうとかっていうのがマルチ・ビューじゃないかなと私は考えてます。

それでですね、結論になっちゃうなあ。それでですね、マルチ・ビューの効用、これ、もともとふられたタイトルなんですけど、一番目に人それぞれ見方がちがうだろうと。あるというか、人にある人の見方を押しつけても仕方ないですし、おまえの考えかたはこうだといってもしょうがないですから、やっぱり見方がちがう。二番目にですね、同じ対象についてですね、さっきの絵にもあったんですけど、同じこれ、対象を示しているんですけど、一つの対象についても、様々な見方をこうゆうふうにですね、比較対象できるというのがマルチ・ビューでいいんじゃないかと。もう一つ、例えば、こういう一つの対象について何か、例えば、ディスカッションとか討論、ディスカッションでも、あるいは、例えばもっとＢＲＥＡＫ　ＤＯＷＮしていて、解析しようとしている時でも同じなんですけど、いろんなふうに人には見えているわけで、それをこんなふうに見えると、おれにはこう見えると、マウスに見えると、ブタに見えると、ウサギに見えると、たまにはオレンジに見える人もいるかもしれませんけども、そう人がてすね、おれにはこう見えるんだというふうに言える。それがいい。だから、それによって、おまえはどうしてオレンジに見えるんだと聞けるわけですから、お互いの意思を疎通するためのタタキ台になるんじゃないかと。マルチ・ビューがですね。そんなふうに見ていくと、多面的な視点の表現がマルチ・ビューによってできて、その表現を認識することによって、考え方の違いとか、解釈の違いといったことが分かるんじゃないだろうか。そうゆうふうにですね、じゃあいろんな見方があるんだと認識するとですね、さらにですね、さっきまでは４つ目しかなかったんですけども、

こういう視点もあるんじゃないかと、５つ目の視点ですとか、６つ目の視点とかですね、そういう新たな視点が思い浮かんでくるんじゃないか。というふうに考えます。それでですね、いちようＰＷＢという題もいただいてますんで、じゃあＰＷＢとはどうなんだろうという話をしようと思います。手元にある資料の中の一番最後のほうにもあると思うんですけども、ＰＷＢでのマルチ・ビューはサンプルですけども、こんなかんじです。マルチ・ウィンドウの形態はもちろんですけど、マルチ・ビューといっているのは、ＰＷＢでいっているようなものを表せる一つ一つのウィンドウですよね。ウィンドウは一つの対象をこういうふうに、こうゆうモデルに解釈した、こうゆうモデルに解釈した、、こうゆうモデルに解釈したと考えたら、ＰＷＢの表現はマルチ・ビューだと思うわけです。

後ろから、結論から言っているんで、なかなか論理の展開が苦しんですけども、それで、ＭＭＩですから、ＰＷＢのＭＭＩという立場から見たとき、マルチ・ビューというのは、？？？でもＭＭＩでもそうなんですけど、見たときに個々のコンポーネントというか、コンポーネントはどうなるのかというこれは位置づけなんですけども、ここにＰＷＢにとってのユーザーでもあり、デベロッパーでもあるんですけど、Ｐｒｏｔｏｔｙｐｅｒという人がいます。この部分にですね、ＵＮＩＸのｓｈｅｌｌですね、Information Browserというｓｈｅｌｌというかインターフェイスがあります。インターフェイスというかマネージャーのかな。それで、この下にですね、下のこの部分のところがＰＷＢで言っている情報の、ＰＷＢの情報の部分で、その一番下にユーザーからの要求ですね、システムに対する要求を、要求を持っているファイルがあると考えます。それらの要求、これは一番最初にお見せしましたマルチ・ビューの時の対象になるんですね。つまり、このRequirement Informationというこの部分がこれになっていて、ここにモデルが６つ定義してあるんですけど、この６つのモデルがマウスでありブタでありウサギでありオレンジであるとか、見方の違うというか、見方の違ったものになっています。それで、これはあまり構造を言っても意味がないと思うんですけど、ここにｅｄｉｔｏｒがありまして、ｅｄｉｔｏｒがあって、それがモデルに対応していて、このｅｉｄｔｏｒを通してモデルを作るこかできる。その作られたモデルは、アニメーションという機能がありまして、アニメーション機能によってこの作られた６つのモデルかそれぞれ関連して並列に実行する。並列に実行するというのは、見えかたでは残念ながら今、並列に実行していないんですけど、順々にこうモデルがスイッチしながらこう実行しているように見えちゃうんですけど、それは今使っているバージョンのＳｍａｌｌｔａｌｋの能力のせいで、次の、もう既にリリースされている２．３のＳｍａｌｌｔａｌｋでは、並列に動くようになるんです。ですから、もう少しそれらしい、動きになると思うんですけども、現在のところは古いバージョンを使っていますので、モデルごとに実行しているように見えます。シーケンスに実行しているように見えます。

それでですね、どんなふうにしてモデルを書いていくか、どんなモデルあるのという話になるんでしょうね。残念なから、これ最新、いまのフェーズを言いますと、去年の１０月の末でＰＷＢの一次研究開発が終わりました。ここにでてくる資料はその時点での、そのバージョンでのビューを？？？して作ったものです。そのあと、実用かのプロジェクトというのをおこしまして、日本語化とか移植ですね、もともとのものはエレクトロニクスの４４０６というある？？？ですけども、エレクトロニクスのＳｍａｌｌｔａｌｋマシンというもので作られたものですけど、それらを様々のＳｍａｌｌｔａｌｋマシン上に移植するということと、それから漢字化ですね、日本の市場を考えるためいは漢字化が欠かせないので漢字化、それから使い勝手の悪いところもたくさんありますので、それを直すとかですね、そうゆうふうなプロジェクトが進行していまして、刻々ですね、イメージが外見が変わってきています。そういう意味で、この部分はちょっさ違うんですけど、とりあえず、あるバージョンのＰＷＢはこうだったという説明なんですけど、去年の終了時点のＰＷＢというのは、ＰＷＢのトップの意味ですね、ですからこのInformation Browserというもの、まあｓｈｅｌｌみたいな機能だと言ったんですけども、そういう機能を持っている。これというのは、これも実はさっき、柳瀬さんに説明していただきましたＸＭＥの中でてきたラムちゃんのアイコンとかというようなかっこいいアイコンを作ればよかったんですけども、あまりそういう才能のある人がいませんでして、安易にアルファベットをこう動かしてしまっている。これはキャラクタじゃなくていちようアイコンなんですけどね。安易にそうなってしまっているだけです。それで、この真ん中にあるＴというハイライトというというか、リバースになっている部分が、Original Requirementという意味で、

この部分ですね、こういうふうな形で表していって、Original Requirement sがここに展開されている。これから、作られたモデルが一つ、二つ、三つ、四つ、五つ、六つですね。このもう一つTとあるのは、Original Requirementにもう一つAdditional Requirement いうことでもう一つTをつけてあるんですけども、一、二、三、四、五、六という六つのモデルを定義して、その、定義してあってかつそのモデルとRequirementとの関係をビジュアルに見せているという図てす。

次が、Functional Structure Modelというモデルで、このモデルの提案者はＤＲＡＳの開発者である新田君がオリジナルなモデルの提案者ですけども、どういうふうなモデルかといいますと、この細長い角のとれた楕円みたいなこういう部分がある動きと操作を表しているんですね。ちょっとコーナーがまるくなったものがオブジェクトで操作される対象を表している。これ自身、四角はこの操作の主体ですね。ですから、実行主体というんですか。それらの間をアクションを中心に結んでいったモデルがこのFunctinal Structure Modelとにうものの定義です。

オジェクトはこういうふうにインクロージャというかビューの上に浮いたようなイメージで影ができて、浮いたようなイメージでHierachyと、階層化されています。階層化については、いまのバージョンではもっといろんな解釈ができるんですけど、この時点では、このような解釈がなされています。その次のモデルが、User Interface Modelというモデルで、先程の柳瀬さんの説明、プレゼンテーションでありましたように、古閑さんのプレゼンテーションでもありましたように、ユーザーに見えるというスクリーンな現れるといったものに、そういった部分のシミュレーション、あるいはモデルをですね、簡単に作ろうという目的で作られたもので、こういうふうなタイトルをだしていて、こちらに計測データが現れてくる。それで、こちら側にあるのは、どっちがどっちだったか今忘れちゃったんですけど、二重丸がＯＫで一重丸が？？？、要するにこれインジケータになっているんですけど、ＯＫというのもインジケータになってます。そうゆうふうな、例えば、この血圧が、血圧に対するアンダーラインを越えたか越えていないか。結局、脈拍です。脈拍に対するアンダーラインを越えたか越えていないかというような感じで、越えたものについて、つまりおかしかったものについてインジケイトする、表示するというかたちになってます。あとちょっと、操作上の問題ですけども、つぎに進むに向かって、ちょっとやめちゃいます。そうゆうふうなスクリーンというか、User Interfaceのシミュレーションが簡単にできるようなモデルです。これを制御するために、こういうふうな、これｆｏｒｍｓよんでるんですけども、この時点ではｆｒａｍｅと呼んでたんですけど、ある一枚のがめんですね。ある一枚の画面を定義した時に、その画面とほかの画面との対応関係を表したのがこのダイアグラムでして、ですから、これとこれはセットです。こいつの制御部分がこっちです。今はまたちょっと変わってきました。これは、なんの変哲もないstate transition diagramです。いちばん簡単なやつです。ｒｅｓｅｔというのはシステム的にｒｅｓｅｔとして設けているんですけども、本来的にはnormalとabnormalしかないです。いっこでも？？、alarmのあとabnormalとして実行している。？？？？？？？？？それで、ｎｅｘｔという、次の段階の計測というかシミュレーションにいったときには、またｒｅｓｅｔの状態に戻ってから、計測していて、それでｎｏｒｍａｌのあとａｂｎｏｒｍａｌを実行する。それを繰り返す。それが終われば、ｒｅｓｅｔする。あと、このモデルはPetrinetにいちよう基づく、ある種の制御をですね、制御の中では？？？さんのモデルで、IOPMstartということで、入口、単なる入口を示しているんですけども、あとこれが、何かの動きというか、プロセス、何かの処理を表している。ここで、これは、並列の向きを示しています。ですから、こういうふうなルートとルートを並列して実行しています。ここのひし形の部分はある条件をチェックしにいって、ＯＫであればこつちに進む。そうでなければこっちというかたち。そんなかたちで、条件分岐、並列分岐、それから、同期するんですけども、同期終了とかが記述できるモデルになってます。それでですね、あとは、そんなふうなモデルがありまして、それを動かそうというんですけども、動かすというよりもイメージなんですけども、基本的に今のところ二つ、いちようモードがありまして、Fast mode、速い、実行が速いモードでは、リバースになっていて、ハイライトになっている部分が移動していることによって実行場所を示している。 Slow modeというか、visual modeというかそういうモードでは、こういうふうなボールというか黒い玉がですね、ずずずずっと動いていくようなそうゆうふうなモデルです。そういうふうなモデルを見せる方がvisualでうけがいいです。それから、こんなふうに動かしてみてどうするのという話も当然、でてくると思うんですけ

ども、それに対してはいちよう、こうゆうふうにqueuring model なんですけども、このquering modelのＣＰＵというようなqueue service modelのＣＰＵとしているんですけども、ここのPerformanceですね、動かした時のJobs in queueでキューの変化ですね、ここにあるキューの変化、そういうかたちが時間と共に変化していくとかですね、それから、ここのJob sizeの変化ですとか、Arrival rate ですか、そういうものの変化というのをこうvisualにこうできます。

というようなわけで、色々とできるんですけども、それでですね、マルチ・ビューにまたちょっと話をもどすんですけども、マルチ・ビューでやろうとしたときに、マルチ・ビューというそうゆうような何か表現形態があるときに、なにができるかと、最初にみんなでディスカッションしているときに、多面的な見方を追求していくときにいいなんてことも思ってたんですが、一人でやる時もいいんことがあるんじゃないかな。それは、自分自身で何か考えている時に、一つのことを考えているかと思うと、裏で別のことを考えていて、別の方の考え方にスイッチしたりですね、そんな脈りゃくのない考え方を私はするんですけど、そういうときにですね、やっぱり、マルチ・ウィンドウで、マルチ・ビュー・サポートのようなシステムがあると、あるビューをおこして、ある考え方を書いておいて、別のビューをおこして、考え方をスイッチするときには別のビューにスイッチしておこしてくる。こんなふうに、同時に考えていることを書いていって、あちこちに考え方がとんでしまうような時のフォローもちゃんとできる。それで、ずいぶん強引なんですけども、そういう時につれづれにいろんな考え方を表すようなことができるようなツールがあったら、とＰＷＢで考えちゃったんですけど、いいなと思います。そうすると、もっといろんな考え方が発展するような気がします。色々書いた、いろんな見方のビューに影響されてでてくると思います。それで、一番いいのは自分の考えることが勝手にスクリーンに現れたり、どっかに記憶されたりするのがいいんですけども、なかなかそういう難しいことは当面、できそうにありません。そういうことができれば、それを見ながら考えを組み立てていったり、遊んでみたり、そのようなことができる。そうようなことができるツールができると、一人何役、一人で何役もできるような統合化ができる。そんなふうに思います。じゃあ、どんなイメージ、例えばイメージというふうなことかを聞きたいと思うんですれど、例えば、Knowledge Navigatorですね、これはＡＰＰＬＥのジョン・スカリーが今年の一月でしたっけ、マック・エキスポでデモビデオを作って説明したものなんですけど、そうゆうものですね。あれは、もともとダイナブックのアプリケーションだと思うんですけども、そういうようなものがあればいいなと思います。今のところ、最高だなと思います。

現実ではどうかといいますと、色々あると思うんですけど、私が知っているのはＱＡという、Ｑ ａｎｄ Ａだったかな、というツールがＩＢＭのＰＣ上にのっていまして、それでなんか考える、書こうとする時にその自分が入力した、残念ながら、音声認識していないんですけども、自分が入力したセンテンスをかえしてきてくれて、ここんとこおかしんじゃないとか、それから、こういうことはどうなのというか、適当に Questionというかきいてくれる。それに対して、答えていく、Answerしていくとそうするとなんとなくまとまっていくというツールがあるんですけども、そういうようなものがもう少し、ＭＭＩのようになるといいんじゃないかなと思っています。そういうようなThinking Toolが欲しいと思います。以上です。まとまりのないところで、すいませんでした。

野村： 色々な話をありがとうございました。なにから聞いていいのか、わからないでいますが。

小林（富士通）： ？？？？、最初に話した方とはちょっと、視点が違いますよね。

塩谷： 一番最初にはじめたところというと。

小林： いや、二人のね。二人の視点はかなり似ていると思うんですが、一番、なんかこう、いろんな話をされちゃって、さっきから、なんといいますか、Thinking Tool とかそういうところに、それを施行ってことを支援するような何かを探索しているっていう感じなんですか。ちょっと、ちゃんと理解して聞き取りたいんで。

塩谷： はい、真ん中で考えたＰＷＢの部分は今やっていることを、単に説明しただけでして、最初と最後の部分は、私がそうやりたいな、そんなのが欲しいなと思っているいわゆる、本音の部分です。それで、一番最初の方に説明した手書きのきたない部分は、これでも、結局同じなんですけども、いろんな、マルチービュー使ったいろんな見方を、見方の表現をですね、それせサポートするようなツールはそのThinking Toolとしてはね絶対必

要な要素だと思うんですね。

それで、まず、マルチービューのところを言っておいて、今、マルチービュー、PWBにはマルチービューの表現を取り入れているんだということを説明させていただいて、最後に、最終的には、私が欲しいのはこういうものだというお話をしてみたんですけど。

小林： 話を、ですから、画面上にこう、物理的にでてくる絵の形状なんかっていうより、もうちょっと、真相の方をですね。問題というのは。そうですね。

塩谷： ただ、その画面上の図も、表現とかですね、あるいは、どんなものをレイアウトしたいのかということも、やっぱり、できなければだめだと思います。

小林： もっと、抽象的なことをうまく表現できるような言語。これを捜している感じなんでしょうかね。あなたの頭の中では。そうすると、勿論、画面上の形状というのはありますよね。

塩谷： ええ、そうですね。

本田（YHP）： YHPの本田ですけど、ちょっと関係ないようなんですけど、PWBはあまりよく知らないんで質問させていただくんですけど、これ、このプロセスというのは、いわゆる Visibilityをあげるということだと思うんですけど、マルチ・ビューということでいろんな見方のできるのを、どれだけVisibilityをあげるかということ、PWBもその目的であるモデルを変えたときに、Visibilityをあげるということだと思うんですけど。そういうときに、どれくらい、その問題に遭遇しちゃうか。例えば、PWBで今、患者モニターのシステムをみせたいただきましたが、質問したいことは、患者モニターでしたっけ、それをやるために、どれくらいのコーディングなのか、オペレーションなのか分からないですけど、それをやらないと、アニメーションとかでてこないんですか。例えば、別の問題ですね。患者モニターじゃない、違う問題でやろうとした時に、PWBでそのようなビジュアライト？？するというような、どれくらいの作業力が必要かというようなことを知りたいなと思いまして。

結論はなぜか？？？？というと、例えば最後のところで、 Knowledge NavigatorとQAがあると思うんですけど、 Knowledge Navigatorというのも例えば、患者モニター・システムについてはKnowledge Navigator とXXXのKnowledge Navigatorとは全然、違ったものだと思うんですね。それは、一般的なツールで患者モニターと同じものを少ない記述量でより可視化できるかというのが一つポイントだと思うんですね。そういうことで、現状のPWBというのは、患者モニターで説明されてきたんですけど、それに依存したコーディングとか、作業量とかはだいだいどれくらいだったのか。それとも、あるいは、全然違う別の問題をPWBでやったとしたら、だいだいどれくらいかければ、？？？？？わかったら教えて下さい。

塩谷： PWBはマルチ・モデルで何か、マルチ・モデルを使ってプロトタイプするためのツールとして考えてます。それで、これを基にいくつかモデルを書いていくんですけども、いちばんモデルを書くこと自体は、書く作業自体は簡単に書けます。 editorがsyntax drivenになってまして、例えば、こんな図ですと、クリックして、どういうふうなオブジェクトを作りたいということを指定するだけですから、オブジェクト間にリレーションを結ぶという操作、オブジェクト間には、そのオブジェクト間でしか許されないリレーションしか現れてきませんから、そういう意味では、書くこと自体は楽ですし、シンタクティカルに間違えたモデルは書けなにようになっているんですね。その点は、楽なんですけど、一番やっぱり大変なのは、モデルをつくるところで、例えば、こういうふうなrequirementからこうモデルを作るっていうときに、そのrequirementそのものが自然言語で書かれてまして、ある常識だとか特殊な用語ですとかね、そういったものの知識が前提とされているわけですよね、普通。

そういうものを解釈して、それを立場から言えば、requirementの主観的な判断ですね。こうじゃないかって、やっちゃあいけないわけですよね。そうすると、もとに要求した人が全然考えてもいなかった方向にいっちゃうわけですから、いけないわけなんですけど、そこのところを解釈して書いていく時に、ここのとこどうしても書けない、聞かなきゃ書けないとかって、でてくると思うんですね。そういうときに、その部分を、今はちょっとできないんですけど、ブラックボックスみたいなのにしておいて、書いていって組めなかったところをあとからまとめて、追加インタビューのみたいなことでadditional requestですか、そういうかたちで調べて、また更にその部分についてモデル化していくという作業をしますので、結局分析の方法というのは、かなり変わってきます。でも、それは、PWBが、PWBのモデルを作るのが大変だとかっていう話じゃなくって、分析そのものが大変なんであって、それは、どういうふうなツールを使うおう

が、紙の上だけで分析していこうが、同じ作業だと思うんですね。ですから、分析作業そのものにかかる時間というのは、少なくとも、増えるということはないと思います。PWBを使ったからといって、増えることはないと思います。

それをモデル化して、しかも、先程の主題のマルチ・ビューのところですけど、いろんなモデルを作りますから、そのいろんなモデルを作って、最後にアイメーションで組み合せて動かそうとします。ということは、別々の見方をしていたものを一つのシステムに組み上げようとするわけなんですね。その時、新たにここととこがおかしいとかですね、つまり、相互矛盾ですね。そういうものが、発見されてきたりしますから、そういうことによる、分析効果っていうんですか、解析効果と言うんですか、それが、大きいと思います。それでは、完全に答えにはなっていないんで、PWBでモデル化するのにどれくらい工数がかかるのかという質問はですね、純粋にモデルを書く分だけですよね、分析というのは本質的、必要なものたと思ってますけど。モデルを書く部分について限って考えれば、これを書くのにたぶん一時間とか、これ一枚ですね。例えば、こういう画面スケッチというか、イメージを固定して書くんだったら、一時間とかじゃなくって、十分とかで書けてると思うんですよね。でも、書く作業というのは、そのOriginalのRequirementsを見ながら、あるいは、その自分の中間的な解析といってもいいと思うんですけど、見ながら書くんで、全体としては結構かかると思うんです。

分析時間と書く時間というのは、書きながら分析しているということになると思うんですね。だから、分けられないと思うんですね。結局、分析して？？？にだして書かせるということはできないと思うんです。こうゆうふうなモデルの場合。そういう意味で言えば、書く手間というのは僕は、分析の中に、分析の時間の中に含まれてしまうんじゃないかと思います。

本田： 少し、質問か悪かったかもしれません。今のは、一つの答えだと思いますが、もう一つの見方をすると、例えば、今ので、一種の制御システムですよのね。制御システム的ならモデリングは楽です。単に書くだけでアナリストするだけなら、ワープロがあればいいんてすよね。書いて分析してするだけんなら、ワープロなり、？？？みたいな機械があれば十分なんですよね。モデル化とか、あるいは、アニメーションとかはぼくは分からないですけど、そうしたときに、例えば、シーケンシャルなコントロールなモデルに対しては、アニメーションはしやすいかもしれないけれど、本当の質問というのは、分からないですけども、患者のデータベースをアクセスして、患者のヒストリーを調べてこれでモデル化したいというきとに、例えば、データベースに？？？を与えて結果をもらうということは、アニメーションできないといけないですね。例えば、そのようなものがてすね、PWBをどれくらい有効的に使えるか、あるいは、そういうものをやろうとした時に、全然直さないでも、例えば、データベースのやつでも、マルチ・プロセスの同時に入ってくるようなやつでも、例えば、パフォーマンス、パフォーマンスのモデルというのがありますよね。そういうときには、結構難しいわけですよね。どのくらいトラフィックがあるのか、待ち行列がどれくらいあるのかっていうことも、やり難いものなんですか。そうすると、例えばこういうやり方に、待ち行列とかトラフィックとか、そういうものに対して、どういうモデリングがあるのか、ないのか、それによっては全然、記述できないものもあるかもしれませんね。ないかもしれませんけど。それをどれくらいまで、例えば、範囲とか、適用範囲です、とかいうことはお考えなのかっていうことを聞きたい。

塩谷： 今のおしゃったデータベースに関してですと、いまのPWBですと機能がないです。データベースにアクセスを指定して、その結果によってどうこうするといういう機能は今はないです。そのですね、完全にするのは考えてないですけど、疑似的なファイルにアクセスして結果をもらってきて、その結果によってそのアニメーションが実行中であることは次にはしろうとおもっているんですね。今はどうしているかというと、データベースみたいなものがあった時には、結果をどうこうするというのは、データベースを実際にアクセスした結果ではなくって、さっきだしたんですけど、User Interface Modelというのをさっきだしましたけども、あのモデルで、操作する人がですね、今のところ、この場合でしたら、NGしてくれるとかOKしてくれるとかいうかたちで、対応してます。今のモデルでは。

それで、アクセスした時の、アクセスにいった時のPerformanceについてはですね、簡単なqueuing modelでキューの大きさとサーバの能力で、表現できるものしかできないと。今のところ。かなりqueuing modelをちゃん

とimplementするのはすごい大変なことで、やれば、できるといいなとみなさんおっしゃるんですけど、ものすごい大変なんで、今のとこ一番簡単に、キューの大きさがどのくらいあったときに、サーバの能力がどけくらいあったときに、どのくらいの数を与えたら、どういうふうな結果になるとか、こういうふうな図ですね、そんなものがでるようになってるんです。ですから、そういう意味で言えば、実際のperformanceに対するシミュレーションとか評価というのは、少なくとも今のバージョンでは非常に力が弱くって、User Interface、つまりスクリーン・アウトのようなものですけど、そういったものを作って、実際に動かして評価していくんです。実際にこういうふうなユーザー画面ができたら、？？画面ができたらいいなと作ってみて、ユーザーさんに見せてみて、少しちょっとイメージが違うところを残してあるんですね。そんな感じでやる部分とか、他のモデル、いくつかモデルを書いているわけですから、いくつかのモデルの関連、いくつかのモデルを関連させて見ることによる分析効果というんですか、システムの分析ではなくて、あくまでも関連させようとする人が分析しているんですね。そのような効果ですね。ですから、あまり定量的な効果というのははっきり言えません。

野村： 質問の方は、とりあえず、塩谷さんのことに関する、とりあえず、塩谷さんの発表を終わっていただいて、席に戻って、咳を止めていただいて、大体、全体に関するような質問をしたいと思いますが、休んでる間、質問をしたいと思いますが。

桑野（NTTソフトウェア研究所）： いいですか。NTTソフトウェア研究所の桑野です。今の塩谷さんの説明の中で、二つお話しされたと思うんですけど、まず、Knowledge Navigatorというようなお話というのは、私も大変必要性を感じていまして、例えば、情報生理学だとおもうんですけども、多方面、いろんな方面からの視点をもってですね、ものごとを見て、別の見方をして、新しいアイデアを作り出していくというのは、非常に大事だと思うんです。それに対して非常に賛成で、私もこういうツールが欲しいなと思うんですけど、もう一つよくわからないのがですね、実はその６つのモデルが出できているんですけども、その６つのモデルが出できたもともとのアイデアというのが全然、よく分からなくって、どうしてかというと、実はですね、？？？ツールとかいろんなツールがあるんですけど、そのツールというのは、実はいろんなモデルとかソートとかメソトロジーがあってですね、その上に成り立っているんですね。

実際に、そういうのを使おうと思いますと、本当に使おうと思いますと、メソトロジーを理解、本当に理解していないと全然使えなくって、単にお絵書きツールなんですね。実は。問題を多方面からディコポーズ？？？してですね、新たな視点を見つけるというのは非常に大事なんですけど、一つのものの図のモデルがでできて、これで問題がいろんな方面から分析できるんだよ、というその？？？ですね、６つのモデルの生まれたところのそのアイデアがなにかを、そこのとこを説明して下さい。

塩谷： そこが一番説明できないとこでして、実は。あえて言うならば、メソトロジーをsupportするためのツールとして作っているんじゃなくて、便利ツールを集めて統合してみたという感じですね。ですから、メソトロジー、それしかないですね。やっぱり。なんで６つかというとこですね。６つになったというのは、たまたま６つ考えてもみて、これだけあれば書けるんじゃないかな、というふうに決めただけです。そういう意味で、６つ全部書かなきゃいけないと言っているんじゃなくって、例えば、User Interface Modelを動かせばそれでいいという状況でしたら、User Interface Modelだけで、だけを作って、それを動かして、それを解釈というか、動かしてみると。他に必要なモデルを書いて、それとまた組み合わしてみるというようなかたちで、全て、いつも６つなきゃいけないというふうに考えているわけではないんで。そんな答え方しかできなませんけども。

野村： ということで、PWBやERAについてはデモもあるようてすし、新田さんとか色々いらっしゃるんで、そのへんに伺うことにして、とりあえずMMIということで、せっかく三人の話題提供者もそろっていることですし、話をMMIの方にに戻していって、強引に終わろうかと思っていますが、今三人のお話の中で一番、セッションのテーマは、MMIによるproductなりprocessの可視化っていうようなテーマがあるんですが、visionにするということによる効果というのは、みなさん疑う余地がないと思うんですが、その効果がどれくらいてあるかとかですね、もっと効果をあげ

るにはどういうのをやればいいか、あるいは自分かソフトウェア開発者として、開発のためのｉｎｔｅｒｆａｃｅがどのように改善されることを望んでいるのかというあたりをですね、もう一度、話題提供者の方からひと言ずつお願いしたいと思うんですか、順番にいって、まず古閑さんの方から。

古閑：　ＭＭＩですね。昨日のパーティでマウスのことについて、えらく話を花を咲かせていらっしゃったんですけど、マウスのボタンは、３つがいいとかですね、１つがいいとかですね、私の場合はマウスのボタンは、３つの方が良くって、その理由は、使い込んでいくとマウスが壊れてしまって、１つのボタンじゃだめで、代わりのボタンにするためいは違う２つ目のボタンにしてしまうとか、そんな簡単な理由なんですど。

今のＭＭＩで、不満といったら、マルチ・ビューというか、マルチ・ウィンドウでいっぱいでてくるのはいいんですけど、そいつがどうも整理が汚いですとか、そのへんですね。それがうまく、この作業が終わったら、次は自動的に、目線を変えただけで、ウィンドウが上にでてくるだとか、そこらへんまでいけたら、夢みたいな話なんですけど、いいなとか思います。

柳瀬：　私はですね、今の環境の不満ということですが、私自身、今のＵＮＩＸの環境に満足しているんですね。もちろん、ＸＭＥの話をしましたけども、あれでウィンドウ・ベースのマウス・クリックですとか、そのようなかたちにｉｎｔｅｒｆａｃｅを改善してくれますれども、あれもどちらかと言うと、もひとつだなと、内心しています。そもそも、話題から反れるかもしれませんけれども、Ｕｓｅｒ　Ｉｎｔｅｒｆａｃｅはですね、使うユーザーによって違いますから、私自身はですね、例えば、ｅｄｉｔｏｒなり、ｃｏｍｐｉｌｅｒなり、上位の設計支援とか、いろんな開発工程がございますけども、それ、それぞれのｐｒｏｃｅｓｓ等、ツールというのがたくさんございますけども、それでは、自動的に好きなものを選んできて、それらを何らかのかたちで統合化したようなかたちの、そうゆうふうな環境と言いますか、ｉｎｔｅｒｆａｃｅと言いますか、そういうふうなものを望んでいます。話が飛びまして、申し訳ございません。

塩谷：　どんな環境といいますと。

野村：　仮に、Ｔｈｉｎｋｉｎｇ　Ｔｏｏｌというのが欲しいとう話があったんですが、それをどのように使うために、あるいは、Ｔｈｉｎｋｉｎｇ　Ｔｏｏｌとして使う機能といいますか、こういうものが具体的にこうあればいいなというようなお話はないんですか。

塩谷：　あまり、その、具体的にはあげられないんですけども、もし、そのＴｈｉｎｋｉｎｇ　Ｔｏｏｌが、使えるＴｈｉｎｋｉｎｇ　Ｔｏｏｌがあったとしたら、そしたら、少なくともさっきちょっとあけましたけども、ＱＡのようなかたちで最低、ある文章を与えたときに、まあ与える方法は一番いいのは、音声認識でしょうけども、ある文章を与えたときに、そこから、問題点であるとか、曖昧な点であるとか、それから、関係がありそうなこととか、そういったものの、諸々の要素を示してくれるようなもの。ちょっと抽象的すぎるかな。例えば、そのシステムとはどのシステムですかって聞くのは、あまりにもバカですけどね。要するにそんなことで、何かシステムとかって言った時に、システムというのが今まで、文明の流れの中でシステムというのが、ある一つのことを指すというのであれば、黙ってて欲しいけど、そうじゃなくて、突然、システムというのがでてきて、じゃあ、それについてはどんなのか。そのときに、どんなシステムって聞いてきたらちょっとうるさいからいやだけど、少なくともある時点ではまとめてこれこれおかしいよと、システムって入れたときに、そこんとかだけ光ってて、おかしいなってシステムが、Ｎｏｖｉｇａｔｏｒが思ってるとか、そんなふうなツールがあればいいなとはおもいます。

野村：　それでは、会場からのご意見等を伺いたいと思いますが、セッションテーマを設定してありますけども、いわゆる、開発者の環境としてどうゆうものが欲しいかということにとらわれずに、ＭＭＩがどうよくなっていけばいいか、あるいは、どういうものが欲しいかという点で構わないと思うんですが、意見ありましたらお願いします。

久保：　コメントというか、思い込みと一つと質問を一つと二つ話しますが、その前に私の立場というのをちょっと話させて下さい。

私は技術の人間ですけど、四月からＳＯＮＹのＮＥＷＳのユーザーになっています。　そのころ、今は富士通はＳＵＮのワークステーションの商社やってますけど、

やってなかったんで、たまたまそのＳＯＮＹのリースを使ってるわけですけど、結構意気込んでユーザーになったんですね。ところが、見込みの？？一つにですね、今日はＭＭＩを今ある道具の類いをどこまで良くできるだろうかという、そういう、自分の範囲内でやろうと気持ちもあったんですけど、今、完全に意欲を失っています。ですから、感覚的にはほとんどおっしゃっていることはわからないんです。そういう立場での思い込みをしゃべりますね。柳瀬さん、古閑さんのお話のあったような領域についてはですね、こんなふうにすれどいいんじゃないかという、野村さんがいったのと関連しますけど、データですね、データの構造というものとアプリケーションとプレゼンテーションというんですか、とにかく人間とやり取りするところ。その三つをきちんと非常に安定した構造、アークテキチャ理論的に非常に安定した作りを見いだすことでしょうね。一つはね。それはあまり易しいテーマだとは思いませんけど、それができればですね、あとそれを、その作りの構造の上で支援していくというのは色々ヒントはあるんじゃないかと思いますね。

昨日話題になったハイパーテキストですか、それはまあデータ構造の一つの答えになっている要素がありますね。アプリケーションについて言えば、昨日話題になったんですけど、オブジェクト指向のプログラミングでどうやって支援していくかという、それがＳｍａｌｌｔａｌｋ流がいいのか、Ｍａｃ流がいいのか知りませんけども、そういったかたちで少しずつよくなっていく。それからプレゼンテーションについて言えば、色々なキットですね。とにかく様々な要求に満たすために、Ｃでごりごりやるのはかなわないとありますけど、いろんなキットが用意されてにれば、うまくそれをアプリケーションの支援ツールでもって組み合せたらどうかという、そんなことが、今、はなしを聞きながら、題はないですけど、私の思いつきですから、その思い込みをですね、若干、声援を送ってくれてるというのは、最近、今日は１０月でしたっけ、スティーブン・ジョーンズがザ・キューブというワークステーションを話してましたけど、それか一つの、まあ考えはそういうんじゃないかもしれないけど、まあまあ結構、いいせんをだしてくれるんじゃないかろうかと。それも、ちゃんとバイト？という雑誌の１１月号と、それからＮＥＷＳ　ＷＥＥＫの、ＮＥＷＳ　ＷＥＥＫかなんかの１０月の末の方か、１１月の初めの方かなんかに、簡単な記事かあって、それを読んだこともあって、一つの僕の今の思い込みになってますね。それは、当たっているかどうか分かりませんもけど、それが言いたかったことの一つ。間違っていたら教えて欲しいというくらいですね。

二番目は質問ですけど、柳瀬さんのお話になったＸＭＥというのは、柳瀬さんの惚れ込みもあって、結構良さそうに思えるんですね、ところが、一方で栗原さんは使っていないとおっしゃるんですね。それはなんでなのかということですね。良ければ、それが基本的なところで、みんながそれをどんどん育てていけばいいんですね。柳瀬さんのような人がたくさんでてくれば、いいものができてくるんではなかろうか。ところが、そうではないような動きもあるらしいというのは、いったいなんだろうか。そこが質問ですね。

野村：　それでは、Knowledge　Navigatorというのは、個人的に私もビデオを見ていて興味かあるんですが、最後の質問の方の掘りこんでいる柳瀬さんと使っていない栗原さんのバトルをちょっとやってみたいと思いますので、まず、使っていない栗原さんの方から発言お願いします。

栗原：　私はこのプログラムそのものをしていません。少なくともああいうプログラムもいらないし、ニュースやメールは、メールのやり方は趣味の違いがありますね。ニュースは別のマシンで読むから必要としていない。実は、ログインする、他のマシンからログインするとかいうのは、よくしょっちゅうやっているんで、プログラム化しているんですが、それはＸＭＥじゃなくて、ユーザーアプリケーションが恐らくできちゃっているから、わざわざそのためにＸＭＥを立ち上げても危機にはならないと思います。ただ、私もＸＭＥのプログラムをいくつか書いて持ってますが、作ってはみたんですけども、使わないというんですね。

ですから、私はあれは使わないという方法で見ているわけではないですけど、私はたまたま、他の方法を選択しているというだけのこと。私が一番最初に私が言いました、その、あれはそもそも、画面のいろんなことをテストしてみたいがためのプロンプトとして酒匂が作ったものと、私は思っていて、それが実際に使える段階のものとしてまず、酒匂が全然考えていなと私は思っています。　　故に、たいしたものとは言いませんけど、たまたま、私の趣味にあったものはないから使っていないし、自分でも作ってみたけど、やはり速度の、実際に使

ってみると速度の面とか色々ありますから、そのへんで選択されていないということですね。

柳瀬： XMEを好んでいる私ですけども、確かに、先程言われました通りだと思います。と申しますのは、私自身、XMEでアプリケーションを組もうとしますと、やはり無理があると思います。やはりプロトタイプ的なものしかできないんじゃないかなと思います。最後になぜ私がXMEを好んでいて、好んでない方もいらっしゃりますけども、それは何故かといいます、それは好みの問題じゃないかと思います。

栗原： あと、もう一つ付け加えたいと思いますけど、酒匂のあれを作ったのは、酒匂の個人的な趣味だと、趣味というか、別に仕事であれをやれとかいった話を僕は知らないし、また、あいつはこそこそいつの間にかあんなものを作ったわけです。こういうとこで、これだけ話題を集めていまうと、引っ込みがつかなくなって、次のを作るということを話してますからわからないその時にはどうなるかわからないですけど、今はｓｈｅｌｌのプログラムしか組めないわけですけれど、あいつは変なやつだから、プログラム言語も自分で考えるやつだから、その下でインタプリタするものは、また別の言語になって、そしたら、また別のCの？？？とよんでも構わないじゃないかという話になるかもしれない。それは、格ご期待ということで投げてしまう。

野村： どっちがはやかった？

林（ＳＲＡ）： わたしです。ＳＲＡの林です。久保さんのご質問に答えたくて出てきました。栗原君とは別論で、要するに趣味だという話があったんですけども、最近あの種のメタツールに近いソフトウェアというのは、一発では絶対できませんので。

会場： 林さん、使ってるの？

林： 私ですか、使ってます。一発ではできません、あれは、僕と酒匂でアイデアを爆発させたんです。あれは、パブリックドメイン？？？にするのを決めたのも私なんです。ですから、ご希望の方があれば、いくらでも差し上げますけども。ただ一つ問題がありまして、発注するための人がいませんので、なかなか発送ができないんですけど、力の届く範囲でお送りします。

なぜそういうことをしているかというと、会社の中でさえ、あの種類、あの手のソフトは賛否両論がありまして、社内で育てるということが非常に難しいんです。それで、日本中で育てていくしかないのかなというのが結論で、ある域に達するまではパブリックドメインていようというのが私の判断です。マネージャーとして。

野村： 新田さんどうぞ。

新田（ＳＲＡ）： ＳＲＡの新田です。私はXMEを使っていない方なんで、なぜ使っていないというと、二つ理由がありまして、一つは私のマシンはＸ１１が使えない問いうことと、もう一つは私は、ｓｈｅｌｌとｖｓｃコンパイラがあれば満足なんで。

それで、XMEということで、付け加えておきますと、さっきＳＲＡの栗原さんが言いましたけども、Ｍａｃｉｎｔｏｓｈのユーザー・インターフェイスに似ているということろをＳＲＡの有座さんが私の命令で作りまして、つまり私は自分のＥＲＡというのをXでもちたくて、そのためにはＭａｃｉｎｔｏｓｈのユーザー・インターフェイスが必要だったんで、彼に作って欲しいと言ったら、作ってしまったということです。それを酒匂さんが見ていて、何か適当なもっと簡単にＣのプログラムを使わないで簡単に使えるよなものを、何かおもしろそうだから作ってみようというのが、最初のきっかけだったんです。それが、一つの方法は？？？というか、ＲＡＭであるツールを統合して、一つのかたまりというか、まとまりのあるものに仕上げようという方法が一つあったんです。そうしているうちに、Ａｐｐｌｅのハイパー・カードができたので、あれをまたＵＮＩＸの上でやってみようという方法がまたもう一つあって、それが今のようなかたちになったんだと思います。

中野（阪大）： 大阪大学の中野です。XME使ってます。使ってみてわかったんですけど、学生に使え使えと言って、ちょうど卒研もありますから、簡単にできるんだろうと思ったんですけども、なかなか学生が苦労してます。なぜやりたかったかといいますと、ボタンだとかそういうことを簡単にやりたかったから。そんなことで学生が苦労してメインの研究をなおざりにして一生懸命にボタンを作ってるよりは、いいだろうと思いまして。でも、色々聞いていると、ツールキットとかそういうのがあるみたいだから、そうゆうのを使ってみて判断しよ

うかなと思って。学生の要望をきくとSUNというのが一つの方法なんです。

新田： ちょっと言い忘れたところがあったんてすけど、大事なことを忘れてました。XMEの、私、いつも評価しているところは何かといいますと、さっき自分の好きなようにMacintoshのユーザー・インターフェイスを作り変えたという意見かありました。でも、一方にはMacintoshのようにユーザー・インターフェイスを統一しておけば、どこにいっても使えるとか、違うツールでも使えるという二つの流れがあると思うんですね。XMEはその中にたつというか、いったいどの部品までを標準化すけばいいのか、どっから先を使う人の自由に任せればいいのかという、その中間の、区切りみたいなと言うか、基準みたいなのを示しているような気がします。

渡辺（電力計算センター）： 電力計算センターの渡辺ですけども、柳瀬さんに質問なんてすが、XME云々というんじゃないですけども、あそこになさったようなメインの画面を作ってて、なんか、何度もプロトタイピング的なことで繰り返し部分を作られたって言いましたけも、作ってて何かむなしくなりませんでしたか。

柳瀬： 確かになりました。

新田： といいますのはね、あそこで出していただきましたOHPに具体的に送り先の人の名前だとか、それから、そういうようなものがでてましたよね。あういうものというのは、MMIの可視化ということであれば、特に使う人がもっと自由にカスタマイズできるような機能を付加することを探ることによって、MMIが、例えば、柳瀬さんの作られたようなアプリケーションに関しては必要なんじゃないかなという気がするんですけど。

そういうことで、あの場合、一見なんか、良さそうにエンド・ユーザーに使わせる、使ってもらう人にMMIは良くなったように見えているけども、実はなんかすごい、いいことか、悪いことかはここで言うのは避けたいと思いますけども、そのプログラムを作った人の個性を押しつけているんじゃないかなという気がするんですけど、ユーザーに。

柳瀬： これXME、私が今日お話ししました内容に関しましては、押しつけているというそうゆう概念はございません。XMEは自由に変更とか、簡単にできますんで、実際に使う人が勝手に好きなようにボタンですとか、テザインを考えていただいて、定義付けも勝手にしてもらうような感じで、社会に広めようとかなと思ってます。ですから、ゼロから作るとなると、しんどいんですね、むなしくなっちゃうんですよね。おおもとさえあって、あとその変更とか何かは簡単にできますから、このような簡単なメールのツールてすとか、そのようなものでしたら、XMEで対応できるんじゃないかなあと思います。もちろん、Cでかいてもいいんですけれども、あるものは使おうということで。

それでは、SRAの内部にこしたのも？？もありましたけど、XMEを知らない立場から、ちょっとお話を聞いてみたいんですけども、XMEと、さっき、ツールキットというようなお話もでていましたけで、もしかすると、古閑さんの求めていたものに近いのかなという感じがちょっとしたもんですから。例えば、古閑さんの求めていたユーザー・インターフェイスの部分を構築するツールとしてXMEが仮に使えるものか、あるいは、XMEがプロトタイプをして、Reqirementを確認するためのものだとするんであれば、本来そういう目的で作られたツールがパブリックドメインで流されたことによって、一人歩きをしはじめて、他の使い方を考える人達にとっては、非常に不満があるというようなかたちになっていっているような気がしているんで、それが、違う目的のために改良されていくのか、それは既にツールキットで実現されているのかというあちりを誰か説明できる方いますでしょうか。

古閑： その前に、それは私も聞きたかった話題なんで、私の方から。私がさっきプレゼンテーションするときちょっと、伝えかたが悪かったんですけど、要は、私がやりたいのは、ここまでは、ビクチャーエディタで中間ソースをはきだすんですけど、それから、目的の、ここでコンパイルですね。さっきのジェネレータですけど。コンパイルやって、目的のソースにやって、アプリケーションの上位というのがあるんですけど、アプリケーションの上位とその絵の部分ですね、ビジュアルな部分を一緒にコンパイル、例えば、CCとかやったら、一つのアプリケーション、ビジュアルなアプリケーションができるのかな。それをさっき、ちょっと、それがやりたいなと言っていたら、休憩時間に、それはXMEDですか、それでやれますよという、ありがたい有益な意見をいた

だいたので、実際、どうやってやるのかは差し置いて、やれるのかなということせを聞きたい。

野村： 林さんのツールキットとか、そのへんの関連で回答できるＳＲＡの方はいらっしゃいませんか。

林： ソフトウェアの作り方のお話になるんですけど、今の、もともとＸＭＥに対して認識が違う気がしているんで話したいんですけど、先程の渡辺さんのむなしいんではないですかという話はですね、普通今までのソフトウェアではメールの宛先の人なんていうのは、外側から自由に変えられるように作ってあるんですね。Ｃのプログラムにパラメータ・ファイルがあればいいとか、何かを与えるとできる。ところが、ＸＭＥのような道具を使うと、他人のを見れば、同じものを自分で作れる。それが変更容易性なんだ。なので、あのアプリケーションを作っている人はむなしいとか、むなしくないとかじゃなくて、自分のためだけに作ればいいものですから、他人のためのソフトウェアを用意するんじゃないんで、僕はむなしくないというのが究極のゴールのはずだと。

そのためにあのパラメータを書くのはきついのではないかというはなしがある。今ご覧いただいた、バラバラという、ほとんど何だかわからないようなマジックの列が続いていて、ならば、あのマジックの列を出力するエディタを作ればいいと思います。それはハイパーカードと同じです。ここにテキストがあって、ここにボタンながあっていうふうにビジュアルに貼っていって、そこで何でするかというスクリプトを何か書けるんですね。そのスクリプトの言語を次のバージョンで考えている。ハイパー・トークにならないためようにするには、どしたらいいかというのが悩みなんですけども、とりあず、そんなとこです。

結局はどこにいっているかと言うと、あのままいくと、ユーザーから見ると、いわゆるハイパーカードを使っているのと全く同じようなことにどんどんなっていくんじゃないかなと思って、それだどむなしいんで、もう一歩先はないかなと最近考えています。　全然答えになっていませんね。ライブラリの件は要するにＸのいわゆるツールキットというのがあまりにも弱いので、内部が作ろうとしてたくさん作られていますからで、そのレベルのものです。ライブラリは、あくまでもＣのライブラリですから、Ｃのライブラリを使うとＣのプログラミングをすることになってしまうんで、いかにプログラミングをしなくて、自分のユーザー・インターフェイスを短時間で、自分好みのユーザー・インターフェイスを作れるかというのが、いちようＸＭＥの目指している。今のところ、色々まずい点もありますけど、多少成功している点もあって、誤解されやすいのは、こんなものを作ってむなしくないですかと言われる部分があって、人のために作ったらむなしいですけど、自分のためだけに作れるはずなんです。僕は弱いですけど？？。

野村： ちょっと時間もないんで気にしていたんですが、どうしても言いたいひとを優先に。

野中： 日本電気の野中です。むなしいというのは、結局、将来のことはどうなるかわからないとして、作成者を責めることのならないように気を付けたいんですが、要するに最終的に真のＣの改善をはかるんだったら、既存のものを生かそうというのはだめで、最終的に例えばもっている用語なり、システム？？なり、そこを良くしようとするんだったら、やっぱりアプリケーションに手をいれていかなきゃだめだから、そこを外側に何か一枚皮をかぶせて、ちょっと良くなるでしょうけど、アプリケーションに手をいれなきゃだめじゃないかなというとこがむなしいというふうに思う。

久保： 是非言いたい。ＸＭＥの応援をします。

野村： 今、ゼネラル・チェアマンから延ばしていいというお話がありましたので。

久保： 締めくくりのつもりだったんですけども、はいどうぞ、ごゆっくりやってください。ＸＭＥの悪口の一つに、性能というのをあげてますでしょ。性能といのは、僕は欠陥じゃないというのをお話したいんですけど、速いほどいいというのは、自明なんですけど、速くする仕方というのは、インタプリタをやめてコンパイラに代わるというのをやっちゃいけないというのをね。それは、今のチップを二倍にあげる、三倍にあげる、四倍にあげるとかですね、あるいはＭＭＩよくするための特別のチップを作ればいいわけですから、半導体でけりのつくような話をソフトウェアさんが手をだして工数浪費しちゃいけないと。もっともっと、価値を高めるような仕事をやっていただきたいという。ですから、コンパイラを作るのは、非常に安定した言語ができてからでいいと。とんどんＭＭＩを進む方に力を注ぐのが正解であると私の

言いたいこと。ですから、XMEの欠点ではない。よく覚えておいていただきたい。

野村： 渡辺さん。

渡辺： 渡辺です。僕がむなしいという単語を使ってしまたのが、大変誤解を招いているようで。僕は勘違いをしているかもしれないんですけども、僕は、他の人のためにわざわざ柳瀬さんがたまたまXMEを使って、ユーザー・インターフェイスを作っていて、？？？というのかな、できたことを何回もしてたから、とゆうふうに話を聞いたんで、むなしいんなではないですかということをきいただけなんです。そこらへん、MMI、少なくとも画面のデザインとか、使い勝手に関しては、まあ、出てみる限りでは、Macintoshとか他の方も、まあそれなりにいいように思うし、そういうときに、本当にだから、さっき林さんがああゆう説明をなさったんで、そういうとこの検定をやっていこうとするんだと言ってしまえば、そうなのかもしれないですけども、あれを不特定多数のひとに使ってもらおうということになると、使う人は積極的にメールを出したい先の人のアドレスを簡単にお書きかえるとかということが、ほとんど、バカでもチョンでもできるようになっていればいいと思うんですけど、必ずしもそういうふうに見えなかった。それは、まだまだ慣れたてのXMEというツールだからなんでしょうけど。その時になったら、本質というか、もっともっとこうであっていいんじゃないということを夢のような機能を説明してから、ここでこうしかできなから、こういう機能にしたというふうな話にもっていったほうが、良かったんではないかなというのが私の言いたかったことです。そうしないと、なかなか可視化ということができないんじゃないかなという気がするんです。ただメールを読むぐらいだったら、出す方がなかなか難しいのかもしれないけども、私はメインフレームの上でJunetを読んでる身分なんで、まあ、なんとかゴリゴリいれればいいやとか思ってるんですけど。

柳瀬： ちょっと、コミュニケーションがうまくいってなかったようで、私がむなしいと言いましたのは、実際XMEでゴソゴソ作っているんですけれども、大変時間があるんですね。夜中一人で、何かゴソゴソ、ボタンをどこに配置しようかとか、フォントをどしようかとか、そういうためにむなしいなと言ったわけでして、結局自分のためにやっているという感じです。

野村： だいだい一区切りついたかなというあたりで、僕はちょっと、昨日も言ったことですが、人間というのはね色々の考え方かあるようで、例えば、さっき新田さんが言ったようなVIとCと二種類があれば、なんでもできちゃうという人は、いわゆる、僕らみたいに、ボタンとかそういうようなのがなければいけないようなユーザー・インターフェイスを求める人たちとは違う人種だと思うんですね。必ずしも、ソフトウェア技術者というのはそうゆう人種ばかりじゃなくて、XMEなんかで、自分なりにものすごいいいインターフェイスを作ったと思っても、他の人から見れば、とんでもないと。まだこっちの方が使いやすいとかようなことがあると思うんです。

今、僕がこの場で議論したかったというのは、MMIというのは二つあるわけですね。いわゆるエンド・ユーザーのインターフェイスとツールのインターフェイスとあると思うんです。今のアプリケーションは、残念ながら、できたアプリケーションが自分、エンド・ユーザーが使おうとした時に、エンド・ユーザーも色々あって、例えば、電子レンジすら使えない人もいれば、パソコン使っている人もいると。そういう人達が、自分の思うようなユーザー・インターフェイスに変えられるというような機能がないんですね。全然。だから、そういう機能をアもったプリケーションを作るための道具が一つ欲しいなという話と、作る側の立場でのインターフェイスが、例えばVIとCだけていい人もいれば、もっと高度なエディタがないと、どうしてもソースが書けないというような人がいるということを、僕は認識していて、実はツールというのか本来、開発者側で作れるものであるとするならば、もっと簡単にアプリケーションを作るような環境か欲しいなと、僕は感じています。まとまりになるかどうかわからないですけども、MMIの中のある側面だけを今、議論したんですけども、本当はもっと人間を研究しなきゃいけない。特にコンピュータのエンジニアは、どちらかというと、コンピュータを知っているけども、コンピュータを使わされる立場になってしまう人間に対しては、あまり研究していないような作り方をしてしまっているんじゃないかなというあたりを、ちょっと問題として覚えていただきたいなと思っています。

勝手に、職業乱用して自分の意見を言ってしまいましたけど、このセッションは色々盛り上がったんですが、終わりにしたいと思います。

最後にプレゼンターの三人に盛大な拍手をお願い致し

ます。

3. チィアパーソンサマリ

野村行憲(株式会社岩手電子計算センター)

はじめに

ポジションステートメントを見ると、MMIに対する参加者の期待は、かなり高かったようである。実力が伴わない座長の私としては、この期待の高さが私を不安にさせるのに十分な材料であった。こういう場合は運を天に任せて、出たとこ勝負で当たるのが常套手段である。会議の行方がどこへ行こうが私の知るところではない。参加者には申し訳無いが、全ての責任は私を座長に選んだ誰かさんの責任である。（と思ってしまえば楽な気持ちで参加できるというものである）

MMIへの思い込み

Man Machine Interface (or User Interface) はComputerのHardware/Softwareの進歩と共に、大幅に改善されてきているといえる。 Computerと人間との間で行なわれる情報伝達は、入力と出力の２つに分類されるが、これらはそれぞれのデバイスの進歩とともに改善されてきている。

ハードウエアの進歩から見れば、スイッチパネルの時代を第一世代とすると、タイプライタが第二世代、キャラクタディスプレイが第三世代、ビットマップディスプレイとマウスの時代が第四世代と言えよう。この進歩に伴ってランプの点滅とスイッチの操作だったインターフェイスの手段が、テキストとキィボードになりグラフィックスとマウスになって、よりビジュアルになってきた。

この、ビットマップ・ディスプレイとマウスによる高度なユーザ・インターフェイスを実現した画面を見ると、表示パネルと操作ボタンが模擬的に実現されていることに気づく。これも逆説的で興味深い事である。

セッションテーマに思う

与えられたセッション・テーマは「ソフトウェア（プロセス／プロダクト）の可視化はどこまで可能か？」であった。自分で考えても、どのようなペーパーが寄せられるか、おおよそ見当がつかないものではあった。確かに最近のMMIでは、物事を視覚的、感覚的に表現することが流行っているようであり、正しい方向のような気がしているが、開発環境という観点で見た場合、はたして現在どのような状況であるかは、自分では予想できなかった。つまり、私もこのセッションは、聞く側にまわるべきメンバーであったかもしれない。

ペーパを読んで

話題提供者の３名のペーパーが送られてきて、セッション・テーマを見たときの不安が現実のものとなった。３編のうち２編はツールのレビューであり、他の１編は具体的な開発に求められるラヴ・コールであった。つまり、議論として噛み合う要素の見いだしにくいものと思われた。救いとしては、３編ともMMIを視覚化することが目的であるツールについて論じている点が共通していたことであった。

発表者への根回し

とにかく、不安な私としては、発表者の顔だけでも見ないことには落ちつかないので、セッションの前日に何とか顔合せだけは済ましておいた。根回しどころか、ろくに話もできないありさまで本番を迎えることになったのであった。

発表

発表と、それに対する質疑応答は、おおむね予定通りに進んだが最後の発表で、ささやかなアクシデントがあった。アクシデントというのは、発表の内容がペーパーのそれとは違って、マルチ・ビューについて論じ始めたことだった。これには私も面食らったものの、事のなりゆきを見守るしか術が無く、話の流れが強引にペーパーのツールに持って行かれた時には、やれやれと胸を撫でおろした。

討論

さて、いよいよ問題の討論なのだが、初めの２つの発表がビジュアルなMMI構築用ツールの話で、最後がビジュアルなMMI環境を持つプロトタイパーズ・ワークベンチの話と、マルチビューについての話だったので、討論の焦点をどこに絞ったら良いか悩んだ。

しかし、XMEの発表に対する質問の中で、XMEを否定する意見があったのを思い出し、この点を議論してみることにした。

その結果、ワークステーションを自在に操るＳＥの中

でさえ、ビジュアルな作業環境を歓迎しない人達がいることが明らかになった。このことは図らずもビジュアル化することこそ、MMIを向上させることだと思い込んでいた私（達）の考え方に対する、一つの警鐘を与えてくれたと思う。

セッションを終えて

セッション・テーマである「ソフトウエア（プロセス／プロダクト）の可視化はどこまで可能か？」から、かなり外れたセッションとなった責任はすべて私にあるが、個人的にはビジュアルなインターフェイスを好まない人達がかなり居ることを発見したのが収穫であった。

もっと突っ込んだ内容を期待していた向きには、申し訳無く思っている。

おまけ

このテーマについて、研究したいと思っている人達に、参考となる文献を紹介します。

表題：ユーザー・インターフェースの設計（Designing User Interface (1987)

著者：Ben Shneiderman （米メリーランド大学コンピュータサイエンス学科准教授）

監訳：東 基衛、井関 治

内容は対話型ソフトウエアにおけるヒューマン・ファクタにはじまり、ユーザー・インターフェースの理論・原則から応用に至るまで幅広く解説してあり、参考になるものと思う。

4. 感想

4.1 MMIの発表に関する感想

林 香（SRA）：
MMIのソフトウェア（プログラム）そのものの話に偏っているのでは？
MMIそのもの（形のないもの、コンセプトｅｔｃ）を研究している人の話を聞きたい。

山口郁生（電力計算センター）： ＵＮＩＸはあまり使ったことがないのでなんともいえないが、ウィンドウを作るだけでも難しそうだと感じた。

塩谷和範（ＳＲＡ）： スクリーンレイアウト関連の話になってしまった。他の観点の話にももっていきたかったのですが、非力なため抽象的になりすぎて話がまとまらなかった。すいません。

野中 哲（ＮＥＣ）：
・ＸＭＥは早く次のステップに進んで欲しい。真の対話型ＭＭＩ／ウィンドウ環境を目指して。
・マルチビューとは何なのか良くわからない

三浦あさ子（ＳＲＡ）：
ｕＩ（ｕＩではなくてＭＭＩなのかな）というのは本当にＷｉｎｄｏｗの話をするだけで語れるのかどうか。ＭＭＩという話題の中には、機械が、人間が何かを作ったり考えたりする場合にどういうふうにサポートできるかという視点があるということが分かった。何かちょっと違うんじゃないかなという気がいつもしています。

北野義明（ＫＣＳ）： ＳＲＡ塩谷氏の発表だけが他の２人と違っていたが、個人的にはあの辺りのテーマを突っ込みたい。ＮＴＴソ研の桑名氏の意見にあった情報のリレーション－－しかもソフトウェア開発メソッドに基づくもの－－を深く追求した上で、情報（プロセスやイベントやオブジェクトについての）をモデリングするという方向の重要性をいいたい。

海尻賢二（信州大）：
観点１．如何に（簡単に）作るか。
２．評価（全くなし）
３．ｄｕｃｕｍｅｎｔとの関係

藤岡 卓（三菱電機）：ちょっと期待はずれでした。

桑名栄二（ＮＴＴ）： ＳｏｆｔｗａｒｅＰｒｏｃｅｓｓ可視化の議論が出来なくて残念

黒坂靖子（ＪＩＰ）： ＰＷＤについては、よくわかりませんでした。（勉強不足）ＸＭＥについては、使用してみようかな・・という気にさせる発表で１番分かりやすかった。（現実性については？でも）もっと発表者が現場レベルでの可能性とか、応用性を実例を取り入れて説明して下されば、分野は異なってもなんらかの方向性（思想）をｓｈａｒｅ出来ると思いますが・・（余りにも現場レベルですいませんが）

広瀬隆夫（ＪＩＰ）： Ｘｍｅにたいへん興味がある。ウィンドウ、ボタン等をＣで書いてもほとんどコーディングレベルで同じになってしまう。またすぐに変更したくなってしまう。それならＸｍｅのようなツールで作成した方が良い。

中田修二（シグマ本部）： 全般的なコメントとしてMMＩのセッションは深みが欠けていたという印象が残り少し残念です。古閑氏の発表については現場のニーズが感じられたが、すでにこの方面ではある程度の研究開発が進められている現状が一方にあるので、もし支援システムの開発をされるのであれば相当急いで実現される方が良いとの印象を持ちました。ＸｍｅのＸ１１バージョンが１１月にリリース可とされているので期待しています。機能が本報告のＸ１０版とどの程度強化されているのかに興味があります。ＰＷＢについても同様に、実用版を開発されているそうですが、その強化内容等に興味があります。また、ツールとしてのＰＷＢにくわえてｐｒｏｔｏｔｙｐｉｎｇの方法論／ガイドも提示されれば一般利用者に分かりやすく思われました。

坂本 治（シグマ本部）： 「ビジュアルなＡＰ作成ツールを考える」要求を具体的に説明して欲しかった。

上妻健一郎（電応研）： ＸＭＥの次のバージョンはＷｉｄｇｅｔをベースにしたコンパイラにしていただきたい。

濱田 勉（ＮＴＴ）： “誰のためのMMＩか”ということをしぼりこんだ方が意見を出し易いように思う。

今別府芳暢（ＮＴＴ）： ビジュアルなＡＰ作成 実現させて再度発表して欲しい
ＸＭＥ 自分のＸにも入れてみたい。開発にも十分使えるようにバージョンアップを

渡辺雄一（電力計算センター）： どれもまだMMＩの改善の本質にせまっていない気がする。特に柳瀬さんのやったような仕事はもっとユーザが好きかってに変更可でもいいはず。一方、古閑さんや塩谷さんの仕事の対象は、何かシビアな要求に対するMMＩの決定の支援であると思うが、古閑さんの話はまだ十分にねれていない。（ガンバッテネ）

高橋哲也（神戸製鋼）： 最先端のMMＩとか、将来・未来のMMＩとかについて話題にしてくれる人が１件ぐらいほしい。

田中正則（ＳＲＡ）： 現在私はビットマップディスプレイでＸｗｉｎｄｏｗを使用して仕事をしていますがマルチウィンドウではターミナルエミュレータが動いているだけです。（これは悲しい） 今回の発表を聞いてつくづく思いました。

4.2 MMＩの討論に関する感想

林 香（ＳＲＡ）： 具体的なソフトウェアそのものの話にかたよりすぎたかな？

高橋哲也（神戸製鋼）： ユーザの立場にたったMMＩというよりは、開発者の立場にたったMMＩの話に終始したように感じる。

今別府芳暢（ＮＴＴ）： すべての人が要求を満たすMMＩは難しいが、少しでも良くなくてはいけない。

山口郁生（電力計算センター）： マウスを手で使うのは面倒である。両手をキーボードに置いているときはマウスは使えない。まさか今更ダイヤモンドカーソルでやるのはいやだ。手を使わないマウスがあれば言いと思う。例えば足を使うマウスとか。つまらない話ですが、詳しいＵＮＩＸの話は出来ないので。

本田克己（ＹＨＰ）：
もう少し詳しい議論が欲しかった。
“ＴｈｉｎｋｉｎｇＴｏｏｌ”が欲しいと言うのは分かったが
“ＴｈｉｎｋｉｎｇＴｏｏｌ”を作れるとしても
“ＴｈｉｎｋｉｎｇＴｏｏｌ”の要求定義ぐらいはしたなあ。

塩谷和範（ＳＲＡ）： Ｘｍｅについての立場（見方）の相違がモロに出てきて興味深かった。やはりプログラムは人手にわたると一人歩きを始めるものだと実感した。

野中 哲（ＮＥＣ）： Ｍａｃのゴミ箱が膨れるのに感動してしまった。

熊谷　章（ＰＦＵ）：　ＭＭＩのセッションは、もう少し分類化したほうがよい。例えば、ＣＧ、ビットデータ、音声、通信などと。極めを細かくすると具体的な討論が出来る。または、もっと抽象化し、認知心理学的なアプローチが面白い気がした。

北野義明（ＫＣＳ）：　ＭＭｉについての各人の要求がバラバラなところで、１つのツールについての意見交換が中心になって（昨年に続いて）残念です。何を可視化するのか？　－－を問題にしたい。ＭＭＩは人それぞれ、またそのときの作業毎に違うだろうし、可変なものが必要ではないかと思う。が、可変と言うことは、人の成長がスムーズな場合、自らＭＭＩをうまく選択すれば良いが、例えば「新人が先々どのようなスタイルで仕事をしていくのかということを難しいＭＭＩのツール等を使いながら習得する」ということを疎外するおそれはないかともおもう。

海尻賢二（信州大）：　ＭＭＩはユーザ個々のもの。簡単なカスタマイズが必要。ＸＭＥはプロトタイピングツールであって、ＰｒｏｇｒａｍＧｅｎｅｒａｔｏｒでない。Ｓｕｎｖｉｅｗのｔｏｏｌｐｌａｎｅｓに替わるものが最適。
ｋｅｙｂｏａｒｄとマウスの共用性は。

藤岡　卓（三菱電機）：　ＸＭＥの話に終始してしまったのが残念です。座長のしめくくりあたりから話が進めば良かった。

平山伸一（ＫＣＳ）
ＸＭＥ：ハイパーカードを使って、ハイパートークでスクリプティングをやって思っていること。
（１）こんなソフトウェアを書きたい。
（２）こんなソフトウェアは使いたくない。
故に、ＭＭＩを向上する為にＸｍｅみたいなのを作りたいが・・つかいたくないなあ。

安間文彦（富士通エフアイビー）：　Ｈｙｐｅｒｍｅｄｉａのリンクの可視化について昨日話題になったように、ソフトウェアの可視化としてはプロセス／プロダクト以外に様々な要求があると思う。可視化したい対象のモデル化とその表現方法を整理して再度議論して欲しい。

佐藤千明（長野協同電算）：　ＰＷＢにからんで、ツールが先にあって、そういうツールをいくつか組み合わせたとの話には疑問です。Ｍｅｔｈｏｄｏｌｏｇｙをバックに、その信者になって、それを支援するツールとして提供しないとツールは普及しない。その信者になれなかった人は自分用のツールを自分で作れば良い。

桑名栄二（ＮＴＴ）：　ＳｏｆｔｗａｒｅＰｒｏｃｅｓｓの可視化の議論が出来なくて残念。
（１）ＳｏｆｔｗａｒｅＰｒｏｃｅｓｓとはなにか。
（２）ＳｏｆｔｗａｒｅＰｒｏｃｅｓｓ、開発環境の可視化を行うための手法。何が必要か。

黒坂靖子（ＪＩＰ）：　質問というより討論の場になるならば、各自がもし同じようなことをしているならば、それと対比させながらお互いの利点欠点を、又、全体での問題点（将来への課題）をあげていくような方法も欲しかった。

広瀬隆夫（ＪＩＰ）：　ＭＭＩというよりプロトタイパーについての話題が多かったように思える。少し視点が違っていると思った。

濱田　勉（ＮＴＴ）：　必ずしも、作成者の意図した通りにツールが使われないことがあっても、それは使う人の勝手で、実はそのことはとても大事なことで、使う人がいて始めて価値が生まれるので、予期しない使い方のニーズがあるということに注目するのは大事だと思う。

岩井田浩章（電力計算センター）：
ＸＭＥ：ＵＮＩＸに関して経験不足なので、なにかユーザサイド的な観点になってしまうのですが。パネルデザインのバリエーションの数を増すより、１つ１つのパネルデザインの充実をし、グラフィック機能の強化で“ぱっと見”を追求して欲しい。