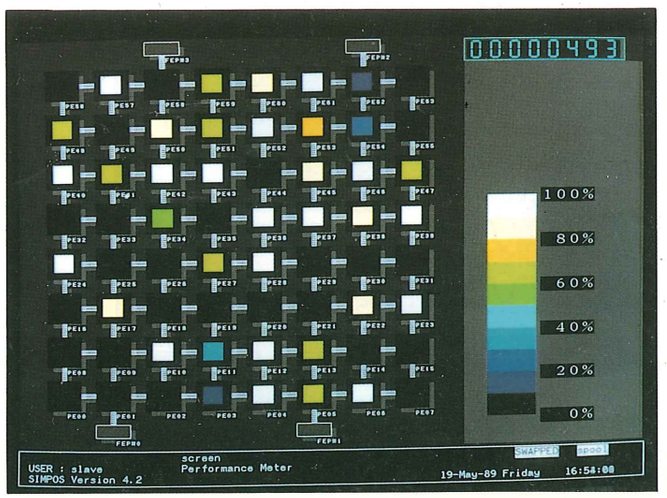
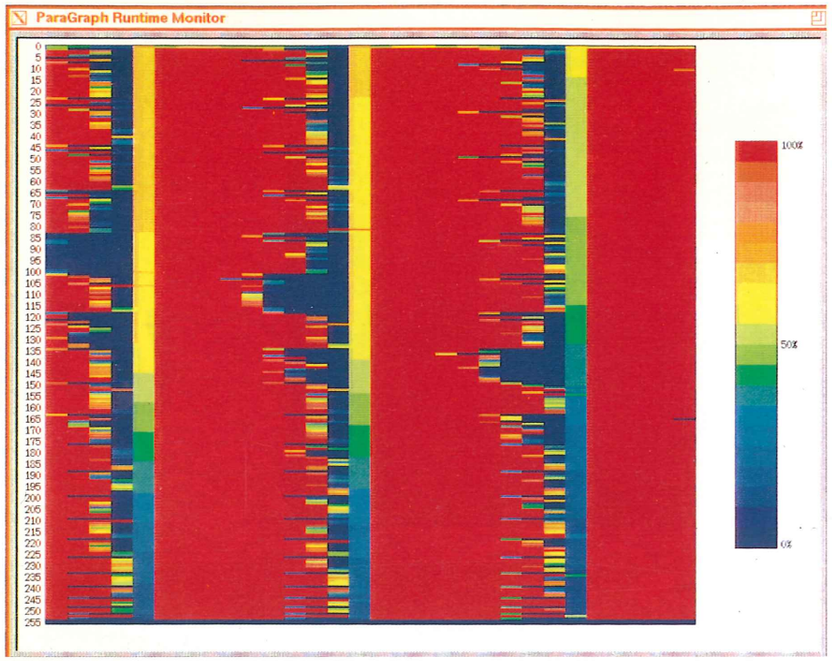
口絵1　並列推論マシン PIM/p
マルチクラスタ構造をもつ最大構成の並列推論マシン．1筐体に32プロセッサ（4クラスタ）を実装し，最大で16筐体，512プロセッサ構成となる．写真は8筐体からなるシステムの全景．第Ⅰ編第5章，第Ⅱ編第3章参照．

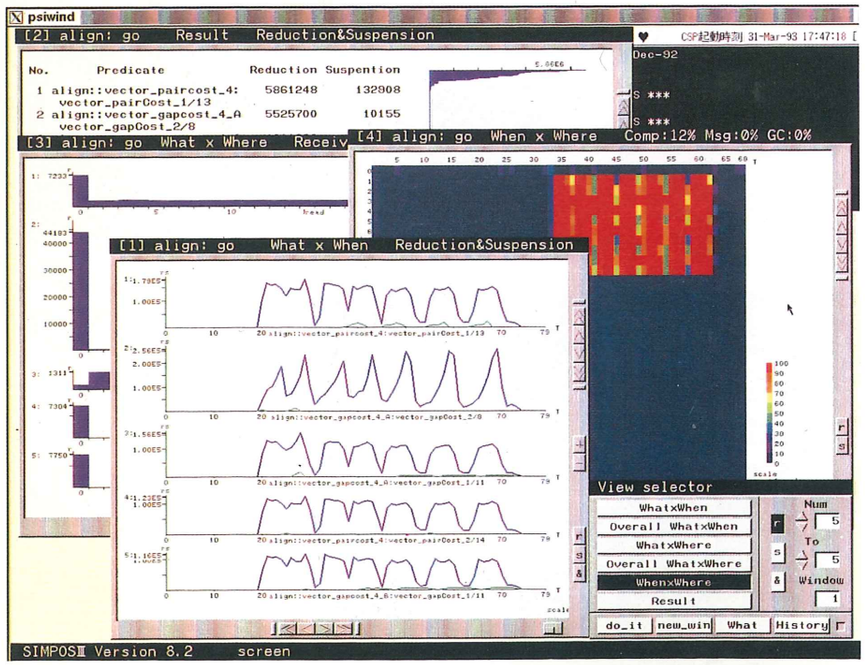
口絵2　並列推論マシン PIM/m
プロセッサを2次元格子ネットワークで接続したシンプルな構成の並列推論マシン．1筐体に32プロセッサを実装し，最大構成では8筐体，256プロセッサとなる．大容量のメモリ（全体で20ギガバイト）も特徴．第Ⅰ編第5章，第Ⅱ編第3章参照．



口絵3　パフォーマンスメータの表示画面
プロセッサの稼働状況を色の変化で実時間表示するモニタプログラム．PIMの実験機であるマルチPSI/V2上にごく初期に作られ多用された．写真は64プロセッサのマルチPSI/V2用の画面．64個ある小さな正方形の色が2秒ごとに更新され，明るい色ほど高い稼働率を示す．第Ⅲ編第2章2節参照．



口絵4　パフォーマンスメータの表示例（PIMOS にてサポート後）
アミノ酸配列の解析プログラムをプロセッサ数 256 台の PIM/m 上で動作させたときのプロセッサ稼働率を X ウィンドウ上にリアルタイム表示した例．縦軸がプロセッサ番号，横軸が時間を表わしており，時間経過とともに表示画面は左へスクロールしていく．桝目の色が赤いほどプロセッサ稼働率が高いことを示す．第II編第5章4節参照．

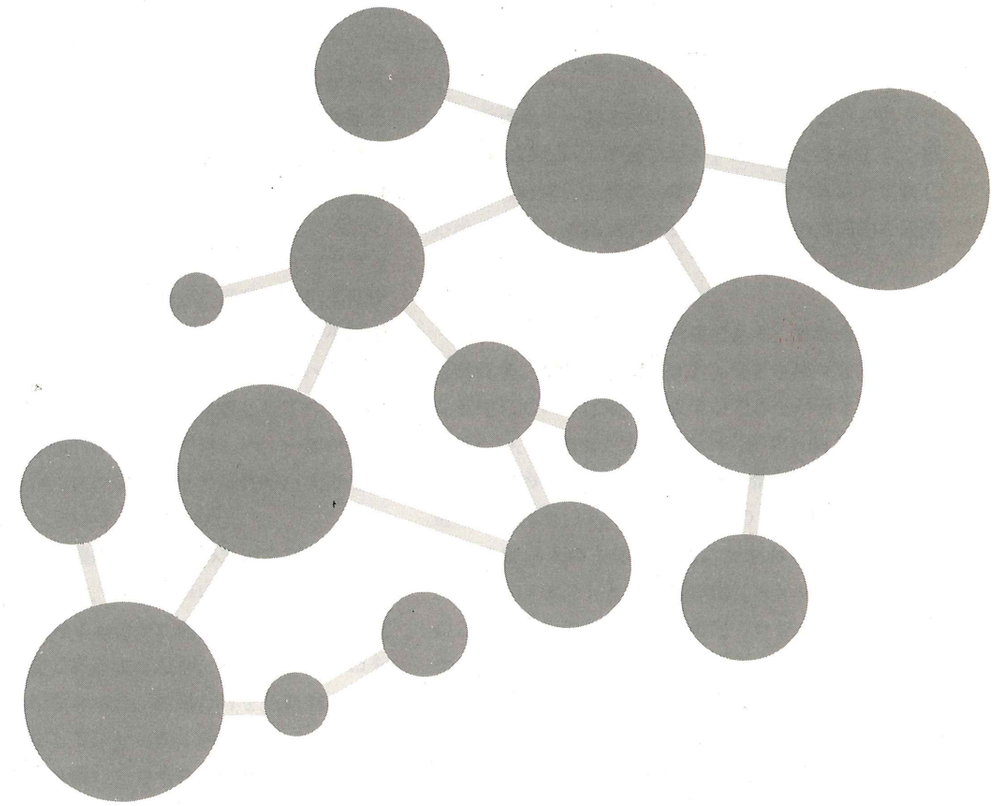


口絵5　ParaGraph の表示例
プログラム実行中に実行に関するログ情報を取り，実行終了後にそれを解析してグラフに表示するツール．PIMOS が提供している．図は口絵4と同じプログラムの解析結果．最前面のウィンドウは，単位時間ごとの述語の呼出し回数とサスペンド回数を示している．ParaGraph では(a)プログラムのゴール実行状況，(b)プロセッサ間通信の発生状況，(c)プロセッサの稼働状況をグラフに表示することができる．第II編第5章4節参照．

第五世代コンピュータの並列処理

――汎用並列処理への道、言語・OS・プログラミング――

瀧　和男　編



プロジェクト雑感 — それぞれの第五

渕 一博
東京大学工学部 電子情報工学科教授
前 ICOT 研究所所長

プロジェクトの成果を総括するこの特集の原稿を眺めて改めてさまざまな感慨が去来するが，最も強く感じるのは，よくここまで成果が有機的にまとまったものだということである．当事者の私が言うと自画自賛になるけれども，この絵は私一人で描いたわけではない．一人では描き上げることは不可能だった．これは何百人の研究者，技術者の合作による超大作なのである．

これはプロジェクトにかかわった研究者たちがそれぞれの思いを込めたものである．ある監督の指揮のもと黙々と作業を続けた結果ではなく，むしろ各人が「それぞれの第五」のイメージに従って伸び伸びと筆をふるってくれた結果の総体である．

結果的にはうまくまとまったが，過程において私が研究者たちに要請したことは，むしろ，全体を小器用にまとめようという意識は捨ててほしいということだった．それぞれのテーマを，研究者としての主体的な意識によって徹底的に追求してほしい．一部が突出してもかまわない．プロジェクトの最終時点まで全力疾走してほしい，ということだった．

結果は自ずから有機的にまとまっていくだろうというのが私の狙いだった．結果について先回りしてくよくよと取り越し苦労するのが，通常のプロジェクト責任者のあるべき姿であるようだが，私にはそのまねをするつもりは毛頭なかった．結果だけを気に病み官僚主義的自己規制をして，せっかくのプロジェクトを損なった前例は洋の東西いくらもある．そんな類のプロジェクトをやるつもりはなかった．もちろん，研究プロジェクトは一種の冒険だから，最後までのすべてを読み切れるものではない．武運拙きときは断固として撤収を図らなければならない．

プロジェクトの発足の際は私なりの熟慮を重ねたつもりだった．まわりには支持だけでなく，不安や不信の渦もなくはなかったが，それに妥協することなく，自信をもって断固として踏み切ることにした．

しかし 10 年は長丁場である．途中で一度は見直すつもりで出発した．実際，プロジェクトの折り返し時点では私なりに再考を繰り返してみた．幸いなことに，基本路線に沿ってさらに前進できると判断することができた．

当初に立てた基本構想を内心で再確認したということだけではない．プロジェクトの前半に育ってきた研究者たちの能力と情熱に賭けることができると判断したのである．それがプロジェクト前半の内面的な成果だった．

研究というのは創造の営みである．外から与えられた計画書，指示書，マニュアルなどを粛々と生真面目に実施していけば期待の成果ができ上がるというものではない．能力もさることながら研究者個人の自発性，うちに秘められたテーマに賭ける情熱と意欲こそが創造的研究にとって本質的要件なのである．

創造が個人の主体的活動にかかわるとすれば，それをプロジェクトとして組織化しスケジュール化することは可能なのだろうか．研究プロジェクトという概念には自己矛盾的要素がありそうである．通常のプロジェクト管理論には創造的研究とは本質的に相反するものが多いと私には思われる．そもそも有効な研究管理論なるものがアルゴリズミックに存在しうるものだろうか．

私たちのプロジェクトは既存技術の改良発展ではなく，コンピュータの新技術の創出を目指したものだった．それは開発研究ではなく，まさに基礎研究であっ

た．(ついでに言えば，基礎研究というのは，論文や学位を目標とするものではない．理論研究だけでもない．それらは副産物にすぎない．技術分野における基礎研究というのは，技術としてはいまだ存在していないものの創造を目指して，試作，実験，評価，理論化などのサイクルを，ときには試行錯誤的に，執拗に繰り返す努力の総体である．)

技術的ストーリーの中身はさておいて，そのような大規模な基礎研究プロジェクトが存立しうるものか，組織論的に不安や不信を感じるシニカルな訳知りたちがいても不思議ではないのである．

命令による強制労働ですめば，こんなに楽なことはない．しかし，それでは馬でさえ水を飲まない．まして研究者にはまさに逆効果である．

実際，研究者を集団化するとかえってマイナスになることが多い．それは下手な並列プログラミングの効果と似ている．1台の効率に劣ることさえある．それは研究者人種のわがままな性格とか規律訓練の不足に起因するというより，創造の組織化ということに内在する二律背反にむしろかかわっているのである．

とはいえ，一個人の力には限界がある．研究者もそれぞれが孤立的に存在しているわけではない．並列プログラミングでは台数以上の効果を本当に出す方法は見つかっていないが，人間の場合はどうか．ときには協調によるスーパーリニア効果が現出することがある．私たちのプロジェクトは，一面において，技術分野でも基礎研究プロジェクトが有効に成り立ちうることの実践的存在証明の試みだったと言える．

プロジェクト発足前の準備的議論の段階から，十数年間のプロジェクト期間中，プロジェクトの中核研究者たちの言動にはなぜか整然とした統合性があった．ときには，「言論統制」があるのかという揶揄もあったらしい．しかし，人一倍理屈好きで意欲的な研究者たちに上からの統制が利くはずがない．遠くからは同じようなことを言っているように見えても，近づいてみれば，それぞれの育ち，教養，思想を反映した多様な個性的な意見の表出が見て取れたはずである．

そのそれぞれの「第五」のイメージは結果的に大きく厚く重なり合っていた．内部に豊富な多様性を含んだ統一は，官僚的統制や一人の人間のカリスマ性によって実現できるものではない．それは，歴史に潜在する未来への必然の流れを，研究者一人ひとりがそれぞれに予感したものの総和によって醸出されたのだろう．

30歳台という研究者人生の最盛期のすべてをこのプロジェクトにささげてくれた研究者も数多い．しかし当人たちにとって，それは滅私奉公，自己犠牲の類ではなかった．いまの世の中それを強制できるはずもない．それぞれが自己の見通しと決断によって選んだ道だった．ここはそのような情熱的な人たちに自己表現の場を提供するものだったのである．

そのような場を用意できたこと自身，ここにも研究者だけでなくそのほかの多くの人たちの情熱的な協力があったのだが，これも歴史的必然という僥倖に恵まれたからなのだろう．

このプロジェクトとその成果は，多くの研究者の夢と情熱＝「それぞれの第五」の産物だった．読者の皆さんには，この特集の記述から，構築された新技術の内容だけでなく，そのこともまた合わせて読み取っていただけるのではないかと思っている．

編者まえがき

第五世代コンピュータプロジェクトは，知識情報処理に適する近未来の新しいコンピュータ技術を目指して，1982年から開始された11年間の大国家プロジェクトであった．

そこでは，論理プログラミングという新しい技術を研究開発の理論的基礎として採用した．それに基づいて，知識処理と並列処理というともに難しい技術を基本から再構築し，それらを結びつけた新しいコンピュータ技術を創造しようと試みたのである．

プロジェクトを始めるにあたっては次のような仮説を立てた．すなわち，「論理」に基づいて上に述べた技術を基本から再構築することによって，従来問題とされた数々の課題が解決され新しい展望が開けるというものであった．

プロジェクトの目的は，知識情報処理のための新技術開発であるとともに，上記の仮説を証明することでもあった．これを進めるため，計算機技術のほとんどすべての部分について，論理プログラミングに基づきゼロから作り直す作業を本当に実行したのである．そのなかには，中核となる論理型言語，それを実行する並列処理ハードウェアと言語処理系，オペレーティングシステムをはじめとするシステムソフトウェア，推論や知識処理の基礎技術，そして応用ソフトウェアとそれらのプログラミング技術に至るまで，具体的なもの作りを伴った数多くの研究開発項目が含まれていた．そうしてでき上がったシステムを実際に使い込んで評価しようという，とてつもなくスケールの大きい計画だったのである．

第五世代コンピュータプロジェクトは，そこですごした数多くの研究者たちにとって，忘れ得ない数々の思い出を残した．技術開発のプロジェクトであってこれほどエキサイティングなものは，もう二度と経験することはできないかもしれない．

プロジェクトの開始当初は，たとえば「1,000台のプロセッサを接続した超高性能な並列推論マシンを最終目標とする」といわれても，それがどんな姿になるのか，そこにどうやって辿り着けばいいのか，皆目見当もつかなかった．プロジェクトの責任者たちは，そこに至る要所と思われる中間地点に，あいまいな中間目標を設定してくれたけれども，そこに至る具体的な道筋は，研究員の一人ひとりが自ら提案し信じるところに従って切り開いたのである．これは渕先生の言葉を借りるならば「それぞれの第五」の営みにほかならない．

第五世代コンピュータプロジェクトを他に比べようがないほどエキサイティングなものにしたのは，「それぞれの第五」を許したプロジェクト管理だったかもしれない．研究開発の詳細目標はすべて自分たちで立てるのであり，その失敗も成功もすべて自分たちの努力と力にかかっていたのだから．そうして築いた大きな成功もそして失敗も，すべて忘れ得ない思い出であった．

「論理プログラミングに基づいて技術を再構築すれば，すばらしい新技術が得られる」という仮説は，一種の天の声であった．信じれば救われ，信じなければ何も生まれない．編者がこれを受け入れたのはプロジェクト前期の終わりであった．信じれば仮説を証明したくなるのであり，それは大きなエネルギーとなった．プロジェクト中期のマルチPSIから，後期のPIM，そして並列応用へ続いた一連の仕事は，このようなエネルギーによっていたかもしれない．そして仮説は証明されたかというと，編者のホームグラウンドとしていた並列処理の領域では，確かに従来技術を越えたという手応えを得た．

従来技術を越えた手応えは，じつはプロジェクト中期の終わりにすでにあった．プロジェクト後期の3年間は，その手応えを確実なものにし，さらに確かめる作業だったかもしれない．そこで確認されたことは，

第五世代コンピュータの並列処理技術は，従来の数値計算しか対象にしなかった並列処理技術に比べて，柔軟性が高く，知識処理に代表されるような動的な性質や不均質なデータを扱う計算問題によく対応でき，高い性能を引き出せるということだった．本書の副題である「汎用並列処理への道」は，そのような技術により，並列処理の適用領域と適用可能性が大幅に広がることの予想と期待を表現したものである．

それでは実現された技術がどのくらいすばらしいか，あるいは従来技術を超越しているかについては，まだ定性的な評価の段階である．しかし各種の本格的応用プログラムの記述性や本書でも示す性能測定結果を見る限り，第五世代コンピュータの並列処理技術がもつ優位性，将来性は，ほぼ疑う余地のないものだと考えられる．

本書の主題は，この第五世代コンピュータの並列処理技術を，なるべくまるごと現在のありのままの姿でお伝えしようというものである．あわせて，編者や執筆担当者らが開発の過程で惚れ込んでしまったそのおもしろさと，将来性への期待についても知っていただこうというものである．

まるごとお伝えしたいという意図は，ここで紹介する技術がまだ若く，確定した評価も十分でないことを考慮して，なるべく多くの方々に技術そのものとそのおもしろさを知ってもらい，もし興味をもっていただけるなら一緒になって使い込み，評価し改良を加えていただきたいということである．そしてまだ若いが大きな将来性を備えたこの技術を，一緒になって一人前に育てたいということである．

このような意図で編集した本書は，以下の３編から構成されている．

第Ｉ編の主題は，まず第五世代コンピュータプロジェクトを歴史的に概観することであり，そして第五世代コンピュータの並列処理技術が生まれ育っていった環境とその成長・発展の流れを描写することである．またそこで活躍した人々の努力の軌跡をたどることにしたい．

第 II 編では，第五世代コンピュータシステムの概要と実現技術を並列処理の側から眺める．核言語 KL1，並列推論マシン PIM，KL1 言語処理系，並列 OS PIMOS について，それらの特徴と技術の要点を解説する．これらは第五世代コンピュータの並列処理技術が，いかなる点で他より優れ，またどのような共通点をもつかを理解するのに役立つであろう．また第五世代コンピュータの良さを取り入れて自らのシステムを設計するような場合の技術資料，または技術のインデックスとしても利用していただけるものと思う．

第 III 編では，第五世代コンピュータの並列プログラミング技術と，本格的な並列応用７種について解説する．ここでの主題は，KL1 プログラミングの実例と技術的なすばらしさ，プログラミングのおもしろさに触れていただくことであり，また並列応用プログラムの開発・実行例を通して，大規模な並列処理の将来性を感じとっていただくことでもある．第 III 編は，論理プログラミングに基づいて技術を再構成した結果，どのような利益が生まれたかを考えていただく本書のなかで最も重要な部分である．第 III 編をご覧になったあと，応用プログラムにとって利益をもたらした技術を確かめるために，第 II 編に戻っていただくのもよいであろう．

これらの執筆にあたっては，各編・節の執筆者の方々をはじめ，ICOT および関連会社の方々，ICOT OB の方々の多大なる協力を頂戴した．心から感謝の意を表するものである．

それでは，第五世代コンピュータの並列処理技術を心行くまでご賞味していただきたい．

1993 年 4 月

瀧　和男

目　次

プロジェクト雑感 — それぞれの第五 i

編者まえがき iii

第Ｉ編　歴史編 **1**

第 1 章　はじめに 3

第 2 章　プロジェクト 10 年間の概観 5

2.1　はじめに 5

2.2　プロジェクト発足までの経緯 5

2.3　FGCS プロジェクトの研究開発成果概要 7

2.4　FGCS プロジェクト推進体制 12

2.5　研究成果の普及と研究交流活動 14

2.6　第五世代マシンの将来展望 15

第 3 章　プロジェクト前期 17

3.1　プロジェクト前期について 17

3.2　パーソナル逐次型推論マシン PSI 18

3.3　並列推論の研究・手探りの時代 25

3.4　並列論理型言語 GHC の誕生 27

第 4 章　プロジェクト中期 29

4.1　本格的並列処理研究の始動 29

4.2　戦略的マルチ PSI プロジェクト 29

4.3　マルチ PSI ハードウェアと PSI-II 30

4.4　苦心と努力の KL1 言語処理系 33

4.5　はじめての本格的並列オペレーティングシステム PIMOS 35

4.6　ハードウェアのターゲット・並列推論マシン PIM を設計する 36

4.7　並列ソフトウェアの研究・事はじめ 37

4.8　死ぬ思いでマルチ PSI/V2 を動かした FGCS'88 39

第 5 章　プロジェクト後期 41

5.1　プロジェクト後期について 41

5.2 並列推論マシン PIM・複数モデルを試作する . . . 41
5.3 並列ソフトウェア研究の拡大 . . . 43
5.4 PIMOS の改良 . . . 46
5.5 華々しい国際交流 . . . 48
5.6 PIM 稼働す . . . 48
5.7 FGCS'92 . . . 50

第 6 章 むすび . . . 51

第 I 編 参考文献 . . . 52

第 II 編 技術編 **53**

第 1 章 第五世代コンピュータの並列処理技術 . . . 55
1.1 はじめに . . . 55
1.2 大目標とアプローチ . . . 56
1.3 どんな並列処理を目指すか . . . 57

第 2 章 KL1 言語 . . . 61
2.1 はじめに . . . 61
2.2 KL1 言語とは . . . 61
2.3 実行機構の概要 . . . 62
2.4 KL1 で扱えるデータ . . . 66
2.5 プロセスとストリーム通信 . . . 69
2.6 プロセス・ネットワーク . . . 75
2.7 プログラム動作の指定 . . . 80
2.8 一階述語論理と KL1 . . . 82
2.9 むすび . . . 85

第 3 章 ハードウェア . . . 86
3.1 はじめに . . . 86
3.2 マシンアーキテクチャの特徴 . . . 86
3.3 五つの PIM モデルの概要 . . . 87
3.4 要素プロセッサ . . . 90
3.5 ネットワーク . . . 90
3.6 キャッシュシステム . . . 91

第 4 章 KL1 言語処理系の実装 . . . 92
4.1 はじめに . . . 92
4.2 言語処理系の概説 . . . 92
4.3 処理方式の概要 . . . 94
4.4 基本言語機能の実装 . . . 99
4.5 拡張言語機能の実装 . . . 103
4.6 効率向上のための機能と実現 . . . 106

4.7 ノード間処理の実現 . . . 112

第 5 章 並列オペレーティングシステム PIMOS . . . 125
5.1 概説 . . . 125
5.2 既存の OS と違うところ . . . 127
5.3 特徴的な実装 . . . 129
5.4 プログラム開発環境 . . . 137

第 6 章 むすび . . . 146

第 II 編 参考文献 . . . 148

第 III 編 プログラミング/応用編 **151**

第 1 章 はじめに . . . 153

第 2 章 並列プログラムを設計する . . . 155
2.1 並列プログラムを設計するとは . . . 155
2.2 KL1 プログラミングはプロセス指向プログラミング . . . 158
2.3 ペントミノと動的負荷分散 . . . 166
2.4 最短経路問題と高並列オブジェクト指向プログラミング . . . 174

第 3 章 応用プログラム . . . 184
3.1 並列応用プログラムについて . . . 184
3.2 LSI 配線 . . . 188
3.3 論理シミュレーション . . . 196
3.4 遺伝子情報処理 . . . 210
3.5 法的推論 . . . 219
3.6 定理証明系 MGTP . . . 228
3.7 自然言語解析 . . . 241
3.8 データベース . . . 252

第 4 章 むすび . . . 262

第 III 編 参考文献 . . . 263

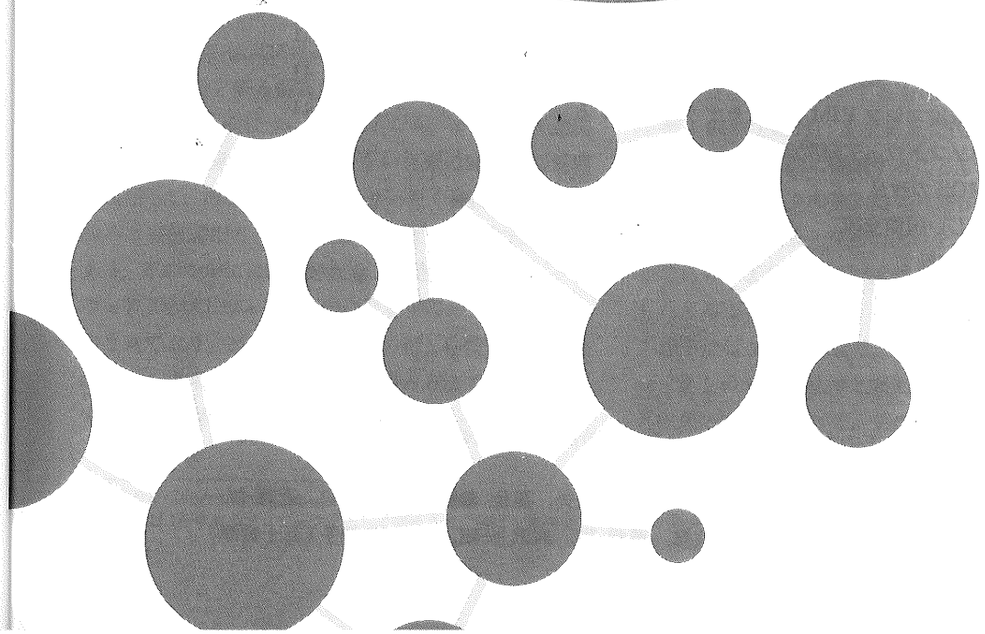
編者あとがき . . . 267

執筆者一覧 . . . 269

第 I 編

歴史編

――10年のプロジェクトと並列処理研究の発展――



第 I 編　目次および執筆者

第 1 章　はじめに	3	**瀧 和男**
第 2 章　プロジェクト 10 年間の概観	5	(以下同じ)
2.1　はじめに	5	
2.2　プロジェクト発足までの経緯	5	
2.3　FGCS プロジェクトの研究開発成果概要	7	
2.4　FGCS プロジェクト推進体制	12	
2.5　研究成果の普及と研究交流活動	14	
2.6　第五世代マシンの将来展望	15	
第 3 章　プロジェクト前期	17	
3.1　プロジェクト前期について	17	
3.2　パーソナル逐次型推論マシン PSI	18	
3.3　並列推論の研究・手探りの時代	25	
3.4　並列論理型言語 GHC の誕生	27	
第 4 章　プロジェクト中期	29	
4.1　本格的並列処理研究の始動	29	
4.2　戦略的マルチ PSI プロジェクト	29	
4.3　マルチ PSI ハードウェアと PSI-II	30	
4.4　苦心と努力の KL1 言語処理系	33	
4.5　はじめての本格的並列オペレーティングシステム PIMOS	35	
4.6　ハードウェアのターゲット・並列推論マシン PIM を設計する	36	
4.7　並列ソフトウェアの研究・事はじめ	37	
4.8　死ぬ思いでマルチ PSI/V2 を動かした FGCS'88	39	
第 5 章　プロジェクト後期	41	
5.1　プロジェクト後期について	41	
5.2　並列推論マシン PIM・複数モデルを試作する	41	
5.3　並列ソフトウェア研究の拡大	43	
5.4　PIMOS の改良	46	
5.5　華々しい国際交流	48	
5.6　PIM 稼働す	48	
5.7　FGCS'92	50	
第 6 章　むすび	51	

この編の執筆に際して，以下の方々にエピソードその他の情報や文章の提供に協力いただいた．

市吉 伸行，稲村 雄，上田 和紀，内田 俊一，川合 英夫，黒住 恭司，後藤 厚宏，近藤 誠一，中島 克人，中島 浩，古市 昌一，宮崎 敏彦，屋代 寛，六沢 一昭，和田 久美子 (五十音順)

第五世代コンピュータプロジェクト (FGCS[†1]プロジェクト) は，知識情報処理に適する近未来の新しいコンピュータ技術を目指して，3 年間の調査・準備期間の後，1982 年から開始された 11 年間[†2]の大国家プロジェクトである．

このプロジェクトは，将来の知識情報処理の理論的バックボーンは「論理」であるとの前提に立ち，第五世代コンピュータシステムの中核となるプログラミング言語として，論理プログラミングを採用した．また，実用的な知識処理を行なうためには，強力な記号処理能力が不可欠との認識から，大規模並列処理によって，これを実現することを目指した．このように，論理プログラミングを核として，知識処理の分野と高度並列処理の分野を結びつけ，新しいコンピュータ技術を創造しようと試みたのである．

第五世代コンピュータプロジェクトは，日本における情報処理関係の国家プロジェクトとしては，いろいろな面で従来の殻を破る斬新で野心的な大プロジェクトとなった．その一面について述べるなら，まだ欧米先進国でも方向性が見定められていない将来の知識情報処理を目指して，あと追いではないまったく独自の技術的枠組みと方法論でもって基礎技術の確立を試みたことである．そのなかには，基礎理論，基礎技術から，ハードウェア，ソフトウェア，応用に至る実際のもの作りまで含まれており，21 世紀に向けての新しい情報処理のために自ら提唱した基礎技術を自前の方法でまるごと試すという前例のない大プロジェクトとなったのである．

もう一つは，プロジェクトを進めるために独自の研究組織を新たに設け，若手の人材をそこに集約して研究開発の拠点としたことである．これが (財) 新世代コンピュータ技術開発機構 (略称 ICOT[†3]:アイコットと読む) である．このことによりメーカの枠にとらわれることなく，自由な発想による独自の研究を十分な予算をかけて進めることができ，またもの作りの段階ではメーカの協力を得て高いレベルの試作を可能とすることができた．同時にこの研究組織は，日本の情報処理技術に関するセンター・オブ・エクセレンス (COE[†4]) として機能し，学会，産業界に対する情報発信基地，情報交流拠点としても計りしれない役割をはたしたと考えられている．

第五世代コンピュータの技術は，大きく分けて知識処理，並列処理，そしてそれらの共通の研究基盤である論理型言語の技術から成り立っているが，本書ではそのなかから並列処理に焦点を絞って，技術と歴史とそのおもしろさを紹介するものである．

第 I 編の主題は，まず第五世代コンピュータプロジェクトを歴史的に概観することであり，そして第五世代コンピュータの並列処理技術が生まれ育っていった環境とその成長・発展の流れを大まかに描写することである．また，そこで活躍した人々の努力の軌跡をもたどることにしたい．

以下ではまず第 2 章で，第五世代コンピュータプロジェクトの発足の経緯と歴史的流れ，研究開発テーマと成果の概要について，プロジェクト全体の流れが大掴みにできることを目指して解説する．また予算規模，

†1　Fifth Generation Computer Systems の略．
†2　当初は 10 年間のプロジェクトとして開始され，後に 1 年延長された．
†3　Institute for New Generation Computer Technology.
†4　Center Of Excellence.

研究開発環境とプロジェクト推進体制，対外交流や成果普及，将来展望などについても簡単に触れる．

続く各章では，第五世代コンピュータの並列処理研究がどのように産声をあげ育っていったかについて，そこで働いた多くの研究者たちのエピソードを交えながら年代を追って解説する．

並列処理だけを取り上げても，実に多くの研究テーマが設けられ並行して進められたことについて，筆者自身いまさらながらに驚くところがあるが，そのためどうしても筆者が直接担当したテーマの周辺については記述が厚く，それ以外では薄めの傾向が出ていることについてはあらかじめご了承願いたい．また筆者と筆者が取材した限られた人たちの経験に基づく記述となっているため，独断と偏見の混じっている可能性についてもご容赦をいただきたいが，少なくとも筆者の愛する並列処理技術がどのようにして形をなし成長していったか，そこには人々のどのような努力があったかが，なるべく伝わるように努めたつもりである．

第 3 章では，並列処理の研究開発が本格化する前の準備段階にあたるプロジェクト前期について述べる．ここでの中心は，自分たちの研究道具となるべき論理型言語用のワークステーション，パーソナル逐次型推論マシン PSI を作る話，並列推論マシンの研究がまだ手探りだった頃の話，そして後に並列処理用核言語の基本仕様として採用されることになる GHC の誕生の話である．

第 4 章では，いよいよ並列処理の本格的研究開発が立ち上がっていくプロジェクト中期について述べる．戦略的マルチ PSI のプロジェクト，GHC を基に核言語 KL1 が生まれるところ，初の本格的並列 OS である PIMOS の開発，並列推論マシン PIM の設計開始，そして第五世代コンピュータ国際会議 FGCS'88 に向けて開発が佳境を迎えるエピソードなどを綴る．

第 5 章では，並列処理の研究開発が本格化し，開発成果も拡大してゆくプロジェクト後期について解説する．大規模な並列推論マシン PIM をいくつも開発する話，本格的な並列応用プログラムの研究開発を立ち上げ開発を拡大してゆく話，PIM に対応するための PIMOS の改良，国際交流の苦労，そして PIM の稼働と最後の第五世代コンピュータ国際会議 FGCS'92 などについて述べる．

第 2 章

プロジェクト10年間の概観

2.1 はじめに

第 2 章では，第五世代コンピュータプロジェクトの全体的な様子が大摑みにできることを目指して，プロジェクト発足の経緯と歴史的流れ，研究開発テーマと成果の概要について手短にまとめる．また予算規模，研究開発環境とプロジェクト推進体制，対外交流や成果普及，将来展望などについても簡単に触れる．第 2 章の内容は，第五世代コンピュータ国際会議 FGCS'92 向けの黒住恭司氏 (元 (財) 新世代コンピュータ技術開発機構 (ICOT) 研究所次長，現 NTT) と，内田俊一氏 (現 ICOT 研究所長，元研究部部長) の著述を基に加筆，編集したものである．いくぶん資料的な記述になっていることをご了承願いたい．なお第五世代コンピュータプロジェクトに関しては数多くの著作や文献が出版されている [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

2.2 プロジェクト発足までの経緯

第五世代コンピュータは，ロジックプログラミング技術に基づく高並列処理と推論処理を基本的枠組みとして，近い将来における知識情報処理を指向したコンピュータシステムである．本節では，そのような方向性を定めることになったプロジェクト発足までの経緯について述べる．

第五世代コンピュータプロジェクトの開始は 1982 年であったが，実はその前に，プロジェクト発足準備のための調査研究委員会が 1979 年に設置され，第五世代コンピュータの枠組みおよびプロジェクト目標の絞り込みを行なうために，たいへんな努力が払われた．

第五世代コンピュータプロジェクトの調査段階における時代背景を要約すると次のとおりである．

- 日本のコンピュータ技術のなかで，ハードウェアを中心とした技術が欧米先進国に追いついた頃
- ソフトウェアの開発・保守コスト増大と要員不足によるソフトウェア危機が叫ばれていた頃
- コンピュータに関する日本の国家プロジェクトの役割が，「欧米の最新技術へのキャッチアップによる競争力向上」から「リスクの大きい先端技術開発による世界のコンピュータ科学への貢献」へと転換すべきことが議論されだした頃

このような状況下で，通商産業省は新しい国家プロジェクトを開始すべく，第五世代コンピュータに関する調査を 1979 年に開始し，1981 年まで継続した．第五世代コンピュータのネーミングは，当時 IBM 社主導で繰り広げられていたメインフレームの世代交替劇がまさに第四世代 (完全に LSI 化されたコンピュータの世代) を迎えようとしていたわけであるが，さらにその先をゆく先端技術開発を指向した通産省の意気込みの表われとも伝えられている．

新しいプロジェクトのための調査検討を行なうため，図 2.2.1に示す調査研究委員会が 1979 年に構成され，第五世代コンピュータの枠組みとプロジェクトとしての目標を提案し，1981 年まで継続した．調査研究委員会には，大学，国公立研究機関，コンピュータメーカから研究者，有識者が招集され，3 年間で延べ百数十人が百回を越える打合せに参加し，そこでは近未来の重要なコンピュータ技術として，以下に示す候補が議

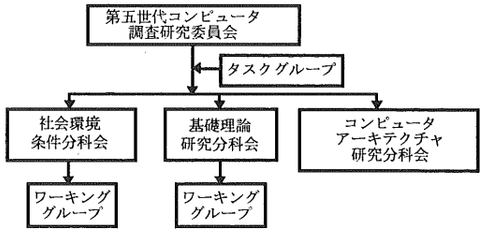


図 2.2.1 第五世代コンピュータ調査研究委員会の構成

論された.

- 知識処理指向の推論コンピュータ技術
- 大規模データベース・知識ベース向きのコンピュータ技術
- 高性能ワークステーション技術
- 機能分散型コンピュータ技術
- 大規模科学計算向きのスーパーコンピュータ技術

これらは，日本のオリジナルな技術開発による国際貢献，将来技術としての重要性，社会的ニーズとの結びつき，国家プロジェクトとしての枠組みと役割などの観点から調査・検討が重ねられ，プロジェクトテーマとして採用するか，見送りあるいは別プロジェクト候補への編入などが議論された.

委員会は 1980 年末に第五世代コンピュータの基本像を定め，さらに技術的観点，社会的インパクト，プロジェクトの枠組みの詳細化に関する検討を継続した. 最終的にまとめられた提案を要約すると，次のようになる.

1) 第五世代コンピュータの枠組み： 並列 (non-Von Neuman 型) 処理と知識ベースを用いた推論処理を基本メカニズムとする. このためのハードウェアとソフトウェアのインタフェースは論理型言語とする (図 2.2.2) .
2) 第五世代コンピュータプロジェクトの目的： 知識情報処理を指向し，現存の方式でのコンピュータの技術的限界に対処しうる革新的コンピュータの技術体系を確立する.
3) プロジェクトの研究開発目標: 第五世代コンピュータのハードウェアおよびソフトウェアの技術を 1 セットとして研究開発する. そのための第五世代コンピュータプロトタイプシステムとして 1000

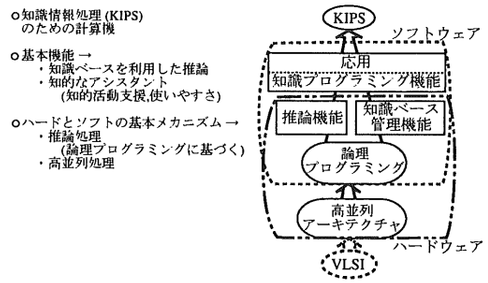


図 2.2.2 第五世代コンピュータの枠組み

台規模の要素プロセッサからなる 100MLIPS(メガリップスと読む. 1MLIPS は 1 秒間に 100 万回推論する性能のことで，100MLIPS はその 100 倍) から 1GLIPS(ギガリップス. 1 秒間に 10 億回の推論) の性能を目指す.

4) プロジェクトの研究開発期間： 前期 (要素技術開発)，中期 (サブシステム開発)，後期 (トータル (プロトタイプ) システム開発) の 3 段階に区分された 10 年間と見積もる.

通産省はこの提案に基づき，第五世代コンピュータプロジェクトを国家プロジェクトとして開始すべく，各方面との折衝を行なった. また 1981 年 10 月に国際会議を開催し，この調査検討結果を発表した. 人工知能関係の著名な海外の研究者や欧米の政府関係者を含め，延べ 300 人が参加し，議論を行なった.

結果はたいへんな反響を呼び，世界のコンピュータ産業界に対する日本の挑戦であるとはやし立てられたり，英国，EC，米国において類似のプロジェクトが次々に発足するきっかけとなった. また AI ブームを巻き起こすきっかけともなった. 一方で，人間のように言葉を理解し思考するコンピュータであるとの誇大報道も手伝って，あまりに野心的で達成が困難な計画であるとの批判も浴びせられた. その理由の一つは，第五世代コンピュータプロジェクトが，大規模プロジェクトによく見られるような明確な目標を備えた開発型のプロジェクトではなく，基礎研究のプロジェクトであって方法論や目標そのものも次第に明確化し舵とりしながら進めてゆくプロジェクトである，ということが正しく理解されなかったためとも考えられる.

この当時の技術的な環境について少し述べるならば，論理型言語はまだ世に出たばかりであり，それ自

身の能力が未解明な段階であった. このため，たとえば OS のような大規模かつ複雑なソフトウェアが記述できないのではないか とか，実行のオーバヘッドが大きく使い物にならないのではないか，といった危惧ももたれていた.

また，高級言語と関連づけられた並列アーキテクチャの研究としては，関数型言語を実行するデータフローマシンの先駆的な研究があったが，知識処理，記号処理用のマシンとしての能力は，まだ明らかではなかった. エキスパートシステムなどの知的なソフトウェアの研究も, 小規模な実験的プログラムを試作している段階であった.

このように，並列推論システムを構築するに必要な要素技術のほとんどが，まだまだ未成熟な段階にあるなかでのプロジェクトのスタートとなった.

2.3 FGCS プロジェクトの研究開発成果概要

2.3.1 FGCS プロジェクトの研究開発ステップ・予算

このような状況のなかで，リスキーで先端的な技術開発を行なうプロジェクトの実施計画を作成するにあたり，10 年の期間を，前期 3 年，中期 4 年，後期 3 年の，三つの期に分け，それぞれの期の研究内容をおおまかに次のように設定した.

- 前期 (1982 年度から 1984 年度)： 第五世代コンピュータとして必要な要素技術の研究開発—ハードウェア，ソフトウェア，基礎理論の多岐にわたる要素技術の研究と，研究開発ツールの開発を行なう.
- 中期 (1985 年度から 1988 年度)： 小中規模の各サブシステムの研究開発— 要素技術の評価と取捨選択を行なう. 見通しの立った要素技術，優れた技術を選び，さらに発展させる. 中期末に，主要技術を統合して，中規模の実験システムを試作する.
- 後期 (1989 年度から 1992 年度)： トータルシステムとしてのプロトタイプシステムの研究開発—中規模実験システムの技術を評価し，優れた技術を選択. それらをさらに発展させて，大規模プロトタイプシステムを試作する. 当初計画に対し 1 年延長し，1992 年度に統合化・評価・改良を行なうこととなった.

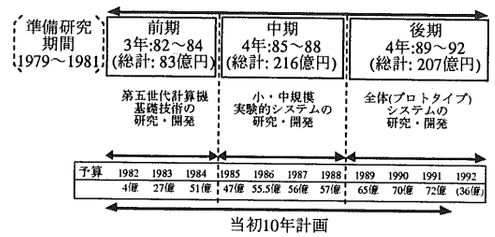


図 2.3.1 FGCS プロジェクトの予算推移

計画を実施するための予算は国の予算であり，各年度ごとに前年度の予算請求に基づき審議され定められてきた. 前期約 80 億円, 中期約 220 億円であり，後期約 240 億円 (総額 540 億円) である.

2.3.2 各ステップにおける研究開発課題

プロジェクトが目標探索型の性格をもち，かつ長期にわたるものであることから，具体的な研究開発計画と技術的目標は，段階を追って詳細化された. すなわち，プロジェクトの開始時点では，前期についてのみ詳細計画を作成し，中期や後期の詳細計画は，一つ前の期の終わりに作成することとした. これは，中期や後期の成果に関する詳細な予測が難しかったことによる. したがって予算や研究開発体制も，研究開発の進捗を評価しつつ決定することとした. 十分な進捗がなければ，中途で打ち切ることもあり得るプロジェクトだったのである.

前・中・後期の各段階における研究開発課題は，以下に述べる状況を考慮しつつ，図 2.3.2に示すように定められた [12, 13, 14]. ここでは説明を省略し，研究項目とその成果についてはあとの各節にて述べる.

第五世代コンピュータ調査研究委員会の提案した 5 グループ 10 課題を基として，前期開始にあたり 3 グループ 9 課題に前期目標が定められた. 前期終了時点までに機械翻訳，および音声・図形画像認識の基礎研究については，民間での開発気運が高まったことから，プロジェクトの対象から外し，メーカの自主研究に委ねることとした.

中期の途中段階において，大規模電子化辞書の開

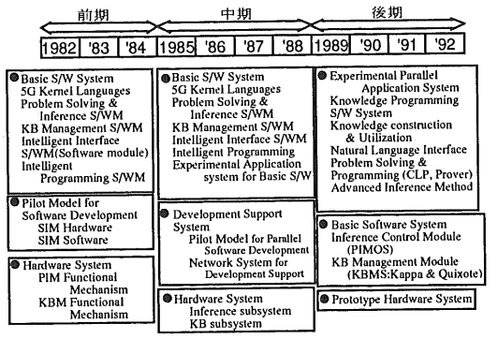


図 2.3.2 前・中・後期における研究開発課題

発については，基盤技術促進センタが出資する日本電子化辞書研究所 (EDR) に研究開発拠点を移した．また逐次論理型言語の ESP(Extended Self-contained Prolog) の汎用 (UNIX) マシンに対する移植についても，同様の形態組織の AI 言語研究所 (AIR) が進めることとなった．それ以外の項目については，プロジェクトのなかで研究開発が継続された．

さらに，本プロジェクト遂行上の特徴として，ソフトウェア開発環境，特にプログラム言語を統一したことがある．これにより開発ツールや研究開発成果の共用や改良に資することが可能となった．ソフトウェア開発環境自体をプロジェクト内で開発したのは，市販製品に適したものがないということもこのプロジェクトの目標の性格からいえることであった．

各段階において，次のとおりの言語とソフトウェア開発環境に統一してきた．

前期: DEC マシン (DEC2060) 上の DEC10-Prolog
中期: PSI / SIMPOS 上の ESP (後述)
後期: マルチ PSI (または PIM) / PIMOS 上の KL1 (後述) (なお，PSI もマルチ PSI シミュレータ＝シュード・マルチ PSI として利用されている)

2.3.3 ハードウェアシステムの研究開発成果概要

ハードウェアシステムの開発は，大きく三つの流れに分かれているが，それぞれのなかでの研究開発項目の推移は，ソフトウェアシステムの研究開発項目とも

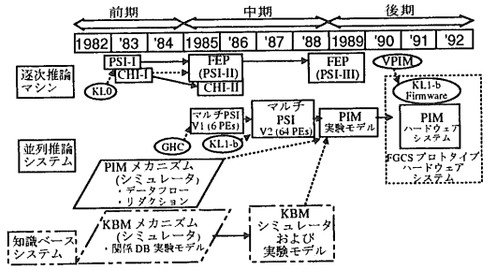


図 2.3.3 ハードウェアシステムの研究開発の推移

密接にかかわっており，プロジェクトの流れを知るうえで重要である (図 2.3.3)．ここでは，ハードウェアシステムの研究開発と成果の概要を少し丁寧に見ていくことにしよう．なお，本書の後の編, 章で扱わない内容に関する詳細は文献 [9, 10, 11] を参照されたい．

A. 逐次型推論マシン

ハードウェアシステムの一つ目の流れは逐次型推論マシンである．

逐次型推論マシンは，Sequential Inference Machine の頭文字を取って，開発プロジェクト名を SIM と呼んでいた．公式名称は,「開発支援システム，ソフトウェア開発用パイロットモデル」であり，自分たちが研究開発に使う道具としての役割と，論理型言語のための専用マシンと, 言語系をはじめて作る小手調べの意味と，論理型言語だけでどのくらいのソフトウェアが書けどの程度の処理性能が得られるかを評価する役割が与えられていた．目標は，そのころ知識処理用に LISP 言語専用のワークステーションである LISP マシンが商用化され始めていたが，そのようなワークステーションの論理型言語版を目指していた．

逐次型推論マシンの研究開発成果は PSI (Personal Sequential Inference Machine) と CHI (Cooperative (back-end type) high performance Inference Machine) である．

PSI (プサイと読む) は KL0 (Kernel Language Version 0) †1マシンとして開発され，前期の PSI-I は約 35KLIPS (LIPS=Logical Inference Per Second:処理速度の単位で，1 秒間の推論実行回数のこと) の性能

†1 KL0: PSI-I の機械語に相当する論理型言語で，Prolog とほとんど同等の機能をもつ．このことからも，PSI は論理型言語のための高級言語マシンであることがわかる．

であった．PSI-I は，中期前半までの ESP†1 プログラムの主要な開発環境用ワークステーション (WS) として約 100 台を製造し使用した．PSI のオペレーティングシステムである SIMPOS (シンポスと読む) はすべて ESP で記述されており，これは論理型言語の能力を試す一つの実験でもあった．

一方, CHI-I (カイ・ワン) は論理型言語の処理速度を追求し，D.H.D.Warren 教授の提唱した WAM 命令セットと高速デバイスの採用により，Prolog の append プログラムで約 200KLIPS の性能を達成した．

これらの開発・利用経験から，論理型言語を中核として大規模なシステムを構成してゆくことへの見通しを得ることができた．

中期において, PSI はマルチ PSI 用の FEP(Front End Processor) および PSI-II として再設計され，約 330～400KLIPS の性能を達成し，CHI は同様に CHI-II として再設計された．PSI-II は，中期中盤以降の ESP プログラム用の WS として約 300 台を製造し，ICOT と関連メーカで使用した．さらに PSI-I と PSI-II は，並列推論マシン実験機であるマルチ PSI の要素プロセッサとして，それぞれマルチ PSI/V1 とマルチ PSI/V2 に使用された．また，並列処理用のプログラム開発環境としても，マルチ PSI シミュレータ (シュード・マルチ PSI) の実行用に利用された．PSI-I, PSI-II ともに，通商産業省からの利用許諾を受けて三菱電機が商品化している．

PSI-III は後期における PSI-II の後継機である．並列推論マシン PIM/m の要素プロセッサとしての利用を目指したとともに，UNIX マシンと結合して三菱電機が商品化した (通商産業省からの利用許諾による)．

B. 並列推論マシン

ハードウェアシステムの二つ目は並列推論システムである．並列推論マシン PIM (Parallel Inference Machine, ピムと読む) の研究開発は次のように進められた．

前期は，本格的な研究開発に入る前の要素技術の研究・実験を行なった．8～16 台規模の実験用ハードウェアシミュレータやソフトウェアシミュレータをデータフロー方式やリダクション方式に基づいて試作した．また負荷分散についてもいくつかの方式を試した．

中期においてはまず，6 台の PSI-I を接続し，並列論理型言語である GHC をはじめて並列実行するマルチ PSI/V1 を開発した．この性能は数 KLIPS 程度であるが，マルチ PSI/V1 の開発により GHC の実装や小規模の並列プログラムに関する開発経験を得ることができた．マルチ PSI の役割は，並列推論マシン PIM のためのソフトウェア開発環境を提供するマシンであるとともに，KL1 言語の処理方式，実装方式を開発評価するための実験マシンであった．

ついで 64 台の PSI-II CPU をメッシュ型ネットワークで接続したマルチ PSI/V2 を開発した．並行して本格的な並列論理型言語である KL1 を GHC を基に設計し，マイクロプログラムによりマルチ PSI/V2 に実装した．実効速度は CPU 当たり約 130KLIPS，64 PE (Processing Element=要素プロセッサ) システムでは約 5MLIPS を達成した．これは種々の KL1 プログラムを開発する上でほぼ十分な性能であり，マルチ PSI を並列推論マシン PIM の実験機と呼ぶにふさわしい結果であった．このうえに，本格的な並列オペレーティングシステムである PIMOS(パイモスと読む) を KL1 で記述して実装するとともに，数種の評価用プログラムを開発した．

マルチ PSI/V2 の開発経験に基づき，PIM の設計を中期から開始し，後期に入って大規模モデル 3 種類 (PIM/p 512PE, PIM/m 256PE, PIM/c 256PE) と，小規模実験モデル 2 種類 (PIM/k 16PE, PIM/i 16PE) からなるプロトタイプハードウェアシステムを試作した．これらの各モデルは，ハードウェアの要素技術に対する実験を行なうため，プロセッサの構成や接続の方式は異なっているが，すべて KL1 を実装するよう設計され，PIMOS を含む KL1 プログラムが走行可能である．PIM の各モデル用の KL1 処理系を開発するにあたっては，KL1 言語実行系の共通仕様を記述した VPIM と呼ぶ言語システムを汎用機上に開発し，それを各モデルへ移植する手順をとった．なお，PIM/m に関してはマルチ PSI/V2 とのオブジェクトコード互換性を保って実装した．また PIM 上で動かす並列応用プログラムを研究開発するために，マルチ PSI/V2 を，64PE から 16PE までの大小合わせて約 20 システム製造し，ICOT と関連メーカに設置して並列プログラム開発に使用した．

†1 ESP: PSI のシステム記述言語で，Prolog を拡張した論理型言語の機能とオブジェクト指向の言語機能を併せもっている．

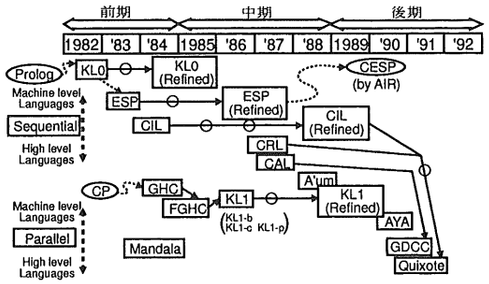


図 2.3.4 第五世代言語の研究開発の推移

C. 知識ベースマシン

ハードウェアシステムの三つ目である知識ベースマシン (KBM) の研究開発は中期まで行なわれた．前期においては，4 台の関係代数演算エンジンをもつ関係データベース実験機 (Delta) を試作した．中期においては，比較・サーチなどのためのアクセラレータを PSI に付加した演繹データベースシミュレータ CHI-II をベースとした多重–多重名前空間をもつデータベース実験システム，および単一化エンジンとマルチポートページメモリからなる知識ベースハードウェアシミュレータを試作した．後期においては，ファイルシステムをもつ PIM とその上の知識ベース管理ソフトウェア (KBMS) の開発に集約し，これらの研究成果を反映している．

2.3.4 ソフトウェアシステムの研究開発成果概要

ソフトウェアシステムの研究開発は図 2.3.2の基礎ソフトウェア (Basic S/W System)，知識プログラムソフトウェア (Knowledge Programming S/W System) などの研究開発課題に基づいて行なわれた．以下に研究課題ごとの成果概要を示す．なお本書の後の編, 章で扱わない内容に関する詳細は文献 [9, 10, 11] を参照されたい．

A. 第五世代コンピュータ言語の研究開発成果

逐次論理型言語としては，核言語第 0 版 (KL0) と ESP が前期にまず設計された．KL0 は逐次型推論マシン PSI の機械語として Prolog をベースに設計され，PSI-I においてはマイクロプログラムによって直接実行された．一方 ESP は，KL0 に OS や応用プログラムなどの大規模ソフトウェア開発のためのモジュラープログラミング機能 (オブジェクト指向言語機能として実現) やマクロ記述機能が付加されたものである．ESP プログラムは KL0 に変換されたのち実行される．

並列論理型言語としては，Parlog や CP (Concurrent Prolog) をベースとして GHC (Guarded Horn Clauses) が前期に提案され，FGHC (Flat GHC) を経て核言語第 1 版 (KL1) の基本仕様となった．KL1 は OS 記述機能など多くの機能を FGHC に追加したものであるが，マシンレベルの言語 KL1-b(base)，中核言語の KL1-c(core)，並列プロセスの制御を記述する KL1-p(pragma) から構成される．マルチ PSI や PIM はこの KL1-b に基づいて設計された．並列 OS (PIMOS) を含む種々の並列プログラムは KL1-c と KL1-p で記述されており，コンパイラで KL1-b に変換されたのち実行される．

また，KL1 から大きな影響を受けた並列オブジェクト指向言語 A'um(ア・ウン) が中期に設計され, 後期になって試作された．さらに KL1 の上位言語として，モジュラープログラミングの機能などを入れた AYA が研究されている．

高水準言語としては，個々の適用分野の記述のために数種類の逐次型言語がまず設計された．CIL (Complex Indeterminate Language) は, 自然言語処理における意味や状況記述をするために Prolog を拡張したものである．CRL (Complex Record Language) は, 非正規型関係データベース (DB) ソフトウェア上の演繹 DB のための内部表現に用いられる知識表現言語の一種として開発された．CAL (Constrainte Avec Logique) は, 問題を制約として宣言的に記述する逐次制約論理プログラミング言語である．

並列論理型の高水準言語としては，まず Mandala が知識表現言語として前期に提案されたが，並列処理環境や経験などの不足により, 中期においては知識表現言語としては前述の逐次型言語をまず開発することとした．後期において，CIL と CRL の開発経験に基づき，並列型知識表現および知識ベース言語として *Quixote* が設計された．*Quixote* は演繹オブジェクト指向言語であり，知識ベース管理システム (KBMS) におけるユーザインタフェースとしての重要な役割ももっており，現在も研究が続けられている．GDCC (Guarded Definite Clause with Constraints) は CAL の開発経験に基づく並列型制約論理プログラミング言語として試作された．

B. 基本ソフトウェアの研究開発

前期において，逐次型推論マシン PSI のためのプログラミング/ オペレーティングシステム (SIMPOS) を ESP を用いて開発し，知識処理プログラムなどの開発環境として利用を始めた．その後中期において，利用結果などを反映し機能拡張と改良を継続した．

並列 OS としては，本プロジェクトの目的に適した先行する技術はなく，中期後半になって SIMPOS の経験なども反映して，マルチ PSI/V2 上にはじめての並列推論マシン用 OS (PIMOS) を開発した．PIMOS は，後期に入って KL1 プログラムの開発環境としての利用経験を反映させながら改良を続けた．このほか，汎用機上の KL1 言語系として，PIMOS 開発支援システム (PDSS) が PIMOS の開発に先だって中期に開発された．また並列ソフトウェアの開発支援プログラムとしては，プロセッサの稼働状況などを計測表示する ParaGraph が後期に開発されている．

データベース/知識ベース (DB/KB) 管理ソフトウェアとしては，前期にまず関係データベース管理ソフトウェア (Kaiser) を試作した．ついで中期において，自然言語処理，定理証明や種々のエキスパートシステムに適用される大規模 DB/KB の構築に必要な機能をもつ Kappa-I および Kappa-II を開発した．Kappa-I および Kappa-II は非正規関係モデルに基づく演繹オブジェクト指向 DB 管理システムのデータベースエンジンを目指したものである．後期において，PIM 上の分散ディスクに格納され分散 DB の管理機能をもつ Kappa の並列版 (Kappa-P) を開発しており，この Kappa-P と *Quixote* が KBMS を構成している．

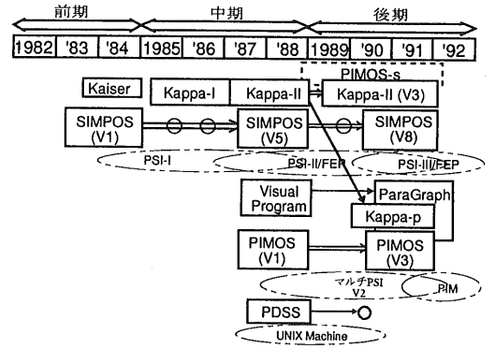


図 2.3.5 基本ソフトウェアの研究開発の推移

C. 問題解決プログラミング技術の研究開発

自動定理証明と自動プログラム合成についての処理の類似性の観点から，前期以来，証明技術研究を継続してきており，まず前期において証明支援システム (CAP) 実験版を試作し，中期において改良・拡張を行なうとともに, 等号に対する数式変換を行なう項書換えシステム (TRS および Metis) を開発した．プログラム検証合成実験システム (Argus) についても前・中期にかけて試作・改良などが行なわれた．

これらの研究成果は後期において定理証明の研究に集約され，推論エンジンとして法的推論システムに適用されている．このほか，メタプログラミングや部分計算技法，学習メカニズムなどの高次推論・学習に対する基礎研究が行なわれた．

D. 自然言語処理技術の研究開発

自然言語処理のためのソフトウェアツールとして，前期において BUP(Bottom Up Parser) や小規模電子化辞書の実験版を試作し，中期において汎用日本語処理ツール (LTB: Language Tool Box) としてまとめ，さらにツールの追加や改良などを行なった．LTB には形態素解析の LAX (Lexical Analyzer)，構文解析の SAX (Syntactic Analyzer)，文生成 (Text Generator)，言語データベースなどが含まれており，ESP 記述で開発されたが，後期において汎用 UNIX ワークステーションでも走行可能なように CESP (Common ESP:UNIX マシンで動く ESP 処理系) 版が開発された．

談話理解実験システム DUALS は，これらのツールの機能検証と自然言語理解の研究のための実験システムとして，前期に第 1 版を試作し，中期においては第 3 版まで改良した．後期においては，これらの研究結果を踏まえて並列自然言語処理実験システムとして立論システム (Dulcinia) の試作を行なった．

E. 知識利用技術と並列応用実験システムの研究開発

知識利用実験ツールとしては，中期において，仮説推論技術に基づくツール (APRICOT) や定性推論技術に基づくツール (Qupras) を試作した．後期においては, 仮説ベース推論や事例ベース推論メカニズム

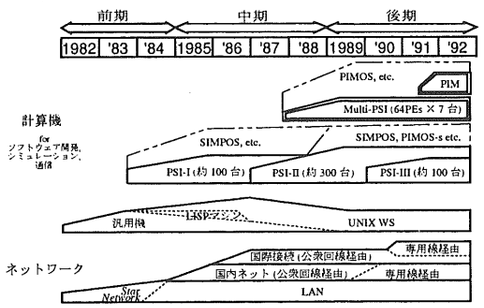


図 **2.3.6** 研究開発環境の推移

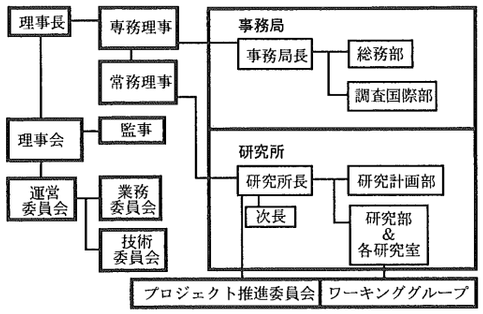


図 **2.4.1** ICOT の組織

に基づく応用実験システムの試作を通じて研究を行なっている．応用実験システムとしては，前期における Prolog 記述による論理回路設計支援と配線支援の CAD 実験システムの試作に始まり，中期においては応用分野を拡大し，配置と論理設計 CAD，故障診断，プラント制御，囲碁対局システムなどの ESP 記述による試作を行なった．

中期後半においてマルチ PSI，PIMOS の開発により並列応用ソフトウェアの研究開発が可能となり，まず KL1 記述による並列プログラムの開発実験と並列システムの評価のために，小・中規模のプログラムを試作し，後期にも並列アルゴリズムなどの改良が行なわれた．並列構文解析プログラム (PAX)，ペントミノプログラム，最短経路プログラム，詰め碁プログラムなどである．

後期においては，以上述べた中期までの開発成果を踏まえ，応用分野をさらに増し，KL1 記述による並列応用実験システムとして，LSI-CAD システム (論理シミュレーション，配線，セル配置，論理回路設計)，遺伝子情報処理，事例ベース推論に基づく法的推論システム，および故障診断・プラント制御・囲碁対局のエキスパートシステムを開発している．

2.3.5 研究開発環境

2.3.2節で述べたように研究開発環境のベースとなる言語は，前期は Prolog，中期は ESP，後期は KL1 として統一されてきた．

そのため，前期においては汎用計算機 DEC2060 上の DEC10-Prolog を利用した．中期においては，逐次論理型言語 ESP のためのプログラム開発実行環境として SIMPOS と PSI (I と II) を利用した．後期においては，並列論理型言語 KL1 のためのプログラム開発実行環境として PIMOS と並列推論マシン (マルチ PSI および PIM)，さらにその端末兼プログラム開発用シミュレータとして PSI (II および III) を利用している．このほか，PIM 設計のためのシミュレーションやネットワーク通信のために汎用機も利用している．

コンピュータネットワークシステムとしては，構内接続には LAN を，国内・海外接続へもゲートウェイを通して接続を行なっている．

2.4 FGCS プロジェクト推進体制

本プロジェクトを遂行するための中核非営利組織の財団として 1982 年 4 月に (財) 新世代コンピュータ技術開発機構 (ICOT) が設立され，同年 6 月から研究開発を開始した．

ICOT の設立は，創造的研究開発を一体として進めるために，次に示すような中核組織の必要性と効率性を考慮して行なわれたものである．

- 研究開発を行なうにあたり，10 年という長期間にわたる研究を第五世代コンピュータという統一的枠組みのなかで行なうために，強力なリーダーシップに基づくテーマの選択や指導が必要だったこと．
- 本研究分野での研究者を，集中研究所のなかで迅速に育成することが必要だったこと．
- 外部の組織や研究者と交流するための中核が必要だったこと．

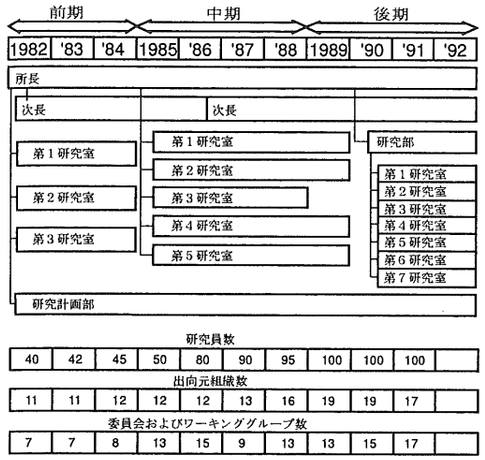


図 **2.4.2** ICOT 研究所組織の推移

ICOT は事務局と研究所からなり，ICOT 研究所内の組織は研究進展に合わせて変更されてきた (図 2.4.1 参照)．ICOT 研究所の全研究者は国立研究所，公的機関，企業からの出向者であり，研究者の育成を行ないつつ創造的研究を行なう必要性から，35 歳以下の若手中心に集められ，3〜4 年ごとにローテーションが行なわれた．これにより組織の活力の維持や効果的な技術の普及が行なえた．研究者のローテーションのために，研究活動の一貫性や研究能力の維持向上の面で多大な努力を要したが，一方若手研究者集団 (平均 30 歳程度) の維持や組織変更時の融通性をもたらした．

1992 年までに，延べ 184 人の研究者が出向しており，平均 3 年 8 か月の在職機関 (約半数の現職を含む) となっている．研究者数は年々増加しており，当初の 40 人から中期末の約 100 人になって 1992 年現在に至っている．出向元組織数は当初の 11 から 19 までに増加してきた．これは中期において，約 25 社からなる一般賛助会社からの出向を受け入れたことによるものである．

ICOT の研究所組織は，前期には研究計画部と 3 研究室であったが，中期には 5 研究室に増加した．さらに 1990 年には，1 研究部 7 研究室体制へと変更したほか，研究室のテーマ分担も研究の進捗に合わせて変更してきた．1992 年時点での研究室の分担と研究所組織を図 2.4.3に示す．

研究交流の一環として，毎年数名の海外研究者を数週間招聘し，特定の研究テーマで ICOT 研究者と意

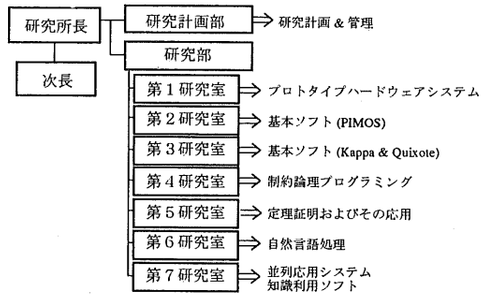


図 **2.4.3** ICOT 研究所組織と研究テーマ

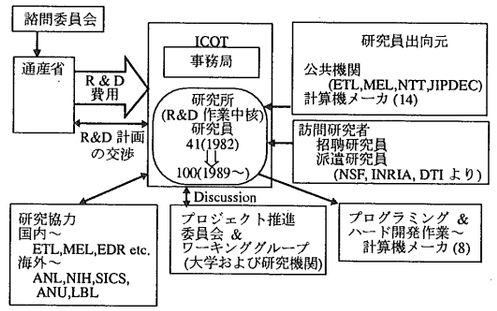


図 **2.4.4** FGCS プロジェクト推進体制

見交換を行なってきており，現在までに延べ 74 名を 12 か国から招聘した．

また，8 名の長期 (約 1 年) の研究者の派遣受入れを外国政府機関との覚書に基づき行なってきている．この覚書は米国 NSF (National Science Foundation)，仏国 INRIA (Institute National de Recherche en Informatique et Automatique)，英国 DTI (Department of Trade and Industry) と締結している．

全体のプロジェクト推進体制を図 2.4.4に示す．

研究開発にかかわる全費用は，通商産業省から ICOT への委託契約に基づき国が負担している．各段階と毎年の研究計画は通商産業省の承認を受けるが，このために通商産業省は本プロジェクトの諮問のための委員会を設置し，プロジェクトの計画と成果評価についてのアドバイスを受けてきた．

ICOT は中核的研究開発を行なうとともに，ハードウェアの製造やソフトウェアの開発などについてはコンピュータメーカに再委託契約を行なうことにより，全体のプロジェクトを遂行してきた．

このほか，ICOT は委員会やワーキンググループ

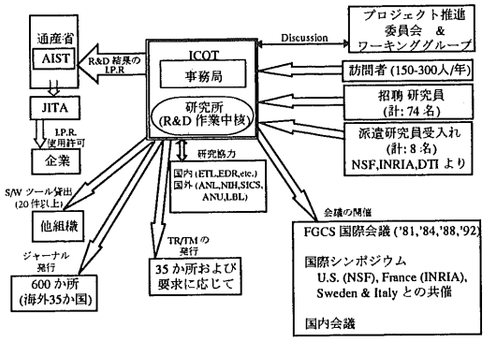


図 2.5.1 研究交流と研究開発成果の普及の枠組み

(WG) を設置し，大学や研究機関の研究指導者・研究者からの全体計画と成果や特定の研究テーマごとについての意見交換を行なってきた．この WG も研究進捗に合わせて変更してきているが，1WG 当たり 10～20 人程度の委員からなり，全体としては毎年 150～250 人程度の委員により構成されてきた．ほかに研究交流や研究成果普及の枠組みがあり，次章で述べる．

2.5 研究成果の普及と研究交流活動

本プロジェクトは国家プロジェクトとして，世界のコンピュータ科学分野への貢献を重視しており，ICOT 活動に関して研究開発の成果のみならず，研究のアイディアや過程についても論文や資料を通じて公表・説明を行なうこと，および外部の研究者や機関との研究交流を行なうことの努力を続けてきている．この努力は，並列処理や知識情報処理技術の進歩への貢献のみならず，この分野での研究コミュニティの拡大にも役立ってきており，今後ともプロジェクトの成果の普及を通じて，さらに拡大していくものと期待している．

また，多数の外部研究者から，ICOT 研究者との意見交換などを通じて，本プロジェクトは多大な貢献を受けてきている．たとえば，並列論理型言語の母体である GHC は Parlog や CP の研究者との交流の結果生まれており，ハードウェアシステム (PSI など) の性能改善には Warren 教授の提唱した WAM 命令セットの導入効果が大きく寄与している．

このような研究交流や研究開発成果の普及のための全体的枠組みを図 2.5.1 に示し，主な項目について以下に説明する．

a. 研究成果の公表

研究活動と成果の公表のために，ICOT ジャーナルと研究資料の出版・配布を行なっている．ICOT ジャーナルは季刊で ICOT 活動と研究資料の紹介を掲載しており，海外 35 か国，約 600 か所に配布されている．研究資料には研究論文 (TR) と研究メモ (TM) があり，現在までに約 700 の TR と約 1100 の TM を発行しており，その約 1/3 は英語版である．この TR/TM は海外約 30 か所に定期送付するとともに，要望に応じて個別に送付している．

b. 対外交流

ICOT 研究者が外部研究者と意見交換や研究討論を行なう次のような機会がある．

- 国内外での学会・ワークショップなどで論文発表や意見交換を行なっており，現在までに約 450 回の国際会議，約 1800 回の国内学会などで発表を行なってきた．また，多くの海外研究機関を訪問し，ICOT 活動の紹介や研究交流も行なってきた．
- 毎年，150～300 人程度の研究者や専門家などの ICOT 訪問を受け入れ，ICOT 活動の紹介や意見交換などを行なっている．
- 前章で述べたように，研究者の招聘や海外政府機関からの派遣研究者の受入れは中期に入って開始され，その研究成果は論文として発表されている．

c. シンポジウムなど

ICOT は種々のシンポジウムやワークショップ (WS) を開催し，研究成果や活動状況の発表や意見交換を行なってきた．

- 2.1 節で述べた FGCS 国際会議'81 に引き続き，前期の成果を FGCS 国際会議'84 (1984 年 11 月開催) で発表した．その後，中期の成果を FGCS 国際会議'88 (1988 年 11 月開催) で，後期の成果を FGCS 国際会議'92 (1992 年 6 月開催) でそれぞれ発表した．
- 国別シンポジウム・WS を次のとおり開催してきた．
  - 1983 年以降，日瑞 (または日瑞伊) WS を 7 回開催 (スウェーデンコンピュータ科学研究所 (SICS)，伊ピサ大学と共催)
  - 1986 年以降，日仏 AI シンポジウムを 4 回開催 (仏国 INRIA と共催)
  - 1987 年以降，日米 AI シンポジウムを 4 回開催 (米国 NSF と共催)
  - 1989 年以降，日英 WS を 2 回開催 (英国 DTI と共催)

d. 知的所有権

本プロジェクトの研究開発費用は全額国の負担であることから，特許などの知的所有権 (IPR) は日本政府に帰属する．この IPR は工業技術院 (AIST) が管理し，利用を希望する企業に対し，有償で差別なく許諾される．この利用に関する無差別な許諾は，本プロジェクトの成果としての IPR が政府所有になることにより可能となっており，普及の一つの枠組みとなっている．なお，すでに PSI や SIMPOS については，企業が利用許諾を受け商用化している．

e. 中間成果物

AIST により管理される段階にない本プロジェクトで研究開発中のソフトウェアは，研究開発ツールとして営利目的以外の利用目的に対して，利用を希望する企業・大学などに ICOT から貸与されてきた．

また 1992 年からは，ICOT フリーソフトウェアとして，公開に適すると判断されたソフトウェアの多くをソースプログラムやドキュメントを含めて自由にファイルコピーができるよう管理運用を始めている．

f. 海外との共同研究

論理プログラミング分野での研究交流の一環として，米国アルゴンヌ国立研究所 (ANL)，米国国立衛生研究所 (NIH)，米国ローレンスバークレイ研究所 (LBL)，スウェーデンの SICS，オーストラリア国立大学 (ANU) との共同研究活動を行なっている．

2.6 第五世代マシンの将来展望

LSI 技術は過去において，3 年で約 4 倍のトランジスタ数となる進歩を着実に遂げてきた．本プロジェクトは，この LSI の進歩を利用することにより，高並列処理でマシンの性能を大幅に向上させ得ることが可能かつ必要との認識に基づいている．

推論マシンの場合，通常の CPU より幾分多くの回路を必要とすることから，最初の PSI-I は 10 枚以上

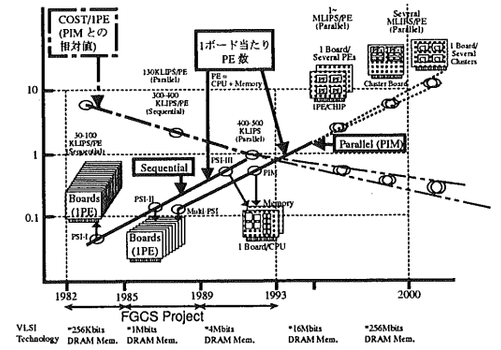


図 2.6.1 第五世代マシンの PE サイズとコスト傾向

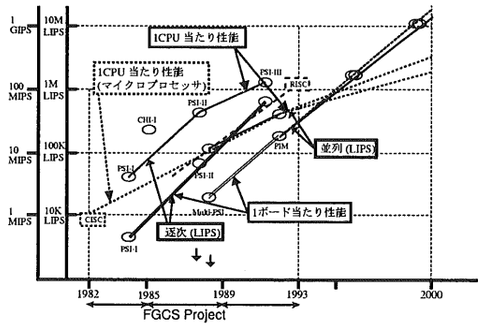


図 2.6.2 第五世代マシンの性能傾向

のボードが必要であったが，マルチ PSI/V2(PSI-II) では 4 枚に，PIM では 1 枚に減少してきた．ほかにメモリ 80MB に対して，PSI-I は 16 枚，マルチ PSI/V2 は 4 枚，PIM(PIM/m) は 1 枚となった．

要素プロセッサ (PE:CPU とメモリ) 当たりのボード数，および PE 当たりのコストの過去の傾向と今後の予測値を図 2.6.1に示す．

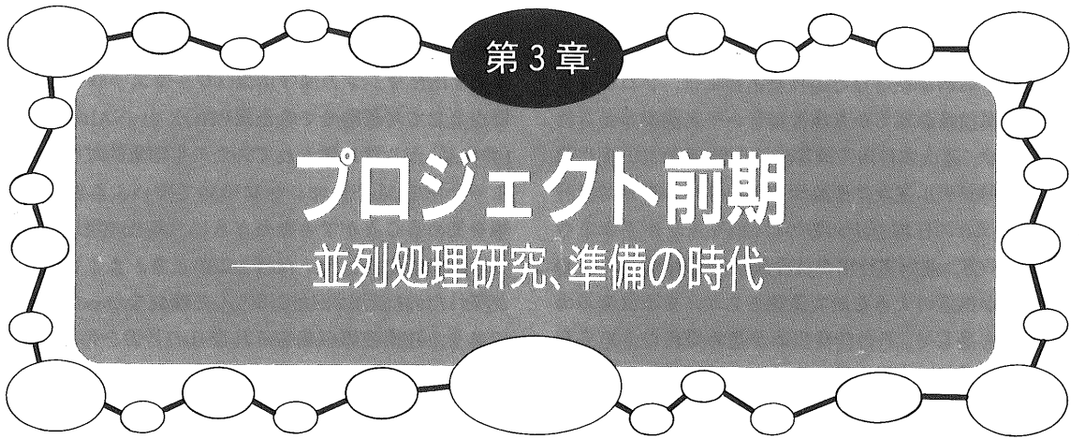
図 2.6.1においては，2000 年までには 10 程度の PE が 1 ボードに搭載され，デスクサイド筐体に 100PE 程度，大型筐体で 1000PE 程度が実装され得ると予測できる．また，コストについても 3 年で約半減していくことになると予測している．

第五世代マシンの性能予測を，本プロジェクトの現在までの実績値をベースとして図 2.6.2に示す．逐次型推論処理性能は，現在までに 3 年で約 4 倍になっている．並列型推論処理性能向上は逐次型の場合よりは小さく，マルチ PSI から PIM で約 2.5 倍程度である．図 2.6.2には，参考までに現在までの CISC および RISC

マイクロプロセッサの性能向上も示してある．将来のボード当たりの性能は，2000 年ころには PIM の 100 倍程度の 20MLIPS を，大型筐体の推論マシンの性能では 1GLIPS を達成可能と予測している．

本プロジェクトにおいて，CAD，定理証明，遺伝子情報処理，自然言語処理，法的推論などのいくつかの応用分野についてのプログラムを開発してきたが，これは近い将来における並列処理の適用分野の開拓の目的も含まれているものである．

並列処理の技術は近い将来においてコンピュータ処理へ汎用的に適用され，産業・社会の種々の分野へ拡大されていき，今後の並列処理技術と知識処理技術の進歩に伴って，徐々に並列処理応用プログラムの実用化が進むものと考えられる．



3.1 プロジェクト前期について

プロジェクト前期は，あらゆることが手探りの状態から始まった．並列推論システムの研究はいうに及ばず，自分達が使う計算機環境の整備から文献集めや基礎技術の勉強まで，ほとんどゼロからの出発であった．

1982 年の 6 月 1 日に着任してすぐ次の日だったかに，論文の束をどさっとわたされ，「こんなに読むの?」と目を白黒させた記憶がある．このときの研究者はまだ 40 名で，パーティションで向かい側と仕切られた机の列が，がらんとした大部屋のなかにただ並んでいるだけの環境であった．窓の向うにそびえ立つ東京タワーがひたすら印象的であった．計算機といえば，Prolog が走るパソコンとして，AppleII と沖電気の if800 が数台あるだけであった．少し遅れて C-Prolog が走る PDP-11 が入り，冬を迎えてようやく待望の DEC10-Prolog が使える DEC2060 が届き，VT-100 というただのキャラクターディスプレイ端末が，当時としてはぜいたくにも一人に 1 台ずつ与えられた．これらの機器調達を指揮していたのは，それらの使用経験をもち米国の研究環境のすばらしさをも心得ていた電子技術総合研究所からきた人々であったように思う．

夏から秋にかけて，着任したての研究員たちが Prolog の勉強や文献読みに励んでいる間，ICOT の首脳陣 (渕一博 ICOT 研究所長を筆頭に，電総研，NTT，情報処理開発協会などからの出向者で固められていた) はといえば，具体的に何をどのように研究してゆくのか，何を作ってゆくのかといった研究計画の詳細化に追われていたに違いない．プロジェクト前期の研究テーマのなかで，最終的な並列推論システムと深いかかわり合いをもつことになるのは次の三つであった．

第一は，プロジェクトの準備段階から比較的その開発方針が具体的に決まっていた「開発支援システム，ソフトウェア開発用パイロットモデル，逐次型推論マシン」と呼ばれる長い名前のものであった．そのなかでも，パーソナル逐次型推論マシン (PSI) は，論理型言語を使って知識処理のソフトウェアの研究を進めるためのワークステーションと位置づけられていた．MIT や XEROX Palo Alto 研究所の LISP マシンが目標イメージとされていた．パーソナル逐次型推論マシンは，ソフトウェア研究用の道具であるだけでなく，ほかにも重要な役割が与えられていた．一つは，論理型言語を効率よく実行する専用マシンの実現技術を身につけることであり，もう一つは，論理型言語で OS をはじめとするソフトウェアをすべて記述し，論理型言語の実用性を確認するとともにその取扱いに習熟することであった．

第二は，並列推論マシンを実現するための基本方式の研究であったが，これが難物であった．当時，論理型言語としては Prolog ぐらいしかなく，並列処理といえば，データフローやリダクション方式が研究されていた．逐次型計算機の CPU とメモリ間の情報の往復が，フォンノイマン・ボトルネックと呼ばれて性能向上を阻害する元凶とされていた．データフローやリダクション方式は，ボトルネックを解消する救世主 (かもしれない) として期待を集めていた．したがって，こ

れらの期待される方式で Prolog の並列実行を試みることから仕事が始まった.

システムの中核となる論理型言語には,プロジェクトの準備段階のうちから核言語という名前が与えられていたが,逐次実行用の核言語第 0 版 KL0 は,先に述べたパーソナル逐次型推論マシンの開発のなかで設計された.その仕様を決めるのには Prolog が手本とされた.一方,並列実行用の核言語 (核言語第 1 版 KL1) の開発が第三の大きな研究課題であり,まさに産みの苦しみを通して,その仕様固めが進められた.そうしてプロジェクト前期末にようやく,KL1 の基本仕様となるべき並列論理型言語 GHC が生まれた.

以下では,これらの 3 種の研究開発について,その概要をエピソードを交えながら紹介する.なお次節は筆者が直接担当したテーマであり,記述が他より詳細になっていることをご了承願いたい.

3.2 パーソナル逐次型推論マシン PSI

3.2.1 技術調査を命ず

1982 年 6 月末,ICOT 研究所がスタートしてひと月もたたない頃,一つの海外出張命令が出された.「論理型言語専用のワークステーションを開発する.手本は米国で商用化されている LISP マシンである.ついては関連技術を調査し報告すること.」出張命令を受けたのは,言語開発のグループに配属されていた近山隆,自然言語処理の安川秀樹,ハードウェア開発の瀧和男であり,いずれも 28 歳から 30 歳の若手研究者だった.人工知能と知識処理研究の動向調査を兼ねた米国出張であった.

最初の訪問地ピッツバーグでは,AI 関係の大きな国際会議である AAAI が開かれており,デモンストレーションのブースでは,発表されたばかりの各社の LISP マシンを目のあたりに見ることができた.第五世代コンピュータプロジェクトのスタートとともに AI ブームが登り坂で,会場は活気に溢れていた.

そこでは,XEROX 社の LISP マシンシリーズ 3 機種 (Dolphin, Dandelion, そして ECL デバイスを使った超高速 LISP マシン Dorado) をはじめとして,MIT 出身の人たちが作った二つの LISP マシン会社 Symbolics 社と LMI 社からは,MIT でプロトタイプが作られた CADR マシンの商用版,そしてまだ部分稼働状態の Symbolics3600 が展示実演されていた.ほかに LISP マシンの保守用に,ワークステーションの原点として名を馳せていた XEROX 社の Alto マシン (非売品) が片隅に置かれていたのも印象的だった.ここ一か所で,LISP マシンについてずいぶん多くの情報を集めることができたとともに,各社の現地オフィスも精力的に訪ねた.ただしこのとき,まともに英語が喋れたのは近山だけでほとんど喋れなかったのが瀧であり,技術情報収集もそれなりの苦労を伴った.

西海岸に移って,サンフランシスコ近郊では,設立後 4 か月ほどしかたたない Sun Microsystems 社を訪れた.われわれが作るワークステーションにもマルチウィンドウシステムを装備することが必要条件とされており,そのためにビットマップディスプレイのハードウェアが必要だった.当時国内では,関連技術はまだ研究・開発中であり,すぐ使えるものがなかった.そこで Sun ワークステーションのビットマップディスプレイ部分を流用できないか,調査することが目的だった.けれども Sun-1 の試作品を見た結果は,信頼性に問題ありということになった.思い出すのは,いかにも町工場で組み立てたようなキーボードを押すと,キーがひっかかって戻らなかったことである.当然この時点では,同社と UNIX 文化の今日の隆盛を予測することは困難であった.

ロサンゼルスでは,Symbolics 社と LMI 社をそれぞれ訪れて歓待を受けた.技術的な質問に答えてもらうとともに,工場見学もさせてもらった.結論は,「ハードウェアは特に驚くほどのものではない」「ソフトウェアは触って使い込んでみる必要がある」「ビットマップディスプレイは自前で何とかする必要がある」ということだった.

ちなみにこのとき,瀧は LISP マシンを作った経験があり,近山は高速の LISP 処理系の開発者であり,安川は自然言語分野での LISP のユーザだった.後に ICOT では Symbolics3600 を購入し,プロジェクト中期のはじめ頃まで使った.

3.2.2 1 年で開発せよ

夏がすぎ秋に入って,ICOT の研究員たちは Prolog のプログラミングと Prolog の言語処理系の勉強,それぞれの担当分野の文献読みに励む毎日であった.Prolog の使用経験をもつ数名の研究員が先頭に立って勉強会を企画した.研究員はすべて出向者であったが,渕所長の「年齢の上限を 35 歳にする」という方針がみごとに成功して,メーカの壁をまったく感じさせない研究所の雰囲気ができていた.

秋も深まった頃,渕所長からお達しが出た.「そろそろ SIM-P の設計にとりかかってもらいたい.1 年で開発するように.」SIM というのは逐次型推論マシン (Sequential Inference Machine) の開発コードで,並列推論マシン (Parallel Inference Machine) と対になった名称である.SIM-P はそのなかのパーソナル版,すなわちワークステーション仕立てにしたシステムのことで,のちにパーソナル逐次型推論マシン PSI と命名されるシステムである.

このお達しには,担当研究員一同ギョッとした.誰も作ったことのない論理型言語専用のワークステーションで,しかもこれから自分たちが研究の道具として使っていけるだけのしっかりしたものを 1 年で開発せよといわれたのである.1 年はあまりにも短かすぎるという研究員の反論に,渕所長はこともなげに「LISP マシンができているのだから Prolog マシンだってできるだろ.」こんなことをいえるのは,作るモノのことをまったく知らない人か知り抜いている人かのどちらかである.所長は電総研で LISP マシンや Prolog 言語の経験もあることから,どうも後者のようであった.あとから思えばこのとき渕所長は,プロジェクトの先行きを占うのに PSI 開発の成功に賭けていたのかもしれない.

渕所長の信念を支えていたのは,やはり電総研からやってきた古川康一第二研究室長,横井俊夫第三研究室長,そして若手の内田俊一第一研究室室長代理らの ICOT 研究所首脳陣だった.夏の米国における技術調査を受けて,これらの人たちはプロジェクト前期の目玉ともなるべきパーソナル逐次型推論マシン PSI のハードウェアと,中核となる論理型言語と,その言語で記述する OS をはじめとしたソフトウェアシステムの開発方針について,連日のように議論を重ねたに違いないのだが,詳しい様子はヒラの研究員には見えてこなかった.

パーソナル逐次型推論マシン PSI の開発には,いくつもの重要な役割が与えられていた.一つは,論理型言語をベースとして自分たちが研究開発を進めるための使い勝手のよい道具を作ること,また一つは,論理型言語を効率よく実行する専用マシン作りの小手調べとなること,さらにもう一つは,論理型言語で OS をはじめとするソフトウェアをすべて記述し,論理型言語の実用性や処理性能を確認するとともに,その取扱いに習熟することであった.あわせて,当時騒がれていたソフトウェア危機に対処する一つの方法として,オブジェクト指向を採り入れ,ソフトウェアの生産性を高める試みも盛り込むことになっていた.これらの盛りだくさんな企てを進めるために,およそ次のような役割分担がされていた.核言語は古川,PSI のソフトウェアシステムは横井,ハードウェアシステムは内田であった.

1982 年 12 月,ICOT には大型機である DEC2060 が導入され,その上で当時としては高速で安定した DEC10-Prolog が利用可能になったが,研究者が高度な研究のためのツールとして共同利用するにはパワー不足であった.そのため,PSI を作って各人に少なくとも DEC2060 上の DEC10-Prolog と同等の推論パワーを提供することが目標とされた.

3.2.3 PSI を設計する

秋の終わりから PSI の開発担当グループはにわかに忙しくなった.「1 年でどうやって開発するのだ」という思いもあったが,渕所長の決意は固いのだから,とにかくやってみようということになった.開発するものは山ほどある.

ここで「開発」という言葉ばかり出てきて「研究」が出てこないのであるが,それは短い期間で作り上げなければならないという気持ちの表われのほかに,「作って使い込んで試してみることが研究である」という一つのスタイルを象徴していることのように思われる.実際にプロジェクトを通して,ハードだけでなくソフトウェアもしっかり作って試してみることが,重要な方針として貫かれてきた.もちろん,しっかりした設計思想や方針なしに作ることはいけないのだが,「机に向かうだけが研究ではない」というのが所長の言葉であった.

PSI を開発するのにやらなければならないことは大きく三つに分かれた.中核となる論理型言語,核言語第 0 版 (KL0) とシステム記述言語の設計,オペレーティングシステム (OS) とプログラミングシステ

ム (PS) の開発，そしてハードウェアの開発である．これらを並行して進めるために，数人ずつの研究開発グループが組織された．

PSI のシステムイメージは LISP マシンが参考にされており，逐次論理型言語 KL0 を実行する高級言語マシンになるというのが研究者間での了解事項であった．KL0 はハードとソフトのインタフェースであるとともに，高級言語マシンの機械語として位置づけられた．本来なら，KL0 の設計が完了してからハードウェアと OS/PS の設計に入るのが順序なのだが，1 年で作るとなるとそうもいっていられない．KL0 はとりあえず Prolog とほとんど同じレベルの言語とし，その上にシステム記述用の言語としてオブジェクト指向機能を入れた論理型言語 (のちに ESP と命名される) を実現するというストーリーにして，言語設計と OS/PS の開発とハードウェア開発を全部並行して走らせた．

KL0 の検討には近山と横田実，服部隆が抜擢され，3 か月ほどの集中検討を行なった．Prolog をベースとして PSI の機械語にふさわしい言語仕様を模索した．その後, PSI の開発が進むのと並行して近山があとを引き継ぎ，KL0 の上に実現するシステム記述言語 ESP(Extended Self-contained Prolog) を設計した．

オペレーティングシステムとプログラミングシステム (OS/PS) の開発も難物と考えられた．当時 OS は，日本のコンピュータメーカにとって独自開発など考えることもできない複雑極まる高度ソフトウェアと思われていた．それを論理型言語とオブジェクト指向という実績のない技術で作り上げようというのだから，これは大変な実験である．Prolog のプログラミングとともに，オブジェクト指向による設計方法の勉強と記述実験から始めた．黒川利明，辻順一郎，服部，近山らがこれを進めた．黒川は電総研の LISP システム他を手掛けた LISP のプロであり，服部は OS の造りに詳しかった．

PSI のハードウェアは，本来 KL0 を高速実行するための専用ハードであって，そのためにカリカリにチューンしてあるべきものだが，KL0 の仕様も固まっていないのだからチューニングなど望むべくもない．KL0 はほぼ Prolog と同じだということで，Prolog を効率よく実行することのできそうな融通のきくハードウェアの設計を目指した．高級言語マシンの言語処理系はハードウェアの一部だということで，KL0 の言語処理系もハードグループが担当した．チーム構成は，山本明，西川宏，横田，瀧であった．

横田は，日本電気で COBOL 用の高級言語マシンを開発した経験と Prolog 処理系の経験をもち合わせており，KL0 の実行メカニズムを設計するとともにハードでサポートすべき機能を洗い出した．瀧は LISP マシンの経験を生かして西川と一緒に，マイクロプログラム制御で KL0 を実行するためのハードウェアの骨格を設計した．山本はワークステーションに必須の入出力部分を担当した．基本検討段階では，内田を交えて何回も深夜に及ぶミーティングを繰り返したのち，次第に詳細検討に進んだ．

冬を迎える頃，内田から指令が飛んだ.「何を作らされるかわからないので，メーカがビビっている．PSI とはこういうものだということを製造メーカにわからせるための機能仕様書を作ること.」 ICOT 首脳陣がちょうどプロジェクトの研究開発テーマの何をどのメーカが担当するかの最終的な詰めを行なっていた頃であろう．

暮れから新年にかけて，正月休みもそこそこにハードウェアチームは PSI の機能仕様書作りに没頭した．なお, この頃はまだ PSI ではなくて SIM-P と呼ばれており，Personal Sequential Inference マシンの頭文字をとった PSI という名前と，横田が提案したΨマークが定着したのは 1983 年の後半であった．

機能仕様書はいつの間にか 2cm の厚みになった．データワードにはタグビットをもち，大きなレジスタファイルと Prolog のユニフィケーションに役立つ 2 組のアドレスレジスタを備え，タグを判定して高速分岐できる水平型マイクロ命令制御の専用 CPU が設計された．Prolog のスタック操作に適するコマンドを備えたキャッシュメモリまで，使用 IC の品種も指定して設計できていた．一つの懸案は，割込みやメモリ管理などのハードと OS のインタフェースであったが，これには必要とされる機能を組込み述語としてマイクロプログラムで実現するつもりにして，とりあえず必要とされるであろう組込み機能の一式を瀧がまとめてでっちあげた．こうして，機能仕様書が完成し，作ってくれるメーカとの対面を待つばかりとなった．思えば理想的な Prolog 専用ハードウェアには少し距離があったものの，とりあえずは作れそうなシステムの設計がこれだけ短期間のうちにまとまったのは驚異的なことだったかもしれない．

3.2.4 幸運に恵まれたメーカスタッフとの巡り合わせ

PSI の開発担当は三菱電機と沖電気に決まった．何を作らされるか心配していたのが，機能仕様書を見せられて結構当たり前の CPU だったので安心した，とメーカ担当が話していたことをあとになって内田から聞かされた．

1983 年 3 月に入って，ICOT と三菱電機の担当者レベルの開発ミーティングが始まった．三菱のハードウェア担当は，情報電子研究所の中島克人，中島浩，三石彰純らであった．非常に力のあるスタッフを揃えてくれたという印象だったが，すぐにこれは理想に近いメンバであることがわかった．中島克人 (以下，克人と略す) は京大長尾研で LISP マシンを手掛けた経験をもち，その時期はちょうど瀧が神戸大で LISP マシンを開発した頃で, 研究会でも顔を合わせていた．中島浩 (以下，浩と略す) は克人の後輩で，京大萩原研でマイクロプログラム計算機の開発経験をもっていた．また，三石も阪大安井研で LISP マシン開発メンバの一人だったのである．いずれも高級言語マシンの勘どころは説明しなくても押えていて，技術的に詳細な話へ進むのは早かった．

定期的なミーティングを開いて，1 か月程度で技術的な詳細を伝え終わった．機能仕様書でかなりの詳細設計まで済んでいたので，あとは製造ラインに適合するように使用部品などを手直しして回路図を起こせばよいはずだった．ところが, ちょっとした番狂わせがもち上がった．

> 「当初 ICOT も三菱電機の担当者も試作レベルでよいとの判断から, メモリを除いて RAS 機能 (信頼性を上げるための異常や故障の検出機能の類) がほとんど設けられなかったが，いざ三菱電機の工場へ製造の依頼にいくと，『こんな信頼性の低いものは作れない』と断られ，パリティ機能などの付加のため，設計の大変更を余儀なくされた.」(克人談)

このおかげで CPU のプリント板枚数は 8 枚から 12 枚に増え，再設計のために製造開始は 1 か月半ほど遅れたのだが，ともあれ工場のラインに乗り, 信頼性も高いマシンができてくることになった．

この三菱スタッフとの出会いにはもう一つの裏話がある．1983 年当初，巷ではかの IBM 事件がコンピュータ業界を揺さぶっていた．某日本の大手コンピュータメーカ社員が，IBM の次期拡張 OS の技術情報を非合法的に入手しようとしたところをオトリ捜査で逮捕されたのである．この事件をきっかけに，国内メーカ数社における IBM 互換計算機の開発計画は大きな軌道修正を迫られた．この事件がなかったならば，三菱のあれだけのスタッフが PSI 開発にはまわってこなかっただろうという噂も囁かれたが，真偽のほどは定かでない．

一方，沖電気もハードウェアを製造することになっていた．こちらは ICOT の仲介で三菱電機の開発結果を一部引き継いで，マイクロプログラムレベルで互換性のある PSI を時期的にはやや遅れて製造した．

また懸案の一つであったビットマップディスプレイは，幸いにも両社にそれぞれ試作レベルの技術があり，これを手直しして PSI の汎用 I/O バスであるマルチバスに接続することで問題をクリアした．

3.2.5 クリスマスプレゼント

設計開始当初は，どうやって 1 年で開発を終えるのか予想もつかない状態だったのが，いろいろな幸いと担当者たちの努力と，もしかしたら所長の念力のおかげで何とか工場のラインに押し込むことができ，あとはでき上がるのを待つ段階まできていた．

その間も，KL0 処理系の開発と，マシンが届いてから処理系を動かすための環境整備に忙しい日々が続いた．最初に届くマシンは，マイクロプログラムの実機デバッグの機能などもきわめて不十分な裸同然のマシンのはずであったから，ミニコンピュータの PDP-11 を別に用意してこれにパラレル I/O をつなぎ，そこを通してマイクロプログラムやメモリ上のデータのロードをはじめとして，あらゆるマイクロプログラム動作を外部から制御できるインタフェースを用意した．そして, PDP-11 上に PSI のマイクロプログラムデバッグ用のコンソール機能を実現した．このインタフェースは PSI にとって，命を吹き込むヘソの緒のようなものだった．

1983 年のクリスマスイブ，ついに待望の PSI 1 号

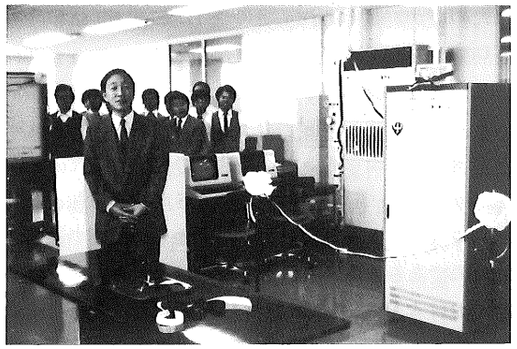


図 3.2.1 PSI 1 号機の火入れ式 (1983 年クリスマス). 左は渕所長. PSI の前面扉に貼ってあるのは横田氏考案のΨのロゴマーク

機は三菱電機の計算機製作所から ICOT に到着した. この前後にはいろいろな逸話が伝えられている.

「三菱電機では, PSI の組立てのための場所がなかなか確保できず, 結局組立て作業が行なわれたのはプレハブ建ての倉庫の一角であったため, 納入時期ともあいまって, PSI の完成はよくキリストの誕生になぞらえられたものである.」(川合談)

「三菱で最初に動かした 0 号機は, 山ほどジャンパー線を飛ばしたうえ IC を載せるスペースがなくなってしまって, それでも調整の期限に迫られ IC を二階建てに半田づけして急場をしのいだ.」(浩談)

「キンピカ IC の載った 5MB のメモリボードが 400 万円もするそうだが, 1 枚抜いて秋葉原へ売りにいこうか[†1]. 16 枚も差さってるから 1 枚ぐらいわかるまい.」(詠人不知)

こうして納入された PSI 1 号機は, クリスマス当日に無事火入れの儀式を迎えることができた (図 3.2.1). このイベント以後, 試作マシンの ICOT 搬入は, クリスマスプレゼントとして行なわれることが恒例となった. ともかくも, はじめは誰しも 1 年でできるとは思わなかった PSI の開発は, ハードウェアに関しては渕所長の号令どおりに本当にほぼ 1 年で仕上がったのであり, メーカにおける通常の製品開発スピードをも上回ったのである. かなりの人にとってこれは奇跡と映ったに違いないし, プロジェクトの幸先のよさを印象づけることになった. かくして目標はハードウェア作りから, OS を含めて動かすこと, そして 1984 年秋の国際会議 FGCS'84 に出展することへと移った.

3.2.6 KL0 処理系と ESP

PSI の到着後, KL0 処理系の開発チームが俄然忙しくなった. 何せ 3 月には OS 開発グループに, 動く形で引きわたさなければならないのである. 1984 年 1 月はじめに PSI 2 号機が到着して, KL0 処理系マイクロプログラムの実機デバッグは本格的になった.

KL0 はほとんど Prolog と同じレベルの言語であって, これを実行する処理系をマイクロプログラムで実現するというのは, マイクロプログラムで高級言語のインタプリタを書くことと同義であった. そんな複雑なものをマイクロプログラムでは無理だと思うのが常識的であったが, 高級言語マシンの経験者が 4 人も含まれている合同チームはそれを当たり前のことのようにやってしまった. このためには三つの努力を払った. 一つはハードウェア設計の段階でマイクロプログラムの能力が高くなるように, そして記述によけいな注意を払う必要がないように心がけた. 二つ目はほとんど C 言語と同じ気分でプログラムが書けるように, マイクロプログラムアセンブラをうんと賢くした. これを実現するのに Prolog を用いた. 三つ目はデバッグを楽にするために, レジスタ転送レベルの PSI のシミュレータを DEC2060 上に作り, 実機に移ってからのデバッグの負担を軽減した. これらのおかげで, PSI のハードが到着する以前にマイクロプログラムのある程度の部分がシミュレータ上で動いていて, それをハードウェアのテストに利用することができた.

マイクロプログラムの開発は横田が指揮をとり, 西川, 山本や, 三菱電機の克人, 池田, 三菱の外注陣, そしてシミュレータの開発には沖電気からも参加した. デバッグ段階ではシミュレータが威力を発揮して, 実機に移ってから出るバグの割合は低かった.

こうして 4 月には KL0 処理系が何とか動くようになり, OS 開発部隊にはまだバグを抱えた状態で引きわたした. その後横田の指揮で, バグ取りと並行してコードの見直しが行なわれ, 処理速度を改善する努力が行なわれた.

一方, KL0 の上に実装されて OS の記述に使われるはずだった ESP はというと, 1983 年の後半になって近山の手により設計が進められ, 最初は DEC2060 上に OS 開発用の処理系が Prolog を用いて実現された. Prolog と同じレベルの KL0 に対して, オブジェクト指向の機能をどのように入れユーザに見せるかについて, 言語グループのなかで激しい議論が行なわれたようである. ESP プログラムは, コンパイラにより KL0 に変換されて実機上で実行される.

3.2.7 SIMPOS を動かす

オブジェクト指向で OS を設計して論理型言語の ESP で書くという試みは, ある意味で大変な冒険だったといえる. SIMPOS は Sequential Inference Machine Programming and Operating System の頭文字を取ったものであるが, OS 開発グループのなかでは「進歩す」という字を当てて頑張っていた.

オブジェクト指向のモデルに従って OS 機能をモジュール (オブジェクト指向ではクラスと呼ぶ) に分割整理し, それぞれの機能仕様をなるべく形式を整えて記述する作業 (クラス仕様書の作成) が続けられ, 1983 年の終わりから 1984 年の前半にかけて, ESP でのプログラミング作業が最盛期を迎えたように思われる. オブジェクト指向のプログラム開発など, 当時誰も経験したことがなかったので, プログラム設計にしっかり時間をかけてクラス仕様書にすることが重要な作業であったし, 次にそれを ESP で記述する段階では, プログラムのコツを ESP の設計者自ら各担当に伝授してまわることが必要であった.

SIMPOS の開発は, オペレーティングシステムとプログラミングシステムの両方を含む幅の広いものであり, そのため開発チームも ICOT メンバをはじめとして, 三菱電機, 沖電気, 日本電気, それらの関連外注を含む大規模なものとなり, 最盛期には 40 名前後のメンバが働いていた.

1984 年も 6 月に入って, ようやく PSI 上での SIMPOS の実機デバッグが本格的に始まった. この頃までに, PSI の実機へプログラムをロードしたりデバッグしたりする環境はしだいに整い, PDP-11 からつながるヘソの緒を経由しなくても PSI 本体のフロッピーディスクが使えるようになり, とくにプログラムのロード時間は大幅に短縮された.

「当時, PSI へのプログラムロードは, DEC2060 上でクロスコンパイルされた後, PDP-11 (LSI-11) と呼ばれるミニコンを経由してダウンラインロードされていた. しかし, このロードが非常に遅く, デバッグ中に一度 PSI を落としてしまうと, 再立上げに 1 時間近くかかっていた.

その後 SIMPOS のデバッグや開発が進むにつれて, ロード元が PSI のフロッピーディスク (8 インチ) を経て, 内蔵ハードディスクへと移り変わり, 立上げ時間は最終的に 10 分程度となった. (なお, このときの PSI に装備されていたディスクの容量は 72M バイトで, 主記憶容量の 80M バイトよりも小さかった.) われわれは, この立上げの高速化を, 立上げの間に読める漫画の冊数で実感することができた.」(川合談)

こうして PSI からヘソの緒が取れ, システムとして一人立ちできるようになってきたのであるが, SIMPOS のデバッグが順調に進み始めたと思った矢先, 重大な問題がもち上がった.

「ウィンドウシステムのデバッグに移ったところで大きな壁にぶつかった. 一言でいってしまえば, 遅すぎるのである. ビットマップディスプレイにウィンドウが表示されるまではよいが, マウスを動かしてマウスマーカが数ドット移動するのに数分もかかってしまう. PSI は, この年の秋に開かれる FGCS'84 の目玉であり, さすがにこれでは使いものにならないということで, ファームウェア担当者と協力して, 近山をはじめ, プロセス管理担当の吉田 (当時 MRI, 現在九大), ウィンドウ担当の飯間 (沖電気), 中澤 (沖電気), 榎本 (松下電器) などが必死の高速化を行なった.

当時の SIMPOS は, カーネルや入出力部分の細部まで忠実にオブジェクト指向的にプログラミングされていた. このため高速化は, オブジェクト呼出しそのものの高速

†1 当時 256 キロビットのダイナミックメモリ (DRAM) チップがようやく出だしたところであり, 初期のロットには金メッキが施されていたので, いやがうえにも貴重品の雰囲気が漂っていた. 1 システム 80 メガバイト (MB) のメモリは当時の大型コンピュータ以上で人々を驚かせた.

図 3.2.2 国際会議FGCS'84におけるPSIのデモ風景．PSIは箱のデザインが新しくなった．手前下はハーモナイザのデモに使ったヤマハのDX7

化と，プログラムの内側のループにあたる部分をなるべくオブジェクト呼出しが少ないように書き直すことで行なった．

デバッグの進行と並行して，PSIの実機も次々と納入されてきたため，デバッグの効率も上がってきた．ただ，PSIはエラーで落ちたときにブザー音(ピー音)を発するが，電源投入時にも同じ音を発するため，デバッグ中にほかのPSIに電源を入れられると非常に心臓に悪かった覚えがある．」(川合談)

こうしてSIMPOSは，バグ取りと改良を繰り返しながら，一歩一歩使いものになるレベルに近づいていった．このとき内田が感慨を込めていったことは，「ハードは3倍も速くすることは大変なことだが，ソフトは元のプログラム次第で100倍速くすることもできてしまう．」それに対して近山は，「ESPを使うと100倍遅くてもとりあえず正しく動くソフトが書けるというのはすごいことなのだ」とやり返した．どちらも忘れられない言葉である．

3.2.8 FGCS'84 国際会議

PSIは7号機以降は箱の角がとれて見やすいデザインとなり，このモデルがFGCS'84に出展されることになっていた．夏から秋にかけてのSIMPOS高速化の努力が功を奏して，マルチウィンドウを使った入出力も見苦しくない程度に高速化が進み，バグをとって安定動作させることに注意が注がれてきた．

一方で，FGCS'84に出すのにOSだけ動いていても仕方ないので，見せるためのプログラムの準備が秋になってから始まった．OSグループのほうでは，OS自体の説明をグラフィック表示するプログラムと，15パズルの縮小版の8パズル(15パズルでは時間がかかりすぎて解けない)と，動き始めたばかりのネットワーク機能のデモを行なうことになった．けれども，もう少し派手なものも，と思っていた瀧は，知識処理らしくて，ハードウェアも十分に動かして，見ていても楽しいネタを近山と相談して計画した．これはPSIの上の応用プログラムらしいプログラムの第1号となった．

ハーモナイザと名づけられたプログラムは，ミュージックシンセサイザのヤマハDX7を使って，指1本で弾いたメロディを取り込み，その場で4声のハーモニーをつけて演奏するというものになった．瀧はPSIとDX7をつなぐためのMIDIインタフェースも作ってしまった．和声の進行に関する知識は，近山の奥さんから音大の教科書を借用して，4分冊の本の1冊目の半分だけから知識を拾い出してプログラムに組み込んだ(あとは時間切れで入らなかった)．こうしてできたプログラムは，論理型言語で書いてありながら実時間の入出力もし，知識処理もし，そして魅力的な音も奏でる実にデモンストレーション向きのものになった(図3.2.2)．入れた知識のせいか，クラシックな曲でないとハーモニーが美しくないというおかしな特徴もあったのだが．11月末のFGCS'84直前まで，約2か月のプログラム開発期間であり，たいそう切迫した時期のはずだったが，瀧はこの開発をほとんど趣味のように楽しんでしまった．

FGCS'84のデモでは，これがほんとうに2年少しの期間でまったく新規に作られたコンピュータシステムであって，しかも論理型言語ですべてが書かれているシステムなのだろうかという，疑いと驚嘆の思いを観衆に抱かせたに違いない．実際に，この計算機は何をベースに作られているのか(元があるはずだ)，といった質問や，論理型言語で書かれているのがほんとうなら見せてみろ，といった声や，アメリカのDARPA(国防総省)からは，売るなら買いたいといった声がかかった．論理型言語に関する実績がまだなかった時代(特に米国では)といった背景も影響していただろう．

しかし表の華やかさに比べて，裏方の神経の使い方は並大抵ではなかった．

「このときのPSIは，安定動作というにはほど遠く，マウスのわずかなクリックミスなどでもマシンが即ダウンしてしまうため，デモはまさに薄氷を踏む思いであった．また，ガーベジコレクション(GC)も実装されていなかったため，1回のデモが終わるとすぐに立ち上げ直した．立上げはフロッピーディスクから行なったが，SIMPOSには多数のパッチ，パッチのパッチ，パッチのパッチのパッチ...があたっており，14，5枚のフロッピーを順序正しく読み込ませる必要があった．またアプリケーションプログラムを実機でコンパイルするソフトなども未実装で，すべてICOTのDEC2060でクロスコンパイルしたものをもってきていた．」(川合談)

そんなことを観客は知るよしもなく，立派に動いているシステムだと思ってくれたようである．

この国際会議の成功で，第五世代コンピュータプロジェクトの好調な滑り出しをいやがうえにも世界に印象づけることになった．またPSIの完成によって，大型計算機と同じ推論計算パワーが研究員一人ひとりに与えられる道が開かれた．ハードウェアグループにとっては論理型言語専用マシンを作るときの勘どころがつかめてきたし，OSグループにとってはOSなど自分たちで作れるのだという自信となった．そしてICOT研究所の首脳陣は，論理型言語を核にしてプロジェクトを進めることはやはり間違っていない，という確信を深めることになった．

3.3 並列推論の研究・手探りの時代

3.3.1 プロジェクト前期の並列推論の研究

プロジェクトの開始にあたって，その最終目標の一つに，プロセッサ1000台からなる超高性能な並列推論マシンが掲げられていた．1000台という数は，マシンを構成するためのLSI技術の発展を予測したうえで，実現できそうな数としてはじき出されたものであった．けれどもそれをどのような方式で実現するか，その上で実行するプログラム言語は何になるのか，そもそも1000台ものプロセッサを効率よく動かすにはどうすればよいのか，研究の手がかりをつかむことすら容易ではない状況だった．

したがってプロジェクトの前期は，並列推論言語，並列推論の実行モデル，並列推論マシンの方式に関して，研究の手がかりをつかみ研究開発の方向性を確定していくことが重要な仕事だったのである．そのために，要素技術として重要となりそうな数個のテーマについて研究開発をスタートした．並列推論言語については次節で詳しく述べることにし，ここでは前期の並列推論モデルおよびマシンの研究開発について簡単に紹介する．

3.3.2 キーワードは非ノイマン・マシン

プロジェクト開始当時，計算機の新しい潮流として「非ノイマン・マシン」という言葉が流行していた．ノイマン型計算機の実行モデルが，記憶装置と処理装置との間の細い接続路を命令とデータが往来するものであって，これが計算機の処理性能向上を阻害する隘路(フォンノイマン・ボトルネック)であるといわれていた．またこのモデルに基づいて，変数の値の書換えを用いて計算アルゴリズムを記述する「ノイマン型言語」も，ソフトウェア工学的に見て多くの問題を含むとされていた．これらの問題を解消するための実行モデルや新しい計算機の一連の提案が「非ノイマン・マシン」と総称されていたのである．

データフローモデルやリダクションモデルはその代表であった．また非ノイマン型言語としては関数型言語が主流と考えられていたが，ICOTでは論理型言語を非ノイマン型言語の有望株ととらえていた．

こうして，論理型言語の実行過程を「節定義を用いたゴールの書換え」ととらえて，トップダウンにマシンの実行モデルを構成するリダクション方式およびそれに基づく並列推論マシンの研究が始まった．もう一つの方式として，計算過程をひたすらデータの流れに注目して再構成したデータフローグラフを実行するデータフローマシンの研究も行なわれた．これはマシンの新しい実行モデルからボトムアップに並列推論マシンを構成するアプローチであった．

これらの研究開発にあたって，実行対象となるべき論理型言語を決める必要があったが，当時は論理型言語自体がまだ歴史の浅い研究対象であって，並列実行向きに洗練された論理型言語などは探すべくもなかった．結果としてとりあえず採用されたのがPrologまた

は pure Prolog[†1]であり，OR 並列の実行モデルが主に試された．前期の終わり頃になってようやく，AND 並列 (ストリーム並列) の Concurrent Prolog (後の KL1 に非常に近い言語．3.4節参照) が実験対象に加えられた．

3.3.3 プロジェクト前期の並列推論マシン

プロジェクト前期の並列推論マシンの研究開発は，当時の第一研究室で，NTT から来た村上國男室長のもとで進められた．一方並列論理型言語の研究は，第二研究室の古川室長のもとで始まっていたが，どちらも手探りの状態からのスタートであり，研究活動はこの時点ではほとんど独立に進められた．

前期の並列推論マシンは異なる方式により 3 種試作され，いずれもプロセッサ数 8 から 16 台の規模であった．

リダクションモデルに基づく並列推論マシンの試作は日立が担当した．推論モジュールと構造メモリモジュールをネットワークで接続した構造のハードウェアを設計し，それぞれのモジュール機能は 68000 マイクロコンピュータで実現した．これはターゲットマシンのハードウェアシミュレータという位置づけであった．推論モジュールは，プロセスを格納しリダクションを制御するユニットと，ユニフィケーションを行なうユニットからなり，構造メモリモジュールは構造体データの格納と管理に使われた．

データフローモデルに基づく並列推論マシンは沖電気が担当した．こちらはデータフローグラフを実行するデータフローマシンを論理型言語用に設計し，まったくのゼロから試作した．1983 年から設計を始め 1985 年に完成した．こちらも，要素プロセッサモジュールと構造メモリモジュールがネットワーク接続された構造であったが，要素プロセッサモジュールはいわゆるサーキュラパイプライン型のデータフローマシンの構造をもっていた．

> 「実験機であったので大きさや見てくれに関する制限や要求はなかった．当時における外見上の大きさ最大のデータフロー計算機だったかもしれない．4 筐体で 1.2 トン以上あり，管理部の人が部屋の床が抜けるのではないかと心配しだした．」(六沢談)

前期の並列推論マシンにはもう一つあり，こちらは富士通が担当した．Prolog の並列実行における負荷分散方式が主要な研究テーマであり，株分け方式と呼ぶ新しい方式を提案し試した．ハードウェアはマイクロコンピュータを 2 種類のネットワークで接続したもので，一方は負荷分散用，もう一方は構造体データを格納するメモリとの接続用であった．Prolog の実行木から部分木を切り出して負荷分散対象とする方式で，プロセッサ 16 台程度までのシステムでよい成績を上げた．

3.3.4 前期の並列推論研究のまとめ

前期の並列推論研究について，執筆者の個人的見解によりまとめを行なう．

1985 年まで継続された前期の並列推論マシンの研究開発は，全体として，非ノイマン型の主要な方式で論理型言語を実行すること，そしてそれを行なうマシンを試作することへの，実験的取組みであったといえよう．

そのなかで得られたことをまとめると次のとおりである．

1) 論理型言語の並列実行モデルと，その実現方式に関する基礎技術を習得した．
2) 対象言語が確定しない状況で研究を深めることの難しさを味わうこととなったが，そのなかで GHC が並列論理型言語の本命として浮上してきた (次節参照)．
3) いくつかの負荷分散手法について知見を得たが，また自動負荷分散の難しさについても経験し，並列ソフトウェア研究の重要性を認識した．
4) 構造メモリモジュールを推論モジュールと分けて配置するアーキテクチャは，性能面でかならずしも成功していない．
5) 当時のデータフロー方式は投入したハードウェアに比べると十分な性能が出るとはいえない．

これらの知見は，プロジェクト中期以降の並列推論マシンの研究開発方針に大きく反映されることになった．

†1 Prolog から逐次実行にかかわる機能や assert などの状態を残す機能を取り去ったもの．

3.4 並列論理型言語 GHC の誕生

3.4.1 Prolog から並列論理プログラミング言語へ

Prolog や pure Prolog に基づいてプロジェクト前期の並列推論マシンの研究開発が進められていた頃，第二研究室では古川をはじめとして竹内彰一らのグループが，並列論理型言語の本命を探し出すために奮闘していた．この言語は，並列推論マシンのための中核言語，核言語第 1 版 (KL1) となるべきものであった．

すでに前期 1 年目の間に，Prolog では並列処理の汎用言語とするには無理があり，十分な表現力と並列性と簡潔さを備えた新たな並列論理型言語を必要とするという認識が育っていたように思われる．この考えは，海外の研究者たちとの交流とその人たちの協力により，具体的な研究成果へと結実していった．

> 「KL0 の設計のときは，Prolog という，踏んでもくずれる心配のない，安定した土台があったわけだが，KL1 の場合は，そのような安定した土台の設定から始める必要があった．
>
> 幸いにして，この候補あるいはたたき台となる言語を提案している海外の研究者がいたので，ICOT では 1982 年と 1983 年に彼らを招聘研究員として数週間ずつ招き，密度の濃い共同研究を行なった．イスラエル・ワイツマン研究所の新進気鋭 (文字どおり大変な意気込みで研究してくれた) の Ehud Shapiro と，ロンドン大の Keith Clark，Steve Gregory の各氏である．Shapiro は Concurrent Prolog，Clark と Gregory は PARLOG (parallel programming in logic) という並列論理プログラミング言語を提案していた．」(上田談[†1])

竹内らのグループはこれらの言語を詳細に検討し，とくに Concurrent Prolog については，設計者の Shapiro とともにいろいろな問題のプログラム記述を試みた．そのなかで新しいプログラミングテクニックやスタイルの提案などをいくつも行ないながら，言語の記述力を確かめていった．その結果，1984 年の春には，Concurrent Prolog が Prolog の並列版に比べて以下の点で優れているという結論に達した．

1) 並列実行可能な汎用言語
2) 十分な表現力
   - 多様な並列アルゴリズムの記述に耐える
   - オブジェクト指向プログラミングに適する
3) 論理型言語としての性質のよさ
   - 簡潔さのためにハードウェア化が容易
   - 履歴保存が可能でありバグ取りの助けとなる

この時点から Concurrent Prolog は約 1 年の間，KL1 の基本仕様を与える言語としていろいろな角度から検討が続けられることになった．けれども ICOT のすべての人がこの決断に賛同したわけではなかった．

> 「Concurrent Prolog は，論理プログラミングの特徴とされていて，Prolog の目玉ともなっていた『探索』機能 (たとえばパズルを解くプログラムを書いたときに，すべての可能性を自動的に調べつくしてくれる機能) を失っていたことが，不評の大きな原因となっていた．さらに，言語を使う側からばかりでなく，並列コンピュータアーキテクチャを設計する側の人々からも，効率のよい処理系が作れる見通しが乏しいという批判があがっていた．」(上田談)

上田が竹内らのグループで活動を始めたのは 1983 年であり，Concurrent Prolog が KL1 の候補に浮上するところから，いくつもの批判を浴びせられるまでの経過をつぶさに見，また当事者として悩んでいた．この頃，宮崎敏彦は沖電気にいて言語処理系の開発を受注する立場にあった．

> 「どんな分野でもそうだと思うが，同じ問題に 1 年以上も共同で取り組んでいると，いろいろなことがわかってくるものである．記述能力に対する批判にはプログラミング技法を示すことで，効率のよい処理系が作れないという批判には，とりあえず逐次コンピュータの上で簡単だが効率のよい処理系を作ってみせることで，答えることができた．

†1 本節の上田氏の文章は，「通信工業」Vol.27, No.3 から引用したものである．

そして大局的には，われわれの方向がそれほど無謀なものではなかったという心証が形成されていった.」(上田談)

「当時，Shapiro によって作られた Concurrent Prolog のインタープリタが DEC10-Prolog 上で稼働していた．ただし，Prolog の上で動くインタープリタなので実行速度はそれほど期待できないものだった.

このインタープリタを基に

1) ガード中のゴールを組込み述語に限定し
2) ユニフィケーション部分をコンパイルして (といってもターゲットは Prolog のコード)

高速に実行する処理系が近山によって開発された．その後，上田の手によって，ゴールのスケジューラ部分もコンパイルするよう改良が加えられ，よいコンパイラをもつ Prolog 処理系さえあればかなり高速に Flat Concurrent Prolog が動作するシステムが完成した.」(宮崎談)

3.4.2 GHC の誕生

第五世代コンピュータ国際会議 FGCS'84 に向けて，逐次型推論マシン PSI や SIMPOS のグループがラストスパートをかけていたころ，言語グループは引き続き，KL1 の基本仕様として Concurrent Prolog を用いることの妥当性を検討し続けていた．FGCS'84 が成功のうちに幕を閉じる頃，上田がもっていた一つの着想は具体的な形をなし始めていた.

「Concurrent Prolog のより本格的な処理系を考えるなかで，いくつもの釈然としない点が浮かびあがってきた．そこで，Concurrent Prolog の言語仕様を徹底的に再検討することとした.

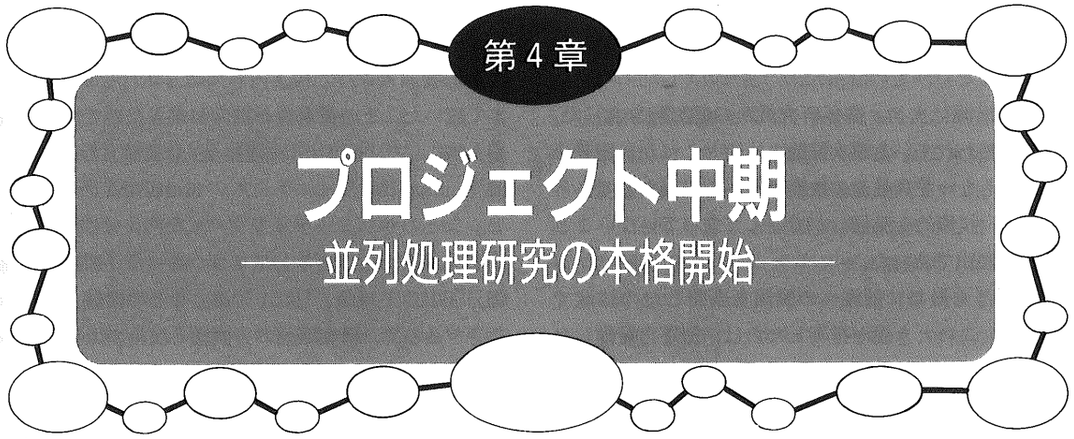
この作業を通じて，並列プログラミング言語としてはすばらしく単純に見えた (当時の) Concurrent Prolog にも，掘り下げて解析するといろいろ厄介な点があることがわかってきた．何かうまい解決策はないかと考えるうち，1984 年末になって，私が Concurrent Prolog よりもさらに単純な言語が作れることをふと見いだし，グループに提案した．幸いグループ内の評判はおおむねよいものであった.

言語の名前は，1 か月ほどのちに，Guarded Horn Clauses (略称 GHC) に決めた．プログラムを，ガードという機構をもった Horn 節 (Horn clauses) の集まりとして書くことからつけたものである.」(上田談)

GHC は言語グループの外でも，簡潔で明解な言語仕様のために評判はよく，これを KL1 の基本仕様として採用することに異議を唱えるものはもう現われなかった．時期は 1985 年の春であり，まさにプロジェクトの前期が終了し，中期が始まろうとするころであった.

GHC の誕生に関して，上田は次のようなコメントを残した.

「GHC は，実用的な見地からは Concurrent Prolog と大きく変わるところはない．つまり，GHC でも Concurrnet Prolog でも，同じ問題をプログラムすれば，たいがいは同じようなプログラムになる．ではどこが新しいかというと，その "同じようなプログラム" で共通に用いている本質的機能 (具体的には，並列に走るプロセスの間の同期をとる機能) を，より単純な規則によって与えたところが新しいのである．この単純化が，"よくわからなかった" 並列論理プログラミング言語をよりよく理解するきっかけとして，また関連する研究を促進するきっかけとして，大いに歓迎されたわけである.」(上田談)



4.1 本格的並列処理研究の始動

プロジェクト前期の終わり，1984 年から 1985 年にかけては，ICOT の並列処理研究にとって苦しい時代であったように思われる．一方で逐次型推論マシン PSI，その OS SIMPOS，システム記述言語 ESP などの研究開発が着々と成果を上げているとき，並列推論マシンや並列論理型言語 (核言語 KL1) の研究開発は，要素技術の研究や言語の提案の段階にあって，まだ 1,000 プロセッサの並列推論マシンへの見通しが得られない段階だったからである．すでに前章で述べたように，核言語 KL1 が決まらなかったこと，それに関連してハードウェアの研究と言語・ソフトウェアの研究が連動できず研究開発が加速しなかったことなどが反省点として浮上していた.

しかし，状況はこのころから大きく展開を始めることになる．いや，当時は大きく展開しているという意識は当事者のなかにはまだなく，苦しいなかで最良の策を一つひとつ探し出しているという思いだったに違いない．一つの動きは，1985 年春に GHC が KL1 の基本仕様としてほぼ認められるようになったこと，プロジェクトの中期に入りいよいよ並列推論に本気で取り組む時期だという意識が高まってきたこと，中期から室長になった内田の提案でマルチ PSI のプロジェクトが動き出したことである.

このマルチ PSI プロジェクトは，前期の並列推論研究の反省に立って企画されたという色彩が強い．つまり，「本格的な並列推論マシンを待っていては核言語 KL1 や並列ソフトウェアの研究がなかなか始まらず，そのために並列推論マシンの研究自体も加速しないという悪循環の懸念が強い．ついては性能や規模の点では理想から遠くても，とにかく早くソフトの研究に使えるマシンを作ってしまおうではないか」という考えに基づいていた．この計画は大成功を収め，マルチ PSI の開発を軸に，KL1 言語の設計，KL1 言語処理系と並列 OS PIMOS の試作といったプロジェクト中期の最大の成果が次々に生み出されることになった．またマルチ PSI の計画が順調に進んだおかげで，後期最終目標となるべき並列推論マシン PIM の研究開発にも落ち着いて取り組むことができた．いや，落ち着いてという表現は適切でなく，PIM の計画が軌道に乗るまでには実はずいぶんの苦労を伴ったのだが，一方でマルチ PSI 計画が進んでいたために，その問題点は表だって取り上げられることは少なかったというべきかもしれない.

以下では，並列処理研究の展開していった経過を，研究開発トピック別にもう少し詳細にたどってみよう.

4.2 戦略的マルチ PSI プロジェクト

4.2.1 マルチ SIM の提案

当時第三研究室室長代理だった内田からマルチ SIM の開発提案があったのは，1984 年 11 月はじめの FGCS'84 が無事終了したすぐあとだった[†1]．PSI を

†1 このころはまだマルチ PSI と呼ばずに Sequential Inference Machine の頭文字をとってマルチ SIM と呼んでいた.

何台も接続することによって，並列ソフトウェアの研究開発環境として早期に利用できる簡便な並列マシンを作ろうというものだった．

この計画は当初，関係研究員から総反対を食った．PSI に使われている逐次推論の技術とこれから開発せねばならない並列推論の技術がずいぶん異なるであろうこと，技術的な見通しがほとんど立っていないことなどが理由であった．

けれども最初に計画への賛同を表明したのは瀧であった．このとき瀧が提案したのは，PSI を接続したマルチ PSI の実行環境を，1 台の PSI の上できわめて忠実にシミュレートできるシミュレータだった．これをシュード (pseudo)・マルチ PSI と名づけた[†1]．マルチ PSI と PSI では，ベースとなるハードが同じであることを利用しており，マルチ PSI ができ上がる前からマルチ PSI 用の言語処理系やソフト開発が始められることを利点としていた．これはハードとソフトの研究開発を協調させながら段階的に拡大してゆくのにきわめて適していた．

シュード・マルチ PSI から始まるマルチ PSI 計画は，そこで開発することになる KL1 や並列ソフトの技術が，最終的に PIM に移行しやすいものになるかどうかの見通しは別として，マルチ PSI ハード，KL1 言語処理系，並列 OS を含めた総合的な開発がスムーズに運びそうな点が受け入れられ，これから数か月かけて計画の肉づけと詳細化が進められた．

4.2.2 マルチ PSI 計画

マルチ PSI 計画では，マルチ PSI のハードウェアを開発し動かすことを計画推進の牽引車にして，それと連動させた言語およびソフトウェアの重要な研究開発計画が併設された．これらの計画を詳細化する作業は，内田，瀧，近山らの新生第四研究室メンバが中心になって案をつくり，古川，竹内，上田らの新第一研究室メンバ (KL1 言語グループ) が加わって技術内容も含めた検討を行なった[†2]．四研からはほかに PSI グループの横田，西川や PIM グループの伊藤徳義が加わり，一研からは宮崎敏彦や田中二郎らの新しく出向してきた研究員たちが参加した．ミーティングは「multiSIM 検討会」と称して 1985 年 3 月から始まり，次第に参加メンバと検討内容の幅を広げながら 10 月まで続いた．その後規模が拡大しすぎたので情報交換を主体にした「multiSIM 連絡会」に衣替えした．この間，「multiSIM ハードウェア」「multiSIM ソフトウェア」などのサブミーティング/サブグループを派生しながら，並行して立ち上がりつつあった「PSI の小型化」「GHC 処理系」「KL1 言語」などの検討グループを巻き込んで，研究開発の方向性と技術の大枠を固めるためのダイナミックな活動を展開した．

1985 年 2 月に草案が作られ夏までに詳細化されたマルチ PSI および関連の計画は次のようなものになった．

マルチ PSI は 2 回作る．マルチ PSI/V1 では前期に開発した PSI-I を数台接続する．KL1 言語処理系の初期試作実験に用いる．1986 年中頃の稼働を目指す．

マルチ PSI/V2 の要素プロセッサ (PE) として使えるように，PSI の小型化・高速化を並行して進める．この小型化 PSI(PSI-II) の CPU を 64 台接続してマルチ PSI/V2 を構成する．この上では，KL1 言語の本格的な並列処理系，並列 OS PIMOS，簡単な応用プログラムを開発する．1988 年の稼働を目指す．また PSI-II は PSI の後継機として量産する．

それぞれのマルチ PSI の開発に先立って，マルチ PSI のシミュレータであるシュード・マルチ PSI/V1，シュード・マルチ PSI/V2 を開発する．それぞれ PSI および PSI-II の上に実現する．これは KL1 言語処理系の開発システムと，OS を含む並列プログラムの開発・実行環境として利用する．

このように盛りだくさんで意欲的な研究開発計画ができ上がったのだが，これらが 1988 年末に一通り動くようになるかもしれないと思っていた者は，当時はごくわずかにすぎなかった．

4.3 マルチ PSI ハードウェアと PSI-II

4.3.1 PSI を小型化する

PSI を小型化，高速化して量産し，ICOT の研究員一人ひとりが自分のマシンをもてるようにしたいというのは，プロジェクト前期からの念願であった．このため，PSI 小型化・マルチ化検討準備会と称するミーティングを 1985 年末から始めた．これには内田をはじめ ICOT の PSI グループと，PSI 開発・製造の主担当となった三菱電機の担当者が参加した．

PSI の研究開発以降，ICOT でのもの作りに関する検討の進め方には一つのスタイルができていた．すなわち定期的なミーティングを設定しておいて，特に宿題を設けなくても次の回までに未検討課題に対する提案をもち寄るというものだった．このやり方は提案競争の面があって，同じテーマを複数の人が違った角度から考えてきたり，人の考えないテーマを探し出したりして，検討会をたいそう活発なものにした．

PSI 小型化ミーティングでも状況は同じであり，参加者は各自の得意とする技術分野とからめて (たとえばマシンアーキテクチャ，ハードウェア実装，マイクロプログラムによる言語実装，開発工程，その他) 自分の描いた PSI-II 像を次々に提案した．当時，提案は数枚の手書き資料がほとんどだったが，それに対して参加者は思い思いの意見や対案をやかましいほどにぶつけた．

念願だった PSI-II の開発を，これから本格化せねばならない並列処理の研究開発とうまくリンクさせたのは，内田の手腕というべきだろう．彼の努力によって，三菱電機は PSI-II の開発と合わせてマルチ PSI の開発も引き受けることになった．これにはメーカが受けやすいように，PSI および PSI-II の商品化の許諾という条件が用意されていた．PSI からマルチ PSI に至る開発・製造の流れは，ICOT のハード開発のなかでは最もスムーズに進んだものといえる．

PSI 小型化検討の一つの集大成は，1985 年 4 月末に三菱電機熱海保養所で開かれた PSI 小型化・マルチ化検討合宿 (2 泊 3 日) で実現した．参加者は内田，瀧，近山，横田，西川，山本の ICOT PSI 関係者オールスターキャストと，三菱電機の研究所からは方式提案の中心を担った中島浩，中島克人，製造部隊から池田，京，そしてマイクロプログラム開発を分担していた沖電気からは細野，三井が参加し，深夜に及ぶ熱い討論を繰り広げた．この合宿で PSI-II の開発に基本的な目処がつき，そのあとは三菱電機による詳細設計・製造へと移行することになった．この前後の模様については次のように述懐されている．

「1983 年に，当時 SRI(Stanford Research Institute) にいた D.H.D. Warren (現在英国ブリストル大学教授) は，WAM (Warren Abstract Machine) という Prolog 用の抽象機械命令セットを提案した．WAM では，コンパイラによる静的解析で大幅な処理の高速化を実現できると期待された．

1984 年末に PSI-II の検討を開始していた三菱電機のアーキテクチャ部隊は，早速 PSI 上で WAM 用にマイクロプログラムの改良版を試作し，評価を行なった．評価の結果，1.5 倍の性能が得られることと，WAM 向けにアーキテクチャを変更し，さらにゲートアレイ LSI の使用などによる実装レベルの改良によりさらに 2 倍，合計で 3 倍の性能が得られる見通しを得た．そして，1985 年 4 月の合宿後から始まったハードウェアの詳細設計，そして引き続く製造は，1986 年のクリスマスに完了し，DEC20 上のシミュレータでデバッグが進められていたマイクロプログラムも 1987 年の夏頃には一通りの完成を見た．」(克人談)

PSI-II はその後，大幅に改良された SIMPOS を搭載して 1987 年末には ICOT 内の研究者に提供され始めた．性能は，後に出たクロック高速化版では 400K LIPS を達成し，ワークステーションとして，またマルチ PSI/V2 のフロントエンドプロセッサとして大活躍した．製造台数は約 300 台に達し三菱電機製が中心であったが，沖電気も製造した．

4.3.2 マルチ PSI/V1

マルチ PSI/V1 の設計は，1985 年 5 月頃から細々と始められた．当時，ICOT 内も三菱電機でも，開発の主力部隊は PSI-II に投入されていて，まだ検討段階の multiSIM にまわせるほどの余裕はなかった．瀧は PSI 小型化・マルチ化合宿で PSI-II 本格開発のキックオフをはたしたあと，内田から許しを得てまだ公式には認められていなかったマルチ PSI/V1 の開発準備にとりかかった．

マルチ PSI/V1 では，大型冷蔵庫ほどもある PSI を入出力装置も含めてまるごと 6 セット用意し，それを専用の接続ネットワークでつなぐことにした．新規に

†1 1984 年 11 月 29 日の最初の提案書では seudo-multiSIM となっている．

†2 中期からは 5 研究室体制となり，PSI，SIMPOS，マルチ PSI，PIM の開発は，第四研究室の内田の下に統合され，言語グループは新第一研究室の古川の下に再編された．

作るのはネットワークボードだけである．言語処理系については，当時第一研究室で GHC 処理系の開発を進めておりこの技術をもらってくることにした．

最大の問題はネットワークの方式であったが，瀧は実現の容易さを重視して 2 次元メッシュの接続トポロジを選び，のちに wormhole routing と呼ばれるようになるメッセージ交換方式を独自に考案した[†1]．考案した方式はよさそうに見えたが，ネットワーク方式に関するサーベイをさぼったので，その新しさ[†2]については意識していなかった．

接続ハードウェアの詳細設計は 9 月までにほぼ完了し，そのあと三菱電機の製造部隊に引き継ぐ作業を行なった．このころから ICOT 側には，富士通から出向してきた木村康則が加わった．また三菱電機側の担当は，前期に PIM グループにいた益田嘉直であった．

ネットワークボードの開発は 1986 年 7 月末に完了し，6 台版マルチ PSI/V1 のハードウェアが完成した．ネットワーク制御マイクロプログラムのテストを経て，9 月末にはフラット GHC の並列処理系が動き出し，初期の並列プログラム実験環境として活躍を始めた．

4.3.3 マルチ PSI/V2

マルチ PSI/V2 検討会は，PSI-II の開発が三菱電機の工場に移り，浩や克人らの三菱の研究所メンバに余裕の出てきた 1986 年 5 月末から始まった．マルチ PSI/V2 では PSI-II の CPU を使うことにしていたが，PSI-II に対してプリント板をどの程度共通にできるか，実装をどうするか，同期クロックの分配をどうするか，立上げ・保守・デバッグのためのコンソール機能をどうするか，などが主要な検討テーマとなった．64 台ものプロセッサを 8 本の大型筐体に実装するシステムは，規模としては大型コンピュータであって，しかも類似の技術が社内にないということで，製造を担当する益田はいたく慎重になっていた．

その後，三菱電機側からの提案を主体にした検討会を続け，8 月頃までには基本的な方針を確定して詳細設計に入り，1987 年 2 月には実装設計が終わり，6 月にはプリント版と LSI の製造を開始した．

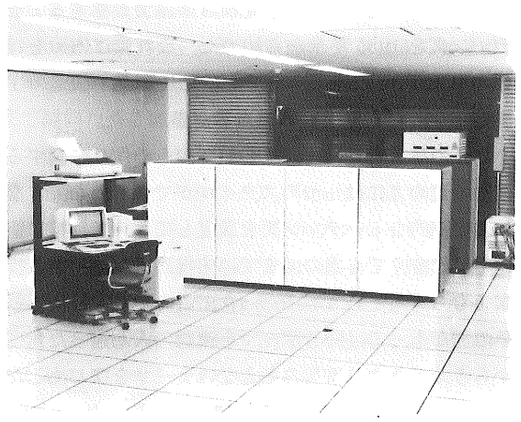


図 4.3.1 ICOT のマシンルームに搬入されたマルチ PSI/V2．筐体 1 本当たり 8 プロセッサ，8 筐体全体で 64 プロセッサ構成のシステム．手前はフロントエンドプロセッサ (FEP) の PSI-II

マルチ PSI/V2 の要素プロセッサはボード 8 枚で構成し，8 プロセッサ分を 1 本の大型筐体に実装することになった．8 枚のプリント板は 6 枚まで PSI-II のプリント板と共通化することができた．これは，ボード供給の安定性やエージング済みのボードを安心して使えるという大きな利点を生み出すことになった．またネットワークは，基本方式についてマルチ PSI/V1 を踏襲しながら約 10 倍の高速化を図った．

64 台ものプロセッサを相手にした保守・立上げ・デバッグ方式は，一つの研究課題だった．メンテナンスパスと呼ぶ専用のインタフェースを設けて，別に用意した PSI-II を接続し，PSI-II の上にコンソール機能をソフトで実現する方針とした．このソフト開発は，富士通 SSL からきた杉野栄二が担当した．ユーザインタフェースを含む高機能なソフトの開発には，ESP のオブジェクト指向機能が役立った．

一方，PIMOS の入出力システムの一部として PSI-II を働かせるために，マルチ PSI/V2 本体のネットワークを延長して PSI-II と接続した．この入出力システムとして働く PSI-II をフロントエンドプロセッサ (FEP) と呼んだ．

マルチ PSI/V2 の最初の 1 筐体は，1987 年のクリスマスプレゼントとして ICOT に搬入され，KL1 処理系マイクロプログラムの実機テストに使用された (図 4.3.1)．その後，KL1 処理系開発，PIMOS 開発，

†1 wormhole routing はカリフォルニア大学の人達が提唱したことになっている．

†2 wormhole routing を実マシンに適用した世界で最初か 2 番目の例であることがあとでわかった．

デモプログラム開発を並行させるという難事業の末，1988 年 11 月の FGCS'88 直前にほぼ安定稼働状態にこぎつけ，64 プロセッサ版，32 プロセッサ版各 1 台を出展することができた．

4.4 苦心と努力の KL1 言語処理系

4.4.1 研究開発経緯

中期の始まった 1985 年 4 月時点では，KL1 言語はまだ検討段階で基本仕様に GHC を用いるという合意がやっと成立した頃だった．GHC および KL1 の検討グループは古川の新第一研究室に属していた．同グループの竹内，上田，宮崎らに第四研究室の近山が加わって，DEC20 や PSI 上での GHC 処理系の試験実装が試みられた．

1986 年に入ってマルチ PSI/V1 の完成見通しが立ち，その上に実装する GHC 並列処理系の準備を急ぐ必要があるために，宮崎が第四研究室に移って作業することになった．このとき宮崎は，移り先が第四研究室であることをひどく心配した．個人研究を重んじるいわゆる研究的雰囲気の一研とは対象的に，当時の四研は ICOT の工場と呼ばれていて，集団作業でバリバリともの作りに励んでいたからである．ほかの研究室から見るといかにもこき使われそうに見えたに違いない．

マルチ PSI/V1 の処理系開発検討会のことを KL1 実装 TG(タスクグループ) と呼んだ．瀧が主催し，参加者としては宮崎，木村に加えて，前期のデータフロー型 PIM の経験をもつ沖電気からきた六沢一昭，三菱電機からは中島克人とソフトウェア担当の江崎令子，元一研の仕事をしていた三菱総研 (MRI) の市吉伸行らが加わった．KL1 実装 TG は，マルチ PSI/V1 の稼働後は KL1 プログラミングに比重を移し，KL1 プログラミング TG (略して KL1-TG) としてソフトウェアのメンバを増員し活動を継続した．

一方マルチ PSI/V2 の処理系については，1986 年 5 月から始まったマルチ PSI/V2 ハードウェアの検討と並行して，不定期のミーティングで基本方針を議論していたが，1986 年 12 月から本格的な処理系ミーティングをスタートすることになった．こちらは性能も重視した本格的な KL1 処理系を作るということで，KL1-TG のメンバをはじめ，PIMOS の改良作業が一段落ついた近山や三菱電機の中島浩らが加わり，15 名前後の大所帯の検討会を重ねた．1987 年 4 月からは，ICOT に出向となった中島克人が瀧からこの検討会の主催を引き継いだ．マルチ PSI/V2 処理方式検討会と改称されたこのミーティングは，おそらく ICOT のなかで最もホットな議論が繰り返されるミーティングとなった．途中でマイクロプログラム開発に専念する V2 ファームウェア検討会をサブミーティングとして分離しながら，マルチ PSI/V2 が稼働を始めたのちも，処理方式の改良・改訂のために存続し続けた．主要メンバは，克人，近山，浩，宮崎，市吉，六沢，瀧のほか，PSI-II のマイクロプログラム開発から移った稲村雄や，木村，杉野，SIMPOS を手掛けていた吉田かおるなども参加した．

もう一つ忘れてならないのが，KL1 言語自体の設計の経緯である．初期の KL1 言語検討は一研で進められたが，マルチ PSI/V2 の開発段階になって四研に中心を移した．言語仕様の提案とまとめを近山が行ない，フラット GHC からの拡張部分のうち，資源管理，実行優先度，処理効率向上のための組込み機能など，処理系のつくりと関係の深い部分についてはマルチ PSI/V2 処理方式検討会で検討を重ねた．機械語 KL1-b の設計と KL1 コンパイラについては，近山と木村が担当した．言語仕様は，1987 年後半には言語処理系作成に必要な部分から確定され，年末にはほぼ完全な仕様が固まった．

以下では，これらの研究開発経緯をマルチ PSI/V1 処理系，V2 処理系に分けて，エピソードを中心にもう少し眺めることにしよう．

4.4.2 GHC 処理系からマルチ PSI/V1 処理系へ

A. 初期の GHC 処理系

GHC 発明当時は，フルセットの (Flat でない) Concurrent Logic Language (GHC, Concurrent Prolog, PARLOG など) が本命と考えられており，宮崎によって，近山の CP 処理系をベースに Deep Guard を許す GHC 処理系が DEC20 上に開発された．ただし，逐次実行向きの実現方式であった．

その後，一研の KL1 の言語仕様および実現方式検討グループでは，Concurrent Prolog や GHC 向きの

Deep Guard 実現方式について検討を重ね

1) Deep Guard の効率よい実現の困難さ
2) 実際の Deep Guard の必要性

とのトレードオフから，検討対象を次第にフラット GHC へと移行していった．

KL1 のベース言語としてフラット GHC が有力となった頃は，ちょうど PSI-I および SIMPOS の第 1 期目の開発が終わり，ユーザが ESP/SIMPOS を本格的に使い始めた頃である．そのころ筆者らは，上田と近山による DEC20 上の Flat GHC 処理系をもっていたが

1) Prolog 上の処理系が，同期を Busy Wait で実現していたため，Suspend が多く発生する大規模なアプリケーションの開発に適していなかった
2) 並列実行のための実現方式を検討し実証するための処理系がほしかった
3) PSI 上の ESP は，DEC20 上の DEC-10 Prolog と同等の性能をもっていた
4) せっかく開発した PSI/ESP/SIMPOS を使いたかった

などの理由から，宮崎を中心に ESP を使った PSI 上の処理系を開発するプロジェクトを開始した．

B. PSI 上の処理系からマルチ PSI/V1 処理系へ

PSI 上に最初に作られたフラット GHC 処理系は，上田&近山処理系の延長線上に位置し，ESP の論理変数と差分リストを活用するものであったが，Busy Wait をなくすように改良されていた．

その後実現されたもう一つの処理系は，ESP のオブジェクト指向機能を使い，フラット GHC の変数セル，構造体，ゴールレコードといった実装上のデータ構造をすべて ESP のオブジェクトとしてヒープ上に実現した処理系である．この処理系は

1) その頃 Prolog の実現方式として主流になりつつあった WAM をベースに，フラット GHC 向き抽象命令を設計し実現する
2) フラット GHC 向きデバッグ環境を考案し，実装する
3) マルチ PSI/V1 用の処理系として，ネットワークをまたがるポインタとその dereference が可能な実装方式を考案し，実現する

ことを主な目的として，宮崎，江崎 (後の佐藤夫人)，市吉らを中心に開発が進められた．

マルチ PSI/V1 の処理系は，三田国際ビル地下の計算機室で初期の PSI を 6 台接続したシステム上で稼働し始めたが，処理系のデバッグは困難をきわめた．この経験から得たことは，非同期な並列システムの場合バグの再現性がなく，緻密な設計と地道なコードの見直しが特に重要だということであった．

何はともあれ，マルチ PSI/V1 上で (もしかすると世界最初の) 分散ユニフィケーションは成功し，goal 実行の依頼 (いわゆる throw goal) も成功した．この処理系の最初のユーザは KL1 実装 TG のメンバであり，沖電気の井上 (後 ICOT の和田) は KL1 で knapsack 問題などを書いて走らせた．

4.4.3 マルチ PSI/V2 処理系の研究開発

先にも述べたとおり，マルチ PSI/V2 処理方式ほどホットなミーティングはほかになかった．検討会の進め方は例によって提案競争であり，そこでは提案のことを「叩き台」と呼んだ．出てきた叩き台を参加者は納得がゆくまで叩きに叩いて，もう叩いても変形することのない確実な方式が採用されていった．この検討会がとりわけホットだった理由は，一つにはアイディアがたくさん出てきてしかも議論好きの人間が，最も多く集まっていたこと，もう一つには，当時の参加者は気づいていなかったかもしれないが，マルチ PSI 関連では最も重要で最もおもしろい検討課題を扱っていたためではないかと思われる．

この検討会で議論したことは，第 II 編第 4 章「KL1 言語処理系の実装」に出てくるすべての技術内容にわたっている．それらの多くは，「もともと何らかの下地があってその改良を検討した」という類ではなく，どう解決してよいかわかっていないほとんどゼロの状態から，一つひとつ検討を重ねたものである．残っている資料をひもとくと，1987 年 4 月から 1988 年 1 月の間に，実に 54 回のミーティングが開かれており，毎回数種の検討資料が提出されている．これはもう，エキサイティングだけれどもきわめてハードな検討会のマラソンとでもいうべきものだった．

一方，検討した技術の方向性が完璧だったかというと，反省がないわけでもない．マルチ PSI/V2 の目標は，十分高い実行性能と，プロセッサ数が 1,000 台レベルに増加しても問題の起こらない並列実行方式 (処理系のアルゴリズム) の実現だったといえる．このために種々の提案された最適化 (になるかもしれない) 方式をできるだけ実装して評価することを試みた．これは裏を返すと，処理方式検討グループの元気がよくて，次々に考え出す最適化方式を処理系に積み込んだために，方式がやや複雑になりすぎたきらいがあるという反省にもなっている．

処理系マイクロプログラムの実機デバッグは，最後になるほど難しさの度合を増した．特に，メモリ管理機構に関しては，「予期せぬアドレスのデータがいつの間にか破壊されている」など，現場をとらえにくいバグとなった．また KL1 プログラムが実際に並列実行されるようになると，「再現性が乏しい」，「バグがデッドロックの形で現われる」など，処理系のデバッグが非常に困難であることもわかった．そのなかには，処理系の複雑さに起因する部分もあったことと思われる．これらを解決するため，デバッグを容易にするためのトレーサ，スパイ，ブレークポイント機能などを種々のレベルで順次追加改良していく必要があった．けれども最終的には，マイクロプログラム開発チームの優秀さと，PSI のころから培った経験により，デバッグ作業は大変な苦労を重ねながらも，目指す FGCS'88 に何とか間に合うことになった．

4.5 はじめての本格的並列オペレーティングシステム PIMOS

4.5.1 イメージ作りと PIMOS 概念仕様書

マルチ PSI の上で並列 OS を作ることは，1985 年に作られたマルチ PSI 計画の草案の頃から盛り込まれており，PIMOS の名前も近山によってすでにその頃にはつけられていたが，長い間実体のないままできた．

1986 年 5 月からマルチ PSI/V2 ハードウェアの検討が始まるにあたり，KL1 処理系とともに OS の話題も不定期の検討会で取り上げるようになった．この頃から，SIMPOS の開発を手掛けたソフトウェアチームのメンバもミーティングに参加するようになった．第一研究室の田中二郎らからは，Concurrent Prolog 上に開発されていた Logix という OS との対比上の議論なども話題として提供されたが，総じて関係者の間での PIMOS のイメージがまったくばらばらであることが問題となった．

PIMOS 開発着手のタイムリミットが近づいていると感じていた瀧は，PSI-II 用の SIMPOS 改良作業からまだ十分手が離せない近山をなかば無理矢理に誘って，PIMOS のイメージ作りと概念仕様書の作成を提案した．こうして PIMOS の検討は，PIMOS タスクグループとして 10 月にスタートした．メンバは瀧，近山，元 SIMPOS グループの佐藤，吉田，三菱電機からは古市昌一，沖電気からは中澤，井上らが加わり，さらに宮崎，上田，田中，杉野，越村三幸らも参加した．

タスクグループの目的は

- PIMOS に必要な機能の洗出しと，PIMOS の守備範囲の明確化
- 各機能のイメージ固めと，可能なものの詳細化
- PIMOS 概念仕様書の作成 (PIMOS の憲法とする)

であった．PIMOS 第 1 版概念仕様書は 1987 年 1 月に完成し，タスクグループは近山の下に PIMOS グループとして再編され，概念仕様書を出発点とした詳細検討が開始された．

4.5.2 UNIX マシン上の開発環境 PDSS

PIMOS の開発にとって頭の痛い問題があった．それはマルチ PSI/V2 用の KL1 処理系が使えるようになるのは 1988 年に入ってからと予想されることだった．つまりそれ以前から開発を進めるには，代わりとなる開発環境がどうしても必要だった．汎用の UNIX マシン上にこのための処理系を作る仕事は，宮崎が買って出た．

PDSS(Pimos Development Support System) の開発は，PIMOS の設計と並行して 1987 年 3 月に，宮崎一人でスタートした．4 月にはその後の主要開発メンバとなる富士通 SSL の平野，中越が加わり，徐々にイメージを固めていった．

その当時は，PIMOS のプロセス管理で使う予定であった「荘園」の機能を提供する KL1 処理系がなく，そもそも KL1 の言語仕様そのものも確定していなかったため，以下の方針で開発を進めることになった．

1) 開発期間半年程度で，スピードと安定性においてそこそこに使えるものを開発する．

2) 言語仕様が固まっていないため，スピードよりも改造の容易さを重視した実現方式をとる．

3) PIMOS 開発者以外にも利用可能なように，ユーザプログラムの開発環境を提供する．

4.5.3 設計・試作・マルチ PSI/V2 への移植

PIMOS は将来の超並列計算機にも対応できるように，情報を極力分散管理して，OS が並列処理のボトルネックになったりしないことを基本設計理念としている．PIMOS の設計は，階層化，分散化された資源管理方式の検討と，それに関連したタスク管理，デバイスハンドラの設計などをかわきりに意欲的な検討が進められた．

PIMOS はすべての部分が，プロセスがメッセージ交換によってのみ処理を進めるモデル (並列オブジェクトモデル) に基づいて設計され，そのとおり KL1 によって忠実に実現されている．OS がもつべき各種の機能をすべてこのモデルに合わせて設計し実現するところに新しさとおもしろさがあった．

1987 年 7 月末には PDSS 0.5 版がリリースされ，8 月にはその上で PIMOS のサンプルコーディングのテストが始まった．この頃には，SIMPOS を担当していた OS 開発部隊のかなりの人数が PIMOS に移り，開発グループの陣容が整ってきた．10 月末に PDSS 上の MicroPIMOS がリリースされ，使い勝手が大きく向上した．PIMOS はその後，設計の詳細化，本格的コーディング，マルチ PSI/V2 と仕様を合わせた KL1 言語への書換えを経て，1988 年 6 月には PDSS 上で一通りの完成を見た．8 月になってシュード・マルチ PSI/V2 への移植とデバッグを開始し，ほどなくマルチ PSI/V2 実機への移行を行なった．

「全体的に PIMOS のデバッグは楽だった，と思う． やはり同期処理などのキーとなる部分を処理系が肩代りしてくれたことがポイントだろうか．PIMOS をマルチ PSI/V2 へ移植したとき，PIMOS のバグはほとんど (まったく?) 出なかったのには感激した.」(和田談)

4.6 ハードウェアのターゲット・並列推論マシン PIM を設計する

4.6.1 当初の中期 PIM 計画

並列推論マシン PIM の開発は，第四研究室長の内田が自ら陣頭指揮をとった．実現のための技術的見通しが不鮮明だったのみならず，1,000 プロセッサ規模の並列推論マシンの試作にメーカの協力が得られる保証がかならずしもなかったためである．

1985 年 5 月に作られた開発方針案では

1) 中期末までに 100 プロセッサ規模の中期 PIM を開発する．KL1 を機械語とする専用プロセッサと高速ネットワークを新規設計するが，これはマルチ PSI/V2 より高い性能を狙う (単体で 100K LIPS 以上)
2) 後期 PIM のプロセッサに発展すべきものとして，要素プロセッサ内で 10 程度の並列性をもつアーキテクチャを研究開発する．前期 PIM のリダクションやデータフロー方式を評価し反映させる．
3) 後期 PIM 向けに要素プロセッサ 1,000 台以上からなるマシンモデル/アーキテクチャを提案する

を課題としており，要素プロセッサの方式がまだ固まっていなかったこと，マルチ PSI/V2 には性能を期待していなかったことがうかがわれる．これらの研究開発メーカには，前期 PIM を担当した 3 社が候補に上げられていた．

計画を実施するにあたって，メーカからの協力を取りつけること，研究開発グループを組織することが大きな問題であった．

4.6.2 中期 PIM の検討開始

内田にとって一つの悩みは，PSI や SIMPOS 開発に腕を奮った研究員たちが，すでにマルチ PSI と関連計画に貼りついていて，PIM グループの指導者になり得る適当な中堅研究員が残っていないことだった．この点については，1985 年 8 月に NTT から後藤厚宏を迎えて第四研究室室長代理に据え，中期 PIM グループのリーダとした．

9 月から ICOT の PIM グループと富士通，日立，沖 3 社の担当者で PIM 方式検討会を開始した．最初

は前期 PIM3 モデルの勉強会を通して共通の問題意識作りから始め，機械語，ハードウェア，ネットワーク，開発ツールなどの検討サブグループを順次スタートさせた．検討の中心は 100 プロセッサ規模の中期 PIM に置いた．並行して走っているマルチ PSI 計画とは，マルチ PSI 連絡会などにより情報交換した．

1986 年 6 月にはマルチ PSI 関係者も交えて集中討議を行ない，共有バス共有メモリによる密結合クラスタを部分構造としてそれをネットワーク接続する中期 PIM の全体イメージが固められた．

その後は共有メモリを用いたクラスタ内処理方式や，共有バスに接続する並列キャッシュ方式などの詳細検討へと進んだ．

4.6.3 純技術的ではない部分の難しさ

PIM の開発には技術とは別の部分の難しさがあり，これはプロジェクト全体の運営ともからんで，もっぱら内田から上のプロジェクト管理者の役回りであった．

一つは，プロセッサ 1,000 台規模の並列推論マシンを試作することに対するメーカの協力を取りつけることであり，もう一つはソフトウェア開発要員をメーカから確保することであった．

これを実現するには，メーカに FGCS 技術およびプロジェクトの将来性と重要性を正しく認識してもらうことが必要であった．このため，内田らによるメーカ担当への働きかけと，渕所長を伴ったメーカ首脳への説得の努力が続けられた．

「このような説得の努力を続け，富士通が最大構成の PIM の試作を決断するまでにおよそ 2 年を要したように思う.」(内田談)

こうして後期への見通しが立ち始めた 1987 年から，中期 PIM の LSI 化のための詳細設計も進むことになった．けれども期待された時期よりは遅れがあり，スケジュール上は厳しいものがあった．

4.6.4 計画の修正

一方で，1988 年に入って当初の期待よりよい成績を出すことがしだいに確実になってきたのがマルチ PSI/V2 であり，単体プロセッサの性能で 130K LIPS，プロセッサ数は最大 64 台で，中期 PIM の目標をほぼ満たすものとなった．これが確実となった夏頃の段階で，中期 PIM の目標は修正され，設計してきたプロセッサ方式は後期の最大構成の PIM 向けにさらに洗練して，設計・試作を継続してゆく方針が打ち出された．

ちょうどこの時期，前述の 3 社に三菱と東芝を加えた 5 社で後期の PIM 開発を分担する方針が固められた．また PIM グループにおいて，クラスタ構成をもつ PIM 用に検討が重ねられていた KL1 処理系は，三菱を除く[†1]4 社の PIM の共通処理系として使われることを前提に，詳細設計と開発ツールを含む本格的開発に移行する方針が固められた．このための処理系開発手順や必要とされるツール類の検討は，1988 年 3 月頃からすでに着手されていた．

4.7 並列ソフトウェアの研究・事はじめ

4.7.1 初期の並列プログラム

1984 年から 1985 年頃は，Concurrent Prolog や GHC によるプログラミングテクニックの研究が，主に第一研究室の竹内，上田らを中心に進められていた．1986 年後半になって，マルチ PSI/V1 の稼働に伴い，その処理系開発を進めていた KL1 実装 TG のメンバらにより，小規模の評価用プログラムが何本も作られることになった．

「マルチ PSI/V1 の印象としては，6 台の PSI のコンソールが別な方向を向いていて，実行状態や出力結果を見るにも (システムを立ち上げる際にも)，6 台の PSI のまわりを走りまわっていた記憶がある.」(和田談)

性能測定や評価を進めるには，性質の異なるさらに多くのプログラムが必要であった．第一研究室の田中が米国出張 (1986 年 11 月) の際に SRI の Fernando Peraira のところに寄り，試してみるとよいといわれてもち帰った問題は大いに参考になった．

1) Maxflow 問題 (Shapiro の示したもの)

†1 この時期，三菱の PIM の方式はまだ決定されていなかったが，最終的にマルチ PSI/V2 の上位互換性を維持するものとなり，処理系技術もマルチ PSI のものを引き継いだ．

2) Overbeek 氏の半群要素計算の問題 (ICLP'86)
3) Heatflow 問題 (Keller らが FP Conf. で扱った)
4) Waltz's Labeling Algorithm
5) Best Path 問題

これらは 1987 年に入ってすべてプログラム化され，特に Best Path 問題は 5 種類もの違うプログラムが作られて，さながらプログラムコンテストのようであった．なかでも市吉のプログラムは並列オブジェクトモデルに基づくものであり，最短経路問題として長く改良が続けられる元となった．ほかにも旧二研で作られた構文解析プログラム BUP のフラット GHC 版 (井上)，項書換えシステム TRS(高木茂行)，Life Game(杉野) などがあり，作って動かすと KL1-TG にもち寄り皆で議論することを繰り返した．

この頃から，ある程度本格的なプログラムになるとその問題領域の知識がなければプログラムが作れないという，いってみれば当たり前のことが問題となり始め，たとえば BUP のようにほかの研究室に教えてもらいながらのプログラム開発も行なわれた．

実際に並列実行させてわかったことは，簡単な全解探索の問題でさえ負荷分散はやさしくはないことだったが，同時にそれを改善するためのプログラム改良作業がおもしろいことも経験した．さらに研究を進めるには，実用的な性能をもつとともに十分使いやすい言語処理系が必要とされた．

4.7.2 マルチ PSI/V2 上の KL1 プログラミング

1987 年 12 月になって，マルチ PSI/V2 ハードウェア 1 号機の ICOT 納入が近づき，KL1 処理系マイクロプログラムの開発もたけなわのころ，瀧は翌年秋の FGCS'88 にマルチ PSI/V2 を出す場合のデモンストレーションのことを考え始めていた．もっともこれは，KL1 処理系や PIMOS の開発がすべて順調に進んだ場合にだけ可能になることではあったのだが．

KL1-TG で 12 月にブレーンストーミングを行なって 20 ほどの候補をあげ，2 月はじめまでに 10 に絞り込んだ．

テーマ選択にあたっては，適当な大きさの問題，見栄え，本格的応用もできるだけ含めること，問題もプログラムの性質も多様であることなどを考慮した．こ

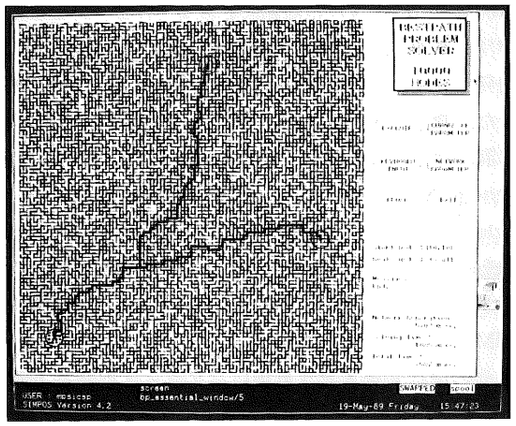


図 4.7.1 最短経路問題の表示画面．点の数 1 万個のグラフ上での，左下開始点から指定した終了点 (2 か所) までの最短経路探索結果を表示している

のときの合言葉は，たとえトイ・プロブレムであっても，トイ・プログラムになってしまうものは出さないことにしようというものだった．この裏を返すと，それまで ICOT に限らず，並列プログラムの例題として出てくるのはトイ・プログラムがほとんどだったからである．

こうして検討に着手し，最後まで生き残った問題は次の四つであった．

1) ペントミノ (詰込みパズル)
2) 最短経路問題 (グラフの点の数 1 万個以上)
3) 自然言語構文解析 (プログラム名 PAX)
4) 詰め碁

ペントミノと最短経路問題は第 III 編で取り上げるので詳細は省略するが，前者はパズルの全解探索問題において動的負荷分散を試みたところが売りものであり，後者は並列オブジェクトモデルに基づくプログラムであって，1 万個以上のオブジェクトが分散アルゴリズムに従って並列に動く (図 4.7.1) ところが新しかった．これらは問題は単純だけれども，プログラムは決して玩具ではないところを目指した．三菱電機の古市昌一，沖電気の和田久美子らが担当した．

自然言語構文解析は，KL1-TG のメンバだけでは歯が立たないので，自然言語グループとジョイントチームを組むことにした．基本となる方式は一研の松本裕治が開発した AX と呼ぶボトムアップ・チャートパーザの一種で，逐次プログラムがすでに動いていた．これの並列化方式と負荷分散に取り組んだ．プログラム構造は，同じく松本の提案によるレイヤード・ストリーム法をとったが，大域的な通信が発生することが予想されていて，負荷分散が課題だった．結果は思わしくないものとなったが，その反省は第 III 編 3.7 節の構文解析に生かされた．

詰め碁ではコンピュータ囲碁のグループとジョイントチームを組んだ．これはゲーム木の探索問題であり，アルファ・ベータ枝刈り法により探索空間を小さくすることが課題であった．問題となったのは逐次性の高いアルファ・ベータ枝刈り法を並列化することで，無駄な見込み計算 (第 III 編 2.4 節参照) を減らすのに KL1 の実行優先度指定の機能を活用した．枝刈りに用いる囲碁の知識があたらない場合ほど，並列処理によるスピードアップが大きいというようなおもしろい結果が，後の解析で明らかとなった．詰め語に関しては初段なみの実力があると評価された．

4.8 死ぬ思いでマルチ PSI/V2 を動かした FGCS'88

4.8.1 専務からの呼出し

1988 年はじめ頃だったか，瀧は廣重専務理事から呼出しを受けた．「秋に予定されている第五世代コンピュータ国際会議 FGCS'88 にはデモンストレーションを予定してるようだが，どの程度金をかけて取り組むべきものか．マルチ PSI/V2 の会場への移設費用はかなり高くつくようだ」という問いかけであった．瀧はここで大見栄を切ってしまった．「今度のデモンズトレーションは，ICOT の並列推論マシンの夢が現実になり始めたことを大衆に知らしめるきわめて重要なものです．最大限の規模で取り組むべきものです．」

けれどもそれは，まったく新しいマシンの上で，新しい言語処理系と，新しい並列 OS と，新しい応用プログラムをほぼ同時に立ち上げるという，誰も経験したことのない難事業に突入することを意味していた．

4.8.2 逆算線表

KL1 処理系マイクロプログラムの開発も，PIMOS の開発も，FGCS'88 を終点としてそこまでに完成させるにはいつまでに何を動かさねばならないかという「逆算線表 (工程表)」を作っていた．この種の計画は概して遅れるものである．遅れると，計画では別の版を何回かに分けてリリースするはずだったものを一つの版にまとめたうえリリース時期を少し遅らせるように変更して線表を引き直した．こうすると当然ながら後ろにしわよせがいく．この線表の引き直しが何回も何回も繰り返され，関係者の間には，しだいに焦りの表情が見えた．

1988 年 7 月以降の開発経過を振り返ってみよう．ここにあげたスケジュールは，何度も遅らせてもうこれ以上は延ばせないというぎりぎりのところで，KL1 処理系と PIMOS それぞれの担当者が必死でキープした結果とお考えいただきたい．

- **KL1 処理系マイクロプログラム関係**
  - **7/18** シュード・マルチ PSI/V2 (GC なし，1PE 動作のみ) 初版リリース
  - **9/18** マルチ PSI/V2 (GC なし，複数 PE 動作) リリース
  - **10/17 頃** マルチ PSI/V2 (GC つき，複数 PE 動作) リリース
  - **11/28** FGCS'88 本番

- **PIMOS 関係**
  - **7/下旬** シュード・マルチ PSI/V2 (1PE 動作のみ) 上で PIMOS デバッグ開始
  - **9/末** 1PE 動作の PIMOS 基本機能デバッグ完了
  - **10/末頃** 複数 PE での PIMOS 基本機能安定動作をほぼ達成
  - **11/28** FGCS'88 本番

マイクロプログラム関係はリリースと書いてあるので一人前の完成品を出しているように見えるが，じつはある部分の機能までしか動かない半完成品を使ってもらって，少しでもマイクロのバグを出そう (出してもらおう) ということを繰り返した．実際，PIMOS のデバッグ開始直後は，マイクロのバグにあたることのほうが圧倒的に多く，マイクロデバッグにつきあっているような状況であった．

これはやむを得ないところがあって，第五世代言語で並列プログラムを書いているのが PIMOS で，第三

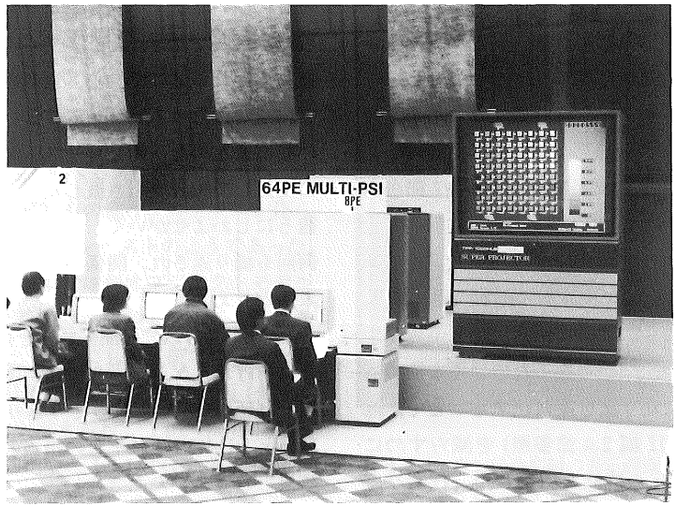


図 4.8.1 FGCS'88 におけるマルチ PSI/V2 のデモ風景．1988 年 11 月 28 日，東京プリンスホテルにて

世代以前の言語で並列プログラムを書いているのがマイクロプログラムによる KL1 処理系だともいえるからである．実際に，実機で出た PIMOS のバグは驚くほど少なく，KL1 の威力を思い知らされた感があった．

4.8.3 OS なしの応用プログラムデバッグ

このように KL1 処理系と PIMOS が追いつ追われつ改版を重ねていた頃，デモプログラムの実機デバッグも実は進んでいたのである．

以下は，デモプログラムの開発スケジュールであるが，とくに開発期間の短さに注目されたい．

- **デモプログラム関係**

  7/中旬 最短経路問題：設計中，ペントミノ：検討中，PAX：未着手，詰め碁：一部コーディング中

  8/中旬 パフォーマンスメータ：設計開始，PAX：検討開始

  9/下旬 最短経路問題および詰め碁：実機デバッグ開始，PAX：コーディング中

  10/中旬 最短経路問題，詰め碁，パフォーマンスメータ：稼働

  10/下旬 PAX 稼働

  11/中旬 ペントミノ稼働[†1]，デモプログラムのヒートラン開始，

  デモシナリオ作成開始

  11/下旬 デモ練習

  11/28 FGCS'88 本番

普通は OS が動く前にデモプログラムのデバッグをしようとしても無理であるが，マルチ PSI のデモプログラムには秘策があった．これには PIMOS がすべてオブジェクト間のメッセージ交換で動作しており，FEP との通信も例外ではないことを利用している．マシンの立上げの際に PIMOS のかわりにデモプログラムを挿入しておくと，デモプログラムから直接に OS なしで FEP の I/O デバイスオブジェクトと通信できるという裏技があったのである．これは，OS も FEP もブートストラップ機能もすべて自前で開発したことの強みであったのだが，とにかく OS の稼働とほとんど時期を同じくしてデモプログラムも動くという綱渡りをやってのけたのであった．

こうして半年前には誰も成功を予想できなかったこと，すなわち KL1 言語処理系と PIMOS とデモプログラムの並行開発はみごとに成功し，FGCS'88 を飾ることができたのであった．

†1 このペントミノはニューロ版で，全解探索版はもっと早く稼働．

第 5 章 プロジェクト後期 ——並列処理研究、発展の時代——

5.1 プロジェクト後期について

FGCS'88 を駆け抜けた研究者たちの幾人かには，「これで最終目標の並列推論マシンが見えた」という思いが確かにあった．けれども現実に動いているものといえば，64 プロセッサのマルチ PSI/V2 であり，できたばかりの PIMOS であり，たった 4 種類のデモ用並列プログラムにすぎなかった．これらを満足のいく並列推論マシンとその上のソフトウェアシステムに育てていくには，やらねばならないことがそれこそ山のようにあった．

ターゲットがはっきりしてきたあと，しっかりした計画を立てそれを着実に攻め落とすという作業は，じつは見ための何倍も難しいものである．ましてや，相手にしている技術がハードもソフトも含めて，すでに手本のない前人未踏の領域にさしかかっているとなればなおさらである．これを進めるためにプロジェクト後期にすべきことは

1) プロセッサ数千台規模の並列推論マシンハードウェアの設計を完結し試作すること
2) その上で効率よく動く KL1 言語処理系を開発すること
3) それらを多くの人に使ってもらえるよう PIMOS を改良・整備すること
4) KL1 の利用者を拡大すること
5) 本格的な応用プログラムを開発し，第五世代コンピュータの有効性を示すこと
6) 広く海外とも交流をもち，成果の宣伝・普及に努めること

などである．

これらのいずれもが，難しさとともにたいへんな人手と努力を必要とするものであった．以下ではこれらの主要なテーマについて，後期における研究開発の経過とエピソードを紹介し，第五世代コンピュータの並列処理技術がしだいに枝葉を広げ，しっかりした木に成長していく様子を描写しようと思う．

5.2 並列推論マシン PIM・複数モデルを試作する

5.2.1 ICOT とメーカの役割分担

並列推論マシン PIM の開発は，最終的に 5 社が担当することが中期の終わりに決まっていた．開発の苦労は，ICOT 側と各メーカ側のそれぞれにたくさんあった．

研究開発の進め方は，ごく大まかにいって次のようであった．ICOT の役割は，PIM のシステム構成とマシンアーキテクチャに対していろいろ注文をつけ，メーカの提案にコメントすること，それから PIM 共通の KL1 処理系である VPIM を設計し，各 PIM (PIM/m を除く) のために供給することであった．KL1 が正しく動けば，PIMOS や応用プログラムの移植は楽なはずだ．

メーカの役割は，PIM のシステム構成やアーキテクチャを提案し，ICOT のコメントを受けて詳細設計したものを製造すること，さらに ICOT から供給される VPIM をそれぞれの PIM (PIM/m を除く) 用に

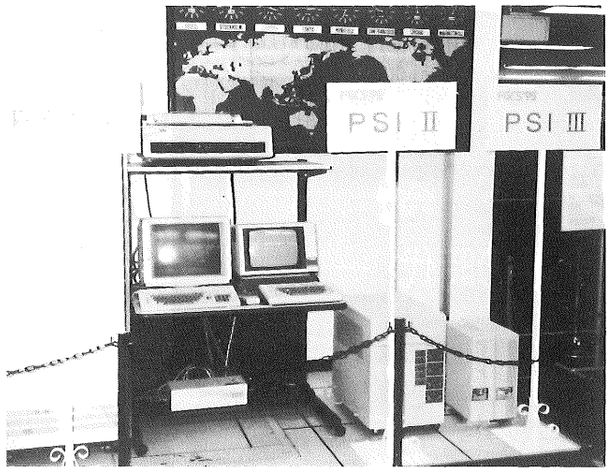


図 5.2.1 PSI-I(左), PSI-II(中), PSI-III(右) の勢ぞろい. FGCS'92 で ICOT のマシンルームに展示された

修正・移植し KL1 が動くようにすること，それが終わると PIMOS，応用プログラムを載せ，システムの評価を行なうことであった．

5.2.2 PIM モデルそれぞれの取組み

各メーカごとの PIM 開発の取組み方は少しずつ違っていた．富士通の担当した PIM/p は，中期 PIM(100 プロセッサ構成) として検討を進めていたものの継続で，システム構成やマシンアーキテクチャに関して，ICOT と富士通がかなり細部まで相談して決めた．また KL1 処理系の VPIM については，富士通 SSL から 10 人を越えるメンバが ICOT に常駐となり，ICOT といっしょになって開発を進めた．VPIM を PIM/p 用に修正・移植する作業と，PIMOS および応用プログラムの移植は ICOT 内で行なった．

日立の担当した PIM/c，沖電気の PIM/i，東芝の PIM/k については，システム構成やマシンアーキテクチャに関して，メーカの提案に ICOT がコメントすることを繰り返してしだいに設計を詳細化していった．VPIM の修正・移植やソフトウエアの移植はメーカで行なった．

三菱の担当した PIM/m は，開発方法もシステム構成もマルチ PSI/V2 を踏襲することになった．すなわち要素プロセッサの LSI を PSI-III(図 5.2.1) として先行開発し，これを専用ネットワークで接続して PIM/m を構成した．マシン命令レベルでマルチ PSI/V2 と互換性をもたせたため，KL1 処理系の開発にマルチ PSI での設計データを多用できたほか，ソフトウェアはマルチ PSI からオブジェクトコードごともってくることができた．

PIM/p，PIM/c，PIM/i については，1988 年 4 月頃からメーカ別のハードウェア検討会を始めており，中期 PIM の検討を反映して同年にはすでに部分試作のための詳細設計を進めていた．PIM/k は 1988 年 12 月から検討会を開始し，翌年春に設計に着手した．PIM/m に関しては，1989 年 4 月にプロセッサ LSI の設計から始め，年末にネットワークの設計に入った．

5.2.3 仮想ハードウェア上の KL1 処理系 VPIM

プロセッサアーキテクチャが大きく異なる 4 種の PIM に対して，すべてに共通に使える処理系を作るのは予想以上にたいへんなことだった．この開発は，後藤厚宏を中心に ICOT の PIM グループで進められた．

まずシンプルな仮想ハードウェアを定め，このうえで KL1 処理系のアルゴリズムを設計する．これを VPIM と名づけた．それを記述するのに PSL と呼ぶ記述言語を新たに設計した．VPIM の仕様は大きくなることが予想されたため，正しさを検証することが問題となった．そこで PSL で記述した VPIM を C 言語に変換し，そのまま汎用計算機で実行できる仕掛けを作った．これは PIM のシミュレータの一種であり，PIM/s と名づけた．PIM/s では，KL1 プログラムを仮想ハードウェアの機械語にコンパイルしたコードが実行される．

PIM/s で試した VPIM の仕様は，仮想ハードウェア上では正しく動くことが保証される．次にしなければならなのは，PSL による VPIM 記述をそれぞれの PIM モデルのネイティブコードに変換することである．これには自動変換ツールを作ること，手作業で書き直すこと，それらの両方などにより，PIM の各モデルに対応して行なった．この変換時に入ったバグの検出には，PIM 実機のシミュレータが必要であった．

PSL で階層記述された VPIM と周辺のツール類を合わせると，予想を越えたたいへんな開発量となった

でき上がった VPIM を各メーカに技術移転するときは，対応するサービス窓口の仕事もたいそう忙しかった．

PSL の検討は 1988 年 7 月から始まり，VPIM 0.1 版 (サンプルコーディング) が 1989 年 4 月に完了した．その後改版を重ね，各社への正式版のリリースが 1991 年春から始まった．

5.3 並列ソフトウェア研究の拡大

5.3.1 並列応用開発を立ち上げる

FGCS'88 が大成功のうちに終了し，プロジェクト後期の仕事を考えていた瀧は，自分が最も力を入れるべきは並列応用の開発を立ち上げることだろうと思うようになった．「ほうっておくと，並列処理に興味のある計算機アーキテクチャ屋やシステムソフトウェアの人たちを除くと，一番ユーザになってほしい応用問題をもっている人たちは KL1 を使ってくれないのではないか」という危惧があった．これは，1986 年から 87 年の KL1-TG や，1988 年のデモプログラム開発の経験から感じていたことだった．

知識処理をはじめとする応用問題をもっている人は，その応用がうまくプログラム化でき速く走りさえすればいいのであって，べつに KL1 で書くことや並列処理をやりたいわけではない．そういうわけのわからないものは，できれば避けたいのが本音である．そんな場合，効果が明瞭でない限りは，KL1 で書いたり苦労して並列実行はしてくれないものである．そして効果は明瞭になっていないのが普通である．

これでは，誰かに応用プログラム開発を頼るなどというのは無理な話である．そうすると，最終的な並列推論マシンの評価作業もままならない．瀧は，デモプログラムの PAX と詰め碁の開発で試みたジョイントチーム方式をこれからの並列応用開発に全面的に取り入れることを考えた．

アイディアは簡単である．開発すべき並列応用のテーマを決めると，マシンや OS を作ってきた並列処理の専門家と，応用領域の専門家を合わせてジョイント開発チームを作るのである．これを応用テーマごとに立ち上げる．そこで出てきたアイディアや工夫は KL1-TG などを活用して相互流通させる．この過程で，並列ソフトウェアの基本的な研究テーマである負荷分散や並列アルゴリズムなどについて議論し研究を深める．このあたりの方法論のまとめは，第 III 編 2.1 節で紹介する．アイディアは簡単であったが，これを実施するにはたいへんな努力とサービス精神を必要とした．

5.3.2 並列 LSI CAD を例として

この方法論にはテストケースが必要である．成功例を見せないと誰もついてきてはくれない．以前から LSI CAD をやってみたいと考えていた瀧は，思い切って自分で並列 LSI CAD ソフトの開発チームを率いることを決心した．瀧は CAD の専門家ではない．ICOT には以前から，知識処理研究の一貫として LSI CAD を取り上げているグループはあったが，CAD そのものの専門家というわけではなかった．

応用開発チームを作るのに，LSI CAD そのものの勉強から始めるという，最も手間のかかる方法をとることになってしまった．あとで聞くと，このようなアプローチを冷やかに見ていた人もいたようだ．

「LSI CAD といえば，従来の計算機でそれなりに成功を収め，たいへんなノウハウの蓄積している分野である．そこに，まったく新しい計算機と新しい言語をひっさげて，素人が乗り込んだところで勝ち目がない．もっと PIM や KL1 が得意とする分野でたたかうべきだ」確かにそういう考え方もある．けれども PIM や KL1 が得意とする分野はすでに自明なのだろうか．瀧は，「LSI CAD は大量計算を必要とする問題を多く含んでいて並列処理に適当だし，だいいち結果の良し悪しがはっきり出るのがよい．目いっぱい頑張って，それで従来の計算機に勝てないようなら，PIM はたかだかその程度のマシンだ」ということまで考えていた．

ICOT は勉強の場をアレンジするにはこのうえないところである．瀧は 1988 年から，並列 LSI CAD のワーキンググループ (WG) を設立した．WG とは，メーカと大学から専門家に出てもらって定期的な会議を開き，講演や勉強会，調査研究，特定テーマでの技術討論，ICOT の研究紹介とそれに対するアドバイスを受けること，などを目的とした委員会の一種である．協力メーカから CAD のプロにきてもらうことに成功し，こちらからは並列処理のことを伝え，むこうからは CAD 技術，ないしはその勉強のしかたを教えてもらった．CAD についてのちょっとした疑問があったとき，質問できる人と顔つなぎができたことが一番ありがたかった．

WG のつてを頼りに，さしあたり目標にしようと考えていた LSI のレイアウト問題の勉強にメーカの研究

所を訪ね，ついでに並列推論マシンの売込みをやったりもした．これはあとになって，メーカが LSI CAD のプロを出向者として出してくれるという思いがけない効果を生んだ．

1989 年度に入って，WG では基礎的な勉強のための講演が一順し，そろそろ ICOT 側からの研究提案もほしい時期になっていた．このころ考えついたのが，並列オブジェクトモデルに基づく配線方式であり，基本となるアイディアを WG で紹介した．

この年の 1 月には，後に並列シミュレーテッド・アニーリングの新アルゴリズム (第 III 編 3.4 節参照) を発明する木村宏一を迎え，春になって後にオブジェクト指向配線プログラム (第 III 編 3.2 節参照) を担当する大嶽能久，タイムワープ論理シミュレーション (第 III 編 3.3 節参照) の松本幸則を加えて，並列 LSI CAD グループは体裁を整えてきた．

オブジェクト指向配線プログラムは 1989 年秋から本格的な開発に着手し，木村の並列シミュレーテッド・アニーリングは 1989 年暮れから新年にかけて考案され，松本のタイムワープ論理シミュレーションは 1990 年 2 月頃から検討が始まった．配線とシミュレーションは 6 月頃には初版が動き始めた．

その後，これらのプログラムは大幅に性能が改善され，並列処理効率も高まり，1991 年度に入ると並列応用のデモプログラムとして ICOT の看板の一角を担うようになった．また木村の並列シミュレーテッド・アニーリングに基づく LSI セル配置プログラムも，大嶽のあとを引き継いだ伊達博，JIPDEC の星らの努力により夏頃には動き始めた．

並列 LSI CAD の研究開発が，何も下地のないところから開発チームを起こして短期間のうちに成果をあげ始めたことは，他の並列応用開発に大きな刺激を与えたとともに，開発方法論として一つの手本を示すことにもなった．

その後並列推論マシン PIM への移行，配置配線一貫処理プログラムの開発，タイムワープ配線方式の考案など，CAD グループは少人数ながら活発な活動を 1992 年まで続けた．特に第五世代コンピュータ国際会議 FGCS'92 においては，LSI CAD は ICOT の代表的な並列応用の一つとして，論文発表やデモンストレーションで重要な役割をはたすことができた．また瀧が一つの目標としていた，逐次型汎用計算機上の LSI CAD と比較して優位性を示すことについても，まだ不十分ながらも一応の結果を示すことができた (第 III 編 3.2 節参照)．

5.3.3 KL1-TG 最盛期

1989 年は，FGCS'88 のデモプログラムを駆け込みで作り上げた研究員たちにとっては楽しい時代だった．KL1，PIMOS，マルチ PSI/V2 に最も習熟していた彼らにとって，プログラムを改良し並列処理性能を高め，性能値を計測し解析する作業は，仕事よりは楽しみそのものであった．ペントミノ，最短経路問題，PAX，詰め碁のプログラム改良と性能解析がいっせいに進められ，新しい結果が出るたびに KL1-TG にもち寄っては議論を繰り返した．

多階層動的負荷分散の考案や，最短経路問題に実行優先度制御をもち込んで見込み計算 (第 III 編 2.4 節参照) を抑制する試みはこの頃に行なわれた．FGCS'88 当時，40 秒以上かかっていたテトロミノ (ペントミノの一つ小さい問題) がプログラム改良で 10 秒足らずに実行時間が短縮され喜んだが，解析するとスピードアップの値はかえって低下していた．64 プロセッサで 50 倍のスピードアップが目標だったから，これを実現するのにいろいろ知恵を絞った．2 階層動的負荷分散にやっとたどりつき，スピードアップの目標を達成するとともに実行時間も 5 秒まで短縮した．

このようなデータの行き交う KL1-TG は，最先端のプログラム改良作業を続ける研究員たちの情報交換の場であり，また新しい応用開発に参入する人たちへの技術移転の場でもあった．

5.3.4 本格的並列応用開発の拡大

1989 年 8 月，第四研究室長の内田は知識処理応用・並列応用の研究開発体制を整備するため，電総研から新田克巳を迎え入れた．翌年の研究所機構改革のための準備でもあった．

新田は研究を続けてきた事例ベース推論とその応用システムを，ICOT の代表的並列応用に加えるため研究活動を開始した．一方内田は，並列処理と知識ベースの新しい応用として注目の度合が高まりつつあった遺伝子情報処理に取り組むべく準備を始めた．遺伝子情報処理は海外でも脚光を浴びており，国際協力や共

同研究のテーマとしての可能性も考慮しての着手だったようである．遺伝子情報処理の研究は，WG を設立して生物学者との交流を深め情報を収集するところから始まった．WG の運営は吉田が内田や新田の助言を受けて行なった．

1990 年 4 月になって，研究所の全面的な組織替えが行なわれ，研究部の下に 7 研究室を置く新体制となった．内田は研究部長に就任し，新田は第七研究室長，瀧[†1]と市吉は同室長代理となった．第七研究室は知識処理応用・並列応用全般を担当する重要組織としてスタートすることになった．

七研には大きく分けて次のような研究開発グループを置いた．

1) 法的推論
2) 遺伝子情報処理
3) LSI CAD
4) 囲碁
5) 並列アルゴリズム

法的推論は事例ベース推論の応用 (第 III 編 3.5 節参照) であり，遺伝子情報処理 (第 III 編 3.4 節参照) と合わせて新田が指揮した．囲碁は，計算機と人で囲碁対局するシステムの並列版の開発であり，CAD とともに瀧が取りまとめた．並列アルゴリズムのグループだけは基礎研究よりの仕事で，並列アルゴリズムと処理効率の解析を行ない，他の開発グループに結果を反映させることを目指して市吉が担当した．これらの研究開発グループは ICOT の正規メンバで 15 名，最盛期ではさらに同数程度のソフトウェア外注を抱えていた．他に，メーカへ研究の再依託を出しているテーマが多数あり，それらの研究指導も行なった．

法的推論，遺伝子，CAD の研究開発は，苦労を重ねながらもおおむね順調に推移し，1991 年 6 月の段階で，法的推論システム HELIC-II の刑法を扱う基本システムや，タンパクのアミノ酸配列解析に関するマルチプルアライメントプログラムなどが稼働した．これらは FGCS'92 までにさらに改良や方式の追加を重ね，論文発表やデモプログラムとして大活躍をした．技術的な詳細については第 III 編を参照されたい．

囲碁対局プログラムは，プロジェクト中期から CGS(Computer Go System) として電総研の実近氏の指導の下に研究が進められたが，後期に入ると KL1 を用いてその並列版も開発するという方針で瀧が加わり，未来技術研究所の沖，七研の清らにより開発が進められた．最終的に，逐次版でコンピュータ囲碁の世界大会で準優勝程度までいったが，残念ながら並列処理で棋力を大幅向上させるには至らなかった．最大の理由は，チェスと違って囲碁を強くする知識処理の方法論がまだ確立されておらず，単なる探索問題として扱っては時間がかかりすぎて解けないこと，そしてそのような状況下ではたとえ並列で 100 倍高速化できてもまだ不足で，新しいヒューリスティクスを見つけるほうがもっと計算時間を短縮するのに役立つといったことが普通に起こるためである．これは未成熟な知識処理領域で並列処理を試みることの難しさを物語る象徴的な例となっている．

ICOT で開発されたその他の並列ソフトウェアとして，最終的に大きな成果をあげたものは次のとおりである．

システムソフトウェアの一つである並列データベース管理システム Kappa-P(第 III 編 3.8 節参照) は，横田第三研究室長の下で河村元夫らにより開発された．中期に開発された Kappa-II を並列化する方向で後期になってから開発が始められ，FGCS'92 までに基本機能が稼働した．

もう一つは，第五研究室の長谷川隆三室長の下で，藤田正幸室長代理，越村らにより開発された定理証明系 MGTP(第 III 編 3.6 節参照) がある．このシステムの誕生にはとびきりのエピソードがあるので紹介しよう．

5.3.5 渕コーディング

1989 年も半ばをすぎると，FGCS'92 のために突貫工事で作った KL1 処理系や PIMOS もずいぶん落ち着いてきたので，渕所長が「自分の席のわきに PSI-II を置いて，シュード・マルチ PSI で KL1 プログラムを動かしてみようかと思うが」という話になった．忙しい所長だから，動いているプログラムを触っていただくだけでも結構な話だということで，さっそくマシンを設置した．所長は，暇を見つけては読みにくいマニュアルの類にずいぶん目を通したらしい．

所長は電総研時代から定理証明系には造詣があり，古川研究所次長や長谷川らとともに，並列推論マシンの応用の一つとしてぜひ定理証明系の研究を盛り立てるべきだと考えていたようである．定理証明系という

†1 第一研究室長を兼務

のは一種の汎用推論エンジンであり，Inference よりはもう少し高度な推論を意味する Reasoning のための技術である．一方 KL1 は，GHC をベースに置いたときから Reasoning の意味での推論言語よりはもっと機能的に低い位置にある基本言語を目指してきた．そのため Reasoning を研究課題とする人たちからは，「KL1 は推論言語といっても，われわれのシステムを書くのにはC言語で書くのとたいしてかわらない」といった失望ないし叱責の声が寄せられていた．所長にしてみれば，「ほんとうにそうなら少し残念なことだ」と思ったに違いない．所長はどう思ったのか，長谷川のところから，当時効率のよいことで注目されていた定理証明系 SATCHMO の論文を取り寄せて読み始めた．

正月が明けた頃，所長からメッセージが届いた．「ちょっとおもしろいプログラムができたんだけど見てくれないか．」これが後々まで語り種となる「渕コーディング」であった．KL1 のどちらかというと低レベルの言語機能を巧みに利用しながら，SATCHIMO が行なっている前向き推論をじつに巧妙に，かつ効率よく実現してしまったのである．この結果は，5月に開催した第一回 KL1 プログラミングワークショップで所長自ら発表し，大きな反響を呼んだ．何せ研究所の最も忙しいトップが，研究員を追い抜いて立派な研究成果を出してしまったのだから．この「大事件」に触発されて ICOT における定理証明系の研究開発は俄然勢いづき，渕コーディングに基づく新しい定理証明系を次々に作り出した．MGTP がそれであり，最終的には並列推論マシン PIM の最大手ユーザの一つに名を連ねるようになった．

5.3.6 KL1 講習会/ワークショップ

KL1 講習会

1989 年，KL1 利用者の拡大に向けて瀧と近山が相談して KL1 講習会を企画した．その頃 KL1 プログラミングのノウハウというと，PIMOS グループと KL1-TG のメンバがもっているだけで，ESP や Prolog しか使っていなかったメーカにとっては，KL1 言語のマニュアルだけわたされてもプログラムが書けるわけはなかった．状況は ICOT のなかでもじつは同じで，研究員の7割がたはまだ KL1 を使い込んでいなかった．そこで後期の間に KL1 プログラミングを浸透させるには「啓蒙活動」を頑張るしかなかったのである．

講習会の準備はテキスト作りから始まった．KL1-TG と PIMOS のメンバから最初の講師ができそうな人たちを選んで，講習項目を割り当て，テキスト書きから始めてもらった．マニュアルではなくテキストなので，何度も集まって題材と表現方法を議論した．KL1 講習会入門編は2月に越村と古市で，初級編は同じメンバで3月に，そして中級編は宮崎と古市で6月に開催した．かなりきつい作業であったが，170 ページを越える立派なテキストができ上がった．その後，好評につき講師の代役も立てながら，受講者数百人規模の講習会を何度も開いた．

KL1 プログラミングワークショップ

講習会は好評ではあったが，最新のプログラミングテクニックやノウハウ，並列実行の成功例などは伝わらない．KL1-TG だけでは，多くのメーカの研究者までカバーして情報交換の場を提供することは無理である．そこで瀧は，ワークショップを開催してそのような情報交換の場とすることを考えた．

初回は約4か月の準備の後，1990 年5月に開催した．ICOT 内外で進められている各種の KL1 プログラム開発の成果および中間結果，プログラミングテクニックの紹介など，基礎ないし共通技術的なものから応用プログラムまで，23 件の発表と討論をまる2日間かけて行なった．このときはまだ，KL1 を使い込んだ人とそうでない人の落差が大きく，非常に基本的な質問や疑問も多く寄せられた．

第2回は 1991 年に開催されたが，このときはさすがに参加者の熟練度は向上していて，十分に深い技術討論が行なわれ，FGCS'92 に向けての KL1 プログラム開発の盛り上がりを予感させた．

5.4 PIMOS の改良

5.4.1 0.7 版から 3.0 版へ

FGCS'88 でマルチ PSI/V2 に載せられていた PIMOS は，じつは 0.7 版と呼んでいた暫定版だった．これにはセルフコンパイラはついておらず，プログラムはすべてクロスシステムでコンパイルしたあと実機にロードするという手順を踏む必要があった．ほかにもプログラミングやデバッグ機能が足りなかった．

このように不足していた機能を，とりあえず急いで

一通り用意したのが PIMOS 1.0 版であり，1989 年7月にリリースされた．以下，PIMOS の着実な改良・改版は，第四研究室の近山 (1990 年4月からは第二研究室長) の指揮の下で PIMOS グループにより進められた．1989 年からはマルチ PSI/V2 を追加製造して，ソフトウェアの再依託研究を受けているメーカに貸し出し，研究開発に使ってもらうことになっていた．これに合わせて内田は PIMOS ユーザ連絡会を組織し，PIMOS のリリースに合わせて連絡会を招集した．

PIMOS 1.5 版は，トレーサ，リスナなどをはじめとして使い勝手を向上させるための機能追加や改良を行ない，1989 年 12 月にリリースされた．

PIMOS 2.0 版の目玉はリモート FEP 機能であった．これはマルチ PSI/V2 の脇にいかなくても，ネットワーク接続されている PSI-II が手元にあれば，そのマシン上でマルチ PSI/V2 に対する入出力がこれまでとまったく変わらずに行なえるというものだった．これはネットワーク接続さえあれば離れていても，たとえアメリカからでも使えることを意味した．実際に海外との共同研究のからみでそのようなことが検討され，1990 年6月には米国アルゴンヌ国立研究所との間で専用回線が開設された．PIMOS 2.0 版は 1990 年7月に外部リリースされた．この頃には PIMOS の機能も拡充され，関連メーカのユーザの数も増えてきた．

PIMOS 2.5 版ではマルチユーザに対応したシステムの分割管理[†1]，プログラムの実行状態を可視化するツールなどがリリースされ，並列ソフトウェアの研究開発ツールとして現場で役立つ機能が充実してきた．リリースは 1991 年1月であった．

そして PIMOS 3.0 版は，いよいよ並列推論マシン PIM に対応する版となるが，PIM では FEP の接続方式が変更になっていることもあり，PIMOS 本体の大改訂と改良を行なうことになった．少数精鋭での大作業となったが，1991 年 10 月にはリリースの運びとなった．

5.4.2 PIMOS 3.0 版を PIM で動かす

日立から出向していた屋代寛にとって，PIMOS 3.0 版の開発は最も大変で思い出深いものとなった (以下，屋代の手記より)．

PIMOS 1.0 版は，とりあえず動くものを，PIMOS 1.5 版～PIMOS 2.5 版はリモート機能などの使い勝手向上を実現してきたが，次の理由により大幅な見直しを迫られ，PIMOS 3.0 版では次の変更を行なった．

- **PIM/x[†2] への対応**

  FEP インタフェースが，マルチ PSI で使っていた KL1 ネットワークから SCSI になるという大幅な変更があり，対応が必要だった．

- **サーバ-クライアント方式の導入 & 標準入出力のサポート**

  UNIX マシンを使っての入出力を標準化するのに伴い，PIMOS の核部分である資源木の全面書き換えを行ない，PIMOS の資源管理部分の構成を大幅に変更した．

上述の変更を速やかに，かつ，確実に行なうため，PIMOS に複数の版が同時に存在することとなった．これには，FEP simulator，SCSI ドライバなどの部品をいくつか作り，それをすげ替えながら，徐々に開発を進めていった．また，ここでは VM (仮想マシン) の概念を導入し，PIMOS 上で次版の PIMOS 開発を容易に可能とする枠組みを作った．

このような新規開発と同時期に，IJCAI'91[†3] をはじめとする実機を用いたデモンストレーションがいくつもあり，開発デバッグ作業の緊張感に拍車をかけた．ちょうど PIM のなかで最初に動くことになる PIM/m が ICOT に搬入され，うまくいけばそれがデモに使えるというきわどいスケジュール下での開発が続いた (下記，スケジュール表参照)．

1991.3 Pseudo PIM/m (KL1-net 版) 上で PIMOS 動作
1991.5 Pseudo PIM/m (SCSI 版) 上で PIMOS 動作
1991.6 第5世代コンピュータシンポジウムで初のリモートによるデモ
1991.8 PIM/m 実機上で PIMOS 動作
1991.8 IJCAI'91 で PIM/m リモートによるデモ

†1 時分割によるマルチユーザではなく，並列推論マシンのプロセッサを複数グループに分割してそれぞれを別のユーザに使わせる空間分割を行なっている．

†2 PIM には，PIM/p，PIM/m，PIM/c，PIM/i，PIM/k の五つのモデルがあった．

†3 AI 関係の代表的な国際会議の一つ．このときはオーストラリアで開かれたが，事前に ICOT へのデモ出展依頼が寄せられていた．

1991.10 PIMOS 3.0 版リリース

上述のような開発・サポートを少人数で行なったため，各開発人員，特に資源木・FEP 開発担当者には，かなり負担があった (屋代はこのため過労から肺炎で倒れ入院することになった)．けれども逆にいえば少数精鋭の部隊であったがゆえに，OS 核部分の全面改訂などという強引とも思われるようなことが可能だったのではないかと思う．

5.4.3 負荷分散手法のライブラリ化

マルチ PSI および PIM の上ではさまざまな応用プログラムが開発されてきたが，初期の頃は応用プログラムごとに個々の負荷分散手法を検討せねばならず，本来の問題を解く部分のプログラム作成に専念できなかった．また，同じようなプログラムを別々の人が開発したり，デバッグに要する時間もたいへんだった．

そこで，典型的なタイプの問題に関しては，いくつかの負荷分散手法をライブラリとして提供することによって，プログラマが容易に並列プログラム記述に使えることを目指した．これは当初 KL1-TG で議論したいろいろな負荷分散手法から，サポートの可能なものを三菱電機の古市らがライブラリ化し，PIMOS に含めて提供するようにしたものである．

1992 年現在提供しているのは，MLB(マルチレベル動的負荷バランサ: Multi-level load balancer) と STB(スタック分割動的負荷バランサ: Stack-splitting load balancer) の 2 種類であるが，今後も増える予定である．

さらに，負荷分散手法のライブラリ化のみならず，基本的な並列アルゴリズムを集めたライブラリなども今後提供される予定である．

5.5 華々しい国際交流

国際交流の裏方のはなし

プロジェクト中期から活発化した 2 国間シンポジウムと，後期から始まった海外との共同研究 (2.5 節参照) にはいろいろな思い出がある．

FGCS'88 のあと，マルチ PSI/V2 が動くようになってからの 2 国間シンポジウムでは，それが日本で開催される場合には必ずマルチ PSI/V2 のデモがついてまわった．

1989 年以降，来訪する人は必ずデモを見たがるようになったので (一説によると ICOT が見せたがったという話もあるが)，一時期はプログラムの開発者がデモに時間をとられて仕事ができなくなることもあった．そこでデモのビデオを作ったりもしたが，大事なお客にはやはり生で上演した．この頻度がばかにならず，KL1-TG と PIMOS のメンバを中心にデモ当番を決めて対応していた．このように，ICOT が有名になって研究以外の雑用が増えることを「有名税」と呼んでいた．

2 国間シンポジウムなどの大きな行事のときはデモも特別仕立てとなり，しかも英語で説明するので，デモ当番はメンバ総出で対応する必要があった．最盛期には，スウェーデン，フランス，アメリカ，イギリスなどの国々と，それぞれ別々の 2 国間シンポジウム/ワークショップをもっていたので，対応する側はかなりの負担を感じていた．別々に開かずに一本化すればいいのに，というのは素人考えで，いずれも 2 国の政府間で取り決めた重要な先端技術交流であり，一本化するというような外国に対して「失礼な」ことはできなかったようである．

海外との共同研究では，ICOT の並列推論技術を使ってもらうことにかなりの重点が置かれていたので，マルチ PSI/V2 をはるばるアメリカまでもっていったり，PSI-II を相手側に設置して通信衛星や海底ケーブル経由で ICOT のマシンを動かしたりした．これには海外出張組と留守番組の両方が綿密な計画のうえで連係作業する必要があり，負担に感じるなどというような程度を越えて，毎回が大プロジェクトとなった．

5.6 PIM 稼働す

5.6.1 PIM/m のデビュー

PIM として最初に完成したのは PIM/m であった．一時は LSI の歩留まりが上がらなかったり，ネットワーク LSI の不具合がずいぶんあとになってから見つかり心配させられた時期もあったが，プロセッサ設計開始からちょうど 2 年後の 1991 年 4 月に，最初の 1 筐体が ICOT に搬入された．

ただちに KL1 処理系マイクロプログラムの実機テストが始まり，テストは順調に進んだ．5 月からはシュード PIM/m (PSI-III 上の PIM/m シミュレータ) による PIMOS のテストも始まった．

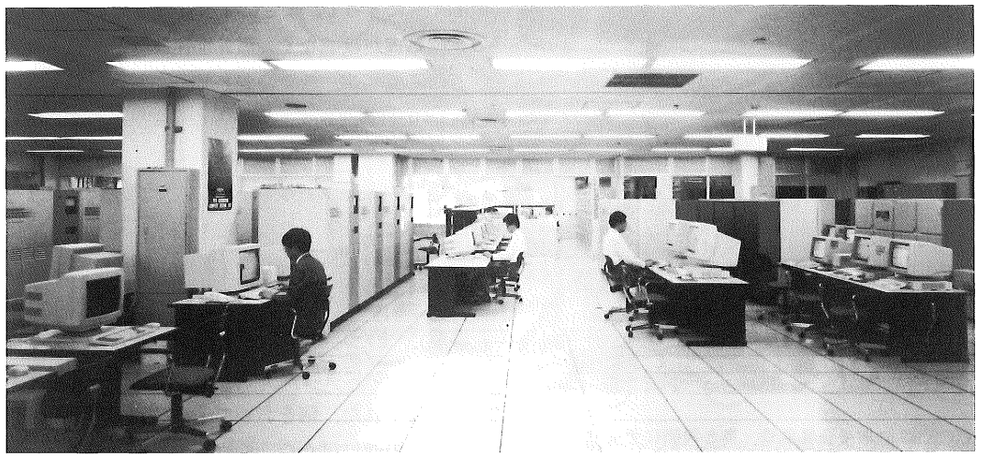


図 5.6.1 ICOT のマシンルーム (1992 年 4 月当時)．左手に PIM/p，右手に PIM/m が見える

7 月に入ると PIMOS がほぼ稼働し，小規模のデモプログラムも動くようになった．PIM で導入された SCSI インタフェースによる FEP 接続部分の不安定さだけが一つ残された不安要因であったが，マイクロプログラムの出現しにくいバグのたぐいも順調に取れていった．この時点で，8 月までに安定稼働できそうな見通しが得られた．

その後 SCSI インタフェースも落ち着き，8 月の IJCAI'91 のデモでは，オーストラリアからの遠隔使用により PIM/m を初披露することに成功した．関係者の苦労を反映しない言葉かもしれないが，納入から 4 か月，じつに鮮やかなデビューといってもいいだろう．

その後マイクロプログラムの高速化の改良を続け，11 月からは FGCS'92 向けの筐体が続々と入荷し始めた．マルチ PSI との互換性により，すぐに並列応用プログラムの開発者にも使われるようになった．1992 年 3 月には，ついに 8 筐体 256 プロセッサの最大構成のシステムが組み上がり，FGCS を待つばかりとなった (口絵 2 参照)．

5.6.2 VPIM の移植とハードウェアデバッグ

1991 年 2 月に VPIM1.0 がリリースされてから，各 PIM 対応の VPIM の修正と移植作業が本格的となった．

PIM/p の場合は，春から試験用の試作機を ICOT に搬入し，富士通社内と並行してハードウェア試験を始めた．試験・保守用のサービスプロセッサ機能をさらに整備しなければならない段階だった．一方 PIM/p 用 KL1 処理系については，VPIM からの移植作業が徐々に進み，実機シミュレータで評価用プログラムを動かすとともに，8 月頃には実機の 1 プロセッサで append プログラムが苦心の末動き始めた．ハードウェアはまだバグを抱えており，コンパイラのコード生成でこれを避けるようにしていた．

その後 10 月頃には，1 プロセッサで queen 問題や最短経路問題が動作するようになったが，マルチプロセッサではまだ安定しなかった．年末から 1992 年はじめにかけて到着した新しいハードウェアではバグが大幅に収束し，複数クラスタのテストが進むようになった．2 月になると，複数クラスタで多くのテストプログラムが実行可能になり，SCSI 部分のテスト終了を待っていよいよ PIMOS の移植にかかることになった．それと並行して，不安定なプログラムの原因究明を続けたが，ハードウェアで致命的な欠陥が見つかるともう FGCS までに作り直しがきかない時期に入り，開発チーム一同緊張の日々が続いた．

PIM/c，PIM/i，PIM/k の場合は試験をメーカ側で進めていた．ハードウェアテストの進み具合などで多少の差異はあったが，処理系の移植に関してはいず

図 5.7.1 FGCS'92 のデモ会場風景．奥と両側にデモブースが並んでいる

れも苦労を重ねながら 3 月を迎えた．

5.6.3 PIM の稼働

3 月に入って PIM/p のデバッグは，PIMOS，KL1 処理系ともに順調に進み，また FGCS 用のハードウェアも続々と到着した．ついに 4 月上旬になって，PIMOS のブートストラップ動作が完動し，4 月下旬には PIMOS の上で規模の大きな応用プログラムが動き始めたのであった．世界ではじめて，マルチクラスタ構成で 256 台のプロセッサを接続した大規模並列推論マシンが稼働した感動の一瞬であった (口絵 1 参照)．

PIM/p はその後，FGCS が終ってから追加製造のプロセッサを加え，最大構成の 512 プロセッサシステムに拡張された．

PIM/c の場合は，3 月に FGCS 用の筐体が ICOT に搬入され，そこから日立の開発部隊を ICOT に移して PIMOS の稼働に向けての処理系デバッグ作業を加速した．PIM/i と PIM/k は 5 月の ICOT 搬入直前までメーカでデバッグを継続した．そしていずれも 5 月に入って，PIMOS および FGCS のデモ用プログラムを稼働させるに至った．じつにきわどい開発スケジュールではあった．

5.7 FGCS'92

1992 年 6 月 1 日から 5 日間，FGCS'92 は，筆者ら第五世代コンピュータプロジェクトを駆け抜けてきた者たちにとって，まさに総決算の一大イベントだった．並列処理だけでなく，第五世代コンピュータ技術のあらゆる成果が，論文発表として，デモ展示として，さらに ICOT オープンソフトウェア[†1]として，国際会議 FGCS'92 の下に結集されていた．

会場の東京プリンスホテルと ICOT の間は専用回線で結ばれ，すべてのデモンストレーションでは，会場のデモブースから ICOT のマシンルームに設置された PIM やマルチ PSI/V2 を遠隔使用した (図 5.7.1 参照)．また発表のなかで，PIM からの出力画面を大スクリーンに実時間で映し出し，説明に使用する試みも行なわれた．何をおいても，やはり PIM は FGCS の一つの顔にほかならなかった．

20 件のデモはすべて並列処理を用いたものであり，第 III 編 3 章で扱った七つの応用プログラムも全部含まれていた．なかでも計算量が大きく多数のプロセッサを必要とする LSI 配線，論理シミュレーション，遺伝子情報処理，定理証明系 MGTP は，256 プロセッサの PIM/m を使って実行され，並列処理の効果を見せつけていた．また，これらの一部は PIM/p でも実行された．非常に充実したデモに対して，海外からも高い評価の声が寄せられた．

それだけのものを準備するのに，関係者の努力はたいへんなものであったが，それでも FGCS'88 のときと比べると，プログラムそのものの開発，デモの演出や練習，説明者の態度など，あらゆる面で何かしらの余裕が感じられた．

この余裕はどこからきたのかと考えてみるに，マシンの進歩でありソフトウェア技術の進歩であり，そこで働いてきた人たちの成長であり，そのようなものによっているのは間違いない．しかし，4 年前，FGCS'88 のときに KL1 処理系，PIMOS，デモプログラムの並行開発をかろうじて成功させたときのあの並列処理と比べるならば，いま FGCS'92 のそれは，比べものにならないほどの広がりと技術としての深まり，そして何よりも大きなうねりとなって新しい時代に向けて動き出すエネルギーを秘めていると感じずにはいられない．

†1 ICOT の中間成果としてのソフトウェアを広く無償公開するルールが，このとき始めて実現した．

第 6 章 むすび

1982 年 6 月 1 日に ICOT 研究所がスタートした．並列推論技術への，そして並列推論マシンへの大きな夢をのせて動き出した第五世代コンピュータプロジェクトは，10 年の歳月を経て夢の多くを現実に変えながら，1992 年 6 月 1 日から 5 日間の最後の第五世代コンピュータ国際会議 FGCS'92 における研究成果報告を盛会のうちにはたし，さらに技術改良を続けたのち，1993 年 3 月 31 日にその幕を閉じた．

思えばプロジェクトが開始された時点では，本当にただの夢にすぎなかったプロセッサ 1,000 台規模の並列推論マシンが，10 年を経て目の前に存在し現実に並列推論を実行しているさまは，長いながいプロジェクトを走り続けた研究者たちにとって，まさに夢のような現実だったともいえる．本編をご覧になって，夢を現実に変えていった数々の営みと，それらを結んだ大きな流れを読み取っていただくことができただろうか．

歴史編に登場したプロジェクトの歴史のひとこまひとこまは，渕所長の言葉を借りるならば，やはり「それぞれの第五」であったろう．KL1 言語，PIM，KL1 言語処理系，PIMOS，並列応用プログラムのいずれをとっても，中心として働いた人がいるにせよ，生み出された技術はいく人もの研究者の合作というべきものである．ただ文章にしたときには，「それぞれの第五」の一つひとつを書き分けることも「それぞれの第五」どうしのぶつかり合いや葛藤を表現することも十分にはできなかったように思う．これは筆者の表現力の未熟さによるものとして，どうかお許しいただきたい．本編で引用しきれなかった多くの人たちの「それぞれの第五」があったことを特に記しておきたい．

第五世代コンピュータの技術は，論理プログラミングに基づく知識処理と並列処理の技術であることはすでに本編の冒頭で述べたとおりである．これを凝縮した表現が「並列推論」であった．それではこの並列推論技術が，プロジェクトを通してどのレベルまできたかについて，ごく私的にコメントしておこう．

推論に必要な並列処理の技術は，本編でたっぷり扱ったように，KL1 言語，PIM，KL1 言語処理系，PIMOS，並列応用プログラムの技術として，また一揃いのまとまりのある技術として，どこに出しても恥ずかしくない世界最先端レベルに達したと考えられる．これがどのような特徴を有し，具体的にどんな姿をしていて，またどのようにおもしろいかということは，続く第 II，第 III 編のメインテーマである．ここに含まれる KL1 言語まわりの技術は，マシンに直結した低レベルの推論技術といえる．

それでは知識処理に直結する機能的に高レベルの推論技術はどうかというと，こちらはより難しい技術であり，プロジェクトのなかでも基礎研究寄りの仕事が続けられ，いくつかの成果が得られていた．それがプロジェクト後期に入って大きく前進し，第 III 編で扱う法的推論や定理証明系 MGTP として具体的な姿を現わし始めている．これからが楽しみな技術といえるであろう．

気になる ICOT 研究所のその後であるが，後継プロジェクトである「第五世代コンピュータの研究基盤化プロジェクト (1993 年から 2 年間の予定)」の推進母体として現在も存続している．

第 I 編　参考文献

[1] エドワード・ファイゲンバウム，パメラ・マコーダック著，木村繁訳：第五世代コンピュータ 日本の挑戦，TBS ブリタニカ (1983)

[2] 渕 一博, 廣瀬 健：第 5 世代コンピュータの計画, 海鳴社 (1984)

[3] 廣瀬 健, 渕 一博：第 5 世代コンピュータの文化, 海鳴社 (1984)

[4] 渕 一博, 赤木 昭夫：第 5 世代コンピュータを創る — 渕 一博に聞く —, 日本放送出版協会 (1984)

[5] 上前 淳一郎：ジャパニーズ・ドリーム — 未知の森へ第五世代コンピュータ, 講談社 (1985)

[6] 松尾 博志：スーパー頭脳集団 電総研, コンピュータ・エージ社 (1987)

[7] 上前 淳一郎：めざすは新世代コンピュータ, 角川文庫 (1988)

[8] 今岡 和彦：我が志の第五世代コンピュータ — 渕一博と ICOT の技術戦士たち, TBS ブリタニカ (1989)

[9] *Proc. FGCS'84 : International Conference on Fifth Generation Computer Systems,* Tokyo (November 1984), also reprinted and published by Ohmsha

[10] *Proc. FGCS'88 : International Conference on Fifth Generation Computer Systems,* Tokyo (November 1988), also reprinted and published by Ohmsha

[11] *Proc. FGCS'92 : International Conference on Fifth Generation Computer Systems,* Tokyo (June 1992), also reprinted and published by Ohmsha

[12] Kurozumi, T. : Overview of the Ten Years of the FGCS Project, *Proc. FGCS'92 : International Conference on Fifth Generation Computer Systems,* pp.9–19, Tokyo (June 1992)

[13] Furukawa, K. : Summary of Basic Research Activities of the FGCS Project, *Proc. FGCS'92 : International Conference on Fifth Generation Computer Systems,* pp.20–32 (1992)

[14] Uchida, S. : Summary of the Parallel Inference Machine and its Basic Software, *Proc. FGCS'92 : International Conference on Fifth Generation Computer Systems,* pp.33–49 (1992)

第 II 編

技術編

第五世代コンピュータの並列処理技術