

JAERI-Review

JP0450291

2004-003



共用炉物理コードシステムの基本概念の検討

-共用炉物理コードシステム(CCS)の構築WPの活動報告-

2004年3月

炉物理研究委員会

日本原子力研究所

Japan Atomic Energy Research Institute

本レポートは、日本原子力研究所が不定期に公刊している研究報告書です。

入手の問い合わせは、日本原子力研究所研究情報部研究情報課(〒319-1195 茨城県
那珂郡東海村)あて、お申し越し下さい。なお、このほかに財団法人原子力弘済会資料
センター(〒319-1195 茨城県那珂郡東海村日本原子力研究所内)で複写による実費
頒布を行っております。

This report is issued irregularly.

Inquiries about availability of the reports should be addressed to Research
Information Division, Department of Intellectual Resources, Japan Atomic Energy
Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 〒319-1195, Japan.

© Japan Atomic Energy Research Institute, 2004

編集兼発行 日本原子力研究所

共用炉物理コードシステムの基本概念の検討
—共用炉物理コードシステム(CCS)の構築WPの活動報告—

日本原子力研究所
炉物理研究委員会*

(2004年1月22日受理)

本報告書は、炉物理研究委員会の下に、平成13、14年度の2ヶ年間設置された共用炉物理コードシステムの構築WP（ワーキング・パーティー）の活動内容を取り纏めたものである。共用炉物理コードシステムの構築WPでは、我が国の炉物理研究者がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築することを目的に、2カ年の期間で4回の会合を持ち、炉物理コードシステムの現状や先端計算機技術などの調査を行うとともに、コードシステムの基本概念を検討した。

日本原子力研究所：〒277-0842 千葉県柏市末広町14-1

※「共用炉物理コードシステムの構築WP」：松村哲夫（電中研）、小林 啓祐（京都大学名誉教授）、別所 泰典（NUPEC）、池田 一三（三菱重工）、飯島 進、森 貴正（原研）、山岡 光明（東芝）、横山 賢治（サイクル機構）、石井 一弥（日立）、内藤 哲孝（NAIS）、日山 伸行（原燃工）、山本 徹（NUPEC）、岩崎 智彦（東北大）、中島 鐵雄（NUPEC）、杉 暉夫（原研）、中川正幸（原電）

Basic Concept of Common Reactor Physics Code Systems

- Final Report of Working Party on Common Reactor Physics Code Systems (CCS) -

Research Committee on Reactor Physics*

Japan Atomic Energy Research Institute

Suehiro-cho, Kashiwa-shi, Chiba-ken

(Received January 22, 2004)

A working party was organized for two years (2001-2002) on common reactor physics code systems under the Research Committee on Reactor Physics of JAERI. This final report is compilation of activity of the working party on common reactor physics code systems during two years. Objectives of the working party is to clarify basic concept of common reactor physics code systems to improve convenience of reactor physics code systems for reactor physics researchers in Japan on their various field of research and development activities. We have held four meetings during 2 years, investigated status of reactor physics code systems and innovative software technologies, and discussed basic concept of common reactor physics code systems.

Keywords: Activity Report, Working Party, Reactor Physics, Common Code System, Innovative Software Technology

* Working Party on Common Reactor Physics Code System

MATSUMURA Tetsuo (CRIEPI), KOBAYASHI Keisuke (Kyoto Univ.), BESSHO Hironori (NUPEC), IKEDA Kazumi(MHI), IJIMA Susumu, MORI Takamasa (JAERI), YAMAOKA Mitsuaki (Toshiba), YOKOYAMA Kenji (JNC), ISHII Kazuya (Hitachi), NAITO Yasutaka (NAIS), HIYAMA Nobuyuki (NFI), YAMAMOTO Tooru (NUPEC), IWASAKI Tomohiko (Tohoku Univ.), NAKAJIMA Tetsuo (NUPEC), SUGI Teruo (JAERI), NAKAGAWA Masayuki (JAPC)

目 次

1. 緒言	1
2. 活動報告	2
2.1 諸機関の炉物理コードシステムの調査	2
2.2 先端計算機技術の活用	6
2.2.1 既存の解析コードシステムにおける例	6
2.2.2 解析ストリーム構築のために利用可能な技術	9
2.2.3 フランスにおける次世代シミュレーション開発計画	14
2.3 統合型炉物理コードの調査	16
2.3.1 JOINT	16
2.3.2 VSOP	18
2.3.3 ORIGEN2	24
2.3.4 JACS	30
2.3.5 統合型炉物理コードのまとめ	32
2.4 共用炉物理コードシステムへの要求事項	33
3. 共用炉物理コードシステムの全体概念	36
3.1 ストリームプログラムの構造	36
3.1.1 スクリプト言語の利用	36
3.1.2 Fortran90/95 の利用	37
3.1.3 ストリームプログラム構造のまとめ	38
3.1.4 既存コードの連携のイメージ	39
3.2 開発すべきソルバー	39
3.3 インターフェース及び全体構成	40
3.3.1 XML の利用案	41
3.3.2 機能毎のモジュール群化案	42
3.3.3 最小データ・インターフェース案	46
4. 結言	49
謝辞	50
 付録	
1-1 サイクル機構のコードシステムの現状	51
1-2-1 原研のコードシステムの現状	59
1-2-2 原研（高速炉）のコードシステムの現状	66
1-3 原燃工のコードシステムの現状	67
1-4 電中研のコードシステムの現状	69
1-5-1 安全解析所のコードシステムの現状	72

1・5・2 原子力安全解析所のコードシステムの現状（説明用OHP資料）	73
1・6 東芝（高速炉）のコードシステムの現状	80
1・7・1 三菱（軽水炉）のコードシステムの現状	81
1・7・2 三菱（高速炉）のコードシステムの現状	82
1・8 日立のコードシステムの現状	83
2・1 共用炉物理プログラム基本システムの提案（小林）	85
2・2 小林案に対する検討結果（横山）	96
2・3 コマンド型のシステム案（松村）	108
3・1 第1回会合議事録（平成13年11月26日）	123
3・2 第2回会合議事録（平成14年2月22日）	128
3・3 第3回会合議事録（平成14年7月19日）	133
3・4 第4回会合議事録（平成14年12月6日）	137
4・1 原子力学会2003年春の年会予稿	142
4・2 原子力学会2003年春の年会発表	143

Contents

1.	Introduction.....	1
2.	Investigation of Reactor Physics Code Systems.....	2
2.1	Reactor Physics Code Systems of Each Organization.....	2
2.2	Innovative Software Technologies.....	6
2.2.1	Examples of Application for Reactor Physics Code Systems	6
2.2.2	Available Technologies for Constructing Stream Program.....	9
2.2.3	Plan of Next Generation Simulation Development in France.....	14
2.3	Integrated Reactor Physics Codes / Code Systems.....	16
2.3.1	JOINT	16
2.3.2	VSOP	18
2.3.3	ORIGEN2.....	24
2.3.4	JACS.....	30
2.3.5	Summary.....	32
2.4	Requirements of the Common Reactor Physics Code System	33
3.	Basic Concept of the Common Reactor Physics Code System.....	36
3.1	Structure of Stream Programs.....	36
3.1.1	Script Languages	36
3.1.2	Fortran90/95 Languages	37
3.1.3	Summary of Discussion	38
3.1.4	Examples of Linkage of Existing Reactor Physics Codes	39
3.2	Solvers to be Developed.....	39
3.3	Interface and Design	40
3.3.1	Interface Design with XML	41
3.3.2	Module Structure Design.....	42
3.3.3	Minimum Data Interface Design.....	46
4.	Conclusion.....	49
	Acknowledgement	50
 Appendices		
1·1	Reactor Physics Code Systems of JNC	51
1·2·1	Reactor Physics Code Systems of JAERI.....	59
1·2·2	Reactor Physics Code Systems of JAERI (FBR).....	66
1·3	Reactor Physics Code Systems of NFI.....	67
1·4	Reactor Physics Code Systems of CRIEPI.....	69
1·5·1	Reactor Physics Code Systems of NUPEC	72

1·5·2 Reactor Physics Code Systems of NUPEC (Handout of Presentation).....	73
1·6 Reactor Physics Code Systems of Toshiba (FBR)	80
1·7·1 Reactor Physics Code Systems of MHI(LWR)	81
1·7·2 Reactor Physics Code Systems of MHI(FBR)	82
1·8 Reactor Physics Code Systems of Hitachi	83
2·1 Proposal for Common Reactor Physics Code Systems (Kobayashi).....	85
2·2 Evaluation of the Kobayashi Proposal (Yokoyama).....	96
2·3 An Example of Command-base Code Systems (Matsumura)	108
3·1 The Minute of 1st WP Meeting (Nov. 26, 2001).....	123
3·2 The Minute of 2nd WP Meeting (Feb. 22, 2002).....	128
3·3 The Minute of 3rd WP Meeting (July 19, 2002)	133
3·4 The Minute of 4th WP Meeting (Dec. 6, 2002).....	137
4·1 An Abstract of Presentation for AESJ Annual Meeting (2003 Spring).....	142
4·2 A Handout of Presentation on AESJ Annual Meeting (2003 Spring).....	143

1. 緒言

共用炉物理コードシステムの構築WPは、我が国の炉物理研究者がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築することを目的に、平成13、14年度の2カ年の期間で4回の会合を持ち検討を実施した。平成12年度に原子力学会「共用炉物理コードシステム」特別調査専門委員会が設置された⁽¹⁾。ここでの1年間の議論を踏まえ、小林啓祐・京都大学名誉教授より、平成13年度の第71回炉物理研究委員会で提案され、設立に至った。

WPは以下の委員11名、オブザーバー4名、常時参加者1名（計16名）の参加を得て、活動した。

- ・リーダー：松村哲夫（電中研）
 - ・WP幹事：小林 啓祐（京都大学名誉教授）、別所 泰典（NUPEC）、池田 一三（三菱重工）、飯島 進（原研）*
 - ・WPメンバー：
 - 山岡 光明（東芝）、横山 賢治（サイクル機構）、石井 一弥（日立）、内藤 健孝（NAIS）、日山 伸行（原燃工） - ・オブザーバー：
 - 山本 徹（NUPEC）、岩崎 智彦（東北大）、中島 鐵雄（NUPEC）、杉 嘉夫（原研） - ・常時参加者：
 - 中川 正幸（原電）
- （*：途中、森 貴正に交代）

WPでは、炉物理部会会員がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築することを目的に、以下の4回の会合を開催し、海外での共用炉物理コードシステム（ERANOS、等）の調査、諸機関の炉物理コードシステムの調査を踏まえて、どのようなコードシステムを作るか、開発が必要な炉物理コードは何か、開発・整備の体制をどのように作るか、などの検討を実施し、コードシステムの基本概念を取り纏めた（詳細は付録3-1～4の議事録参照）。

- 第1回会合：平成13年11月26日（火）：リーダー・幹事の選出、活動計画
- 第2回会合：平成14年2月22日（金）：各機関のコードシステムの紹介
- 第3回会合：平成14年7月19日（金）：コードシステムの議論
- 第4回会合：平成14年12月6日（金）：取り纏めの議論

本資料は、共用炉物理コードシステムの構築WPの2年間の活動内容を取り纏めたものである。

【第1章の参考文献】

- (1) 共用炉物理コードシステム特別専門委員会、「共用炉物理コードシステム特別専門委員会報告書」、JNC TJ9420 2001-007(2001)

2. 活動報告

2.1 諸機関の炉物理コードシステムの調査

WP 参加者より提供された資料を基に、以下の諸機関の炉物理コードシステムの調査を行った。表 2.1-1 に諸機関の炉物理コードシステムの概要を示す。

- ①JNC (付録 1-1)
- ②原研 (付録 1-2)
- ③原子燃料工業 (付録 1-3)
- ④電中研 (付録 1-4)
- ⑤安全解析所 (付録 1-5)
- ⑥東芝 (付録 1-6)
- ⑦三菱重工業 (付録 1-7)
- ⑧日立 (付録 1-8)

(順番は発表順)

なお、表 2.1-1 に示す諸機関の炉物理コードシステムには、WP 参加者より提供された資料により明示された炉物理コード、コードシステムのみを記載している。実際には、諸機関でより広範囲な炉物理コード、コードシステムおよびデータが利用されている。

諸機関の内、開発コードを原則公開可能な機関は原研とサイクル機構 (JNC) であった。このため、原研と JNC の核特性解析コードシステムをより詳細に比較した (表 2.1-2 参照)。これらの調査により、以下のような事柄が抽出された。

- ・ 多群定数ライブラリを用いた Cell (燃料棒、燃料集合体、燃料要素) 計算、炉心計算、燃焼計算の他に、感度解析、炉定数調査、核設計精度評価、共分散処理、動特性、過渡解析、 γ 線計算、装荷パターン作成、炉心安定性解析、遮蔽、臨界安全など非常に広い範囲で多くの炉物理コード、炉物理コードシステムが利用されている。
- ・ 炉型と計算コードは概ね対応しており、軽水炉と FBR、あるいはガス炉など炉型を跨って利用されているコードは少ない。また、使用している多群定数ライブラリーも炉型によって異なる。唯一、MVP などの連続エネルギーモンテカルロ法を用いた計算コードは炉型に関わらず共通的に利用されている。
- ・ 各機関の役割に応じて、保有・開発しているコードには特徴がある。例えば、JNC は、感度解析、炉定数調整・核設計精度評価、共分散処理に関わるコードが充実している。また、プラントメーカーは、装荷パターン作成などのコードを活用している。

現状では、炉物理コード、炉物理コードシステムは目的に応じて、個別に利用されており、炉型毎にはシステム化されつつあるが、総合的な共用炉物理コードシステムの構築の必要性が確認された。

表 2.1-1 諸機関の炉物理コードシステムの調査 (1/2)

	原研	JNC	NUPEC	三菱	東芝	日立	NFI	電中研
多群定数ライブラリー	SRAC(Fast, Thermal, MCRO SS), JFS-3							MGCL(LWR), J FS(FBR)
Cell(燃料棒、燃料集合体、燃料要素)計算	SRAC(Pij), A NISN, TWOINT, N, ASMBURN, SLAROM, LAROM, PEACO -X, DELIGHT	CITATION, JO INT, SLAROM	CASMO-4	ODD, PILOT, P LATINA, PIJ- HEX, LEOPARD	COSMOS, SLAROM	HINES, SRAC	NULIF, GDLUX , ASY5G, CASMO	FLEX-BURN, S LAROM, SRAC
炉定数作成		NJOY, LINEAR , RECENT, SIG MA1,	NJOY, LINEAR , RECENT, SIG MA1					NJOY, LINEAR , RECENT, SIG MA1
モンテカルロ法	MVP, GMVP					VIMONT, MVP-B URN, MCNP		MVP-BURN, MC NP
炉心計算	SRAC, CITATION ON, COREBN MOSRA, CITATION	CITATION, TR ITAC, TWOTRAN, NSHEX, MOS ES	SIMULATE-3	HANYO-DIF, E ENSEMBLE-K, ENSEMBLE-TR IZ, HIDRA, PA NDA, PHENIX- P, ANC	FASTMUGY, NE W3D, CITATION, TW OTRAN-II, TR ITAC	PANACIA, COS NEX, CITATION	SHARP, SIMUL ATE-3	CITATION, COMPWR2
燃焼計算	BURN	CITATION, OR IGEN		2DBURN, TRIANG LE, HEXSZ, TRISTAN	STANBRE, VET SA, ORIGEN			CITATION, OR IGEN
摂動計算	PERKY, SN-PERT RT-II, SN-PERT RT-3D	PERKY, SNPER T		HANYO-PERT, PARM, TRI-PE RT, ENSEMBLE -PERT	PHOBOS			PERKY

表 2.1-1 諸機関の炉物理コードシステムの調査 (2/2)

	原研	JNC	NUPEC	三義	東芝	日立	NFI	電中研
γ線計算				2DBURN-G, TR ANGLE-G				
感度解析	SAGEP	ABLE, ACCEPT		SAGEP				
炉定数調整・核設計 精度評価		ERROR J	EUREKA-JINS /S		TRACG, STAND			
共分散処理						INSIGHT, GAL LOP, OPAL		
動特性・過渡解析	MLK3D							
荷重パターン作成				SKETCH - INS /TRAC-BF1				
炉心安定性解析								
遮蔽・臨界	SRAC, DANTSY S					SCALE, ORIGE N		

(注) 表には WP 参加者より提供された資料により明示された炉物理コード、コードシステムのみを記載している。実際には、諸機関でより広範囲な炉物理コード、コードシステムおよびデータが利用されている。

表 2.1-2 原研と JNC の核特性解析コードシステム

	原研			JNC
	熱炉	高速炉	ガス炉	
システム	SRAC	EXPARAM	NDCS (Nuclear Design Code System)	JOINT コードを用いて連携
多群定数ライブラリ	SRAC : Public Library (Fast, Thermal, MCROSS) PDS ファイル	JFS-3		JFS-3
炉定数作成				NJOY, LINEAR, RECENT, SIGMA1
Cell 計算	SRAC : 衝突確率法 (P_{ij}), ANISN, TWOTRAN, ASMBURN	SLAROM, PEACO-X	DELIGHT	CASUP , JOINT, SLAROM, SLAROM-UF, PEACO, SRAC95
モンテカルロ法	MVP, GMVP	同左	同左	MVP
炉心計算	SRAC: CITATION, COREBN, MOSRA : 拡散ノード法	POPLARS (CITATION), TWOTRAN-II, DAN TSYS	CITATION	CITATION , TRITAC, TWOTRAN, NSHEX (3次元6角体系輸送計算)、MOSES
燃焼計算	BURN			CITATION、ORIGEN
擾動計算		PERKY, SN-PERT-II, SN-PERT-3D		PERKY, SNPERT
感度解析				SAGEP
炉定数調整・核設計精度評価				ABLE, ACCEPT
共分散処理				ERRORJ
過渡解析	MLK3D			

(注) 表には WP 参加者より提供された資料により明示された炉物理コード、コードシステムのみを記載している。実際には、諸機関でより広範囲な炉物理コード、コードシステムおよびデータが利用されている。

2.2 先端計算機技術の活用

現在、サイクル機構では、炉物理に限らず高速炉開発に必要な原子力関連の解析コードについて、次世代型の解析システムの可能性について検討を行っている⁽¹⁾。ここでは、先端計算機技術の活用例に関する調査結果について簡単にまとめる。

2.2.1 既存の解析コードシステムにおける例

まず、炉物理関連の解析コードとして、フランス原子力庁が中心となって開発を進めている欧州炉物理解析システム ERANOS⁽²⁾⁽³⁾について紹介する。ここでは、計算機技術という観点からの紹介であり、炉物理的な特徴は省く。これらの技術は、最先端の計算機技術とは呼べないかもしれないが、参考になる点は多いと考える。

また、原子力以外での試みとして、地球流体分野で作成されている地球流体電腦ライブラリ⁽⁴⁾についても簡単に紹介する。このライブラリは、標準データ構造と対応した処理手法を提案することにより、データの加工交換を容易にし、問題考察能力を高めたいとの要求から作成が開始されたものである。専門分野は異なるものの、共用炉物理コードシステムに求められるニーズとよく似ており、参考になる点があると考えられる。

(1) 欧州炉物理解析システム ERANOS

ERANOS は、CEA がソフトウェア会社と共同で独自に開発した ALOS と呼ばれるソフトウェア開発環境を用いて開発されている。この中で、LU、ESOPE と呼ばれる 2 つのプログラミング言語が独自に開発されており、ERANOS はこの 2 つのプログラミング言語を使って開発されている。ソフトウェアとしてみた場合の大きな特徴として、(1)スクリプト言語（ユーザー言語）LU とシステム言語 ESOPE による「複合言語化」、(2)オブジェクト指向的なデータ構造中心の設計、の 2 点を挙げることができる。ERANOS では、基本的な開発言語として Fortran の拡張版である ESOPE を利用することで計算速度を維持しつつ、同時に、LU と呼ばれる一種のプログラミング言語を採用することで、使いやすい環境を提供している。また、内部で利用されるデータはすべて EDL と呼ばれるデータ形式で定義されており、このデータを中心としたモジュール化設計を採用しているため、解析手法やデータの組み合わせの自由度が高く、拡張性や再利用性に優れた解析システムとなっている。また、ESOPE は、Fortran77 にはない動的なメモリ管理機能を持っており、ERANOS ではメモリ管理は動的に行われているようである。

また、外部記憶装置に一時的にメモリを確保する機能として GEMAT というシステム、データの永続化（シリアル化）のための機能として ARCHIVE というシステムが利用されている。これらのシステムを利用することによって、ハードディスクへの読み書きの操作と、炉物理計算とが分離されているようである。上記の ESOPE、LU、GEMAT、ARCHIVE、EDL 等を含めて ALOS ソフトウェアと呼ばれている。

ユーザーは ERANOS で解析する場合、LU を使って解析コードを「関数（サブルーチン）」を呼び出すような形で利用する。また、炉物理的な概念、例えば、中性子束、核反応断面積、体系等は、EDL 形式で定義されており、ユーザーはこれらのデータをひとつの「変数」として簡単に取り扱える。ユーザーは問題に応じて、一連の解析の流れ（以後、解析ストリーム）を、炉物理

解析用に特化した「関数」と「変数」を使った簡単なプログラムとして表現することで解析を実施する。また、LU 上でデータ型や配列数の宣言等は自動的に処理される上に、炉物理的な概念を単位としてデータや機能が定義されているため、コンピュータに関わる煩雑な問題は隠蔽されており、炉物理の問題に意識を集中できるため、分かりやすく使いやすい解析システムとなっている。また、単に利用しやすいというだけでなく、LU で記述された解析スキームは、ERANOS の新たな「関数」として登録可能であり、機能の拡張性にも優れている。

このように、ERANOS では、炉物理解析に必要な機能やデータを「部品」として用意しており、ユーザーが利用する段階で組み合わせて使うという方針が採られているため、自由度が高く、機能・データの再利用性、拡張性に優れている。これらの性能は、(1)、(2)を技術基盤として実現されていると考えられる。図 2.2-1 に、ERANOS を参考にした場合の共用炉物理コードシステムのイメージ案を示す。

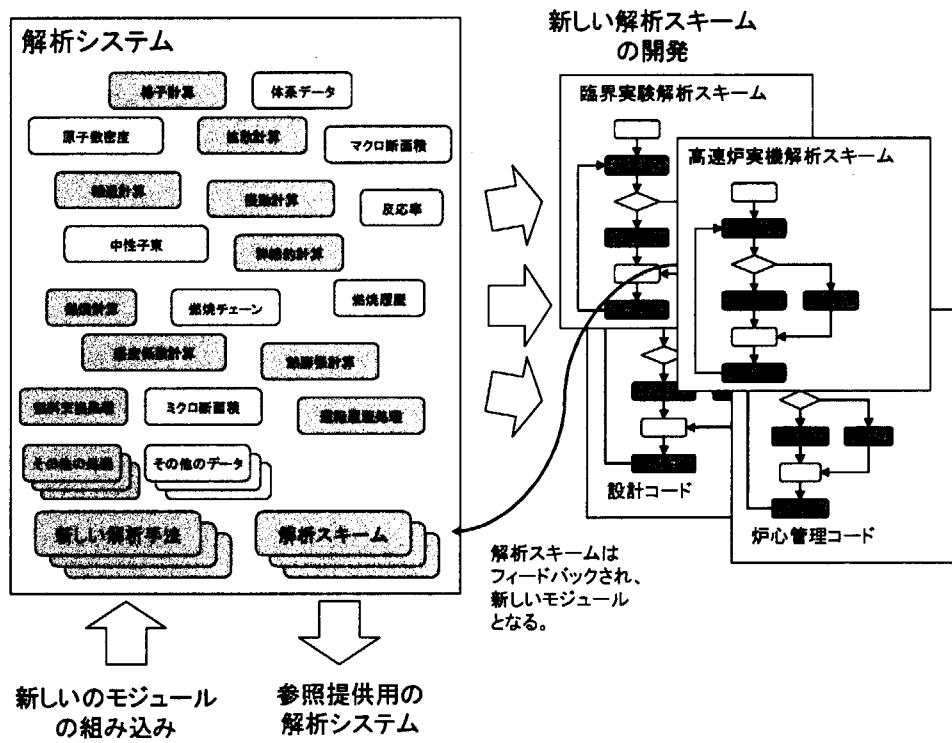


図 2.2-1 共用炉物理コードシステムに求められるシステムのイメージ案

(2) 地球流体電腦ライブラリ

原子力分野ではないが、地球流体の分野では、科学技術振興事業団の計算科学技術活用型特定研究開発推進事業として、平成 10 年度に「地球惑星流体现象を念頭においてた多次元数値計算データの構造化」というテーマでプロジェクト研究が行われた⁽⁴⁾。

地球や惑星での流体现象（大気や海洋、マントルや中心核）における研究では、得られた多次元データを認識可能な形に整形するのに多大な労力がかかることが問題となっていた。これらの問題を解決するために、データのオブジェクト指向的な構造化の導入により、数値計算や観測データ解析で関連する問題を整理合理化することを目的として、このプロジェクト研究が実施され

た。

このプロジェクト研究の報告書では、地球惑星流体现象にまつわる問題点は、

- ・ 大規模、すなわち、サイズが大きい、あるいは、数が多い
- ・ 多様、すなわち、個人、グループ、プロジェクトなどの生産元ごとにデータの形式構造が異なっている

の二点にまとめることができるとしている。これらの課題に対して、このプロジェクト研究では、標準データ構造と対応した処理手法を提案することにより、データの加工交換を容易にし、問題考査能力を高めたいという要求があった。この要求を達成するために、必要となるデータ構造の性質は、

- ・ ネットワーク透過性の確保、すなわち、機種/OS 非依存のバイナリデータであること
- ・ 自己記述的なデータ構造とデータ処理、すなわち、データ自身がその中身を語ってくれる自己記述性とデータ自身がどう処理して欲しいか語ってくれるオブジェクト指向であること

の二点であるとしている。

このプロジェクト研究では、上記の目的を達成するために、多次元データに対して、以下の三層に分けたアプローチを採用した。

- ・ データの構造
- ・ データの処理と表示を念頭においていた属性リストの整備
- ・ NetCDF⁽⁵⁾を機種/OS 依存しないバイナリ形式として採用
- ・ データの処理
- ・ 地球流体での数値処理に適した Fortran90 によるライブラリ (Fortran によるオブジェクト指向的実装)
- ・ オブジェクト指向スクリプト言語 Ruby⁽⁶⁾によるライブラリ
- ・ データの可視化
- ・ 研究教育現場で自由に流通させることができ（学生への配付が可能）な最低限の可視化資源の確保

この研究では、NetCDF と呼ばれるデータ形式を採用し、多次元データの格納形式を提案している。このデータ形式の特徴としては、

- ・ 自己記述的
- ・ 非格子状データのサポート
- ・ 数値データと可視化情報の統合
- ・ 機種・OS 非依存性

といったことが挙げられている。

このプロジェクト研究では、地球流体科学における大規模な多次元データの可視化処理が主目的である。原子力関連の解析コードの連携においても、大規模な多次元データが扱われており、これらのデータに対する連携処理が、現在、我々が解決すべき課題であるので、この研究によって得られた知見を利用できる可能性は高いと考えられる。

データ処理に関しては、従来から整備されていた Fortran77 ベースのツールと親和性の良い

Fortran90 や C 言語による実装と、オブジェクト指向スクリプト言語である Ruby による実装の、二種類の実装が試された。Fortran90 による実装がメインであり、Ruby による実装は、試験的なものであったようであるが、この試験的実装により、Ruby により、柔軟で軽快なデータ解析環境が提供される可能性が示せ、かつ、NetCDF のようなデータ形式も容易に扱うことができるとしている。

ただ、現状の問題点として、Ruby のコミュニティにおいて、数値処理の経験が少ないとこと、スーパーコンピュータ上への効率の良い実装、巨大データに対する実装等を挙げている。

なお、この研究では、当初、Ruby ではなく、欧米で既に流通している Python⁽⁷⁾を利用する計画であったらしいが、Ruby の方がより純粹にオブジェクト指向であり、かつ、言語開発者が国内にいるので、より緊密な連係がとれるであろうとの判断から、Ruby を採用することにしたようである。

2.2.2 解析ストリーム構築のために利用可能な技術

サイクル機構で検討を行っている次世代解析システムでは、解析コードを連携させて利用するためには、計算の流れを制御するためのプログラミング言語（制御言語）と、実際の計算を行わせるためのプログラミング言語（計算言語）が必要であると考えている。この考え方を共用炉物理システムに当てはめると、解析ストリーム構築を行うのは、「制御言語」の役割であると考えられる。ここでは、制御言語と計算言語の概念と、これらに利用できると考えられる技術について紹介する。

なお、熱過渡応力解析等では、原子炉冷却材ループを記述するためには、GUI (Graphical User Interface) によるインターフェースが重要であると考えているが、炉物理解析システムとしては、GUI を利用したいというニーズは弱いと思われる所以、ここでは、GUI による制御については省略する。詳細は参考文献(1)を参照されたい。

(1) システム言語とスクリプト言語

制御言語と計算言語という考え方によく似た概念として、Ousterhost が定義したスクリプト言語（Scripting Language）とシステム言語（System Programming Language）という概念がある⁽⁸⁾。Ousterhost の分類によると、プログラミング言語は、スクリプト言語とシステム言語の二つに大きく分けることができる。参考文献(8)の中では、スクリプト言語とシステム言語の特徴について、以下のように説明されている。図 2.2-2 に参考文献から図を引用する。

①スクリプト言語 (Scripting Language) :

- ・ 既存の強力な機能や要素を結合 (Gluing : 粘付け) して利用するための言語。
- ・ データの型付けが緩いため、要素の組合せを簡素化することが可能であり、迅速なアプリケーション開発を促進する。
- ・ 短いプログラムで高度な処理が可能であるが、実行効率は悪い。

②システム言語 (System Programming Language) :

- ・ メモリ等のコンピュータの基礎レベルから、データ構造やアルゴリズムを構築するための言語。

- データの型付けが強いため、複雑なデータ構造を管理するのを援助する。
- プログラムは長くなるが、実行効率は良い。

スクリプト言語は、短いプログラムで高度な処理が可能であることを特徴としており、プログラム一行あたりに処理可能な仕事量が大きいことが特徴である。その代わりとして、システム言語に比べて実行速度は遅くなる。もう一点、スクリプト言語の特徴として、データの型付けが弱いということが強調されている。

更に、スクリプト言語とシステム言語はそれぞれ、以下のような特徴を有しており、必要とされる条件に対して、それぞれ適正があると述べられている。

また、以下では、解析コードシステムに対する要件という観点からは、どちらの言語が適しているかについて考察した結果を、簡単にまとめた。

①スクリプト言語を選択すべき要件

- | | |
|---------------------------|-----------------|
| ・ 既存の要素を組合せることが主な仕事か？ | Yes (解析コード群の連携) |
| ・ 様々な異なった事象の操作が必要か？ | Yes (対象は複雑で様々) |
| ・ グラフィカルユーザーインターフェースを含むか？ | Yes (結果表示、入力等) |
| ・ 機能は速い速度で進化するか？ | Yes (解析手法は進歩する) |
| ・ 拡張性は必要か？ | Yes (解析対象は変化する) |

②システム言語を選択すべき要件

- | | |
|------------------------|---|
| ・ 複雑なアルゴリズムやデータ構造が必要か？ | Yes (ともに複雑) |
| ・ 大きなデータ処理や計算の速度は重要か？ | Yes (ともに重要) |
| ・ 機能はよく定義されており、変化は遅いか？ | Yes&No
(物理的概念はよく定義されているが解析手法などは変化する) |

このような観点から、解析コードシステムについて考えると、ほとんどが必要な条件であることが分かる。従来の解析システムでは、「複雑なアルゴリズムやデータ構造」、「大きなデータ処理や計算の速度」が重要視されてきたため、システム言語である Fortran が主に利用されてきたと考えられる。

しかしながら、解析ストリームの構築は、すなわち、解析コードの連携や新しいモジュールの組み込み等の変化への対応のことであり、スクリプト言語を選択すべき要件も強くなってきていくことがわかる。

表 2.2-1 に、参考文献(8)で述べられているプログラミング言語の分類と変遷をまとめた。この分類では、JCL (Job Control Language) はスクリプト言語となる。既存の炉物理解析コードは、多かれ少なかれ、JCL で操作されることが前提になっていることが多い。JCL より優れたスクリプト言語である Perl、Python、Ruby 等を導入することで、解析ストリームの構築に柔軟に対応できる可能性が高い。

一方で、計算速度等の要件は満たさなければならないので、システム言語とスクリプト言語の両方の特徴を兼ね備えた開発環境が理想的である。

このような観点からは、Perl、Python、Ruby 等のスクリプト言語を採用する場合、SWIG⁽⁹⁾ と呼ばれる C 言語、C++ 言語との結合ツールが存在しており、スクリプト言語とシステム言語の

融合は技術的に可能である。また、Python では、C 言語だけでなく、Pyfort⁽¹⁰⁾、f2py⁽¹¹⁾と呼ばれるツールが開発されつつあり、Python と Fortran を結合して利用することも可能となっている。なお、先に紹介した、地球流体電腦ライブラリでは、既存の Fortran のサブルーチン群を、一旦、f2c と呼ばれるツールで C 言語に変換してから、Ruby と結合している。直接、Ruby と Fortran をつなぎあわせるツールはまだないようであるが、C 言語を介することで、実質的に既存の Fortran ルーチンと Ruby が結合されている。

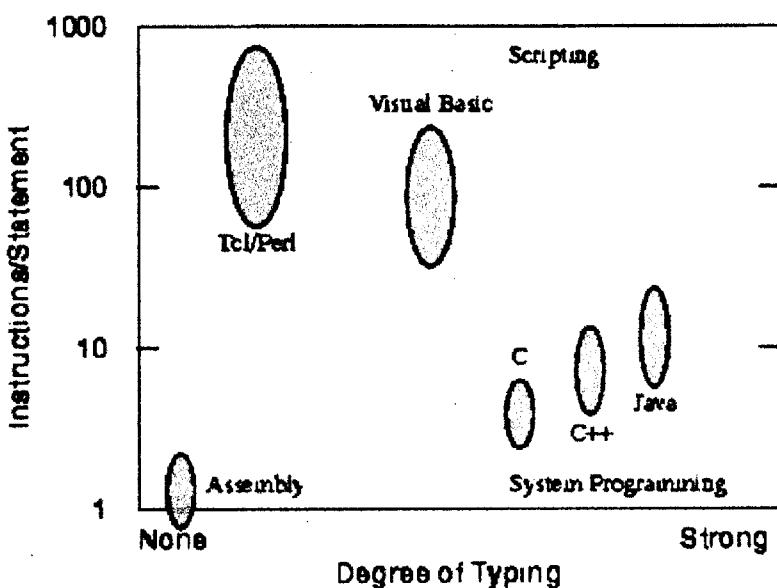


Figure 1. A comparison of various programming languages based on their level (higher level languages execute more machine instructions for each language statement) and their degree of typing. System programming languages like C tend to be strongly typed and medium level (5-10 instructions/statement). Scripting languages like Tcl tend to be weakly typed and very high level (100-1000 instructions/statement).

図 2.2-2 プログラミング言語の比較⁽⁸⁾

表 2.2-1 Ousterhost によるプログラミング言語の分類と変遷

	・スクリプト言語	システム言語	計算機システム
1970 年代	JCL (Do ループ機能なし)	PL/1、Fortran、assembly	OS/360
1980 年代	Unix shell (サブルーチン機能なし)	C	Unix
1990 年代	Visual Basic	C、C++(Active X)	PC(Windows)
現在 (1998 年論文 発表時)	JavaScript、Perl、 Tcl、Rexx、...	Java	Internet World
将来	Python、Perl Ver.5、 Object Rexx、Incr Tcl、... (オブジェクト指向のサポート)	C、C++、Java、...	---

(2) 統一化されたデータ形式（インターフェース）

様々な解析コードを連携するためには、ある解析コードに特化したデータ形式ではなく、一般的な物理事象を基本とし、より高度に抽象化されたデータ形式を採用する必要がある。

前述のように、制御言語として優れたスクリプト言語を採用することで、解析ストリームを構築しやすくなることは期待できるが、スクリプト言語を採用しただけでは、自由に解析ストリームを構築できるようになるとは考えられない。どのような形で解析ストリームを構築するにしても、各解析コード間で統一化されたデータをやりとりする必要がある。

このような統一化されたデータとして、ユーザー・開発者が共通に思い描く物理的なイメージに対応するものを定義することができれば、各解析コードは、常にその統一されたデータ形式を介してデータのやり取りを行うことができる。このようにすることで、状況に応じて、解析コードを取り替えることが容易になると考えられる。

このようなニーズに応えるためには、データ形式として、以下のような特徴を有している必要があると考えられる。

- ・ 拡張可能であること
- ・ 自己記述的であること
- ・ 機種・OS 依存性がないこと

新たな解析コードを追加するときに、データ形式を拡張しなければならなくなることは必ず起こりうる。このとき、なるべく既存の解析コードに影響を与えずにデータ形式を拡張できるようにしておく必要がある。

また、数値データの羅列だけのデータ形式では、各解析コードは、どのようなデータが渡されたのかを理解できない。もし、データ中に、データそのものだけでなく、どのような物理データであるかも記述されていれば（自己記述的）、解析コードがデータの内容を理解して適切な処理を行う余地ができる。また、ユーザーにとってもデータの内容を理解しやすくなり、自作のコードの接続なども容易になる。

このような特徴を持つデータ形式としては、以下のものが挙げられる。

- (a) XML
- (b) HDF、NetCDF
- (c) CORBA

(a) XML

XML (eXtensible Markup Language : 拡張可能なマークアップ言語) は、汎用的なデータ記述言語で元は出版業界で使うために文書記述言語として ISO が標準化した SGML (Standard Generalized Markup Language) から派生したものである。SGML はその規約が複雑なため、厳密に取り扱うのが難しいとされているが、XML は SGML に比べると簡略されている。また、XML は、ホームページの記述に利用される HTML (Hyper Text Markup Language) の拡張版であるため、インターネット上で利用されることを前提に設計されている。XML を利用したネ

ネットワーク技術としては、XML-RPC (Remote Procedure Calling Protocol) や SOAP (Simple Object Access Protocol) が有名である。これらは、既存のインターネット技術である HTTP (Hyper Text Transfer Protocol) や SMTP (Simple Mail Transfer Protocol) と XML を使って RPC を実現しようというものである。HTTP、SMTP は、それぞれ、ホームページの閲覧、電子メールの配信に用いられている技術である。

XML を統一データ形式として採用することにより、インターネット技術を利用した並列計算や分散処理などの新しい技術を利用できる可能性も開かれることになる。

また、XML 形式のデータを処理するためのシステムとしては、SAX (Simple API for XML)、DOM (Document Object Model) といった規格が作成されているが、これらのシステムは、Fortran 用には開発されていないため、Fortran で直接 XML を処理することは難しい。C や C++、Java、Perl、Python、Ruby 等では、XML の処理機能の整備が精力的に行われている。

(b) HDF、NetCDF

HDF (Hierarchical Data Format)⁽¹²⁾は、科学データの保存や管理・交換を目的としたファイル形式として、米国イリノイ州立大学に設置された NCSA (National Center for Supercomputing Applications) で開発されたデータフォーマットである。HDF は、名前の示すとおり、階層構造をもつたデータ形式であり、ファイル自体に、フォルダのような入れものを組み込めるようになっている。このため、複雑なデータをひとつのファイルの中でパソコンのファイルのように整理することが可能である。

HDF は巨大データの格納および並列計算機での使用を想定しており、保存と入出力を効果的に行うことに重点が置かれている。HDF5 でのデータ保存は、環境科学、中性子散乱、非破壊試験、航空宇宙、数値流体力学といった科学技術分野から映画製作まで、多岐にわたって利用されている。後述のフランスの次世代型シミュレーションコードの開発においても、データフォーマットのひとつとして検討されている。

NetCDF (Network Common Data Form) は、Unidata (<http://www.unidata.ucar.edu>) で開発された 共通データフォーマットである。Unidata は全米の大学や研究所間で、データやそれらのデータを処理するためのツールなどを共有化するための組織である。NetCDF は、主に数値解析で必要となるデータ形式をターゲットとしており、HDF とよく似ている。なお、前述の地球流体ライブラリでは、この NetCDF がデータフォーマットとして採用されている。

HDF、NetCDF ともに、数値データを取り扱うことを前提としており、自己記述的であるとともに、機種・OS 依存性が発生しないように考慮されている。

(c) CORBA

CORBA (Common Object Request Broker Architecture) とは、OMG (Object Management Group (<http://www.omg.org>)) が策定する標準仕様で、分散システム環境のインフラを整備、標準化するものである。

CORBA は、あらゆるオブジェクトのインターフェースを IDL (Interface Definition Language) と呼ばれるインターフェース定義言語で定義する。この IDL の存在により、異機種

異言語相互運用性（様々なプログラミング言語、プラットフォームにおける透過性）を達成している。この異機種異言語相互運用性こそが CORBA の持つ最も大きな特徴である。

CORBA は、プログラミング言語に全く依存しない非常に汎用的なものであるが、仕様は複雑である。

2.2.3 フランスにおける次世代シミュレーション開発計画

フランスでは、現在、炉物理だけでなく、原子力全般、更には工学全般に対して、シミュレーションコードのための統一的なプラットフォームの開発を進めている。この共通プラットフォームは、オープンソースで構築することを目指しており、SALOME⁽¹³⁾と呼ばれるプラットフォームの開発が進められている。

SALOME の仕様書 (Version 0.1) は、ホームページから入手できるようになっている。Version 0.1 ということもあり、まだ、空白の部分も多いが、この仕様書から技術的な内容がある程度分かる。

この SALOME では、全体的な制御やコンポーネントの結合を行うために、先に紹介したスクリプト言語 Python が利用されている。一方、計算速度の必要な部分には、C++や従来の Fortran によるルーチンが利用されるようである。各プログラミング言語間の結合には、CORBA の利用が検討されており、データフォーマットとしては、HDF、XML 等が検討されている。これらの技術を基盤とし、ネットワークを利用した分散処理システムも考慮されている。

また、ユーザインターフェースとしては、Python によるテキストベースのユーザインターフェースだけではなく、Qt を利用した GUI もあわせて用意する予定のようである。Qt は Windows、Unix、Macintosh 等に対応したクロスプラットフォームの C++GUI アプリケーションフレームワークである。更に、複雑な形状のモデル作成には、CAD (Computer Aided Design) の技術が利用されるようであり、CAD ツールとしては、OpenCASCADE というシステムが利用されている。また、可視化も重要な要素と考えられており、可視化には VTK が利用されるようである。

以上、ここに挙げられたツール群はすべてオープンソースであり、基本的に無償で利用することが可能である。しかしながら、SALOME 本体のシステムについては、試験的なものが公開されているだけである（2002 年 11 月現在）。

なお、このフランスの計画に関して、2002 年 7 月に、フランス原子力庁 (CEA) からサイクル機構に対して共同開発の提案があった。2002 年 10 月の CEA とサイクル機構の間で、次世代のシミュレーションコードの開発について、共同で検討を開始することで合意された。

【第 2.2 節の参考文献】

- (1) 横山賢治、他、「工学系モデリング言語としての次世代解析システムの開発 (I) —課題および要素技術の調査—」、JNC TN9420 2002-004(2002).
- (2) J.Y.Doriath, et al., "ERANOS1: The Advanced European System of Codes for Reactor Physics Calculations," Proc. Int. Conf. on Mathematical Methods and Supercomputing in Nuclear Applications (M&C + SNA'93), 19-23 Apr. 1993 Karlsruhe, Germany, pp.177 (1993).

- (3) Gerald Rimpault, et al, "The ERANOS Code and Data System for Fast Reactor Neutronic Analysis," Proc. Int. Conf. on the New Frontiers of Nuclear Technology, PHYSOR2002, Seoul, Korea, October 7-10, 2002.
- (4) 林洋介、他、「地球惑星流体現象を念頭において多次元数値データの構造化」、平成10年度計算科学技術活用型特定研究開発推進事業（短期集中型）研究開発終了報告書（2000年4月）。
[\(http://dennou.h.gfd-dennou.org/arch/davis/workshop/2000-03-09/\)](http://dennou.h.gfd-dennou.org/arch/davis/workshop/2000-03-09/)
- (5) Russ Rew, "NetCDF User's Guide for C, An Access Interface for Self-Describing, Portable Data Version 3," Unidata Program Center (1997).
[\(http://www.unidata.ucar.edu/packages/netcdf/\)](http://www.unidata.ucar.edu/packages/netcdf/)
- (6) まつもとゆきひろ、石塚圭樹、「オブジェクト指向スクリプト言語 Ruby」、ASCII Software Science Language ISBN:4756132545 (1999年11月)。
- (7) Guido van Rossum, Fred L. Drake, Jr., "Python Reference Manual", Python Documentation, Online Document.
[\(http://www.python.org/\)](http://www.python.org/)
- (8) J.K.Ousterhout, "Scripting, Higher Level Programming for the 21st Century", IEEE Computer Magazine (1998).
- (9) David M. Beazley, "SWIG Reference Manual Version 1.1", Department of Computer Science University of Utah Salt Lake City, Utah 84112 (June 1997).
- (10) Paul F. Dubois, "Pyfort Reference Manual Version 6", Program for Climate Model Diagnosis and Intercomparison Lawrence Livermore National Laboratory Livermore, CA (2000).
- (11) Pearu Peterson, "f2py Fortran to Python Interface Generator Second Edition Revision : 1:16", (2001).
- (12) Unidata, "A User's Guide for HDF," [\(http://hdf.ncsa.uiuc.edu/HDF5/\)](http://hdf.ncsa.uiuc.edu/HDF5/)
- (13) Open Source framework for pre/post processing, SALOME
[\(http://www.opencascade.org/SALOME/\)](http://www.opencascade.org/SALOME/)

2.3 統合型炉物理コードの調査

炉物理コード（あるいは機能）を統合化したコード／システムとして、前述の欧州炉物理解析システム ERANOS の他、JOINT、VSOP、ORIGEN2 コード、JACS コードシステムの調査を実施した。

2.3.1 JOINT

JOINT コードは、原研で開発された高速炉の核特性解析に用いる種々のコードのインターフェースプログラムである⁽¹⁾。使用される断面積の形式を統一し全て PDS ファイルに収納される。図 2.3.1-1 に示す様に、実効断面積作成コード SLAROM、ESELEM5、EXPANDA-G と拡散・輸送計算コードなどを行う CITATION-FBR、ANISN-JR、TWOTRAN2、PHENIX、3DB、MORSE、CIPER、SNPERT などを結合している。また、断面積の編集や図形出力等を行うサービスプログラムが付随されている。

JOINT への入力は、JOINT コード自体への入力と、各単体コードの入力から構成される。各単体コードの入力は断面積関係の入力が僅かに変わるだけで、殆ど、オリジナルコードの入力と同じである。制御文は、JOINT 用制御文の後に単体コードの制御文が続き、両者の間で入力ファイルと断面積ファイルが引き継がれる。

PDS ファイルは原研で開発された分割型直接アクセスファイルであり、各データはメンバー名を付けてファイルに登録される。メンバー名は “SAMPLIS0” などの 8 文字で定義される。また、各メンバーのファイルには ID、巨視断面積、微視断面積、核分裂スペクトル、バックリング、中性子束などが一定のルールで収められる。尚、PDS ファイルは元々大型計算機用に開発されたが、現在は、ワークステーションや PC 用のものも開発されている。

JOINT を含む各計算コードの計算制御はマクロ形式などを活用した JCL で制御される。

【第 2.3.1 節の参考文献】

- (1) 中川正幸、阿部純一、佐藤若英：「高速炉の核特性解析用コードシステム」、JAERI-M 83-066, (1983).

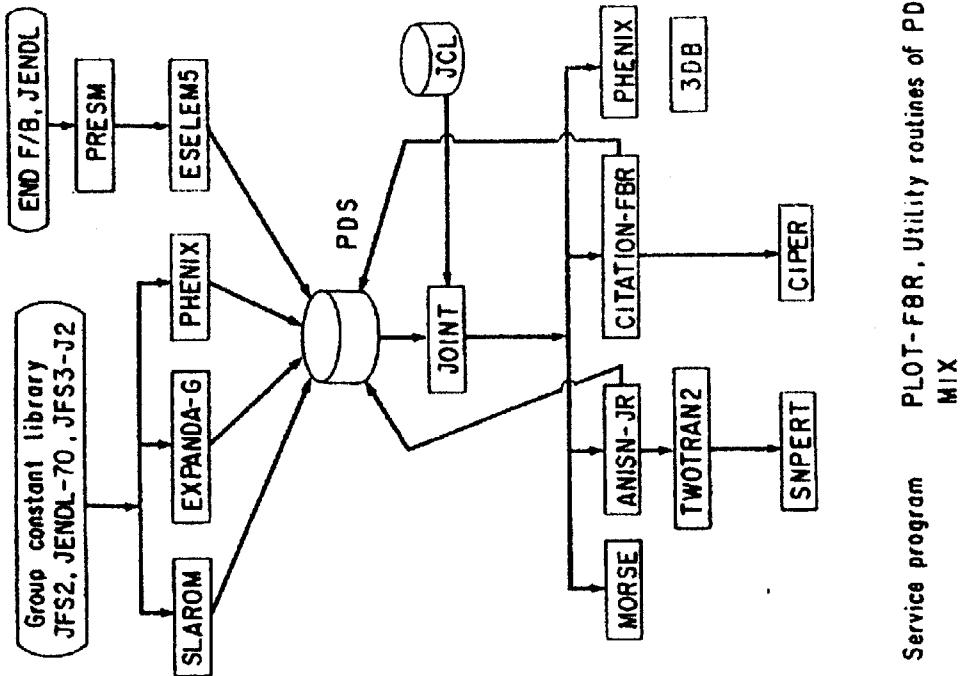


図 2.3.1-1 JOINT コードをインターフェースとした高速炉の核特性解析システム⁽¹⁾

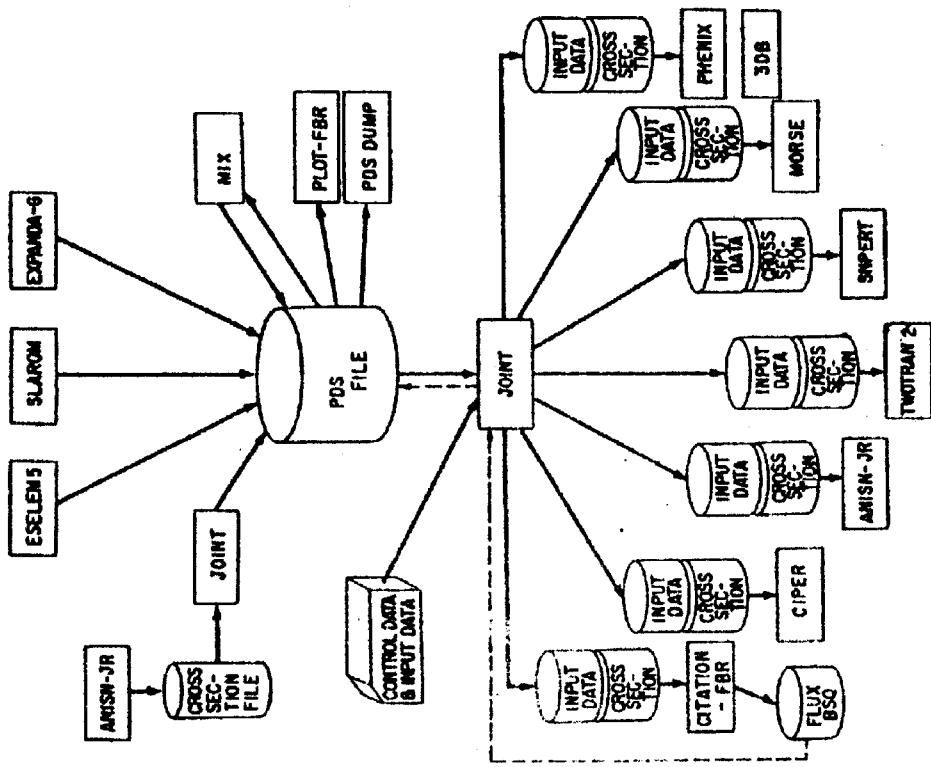


Fig. 2 Data flow diagram in code system

Fig. 1 Calculation flow in code system

2.3.2 VSOP

(1) 機能

VSOP(Very Superior Old Program)はドイツ・ユーリッヒ原子力研究所において開発され、球状燃料(ベブル)の高温ガス炉の開発に使用され、広くこのタイプの炉(ブロック型高温ガス炉、軽水炉、重水炉にも適用可能とされている)に対して特性を計算できる。このコードはステップ毎に断面積作成、2次元あるいは3次元拡散計算、燃料の移動を模擬した燃焼計算、熱計算、燃料サイクルコスト計算を行う。

(2) VSOPの構成と特徴

1)既存の公開コード(ZUTDGL、GAM、THERMOS、CITATION、THERMIX)を組み込み、前処理のモジュール(DATA2、BIRGIT/TRIGIT)、コントロールのモジュール(VSOP)と組み合わせ、炉心計算ができるようにした統合型のコードシステムである。

図 2.3.2-1 : VSOP の物理モデル

図 2.3.2-2 : VSOP の構成

2)特徴は以下の通り。

長所

- 機能が多い(燃焼計算、反応度係数計算、温度計算、物質収支、燃料サイクルコスト計算)。
- パラメーターサーベイ、一連の設計作業が効率的にできる(と思われる)。
- 崩壊熱計算コード ORIGEN、輸送計算コード DOT などにデータを支給するためのヒストリーファイル、ステータスファイルが出力される。

短所

- いろいろなコードを接続した結果、入力データが専用に開発されたものに比べ多い(と思われる)。
- 新たな機能の追加は容易ではない(と思われる)。

(3) モジュール

①DATA2 : 燃料データの前処理

高温ガス炉の燃料仕様を入力し、以下のモジュールで使用する原子数密度、形状データに換算する。

②BIRGIT : 炉心データの前処理

高温ガス炉の炉心仕様を入力し、まずVSOPのモデルを作成し、その後CITATION、THERMIXのモジュールで使用する原子数密度、形状データ、出力分布データに変換する。

図 2.3.2-3 : VSOP から CITATION への原子数密度の変換

図 2.3.2-4 : VSOP から THERMIX への形状データの変換

③ZUTDGL : 共鳴積分計算

ZUTDGL は内蔵された核定数(ENDF/B-IV,-V,JEF-1 他から選択)から Th232,U238 の共鳴積分計算を行い、断面積を作成する。

④VSOP

燃料、炉心条件を入力し、断面積作成(GAM,THERMOS)、燃焼計算(CITATION)、燃料サイクルコスト(KPD)、燃料交換・管理を行う。

⑤GAM-1：熱外中性子以上の中性子スペクトル計算

GAM-1 は熱外中性子以上の中性子スペクトル (10MeV から 0.414eV、68 群) を計算する。このとき、ZUTDGL により計算された分離共鳴および非分離共鳴の断面積を使用する。計算は均質組成、 P_1 近似で計算される。中性子の漏洩はバックリングで考慮する。GAM-1 の炉定数の核種数 (天然組成の元素、ランプ化 FP を含む) は 181 種類である。

⑥THERMOS：熱中性子の中性子スペクトル計算

THERMOS は熱中性子領域の中性子スペクトル (0 から 2.05eV、80 群) を計算する。THERMOS は吸収物質と散乱物質の 2 つからなる。散乱物質は数個から 18 個の温度点毎に別のものとして与えられている。

GAM-1 と THERMOS からの計算から少数群群数 (2 から 8 群) の断面積を作成し、炉心計算に用いる。

⑦CITATION:

2 次元拡散計算、3 次元拡散計算を行う。

⑧FEVER

45 個 (標準) の核分裂生成物からなる燃焼計算、シャッフリングを行う。全核種は 200 個。

⑨THERMIX/KONVEX

定常時およびスクラム後の過渡時の流動計算および温度計算を行う。燃料、減速材温度は中性子計算にフィードバックする。

⑩KPD

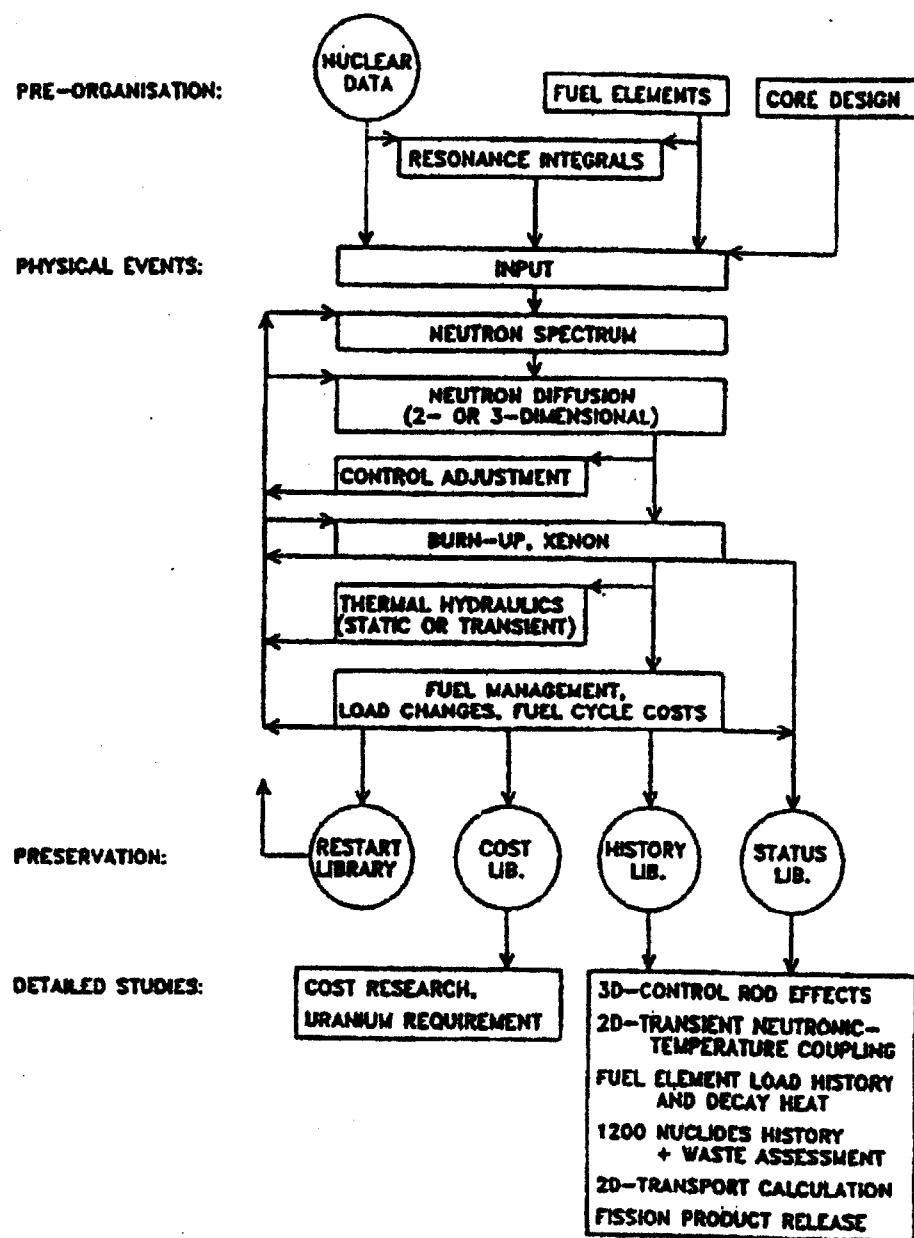
各燃料管理毎に現在価値に換算した燃料サイクルコストを計算する。

⑪LIFE

燃料の履歴を崩壊熱計算等のために蓄積する。

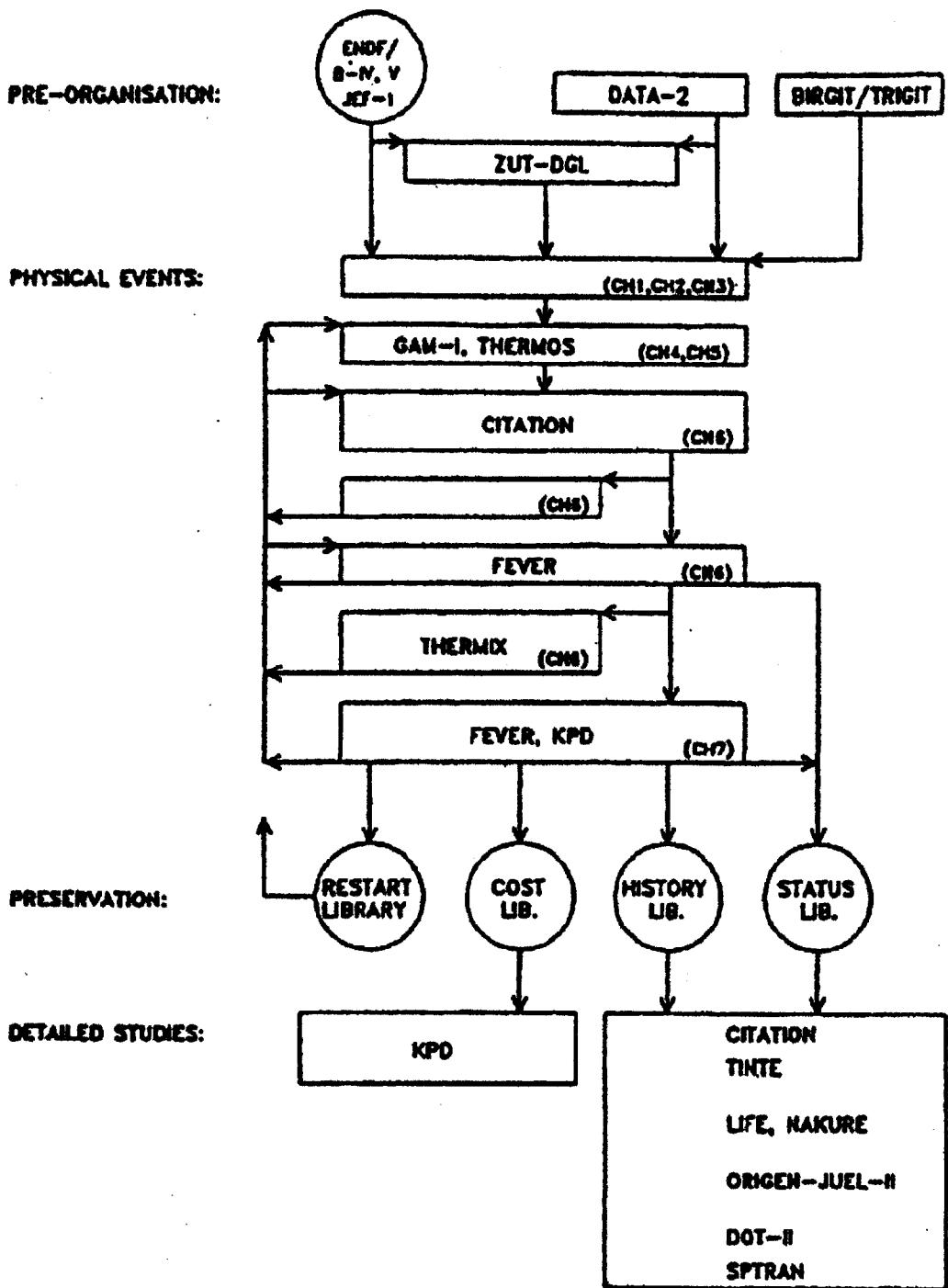
【第 2.3.2 節の参考文献】

- (1) T.Teucher, K.A.Haae, H.J.Rutten, H.Brockmann, H.Gerwin, U.Ohlig, W.Scherer, VSOP(94) Computer Code System for Reactor Physics and Fuel Cycle Simulation, (1994).



V.S.O.P.-Physics Simulation

図 2.3.2-1 VSOP の物理モデル⁽¹⁾



V.S.O.P. The Basic Programs

図 2.3.2-2 VSOP の構成⁽¹⁾

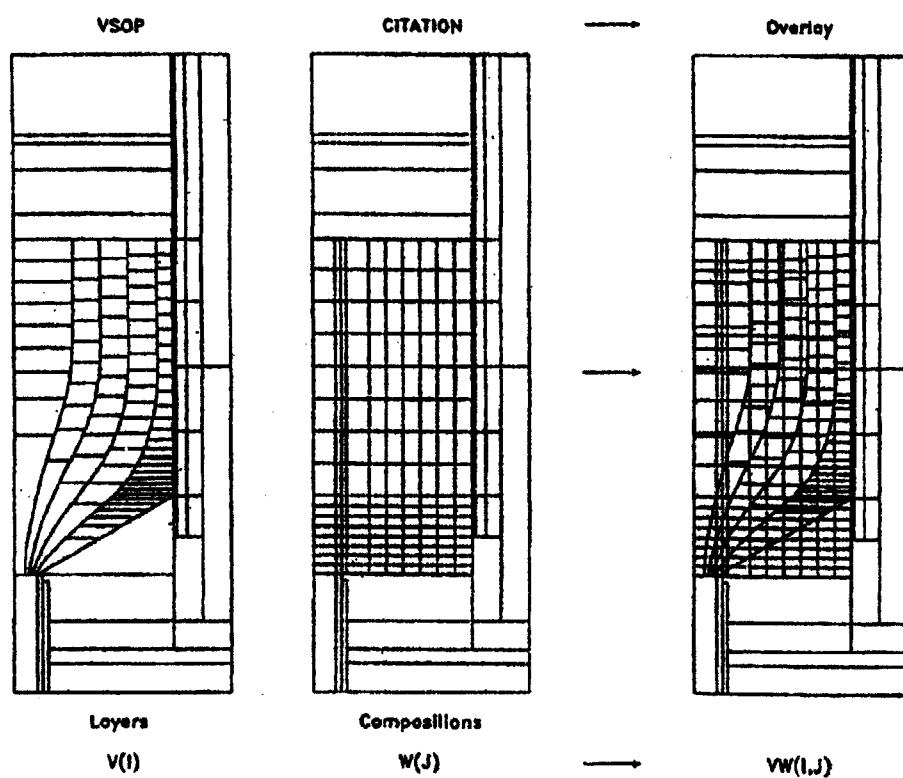


図 2.3.2-3 VSOP から CITATION への原子数密度の変換⁽¹⁾

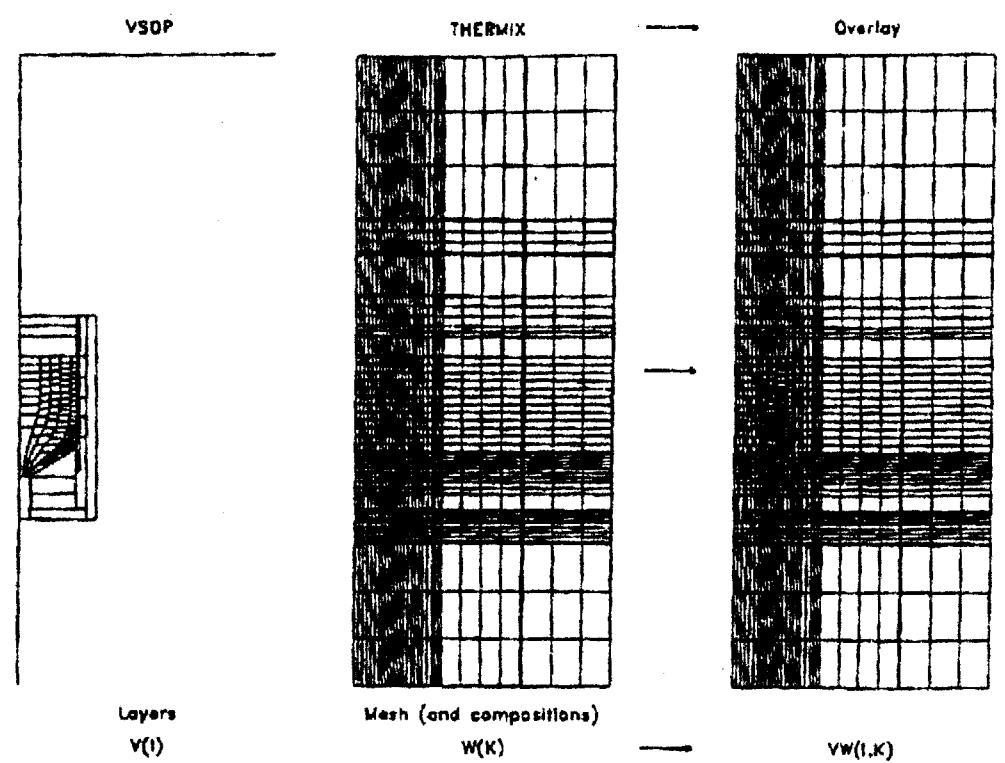


図 2.3.2-4 VSOP から TERMIX への形状データの変換⁽¹⁾

2.3.3 ORIGEN2

共用炉物理コードシステムの構築の観点で ORIGEN2 コードのコマンド機能を紹介する。ORIGEN2 は ORNL で開発された 0 次元の燃焼・生成計算コードである⁽¹⁾。ORIGEN2 コードでは各核種の量をまとめた VECTOR の概念があり、12 個の OUTPUT VECTOR と幾つかの STORAGE VECTOR とがある。VECTOR には H-1 から Po-211 までの放射化核種、He-4 から Es-255 までのアクチニド核種、H-3 から Yb-172 までの核分裂生成核種の量 (Gram·Atom) が収められる。

燃焼計算は 12 個の OUTPUT VECTOR を用いて、燃焼ステップ毎の核種組成の変化として行われる。即ち、燃焼計算後には、12 個の OUTPUT VECTOR には燃焼ステップ毎の核種組成が収納されている。STORAGE VECTOR は種々のコマンド操作に利用される。ORIGEN2 コードのコマンド機能には、以下の様なものがある。

- ・ADU 2つのベクトルを加える
- ・BUP 燃焼計算
- ・DEC 崩壊計算
- ・DOL DO ループ
- ・INP 核種組成などの読み込み
- ・IRF フラックス量を規定した照射
- ・IRP 出力密度を規定した照射
- ・LIB ライブラリー印刷制御
- ・LPU ライブラリー修正入力
- ・MOV ベクトル間の核種組成の移動
- ・OPTA アクチニド核種出力制御オプション
- ・OPTF 核分裂生成核種出力制御オプション
- ・OPTL 放射化核種出力制御オプション
- ・OUT 計算結果出力
- ・PCH OUTPUT VECTOR のパンチ (ファイル) 出力
- ・PHO γ 線ライブラリー読み込み
- ・PRO 再処理計算
- ・REC ループカウンター
- ・WAC 核種の合計

即ち、これらのコマンドを組み合わせる事により、

- 1)燃料核種組成の読み込み
- 2)燃焼度 33GWd/t までの燃料の燃焼計算
- 3)被覆管、構造材の核種組成の読み込み
- 4)被覆管、構造材の放射化計算
- 5)使用済燃料核種組成と被覆管、構造材核種組成の混合
- 6)再処理
- 7)高レベル廃棄物の崩壊計算

8)被覆管、構造材の崩壊計算
を一連の計算で実施する事が可能である(図2.3.3-1参照)。

【第2.3.3節の参考文献】

- (1) A.G.CROFF, "ORIGEN2: a versatile computer code for calculating the nuclide compositions and characteristics of nuclear materials," Nuclear technology 62 (1983).

Table A.1 (continued)

```

63      NYTF=1
64      NYTF=0
65 C CALL SUBROUTINE TO READ CARD INPUT FROM UNIT 5, PRINT IT ON UNIT 6, AND
66 C WRITE IT ON UNIT 50. UNIT 50 IS THEN REWOUND AND ORIGEN2 READS THE DATA
67 C FROM UNIT 50.
68      CALL LISTIT(5,6,50)
69      PRINTW(50)
70 C MAIN1 HANDLES THE MISCELLANEOUS INITIALIZATION DATA
71      1 CALL MAIN1(NYTF,SPNU,ALPEN,NUCRW,NUCSFU,NY,YY,ANHUL,ABEXP)
72 C MAIN2 READS THE ORIGEN2 COMMANDS
73      2 CALL MAIN2(NSTP)
74 C MAIN3 EXECUTES THE ORIGEN2 COMMANDS
75      3 CALL MAIN3(
76          3 LONG,STTFFS,ISTOII,IS,BSTOTI,  LE, 3K, LC,IPD,
77          NUCAR,NUOB,ED,LOC,MCF,MGR,MGR,NUFILD,NUOB,ED,LOC,P,HMAX,KAP,
78          SLOCA,NUUDPF,   CINN,CSUM,           S,
79          SNCLQ,FG,TOCAP,GENHEU,ALPHAB,SPONT,STNU,PISS,ANPC,ENPC,ISTORE,
80          SDIS,E,CGR,YIELD,A ,XT,XP1B,XTEMP,D,F,CORFF,MPROD,   XHEU,
81          SALTIN,NUCIN,NUCSFU,NY,PPSF,FLA,ABUND,RAWLV,LAS)
82 C THIS "GO TO" PROVIDES THE MECHANISM FOR EXECUTING MULTIPLE PROBLEMS WITHIN
83 C A SINGLE JOB.
84      GO TO (1,2,3,4),NSTP
85      4 CONTINUE
86      CALL Q105P(6)
87      STOP 100
88      END
89 /*
90 //LEED-STEPLIN DD DSN=SY81.VSPGE,DISP=SHR
91 //LEED-NEX DD DSN=CHRETECH.AGC14198.0208J,DISP=SHR
92 // DD DISP=SHR,DSN=CHRETECH.Q105P.DUANTO
93 //LKD0.SYSIN DD DISP=SHR,DSN=CHRETECH.AGC14198.020VLF
94 //GO.PT07P001 DD SYSCUT=8,DCB=(RECFM=FB,LRBCL=60,BLKSIZE=3520)
95 //GO.PT09P001 DD DSN=CHRETECH.AGC14198.HECAY,DISP=SHR
96 // DD DSN=CHRETECH.AGC14198.XPMRG,DISP=SHR
97 //GO.FT10P001 DD DSN=CHRETECH.AGC14198.PHOTON,DISP=SHR
98 //GO.FT11P001 DD SYSCUT=1,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
99 //GO.FT12P001 DD SYSCUT=4,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
100 //GO.FT13P001 DD SYSCUT=4,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
101 //GO.FT15P001 DD SYSCUT=4,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
102 //GO.FT16P001 DD SYSCUT=1,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
103 //GO.FT15P001 DD DSN=66AGC,UNIT=SYSSDA,
104 // DCB=(RECFM=FB,LRBCL=60,BLKSIZE=3200),SPACE=(3200,(50,50),RLSE)
105 //GO.FT15P001 DD SYSCUT=4,DCB=(RECFM=VBA,LRBCL=137,BLKSIZE=1100)
106 //GO.FT05P001 DD *
107      92 1 0.99
108      94 1 0.994
109      -1
110      5 1 0.1
111      -1
112      2 15
113      -1
114      DRS      ONE METRIC TON OF PHB FUEL
115      RDA      -1 = FRESH U FUEL WITH IMPURITIES (1 MT)
116      RDA      -2 = FRESH KENDALOY COMPOSITION (1 KG)
117      RDA      -3 = FRESH SS 304 COMPOSITION (1 KG)
118      RDA      -4 = FRESH SS 302 COMPOSITION (1 KG)
119      RDA      -5 = FRESH INCONEL COMPOSITION (1 KG)
120      RDA      -6 = FRESH MICROBASE COMPOSITION (1 KG)
121      RDA      WARNING: VECTORS ARE OFTEN CHANGED WITH RESPECT TO THEIR CONTENT.
122      RDA      THESE CHANGES WILL BE NOTED ON RDA CARDS.
123      CUT      5 0.01 -1
124      LRF      1 1 1
125      LPD      380900 551370 -1
126      LPD      010030 060140 -1
127      LPD      902020 -1
128      LPD      380900      -1

```

ADR

図 2.3.3-1 ORIGEN2 でのコマンドの組み合わせ入力例 (1/4)

Table A.1 (continued)

組成の読み込み

燃烧計算

放射化計算

図 2.3.3-1 ORIGIN2 でのコマンドの組み合わせ入力例 (2/4)

Table A.1 (continued)

195 STP 2
 196 2 922340 240.0 922350 32000.0 922360 967.710.0 0 0.0 FUEL ACTINIDES
 197 0 930000 1.0 050000 1.0 060000 89.4 070000 25.0 FUEL INPUTS
 198 0 680000 134854.0 050000 10.7 110000 15.0 120000 2.0 FUEL INPUTS
 199 4 130000 16.7 180000 12.1 150000 35.0 170000 5.3 FUEL INPUTS
 200 4 200000 2.0 220000 1.0 230000 3.0 240000 4.0 FUEL INPUTS
 201 4 250000 1.7 260000 1.0 270000 1.0 280000 2.0 FUEL INPUTS
 202 4 290000 1.0 300000 40.3 420000 10.0 470000 0.1 FUEL INPUTS
 203 4 480000 25.0 490000 2.0 500000 8.0 640000 2.5 FUEL INPUTS
 204 4 780000 2.0 820000 1.0 830000 0.4 0 0.0 FUEL INPUTS
 205 0
 206 4 400000 379.11 500000 16.0 260000 2.25 240000 1.25 ZIRC-4
 207 4 280000 0.02 130000 0.024 050000 0.00033 450000 0.00025 ZIRC-4
 208 4 060000 0.120 270000 0.010 290000 0.020 720000 0.078 ZIRC-4
 209 4 010000 0.013 250000 0.020 070000 0.080 080000 0.950 ZIRC-4
 210 4 160000 0.035 220000 0.020 750000 0.020 230000 0.020 ZIRC-4
 211 5 920000 0.0002 0 0.0
 212 0
 213 4 260000 692.45 240000 190.0 280000 90.0 250000 20.0 SS-304
 214 4 060000 0.8 150000 0.45 160000 0.3 140000 10.0 SS-304
 215 4 070000 1.3 270000 0.8 0 0.0
 216 0
 217 4 260000 180.0 240000 190.0 260000 525.0 130000 6.0 INC-718
 218 4 060000 0.1 270000 6.7 290000 1.0 250000 2.0 INC-718
 219 4 420000 30.0 070000 1.3 410000 55.53 160000 0.07 INC-718
 220 4 140000 2.0 220000 6.0 0 0.0 INC-718
 221 0
 222 4 260000 0.47 240000 149.5 280000 743.4 600000 0.1 NICK-50
 223 4 130000 0.1 050000 0.05 060000 0.1 270000 0.38 NICK-50
 224 4 250000 8.1 070000 0.066 080000 0.03 150000 103.1 NICK-50
 225 4 160000 0.1 18. 10 0.51 220000 0.1 780000 0.1 NICK-50
 226 0
 227 SAS ONE METRIC TON OF INITIAL HEAVY METAL
 228 CGT -1
 229 LIP 0 0 0
 230 OLPO 380900 351370 -1
 231 LIB 0 1 2 -3 0 0 0 9 3 0 1 1
 232 PFO 101 102 103 10
 233 MOV -9 -8 0 1.0
 234 RDA *** REPROCESSING MODULE ***
 235 BCA FUEL IS REPROCESSED AT THE TIME SPECIFIED ON THE NEXT CARD
 236 DEC 160.0 -10 1 4 4
 237 PRO 1 -9 12 -3 CALCULATE 0.05% OF FUEL
 238 PRO 1 10 -5 -2 SEPARATE VOLATILES AND PUT IN -5
 239 PRO 10 8 -2 -1 PUT ELW IN -1
 240 PRO 8 -4 -3 -8 PUT U IN -4 AND PU IN -3
 241 BLS ONE TONNE OF INITIAL HEAVY METAL AT A REPROCESSING TIME OF 160 DAYS
 242 RDI *** ELW DECAY MODULE ***
 243 TIT DECAY OF HIGH-LEVEL PUR-G WASTE; BURNUP=33,000 KWD/STKES
 244 BOV -2 1 0 1.0
 245 HEB 1 * ELW
 246 DEC 0.5 1 3 5 4
 247 DEC 1.0 3 4 5 0
 248 DEC 3.0 4 2 5 0
 249 DEC 10.0 2 3 5 0
 250 DEC 30.0 3 4 5 0
 251 DEC 100.0 4 5 5 0
 252 DEC 300.0 5 6 5 0
 253 DEC 1.0 6 7 7 0
 254 DEC 3.0 7 8 7 0
 255 DEC 10.0 8 9 7 0
 256 DEC 30.0 9 10 7 0
 257 DEC 100.0 10 11 7 0
 258 DEC 300.0 11 12 7 0
 259 DEC 1.0 -4 12 8 0
 260 RDA OPT CARDS HERE
 再处理
 前壤計算

図 2.3.3-1 ORIGIN2 でのコマンドの組み合わせ入力例 (3/4)

Table A.1 (continued)

```

261 OPTL 8 8 8 8 2 8 1 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
262 OPTA 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
263 OPTF 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
264 OUT -12 1 -1 0
265 OUT 12 1 -1 0
266 BDR **** STRUCTURAL MATERIAL DECAY MODULE ****
267 TTX DECAY OF PER STRUCTURAL MATERIAL WASTE: 33,000 KWD/MTRH
268 TOT -8 3 0 1.0
269 RDA FUEL IS REPROCESSED AT THE TIME SPECIFIED ON THE NEXT CARD
270 DEC 160.0 3 1 4 4
271 ADD -9 1 0 1.0
272 SED 1 SH=0.05% P
273 DEC 0.5 1 3 5 5
274 DEC 1.0 3 8 5 0
275 DEC 3.0 9 2 5 0
276 DEC 10.0 2 3 5 0
277 DEC 30.0 3 4 5 0
278 DEC 100.0 4 5 5 0
279 DEC 300.0 5 6 5 0
280 DEC 1.0 6 7 7 0
281 DEC 3.0 7 8 7 0
282 DEC 10.0 8 9 7 0
283 DEC 30.0 9 10 7 0
284 DEC 100.0 10 11 7 0
285 DEC 300.0 11 -8 7 0
286 DEC 1.0 -8 12 8 0
287 OUT 12 1 -1 0
288 OUT -12 1 -1 0
289 END
290 /*
291 //GO.PT03P001 DD +
292 3 380900 5 30.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
293 3 0.0 0.0 0.7 0.0 0.1 0.1
294 3 551370 5 25.0 0.0 0.0 0.0 0.0 0.0 0.0
295 3 0.0 0.0 0.1 0.0 1.0 1.0
296 21 010030 0.001 5=0.0 -1.0
297 21 060140 0.01 5=0.0 -1.0
298 41 902320 3.0 0.001 0.0001 0.02 0.0 0.0 -1.0
299 61 380900 0.08 5=0.0 1.0
300 61 0.001 0.004 0.1 0.02 0.019 0.08 0.006 0.006
301 070150 060140 1.5E-04 X15(N,D PLUS N,NP)C14 CROSS SECTION
302 080160 060140 2.0E-06 016(N,NP)C14 CROSS SECTION
303 3 380900 5 30.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
304 3 380900 0.0 0.7 0.0 0.1 0.1
305 3 551370 5 25.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
306 3 551370 0.0 0.1 0.0 1.0 1.0
307 /*
308 /*
INCO02I STOP 0

```

崩壊計算

図 2.3.3-1 ORIGEN2 でのコマンドの組み合わせ入力例 (4/4)

2.3.4 JACS

JACS コードシステムは臨界安全の解析を目的に日本原子力研究所で開発された計算コードシステムである⁽¹⁾。核データファイルから多群定数ファイル (MGCL137 群) を作成する MGCL-ACE コードから、臨界安全解析を行う ANISN-JR, KENO-IV などのコード群から構成されており、臨界安全ハンドブック^{(2), (3)}の作成に活用されている。インターフェースとしては、MGCL137 群ファイルから ANISN-JR, KENO-IV などの入力（断面積）ファイルを作成出来るように構成されており、JCL の制御により連続計算が可能なシステムとなっている（図 2.3.4-1 参照）。

【第 2.3.4 節の参考文献】

- (1) Y. Naito, et al., "MGCL-PROCESSOR: A Computer Code System for processing Multigroup Constant Library MGCL", JAERI-M 9396 (1981).
- (2) 臨界安全ハンドブック、科学技術庁原子力安全局核燃料規制課編、にっかん書房 (1988)
- (3) 臨界安全ハンドブック第2版、JAERI-1340(1999).

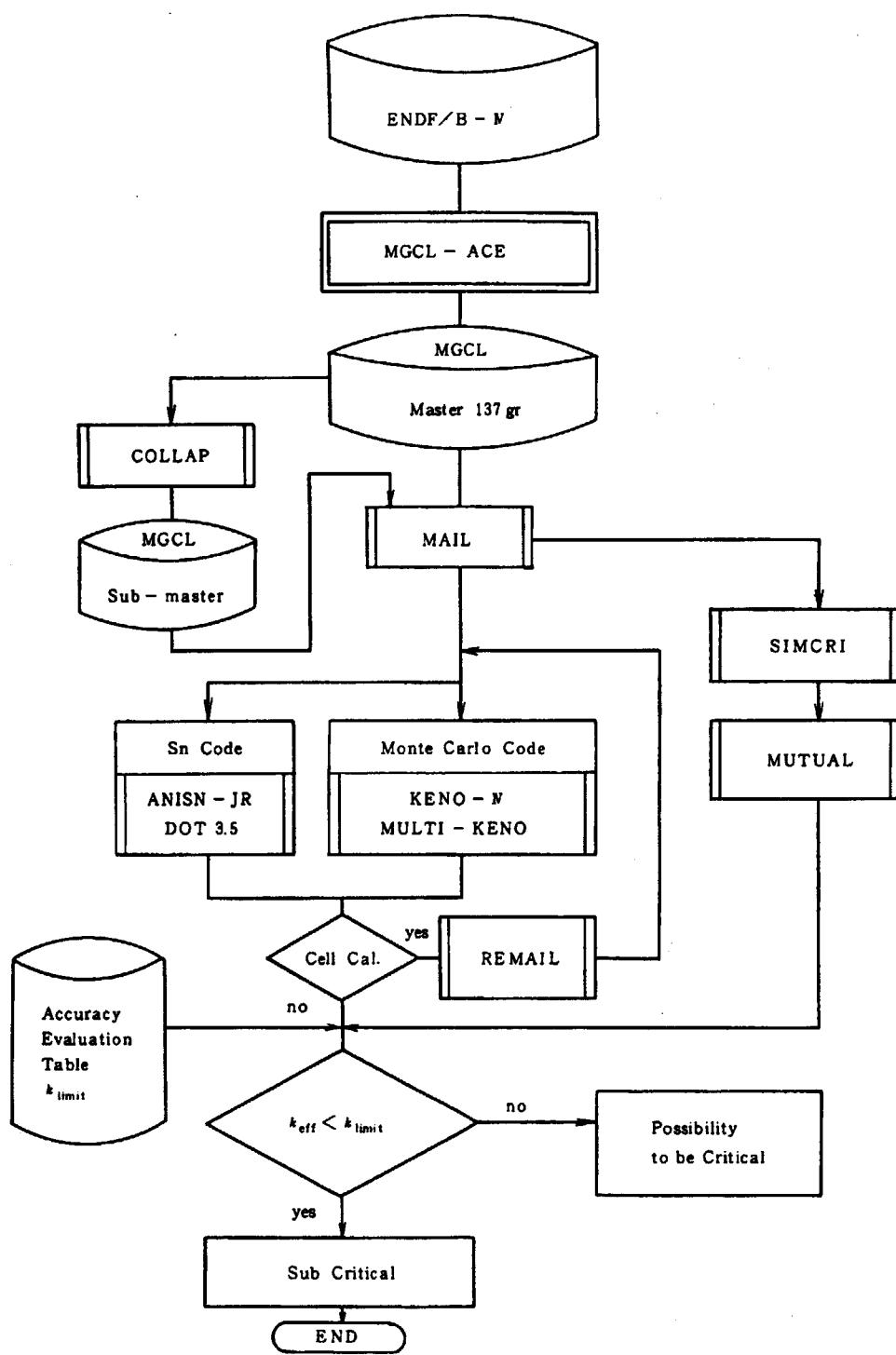


図 2.3.4-1 JACS コードシステムにおける計算の流れ

2.3.5 統合型炉物理コードのまとめ

調査した統合化炉物理コードシステムの要点を表 2.3.5-1 に纏める。

表 2.3.5-1 統合化した炉物理コードシステムの調査

名称	ERANOS	JOINT	VSOP	ORIGEN2	JACS
コード/システム	システム	システム	コード	コード	システム
結合方式	モジュール型	インターフェース型	統合型	統合型（コマンドでサブルーチンを呼び出す）	インターフェース型
言語	開発環境 /ALOS ユーザー言語 LU/システム 言語 ESOPE	Fortran	Fortran	Fortran66	Fortran66
長所	ユーザーに使い易い。ユーザーからは機種依存が少ない	拡張性が高い	一体感があり、操作性が良い	一体感があり、操作性が良い	拡張性が高い
短所	言語から開発する必要がある	機種依存性がある	拡張性が少ない	拡張性が少ない	機種依存性がある

調査の結果、以下の事柄が抽出された。

- ERANOS のように言語から作成すれば、ユーザーからは機種依存が少なく、ユーザーインターフェースが統一された利用者からは便利なコードシステムが構築出来でき、共用炉物理コードシステムの 1 つの方向性を示している。但し、開発費用の問題が懸念される。
- 従来の炉物理コード、システムは Fortran で構築されたものが多く、Fortran 言語が利用出来る必要がある。一方、スクリプト言語、Fortran90/95 言語など計算機技術の進歩もあり、共用炉物理コードシステムの構築に当たっては、先進の計算機技術の活用が望まれる。
- 統合型（1 つのプログラムに仕上げたもの）は、一体感があり、操作性は良いが拡張性が少ない。また、計算に必要な入力データ、メモリー等の計算資源が増加する傾向がある。
- 機能を予め定めて設計するよりも、コマンド型で機能を制御する方式の方が拡張性は高い。
- 種々の既存のコードをシステム化するのであれば、データの遺り取りを行うインターフェース型が優れている。但し、JCL で制御する方式には機種依存性がある。また、目的の異なった異種のコードを結合するには、コード相互のインターフェース（データの遺り取り）の開発に手間が掛かり、インターフェースを合理化する工夫が必要である。

2.4 共用炉物理コードシステムへの要求事項

WP参加者から現行の炉物理コード、コードシステムに対する問題点を調査した結果、以下の様な希望、意見が提出された。

<計算プラットフォーム>

- ・ 互換性が高いこと。UNIX、ウインドウズ、LINUX および大型計算機等への導入ではプログラムの変更を殆ど必要としないこと。
- ・ ベクトル計算機や並列計算機の性能も出るようにして欲しい。
- ・ 自分処の計算機でも（手間無く）稼働する様にして欲しい。

<言語>

- ・ 機種間での移植性を高めるため、応用プログラムは Fortran などの高級言語で書き、アセンブラー言語などの機種依存性の高い低級言語は使わない。
- ・ 先端的なソフトウェア技術を活用して欲しい。

<既存の資産活用>

- ・ 今までに蓄積された膨大な炉物理プログラムおよびデータを活用出来ること。また、それに必要とされるプログラムおよびデータの書き換え作業は可能な限り少ない労力で容易に出来ること。

<計算対象>

- ・ 小型炉・高転換炉・ガス炉等に適用出来、汎用で精度が高く、ユーザーフレンドリーなコードが必要である。
- ・ 革新炉の核計算、臨界安全、遮蔽計算などへの柔軟な対応が可能である必要がある。

<計算環境>

- ・ 現状は、解析コードの数が多く、各計算コードに入出力形式が不統一。
- ・ 開発したプログラムだけでなく、海外から導入したプログラムも利用出来る様にして欲しい。

<計算機能>

- ・ 軽水炉、FBR の中間的な中性子スペクトル炉心に対して精度良い実効断面積が計算出来ない。
- ・ FBR での 2 重非均質効果（非均質集合体など）、2 次元格子計算が必要である。
- ・ 多様な炉型を対象とする場合に既存の解析コードで対応出来ない。
- ・ 多群の集合体平均炉定数と不連続因子を計算出来る集合体解析コードが必要（CASMO-4 では不連続因子が 2 群まで）。
- ・ より高次の計算理論に基づいた近似の少ない、大容量・高速処理の機能を有する計算コードが必要である。
- ・ 種々の核解析手法を簡単に使える様にしたい。

<検証>

- ・ 解析精度などが十分検証されている必要がある。

<開発体制>

- ・ オールジャパンで対応出来る体制が欲しい。
- ・ 海外との連携も必要である。

<保守性>

- 核データライブラリーの更新などにも容易に対応出来る様にしたい。
- 解析手法などのマニュアルが充実している必要がある。

これらの問題点・意見を以下の6点に集約した。

- 既存コード・データを容易に結合して利用出来るストリームプログラムの開発が必要である。このストリームプログラムはプラットフォーム依存性の無い言語で記述する必要がある。この際、既存コード・データは最少の労力で利用（サブシステム化）出来る必要がある。
- 既存コードでカバー出来ない計算機能があり、これを補う新規のソルバーが必要である。
- コード相互、また外部コード・データとも連携が容易なインターフェースを構築する必要がある。
- 検証、ベンチマーク問題の抽出を含め、開発した共用炉物理コードシステムは十分に検証されている必要がある。
- 共用炉物理コードシステムは解法・機能を記述した十分な資料の作成や対外活動にも注意を払う必要がある。
- コードシステムを維持・発展させる仕組みが必要である。

集約した意見を基にした共用炉物理コードシステムのイメージを図2.4-1に示す。具体的には、以下の開発手順（案）が考えられる。

- 共用炉物理コードシステムのシステム設計
- ストリームプログラムの開発
- 既存コード・システムのサブシステム化
- 必要なサブシステムの開発
- 必要なインターフェースの開発
- 検証、ベンチマーク問題の抽出、共用炉物理コードシステムの検証
- 共用炉物理コードシステム資料の作成、対外活動

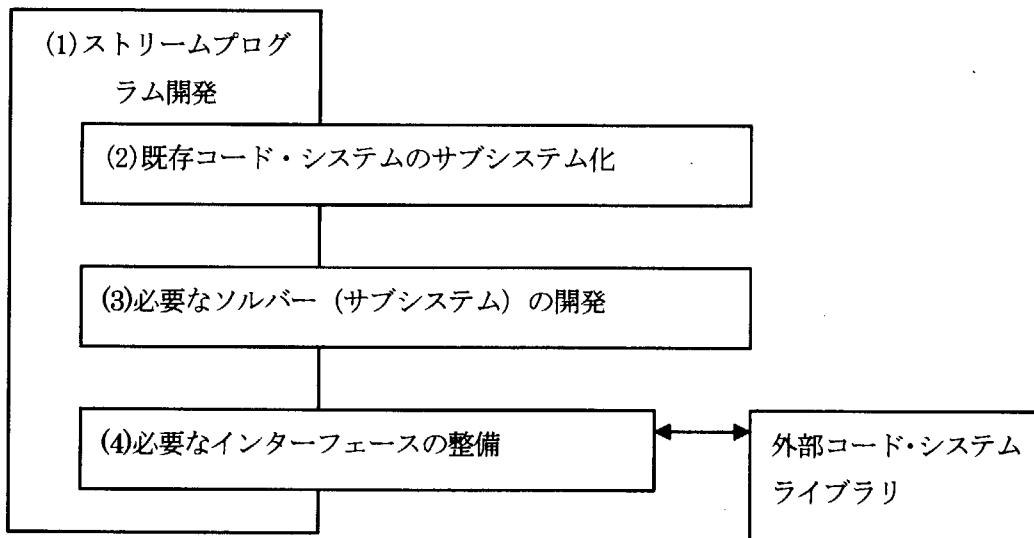


図2.4-1 共用炉物理コードシステムの概念

特に、共用炉物理コードシステムの骨格となるストリームプログラムについては、プログラムを緩く結合するモジュール型、複数の既存コードを1つのプログラムに構成する統合型、入出力ファイルの連結を基本とするインターフェース型が考えられる。これらを比較評価した（表2.4-1）。評価項目として、開発手順（案）も踏まえて、以下を設定した。

- ・ プラットフォーム依存性の少なさ
- ・ 既存コード・データの利用の労力
- ・ インターフェース構築の容易さ
- ・ 拡張性
- ・ ユーザーの使い易さ
- ・ 開発の容易さ

モジュール型、統合型は開発言語がプラットフォーム依存性が少なければ、システムのプラットフォーム依存性も少なくなる。インターフェース型は構成する元のコード、システムはFORTRANなどのプラットフォーム依存性の少ない言語を使用しているが、機種に依存するJCLなどを利用するとプラットフォーム依存性が問題となる。

インターフェース型は既存コード・データをそのまま利用するが、統合型では1つのコードに統合するために、プログラムの大幅な修正が必要となる。モジュール型はこの中間と考えられる。

ユーザーの使い易さは、統一的な設計で開発される統合型が優れている。また、モジュール型でも統一的なユーザーインターフェースが構築可能である。一方、インターフェース型で、計算コードの入力データを計算順序毎に重なるだけでは、使い勝手が悪く、ストリームプログラムに工夫が必要と考えられる。

表2.4-1 ストリームプログラムの構造の比較

形式	モジュール型	統合型	インターフェース型
プラットフォーム依存性の少なさ	○	○	△* ¹
既存コード・データの利用の労力	△	×	○
インターフェース構築の容易さ	○	×	○
拡張性	○	×	○
ユーザーの使い易さ	○	○	△* ²
開発の容易さ	△	○	○

*1 : 機種依存のJCLでは依存性が無い。

*2 : 単純な結合では使い勝手が悪く、工夫が必要。

3. 共用炉物理コードシステムの全体概念

ここでは、第 2.4 節で示した開発手順案を踏まえて、共用炉物理コードシステムの各項目を検討した。

3.1 ストリームプログラムの構造

「既存コード・データを容易に結合して利用出来るストリームプログラムの開発が必要である。このストリームプログラムはプラットフォーム依存性の無い言語で記述する必要がある。この際、既存コード・データは最少の労力で利用（サブシステム化）出来る必要がある」との要請に応えるため、先端計算機技術の活用の調査結果も踏まえて検討を行い、以下の 2 つの案を抽出した。

- ・スクリプト言語の利用
- ・Fortran90/95 の利用（モジュール機能の利用）

3.1.1 スクリプト言語の利用

複数コードシステムを連結して使用する場合、複数のプログラムを一つの「プログラム」としてまとめてシステム化する方式があった（SRAC 等）。この方式で異なるプログラムを結合していく方式ではプログラムの複雑化と各モジュールの制御をどのように記述するかを決定するために高度な設計を必要とする。これに対して各プログラムはそれら自身独立して OS 上で動作するプロセスとして使用し、それらのプロセスのコントロールを他の言語やデータで記述された「ストリームプログラム」上から行う方式の方が柔軟性が高い。ストリームプログラムの記述言語としては Fortran も考えられるが、ここでは汎用スクリプト言語を採用することを提案する。候補は Perl、Ruby、Python などである。

これら汎用スクリプト言語を使用する利点として以下が上げられる；

- ・コンパイルの必要がないこと。
- ・プログラミング言語として理解、習得しやすいこと。
- ・フリーソフトウェアであること（Fortran90 以降はおおむね有料）。
- ・動作する環境として UNIX、Windows いずれもサポートしていること。

もちろんこれらスクリプト言語は各 OS 上でのコマンドとしてプログラムを起動できる。また PerlTk などの使用により GUI (Graphical User Interface) やグラフ表示などをサポートできる可能性がある。

記述例として、Perl を使用した計算のコントロール（引数に指定したパラメータ群による複数計算の一括処理）の例を以下に示す。

```

while(<@ARGV>) {    ;# コマンド行の各引数に対して...
  &create_srac_input($_); # SRAC の入力を作成する何らかの関数呼び出し。
  system("srac") ; # SRAC の呼び出し
}

```

スクリプト言語を利用する場合には、既存のコード、システムを全く修正する必要が無い構築

法も可能と考えられる。

3.1.2 Fortran90/95 の利用

日本原子力学会「共用炉物理コードシステム特別専門委員会」報告書⁽¹⁾に書いたように、本プログラムシステムは、可能な限り多くの計算機でかつ少ない書き換え労力で今までに作られた多数の炉物理プログラムが使え、また最新のプログラムも便利に使えることが必要である。そのためには応用プログラムは、Fortranなどの高級言語で書かれる必要があり、機種依存性のあるアセンブラー言語などの低級言語の使用は避ける事にする。このような高級言語で書かれたプログラムは、現在広く使われているワークステーション、パソコンおよび大型計算機等で殆ど書き換え無しに共通に使えることが期待できる。

Fortran90/95 の MODULE 文を使うと、独立な複数個の Fortran プログラムを Fortran プログラムで 1 個ずつ読み出して順々に実行することが出来る。従ってパソコンまたはワークステーションの RAM メモリは 1 個のプログラムが走る大きさであれば良い。入力および出力データは原則としてハードディスクまたは CD 等に入れてあるファイルから読み出しそしてそこへファイルとして書き出す事にする。この様にすると、多くのプログラムを種々のプログラムの組み合わせで走らせて個々のプログラムの内部を書き換える必要はない。在来のプログラムを利用する場合、書き換えるところはメインプログラムをモジールプログラムと宣言すること、入出力ファイルの名前を読み込む部分を追加するわずかな手間を掛けるだけである。この方法を使えば、例えば広く使われている SRAC コード、輸送コード TWOTRAN や TORT、モンテカルロコードの MVP なども容易に組み込むことが出来よう。具体的な方法は付録 2・1、2・2 に示す。そこに書かれている簡単な Fortran プログラムはウインドウズパソコンで走ることが確認されているが、その後ワークステーションでも走ることは確認されている。

【第 3.1.2 節の参考文献】

- (1) 小林啓祐、「共用炉物理プログラム基本システムの提案」、共用炉物理コードシステム特別専門委員会報告書, JNC TJ9420 2001-007 (2001).

3.1.3 ストリームプログラム構造のまとめ

ここでは、ストリームプログラムの各種形式と、本WPで抽出した、スクリプト言語の利用およびFortran90/95の利用の違いを以下に整理する。尚、付録2-2には横山委員によるストリームプログラムの評価も記載されている。

表3.1.3-1 スクリプト言語およびFortran90/95（モジュール機能）の比較

種類	スクリプト言語	Fortran90/95 (モジュール機能の利用)
利点	<ul style="list-style-type: none"> 元の計算コード、システムをそのまま利用することも可能。 ネットワーク上の利用が容易。 GUI化が容易。 WWWでの利用も可能。 	<ul style="list-style-type: none"> 元の計算コード、システムを殆ど修正する必要が無い。 Fortranだけで構築できるため、既存コードとの融合性が高い。 データの受け渡しをファイル経由だけでなく、メモリー領域経由でも可能。
欠点	<ul style="list-style-type: none"> スクリプトプログラムの習得が必要。 TEXTデータの取り扱いが中心であり、バイナリーデータの利用に工夫が必要。 データの受け渡しはファイル経由のみ。 	<ul style="list-style-type: none"> 現時点では、フリーなFortran90/95コンパイラが無い。 ネットワーク機能、GUI機能は別途附加する必要がある。

以上の様に、スクリプト言語の利用およびFortran90/95の利用にはそれぞれ特徴があり、図3.1.3-1に示すような両者の特徴を生かした複合システムが望ましいと考えられる。

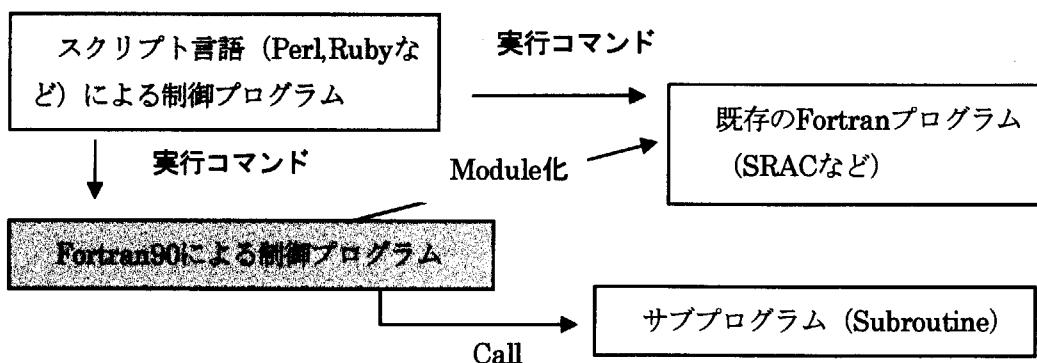


図 3.1.3-1 ストリームプログラムの構造のイメージ

3.1.4 既存コードの連携のイメージ

ここでは、前節までに述べたストリームプログラムの利用による既存コードの統合により、新たな活用が期待出来る内容を例示する。

(1) 六角形状燃料要素を持つ革新炉での炉心解析

近年、低減速炉など、軽水を利用するため、熱中性子、熱外中性子を含む一方で、六角形状の燃料要素を持つ炉心概念が検討されている。FBRは六角形状の燃料要素で構成されるが、FBR解析コードでは、熱中性子、熱外中性子の取り扱いは一般的ではない。

このため、SRACなどの熱炉の燃料要素解析コードにより、燃料要素の計算を行い、高速炉の3次元六角形状輸送コード(NSHEx)などをを利用して炉心解析を実施する事が考えられる。

(2) 热炉での断面積等の感度解析

軽水炉、ガス炉あるいは革新炉においても、解析精度の評価のため断面積等の感度解析が求められる場合があるが、このためのコード整備は十分では無い。一方、高速炉解析では高速炉用感度解析コード(SAGEP)などの利用が進んでいる。このため、SRACなどの熱炉解析コードで解析を行い、この解析結果(感度係数)を高速炉用感度解析コード(SAGEP)により感度解析を行うことが考えられる。

(3) 革新炉の核热水力炉心解析

既存の核热水力解析コードは炉型依存性が強く、そのまま低減速炉などの革新炉を解析することが困難である。一方、MVPなどのモンテカルロ法で革新炉の核特性解析は可能ではあるが、核热水力計算は困難である。

このため、革新炉の燃料要素の解析が可能なSRACなどの熱炉解析コードと軽水炉の热水力コードを組み合わせることが考えられる。

これ以外にも、炉物理コード、システムと遮蔽計算コードを組み合わせて、原子炉圧力容器の照射脆化評価の基本となる照射量を詳細に解析することなどが考えられる。

3.2 開発すべきソルバー

ストリームプログラムの利用による既存コードの統合により、新たな活用が期待出来る一方で、既存のコード、システムでは十分な解析が出来ず、新たなライブラリー、ソルバーの開発が必要である事が指摘された。ここでは、WPで議論された共通に開発すべきライブラリー、ソルバーを例示する。

(1) 標準多群ライブラリーの開発

SRACなどの熱炉解析コードではSRAC用の多群断面積ライブラリーが使用されている一方、FBRではJFSなどの利用が一般的である。一方、連続エネルギーモンテカルロ法では熱炉、高

速炉、革新炉に共通の断面積ライブラリーが使用されている、このため、熱炉、高速炉、革新炉などに共通的に利用出来る標準多群ライブラリーが有れば、核データの共通なベースでの検証などが可能になると期待出来る。

また、遮蔽計算ともライブラリーを共通する事が考えられる。遮蔽計算の原子炉の炉心解析と異なるところは、散乱の PI 次数が異なるところと、エネルギー一群の切り方が違うところぐらいであり、炉物理コードシステムに組み込むことは可能と考えられる。この際、連続エネルギー計算コードで使用する核定数発生ツールを使用して、高次の PI 散乱断面積テーブルを作成できるようにしておく。多群散乱断面積の定義は、散乱して出て行くものに着目する場合と入ってくるものに着目する場合の二種類があり、計算コードにより異なるので、変換可能にして置く必要がある。

(2) キャラクタリスティック (MOC) 法による輸送計算コード

キャラクタリスティック法など新しい計算手法による輸送計算コードは計算形状の自由度が高く、燃料要素や炉心解析への適用が期待される。新しい計算手法による輸送計算コードシステムは、海外では開発が進んでいるものの、我が国では公開の共通コードシステムは構築されていない。このため、各機関の協力により、キャラクタリスティック計算コードシステムを構築することが望まれる。

その他、3 角メッシュ、6 角メッシュの計算が高速に可能な輸送計算コードなどが候補に挙げられる。

3.3 インターフェース及び全体構成

共用炉物理コードシステムのストリームプログラムとして、インターフェース型を採用する場合でもモジュール型を採用する場合でも、計算コード、システムの計算入出力の受け渡しは必要となる。一般には、各計算コードは独自の入出力ファイルを利用する事が多く、これらの多様な入出力ファイルを繋いでいく必要がある。しかし、 n 種の入出力ファイルを相互に繋ぐためには $n \times (n-1)$ 個のインターフェースコードが必要となり現実的ではない。このため、1 つの標準フォーマットを定義し、これとの受け渡しを考えれば n 個のインターフェースコードで済む計算となる。

これまでにも、米国などで標準インターフェース・ファイルの設定 (CCCC など) が試みられてきているが、全てのデータ項目を盛り込むと、ファイル構造が複雑になり、使い勝手が悪くなる。近年、ファイル自身に構造を記述することで広範囲な種類のデータを統一的なルールで取り扱う考え方が出てきた。代表的なものは XML(eXtensible Markup Language) であり、自分自身のデータの構造を記述する DTD (Data Type Definition) 部とそれに従って記述されたデータ部分からなるテキストデータからなる言語 (ルール) を有し、様々な形式のデータの保存および活用の方法として注目を集めている。尚、XML などについては、第 2.2 節「先端計算機技術の活用」で説明がなされているので、ここではインターフェースの観点で概要を第 3.3.1 節に示す。

共用炉物理コードシステムの構築には、ファイル構造とともに、計算機能を有効に統合する機

能的なインターフェースの検討が必要である。ここでは、本WPで提案された、インターフェース及び全体構成に関する各委員の案を説明する。

池田委員は、以下の様な項目に関する共用炉物理コードシステムのルール化を行うことを提言した。

- 1) ソース、マニュアル、サンプルデータ、サンプルアウトプット、説明書、検証例（ベンチマーク問題の計算結果）、関連論文の収納とこれを記載した登録カードに関するルール
- 2) データの分類とその書式のルール
燃料データ（PWR、BWR、FBR、ガス炉）、炉心データ、運転データ
- 3) コードの機能による分類とインターフェースのルール

1. コードの機能

セル計算、実効断面積計算、スペクトル計算、体系計算（拡散、Sn輸送、モンテカルロ）、出力分布、燃焼計算、反応度係数計算、崩壊熱、放射能計算

2. インターフェースのルール

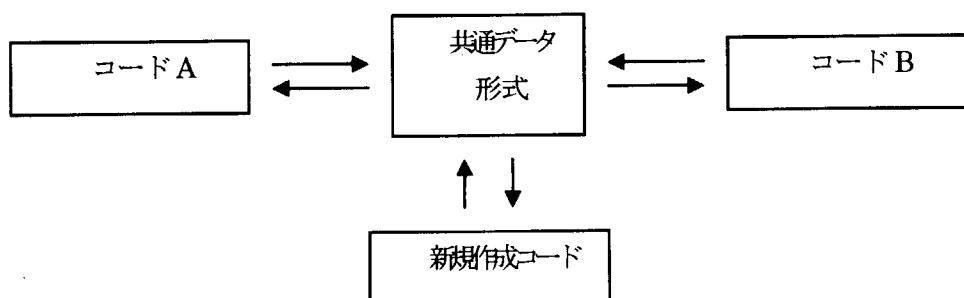
3. 断面積形式、中性子束形式のルールは CCCCに準拠する。

山岡委員は、池田委員の提案も踏まえて、共用炉物理コードシステムとして、断面積作成モジュール、中性子束計算モジュール群などを定義することにより、計算機能、インターフェースを標準化することを提案した。山岡委員の案を第3.3.2節に示す。

松村委員は、各計算コード、システムの入出力ファイルを全て引き渡すので無く、炉物理的に最低限必要な情報だけを形式を定めて引き渡すことを提案した。この様な検討の叩き台として松村委員が提案したコマンド型のコードシステム構造とインターフェースの例を第3.3.3節に示す。

3.3.1 XMLの利用案

複数プログラムを連結して動作させる際にはデータの受け渡しが必要となる。その際共用炉物理コードシステムがカバーするすべてのコードのデータ形式間の変換プログラムを作成する場合組み合わせ数が多くなり得策とはいえない。そこで、共通のデータフォーマットによる「中間データ」形式データを作成し、それを介したデータのやりとりを行う方法を考えられる。



共通データの形式として満たすべき条件を以下のように考える。

- ・ 多岐にわたる炉物理計算データを明確に分類して格納できること。
- ・ データ操作が容易であり、プログラミングインターフェースが豊富であること（データコン

データの作成が容易)

現状でこのような条件を満たす形式として XML(eXtensible Markup Language)が上げられる。XML では自分自身のデータの構造を記述する DTD (Data Type Definition) とそれに従って記述されたデータ部分からなるテキストデータからなる言語であり、様々な形式のデータの保存および活用の方法として注目を集めている。

XML データの例)

```
<!DOCTYPE Cross_sections [
  <!ELEMENT nuclide (#PCDATA)>
    <!ATTLIST nuclide number_of_group CDATA #IMPLIED>
  <!ELEMENT total (#PCDATA)+>
  <!ELEMENT absorption (#PCDATA)+>
  <!ELEMENT fission (#PCDATA)*>
  <!ELEMENT fission_spectrum (#PCDATA)*>
]>    (ここまでが DTD、以下がデータ)
<Cross_sections>
  <nuclide number_of_group="3"> Pu239 </nuclide>
  <total>671 150.7 251.3</total>
  <absorption>655.1 123.0 101</absorption>
  <fission>550 21.0 200.0</fission>
  <fission_spectrum>0.3 0.1 0.6</fission_spectrum>
</Cross_sections>
```

科学技術計算のデータの XML の適用 (拡張) については NASA の XDF (eXtensible Data Format, http://xml.gsfc.nasa.gov/XDF/XDF_home.html) などの研究もあり (Perl および JAVA によるデータ操作も可能) これらの成果も取り入れた炉物理データ格納用の XML 形式の設計を行うことも可能である。

3.3.2 機能毎のモジュール群化案

共用炉物理コードシステムとして、さまざまな議論がなされている。ここで、決定版ともいえる各種炉型に対して高精度な結果を与える汎用な決定論手法コードをゼロから開発するのはもちろん、有力なコードをベースにしても、あるいはいくつかのコードを統合していくとしても労力と予算の面から無理がある。また、決定論手法には何らかの近似が入るのは避けられず、ユーザーの要望も常に精度の高いものへ向かうため、現状での決定版を苦労して作成してもすぐに陳腐化する可能性がある。これに対して、MVP のようなモンテカルロコードはほぼ理想的なレベルになっており、事実上汎用コードはモンテカルロ以外にはないと考えられる。(統計精度の問題が残り、計算機速度に依存するとしても)

しかしながら、設計計算、空間依存動特性など決定論手法のニーズは将来も変わらないものと

考えられる。ここでは以上をふまえて、以下を目標とすることを考えた。

- ・ 我が国の軽水炉などの熱炉の核計算コードシステム、高速炉の炉心解析コードシステムの計算機能の相互利用、融合化
- ・ 種々の熱炉、高速炉、遮蔽などのライブラリーを簡単に使えるシステムの整備
- ・ 新たに開発された最新手法に基づくコードを比較的簡単に導入できるシステム
- ・ 開発労力、開発資金の最小化

既に、本 WP では「各コードがサブシステム化され、それぞれがインターフェースによりつながれ、ストリームプログラムで実行制御されるという概念」が提案されている。以下に提案するのは、本質的には同じものであるが、若干の具体化を進めたものである。但し、ストリームプログラムについては未検討である。

具体的には池田委員の提案にある、モジュール型とインターフェース型を取り入れたもので、システムは機能ごとのモジュール群から構成されている。

<機能ごとモジュール群の例>

- ・ 断面積作成モジュール群
- ・ 中性子束計算モジュール群
- ・ 線源計算モジュール群
- ・ 摂動計算モジュール群
- ・ ——モジュール群

これを図 3.8.2-1 に示す。機能ごとのモジュール群の中には、SRAC、MOSRA などの統合化コードをモジュール化（サブシステム化）したものや、各種の単独コードを含んでいる。高速炉での JUPITER などの実験解析システムのなかからのコードもモジュールとして含まれている。基本的には、各種炉型の計算に必要なだけの機能ごとモジュール群を集めた集合体がこのシステムである。例えば、

・ 断面積作成モジュール

LWR、高転換炉； SRAC

FBR ; SLAROM

ガス炉 ; VSOP

・ 中性子束計算モジュール

CITATION (DIF3D) ; 拡散計算

TWOTRAN(TWODANT), THREEDANT ; 輸送計算

などが考えられる。

新たに、外部で開発されたコードも上記モジュール群の中に取り入れていく。

本システムのために新規に開発するモジュールは基本的になしとするが、断面積作成部分など炉型に汎用なものを作成する方向も考えられる。いずれにしても、拡張性も考えると機能ごとのモジュール作成がよいと考えられる。

機能ごとモジュール群間のデータ受け渡しのために、インターフェースコードを設ける。例えば、断面積作成モジュール群と中性子束計算モジュール群の間はインターフェースコードで断面積を受け渡すが、両モジュール群の各モジュールのどのような組み合わせに対しても、受け渡し

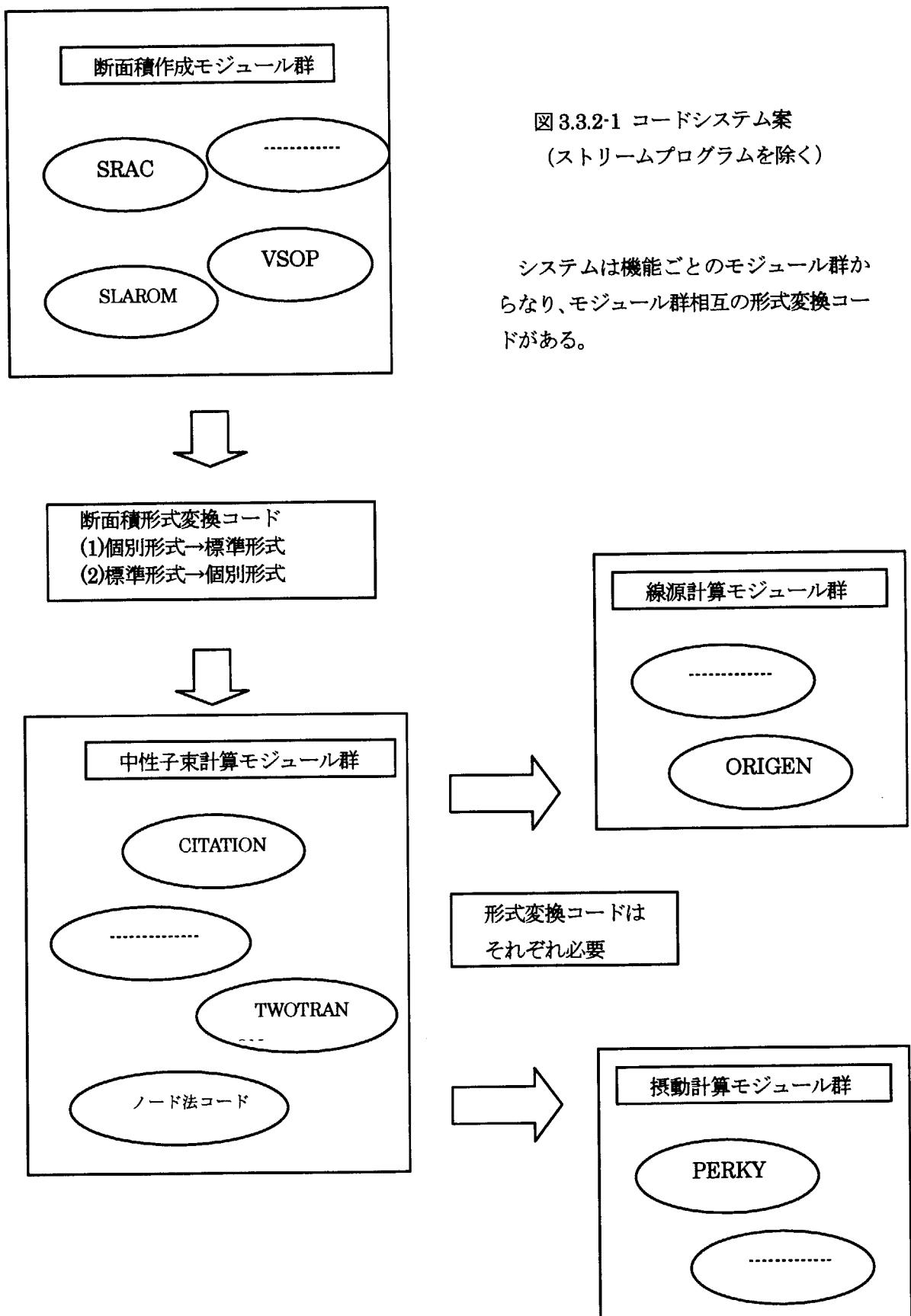
が可能なようになっている。これによって、両モジュール群から必要なモジュールをどのように選定しても、データ受け渡しが可能である。

機能ごとモジュール群をつなぐ形式としては、日本総研・佐々木氏の提案されたような共通データフォーマットを設定する。これは、個別フォーマットと共通データフォーマットの間での相互変換を行うものである。これには、熱炉、高速炉、遮蔽などを含める汎用化されたものが良い。具体的なフォーマットとして XML の使用は有効と思われるが、言語が本 WP の推奨する Fortran90 でないことと、エネルギー群数や組成数によってはファイル容量が大きくなる可能性があるので検討が必要である。

このようなシステムとすることで、新たに開発されたモジュール（コード）を取り込んでいけば、常に最新の手法を有するシステムとして保つことができる。

さらに以下のように効率的な開発が可能となる。

- 各モジュール群ごとに共通データフォーマットを設定して、これを業界標準化することにより、新たに開発されたコードで採用されていけば、フォーマット変換の手間がなくなってくるものと思われる。これには、フォーマットの十分な検討が必要となる。
本システムを業界標準化することで、モジュールごとの開発が促進され、スムースに本システムへ取り込むことができる。



3.3.3 最小データ・インターフェース案

炉心シミュレータを題材に共用炉物理コードシステムに求められる要件を検討した。共用炉物理コードシステムには以下の様な機能が必要と考えられる。

1. 2、3次元の炉心の拡散／輸送計算（ノード法など）
2. 燃焼計算機能（マクロ断面積、ミクロ断面積、核種組成計算）
3. 炉定数計算（ピンセル、燃料集合体、4燃料集合体、全炉心、その他）
4. 熱水力計算（ボイド率（BWR）、冷却材温度（PWR）、熱的制限値評価など）
5. 臨界調整機能（臨界ほう素濃度、・・）
6. 各種反応度係数計算
7. Xe トランジエント計算
8. 冷態停止反応度計算機能
9. 最適運用計算（燃料装荷、制御棒パターンなど）
10. その他

単純に、炉心の拡散／輸送計算コードや燃焼計算コードを並べただけで炉心シミュレータにならないため、目的に応じて組み上げる努力が必要と考えられる。多数の計算コードの包括的なファイルインターフェースを設計し合意する事は困難と考えられるため、必要最低限のインターフェースを決め、後で、必要に応じて個々に追加する方法が考えられる。ここでは、この検討のための叩き台として、Fortran90/95 の機能を活用したコマンド型の炉物理コードシステムの案を示す。この例では必要最低限のデータ・インターフェースはファイル経由ではなく、メモリー領域経由である（付録 2-3 参照）。

メイン・ルーチン（ストリームプログラムに相当）

サブ・プログラムの制御のみを行う。

必要最小限の情報を取り扱う。

例：物質領域数、エネルギー群、マクロ断面積、物質領域毎の多群中性子束、・・・（メッシュ毎の中性子束などは扱わない）

<プログラム例でも以下の変数しか扱わない>

```

REAL*8, ALLOCATABLE :: DD(:, :, SG(:, :, SGR(:, :, SGNF(:, :,      .. ..
*          FLXR(:, :, XXR(:, :, YYR(:, :, ZZR(:, :,      .. ..
*          DDFEW(:, :, SGFEW(:, :, SGRFEW(:, :, SGNFFE(:, :, ,
*          FLXRFE(:, :),

INTEGER, ALLOCATABLE :: MR(:, :, IXR(:, :, IYR(:, :, IZR(:, ,
*          MR1(:, :, MR2(:, :, IGF EW(:, :, MRF EW(:, :

```

<以下の様にコマンド型でサブ・プログラムの制御のみを行う>

ccccccccccccc 3D Diff.

```
if(string(1:10).eq."SUB_DIF3DE") then
```

```

call DIF3DE(IXR,IYR,IZR,XXR,YYR,ZZR,IGM,IXRM,IYRM,IZRM,
*           MRM,MR,DD,SG,SGR,SGNF,FLXR,INU,IOU)
c 3次元拡散計算を実施する。
go to 1
endif

```

現在は、以下のコマンドが含まれている。

- | | |
|------------------|---------------------------------|
| 1. DIF3D_PREP | 3次元拡散計算の準備を行う。 |
| 2. DIF1D_PREP | 1次元拡散計算の準備を行う。 |
| 3. SUB_MACROI | マクロ断面積を read する。 |
| 4. SUB_DIF3DE | 3次元拡散計算を行う。 |
| 5. RESET_DIF3DE | 3次元拡散計算で使用したメモリーを解放する。 |
| 6. SUB_DIF1DE | 1次元拡散計算を行う。 |
| 7. RESET_DIF1DE | 1次元拡散計算で使用したメモリーを解放する。 |
| 8. ESET_MACRO | 使用したマクロ断面積のメモリーを解放する。 |
| 9. XCONDENSE | 1,3次元計算結果の縮約を行う |
| 10.XCON_TO_MACRO | 縮約されたマクロ断面積の値を次の計算のマクロ断面積に移動する。 |
| 11.PRN_CON_MACRO | 縮約されたマクロ断面積の値を印刷する。 |
| 12.DEF_PUT_FILE | ファイル出力の準備を行う。 |
| 13.PUT_CON_MACRO | 縮約されたマクロ断面積の値をファイル出力する。 |

<目標としては次の様なコマンドを追加する事が考えられる>

14. 1次元輸送計算（板、円筒、球）
- 15.2 次元輸送計算（集合体計算など）
- 16.燃焼計算
- 17.マクロ断面積入力（ライブラリーから）
- 18.炉定数テーブル作成（燃焼度依存、その他）
- 19.炉定数テーブル内挿
- 20.炉心ノード法計算
- 21.熱水力計算（BWR、PWR）
- 22.各種計算結果の出力
- 23.各種計算結果のファイル出力
- 24.必要情報のファイル入力
- 25.その他

サブ・プログラム（複数・・目的別）

当該計算モジュールの全ての処理を行う。

例：メモリーの確保、入力データの読み込み、出力、ファイルの割り当て・・・

<サブ・プログラムで必要なメモリーなどを自分で確保する>

```
subroutine DIF3DE(IXR,IYR,IZR,XXR,YYR,ZZR,IGM,IXRM,IYRM,IZRM,
*                  MRM,MR,DD,SG,SGR,SGNF,FLXR,INU,IOU)
REAL*8, ALLOCATABLE :: SO(:, :, :), FLX(:, :, :), FLXO(:, :, :),
*                      CS(:), CM(:), SOO(:, :, :), XX(:), YY(:), ZZ(:)
REAL*8, ALLOCATABLE :: CX(:, :, :), C1(:, :, :), C2(:, :, :),
*                      C3(:, :, :), C4(:, :, :), C5(:, :, :), C6(:, :, :)
```

<サブ・プログラムの独立性の確保方策>

1. 既存コードをサブ・プログラム化する。モジュール化することも可能である。
(CITATION など膨大なコードは対象とせず、単機能の計算モデル、コンパクトな計算コードを中心に考える)
2. メモリーのサブ・プログラム毎の独自の確保
3. 入出力もサブ・プログラム毎に独立
4. ファイル入出力も open、close をサブ・プログラム内で実施する事で対応
5. COMMON 文の不使用。
6. その他

サブルーチン群（多数）

特定の機能を共通的に分担する。

サンプル計算の内容（付録 2・3 参照）

1. 1次元計算（3群） マクロ断面積は読み込み
2. マクロ断面積の縮約（3群→2群。計算体系平均）
3. 縮約後のマクロ断面積のファイル出力
4. 記縮約マクロ断面積を用いた3次元拡散計算（2群）
5. IAEA3次元ベンチマーク問題（2群、別途計算）

コマンド型の本炉物理コードシステム案の検討により、計算コードの計算結果全てをインターフェース化する事無く、Fortran90/95 のメモリ領域を動的に確保、開放する機能を活用することにより、他の計算に必要な最低限のデータを決まったルールで引き渡す方法についても実用化の可能性があると考えられる。

また、インターフェース・ファイル構造として、以下の様な単純なファイル構造でも高い柔軟性の達成が可能と考えられる。

WRITE(1) (I(i),I=1,8),(A(j),j=1,I(1))

ここで、I(1)はデータ（A）の数、I(2)はデータのタイプ（0；文字型、1：整数、2：実数、な

ど)、I(3)はデータの番号(マクロの拡散係数は1000など)I(4)～I(8)はデータのキー(温度、燃焼度など)を自由に設定する。付録2・3はテキスト出力であるが、上記の様なバイナリ・出力も可能である。

4. 結言

我が国の炉物理研究者がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築することを目的に、16名のメンバーで平成13、14年度の2カ年の期間で4回の会合を持ち、望ましい共用炉物理コードシステムについて検討を実施した(2003年春の日本原子力学会年会の発表内容、付録4・1、2参照)。

【調査・課題抽出】 参加機関の炉物理コード、先端計算機技術、統合型炉物理コードなどの調査を行うとともに、参加者から、既存コードシステムの問題点、希望などを調査し、①プラットフォーム依存性の無いストリームプログラム、②既存コード・データの最少の労力での利用(サブシステム化)、③既存コードでカバー出来ない計算機能を補う新規のソルバー、④コード相互、また外部コード・データとも連携が容易なインターフェース、⑤検証、ベンチマーク問題の抽出を含め、十分な検証、⑥解法・機能を記述した十分な資料の作成や対外活動、⑦維持・発展させる仕組み、の必要性が抽出された。

【コードシステムの基本概念】 コードシステムはストリームプログラム、既存プログラムのサブシステム化、必要な新規ソルバー、インターフェースからなる。ストリームプログラムの開発言語としては、先端計算機技術の活用の調査結果も踏まえ、Perl、Ruby、Pythonなどのスクリプト言語とモジュール機能を持つFortran90/95を抽出し、特徴を比較した。この結果、スクリプト言語とモジュール機能を持つFortran90/95を併用する方策を提言した。また、必要なソルバーとして、標準多群計算用ライブラリー、キャラクタリストラクチャによる輸送計算コードなどの開発課題が抽出された。

インターフェースとして拡張性の高いXML(eXtensible Markup Language)の利用やその考え方を利用した新規のインターフェース言語の設計が考えられる。また、今後の一層の検討、議論が必要であるが、計算機能のモジュール化などの提案があった。

共用炉物理コードシステムの利用としては、CD-ROMの配布やサーバーからのダウンロードにより、手元のPCやEWSで利用する方法やWWWやTelnetを用いてネットワークで利用する方法も考えられる。入力はコマンド入力の方法やメニュー選択方式が考えられる。出力もデジタルデータの他、グラフィック出力が考えられる(図4-1)。

【今後の課題】 開発のためのより具体的な手順・体制とともに、開発したコードシステムをその後、維持・発展させる仕組みの構築が必要である。

今後、より具体的な開発手順としては、システムの詳細設計を実施する必要があるが、公募などへの応募により、活動資金を得て活動することが考えられる。また、開発したシステムの検証

作業にも資金が必要である。共用炉物理コードシステムに限らず、広く原子力コードをコードシステム化する方策も考えられる。

維持・発展させる仕組みの構築は今後の課題であるが、メンテナンスと広く活用されるシステムとするために、オープンソース化を行い、ITBL(IT-Based Laboratory)計画の中の1つのタスクとして維持管理されることを提案する事も考えられる。

謝辞

本活動報告の作成に当たって、株日本総合研究所の佐々木誠氏の協力を得ました。ここに、謝辞を述べるとともに、平成15年9月に死去されました佐々木誠氏の御冥福をお祈りします。

利用環境

①自分のPC、EWSなどで

入力方法

①コマンド型で

- Use JFS3-3
- Cal Pin-cell
- Save flux ...

②メニュー方式で

- Library
- JFS
- SRAC lib
-

出力方法

- ①デジタルデータ出力で
- ②グラフ、GUI出力で

図4-1 共用炉物理コードシステムの利用のイメージ

付録 1-1 サイクル機構のコードシステムの現状

第二回共用炉物理コードの構築WP資料

2002年2月22日

改訂 2002年12月13日

サイクル機構・中性子 Gr 横山賢治

第一回WP会合で、次回会合までの準備としてあげられた項目に関して調査した。本資料に、サイクル機構の炉心解析コードの現状、欧州炉物理解析システム ERANOS のインターフェイスの特徴、共用炉物理コードに関する提案についてまとめる。

1. サイクル機構の高速炉炉心解析コードシステムの現状

サイクル機構で行っている FBR 炉心の計算法について、主に解析コードを紹介する。表 1 にサイクル機構で利用している解析コードをまとめた。必ずしも、サイクル機構で開発されたコードではないが、必要に応じて、オリジナル版から改良を加えられているものが多い。

(1) 炉定数作成

JFS-3 ライブライアリは原研により開発されたものであるが、サイクル機構でも作成できるように炉定数作成コードの整備を進めている。また、現在、「次世代炉定数」として、エネルギー群数の拡張、超微細群による格子計算を実行するためのライブラリーを作成できるように整備を進めている。

炉定数は、共用炉物理コードの基本となる入力データであるため、共用炉物理コードとしては、ENDF フォーマットから炉定数を作成する機能も含まれるべきであると考えられる。

(2) 実効断面積作成

阪大で開発された Tone の方法に基づく CASUP コード、原研で開発された SLAROM コードを主に利用している。また、SRAC の PEACO ルーチンを SLAROM に組み込んだ SLAROM-UF コードの整備も進めている。一方で、実用化戦略調査研究では、減速材を含んだ高速炉体系の検討を行うこともあり、これらの検討には、軽水炉用の SRAC コードシステムを利用することもある。

高速炉と軽水炉の中間的な中性子スペクトルを持つ炉心に対して、精度良く実効断面積を求めるができる解析システムの開発が望まれる。また、常陽、もんじゅに対しては、二重非均質効果は、1 次元のリングモデルにより精度良く解析できることが確認されているが、減速材を非均質に装荷した集合体の設計計算などのニーズもあり、2 次元格子計算の開発も望まれる。

(3) 炉心計算

2、3次元拡散計算には CITATION、2次元輸送計算には TWOTRAN、3次元輸送計算には TRITAC が用いられる。また、高速炉実機の解析には、3次元六角体系を取り扱うことができる輸送ノード法に基づく NSHEX も開発されている。

(4) 燃焼計算

CITATION に、詳細な燃焼チェーンを取り扱う機能を組み込んでおり、燃焼計算は基本的に CITAION で行う。一点炉近似が可能な場合は、ORIGEN2 も利用される。また、修正粗メッシュ法に基づく炉心設計コード MOSES の中性子束ソルバーを NSHEX に置き換えた、NSHEX-BURN も開発されている。

(5) 摂動計算

拡散理論に基づく PERKY コード（2、3次元）、及び、輸送理論に基づく SNPERT（2次元）、SNPERT-3D（3次元）コードが用いられる。状況に応じて、拡散・輸送理論、2・3次元体系が用いられる。

(6) 感度解析

静核特性に対しては、一般化摂動論に基づく SAGEP コードが用いられる。感度係数の計算には主として 2 次元体系が用いられるが、SAGEP コードは、3 次元体系も取り扱えるよう拡張が施しており、状況に応じて 3 次元体系でも感度係数の計算を行うことは可能である。燃焼核特性に対しては、SAGEP-BURN コードが用いられる。

(7) 炉定数調整計算・核設計精度評価

臨界実験や高速炉実機で得られた実験値に対して、上記の解析コードを用いて詳細な解析を行い得られた結果（C/E 値）は、感度係数を用いて、核データ（炉定数）の精度向上に反映される。このために、炉定数調整計算コード ABLE が用いられる。ABLE コードの結果は、JFSAJS コードにより処理されて最終的な統合炉定数が作成される。また、調整後の炉定数（統合炉定数）を用いた場合の高速炉核設計精度は、ACCEPT コードにより評価することができる。

(8) 共分散処理システム

炉定数調整計算では、核データの誤差（共分散）が必要である。この共分散データは、ENDF フォーマットの共分散データとして評価・整備されているが、炉定数調整計算においては、エネルギー群毎の値に処理してから利用する。このため、NJOY の ERRORR モジュールを改良した ERRORJ コードを利用する。

公開性：

ほとんどの解析コードは、公開コードを利用しておらず、必要に応じて、サイクル機構で改良・整備を行いつつ使用している。サイクル機構は、情報公開の方針として原則公開の方針を探っており、①公開コードをベースとしていること、②公開報告書が発行されており外部から引用可能であること、の2点を満たしているものについては、プログラムソースを含めて既に公開されているという取り扱いである。唯一、TIMS コードは開発元の要請により非公開となっている。

課題：

これらの解析コード群は、JOINT コードにより連結されているが、計算制御には、大型計算機上では JCL、UNIX ワークステーション上では、シェルスクリプトを用いる。これらの解析コードシステムは、大型高速炉の臨界実験である JUPITER 実験解析を主目的として開発してきたという背景から、臨界実験解析に対してはよく整備されている。実際、(2)、(3)、(5)の解析コードの連携については、JOINT コードが整備されており、解析コードを必要に応じて取り替えて、計算することはそれほど難しくない。

しかしながら、解析システム内で利用する解析コードの数は非常に多くなっており、各解析コードの連携は煩雑となっている。また、解析コードはそれぞれ独立になってはいるものの、格子計算と炉心計算でオプションを適切な組み合わせで設定しておかないと、正確な結果がでないといった状況も発生しており、システム全体を熟知（炉物理的側面はもちろん熟知する必要があるが、炉物理とは関係のないコンピュータシステム上の都合も多い）していないと間違った結果を得るといった問題もある。また、基本的に、オリジナルの解析コードの入出力形式をそのまま（変更しても一部分）利用しているため、各解析コード毎に入出力形式が異なっており、ユーザーは各解析コードの入力形式を知る必要があり、データの取り扱いが複雑で、混乱を招きやすいという問題もある。

また、実用化戦略調査研究においては、多様な炉型を対象として研究が行われるため、新たな炉心概念の設計検討が行われる場合には、既存の解析コードでは対応できないことが多い。この場合、新たな解析手法を早急に用意する必要があるが、システムの改良には、システムの全貌を熟知した管理者が多くの時間と労力をかける必要がある。

一方、解析システム内の解析コードが増えるに従い、解析コード間で同様の機能を重複して持つものが増えるという問題も存在する。例えば、格子計算コードは、ある特定の計算機能を除けばほとんど同じ機能を持つものが重複して存在する。また、燃焼計算機能は、単体で存在コードもあれば、同様の機能が、複数の炉心計算コードに重複して含まれていたり、感度解析システムの一部に含まれていたりする。解析システム内に、全く同じ機能、あるいは、少しだけ異なる機能が重複して存在すると、利用者が混乱するだけではなく、管理も難しくなる。このため、サイクル機構の解析システムでは、重複機能は極力排除して統一化していくという努力を常に払っている。しかしながら、機能を統一化することに

より、各解析コードは徐々に巨大化することになるので、この統一化作業は次第に難しくなっていく傾向がある。

共用炉物理コードが、本当にユーザーに共用してもらえるためには、やみくもに同じような機能を持つ解析コードを追加していってはユーザーの混乱を招くことになると考えられる。このため、なるべく少ない労力で、解析システム内に重複機能を持たずに済ませられるような仕組みも備えておく必要があると考えられる。

2. 欧州炉物理解析システム ERANOS の特徴について

(1) インターフェイス

LU (Langage de Utilisateur) と呼ばれる独自の入力形式を持っており、ユーザーはこの形式で解析内容を入力してシステムを利用する。ERANOS の中では、格子計算コード ECCO やノード法に基づく輸送・拡散計算コード TGV/VARIANT 等の多くの解析コードが利用可能であるが、これらはすべて、関数を呼び出すような形で利用する。例えば、TGV/VARIANT を利用するときは、以下のような形式で入力を記載する。

```
VARIANT->FLUX . . . (計算オプション) . . . ;
```

このようにすることで、FLUX (ユーザーが自由に名前を定義) という変数に中性子束の計算結果が入力される。なお、このデータを他のモジュールに渡したいときは(FLUX)のようにして括弧で囲んで使う。また、データの中身を見たい場合には専用のモジュールを使ってテキスト形式で出力させることができる。入力形式は自由フォーマットであり、ユーザーが見やすいように自由に空白や改行を入れることができる。また、計算オプションは基本的にフランス語または英語のキーワードで指定するため、入力を見ただけでも、ある程度は内容を推測できるようになっている。個人的な印象としては、原研で開発されたモンテカルロコード MVP の入力形式に似ていると感じたが、MVP の入力形式よりも更に強力な機能を持つようである。入力ファイルの中で、ユーザーは自由に変数を定義することが可能 (MVP にもシンボリックパラメータという機能があるが、ERANOS では配列も取り扱える) であり、For 文や If 文のような制御機能、更に、サブルーチン作成の機能等も有する。また、ファイル操作も入力ファイルの中から直接行うことができ、通常、シェルスクリプトで制御する部分も入力ファイルの中から制御できるようになっている。LU の名前からも分かるとおり、ERANOS のユーザーインターフェイスは、単なるカード形式の入力ではなく、プログラミング言語の体裁を取っている。

なお、ERANOS 用の入力ファイルを作成した後、計算を実行するには、コマンドプロンプトで、「eranossup (入力ファイル名)」と実行するだけで良い。

(2) 異機種間の互換性

前述のように、ERANOS 独自のユーザーインターフェイスにより、計算オプションの入力だけでなく、どの解析コードをどういう順序でつないで使うかという計算制御も行っているため、一旦、他の機種に移植してしまえば、ユーザーは全く同じ入力形式を利用することができる。このため、計算制御に、MS-DOS のバッチプログラムを利用した場合に、UNIX ではそのまま利用できないといった問題は発生しない。ただし、他の機種に移植するという作業がなくなるという訳ではなく、ERANOS は、MS-DOS やシェルのように解析コードをつなぎ合わせて実行する機能まで含んでいるだけと考えるべきである。

なお、前述の `eranossub` はシェルスクリプトで書かれているが、少なくとも、カダラッシュ研究所で利用する計算機については対応しており、どの計算機を使っているかは内部で自動的に判断してくれる（機種に応じた移植作業として、このシェルスクリプトの書き換え作業は必要ということになる）。

2. 共用炉物理コードシステムについての提案

(1) 考察

小林先生の提案されている「プログラム基本システム構築の原則」(JNC TJ9420 2001-007, p.150) では、1. 互換性、2. 移植性、3. 再利用性、4. モジュール性、5. 拡張性、をそれぞれ確保するということであるが、これらは、いわゆる、オブジェクト指向技術で得られる品質であると考えられる。これらの品質を、オブジェクト指向プログラミング言語ではない、MS-DOS バッチ言語や UNIX のシェルスクリプト言語、Fortran の組み合わせだけで、実現しようとするのは難しいのではないかと思われる。

一方、6. ジョブプログラムの記述が簡単、7. モジュールの選択による幅広い利用者の要求を満たす、という要件に関しては、ERANOS に見られるような、ユーザーインターフェイスを一種のプログラミング言語とするという方法は有効と思われる。MS-DOS のバッチ処理もプログラミング言語の一種であるので、小林先生の提案も、基本的に同様のアプローチと思われる。また、決定論的手法ではないが、MVP は比較的洗練されたユーザーインターフェイスを持っており、多くのユーザーを獲得した（共用された）要因になっているのではないかと思われる（フリーフォーマットによるプログラミング言語的な入力形式、グラフィカルユーザーインターフェイスによる入力体系の確認等）。原子力以外の分野では、数式処理システムである Mathematica や Maple、行列計算システムである Matlab (Octave⁽¹⁾)、統計処理システムである S (R⁽²⁾) 言語等、科学技術計算に関連する汎用システムなどでは、ユーザーインターフェイスは、一種のプログラミング言語となっているものも多い。

(2)提案

共用炉物理解析コードシステムの共通のユーザーインターフェイスとしては、プログラミング言語のような形式をとるのが有効ではないかと考えられる。しかしながら、ERANOS のように独自のユーザーインターフェイスを新たに開発するのは膨大な労力と予算が必要であることは容易に想像できる。

一方、小林先生も、低級言語の使用を避けて高級言語で書くべきであるといったことを提案されておられるが、Fortran 等の高級言語よりも更に高級という意味で、超高級言語 (VHLL : Very High Level Language、第4世代言語、高水準言語等とも呼ばれる) と呼ばれることもあるプログラミング言語 (スクリプト言語) が存在するようである。小林先生の提案の延長として、また、原則1～5を実現させるためにも、共用炉物理コードシステムのユーザーインターフェイスとして、これらを利用できないか検討することを提案する。

これらのスクリプト言語としては、Tcl⁽³⁾、Perl⁽⁴⁾、Python⁽⁵⁾、Ruby⁽⁶⁾等がある。なお、これらのスクリプト言語の多くは、オブジェクト指向言語でもあり、うまく利用できれば、システムの原則1～5の実現にも貢献できるのではないかと考えられる。また、これらの多くはフリーウェアであり、商用ソフトに組み込むことを許可しているものもあるようであるので、共用炉物理コードシステムに組み込んで、配布しても法的に問題がないものもあると思われる (詳細な調査は別途必要と思われる)。また、UNIX、Windows、Macintosh 等、有名なOSには既に移植されているようであり、互換性の問題も解決される可能性がある。

共用炉物理コードシステムの課題は、原子炉物理の課題もさることながら、ソフトウェア設計やプログラミング技術に対して、解決すべき大きな課題があるのではないかと考えられる。システムの原則1～5は、単にオブジェクト指向プログラミング言語を採用するだけで実現できるような簡単な問題ではないと想像される。どのようなユーザーインターフェイスを採用するかよりも、解析コード間でいかにうまくデータ形式共通化して定義できるかが、共用炉物理コードの設計を成功させる鍵と思われる。ソフトウェア設計やオブジェクト指向技術の専門家の力を借りずに、原則1～5を達成できるような共通のデータ形式を定義して、巨大な解析システムを設計するのは難しいのではないかと思われる。

また、ここで提案したスクリプト言語は必ずしもメジャーとはいはず、開発途上のものもあるので、こういった言語を共用システムのユーザーインターフェイスに用いるのはリスクが大きいといった問題点もあると思われる。しかしながら、解析スキームを定義するためのユーザーインターフェイスとしては、何らかのプログラミング言語的なものが必要であることは間違いないと思われる。ここにあげたものは、候補のいくつかであり、他にもっと良いものがないか、あるいは、他の科学技術計算の分野で前例がないか、調査・検討することを提案する。

<参考情報>

インターネット上に関連すると思われる研究や試みが散見されるので、以下にいくつか紹介する。

○数値計算への応用

・ Python :

Numerical Python⁽⁷⁾ : C 言語による配列計算の高速化 (LLNL)

PyFort⁽⁸⁾ : Fortran との結合 (LLNL)

・ Ruby :

Narray⁽⁹⁾ : C 言語による配列計算の高速化 (京都大学&宇宙科学研究所)

○特定の科学分野への応用

Bioruby⁽¹⁰⁾、Bioperl、Biopython 等 : 生物学 (日本バイオインフォマティック学会)

Ruby 可視化、数値計算ライブラリー⁽¹¹⁾ : 地球流体科学 (京都大学&宇宙科学研究所)

<参考 URL>

- (1) <http://www.octave.org/>
- (2) <http://www.r-project.org/>
- (3) <http://tcl.activestate.com/>
- (4) <http://www.perl.org/>
- (5) <http://www.python.org/>, <http://www.python.jp/>
- (6) <http://www.ruby-lang.org/>
- (7) <http://sourceforge.net/projects/numpy>
- (8) <http://sourceforge.net/projects/pyfortran>
- (9) <http://www.ir.isas.ac.jp/~masa/ruby/>
- (10) <http://bioruby.org/index.html.ja>
- (11) http://www.kurasc.kyoto-u.ac.jp/radar-group/tsudal/kenkyu_visu.html

表1 サイクル機構で利用、開発・整備している高速炉核特性解析コード

分類	解析コード	公開の有無
炉定数作成	ENDF/B 形式核データ処理コード NJOY	公開
	断面積データの線型化コード LINEAR	公開
	共振パラメータを考慮した断面積計算コード RECENT	公開
	ドップラー効果を考慮した断面積計算コード SIGMA1	公開
	共鳴重核種の群定数作成コード TIMS	非公開
実効断面積作成	高速炉用セル格子計算コード CASUP	公開
	高速炉用セル格子計算コード SLAROM	公開
	汎用超微細群スペクトル計算モジュール PEACO 内蔵格子計算コード SLAROM-UF	公開
炉心計算（燃焼計算含む）	2・3次元拡散計算コード CITATION (燃焼計算機能を含む)	公開
	2次元輸送計算コード TWOTRAN-II	公開
	2次元輸送計算コード TWODANT	公開
	3次元輸送計算コード TRITAC	公開
	3次元輸送ノード法計算コード NSHEX (燃焼計算機能を持つ NSHEX-BURN)	公開
擾動計算	2・3次元拡散擾動計算コード PERKY	公開
	2次元輸送擾動計算コード SNPERT	公開
	3次元輸送擾動計算コード SNPERT-3D	公開
感度解析	一般化擾動論に基づく 2・3次元感度解析コード SAGEP	公開
	燃焼核特性感度解析コードシステム SAGEP-BURN	公開
燃焼計算	燃焼計算コード ORIGEN	公開
炉定数調整・核設計精度評価	一般化最小自乗法に基づく炉定数調整コード ABLE	公開
	実機核特性設計予測精度評価コード ACCEPT	公開
	共分散データ処理コード ERRORJ	公開
その他	高速炉核特性解析コードインターフェイスプログラム JOINT	公開
	2・3次元反応率分布計算コード LAGOON	公開
	連続エネルギーモンテカルロコード MVP	公開
	汎用核計算コード SRAC95	公開

付録 1 - 2 - 1 原研のコードシステムの現状

共用炉物理コードシステムの構築WP資料

2002.2.22

原研 飯島 進

原研の計算コードシステム

(1) 熱中性子炉の計算コードシステム (SRAC システム)

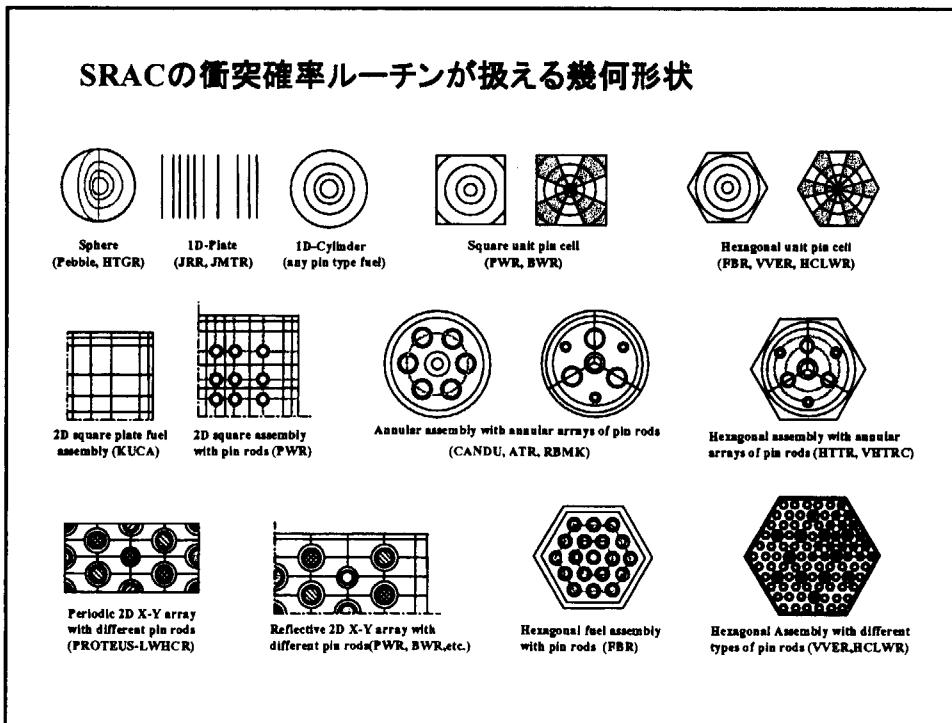
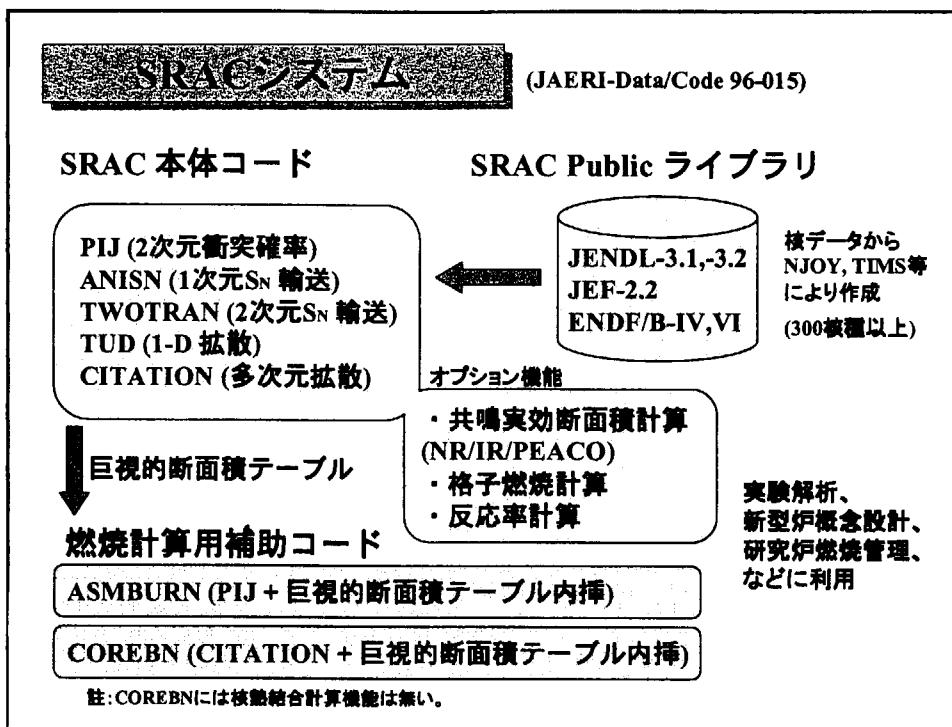
(2) ガス炉計算コードシステム

(NDCS; Nuclear Design Code System)

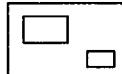
(3) 高速炉核特性計算コードシステム (EXPARAM)

(4) 臨界安全 (特別にコードシステムとよぶものはない)

SRAC システムに輸送コードとして DANTSYS を使用



その他の要素コードが扱える幾何形状

S_N 輸送		拡散	
ANISN(3)	TWOTRAN(3)	TUD(3)	CITATION(12) / COREBN(12)
 X (Slab)	 X-Y (2D-Slab)	 X (Slab)	X (1D-Slab) R (1D Cylinder) Rs (Sphere)
 R (Cylinder)	 R-Z (2D-Cylinder)	 R (Cylinder)	X-Y (2D-Slab) R-Z (2D-Cylinder) R-θ (2D-Circle) Tx-Ty (2D-Triangular) Hx-Hy (2D-Hexagonal)
 Rs (Sphere)	 R-θ (2D-Circle)	 Rs (Sphere)	X-Y-Z (3D-Slab) Tx-Ty-Z (3D-Triangular) Hx-Hy-Z (3D-Hexagonal) R-θ-Z (3D-Cylinder)

MVP/GMVPシステム

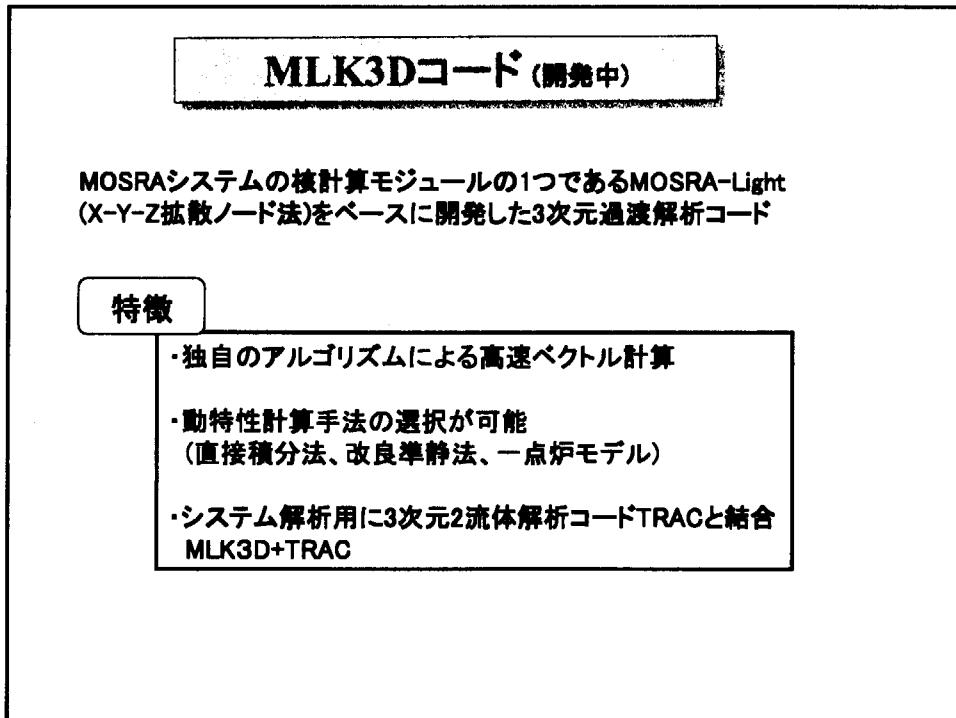
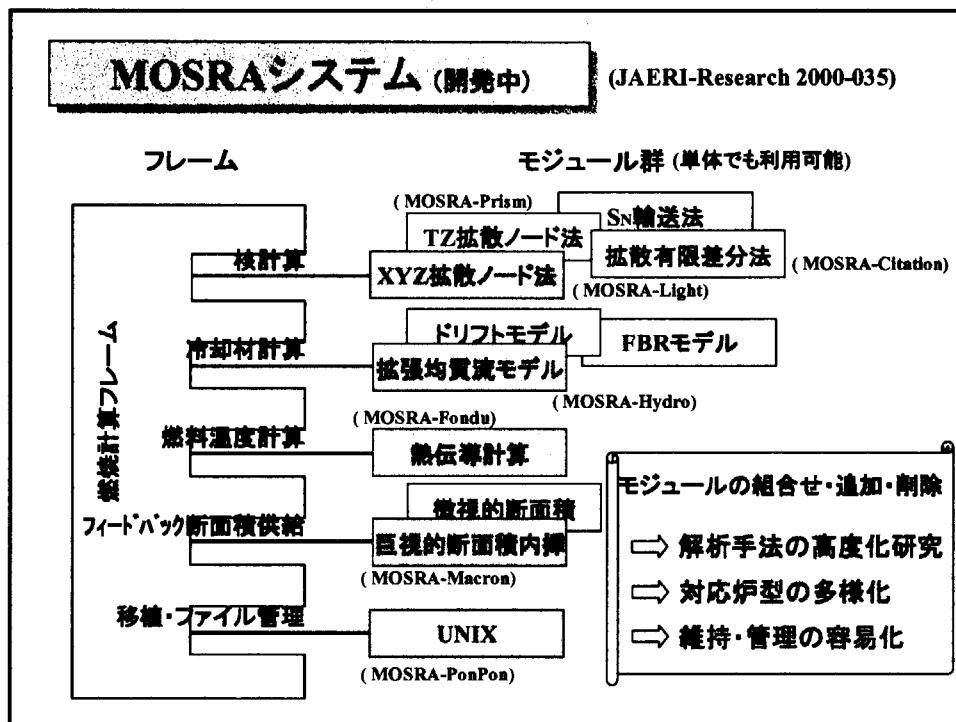
(JAERI-Data/Code 94-007)
<http://gaia.tokai.jaeri.go.jp>

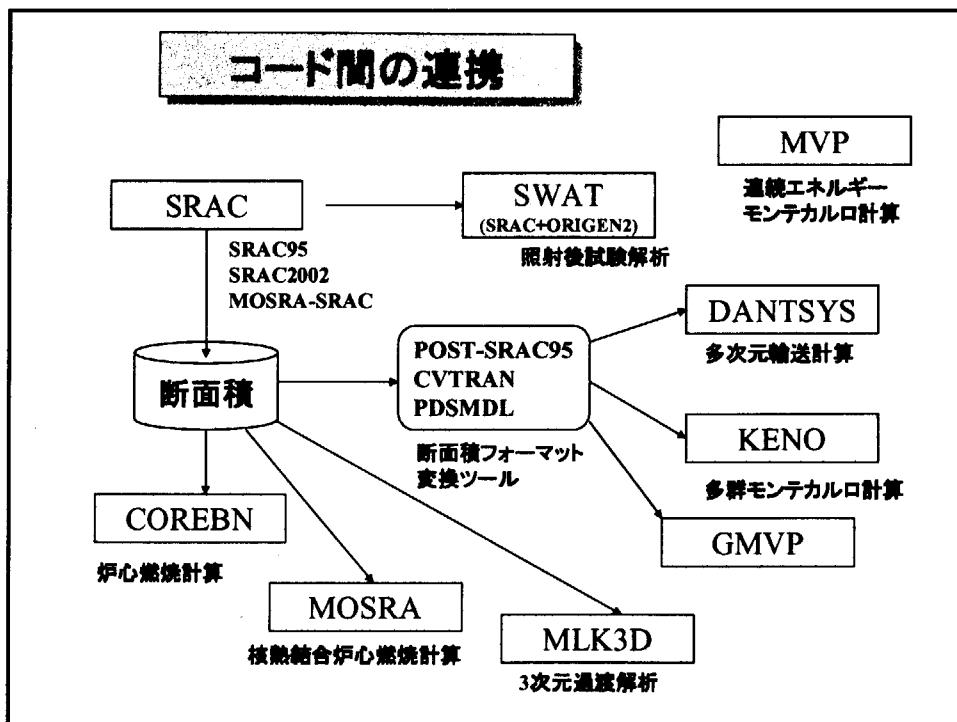
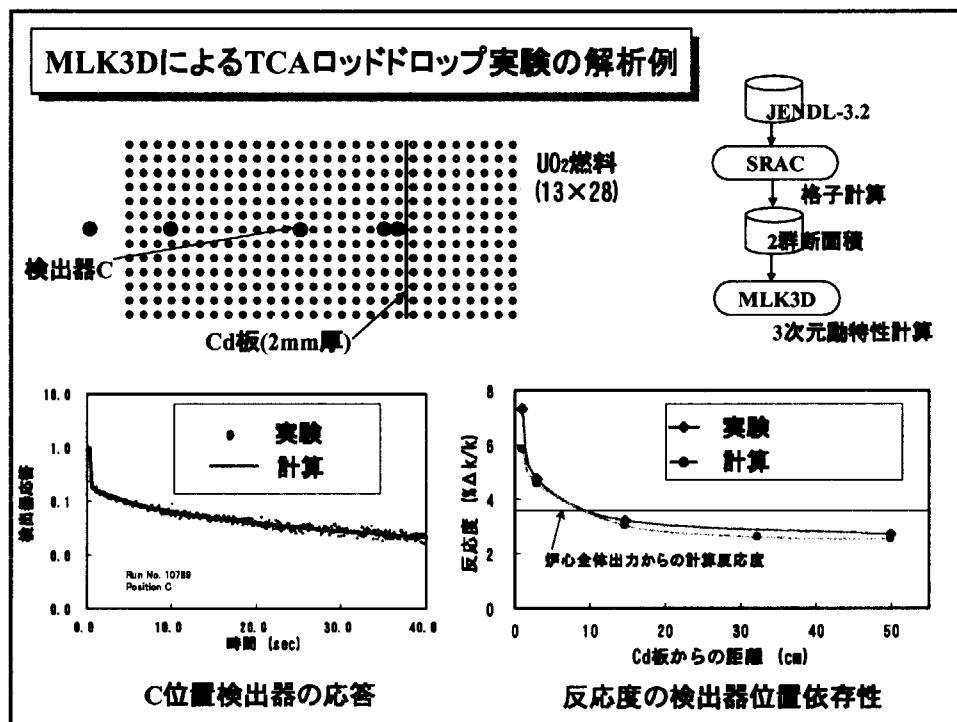
汎用中性子・光子輸送計算モンテカルロコード

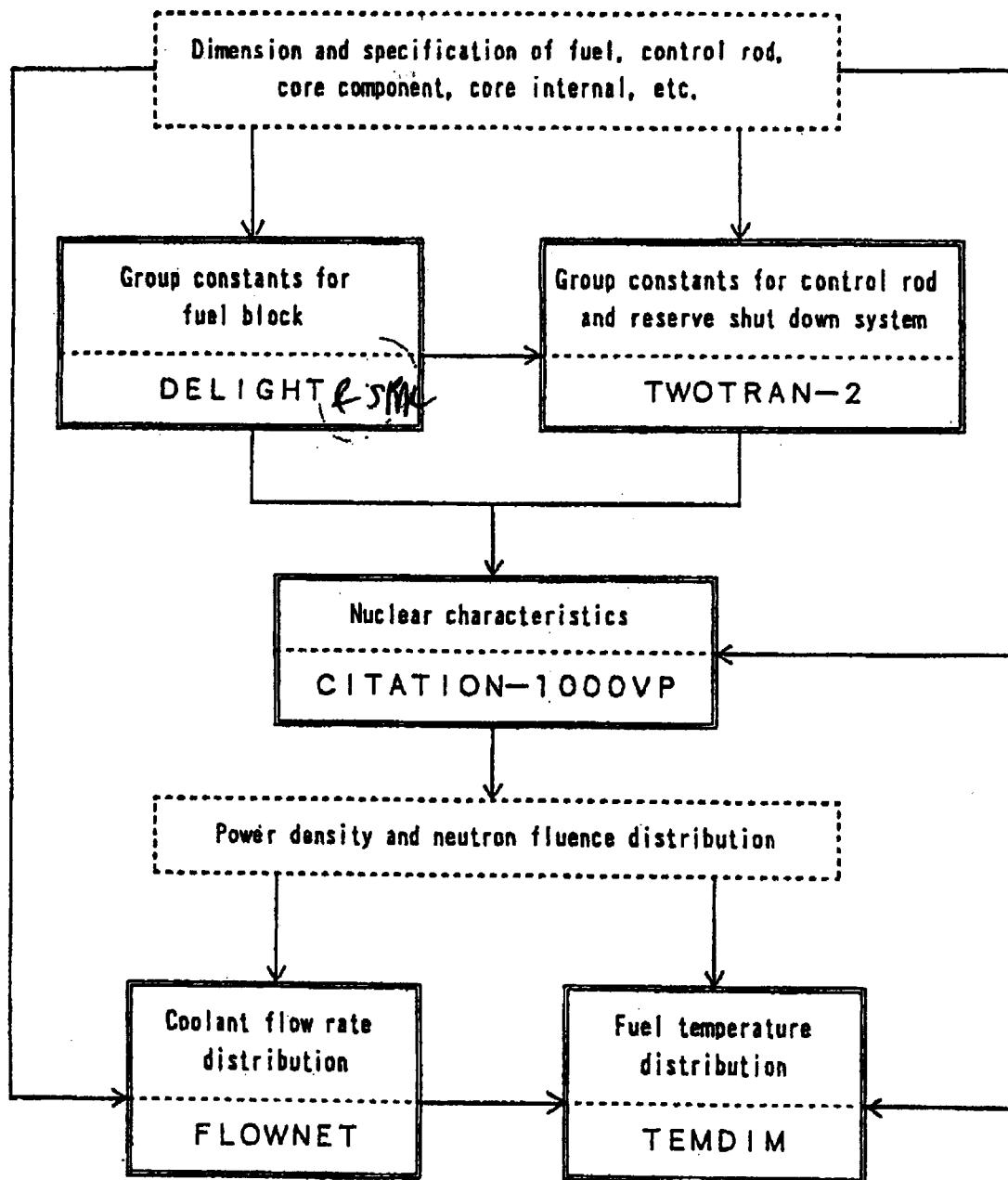
特徴

- ・高速ベクトル計算または並列計算(MPI/PVM)
- ・連続エネルギー法(MVP)と多群法(GMVP)
- ・任意温度の連続エネルギーMVPライブラリ生成
(ARTユーティリティまたはMVPの自動ライブラリ作成機能)
- ・大量の被覆燃料粒子を確率論的幾何形状モデルで表現
- ・燃焼計算(MVP-BURN)

原理的には(計算資源が十分であれば)、10億個の被覆燃料粒子を含む炉心を制御棒を駆動させながら温度変化を考慮して並列ベクトル計算で燃焼させることができる。







1. Program structure of the NDCS and thermal-hydraulic codes (FLOWNET and TEMDIM).

ENCE AND ENGINEERING VOL. 122 FEB. 1996

NDCS (HTTR) 計算コードシステム

(1) 燃料体、照射試料および反射体の少数组定数の作成

DELIGHT; one-dimensional lattice burnup cell calculation code

高温ガス炉用燃料格子特性解析計算コード

SRAC を一部併用

(2) 炉心特性計算

CITATION-1000VP

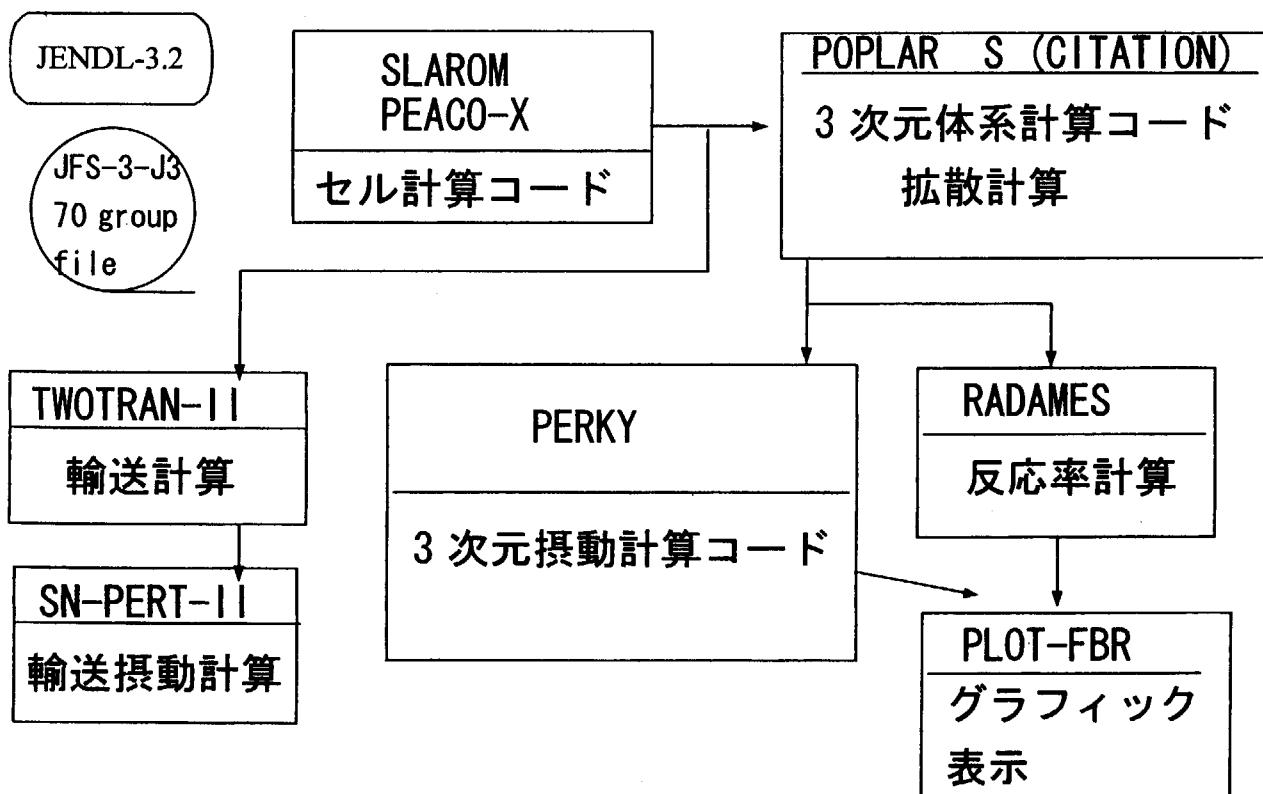
燃焼度の異なる領域が多数存在する体系計算に対応するため、ゾーン数を拡大

文献

- 1) K. Yamashita, et al.; Nuclear Design of the High-Temperature engineering Test Reactor (HTTR)
Nuclear science and engineering 122 (1996)
- 2) 山下 清信、他；高温工学試験研究炉（HTTR）の臨界試験、(I)
日本原子力学会誌 Vol.42, No. 1, 2000

付録 1-2-2 : 原研（高速炉）のコードシステムの現状

高速炉核特性計算コードシステム EXPARAM (原研)



高速炉核特性標準計算コードシステム

計算コード	機能	公開
SLAROM	衝突確率法、セル計算コード	○
PEACO-X	超詳細群衝突確率法 セル計算コード	○
POPLARS	3次元拡散計算コード (CITATION-FBR)	○
PERKY	3次元拡散・摂動計算コード	○
TWOTRAN-II	2次元輸送計算コード	○
SN-PERT-II	2次元輸送、摂動計算コード	○
DANTSYS	3次元輸送計算コード	
SN-PERT3D	3次元輸送摂動計算コード	

付録 1－3：原燃工のコードシステムの現状

平成 14 年 2 月 22 日
原子燃料工業株式会社

原燃工の炉物理コードの現状について

1. 序

PWR 炉心設計業務で使用している炉物理コードの現状について説明する。

2. 使用計算コード一覧

表 1 に使用している炉物理コードを示す。臨界・遮蔽計算には公開コードを使っているが、PWR 炉心設計で使用しているコードはいずれも許認可コードや商業ベースのコードであり、現時点では公開の予定はない。

3. 必要とするコード

PWR 燃料の多様化、並びに客先の経済性追求の要望に応えるために、より高次の計算理論に基づいた近似の少ない、そして計算環境の進歩に沿って大容量・高速処理の機能を有する計算コードを望んでいる。

—以上—

表1 使用している炉物理計算コード一覧

用途	システム名称	コード名	概要	公開性
炉心設計	改良 NULIF	NULIF	MUFT-SOFOCATE 型輸送計算コード。(高速群 54 群、熱群 40 群)	公開の予定なし
		GDLUX	ライブラリは ENDF/B から作成。	
	ASY5G		エネルギー 5 群、拡散理論に基づく 2 次元集合体燃焼計算コード。	
	SHARP-XY		エネルギー 2 群、拡散理論に基づく 2 次元体系（径方向）の炉心計算コード。	
	SHARP-A		エネルギー 2 群、拡散理論に基づく 1 次元体系（軸方向）の炉心計算コード。	
	CASMO/	CASMO4	エネルギー多群、輸送理論に基づく集合体燃焼計算コード。	
		SIMULATE-3	エネルギー 2 群、拡散ノード法に基づく 3 次元体系の炉心計算コード。	
		INSIGHT	連続する多数サイクルにわたって燃料運用の最適化を行なうツール。	
(次世代用)	GALLOP		単一サイクルの燃料運用の最適化を行なうツール。	ORNL より入手可能
		OPAL	平衡サイクルの燃料運用の最適化を行なうツール。	
	SCOPE		エネルギー多群、輸送理論に基づく 3 次元詳細メッシュ計算コードとして開発中。	
	SCALE		多群モンテカルロ計算コード。臨界安全評価に使用。	
臨界・遮蔽	ORIGEN		燃焼計算コード。ソースターム計算に使用。	RIST から入手可能

付録 1-4 : 電中研のコードシステムの現状

共用炉物理コードシステムの構築WP資料（案）

2001年12月

(財)電力中央研究所 松村 哲夫

1. 電中研における軽水炉炉心解析コードシステムの現状

電中研では軽水炉炉心解析コードシステムと FBR 炉心解析コードシステムを整備している。この内、報告者が把握している軽水炉炉心解析コードシステムについて紹介する。

(1) 炉定数関係

原研で開発した JACS システムの多群定数ライブリー (MGCL) を主に使用している。原研では臨界安全解析などに利用しているが、電中研では軽水炉炉心解析用に、ライブリーのフォーマットを変更して使用している¹⁾。また、原研で作成された MGCL ライブリーを利用する他、NJOY システムなどを用いて、JENDL などから独自に MGCL ライブリーを作成するシステムもあり、必要に応じてライブラリーを作成している。現在使用している MGCL は JENDL-3.2 から作成し、エネルギー群数は 26 群である²⁾。

MGCL ライブリーは Bondarenko 型の多群定数であり、自己遮蔽因子 (f-Table) の内挿により実効断面積を作成する。また、²³⁸U の相互遮蔽因子には G テーブルを。温度因子は、293K, 560K, 900K を基準点とした内挿法による F テーブルで対応する。

アクチニドは 18 核種、FP は 45 核種、その他の FP は 1 つにまとめた仮想核種 (lumped FP) からなる燃焼チェインを利用する。

[文献]

- 1) 松村哲夫、他：「軽水炉用多群定数ライブラリーの開発」、電力中央研究所報告 279065 (1980)。
- 2) 小室雄一、他：「臨界計算用多群定数ライブラリー MGCL-J3 の作成と検証」、JAERI-M 93-190 (1993)。

(2) 燃料集合体の中性子輸送および燃焼計算コード

燃料集合体の解析を目的に、独自に、2次元の輸送計算コード（FLEXBURN）を開発・利用している³⁾。

・ 計算手法の概要

任意の凸四角形を計算メッシュに用い、discrete ordinates 法（Sn 法）に基づき、各計算メッシュを Sn 角度方向ごとの中性子が透過、漏洩する確率を用い、中性子輸送計算を2次元で行う。

燃料集合体の計算には燃料棒セルなどの取り扱いは不要で、体系を表現する計算メッシュと燃焼領域を細かくすることにより、燃料ペレット内の詳細な燃焼計算が可能である。

中性子輸送計算により多群の中性子束を算出した後、1群に縮約した実効断面積を用い、燃焼計算を Matrix exponential 法（ORIGEN2 型）⁴⁾により行う。

[文献]

- 3) 亀山高範、他；"Neutron transport code FLEXBURN by Sn method with transmission probabilities in arbitrary square meshes for light water reactor fuel assemblies," Nuclear Science and Engineering, 123, 86 (1996).
- 4) A. G. CROFF, "ORIGEN2: a versatile computer code for calculating the nuclide compositions and characteristics of nuclear materials," Nuclear technology 62 (1983).

(3) 炉心計算コード

PWR の炉心特性解析を目的に、独自に、3次元の炉心シミュレータ（COMPWR2）を開発・利用している。

・ 計算手法の概要

X Y の2次元の拡散計算と軸方向の1次元の拡散計算を組み合わせ、高速群、熱群の2群の断面積を用いる3次元の炉心シミュレータを構成している。熱水力計算機能、臨界ほう素濃度計算機能、燃料集合体交換機能などを有し、電力会社で使用する一連の炉心特性解析の機能を有している。

(4) 詳細計算コード

上記の計算コードの他、原研で開発された連続エネルギー・モンテカルロコードMVPに燃焼計算機能を付加したMVP-BURNの作成・利用、MCNPの整備・利用を行っている。

公開性：(2)、(3)に示したコードは開発費用の観点から公開性に制約があり、提供については個別の対応となっている。

安全解析所における炉心解析コードの現状と今後必要とするコード

付録 1-5-1: 安全解析所のコードシステムの現状

コード	対象	核計算モデル		コードの制約、課題等	今後必要とするコード
		解析手法	核定数エンネルギー群数		
CASMO-4 静特性	燃料 集合体	二次元輸送計算 (Characteristics 法)	<ul style="list-style-type: none"> ENDFB-IV (40 or 70 群) 16~8 群輸送計算 集合体平均 2 群核定数、2 群の中性子束不連続因子の計算 	<ul style="list-style-type: none"> 重核種: ● 18 核種: ● FP 核種: ● 29 核種 +2 疑似核種 一方、中性子束不連続因子は 2 群までである。 したがって、近代ノード法で多群の炉心拡散計算を実行するのに必要な核定数が作成できない。 	<p>[必要性]</p> <ul style="list-style-type: none"> 現在、高燃焼度炉心や MOX 炉心の解析を 2 群で行っているが、今後、多群で実施し、エンネルギー群数の影響を把握しておく必要があると思われる。 近代ノード法で多群拡散計算ができるコードとして、原研の MOSRA-light コード（非公開）や 安解 所の SKETCH-INS コードがある。 しかしながら、現時点では、公開コードで多群の中性子束不連続因子を作成するコードはない。 <p>[必要なコード]</p> <ul style="list-style-type: none"> 近代ノード法で多群の炉心拡散計算を実行するのに必要な多群の集合体平均核定数と中性子束不連続因子を作成できる集合体核計算コードが必要である。
			<ul style="list-style-type: none"> 三次元常拡散計算 (近代拡散ノード法) 	<ul style="list-style-type: none"> 集合体単位の 2 群核定数 2 群の中性子束不連続因子 	<ul style="list-style-type: none"> 集合体平均核定数のエンネルギー群数は 2 群で固定されており、多群での三次元定常核熱水力計算ができない。
SIMULATE-3 動特性	炉心				<ul style="list-style-type: none"> コードは近代ノード法により多群の動特性計算が可能であるが、多群の核定数と中性子束不連続因子を算出するコードがない。
SKETCH-INS 動特性	炉心	三次元動特性計算 (近代拡散ノード法)	<ul style="list-style-type: none"> 集合体単位の多群核定数(通常 2 群) 多群の中性子束不連続因子(通常 2 群) 	-	

付録1-5-2 原子力安全解析所のコードシステムの現状

CRPC-2000(2)-

第2回 共用炉物理コードシステム特別専門委員会



原子力安全解析所における 炉心解析コードについて

平成12年10月23日

原子力安全解析所
中島 鐵雄

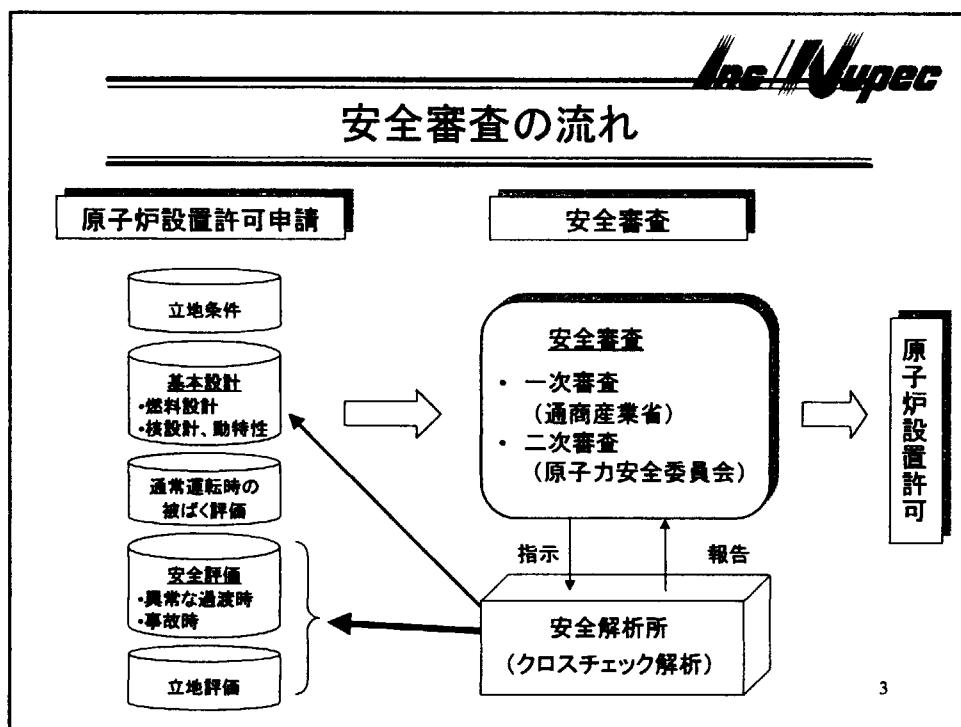
1



報告内容

- 安全審査の流れ
- 安全解析所における炉心解析コード
- 炉心核特性解析コードCASMO-4/SIMULATE-3
- 反応度事故三次元解析コードEUREKA-JINS/S
- 炉心安定性解析コードSKETCH-INS/TRAC-BF1

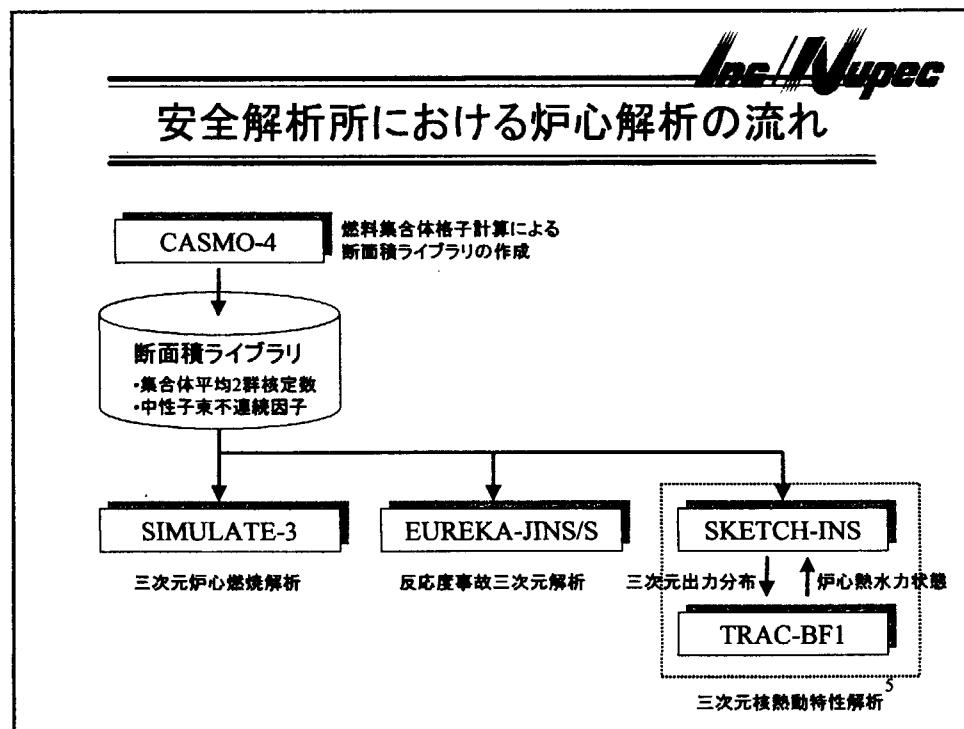
2



Inc Nupec

安全解析所における炉心解析コード

コード			核計算モデル				熱水力モデル
			対象	解析手法	核断面積	燃焼計算	
静特性	通常時	CASMO-4	燃料集合体	二次元輸送計算 (Characteristics 法)	<ul style="list-style-type: none"> • ENDF/B-IV (40or70 群) • 16~8 群輸送計算 • 集合体平均 2 群核定数、中性子束不連続因子の計算 	<ul style="list-style-type: none"> • 重核種: 18 核種 • FP 核種: 29 核種 +2 模似核種 	-
		SIMULATE-3	炉心	三次元拡散計算 (近代ノード法)	<ul style="list-style-type: none"> • 集合体単位の 2 群核定数 • 中性子束不連続因子 	(燃焼履歴追跡 出力分布計算)	
動特性	安定性	SKETCH-INS ↓↑ PVM TRAC-BF1	炉心	三次元拡散動特性計算 (近代ノード法)	<ul style="list-style-type: none"> • 集合体単位の 2 群核定数 • 中性子束不連続因子 	-	-
		TRAC-BF1	プラント	-	-	-	2流体モデル
事故時		EUREKA-JINS/S	炉心	<ul style="list-style-type: none"> • 準静近似の一点炉動特性計算 • 三次元拡散計算 (CITATION) 	<ul style="list-style-type: none"> • メッシュ単位の 2 群核定数 	-	均質流モデル
						4	



NEC/Nupec

最近のクロスチェック解析の実績(炉心)

分類	事象	解析対象炉			使用コード
		平成 10 年度	平成 11 年度	平成 12 年度	
基本設計	炉心安定性			大間フル MOX-ABWR	SKETCH-INS/TRAC-BF1
安全評価	制御棒飛出し	高浜 3,4 号 部分 MOX PWR			EUREKA-JINS/S
	制御棒落す	福島第一 3 号 部分 MOX BWR		大間フル MOX-ABWR	EUREKA-JINS/S

6



CASMO-4/SIMULATE-3

- 米国のスタズピック・オブ・アメリカ社で開発された炉心解析コードであり、世界各国で多数の機関(50以上)が導入しており、世界的評価の高いコードである。
- コードの導入
 - 平成3年度に炉心核特性解析コードCASMO-3/SIMULATE-3を購入。
 - 平成7年度に集合体内の核的非均質性をより厳密に扱うCASMO-4を購入。
 - 平成11年度にCASMO-4/SIMULATE-3の最新バージョンを購入。
- CASMO-4の検証解析
 - VIP-BWR臨界試験データ(部分MOX/BWR炉心)
 - EPICURE臨界試験データ(部分MOX/PWR炉心)
 - MISTRAL臨界試験データ(全MOX/PWR炉心)
- SIMULATE-3の検証解析
 - Peach-Bottom 2号炉の運転データ(BWR)
 - Tihange 2号炉の運転データ(部分MOX/PWR)

7



CASMO-4/SIMULATE-3の検証解析

コード	実験分類	計画名/施設	対象炉心	実測データ	検証状況
CASMO-4	臨界実験	VIP-BWR 計画 * (Belgonucleaire)	部分 MOX 炉心 (BWR 体系)	<ul style="list-style-type: none"> • k-eff • 炉心中性子束分布 • 反応度係数 	終了
		EPICURE 計画 * (CEA)	部分 MOX 炉心 (PWR 体系)	<ul style="list-style-type: none"> • k-eff • 炉心中性子束分布 • 反応度係数 	終了
		MISTRAL 計画 (CEA/NUPEC)	全 MOX 炉心 (PWR 体系)	<ul style="list-style-type: none"> • 制御棒値 • 遷発中性子割合 	実施中
	照射後試験	BASALA 計画 (CEA/NUPEC)	全 MOX 炉心 (BWR 体系)	<ul style="list-style-type: none"> • スペクトルインデックス • 中性子吸収材値 	未着手
SIMULATE-3	実機運転データ	全炉心混合 酸化物燃料 照射実証試験 (NUPEC)	<ul style="list-style-type: none"> • 既存ウラン炉心 • 先行 BWR1/3MOX 炉心 • 先行 PWR1/3MOX 炉心 • 大間 ABWR-MOX 炉心 	<ul style="list-style-type: none"> • 燃焼度、燃焼度分布 • 燃焼に伴う組成変化 • 軸方向出力分布 • ベレット内出力分布 	未着手
		Peach-Bottom 2 (米国)	BWR (ウラン炉心)	<ul style="list-style-type: none"> • k-eff バイアス • 炉心中性子束分布 	終了
	Tihange 2 * (ベルギー)	PWR (部分 MOX 炉心)	<ul style="list-style-type: none"> • 燃焼度、燃焼度分布 • 隣界モウ素濃度 	終了	

*: 実験データを購入



EUREKA-JINS/Sの概要

- 原研作成のEUREKA-SPACEを導入し、一部を改良してEUREKA-JINS/Sを作成した。
- EUREKA-JINS/Sの特徴
 - 一点炉動特性モデル(炉心出力)
 - 三次元2群中性子拡散モデル(炉心出力分布)
 - ドップラ、減速材反応度フィードバックを考慮
 - 軸方向一次元チャンネル熱流動モデル
 - 燃料棒内径方向一次元熱伝導モデル
- 主なモデルの改良及び追加
 - RIA評価指針で指定された熱水力相關式の追加
 - 3次元擾動論に基づく反応度計算モデルの追加
 - 核計算及び熱水力計算における領域分割数の拡張

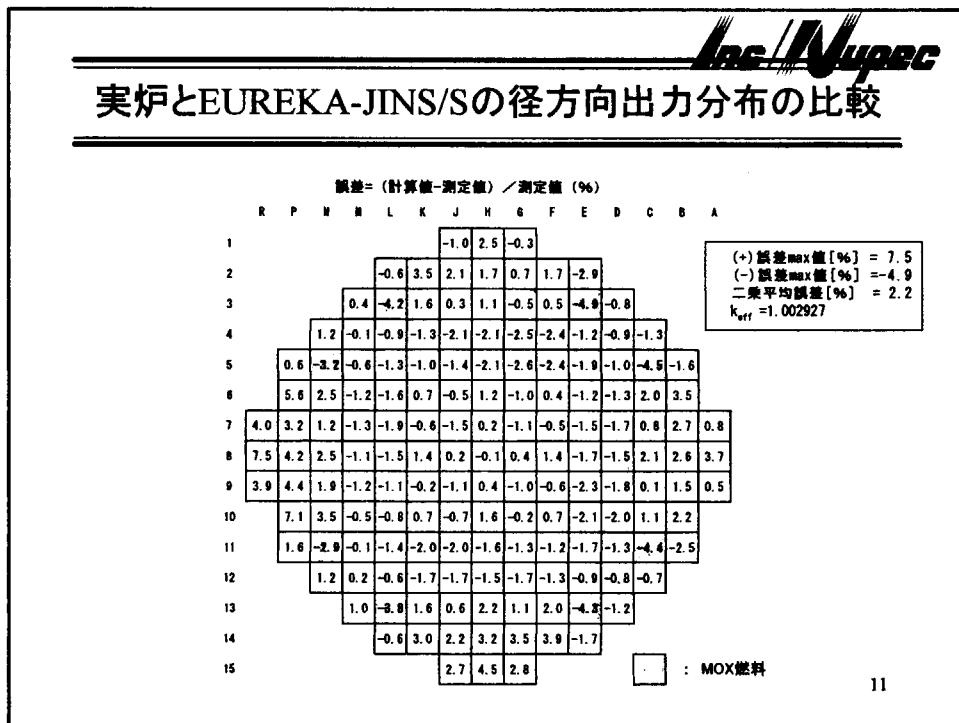
9



EUREKA-JINS/Sの検証解析

- 静特性評価 : Tihange 2号炉運転データとの比較
 - 3ループPWR(17×17型燃料集合体)プラント
 - MOX燃料を8体装荷したサイクル12の末期
 - MOX燃料のPu富化度 : 3.42w/o, 4.89w/o, 7.57w/o
 - 定格出力運転中の径方向出力分布の比較
 - 定格出力運転中の軸方向出力分布の比較
- 動特性評価 : SPERTIII-E型炉心出力暴走実験との比較
 - 1960年後半に米国アイダホ国立研究所で実施
 - 熱出力20MWのPWR、炉心直径約1m、燃料集合体60体装荷
 - 投入反応度 : 約0.65 \$ ~ 1.25 \$
 - 制御棒引き抜き後の炉心出力の時間変化の比較

10



Inc./Nippon

SKETCH-INS/TRAC-BF1の概要

項目		仕様
核動特性	空間次元	三次元 (1バンドル 1 メッシュ)
	基礎式	時間依存拡散方程式
	エネルギー群数	2 群
	遅発中性子	6 グループ
	空間非均質	不連続因子を考慮 (精度向上)
	数値解法	近代ノード法により中性子束分布を関数近似して直接解く(陰解法)
燃料棒伝熱	空間次元	径方向一次元 (軸方向ノードごと)
	基礎式	熱伝導方程式
	熱伝達係数	液単相、サブクール沸騰、核沸騰、遷移沸騰、膜沸騰、蒸気単相、凝縮
	数値解法	差分法
チャンネル 熱流動	空間次元	軸方向一次元 (並行多チャンネル)
	流体モデル	二流体モデル
	基礎式	6 方程式 ・気相/混合相:質量保存式 ・気相/液相:運動方程式 ・気相/混合相:エネルギー保存式
	数値解法	スタガードメッシュ法(半陰解法)
	再循環ループ	圧力容器、再循環ポンプ、気水分離器等

12



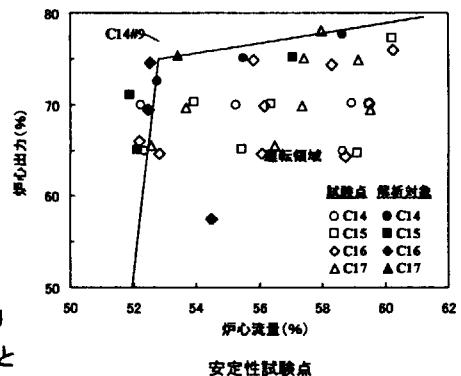
SKETCH-INS/TRAC-BF1の検証解析

■ Ringhals 1号炉プラント仕様

- プラント型式 80万kWe級BWR
- 定格熱出力 2,270 MWt
- 定格炉心流量 7,050 kg/s
- 燃料装荷体数 648体
- 燃料タイプ SVEA-64、8×8

■ 安定性試験

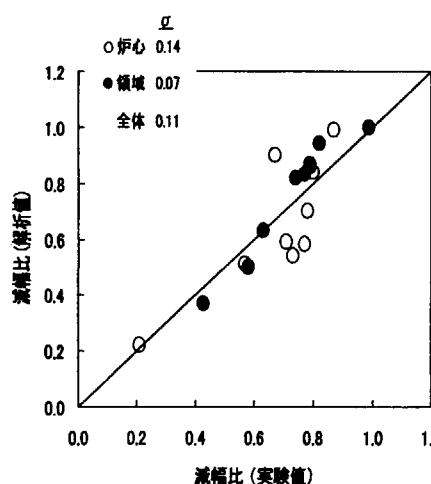
- 中性子束(LPRM)信号のノイズ分析より
炉心安定性及び領域安定性の減幅比と
共振周波数を評価
- サイクル14 試験点9で持続的領域振動を観測
- 高出力・低流量域にある代表ケース10点を解析対象



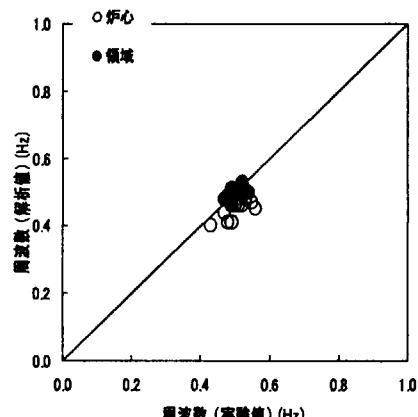
13



Ringhals 1号炉安定性解析結果



減幅比の比較



周波数の比較

付録 1-6 : 東芝（高速炉）のコードシステムの現状

平成 14 年 2 月 22 日

株 式 会 社 東 芝

東芝の FBR 炉物理コードの現状

東芝の FBR 炉心解析業務で使用している炉物理コードの現状を説明する。

1. 計算コード一覧

下表に使用している炉物理コードを示す。実効断面積作成コード・拡散計算コード・燃焼計算コード等は社内開発コードを使っており、現時点では公開の予定はない。この他、公開コードも多数使っている。(MVP, MCNP 等も含まれるが省略)

2. 今後開発すべきコード

共用炉物理コードとしては、さまざまなニーズがあることを考えると、小型炉・高転換炉・ガス炉等にも適用でき、炉型や冷却材に依存せず、汎用で精度が高く、しかもユーザーフレンドリーなコードがあれば良いと考えます。

社内開発コード

分類	計算コード	機能
実効断面積計算	COSMOS	JFS3型断面積ライブラリーをもとにして、多群実効断面積を作成する。さらに、二次元拡散計算によるスペクトルを重みとして少数群実効断面積を作成
燃焼計算	STANBRE	二次元体系で燃料交換を模擬してサイクルごとの実効増倍率・増殖比・中性子束分布・出力分布等を求める。実効断面積作成機能を内蔵
	VETSA	三次元体系で燃料交換を模擬してサイクルごとの実効増倍率・増殖比・中性子束分布・出力分布等を求める。ガンマ線計算も可能
拡散計算	FASTMUGY	二次元拡散計算
	NEW3D	三次元拡散計算、VESTAの燃焼モジュールを省いたもの
摂動計算	PHOBOS	拡散摂動計算コードで断面積による摂動計算、動特性パラメータ計算を行う。

社外導入コード（主なもの）

分類	計算コード	機能	開発元または入手先
実効断面積作成	SLAROM	実効断面積作成	原研
燃焼計算・拡散計算	CITATION	一・二・三次元拡散燃焼計算	コードセンター
輸送計算	TWOTRAN-II	二次元輸送計算	コードセンター
	TRITAC	三次元輸送計算	阪大・JNC
汎用燃焼計算	ORIGEN	ポイント燃焼計算	コードセンター
一般化感度解析	SAGEP	一般化感度解析（断面積による実効増倍率、反応度価値、出力分布等）	阪大・JNC

付録 1-7-1：三菱（軽水炉）のコードシステムの現状

三菱 PWR 核設計コードの現状

コード名	内容	公開
LEOPARD (1次元格子計算コード)	エネルギー多群 MUFF-SOFCAFE 型輸送計算(高選群 54+熱群 172 群)、炉定数は核定数 ENDF/B-III から作成したもの。	LEOPARD で算出した燃料セルの 2 群核定数を HIORA/PANDA で用いる。HIORA と PANDA の計算結果を用いて 3 次元分布を合成する。 従来の PWR 設計手法であり、現在も使用している。
HIORA (2次元炉心計算コード)	エネルギー 2 群詳細メッシュ核計算	公開の予定無し
PANDA (1次元炉心計算コード)	エネルギー 2 群詳細メッシュ核計算	公開の予定無し
PHOENIX-P (2次元格子計算コード)	エネルギー多群輸送計算(セル計算は P1)計算に基づく非均質応答+S4 計算による集合体計算)、炉定数は核定数 ENDF/B-V から作成したもの。	PHOENIX-P で算出した燃料集合体の 2 群核定数を ANC で用いる。ANC 計算では、ビンバワーリカバリー法によって燃料棒毎の出力を算出する。 現在、実機設計に利用している。今後の設計は PHOENIX-P、ANC が中心となる。
ANC (3次元炉心計算コード)	エネルギー 2 群近代ノード法核計算	

メーカーの立場から今後必要となるコード

- (1) 今後、導入される高燃焼度炉心およびMOX 炉心の計算では炉心計算のエネルギー群数の多様化 (~10 群) と輸送計算の導入、定数計算の超多群化などが望まれる。
- (2) 日本原子力研究所で開発中の MOSRA-light コードでは、厳密な計算法の採用、公開の実験データによる検証を重視することを要望する。また現在もその方向で開発されていると理解している。
- (3) 日本で共通のベンチマークとなるコードを作成するには、実機データによる検証がカギとなる。現状は情報の機密性の点から、電力会社側は実機データをほとんど提出しない。しかし、電力自由化の中で該社および許認可手続きの合理化・効率化による経済性向上を目指して情報資源が共有化される可能性がある。

三菱高速炉核設計コードの現状

付録 1-7-2：三菱（高速炉）のコードシステムの現状

分類	コード	機能	内 容	公 開
実効断面積作成	ODD(-BURN)	MICS(三菱作成 ABBN型)、JFS—IIIから均質断面積を作成、1次元並 数、総合計算	もんじゅ、常陽の通常の計算では均質計算で行う。2層非均質効果は PIU+HEX、MVPにより計算する。今後、実機炉心を対象とする計算では基 本計算としてセル計算を取り入れていく。 注:もんじゅの2層非均質効果は0.4Δk(PIU+HEX)、0.5Δk(MVP)	
	PILOT	MICS から円筒体系のセル計算により非均質断面積作成		
	PLATINA	MICS、JFS—IIIから板体体系のセル計算により非均質断面積作成		
	PIU+HEX	MICS、JFS—IIIから六角体系のセル計算により非均質断面積作成		
2次元燃焼計算	2DBURN	2次元 RZ、XY 体系の燃焼計算、出力分布計算	もんじゅ、常陽の通常の計算では2次元 RZ 計算、2次元三角メッシュ計算 を行い、出力分布を合成する。もんじゅ性能試験解析では2次元の組み合 わせとともに3次元計算の解析を実施している。今後、安全審査の整合性 等の事情がある場合を除き、3次元計算で行う。	
	TRIANGLE	2次元三角メッシュ体系の燃焼計算、出力分布計算	注:常陽、もんじゅ体系の2次元計算の組み合わせと3次元計算との出力 分布の差は極一部を除き、1%未満。	公開の予定無し
2次元並数計算	HANYO-DIF	2次元 RZ、XY 体系の並数計算		
	2DBURN-G	2次元 RZ、XY 体系のガソル燃焼計算		
2次元ガソルマ計算	TRIANGLE-G	2次元三角メッシュ体系のガソルマ燃焼計算		
	HEXZ(-BURN)	六角メッシュの燃焼計算		
3次元燃焼計算	TRISTAN	三角メッシュの燃焼計算		
	ENSEMBLE-K	3次元 XYZ の輸送計算	離界面、輸送効果の大きいボイド反応率等については3次元輸送計算を行 う。現在、基本計算に取り入れる予定はない。	
3次元輸送計算	ENSEMBLE-TRIZ	3次元三角メッシュの輸送計算	注:常陽で1.6~2.0Δkもんじゅで0.4Δk。	
	HANYO - PERT、 PARM	2次元並数計算からの中性子束、隨伴中性子束を用いた振動計算	もんじゅ、常陽の通常の計算では2次元 RZ 体系ベースの振動計算を行 う。必要に応じ、3次元計算、輸送計算を行う。	
振動計算	TRI - PERT、 PARM	3次元並数計算からの中性子束、隨伴中性子束を用いた振動計算		
	ENSEMBLE-PERT	3次元輸送計算からの中性子束、隨伴中性子束を用いた振 動計算		

メーカーの立場から今後必要となるコード

- (1) 鮮密なモンテカルロコード MVP,MNCP により合理的な計算時間で出力分布、反応度係数が計算できるようになることが望まれる。
 - (2) ガス冷却炉については離界実験、HTTR 性能試験データを反映し、核燃料バッックが考慮された燃焼計算コード、反応度係数計算コードが望まれる。
 - (3) 新型燃料炉心、ガス冷却高速炉、Pb(Bi)炉心の計算では、離界実験データとともに炉定数化に使用する荷重中性子束の影響の少ない定数計算の超多様化などが望まれる。
- 以上

付録 1-8 日立のコードシステムの現状

2002年7月19日

(株) 日立製作所

日立のBWR核特性解析コードと共用コードの要件

1. 日立のBWR核特性解析コード

日立で使用しているBWR核特性解析コードについて説明する。表1に、自社で開発したコード、あるいは技術提携に基づき他社から導入したコード（及びその改良コード）を記す。いずれのコードも、自社の営業機密に属するか、または提携元との契約に基づき、一般公開はしていない。設計では、主にこれらのコードを使用している。燃料集合体コードの核データライブラリは、NJOYコードにより作成している。表2には、社外から導入した主な公開コードを記す。

表1 自社開発あるいは技術提携で導入したコード（公開不可）

対象	コード名	解析モデル、内容等
燃料集合体	HINES	二次元、輸送+拡散（輸送補正）計算、98群、燃焼計算
	VMONT	三次元、モンテカルロ輸送計算、190群、燃焼計算
炉心	PANACEA	三次元、拡散計算、修正1群、核熱水力結合計算
	COSNEX	三次元、拡散計算（多項式展開ノード法）、3群、核熱水力結合計算
動特性	TRACG	一点～三次元、拡散計算、修正1群、核熱水力結合計算
	STAND	三次元、拡散計算（多項式展開ノード法）、3群、核熱水力結合計算

表2 主な社外導入コード（公開コード）

コード名	解析モデル、内容等
SRAC	二次元、輸送計算、107群、燃焼計算
MVP-BURN	三次元、モンテカルロ輸送計算、連続エネルギー、燃焼計算
MCNP	三次元、モンテカルロ輸送計算、連続エネルギー
CITATION	三次元、拡散計算

2. 共用コードに必要な条件

共用コードが有用であるために必要な条件を、以下に示す。

- (1) 炉型に依存しない汎用性、高精度
- (2) 燃料、炉心の高度化（形状、物理現象の複雑化）に対応できる近似の少ない手法
- (3) 十分な検証、高権威性
- (4) 高速
- (5) 高機能、高い使い勝手
- (6) 高保守性

上記のうち、(3)は特に重要であると考える。メーカーにとっては、その結果で、他を納得させられるだけの、十分に検証された権威のある共用コードが有用である。検証としては、実験データのみでは不十分で、多くの実機データが必要である。また、核的観点だけではなく、熱水力解析手法の検証も必須である。

以上

付録2-1 共用炉物理プログラム基本システムの提案（小林）

共用炉物理プログラム基本システムの提案

(A Proposal of Reactor Physics Program Basic System for Joint Use)

2002.7.19

共用炉物理コードシステム構築 WP 小林啓祐

序

昨年度の共用炉物理コードシステム特別専門委員会報告書の「共用炉物理プログラム基本システムの提案」に下に示すように書いた[1]。やや長いけれども以下に引用する。

***** 引用開始 *****

序 本プログラムシステムは、可能な限り多くの計算機でかつ少ない書き換え労力で今までに作られた多数の炉物理プログラムが使え、また最新のプログラムも便利に使えることを目標にする。そのためには応用プログラムは、FORTRANなどの高級言語で書かれる必要があり、機種依存性のあるアセンブラー言語などの低級言語の使用は避ける事にする。このような高級言語で書かれたプログラムは、現在広く使われているワークステーション、パソコンおよび大型計算機等で殆ど書き換え無しに共通に使えることが期待できる。ここで本委員会で出された幅広い意見を踏まえ、最も簡単に構築できると思われる基本システムを提案する。

1 プログラム基本システム構築の原則

次の原則を置く。

1. 互換性 (exportable or exchangeable) が高いこと、すなわち UNIX システムのワークステーション、ウインドウズパソコン、LINUX のパソコンおよび大型計算機等への導入ではプログラムの変更は殆ど必要としないこと。
2. このような機種間の移植性を高めるために、応用プログラムはFORTRANなどの高級言語で書き、アセンブラー言語などの機種依存性の高い低級言語は使わない。
3. 今までに蓄積された膨大な炉物理プログラムおよびデータは全て使えること。また、そのために必要とされるプログラムおよびデータの書き換え作業は可能な限り少ない労力で容易にできること。
4. プログラムの入れ替え、バージョンアップ、新たなプログラムの追加、保守性および他のプログラムとの連結性が容易なこと、このためにプログラムはモジュール形式とする。
5. 一つのプログラムモジュールは原則として、入力はファイルから読みとり、出力はファイルへ書き出す。この事により、プログラムの保守性と新しいプログラムの追加など拡張性を良くする。
6. 必要とする各モジュールプログラムを選んで走らせるジョブプログラムは簡単に書けること。
7. 必要なプログラムモジュールを適当に選んで使うことにより、種々の計算が行え、幅広い利用者の要求を満たすこと。

2 プログラム基本システム

上記の原則に則り、最も簡単に作り得ると思えるプログラムの基本システムを提案する。原則で述べたように、プログラムモジュールは入力データはファイルから読み込み、ファイルへ出力するように書かれているものとする。プログラムの機種間の互換性および利便性を高めるために必要なことは、各プログラムモジュールで使うデータの読み込みおよび計算結果の出力方法を便利で利用しやすいように決めることである。現在広く使われているFORTRANプログラムでは入出力ファイル名は実名または仮名(変数名)で使うことができる。それぞれの場合について、理解を容易にするためにここでは

簡単な二つのFORTRANプログラムを走らせる場合を例にして説明する。一般には、プログラムモジュールを順次読み出す事により何個でも走らせることが出来る。

***** 引用終わり *****

上の文章を付けて提案した前回の方法では、複数個のFORTRANプログラムを走らせるのに、Windowsパソコンのバッチジョブを使った。このバッチジョブの書き方はWindowsパソコンとワークステーションでは異なる。この差を無くす事は、複数個のFORTRANを走らせるのにFORTRANプログラムを使えば実現できる。FORTRAN90では、プログラムをモジュールとして使う方法があり^{2]}、このMODULE文を使って複数個のFORTRANプログラムを順次走らせる方法を検討したので紹介する。

1 プログラム基本システム

1.1 実名ファイルを使う場合

前回に使った例をそのまま使って説明する。ここで使うプログラムProga.forは例えば3次元多群拡散プログラム、Prog.b.forは3次元多群拡散プログラムを走らせて得られた中性子束を使って摂動計算をするプログラム、この二つの間に走らせるProgconv.forは前のプログラムからの中性子束等を後の摂動計算プログラムへのデータファイルのフォーマットをあわせるためのプログラムである。すなわち、これらの3個のプログラムは表1の順に走らせる。

表1 入出力データファイル名およびプログラム名

入力ファイル名	->	プログラム名	->	出力ファイル名
---------	----	--------	----	---------

1) Progafname.txt
 Progainp.txt -> Proga.for -> Progaout.dat,
 2) Progconfname.txt
 Progaout.dat -> Progconv.for -> Progbinp.txt
 3) Progbfname.txt
 Progbinp.txt -> Progb.for -> Progbout.txt

各プログラムへの入力ファイルの所在を知らせるファイルProgafname.txt等は実名と仮名で書く方法があるが、ここではまず実名の場合を説明する。

Proga.for, Progconv.for および Progb.for 等のプログラムおよびデータファイルのハードディスク上の場所を図1に示す。

```
C:\Data02\ProgSys2\Batch\BatchfA.for
|   |--\Debug\BatchfA.exe
|   |
|   |--\Progafname.txt,Progconvfname.txt,Progbfname.txt
|
|--¥Proga\Fort\Proga.for
|   |
|   |--\Debug\Proga.exe
|
|
|--¥Progconv\Fort\Progconv.for
|   |
|   |--\Debug\Progconv.exe
|
|--¥Progb\Fort\Progb.for
|   |
```

```

|           |--\Debug\Progb.exe
|
|--¥Prog\Fort\Time\Time.for
|
|--\Inout\Progainp.txt, Progaout.dat,
    |--\Progbinp.txt, Progbout.dat

```

図1 モジュールプログラム Proga.for, Progconv.for Progb.for およびそれらを順次走らせるためのFORTRAN プログラム BatchfA.for のハードデスクデレクトリ上の配置

これらの3個のプログラムを順次走らせるのは、Program 1に示すFORTRAN プログラムである。このFORTRAN プログラム BatchfA.for は、Proga.for, Progconv.for および Progb.for をモジュール化したプログラム SProga, SProgconv および SProgb を順に呼び出して走らせる。

Program 1: 3個のFORTRAN プログラムを順次走らせるためのFORTRAN プログラム BatchfA.for
プログラム BatchfA.for のある場所 C:\Data02\ProgSys2\BatchfA.for

```

C   Test Program of Moudule 2002.3.11
      PROGRAM BatchA
      USE MProga
      USE MProgconv
      USE MProg
      USE MPTIME
C
      INTEGER*2 IT,IHRA,IMINA,ISECA,I100THA
C
      IT=0
      CALL SPTIME(IT,IHRA,IMINA,ISECA,I100THA)
C
      WRITE(6,*) 'SProga.for Start'
      CALL SProga
      WRITE(6,*) 'SProgconv.for Start'
      CALL SProgconv
      WRITE(6,*) 'SProgb.for Start'
      CALL SProgb
C
      CALL SPTIME(IT,IHRA,IMINA,ISECA,I100THA)
C
      STOP
      END PROGRAM BatchA

```

通常のFORTRAN プログラムをモジュール化するには、Program 2に示す様に、メインプログラムの文頭に MODULE MProga(モジュール名) を書き、次に CONTAINS と書き、通常のメインプログラムをサブルーチンにする。この時、通常のSTOP 文の代わりに文末に RETURN 文を入れなければならない。それ以外の変更は不要である。それ故、通常の巨大なプログラムも文頭の3行と文末のRETURN 文を追加するだけで、モジュールプログラムにすることが出来る。Program 3, 4 も同様である。

Program 2: MODULE プログラム ファイル名 Proga.for

ファイルの場所 C:\Data02\ProgSys2\Fort\Proga.for

```

C Basic Concept of Reactor Physics Program System for Joint Use
proposed by K.Kobayasi, 2001.1.31
C "Proga.for" written on 2001.1.31, Revised at 2001.1.31.
MODULE MPrga
CONTAINS
SUBROUTINE SProga
CHARACTER*72 Inpfile, Outfile
Dimension RinpData(10),OutData(10)
C
C Added at 02.3.14
INTEGER*2 IT,IHRA,IMINA,ISECA,I100THA
REAL*8 DA,DX
REAL*8 DAM(20000000)
REAL*4 A,X
REAL*4 AM(20000000)
IM=20000000
N=20000000
IT=0
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
C Single Precision
WRITE(6,*) 'Single Precision'
A=0.9999999
X=1.0
DO 10 I=1,N
10 X=A*X
WRITE(6,*) 'Multiplication N=', N,X
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
DO 20 I=1,N
20 X=X/A
WRITE(6,*) 'Division N=', N,X
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
DO 30 I=1,N
30 X=A+X
WRITE(6,*) 'Addition N=', N, X
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA )
DO 40 I=1,IM
40 AM(I)=X
WRITE(6,*) 'Save to Array IM=', IM
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
C Double Precision
WRITE(6,*) 'Double Precision'
N=100000000
DA=0.9999999
DX=1.0
DO 15 I=1,N
15 DX=DA*DX
WRITE(6,*) 'Multiplication N=', N,DX
CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
DO 25 I=1,N
25 DX=DX/DA

```

```

      WRITE(6,*) 'Division N=', N,DX
      CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)
      DO 35 I=1,N
 35 DX=DA+DX
      WRITE(6,*) 'Addition N=', N, DX
      CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA )
      DO 45 I=1,IM
 45 DAM(I)=DX
      WRITE(6,*) 'Save to Array IM=', IM
      CALL PTIME(IT,IHRA,IMINA,ISECA,I100THA)

C     STOP
C     END
C

      OPEN(2,FILE ='Progafname.txt')
      READ(2,*) Inpfile, Outfile
C     READ(5,*) Inpfile, Outfile
      WRITE(6,*) Inpfile, Outfile
      OPEN(UNIT=3,FILE =Inpfile)
      OPEN(UNIT=10,FILE =Outfile,STATUS='UNKNOWN',FORM='UNFORMATTED',
&RECL=2000)
      READ(3,*) N,(RinpData(I),I=1,N)
      WRITE(6,*) N,(RinpData(I),I=1,N)
      DO 100 I=1,N
 100 OutData(I)=RinpData(I)**2
      WRITE(6,*) N,(OutData(I),I=1,N)
      WRITE(10) N,(OutData(I),I=1,N)

C     STOP
      CLOSE(2,STATUS = 'KEEP')
      CLOSE(3,STATUS = 'KEEP')
      CLOSE(10,STATUS = 'KEEP')
      RETURN
      END SUBROUTINE SProga
C

      subroutine PTIME(IT,IHRA,IMINA,ISECA,I100THA)
C     Print time and elapsed time when called, in min, sec and 1/100 sec.
      INTEGER*2 IT,IHRA,IMINA,ISECA,I100THA,IHRB,IMINB,ISECB,I100THB,
 1      IMIND,ISECD,I100THD,IHRBP,IMINBP,ISECBP,I100THBP
      CALL GETTIM(IHRB,IMINB,ISECB,I100THB)
      I100THD=I100THB-I100THA
      ISECBP=ISECB
      IF(I100THD.LT.0) THEN
      I100THD=I100THD+100
      ISECBP=ISECB-1
      ENDIF
      ISECD=ISECBP-ISECA
      IMINBP=IMINB
      IF(ISECD.LT.0) THEN
      ISECD=ISECD+60
      IMINBP=IMINB-1
      ENDIF

```

```

IMIND=IMINBP-IMINA
WRITE(6,200)IT,IMINB,ISECB,I100THB,IMIND,ISECD,I100THD
200 FORMAT(' NO',I2, I4, 'min', I2, '.', I2, 'sec', 3X, 'Interval',
1           2X, I2, 'min', I2, '.', I2, 'sec' )
WRITE(16,200)IT,IMINB,ISECB,I100THB,IMIND,ISECD,I100THD
IHRA=IHRB
IMINA=IMINB
ISECA=ISECB
I100THA=I100THB
IT=IT+1
RETURN
END SUBROUTINE
C
END MODULE MProlga

```

Program 3: MODULE プログラム、ファイル名 Progconv.for
 ファイルの場所 C:\Data02\ProgSys2\Fort\Progconv.for

```

C   Basic Concept of Reactor Physics Program System for Joint Use
C   proposed by K.Kobayashi, 2002.3.13
C   "Progconv.for" written on 2001.1.31, Revised at 2002.3.13.
MODULE MProlconv
CONTAINS
SUBROUTINE SProgconv
CHARACTER*72 Inpfile, Outfile
Dimension RinpData(10),OutData(10)
OPEN(2,FILE ='Progconfname.txt')
READ(2,*) Inpfile, Outfile
C   READ(5,*) Inpfile, Outfile
WRITE(6,*) Inpfile, Outfile
OPEN(UNIT=3,FILE =Inpfile,FORM='UNFORMATTED',RECL=2000)
OPEN(UNIT=10,FILE =Outfile)
READ(3) N,(RinpData(I),I=1,N)
WRITE(6,*) N,(RinpData(I),I=1,N)
DO 10 I=1,N
10 OutData(I)=SQRT(RinpData(I))
WRITE(6,*) N,(OutData(I),I=1,N)
WRITE(10,*) N,(OutData(I),I=1,N)
C   STOP
CLOSE(2,STATUS = 'KEEP')
CLOSE(3,STATUS = 'KEEP')
CLOSE(10,STATUS = 'KEEP')
RETURN
END SUBROUTINE SProgconv
END MODULE MProlconv

```

注 このプログラムにも Prog.a.for と同様に C Added at 02.3.14 以下のベンチマーク計算とサブルーチン PTIME が追加されている。

Program 4: MODULE プログラム ファイル名 Progb.for

ファイルの場所 C:\Data02\ProgSys2\ProgB.for

```
C      Basic Concept of Reactor Physics Program System for Joint Use
C      proposed by K.Kobayashi, 2002.3.13
C      "ProgB.for" written on 2001.1.31, Revised at 2001.1.31.
      MODULE MProgB
      CONTAINS
      SUBROUTINE SProgB
      CHARACTER*72 Inpfile, Outfile
      Dimension RinpData(10),OutData(10)
      OPEN(2,FILE ='ProgBfname.txt')
      READ(2,*) Inpfile, Outfile
C      READ(5,*) Inpfile, Outfile
      WRITE(6,*) Inpfile, Outfile
      OPEN(UNIT=3,FILE =Inpfile)
      OPEN(UNIT=10,FILE =Outfile,STATUS='UNKNOWN',FORM='FORMATTED',
      &      RECL=2000)
      READ(3,*) N,(RinpData(I),I=1,N)
      WRITE(6,*) N,(RinpData(I),I=1,N)
      DO 10 I=1,N
10   OutData(I)=RinpData(I)*2
      WRITE(6,*) N,(OutData(I),I=1,N)
      WRITE(10,*) N,(OutData(I),I=1,N)
C      STOP
      CLOSE(2,STATUS = 'KEEP')
      CLOSE(3,STATUS = 'KEEP')
      CLOSE(10,STATUS = 'KEEP')
      RETURN
      END SUBROUTINE SProgB
      END MODULE MProgB
```

注 このプログラムにも ProgA.for と同様に C Added at 02.3.14 以下のベンチマーク計算とサブルーチン PTIME が追加されている。

Program 5: MODULE プログラム ファイル名 ProgB.for
 ファイルの場所 C:\Data02\ProgSys2\time\Ptime.for

```
MODULE MPtime
CONTAINS
SUBROUTINE SPTIME(IT,IHRA,IMINA,ISECA,I100THA)
C      Print time and elapsed time when called, in min, sec and 1/100 sec.
      INTEGER*2 IT,IHRA,IMINA,ISECA,I100THA,IHRB,IMINB,ISECB,I100THB,
1      IMIND,ISECD,I100THD,IHRBP,IMINBP,ISECBP,I100THBP
      CALL GETTIM(IHRB,IMINB,ISECB,I100THB)
      I100THD=I100THB-I100THA
      ISECBP=ISECB
      IF(I100THD.LT.0) THEN
      I100THD=I100THD+100
      ISECBP=ISECB-1
      ENDIF
```

```

ISECD=ISECBP-ISECA
IMINBP=IMINB
IF(ISECD.LT.0) THEN
ISECD=ISECD+60
IMINBP=IMINB-1
ENDIF
IMIND=IMINBP-IMINA
WRITE(6,200)IT,IMINB,ISECB,I100THB,IMIND,ISECD,I100THD
200 FORMAT(' NO',I2, I4, 'min', I2, '.', I2, 'sec', 3X, 'Interval',
1           2X, I2, 'min', I2, '.', I2, 'sec' )
WRITE(16,200)IT,IMINB,ISECB,I100THB,IMIND,ISECD,I100THD
IHRA=IHRB
IMINA=IMINB
ISECA=ISECB
I100THA=I100THB
IT=IT+1
RETURN
END SUBROUTINE
END MODULE MPTIME

```

2 走らせ方

ウィンドウズパソコンで DIGITAL Visual Fortran を使う場合は、これらのプログラムは次のようにすると走らせることが出来る。

Module Program の走らせ方、操作法

1. エクスプローラーで Proga.for をクリックして、FORTRAN を立ち上げてコンパイルする。同様に Progconv.for および Progb.for をコンパイルして exe ファイル等を作る。
2. BatchfA.for をクリックして、FORTRAN を立ち上げてコンパイルする。図 2 の Workspace 'BatchfA': 1 project(s) が作られる。
3. Project -> Add to project -> files を選択して Proga.for のあるデレクトリとファイル名を指定する。同様に Progconv.for および Progb.for を指定する。図 2 の Workspace に Proga.for 等が追加される。
4. Build -> Build All を選択してモジュールプログラムをリンクする。
5. Build -> Execute BatchfA.exe を選択して走らせる。DOS 窓が開いて、Write(6,*) でかかれたものは窓の中に表示される。

文献1のデータと同じデータを使った場合の出力を図 2 に示す。

```

Workspace 'BatchfA': 1 project(s)
|
|--BatchfA files
|
|--BatchfA.for
|--Proga.for
|--Progconv.for
|--Progb.for

```

```

|--Time.for
|--External Dependencies
|----MProga.mod
|----MProgB.mod
|----MProgconv.mod
|----MPTIME.mod

```

図2 Workspace の表示

コンパイルおよびリンクを行うと、次のメッセージが出力される。

```

Compiling Fortran...
C:\Data02\ProgSys2\Time\Fort\Time.for
C:\Data02\ProgSys2\Progcov\Fort\Progconv.for
C:\Data02\ProgSys2\ProgB\Fort\ProgB.for
C:\Data02\ProgSys2\Proga\Fort\Proga.for
C:\Data02\ProgSys2\Batch\BatchfA.for
Linking...
Debug\BatchfA.exe : warning LNK4084: total image size 735666176 exceeds
max (268435456); image may not run
BatchfA.exe - 0 error(s), 1 warning(s)

```

このリンクの段階で、全イメージサイズが最大値を超えるのでイメージは走らないかもしれないとの警告文ができる。しかし、走らせると不具合無く走り、下に示す出力がDOS窓に表示される。

表2 DOS 窓への出力

NO 0 45min39.24sec	Interval	45min39.24sec	
SProga.for Start			
NO 0 45min39.26sec	Interval	45min39.26sec	
Single Precision			
Multiplication N=	20000000	0.1082267	
NO 1 45min39.43sec	Interval	0min 0.17sec	
Division N=	20000000	0.9999999	
NO 2 45min39.81sec	Interval	0min 0.38sec	
Addition N=	20000000	1.6777216E+07	
NO 3 45min39.98sec	Interval	0min 0.17sec	
Save to Array IM= 20000000			
NO 4 45min40.26sec	Interval	0min 0.28sec	
Double Precision			
Multiplication N=	100000000	6.649760854841028E-006	
NO 5 45min41.15sec	Interval	0min 0.89sec	
Division N=	100000000	1.000000000000000	
NO 6 45min43. 9sec	Interval	0min 1.94sec	
Addition N=	100000000	99999989.0790710	
NO 7 45min43.95sec	Interval	0min 0.86sec	
Save to Array IM= 20000000			
NO 8 45min49.70sec	Interval	0min 5.75sec	
C:\Data02\ProgSys2\Inout\Progainp.txt			
C:\Data02\ProgSys2\Inout\Progaout.dat			
4 1.000000	2.000000	3.000000	4.000000
4 1.000000	4.000000	9.000000	16.000000

```

SProgconv.for Start
NO 0 45min49.79sec Interval 45min49.79sec
Single Precision
Multiplication N= 20000000 0.1082267
NO 1 45min49.96sec Interval 0min 0.17sec
Division N= 20000000 0.9999999
NO 2 45min50.34sec Interval 0min 0.38sec
Addition N= 20000000 1.6777216E+07
NO 3 45min50.51sec Interval 0min 0.17sec
Save to Array IM= 20000000
NO 4 45min59.84sec Interval 0min 9.33sec
Double Precision
Multiplication N= 100000000 6.649760854841028E-006
NO 5 46min 0.76sec Interval 0min 0.92sec
Division N= 100000000 1.000000000000000
NO 6 46min 2.68sec Interval 0min 1.92sec
Addition N= 100000000 99999989.0790710
NO 7 46min 3.56sec Interval 0min 0.88sec
Save to Array IM= 20000000
NO 8 46min23.87sec Interval 0min20.31sec
C:\Data02\Progsys2\Inout\Progaout.dat
C:\Data02\Progsys2\Inout\Progbinp.txt
        4 1.000000      4.000000      9.000000      16.000000
        4 1.000000      2.000000      3.000000      4.000000

SProg.b.for Start
NO 0 46min24.26sec Interval 46min24.26sec
Single Precision
Multiplication N= 20000000 0.1082267
NO 1 46min24.43sec Interval 0min 0.17sec
Division N= 20000000 0.9999999
NO 2 46min24.82sec Interval 0min 0.39sec
Addition N= 20000000 1.6777216E+07
NO 3 46min25. 1sec Interval 0min 0.19sec
Save to Array IM= 20000000
NO 4 46min35.24sec Interval 0min10.23sec
Double Precision
Multiplication N= 100000000 6.649760854841028E-006
NO 5 46min36.17sec Interval 0min 0.93sec
Division N= 100000000 1.000000000000000
NO 6 46min38. 7sec Interval 0min 1.90sec
Addition N= 100000000 99999989.0790710
NO 7 46min38.98sec Interval 0min 0.91sec
Save to Array IM= 20000000
NO 8 46min59.12sec Interval 0min20.14sec
C:\Data02\Progsys2\Inout\Progbinp.txt
C:\Data02\Progsys2\Inout\Progbout.dat
        4 1.000000      2.000000      3.000000      4.000000
        4 2.000000      4.000000      6.000000      8.000000
NO 1 46min59.40sec Interval 1min20.16sec
Press any key to continue

```

3 考察

使用しているパソコンのRAMの量は256MBである。Proga.for, Progconv.forおよびProg.b.forに追加したベンチマーク計算で使っているArrayのサイズは、単精度のAM(20000000)が20,000,000x4Byte=80MB、倍精度のDAM(20000000)が20,000,000x8=160MB、合計240MBで1個のプログラムProga.forを走らせるだけではほぼ全てのRAMが必要である。それでコンパイルの段階ではエラーメッセイジは出ないが、リンクの段階では全サイズで735,666,176B必要で、最大値268,435,456Bを超えてるので走らないかもしれないとの警告文が出る。しかし、実際には何事も無く走るので、このモジュールの使用文USEを使ったモジュールプログラムのCALL文では、全てのモジュールを同時にRAMへ読み込むのではなく、順次1個づつ読み込んで実行していることが分かる。それ故、このモジュール化する方法を使うと、何十個ものプログラムも順次走らせることが可能と思われる。

本方法の長所

1. 在来のプログラムへの書き換えは、既にパソコン上で走っている場合、4行の追加だけである。
2. 全てFORTRANプログラムだけで書かれるので、ワークステーション等での利用でも書き換えの必要が無く、移植は極めて容易である。
3. 多群拡散コード、多群輸送コードなど、個々のプログラムは適当に別々のデレクトリに置いてよいので、管理が容易である。
4. データファイルも任意のデレクトリに置いて良いので使い勝手が良い。
5. いくつかのプログラムを順次走らせるプログラムでは、IF文、DOループなどFORTRANの全ての命令が使えるので、条件で分岐させながら多くのプログラムを種々の組み合わせで走らせることが出来る。

注意または今後検討すべき点

1. デレクトリ名に\My DocumentのようににスペースがあるとFortran ProgramのRead文で、そこで次の文字を次のデータとして読み込むために、ファイル名が正確に読み込まれない。Formatで、72Hを使うと読み込めるはずであるが、それがファイル名として正しく使えるか否かはチェックが必要である。この困難を避けるために、ここではデレクトリ名にスペースが無い様にした。こうするとFortranは正常に走る。
2. 前回の方法では、オープンしたファイルに対してクローズ文を必ずしも書かなくても良かったが、このモジュール文では同じファイルを後で走るモジュールで使う場合は必ずクローズ文を書かねばならない。
3. 例えば、BatchfA.forを走らせて、Proga.forが走って読み込むProgafname.txtは\Batchのディレクトリから読み込まれる。この時、Proga.forの入出力で使ったファイルは閉じる時にCLOSE(N,STATUS='DELETE')とすると消えてしまう。CLOSE(N,STATUS='KEEP')とすると消えずに残る。
4. モジュールプログラムBatchfA.forを走らせる時、パソコンのRAMには個々のモジュールを順次ロードして使うことを確かめる必要がある。もし、全てのモジュールを同時にRAMへ読み込むと使えるメモリが減り実用にならない。モジュールプログラムは一度に読み込まれるのではなく、1個づつ順次読み込まれる事は上記のプログラムを走らせる事により確かめられた。

文献

1. 小林啓祐、「共用炉物理プログラム基本システムの提案」共用炉物理コードシステム特別専門委員会報告書, p.150, JNC TJ9420 2001-007 (2001).
2. ラリー・ニーホフ、サンフォード・リストマー、入門Fortran90、ピアソン・エデュケーション(2001)。

付録 2-2 : 小林案に対する検討結果（横山）

第3回共用炉物理コードシステムの構築 WP 資料

2002年7月19日

サイクル機構 横山賢治

1. はじめに

第2回会合資料（WP01-2-3、WP01-2-4）で、Python、Ruby 等のオブジェクト指向スクリプト言語を利用することを提案したが、これらのスクリプト言語の解析コードシステムへの実績が少ないとや、これまでに作成されている解析コードは、Fortran で書かれているものがほとんどであることから、Fortran により基本システムを構築するのは、現実的な選択肢のひとつであると考えられる。

ここでは、松村リーダ（付録 2-3）と小林幹事（付録 2-1）が提案された Fortran による方法を参考にし、Fortran90 の MODULE 文、TYPE 文の機能を用いた方法を提案する。

2. 各提案に関する検討

ここで、これまでに提案されている Fortran による 2 つの方法についての長所・短所について、筆者なりに検討した結果を述べる。

(1) 松村リーダの提案に関する検討

(長所)

- 物理的意味が明確な計算処理（サブルーチン）群：

計算制御用の簡易コマンドと Fortran のサブルーチンをほぼ一体で対応されており、拡散計算を行う DIF3D、マクロ断面積を印刷する PRN_MACRO、エネルギー群縮約計算を行う CONDEN といったサブルーチンに分かれており、明確な物理的意味を持った単位に機能が分離されている。更に、メモリ管理機能によりサブルーチンが独立化されている。

- 入力データと計算制御の一体化：

入力データと計算制御がひとつのファイルにまとまっているので、どのような入力データと、解析コード（サブルーチン）を使って計算しているのかが明示されていて、分かりやすい。

- 再コンパイル不要：

解析システムが、簡易言語を理解しながら実行してくれるので、簡易言語による計算制御を変更しても、解析システムを再コンパイルする必要がない。

(短所)

- ・ 物理的意味で定義された統一的なデータ形式がない：

計算モデルや処理だけでなく、データについても、物理的意味が明確に分かる単位で定義できること、更に良くなるのではないかと思われる。簡易言語のレベルで、データの受け渡しを統一的に取り扱う方法があると更に良くなるのではないかと思われる。
- ・ 高度な計算制御が困難：

システムが大きくなるにつれて、計算の制御には、条件分岐や繰り返し等が必要になると考えられる。独自に開発する簡易言語に対して、このような機能を追加するのは難しいと思われる。

(2) 小林幹事の提案に関する検討

(長所)

- ・ USE 文によるモジュールの呼び出し：

各解析コードが MODULE 文でラップされているので、必要なものだけを USE 文で呼び出して利用できる。ユーザーは、解析システムの機能をすべて一度に使うことはないはずであるので、必要なものだけ「USE XXXX」とすればよいのは、便利で分かりやすい。
- ・ 簡易言語の作成が不要：

Fortran を使って解析コードの連携を制御するので、新たな簡易言語を作成したり、簡易言語を覚えたりする必要がない。DO 文や IF 文を使った複雑な計算制御も簡単にできる。

(短所)

- ・ システム内部のデータ・計算処理の単位が大きい：

既存のコードをそのまま使うので、物理的意味に比べて、データ・計算処理の単位が大きすぎると思われる。例えば、SRAC のような大型の解析システムがひとつのモジュールになったとしたら、SRAC というモジュールと結合すべき解析コードの組み合わせの数が多くなりすぎ、得策ではないと考えられる。
- ・ 解析コード依存のデータ形式：

計算制御を行うメインプログラム部で、解析コード間でどのようなデータを受け渡しするのかが明示されない。また、新たなコードを追加するためには、既存のコードを書き換えて、入出力を整備することになると思われる所以、新たなコードを追加するたびに、その影響が他のコードに伝播してしまう。

3. 今回の提案

以上の検討結果をふまえて、Fortran90 の MODULE 文の利用に加えて、TYPE 文を利用するこことにより、もう少し進んだモジュラー化プログラミングが可能なことと、及び、各解析コード間でのデータの受け渡しを明示化した Fortran のプログラムを使って解析コードを連携を制御する方法を提案する。

リスト 1 (calc_keff.f) に作成したプログラムを示す。リスト 2 (mapdata.inp) は入力データのサンプルであり、リスト 3 (output.txt) が実行結果である。今回作成したサンプルプログラムでは、データの受け渡し方法を明示化するという点においており、他の部分については、今後、検討を重ねる必要があると考えている。今回作成したサンプルプログラムについても、長所と短所を検討した結果を以下に記す。

(長所)

- データの受け渡しの明示化 :
- 受け渡しするデータは TYPE 文で定義された複雑なデータ構造を持っているため、中性子束、マクロ断面積、体系データ、実効増倍率等、物理的な意味の単位を、ひとつの変数として取り扱うことができる。これにより、物理的なデータとそれに付随する処理をひとまとめにして取り扱うことが可能となる。また、計算制御を行うメインルーチンでは、それぞれ、`flx`、`mac`、`map` というひとつの変数を介して、サブルーチンからサブルーチン（解析コードから解析コード）へのデータの受け渡しが明示化されることになる。
- データ交換のための共通データ形式 :

計算処理と入出力機能はそれぞれサブルーチンに、制御機能はメインルーチンに分離され、各機能のインターフェイスは、すべて TYPE 文で定義された、構造化されたデータを介して行われることになる。TYPE 文で定義されたデータは、データ交換のための共通データ形式として利用することができる。（更に、TYPE 文で定義されたデータ形式と、常に一対一対応したファイルフォーマットを定義できれば、データが、ファイル上にあるのか、データ上にあるのかを意識しなくて済むようになる。例えば、XML 等を利用すれば、TYPE 文で定義されるような構造化されたデータのファイルフォーマットを定義することが可能と考えられる。）

(短所)

- 再コンパイルの必要性 :
- 計算制御の流れを変更する度に、Fortran プログラムをコンパイルする必要がある。制御の流れの変更は、通常の感覚では入力データ作成の過程であり、入力データを変更するたびに解析コードの再コンパイルが必要という状態は、煩雑であり実用的でない。

ただし、この点については、コンパイルの不要なスクリプト言語 Ruby、Python 等を採用し、スクリプト言語から Fortran のサブルーチンを呼び出す形にすることで解決できると考えられる。

- ・ 入力データと計算制御の分離 :

現状では、計算制御部分に、入力データをあわせて記述することができない。入力データは計算制御の一部と考えられるので、松村リーダの提案のように、同一のファイルに記述できた方が良いと思われる。

- ・ 大容量メモリの必要性 :

データを一度に保持する必要があるため、大容量のメモリを必要とする可能性がある。ただし、不要になったメモリを解放して再利用することが可能であるし、問題を解くのに必要な最小限の配列数しか使わないので、固定長配列と比べれば、効率的なメモリ利用が可能と考えられる。

<参考文献>

- (1) 富田博之、「Fortran90 プログラミング」、培風館
- (2) Marin Cournihan, "FORTRAN95", UCL Press
- (3) A. C. Marshall et al., "The University of Liverpool Fortran90 Course Notes" (URL <http://www.liv.ac.uk/HPC/F90page.html>)
- (4) V. K. Decyk, C. D. Norton, and B. K. Szymanski, "How to Express C++ Concepts in Fortran 90," *Scientific Programming*, , vol. 6, no. 4, IOS Press, Winter 1997, p. 363. (URL <http://exodus.physics.ucla.edu/Fortran95/ExpressC++.pdf>)
- (5) V. K. Decyk, C. D. Norton, and B. K. Szymanski, "How to support inheritance and run-time polymorphism in Fortran 90", *Computer Physics Communications* 115, 9, 1998. (URL <http://exodus.physics.ucla.edu/Fortran95/Stopwatch.pdf>)

Jul 18 2002 14:57

calc_keff.f

Page 1

! 共用炉物理コードシステムの構築WP
! コード仕様検討用サンプルプログラム

! 2002.7 サイクル機構 横山

! このプログラムは、Absoft Fortran 90/95 compiler version 3.0で
! 作成しています。Fortran77以前のコンパイラでは利用できません。

```

MODULE macro_xsec_module
!
! マクロ断面積のデータ構造やそれに関連する操作を定義する
! モジュールです。
!
IMPLICIT NONE
TYPE macro
    INTEGER :: ngroup
    REAL, POINTER, DIMENSION(:) :: coef_diff, xsec_abs, xsec_prd
!
! メモリの動的な確保とサブルーチンへ引数として受渡しを可能と
! するために、上記のように宣言します。
!
END TYPE

CONTAINS

SUBROUTINE init_macro(this, ng)
!
! データを初期化します。
! (コンストラクタに相当)
!
    TYPE(macro), INTENT(INOUT) :: this
    INTEGER, INTENT(IN) :: ng
    INTEGER :: i
    WRITE(*,*) '<INIT_MACRO>'
    ALLOCATE (this%coef_diff(ng),           ! メモリを動的に割り当てます。
              this%xsec_abs(ng),
              this%xsec_prd(ng))
    this%ngroup = ng
    DO i=1, ng
        this%coef_diff(i) = 0.0           ! ゼロで初期化します。
        this%xsec_abs(i) = 0.0
        this%xsec_prd(i) = 0.0
    END DO
END SUBROUTINE

SUBROUTINE delete_macro(this)
!
! メモリを解放します。
! (デコンストラクタに相当)
!
    TYPE(macro), INTENT(INOUT) :: this
    WRITE(*,*) '<DELETE_MACRO>'
    DEALLOCATE (this%coef_diff,           ! メモリを動的に解放します。
                this%xsec_abs,
                this%xsec_prd)
END SUBROUTINE

SUBROUTINE print_macro(this)
!
! macro型データの内容を表示します。
!
    TYPE(macro), INTENT(IN) :: this
    INTEGER :: i
    WRITE(*,*) '<PRINT_MACRO>'
```

Jul 18 2002 14:57

calc_keff.f

Page 2

```

      WRITE(*,*) 'Number of Energy Group: ', this%ngroup
      WRITE(*,*) 'Diffusion Coefficient: '
      WRITE(*,*) (this%coef_diff(i), i=1, this%ngroup)
      WRITE(*,*) 'Absorption: '
      WRITE(*,*) (this%xsec_abs(i), i=1, this%ngroup)
      WRITE(*,*) 'Production: '
      WRITE(*,*) (this%xsec_prd(i), i=1, this%ngroup)
END SUBROUTINE

END MODULE

MODULE neutron_flux_module
!
! 中性子束のデータ構造やそれに関連する操作を定義する
! モジュールです。
!
IMPLICIT NONE
TYPE flux
  INTEGER :: ngroup
  REAL, POINTER, DIMENSION(:) :: flux
END TYPE

CONTAINS

SUBROUTINE init_flux(this, ng)
!
! データを初期化します。
!
TYPE(flux), INTENT(INOUT) :: this
INTEGER, INTENT(IN) :: ng
INTEGER :: i
WRITE(*,*) '<INIT_FLUX>'
ALLOCATE (this%flux(ng))
this%ngroup = ng
DO i=1, ng
  this%flux(i) = 0.0
END DO
END SUBROUTINE

SUBROUTINE delete_flux(this)
!
! メモリを解放します。
!
TYPE(flux), INTENT(INOUT) :: this
WRITE(*,*) '<DELETE_FLUX>'
DEALLOCATE (this%flux)
END SUBROUTINE

SUBROUTINE print_flux(this)
!
! flux型データの内容を表示します。
!
TYPE(flux), INTENT(IN) :: this
INTEGER :: i
WRITE(*,*) '<PRINT_FLUX>'
WRITE(*,*) 'Neutron Flux: '
WRITE(*,*) (this%flux(i), i=1, this%ngroup)
END SUBROUTINE

END MODULE

MODULE map3d_module
!
! 3次元計算体系のデータ構造やそれに関連する操作を定義する
! モジュールです。
!
```

Jul 18 2002 14:57

calc_keff.f

Page 3

```

IMPLICIT NONE
TYPE map3d
    INTEGER :: imax, jmax, kmax
    INTEGER, POINTER, DIMENSION(:,:,:,:) :: mapdata
END TYPE

CONTAINS

SUBROUTINE init_map3d(this, imax, jmax, kmax)
!
! データを初期化します。
!
    TYPE (map3d), INTENT(INOUT) :: this
    INTEGER, INTENT(IN) :: imax, jmax, kmax
    INTEGER :: i, j, k
    WRITE(*,*) '<INIT_MAP3D>'
    this%imax = imax
    this%jmax = jmax
    this%kmax = kmax
    ALLOCATE (this%mapdata(imax, jmax, kmax))
    DO i=1, this%imax
        DO j=1, this%jmax
            DO k=1, this%kmax
                this%mapdata(i,j,k) = 0
            END DO
        END DO
    END DO
END SUBROUTINE

SUBROUTINE delete_map3d(this)
!
! メモリを解放します。
!
    TYPE(map3d), INTENT(INOUT) :: this
    WRITE(*,*) '<DELETE_MAP3D>'
    DEALLOCATE (this%mapdata)
END SUBROUTINE

SUBROUTINE read_map3d(this, filename)
!
! map3d型データをファイルから読み込みます。
!
    TYPE(map3d), INTENT(OUT) :: this
    CHARACTER (LEN=*) :: filename
    INTEGER :: i, j, k
    WRITE(*,*) '<READ_MAP3D>'
    OPEN(UNIT=1, FILE=filename)
    DO k=1, this%kmax
        DO j=1, this%jmax
            READ(1,*) (this%mapdata(i,j,k), i=1, this%imax)
        END DO
    END DO
END SUBROUTINE

SUBROUTINE print_map3d(this)
!
! map3d型データの内容を表示します。
!
    TYPE (map3d), INTENT(IN) :: this
    INTEGER :: i, j, k
    WRITE(*,*) '<PRINT_MAP3D>'
    WRITE(*,*) '3D Map: '
    WRITE(*,*) 'imax=', this%imax, 'jmax=', this%jmax,
    &           'kmax=', this%kmax
    do k=1, this%kmax
        WRITE(*,*) 'k=', k
    end do

```

Jul 18 2002 14:57

calc_keff.f

Page 4

```

        do j=1, this%jmax
          WRITE(*,*) (this%mapdata(i,j,k), i=1, this%imax)
        enddo
      enddo
    END SUBROUTINE

END MODULE

MODULE eigenvalue_module
!
! 固有値のデータ構造やそれに関連する操作を定義する
! モジュールです。
!
! 単なるスカラー量なので、いちいち定義する必要はないかもしれません
! が、このようにしておくことで、他のデータ型と全く同様に取り扱え
! ます。また、ここでは、単精度で定義していますが、ここでの実装
! を倍精度に変更しても、基本的に、他の場所を変更する必要が生じ
! ません（最小限の変更で済みます）。
!
  TYPE eigenvalue
    REAL :: keff
  END TYPE

CONTAINS

  SUBROUTINE init_eigenvalue(this)
!
! データを初期化します。
!
    TYPE(eigenvalue), INTENT(OUT) :: this
    WRITE(*,*) '<INIT_EIGENVALUE>'
    this%keff = 0.0
  END SUBROUTINE

  SUBROUTINE delete_eigenvalue(this)
!
! (ここでもメモリを解放したいのですが、配列でない場合のメモリの
! 解放の方法が分かりませんでした。大きなメモリを必要とする配列
! ので、大きな問題にはならないと思いますが、今後の課題です。)
!
    TYPE(eigenvalue), INTENT(INOUT) :: this
    WRITE(*,*) '<DELETE_EIGENVALUE>'
  END SUBROUTINE

  SUBROUTINE print_eigenvalue(this)
!
! eigenvalue型データの内容を表示します。
!
    TYPE(eigenvalue), INTENT(INOUT) :: this
    WRITE(*,*) '<PRINT_EIGENVALUE>'
    WRITE(*,*) 'Keff =', this%keff
  END SUBROUTINE

END MODULE

MODULE diffusion_module
!
! 拡散計算に関するデータ構造やそれに関連する操作を定義する
! モジュールです。
!
  USE macro_xsec_module
  USE map3d_module
  USE neutron_flux_module
  USE eigenvalue_module
!
! 拡散計算で、マクロ断面積、計算体系、中性子束、固有値を

```

Jul 18 2002 14:57

calc_keff.f

Page 5

```

! 取り扱うことを宣言します。
!

CONTAINS

  SUBROUTINE calc_diff3d(mac, map, flx, keff)
  ! データを初期化します。
  !
  TYPE(macro), INTENT(IN) :: mac           ! 入力：マクロ断面積
  TYPE(map3d), INTENT(IN) :: map           ! 入力：計算体系
  TYPE(flux), INTENT(OUT) :: flx            ! 出力：中性子束
  TYPE(eigenvalue), INTENT(OUT) :: keff      ! 出力：実効増倍率
  INTEGER :: i
  WRITE(*,*) '<CALC_DIFF3D>'
  WRITE(*,*) 'Warning: Not Implemented'
  WRITE(*,*) '           Dummy results are returned.'
  DO i=1,flx%ngroup
    flx%flux(i) = 1.0E+15
  END DO
  keff%keff = 0.99999
  !
  ! このサンプルプログラムでは、拡散計算は実装されていません。
  ! データが変更される確認するために、中性子束として1.0E+15、
  ! 実効増倍率として0.99999を返すようにしてあります。
  !
  ! 実際には、mac, mapで入力されたデータを、松村リーダの提案
  ! されたプログラムのDIF3DEサブルーチンに受渡して呼び出し、
  ! 得られた結果を、flx, keffに代入する、という操作を実装する
  ! ことになると思います。
  !
END SUBROUTINE

END MODULE

PROGRAM main
  !
  ! メインプログラム
  !
  ! ここで、計算の制御を行います。Fortranベースで計算の制御を行う
  ! ことになります。
  !

  USE macro_xsec_module      ! 利用したいモジュールを呼び出します。
  USE neutron_flux_module
  USE map3d_module
  USE diffusion_module

  TYPE (macro) :: mac          ! 利用するデータの変数名を宣言します。
  TYPE (flux) :: flx
  TYPE (map3d) :: map
  TYPE (eigenvalue) :: keff
  INTEGER, PARAMETER :: ng=5
  INTEGER, PARAMETER :: imax=3, jmax=3, kmax=3

  CALL init_macro(mac, ng)      ! データを初期化します。
  CALL init_map3d(map, imax, jmax, kmax)
  CALL init_flux(flx, ng)
  CALL init_eigenvalue(keff)

  CALL read_map3d(map, 'mapdata.inp') ! 必要に応じてデータをファイルから
                                     ! 読み込みます。

  CALL print_macro(mac)         ! データの内容を表示します。
  CALL print_map3d(map)         ! (何度でも呼び出し可能)

```

Jul 18 2002 14:57

calc_keff.f

Page 6

```
CALL print_flux(flx)
CALL print_eigenvalue(keff)

CALL calc_diff3d(mac, map, flx, keff) ! 炉心計算（拡散計算）の実行

CALL print_flux(flx)                  ! 計算結果を表示します。
CALL print_eigenvalue(keff)

CALL delete_macro(mac)               ! 不要になったデータのメモリを解放します。
CALL delete_flux(flx)                ! メモリを再利用できるので、この後に他の
CALL delete_map3d(map)               ! 計算を続けていくことが可能だと思います。

END PROGRAM
```

Jul 18 2002 14:57

mapdata.inp

Page 1

```
1 2 3 /
4 5 6 /
7 8 9 /
11 12 13 /
14 15 16 /
17 18 19 /
21 22 23 /
24 25 26 /
27 28 29 /
```

Jul 18 2002 14:57

output.txt

Page 1

```
<INIT_MACRO>
<INIT_MAP3D>
<INIT_FLUX>
<INIT_EIGENVALUE>
<READ_MAP3D>
<PRINT_MACRO>
Number of Energy Group: 5
Diffusion Coefficient:
 0.000000 0.000000 0.000000 0.000000 0.000000
Absorption:
 0.000000 0.000000 0.000000 0.000000 0.000000
Production:
 0.000000 0.000000 0.000000 0.000000 0.000000
<PRINT_MAP3D>
3D Map:
imax= 3 jmax= 3 kmax= 3
k= 1
 1 2 3
 4 5 6
 7 8 9
k= 2
 11 12 13
 14 15 16
 17 18 19
k= 3
 21 22 23
 24 25 26
 27 28 29
<PRINT_FLUX>
Neutron Flux:
 0.000000 0.000000 0.000000 0.000000 0.000000
<PRINT_EIGENVALUE>
Keff = 0.000000
<CALC_DIFF3D>
Warning: Not Implemented
        Dummy results are returned.
<PRINT_FLUX>
Neutron Flux:
 1.000000E+15 1.000000E+15 1.000000E+15 1.000000E+15 1.000000E+15
<PRINT_EIGENVALUE>
Keff = 0.999990
<DELETE_MACRO>
<DELETE_FLUX>
<DELETE_MAP3D>
```

付録 2-3 コマンド型のシステム案 (松村)

1. ソース

c 炉物理共用コードシステム WP コード仕様検討用の叩き台案 2001.12
c 電中研 松村

c このプログラムは Compaq Visual Fortran 6.5 (FORTRAN 95) で作成しています。

c FORTRAN 90 以上の機能 (動的メモリー確保など) を利用していますので、

c FORTRAN 77 のコンパイラーなどでは利用できません。

c メインルーチン

c ここではプログラム (subroutine) の制御のみを行います。

c この検討例では 3 次元の単純な拡散モデルを使用しますのでメインルーチンでは

c 他のプログラムとの協調が必要となる可能性のある物質領域毎の 3 次元の多群の

c 中性子束(FLXR) および領域体系、マクロ断面積など基本的な値のみを認識します。

c

c

```

IMPLICIT DOUBLE PRECISION (A-H, O-Z)

c この検討例ではプログラムの単純化のため、実数は全て倍精度変数とします。
REAL*8, ALLOCATABLE :: DD(:, :, :), SG(:, :, :), SGR(:, :, :), SGNF(:, :, :),
*           FLXR(:, :, :), XXR(:, :, :), YYR(:, :, :), ZZR(:, :, :),
*           DDFEW(:, :, :), SGFEW(:, :, :), SGRFEW(:, :, :), SGNFFE(:, :, :),
*           FLXRFE(:, :, :)
INTEGER, ALLOCATABLE :: MR(:, :, :, :), IXR(:, :, :, :), IYR(:, :, :, :), IZR(:, :, :, :),
*           MR1(:, :, :, :), MR2(:, :, :, :), IGF EW(:, :, :, :), MRF EW(:, :, :, :)

c DD は拡散係数、 SG は吸収断面積、 SGR は除去断面積、 SGNF は  $\gamma$  核分裂断面積
c MR は x, y, z 方向の物質領域番号マップ、 IXR などは x, y, z 方向の領域毎のメッシュ数
c FLXR : 物質領域毎の 3 次元の多群の中性子束、 XXR などは x, y, z 方向の領域幅
c MR1 は 1 次元計算用の物質領域番号マップ、 MR2 は 2 次元用 (未使用)
c IGF EW はマクロ断面積のエネルギー縮約テーブル、 MRF EW は物質領域縮約テーブル
c DDFEW は縮約後の拡散係数。 SGFEW, SGRFEW, SGNFFE は同様の吸収、除去、  $\gamma$  核分裂断面積

CHARACTER*80 string, string0
INTEGER KEY(3)

C 標準入出力機番 UNIT=5, 6
DATA INU, IOU / 5, 6 /
C 初期値のリセット
DATA IXRM, IYRM, IZRM, IGM, IGMF, MRM, MRMF / 0, 0, 0, 0, 0, 0, 0, 0 /
DATA ICASE, IBURN, KEY / 0, 0, 0, 0, 0, 0 /

C 入力ファイルを "reactor.inp" と固定しているのは PC 上の便利のためで、
C 実際には "3DIF3B2 < reactor.inp" などの使い方になると思います。
OPEN(INU, FILE="reactor.inp")
c ファイル出力したい場合は、入力 "reactor.inp" の一行目は必ず "OUTFILE" とします。
C それ以外ではスクリーンに出力されます。
c メインの write 出力ファイルを規定します。
READ(INU, '(A80)', end=1001) string0
if(string0(1:7).eq."OUTFILE") then
read(INU,*) string
L=LEN_TRIM(string)
```

```

OPEN(IOU,FILE=string(1:L))
  write(IOU,'(A80)') string0
  write(IOU,*) "outputfile :",string(1:L)
  endif
  write(IOU,'(A80)') string0
1 READ(INU,'(A80)',end=1001) string
  if(string(1:1).eq."c".or.string(1:1).eq."C".
  * or.string(1:1).eq." ") go to 1
c 入力の最初のカラムが"c", "C", " "ならコメント行として無視します。
  write(IOU,'(A80)') string
c 入力の最初のカラムが"m"または"M"ならメモ行として出力のみを行います。
  if(string(1:1).eq."m".or.string(1:1).eq."M") go to 1
c 入力が"ENDCAL"なら計算を終了します。
  if(string(1:6).eq."ENDCAL") go to 1001

c 以降にメインプログラムでのコマンド判定とその処理を示します。
ccccccccccccccccccccccccccc 3D Diff.
  if(string(1:10).eq."DIF3D_PREP") then
CC    dif3d 計算のためのメモリー領域確保のための最初の INPUT
c memory allocation data
  READ(INU,*) IXRM, IYRM, IZRM, IGM
  write(IOU,'(A17,4I5)') " Reg. & Energy G.", IXRM, IYRM, IZRM, IGM

C   IXRM : x 方向の領域の数、IYRM, IZRM : y, z 方向、IGM : エネルギ一群数
  ALLOCATE (IXR(IXRM), IYR(IYRM), IZR(IZRM), MR(IXRM, IYRM, IZRM))
  ALLOCATE (XXR(IXRM), YYR(IYRM), ZZR(IZRM))

  IXR=1
  IYR=1
  IZR=1
  MR=1
  XXR=1.0
  YYR=1.0
  ZZR=1.0
c XXR: x 方向の領域の幅 単位 : cm、YYR、ZZR : y、z 方向
c MR:x, y, z 方向の物質領域番号マップ
  call dif3d1(IXR, IYR, IZR, XXR, YYR, ZZR, MR, IXRM, IYRM, IZRM,
  *           IX, IY, IZ, MRM, INU, IOU)
c dif3d1 の機能 3 次元拡散計算の領域準備
c 領域毎のメッシュ数 (IXRM 等)、領域幅 (XXR 等)、MR を read し、
c x 方向のメッシュ数 IX、y、z 方向 : IY, IZ および物質領域番号の最大値
c MRM を返す。
  ALLOCATE (FLXR(MRM, IGM))
  FLXR=0.0
c FLXR : 物質領域毎の 3 次元多群の中性子束
  ALLOCATE (DD(MRM, IGM), SG(MRM, IGM),
  *           SGR(MRM, IGM), SGNF(MRM, IGM))
c マクロ断面積 DD ; 拡散係数、SG : 吸収断面積 (除去断面積を含まない)

```

c SGR:除去断面積, SGNF: ν 核分裂断面積

```
DD=1.0
SG=0.0
SGR=0.0
SGNF=0.0
go to 1
endif
```

---途中省略---

```
cccccccccccccccccc MACRO Input
if(string(1:10). eq. "SUB_MACROI") then
call MACROI(IGM, MRM, DD, SG, SGR, SGNF, INU, IOU)
c マクロ断面積を read する。
go to 1
endif
```

---途中省略---

```
ccccccccccc RESET 3D Diff.
if(string(1:12). eq. "RESET_DIF3DE") then
c 3次元拡散計算で使用したメモリーを解放する。
DEALLOCATE (IXR, IYR, IZR, MR)
DEALLOCATE (XXR, YYR, ZZR)
go to 1
endif
```

---途中省略---

```
ccccccccccc RESET MACRO
if(string(1:11). eq. "RESET_MACRO") then
c 使用したマクロ断面積のメモリーを解放する。
DEALLOCATE (DD, SG, SGR, SGNF, FLXR)
go to 1
endif
```

```
ccccccccccc RESET FEW_MACRO
if(string(1:12). eq. "RESET_FEW_MACRO") then
c 使用した縮約後のマクロ断面積のメモリーを解放する。
DEALLOCATE (DDFEW, SGFEW, SGRFEW, SGNFE, FLXRF)
DEALLOCATE (IGFEW, MRFEW)
go to 1
endif
```

```
cccccccccccccccccc CONDENSE Input
if(string(1:9). eq. "XCONDENSE") then
ALLOCATE(IGFEW(IGM), MRFEW(MRM))
call CONDP(IGFEW, MRFEW, IGMF, MRMF, IGM, MRM, INU, IOU)
```

c 縮約テーブル (IGFEW : エネルギー群, MRFEW : 物質領域) を読み込み、それらの最大値
c IGMF, MRMF を求める。

```

        ALLOCATE (DDFEW(MRMF, IGMF), SGFEW(MRMF, IGMF),
*           SGRFEW(MRMF, IGMF), SGNFFE(MRMF, IGMF), FLXRFE(MRMF, IGMF))
        DDFEW=0.0
        SGFEW=0.0
        SGRFEW=0.0
        SGNFFE=0.0
        FLXRFE=0.0
        call CONDEN(IGFEW, MRFEW, DD, SG, SGR, SGNF, DDFEW, SGFEW, SGRFEW,
*           SGNFFE, FLXR, FLXRFE, IGM, MRM, IGMF, MRMF, INU, IOU)
        go to 1
        endif

```

cccccccccccccccccccccc CON_TO_MACRO Input

if(string(1:13). eq. "XCON_TO_MACRO") then

c 縮約されたマクロ断面積の値を次の計算のマクロ断面積に移動する。

C ここでは次の段階の計算を前提としていますので、縮約後のエネルギー群数(IGMF)は
C 次の段階の計算でのエネルギー群数(IGM)と同じになっている必要があります。

```

        IF(IGM.NE.IGMF) THEN
        WRITE(IOU,*) "CON_TO_MACRO CAN NOT EXCED DUE TO IGM.NE.IGMF",
*   IGM, IGMF
        STOP
        ENDIF
        call CTOMAC(DD, SG, SGR, SGNF, DDFEW, SGFEW, SGRFEW,
*           SGNFFE, MRMF, IGM, MRM, INU, IOU)
        go to 1
        endif

```

cccccccccccccccccccccc PRN_CON_MACRO Input

if(string(1:13). eq. "PRN_CON_MACRO") then

c 縮約されたマクロ断面積の値を印刷する。

```

        call PRNCONM(DDFEW, SGFEW, SGRFEW,
*           SGNFFE, MRMF, IGMF, INU, IOU)
        go to 1
        endif

```

cccccccccccccccccccccc PRN_MACRO Input

if(string(1:9). eq. "PRN_MACRO") then

c マクロ断面積の値を印刷する。

```

        call PRNCONM(DD, SG, SGR,
*           SGNF, MRM, IGM, INU, IOU)
        go to 1
        endif

```

cccccccccccccccccccccc DEF_PUT_FILE Input

if(string(1:12). eq. "DEF_PUT_FILE") then

c ファイル出力の準備を行う。

```

READ(INU,*) IOUF, ICASE
  WRITE(IOU,*) "PUT FILE NO(IOUF), CAL. CASE NO(ICASE)", IOUF, ICASE
C IOUF はファイル出力する機番、ICASE は計算全体のケース番号（任意）
  readINU,*)
  string
  L=LEN_TRIM(string)
  OPEN(IOUF,FILE=string(1:L))
  WRITE(IOU,*) "PUT FILE NAME :", string(1:L)

  go to 1
endif

cccccccccccccccccccc PUT_CON_MACRO Input
  if(string(1:13).eq."PUT_CON_MACRO") then
c 縮約されたマクロ断面積の値をファイル出力する。
  IDATA=1000
  CALL PUTCONM(DDFEW, SGFEW, SGRFEW,
*      SGNFFE, MRMF, IGMF, IDATA, ICASE, IBURN, KEY, INU, IOUF)
  go to 1
endif

  write(IOU,*) " not defined command in main"
  go to 1
1001 write(IOU,*) " end of input "
  stop
end

```

---途中省略---

```

subroutine PUTCONM(DDFEW, SGFEW, SGRFEW,
*      SGNFFE, MRMF, IGMF, IDATA, ICASE, IBURN, KEY, INU, IOUF)
c PUTCONM の機能
c 縮約されたマクロ断面積の値をファイル出力する。
  IMPLICIT DOUBLE PRECISION (A-H, 0-Z)
  REAL*8 DDFEW(MRMF, IGMF), SGFEW(MRMF, IGMF), SGRFEW(MRMF, IGMF),
*      SGNFFE(MRMF, IGMF)
  INTEGER KEY(3)
C 任意の KEY (数字) を 3 個指定可能 (例えば温度、ボイド率など)。
C 指定が不要なら ' /' などと入力。
C デフォルト値 0, 0 が出力される。
  readINU,*) KEY

C 制御カード+データの構成
C 制御カード は データの数、データの形式 (0: 文字、1 : 整数、2 : 実数 など)、
C データの番号 (縮約群のマクロの拡散係数(DDFEW)は 1000、
C 吸収断面積(SGFEW)は 1001、など)
C ICASE: 計算ケース番号 (計算全体の番号、任意)、

```

C IBURN : 燃焼度など (任意)

C KEY: 任意の KEY データを 3 個

```
WRITE( IOUF, '(8I10)' ) IGMF*MRMF, 2, IDATA, ICASE, IBURN, KEY
WRITE( IOUF, '(1P6E12.5)' ) ((DDFEW(I, IG), IG=1, IGMF), I=1, MRMF)
WRITE( IOUF, '(8I10)' ) IGMF*MRMF, 2, IDATA+1, ICASE, IBURN, KEY
WRITE( IOUF, '(1P6E12.5)' ) ((SGFEW(I, IG), IG=1, IGMF), I=1, MRMF)
WRITE( IOUF, '(8I10)' ) IGMF*MRMF, 2, IDATA+2, ICASE, IBURN, KEY
WRITE( IOUF, '(1P6E12.5)' ) ((SGRFEW(I, IG), IG=1, IGMF), I=1, MRMF)
WRITE( IOUF, '(8I10)' ) IGMF*MRMF, 2, IDATA+3, ICASE, IBURN, KEY
WRITE( IOUF, '(1P6E12.5)' ) ((SGNFFE(I, IG), IG=1, IGMF), I=1, MRMF)
      RETURN
END
```

2. 入力例

OUTFILE 計算結果をファイル出力する場合は、最初の行は必ず OUTFILE とする。

reactor.out

C 上記は出力ファイル名 c, C はコメント行(出力されない)、m, M はメモ行(出力される)

c コメント行とメモ行はコマンド行(文字入力)の前なら何時でも入力可能

またコマンド行の前の 1 文字目が空白(英字で)の行の場合もコメント行と見なします。

M 炉物理共用コードシステム構築WP コード仕様検討用の叩き台案 入力データ

DIF1D_PREP 1 次元拡散計算準備*****最初の計算

3 3 / IXRM, IGM x 方向の領域の数、エネルギー群数(IGM)

10 10 10 / (IXR(I), I=1, IXRM) x 方向の領域毎のメッシュ数

REGION

2 1 2 / (MR(I, J, K), I=1, IXRM) x 方向の物質組成番号マップ

10.0 10.0 10.0 / (XXR(I), I=1, IXRM) x 方向の領域毎の幅 (cm)

SUB_MACROI

1 / 物質領域番号 燃料領域

1. 2516741, 0.6571641, 0.2286707 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群・・

0.00314569, 0.023063, 0.085576 / (SG(IR, IG), IG=1, IGM) 吸収断面積(除去断面積含まず)

0.047528, 0.092204, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積

0.0044881, 0.010177, 0.13506 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

2 / 水領域

1. 3215452, 0.5811251, 0.1317731 / DM1, DM2, DM3 DIF. OF MODERATOR

0.00025959, 0.0006416, 0.018393 / SGM1, SGM2, SGM3 SGA OF MODERATOR

0.080984, 0.17482, 0.0 / SGRM1, SGRM2, SGRM3 SG-REMOVAL OF MODERATOR

0.0, 0.0, 0.0 / SGNFM1, SGNFM2, SGNFM3 SG-NUFIS. OF MODERATOR

0 / マクロ断面積 入力終了の合図 (LE.0)

SUB_DIF1DE 1 次元拡散計算実行

c PARAM の説明

c ITMAX: Inner Iteration 回数制限、ITOMAX: Outer Iteration 回数制限,

c OMEGA: Inner 加速因子(1なら加速無し), EPS: Inner 収束判定

c OMEGAS: Outer 加速因子(1なら加速無し), EPSS: Outer 収束判定

c PARAM の入力が無い場合はデフォルト値

c 500, 100, 1.3, 1.0E-4, 1.3, 1.0E-4、境界条件は全部反射となる。

PARAM 拡散計算制御パラメータ

500 100 1.3 1.0E-5 1.3 1.0E-5 / ITMAX, ITOMAX, OMEGA, EPS, OMEGAS, EPSS

0.0 0.0 / ALFL, ALFR

C 1.0 1.0 / ALFL, ALFR

c 0.0 0.0 / ALFL, ALFR

c 上記は境界条件(0:反射、1:零値、0-1の間の数値:傾き(J/φ))

c x 方向一+

c PRINT_CON を入力しなければ、デフォルトは全て1

PRINT_CON IFL1: outer 収束状況、IFL2: inner 収束状況、IFL3: inner 収束フラックス

1 1 0 0 0 / IFL1, IFL2, IFL3, IFL4, IFL5 計算過程出力フラグ

EXEC 1 次元拡散計算実行

PRINT_FLX 1次元フラックス出力 このコマンドは何回も入力可
 1 0 / IX1, IX2 x 方向 IX1~IX2 メッシュの範囲 但し、IX2 <= 0 なら最大値(IX)
 END_DIF1DE 拡散計算終了

XCONDENSE マクロ断面積の縮約
 1 1 2 / (IGFEW(I), I=1, IGM) エネルギー群縮約
 1 1 1 / (MRFEW(I), I=1, MRM) 領域縮約
 PRN_CON_MACRO 縮約後のマクロ断面積の印刷

DEF_PUT_FILE ファイル出力のための定義
 7 131225 / IOUF(ファイル出力機番), ICASE(計算全体のケース番号、任意の数字)
 REACTOR.PUT / 出力ファイル名
 PUT_CON_MACRO 縮約後のマクロ断面積のファイル出力
 / KEY(1), KEY(2) を入力可能。任意の数字

RESET_DIF1DE 1次元拡散計算で使用したメモリーを解放する。
 RESET_MACRO 使用したマクロ断面積のメモリーを解放する。

DIF3D_PREP 3次元拡散計算準備*****次の計算
 1 1 1 2 / IXRM, IYRM, IZRM, IGM x, y, z 方向の領域の数、エネルギー群数(IGM)
 10 / (IXR(I), I=1, IXRM) x 方向の領域毎のメッシュ数
 10 / (IYR(J), J=1, IYRM) y 方向
 10 / (IZR(K), K=1, IZRM) z 方向
 REGION
 1 / (MR(I, J, K), I=1, IXRM) /DO J=1, IYRM /DO K=1, IZRM x, y, z 方向の物質組成番号マップ
 10.0 / (XXR(I), I=1, IXRM) x 方向の領域毎の幅 (cm)
 10.0 / (YYR(J), J=1, IYRM) y 方向
 10.0 / (ZZR(K), K=1, IZRM) z 方向

XCON_TO_MACRO 縮約されたマクロを次の計算のマクロ断面積として使用する。
 1 1 / IIFEW, II 縮約されたマクロ(移動元)、新しいマクロ断面積の番号(移動先)
 0 0 / IIFEW <=0 終了

C SUB_MACROI(マクロ断面積のカード入力)も可能。また、物質領域毎の混在の定義も可能。

PRN_MACRO

SUB_DIF3DE 3次元拡散計算実行
 c PARAM の説明
 c ITMAX : Inner Iteration 回数制限、ITOMAX : Outer Iteration 回数制限,
 c OMEGA: Inner 加速因子(1なら加速無し), EPS : Inner 収束判定
 c OMEGAS : Outer 加速因子(1なら加速無し), EPSS : Outer 収束判定
 c PARAM の入力が無い場合はデフォルト値
 c 500, 100, 1.3, 1.0E-4, 1.3, 1.0E-4、境界条件は全部反射となる。
 PARAM 拡散計算制御パラメータ
 500 100 1.3 1.0E-5 1.3 1.0E-5 /ITMAX, ITOMAX, OMEGA, EPS, OMEGAS, EPSS
 0.0 0.0 0.0 0.0 0.0 0.0 / ALFL, ALFR, ALFB, ALFT, ALFZ0, ALFZ1

```

C 1.0 1.0 1.0 1.0 1.0 / ALFL, ALFR, ALFB, ALFT, ALFZ0, ALFZ1
c 0.0 0.0 0.0 0.0 0.0 / ALFL, ALFR, ALFB, ALFT, ALFZ0, ALFZ1
c 上記は境界条件 (0: 反射、1: 零値、0-1 の間の数値: 傾き (J/φ))
c x 方向一+、y 方向一+、z 方向一+の順

c PRINT_CON を入力しなければ、デフォルトは全て 1
PRINT_CON IFL1:outer 収束状況、IFL2:inner 収束状況、IFL3:inner 収束フラックス
1 1 1 0 0 / IFL1, IFL2, IFL3, IFL4, IFL5 計算過程出力フラグ
EXEC 3 次元拡散計算実行
PRINT_FLX 3 次元フラックス出力 このコマンドは何回も入力可
1 0 1 0 5 5 / IX1, IX2, IY1, IY2, IZ1, IZ2 x 方向 IX1~IX2 メッシュの範囲、y 方向・・。z 方向
END_DIF3DE 拡散計算終了

RESET_DIF3DE 3 次元拡散計算で使用したメモリーを解放する。
RESET_MACRO 使用したマクロ断面積のメモリーを解放する。

M IAEA benchmark problem
DIF3D_PREP 3 次元拡散計算準備*****IAEA ベンチマーク問題 3 番目の計算
9 9 4 2 / IXRM, IYRM, IZRM, IGM x, y, z 方向の領域の数、エネルギー群数(IGM)
2 8*4 / (IXR(I), I=1, IXRM) x 方向の領域毎のメッシュ数
2 8*4 / (IYR(J), J=1, IYRM) y 方向
5 10 5 5 / (IZR(K), K=1, IZRM) z 方向
REGION (MR(I, J, K), I=1, IXRM) /DO J=1, IYRM /DO K=1, IZRM x, y, z 方向の物質組成番号マップ
4 4 4 4 4 4 4 4 4 4 / Z=1 BOTTOM
4 4 4 4 4 4 4 4 4 /
4 4 4 4 4 4 4 4 4 /
4 4 4 4 4 4 4 4 6 /
4 4 4 4 4 4 4 4 6 /
4 4 4 4 4 4 4 6 6 /
4 4 4 4 4 6 6 6 6 /
3 2 2 2 3 2 2 1 4 / Z=2
2 2 2 2 2 2 2 1 4 /
2 2 2 2 2 2 1 1 4 /
2 2 2 2 2 1 1 4 4 /
3 2 2 2 3 1 1 4 6 /
2 2 2 2 1 1 4 4 6 /
2 2 1 1 1 4 4 6 6 /
1 1 1 4 4 4 6 6 6 /
4 4 4 4 6 6 6 6 6 /
3 2 2 2 3 2 2 1 4 / Z=3
2 2 2 2 2 2 2 1 4 /
2 2 3 2 2 2 1 1 4 /
2 2 2 2 2 1 1 4 4 /
3 2 2 2 3 1 1 4 6 /
2 2 2 2 1 1 4 4 6 /

```

```

2 2 1 1 1 4 4 6 6 /
1 1 1 4 4 4 6 6 6 /
4 4 4 4 6 6 6 6 6 /
5 4 4 4 5 4 4 4 4 / Z=4 TOP
4 4 4 4 4 4 4 4 4 /
4 4 5 4 4 4 4 4 4 /
4 4 4 4 4 4 4 4 4 /
5 4 4 4 5 4 4 4 6 /
4 4 4 4 4 4 4 4 6 /
4 4 4 4 4 4 4 6 6 /
4 4 4 4 4 4 6 6 6 /
4 4 4 4 6 6 6 6 /
10.0 8*20.0 /(XXR(I), I=1, IXRM) x 方向の領域毎の幅 (cm)
10.0 8*20.0 /(YYR(J), J=1, IYRM) y 方向
20.0 260.0 80.0 20.0 /(ZZR(K), K=1, IZRM) z 方向

```

SUB_MACROI

1 / 物質領域番号 FUEL1

1.5, 0.4 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 0.01, 0.08 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.02, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.135 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

2 / 物質領域番号 FUEL2

1.5, 0.4 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 0.01, 0.085 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.02, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.135 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

3 / 物質領域番号 FUEL2+ROD

1.5, 0.4 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 0.01, 0.13 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.02, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.135 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

4 / 物質領域番号 Ref

2.0, 0.3 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 0.0, 0.01 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.04, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.0 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

5 / 物質領域番号 Ref+Rod

2.0, 0.3 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 0.0, 0.055 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.04, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.0 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

6 / 物質領域番号 black

2.0, 0.3 / (DD(IR, IG), IG=1, IGM) 拡散係数 1群、2群···
 10.0, 10.0 / (SG(IR, IG), IG=1, IGM) 吸收断面積(除去断面積含まず)
 0.04, 0.0 / (SGR(IR, IG), IG=1, IGM) 除去断面積
 0.0, 0.0 / (SGNF(IR, IG), IG=1, IGM) ν核分裂断面積

0 / マクロ断面積 入力終了の合図 (LE. 0)

SUB_DIF3DE 3次元拡散計算実行

PARAM 拡散計算制御パラメータ

500 100 1.3 1.0E-5 1.3 1.0E-5 /ITMAX, ITOMAX, OMEGA, EPS, OMEGAS, EPSS

0.0 1.0 0.0 1.0 0.0 0.0 / ALFL, ALFR, ALFB, ALFT, ALFZ0, ALFZ1

c 上記は境界条件 (0: 反射、1: 零値、0-1 の間の数値: 傾き (J/ϕ))

c x 方向一+、y 方向一+、z 方向一+の順

c PRINT_CON を入力しなければ、デフォルトは全て 1

PRINT_CON IFL1 : outer 収束状況、IFL2 : inner 収束状況、IFL3 : inner 収束フラックス

1 1 0 0 0 / IFL1, IFL2, IFL3, IFL4, IFL5 計算過程出力フラグ

EXEC 3次元拡散計算実行

PRINT_FLX 3次元フラックス出力 このコマンドは何回も入力可

1 0 1 0 10 10 / IX1, IX2, IY1, IY2, IZ1, IZ2 x 方向 IX1～IX2 メッシュの範囲、y 方向・。z 方向

END_DIF3DE 拡散計算終了

ENDCAL 計算終了

3. 出力例

```

OUTFILE 計算結果をファイル出力する場合は、最初の行は必ずOUTFILEとする。それ以・
outputfile :reactor.out
OUTFILE 計算結果をファイル出力する場合は、最初の行は必ずOUTFILEとする。それ以・
M 炉物理共用コードシステム構築WP コード仕様検討用の叩き台案 入力データ
DIF1D_PREP 1 次元拡散計算準備*****最初の計算
Reg. & Energy G. 3 3
X mesh No. :
10 10 10
REGION
Reg. Material No. :
2 1 2
MRM(MAX. Macro No.) : 2
X Mesh width :
1.00000E+01 1.00000E+01 1.00000E+01
SUB_MACROI
Macro reg. : 1
Diff. coef :
1.25167E+00 6.57164E-01 2.28671E-01
Capture XS :
3.14569E-03 2.30630E-02 8.55760E-02
Removal XS :
4.75280E-02 9.22040E-02 0.00000E+00
Nu Fission :
4.48810E-03 1.01770E-02 1.35060E-01
Macro reg. : 2
Diff. coef :
1.32155E+00 5.81125E-01 1.31773E-01
Capture XS :
2.59590E-04 6.41600E-04 1.83930E-02
Removal XS :
8.09840E-02 1.74820E-01 0.00000E+00
Nu Fission :
0.00000E+00 0.00000E+00 0.00000E+00
Macro reg. : 0
SUB_DIF1DE 1 次元拡散計算実行
GEOM. SIZE X, TOTAL VOL 3.00000E+01 3.00000E+01
Mesh total No. x 30
PARAM 拡散計算制御パラメータ
Diff. Param: 500 100 1.30000E+00 1.00000E-05 1.30000E+00 1.00000E-05
Bounbary Co: 0.00000E+00 0.00000E+00
PRINT_CON IFL1:outer 収束状況、IFL2:inner 収束状況、IFL3:inner 収束フラッ
Print FLG : 1 1 0 0 0
EXEC 1 次元拡散計算実行

```

---途中省略---

K-EFF 0.826707092480410
 OUTER ITERATION CONVERGED IT= 8k-eff 0.826707092480410
 PRINT_FLX 1次元フラックス出力 このコマンドは何回も入力可
 Print Flux : 1 0
 FLX (OUTER CONVERGED) IG= 1
 1.25561E-01 1.33280E-01 1.49192E-01 1.74277E-01 2.10075E-01 2.58787E-01

---途中省略---

END_DIF1DE 拡散計算終了
 XCONDENSE マクロ断面積の縮約
 CONDENSE ENERGY(IGMF) : 2
 CONDENSE MATERIAL REGION(MRMF) : 1
 PRN_CON_MACRO 縮約後のマクロ断面積の印刷
 MACRO REGION: 1
 Diffusion Coef.
 8.17579E-01 1.43919E-01
 Absorption XS.
 2.97078E-03 3.17735E-02
 Removal XS.
 1.16343E-01 0.00000E+00
 Nu*Fission XS.
 1.46039E-03 2.68992E-02
 DEF_PUT_FILE ファイル出力のための定義
 PUT FILE NO(IOUF), CAL. CASE NO(ICASE) 7 131225
 PUT FILE NAME :REACTOR.PUT
 PUT_CON_MACRO 縮約後のマクロ断面積のファイル出力
 RESET_DIF1DE 1次元拡散計算で使用したメモリーを解放する。
 RESET_MACRO 使用したマクロ断面積のメモリーを解放する。
 DIF3D_PREP 3次元拡散計算準備*****次の計算
 Reg. & Energy G. 1 1 1 2
 X mesh No. :
 10
 Y mesh No. :
 10
 Z mesh No. :
 10
 REGION
 Reg. Material No. : J= 1 K= 1
 1
 MRM(MAX. Macro No.) : 1
 X Mesh width :
 1.00000E+01
 Y Mesh width :
 1.00000E+01
 Z Mesh width :

1. 00000E+01
 XCON_TO_MACRO 縮約されたマクロを次の計算のマクロ断面積として使用する。
 From, To of MACRO Reg. 1 1
 From, To of MACRO Reg. 0 0
 PRN_MACRO
 MACRO REGION: 1
 Diffusion Coef.
 8.17579E-01 1.43919E-01
 Absorption XS.
 2.97078E-03 3.17735E-02
 Removal XS.
 1.16343E-01 0.00000E+00
 Nu*Fission XS.
 1.46039E-03 2.68992E-02
 SUB_DIF3DE 3次元拡散計算実行
 GEOM. SIZE X, Y, Z, TOTAL VOL 1.00000E+01 1.00000E+01 1.00000E+01 1.00000E+03
 Mesh total No. x, y, z & max 10 10 10 10
 PARAM 拡散計算制御パラメータ
 Diff. Param: 500 100 1.30000E+00 1.00000E-05 1.30000E+00 1.00000E-05
 Boundary Co: 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
 PRINT_CON IFL1:outer 収束状況、IFL2:inner 収束状況、IFL3:inner 収束フラグ
 Print FLG : 1 1 1 0 0
 EXEC 3次元拡散計算実行
 *****OUTER LOOP***** 1
 *****ENERGY GROUP(IG)***** 1
 CONVERGED IT= 1

---途中省略---

Inner Converged FLX J, K, IG= 10 10 2
 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02
 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02
 K-EFF 0.837753286856947
 OUTER ITARATION CONVERGED IT= 1k-eff 0.837753286856947
 PRINT_FLX 3次元フラックス出力 このコマンドは何回も入力可
 Print Flux : 1 0 1 0 5
 5

---途中省略---

FLX (OUTER CONVERGED) J, K, IG= 10 5 2
 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02
 3.06891E-02 3.06891E-02 3.06891E-02 3.06891E-02
 END_DIF3DE 拡散計算終了
 RESET_DIF3DE 3次元拡散計算で使用したメモリーを解放する。
 RESET_MACRO 使用したマクロ断面積のメモリーを解放する。
 M IAEA benchmark problem

DIF3D_PREP 3次元拡散計算準備*****IAEA ベンチマーク問題 3番目の計算

Reg. & Energy G. 9 9 4 2

X mesh No. :

2 4 4 4 4 4 4 4

---途中省略---

SUB_DIF3DE 3次元拡散計算実行

GEOM. SIZE X, Y, Z, TOTAL VOL 1.70000E+02 1.70000E+02 3.80000E+02 1.09820E+07

Mesh total No. x, y, z & max 34 34 25 34

PARAM 拡散計算制御パラメータ

Diff. Param: 500 100 1.30000E+00 1.00000E-05 1.30000E+00 1.00000E-05

Boundary Co: 0.00000E+00 1.00000E+00 0.00000E+00 1.00000E+00 0.00000E+00 0.00000E+00

PRINT_CON IFL1:outer 収束状況、IFL2:inner 収束状況、IFL3:inner 収束フラグ

Print FLG : 1 1 0 0 0

EXEC 3次元拡散計算実行

---途中省略---

K-EFF 1.02977354292116

OUTER ITERATION CONVERGED IT= 101 k-eff 1.02977354292116

PRINT_FLX 3次元フラックス出力 このコマンドは何回も入力可

Print Flux : 1 0 1 0 10

10

---途中省略---

FLX (OUTER CONVERGED) J, K, IG= 34 10 2

4.92607E-07 4.89856E-07 4.84006E-07 4.74358E-07 4.59848E-07 4.39087E-07

4.10510E-07 3.72795E-07 3.25710E-07 2.71252E-07 2.13775E-07 1.56995E-07

1.03466E-07 5.20693E-08 6.26113E-11 7.69780E-14 1.10259E-16 2.07417E-18

1.31096E-18 8.80355E-19 5.36693E-19 2.55300E-19 6.12011E-21 1.28040E-22

2.26764E-24 3.56072E-26 5.30934E-28 7.97416E-30 1.24190E-31 1.99388E-33

3.22773E-35 5.17533E-37 8.16351E-39 1.26015E-40

END_DIF3DE 拡散計算終了

ENDCAL 計算終了

end of input

付録 3-1：第1回会合議事録

第1回共用炉物理コードの構築 WP 議事録詳細

日時：2001年11月26日（火） 13:30～17:00

場所：航空会館 705号室

出席者：（11名、順不同）

松村 哲夫（リーダ、電中研）、小林 啓祐（幹事、元京都大学）、別所 泰典（幹事、NUPEC）、池田一三（幹事、三菱重工）、飯島 進（幹事、原研）、山岡 光明（東芝）、横山 賢治（サイクル機構）、内藤 傷孝（NAIS）、日山 伸行（原燃工）、中島 鐵雄（NUPEC）、杉 輝夫（原研）

資料：

WP01-1-1 共用炉物理コードの構築 WP 第1回会合開催にあたって
(炉物理研究委員会 委員長、事務局)

WP01-1-2 共用炉物理コードシステムの構築 WP の進め方について（松村リーダ）

WP01-1-3 三菱のFBR炉心計算方の現状について（池田幹事）

議題：

1. リーダー及び幹事の選出

小林幹事が司会役となって各委員の自己紹介が実施され、議事録作成担当者が決定された。

詳細版担当者 : 日山委員

簡易版担当者（A4版2ページ程度）: 飯島幹事

（メールによる回覧で詳細議事録にコメントを反映する）

次に、小林幹事からリーダ/幹事の選出について以下の説明があり了承された。

- リーダはWPの運営規則に従って炉物理研究委員会委員である松村委員とすること。
- 幹事は飯島委員、池田委員、別所委員、小林委員の4名とすること。
- 初回打合せ(本日)はこれまでの経緯をよく理解している小林幹事がリーダ代行として議事進行役を行なうこととし、次回以降は松村リーダが実施すること。

2. WPの今後の活動計画

資料 WP01-1-1 を用いて炉物理研究委員会からの以下のコメントが飯島幹事より紹介さ

れた。

- ・ 共用炉物理コード像が明確になっていない。
(異なる機関の要望が、ひとつの共用コード像に収斂できるのだろうか)
- ・ 収斂したとして、既存公開コードとの違いはなにか（例：SRAC）
- ・ 予算要求の議論の是非

小林幹事から、コメントに答える形で“コードシステム“のイメージが説明された。

- ・ 資料 WP01-1-1 には、本 WP の行なうべき事項が簡潔明快に示されており、各委員はこの内容を意に留めて置いてほしい。
- ・ 次に「メーカ、研究機関、大学と異なった機関の要望がひとつの共用炉物理コードに収斂できるだろうか」との指摘については、既に昨年の共用炉物理コードシステム特別専門委員会で議論されている。各機関が期待する機能はそれぞれ異なっていて共通点が少ないことから、共通点だけを実現しようとすると、どの機関にとっても不十分なものとなろう。それ故、各機関が希望する共通点のみを実現するのではなく、原則として希望する機能は全て満たすようなコードシステムを目標としたい。よって、ひとつの共用コード像に収斂させる必要はないと思う。
- ・ 計算コード単体の集合体ではなくて、システムの中にコード単体を容易に取り込むこと（システムへの登録）が可能であり、システム内にあるコード群からユーザが自由に選択して解析ストリームを容易に構築できるようなコードシステムを考えている。
- ・ SRAC コードは、昨年度の委員会で何度も議論されているように、作成したデータを他のコードで利用はできるが、SRAC コードに他のコードを組み込むことを目的として作られていない。それ故、SRAC は、ここで考えているコードシステムとは概念が異なる。
- ・ また、コードをシステムへ登録する場合には、ベンチマーク計算による精度検証結果やマニュアルを含むものとしたい。

これに対して各委員のコメントは以下のとおり。

(松村) RIST と違う点はどこか？

(小林) RIST はコード単体 (stand alone) を集めたものに過ぎず、必要なコードの名前を入力するだけで順次走らせることはできない。ここで考えているコードシステムは、コード A からコード B、C へと必要なコード名やデータの種類がユーザによって容易に設定できて“だだだ……と”順番どおりに流れる機能を有するコードシステムを考えている。

(内藤) 国際協力の観点では、日本はいつも”give & take”的”take”的比率が多すぎると言

われている。このようなコードシステムの完成により、"give"の可能性が増大することを希望する。RIST から配布されるコードの長所は、マニュアル関連資料の整備が充実していることである。

- (杉) RIST のコードを利用したいが、私の力不足もあって使いこなせない。本 WP が求めるコードシステムがあれば、自分の PC に搭載して幅広い計算が可能となるので期待している。そのためには、汎用性を持ったコード/データのインターフェイス（コード群を接続するプログラムシステム）が必要である。コードへのアプローチを容易にすることで、情報公開（第3者が追計算を実施できる仕組みとすること）が実現し、反対派の誤解を解くためにも有意義である。
- (横山) ERANOS のコード/データのインターフェイスは LU という特別に開発した言語を使ってコードモジュールやデータを呼び出す機能を有している。（開発コストは不明）ユーザは LU 言語をまるで “input を作成している” という感覚で利用しており、解析ストリームを容易に構築できる。データの受け渡しは、EDL と呼ばれるデータ転送形式（構造体の形）で実施されるため、データの Format を意識しないでよい。
- (内藤) IPSN が開発した臨界評価コードシステム(CRISTAL)は、データのやり取りにかなり工夫されているようである。コード/データのインターフェイス技術として調査の価値があるだろう。なお、PC で動作させる場合には、搭載ソフトのバージョンアップにも対応できる体制を考えておく必要がある。
- (小林) 自分で PC を使って、Dos コマンドやマクロ命令等を利用すれば順次 Fortran プログラムを呼び出して走らせるインターフェイス機能を実現できることを確認している。昨年度の共用炉物理コードシステム特別専門委員会報告書の添付資料 H に具体的なプログラムを載せている。このバッチジョブは、EWS のシェル言語のプログラムと同じことができるはずである。しかし、機種によりこの部分の互換性・が低下すると問題である。より互換性の高い方法が無いかどうかを検討する必要がある。
- (横山) インタフェイス機能に関しては、異なるマシンに対する移植性も検討が必要である。
- (池田) (メーカの立場からすると) コードシステムの理想像として、近似の少ない参照解を算出してくれるものを望んでいる。
- (別所) APWR、ABWR、ガス炉.....とコードシステムの対象範囲とするユーザは広い。BWR では water rod の形状の取扱い等に近似が含まれている。2 日程度の計算時間でも、参照解を算出できるものがほしい。
- (日山) メーカからコードを公開することは困難と考える。入れ物（コードシステム）だけを利用するユーザもあるだろう。

3. コードシステム構築の計画書の作成

松村リーダが資料 WP01-1-2 を用いて、今後の WP の進め方に関連して曖昧と感じる部分を説明した。また電中研で使用しているコード (FLEX_BURN、COM_PWR2) についても併せて紹介があった。以下の各委員のコメントを示す。

- (松村) 共用炉物理コードシステム特別専門委員会報告書にはモンテカルロコードに関連する記載が見当たらない。決定論的手法を対象としているようにも受け取れる。
- (池田) モンテカルロコードは参考解を算出するコードとして位置付けられる。コードシステムに登録するコードとして決定論的コードのみに限定すべきではない。
- (内藤) 遮蔽・臨界安全解析を除外してほしくない。コードシステムに“入れる物（コード）”はできるだけ自由度を持たせた方が良い。
- (松村) 商用パッケージ (DiffPACK) を利用する案もあるが、コスト高 (200 万) である。さらに C++でインターフェイス部分プログラムする必要がある。私としては推薦できない。また、平成 13 年度の活動範囲としてコード開発は無理であろう。平成 14 年度には何らかのプロトタイプができればと考えている。そこで、インターフェイス機能の確認をしたい。私案だが、ORIGEN2 ように、入力カードの 1 行ずつがコマンドになっているような入力形式もインターフェイス機能の候補と考えている。

4. 予算の獲得方法

小林幹事より、予算確保に向けて説得力のある計画書の作成や、登録するコードや入れ物（コードシステム）の調査が必要である旨の説明があった。これに対して以下のコメントが各委員から出された。

- (内藤) 予算確保はいつ頃を想定しているか？今から平成 14 年度確保は困難であろう。
- (小林) しっかりとした説得力のある計画書を作成することが重要。早く作成できれば良いが、何年か遅れても実現できればよいと考えている。本 WP (2 年間) では概略のレベルでもできればよいと思う。予算確保の可能性があるのは政府の予算で、それが使えるのは大学、原研、核燃料サイクル機構等が候補になるだろう。
- (飯島) 予算確保は炉物理研究委員会での審議が必要である。そのためには、コードシステムのビジョンを明確にする必要がある。例えば、メーカコードと JAERI/JNC コード間では断面積データの取扱いに差異があり、インターフェイス確保が困難との認識を持っている。
- (杉) WP01-1-1 の資料に記載されているが、公募型研究の機会もあると思う。コード

を使い易くして情報公開を図ることで、反対派の誤解を解くこともできるので、PA 予算獲得の可能性も考えられる。中高、高専の先生方の中には自分で計算して教材を作成し、生徒の疑問に答えようする熱心な方が居られる。

5. その他

<メーカのコードの現状紹介>

池田幹事が資料 WP01-1-2 を用いて三菱重工での FBR 解析コードの現状紹介を行なった。また、山岡委員から東芝の解析コードの現状について口頭説明があった。各委員から出されたコメントを示す。

(小林) コードシステムに登録できるコードがあるか?

(池田) 公開にはコード検証作業やマニュアル整備作業を行なう必要がある。これらの負担が小さくない。

もんじゅの安全審査を情報公開するとのサイクル機構殿の方針があり、前向きに検討したいと思っている。

(小林) メーカのノウハウ公開を要求するつもりはないので、公開できるものを登録してくれるといい。

<次回会合までの準備>

- ・ 松村リーダ、内藤委員、横山委員には、それぞれ ORIGEN, CRISTAL, ERANOS のコードシステムのインターフェイス機能や異機種 (EWS/PC) の互換性について可能な範囲での調査をお願いする。
- ・ 各委員は、それぞれの機関の持つコードの現状、現在のコードの問題点、公開できるもの、公開したいもの、公開してほしいものをまとめる。
- ・ どのようなコードシステムを作るか、今後必要とするコード、開発が必要な炉物理コードの提案。
- ・ 内藤委員からコードシステムの管理機関として ITBL や大学（東大）が候補になり得るとのコメントがあつたので、飯島幹事に調査のお願いする。
- ・ これらの調査は、12月末までにインターネットを使って情報交換をする。

<次回の予定>

次回会合は 2 月 22 日の 13:30 からとすることに決定した。

-以上-

付録3-2：第2回会合議事録

第2回共用炉物理コードの構築 WP 議事録案

日時：2002年2月22日（金） 13:30～17:00

場所：航空会館 503号室

出席者：（10名、順不同）

松村 哲夫（リーダ、電中研）、小林 啓祐（幹事、元京都大学）、

池田一三（幹事、三菱重工）、飯島 進（幹事、原研）、

山岡 光明（東芝）、横山 賢治（サイクル機構）、

日山 伸行（原燃工）、中島 鐵雄（NUPEC）、杉 輝夫（原研）

中川 正幸（炉物理委員会委員長、原電）

資料：

WP01-2-1 第1回共用炉物理コードの構築 WP 議事録詳細

WP01-2-2 炉物理委員会 運営会議（松村リーダ）

WP01-2-3 欧州炉物理解析システム ERANOS の特徴と次世代解析システムの提案
(横山幹事)

WP01-2-4 第2回共用炉物理コードシステムの構築 WP 資料（横山委員）

WP01-2-5 原研の計算コードシステム（飯島幹事）

WP01-2-6 原燃工の炉物理コードの現状について（日山委員）

WP01-2-7 共用炉物理コードシステムの構築 WP 資料（松村リーダ）

WP01-2-8 原子炉安全解析所における炉心解析コードについて（中島委員）

WP01-2-8' 安全解析所における炉心解析コードの現状と今後必要とするコード（中島委員）

WP01-2-9 東芝のFBR炉物理コードの現状（山岡委員）

WP01-2-10 共用炉物理コードシステムの構築 WP 資料（池田幹事）

WP01-2-11 共用炉物理コードシステムの構築 WP での議論に向けて（松村リーダ）

1. 議事録確認

事前に配布につき、紹介省略。

2. 委員会からの意見紹介（松村リーダ）

日山：質の意味はないか。精度をいうのかあるいは品質管理をいうのか。

中川：信頼性ということ。

3. 欧州炉物理解析システム ERANOS の特徴と次世代解析システムの提案と JNC のコードシステム（横山委員）

小林：JNC の核特性解析システムの場合、各ステップの計算は一連処理としてひとつの計算で行なっているのか。

横山：ステップ毎に結果を確認しつつ、計算している。

小林：前のステップの出力が次のステップの入力となっているのか。

横山：JOINT コードを介して前のステップの出力を次のステップの入力となるように処理している。

飯島：一度の計算の実行で行なうことも可能だが、計算ミスのないように行なっている。

小林：JNC でも ERANOS のように計算できるか。

横山：JUPITER 実験解析用で燃焼計算ができていない。燃焼ルーチンは各コードに必要で、個々に用意する必要がある。

中川：どの単位で作成するのかがポイントである。以前はサブルーチンであったが、その場合バージョンが増える。細かいスキームで作成するとユーザが使いにくい。将来のシステムを考えて決める必要があるが予見することは難しい。

横山：どの単位で作成するかは専門家が経験と予測から決めるしかないと思う。

小林：オブジェクト指向という意味は。

横山：目的とする操作手順を重視するのではなく、操作対象を重視して設計するということ。炉物理解析であれば、計算手順を重視するのではなく、拡散、擾動、燃焼等の機能要素や、中性子束、断面積、体系データ等の必要なデータ構造を、先に準備しておき、後から自由に組み合わせて目的に沿った計算を行なう。

飯島：プログラム間のデータのやり取りはバイナリーかテキストか

横山：ERANOS ではプログラム間のデータのやり取りは EDL という構造体として扱われる。C 言語の構造体と似ており、ポインタを使ってツリー構造を作っている。データ (EDL) のやり取りは、ファイルではなく、コアメモリ上で行われる。コアメモリ上の EDL は、形式を与えることによりファイルに出力できる。

小林：基本はバイナリーであろう。

4. 原研の計算コードシステム（飯島）

小林：一回の計算として処理可能か。

飯島：シェルをそのように与えておけばできる。実際には行なわない。

池田：ガス炉は温度フィードバック、ガス膨張に伴う冷却材質量流量のフィードバックを組み込んだ核計算はできるか。

飯島：現状、できていない。

飯島：システムを考える上でコード間のデータのやり取りをどのようにするか、倍精度・单精度という問題がある。他方、メモリー容量が大きくなり、余裕は大きくなつた。

中川：コンピュータができるることはコンピュータにやらせるのがよい。コンピュータの発展を見込んだシステムを構築するのがよい。

5. 原燃工のコードシステム（日山）

中川：紹介されたコードは通常利用している設計用コードか。

日山：そのとおり。

飯島：入力データ作成にあたり、ミスの防止はどのように行なっているのか。

日山：この計算コードではファイルの命名を変えるだけで実行している、燃料交換データは別のコードで編集している。

日山：ユーザの要望としては、

- ・ 計算時間短縮化のため並列処理の利用
- ・ 次の世代の詳細モデルコードの開発

があつた。

池田：具体的な必要性に基づく要望か。

日山：設計コードの開発は停滞ぎみであり、新しいものを取り入れ、より精度を高めたいということ。

6. 電力中央研究所のコードシステム（松村リーダ）

小林：ORIGEN のコマンドの意味は

松村：たとえば、IRP というコマンドは燃焼計算をするというコマンド、その後のデータは期間、比出力、燃焼前の組成番号、燃焼後の組成番号等を与えている。

7. 安全解析所における炉心解析コード（中島委員）

小林：2 群の不連続因子を使用している。

中島：途中、7 群定数を作成しているが、不連続因子は 2 群のみ容易している。

中川：安全解析所では炉心計算は常に実施しているのか。

中島：原則、事故解析が担当、炉心計算は指示によりその都度行なう。

小林：Ringhals1 号炉安定性解析結果の図の意味は。

中島：各点は測定点、そのうち黒中実の記号は解析値を伴うもの、次図は計算値と測定値の関係を示す。

8. 東芝のFBR 炉物理コード（山岡委員）

小林：SRAC は使用できるのか。

山岡：FBR の計算と SRAC とで扱うエネルギー範囲が違い、また FBR 設計用コードと断面積の形式に違いがあり、使用していない。

9. 三菱のコードシステム（池田幹事）

飯島：3 次元拡散コードは。

池田：HEXZ、TRISTANである。

10. リーダからのコードシステムの提案（松村リーダ）

小林：宣言文はメインプログラムに書く必要はない。

松村：アローケーションはメインプログラムで用意する必要がある。

小林：既存のプログラムをドライバーとして複数利用した場合、メインで宣言すると変数名が重複することがある。

横山：メインプログラムとサブプログラムとで変数名を変えることを許容すれば解決する。

入出力だけ宣言文で記載し、その他はサブプログラムに用意する。

小林：そのような方法の場合、モジュール化できず、巨大化する。

松村：提案したものはFORTRANの世界ですべてやろうとしている。スクリプト、MS-DOS上の操作は行なわない。

小林：ERANOSの方式とは違う。

松村：メモリーはDynamic Allocationで削減できる。

横山：ERANOSではアーカイブと呼ばれる機能を使用し、出力を制御している。

松村：FORTRANにもクラスの概念はある。

横山：提案されたコマンドを使った制御はERANOSに近い。計算を実行するコマンド、メモリを削除するコマンド、結果を出力するコマンド等に分けるという考え方によく似ている。小林：欧州ではシステム化が進んでいる。日本でのシステム化の取り組みは少ない。具体例を考えてもらったら良い。また新しい言語の利用も望ましい。既存のコードが現状のまま動くことが大事。

飯島：SRACはそのようなことを志向したものである。

小林：SRACには3次元Snコードがない。SRACをコールし、少数群断面積を作成し、インターフェイスコードをコールし、最後に3次元Snコードをコールして計算できるのがよい。同じように労力が少なくて他のコードも計算できるようにする。

松村：ANISNに燃焼ルーチンを追加するだけでも大変である。

小林：確かに汎用的なインターフェイスコードを作成するのは大変である。

飯島：熱炉用コードと高速炉用コードを結合しようとしたら、上方散乱の問題があり、大変であった。おそらくインターフェイスコードも大変である。

小林：過去のコードはそのまま使ったほうがよい。

松村：CITATIONの機能を他のコードシステムに持ち込もうとしたら、大変苦労し、結果全く新しく作り直した。

中川：データの形式、コールの仕方、ストリーム、メインを決めるのがまず大事。ポイントとしてデータ、ソルバー、典型的な出力を用意できるポストプロセッサーを準備し、ユーザが選択できるようにする。たとえば核データ、セル計算、拡散計算、摂動計算のひな型を作つておく。

小林：大型計算から EWS ではプログラムの書き換えが必要であった。FORTRAN90 でシエル相当のものが書けるなら簡単で良い。

中川：SRAC は PDS ベースで構築されていた。松村リーダの提案は FORTRAN 内で作成する方式。今後どのようなものが最適となるかはわからない。

小林：シェルスクリプト的に実行できるのがよい。

日山：プロセスとプロセスとの間に準備が必要なのは面倒である。

中川：形状、寸法、材質等の 1 次情報から原子数密度が自動的に計算され実行できるのが良い。

9. 次回まで打合事項

- ・池田幹事から提案のあった内容についてリーダ、幹事で打合せ案を作成する。
- ・案を委員に E-メール等で諮った上、各分担について次回までに作業を行なう。
- ・できれば春の学会で参集し、協議する。

10. 次回

7月 19 日（予定）

以上

付録3－3：第3回会合議事録

平成14年7月23日
作成 安藤（東芝）

共用炉物理コードシステムの構築 WP 平成14年度第1回会合議事録詳細版（案）

時期： 平成14年7月19日（金） 13時30分から17時30分まで
 場所： 原子燃料工業（株）本社 7F会議室
 出席者： 中川（炉物理研究委員会委員長、原電）、松村（リーダー、電中研）、小林（幹事、元京都大）、飯島（幹事、原研）、池田（幹事、三菱重工）、別所（幹事、NUPEC）、横山（サイクル機構）、中島（NUPEC）、内藤（NAIS）、石井（日立）、日山（原燃工）、杉（原研）、安藤（東芝：山岡代理）（敬称省略）

議題

- (1) 前回議事録確認
- (2) コードシステム概念の取りまとめ方針
- (3) 共用炉物理プログラム基本システムの提案
- (4) コードシステムの今後の開発方針
- (5) その他

配布資料

- WP02-1-1 平成13年度第2回共用炉物理コードの構築 WP 議事録（案）
 WP02-1-2 第72回炉物理研究委員会議事録（案）
 WP02-1-2-1 第72回炉物理研究委員会「共用炉物理コードシステムの構築（CCS）WP」
 平成13年度活動報告
 WP02-1-3 炉物理研究委員会における質疑メモ
 WP02-1-4 「共用炉物理コードシステムの作業について（改訂1）」
 WP02-1-5 松村主査提案の共用炉物理コードシステムの開発手順（案）に対する意見
 （改訂1）
 WP02-1-6 日立のBWR核特性解析コードと共用コードの要件
 WP02-1-7 第3回共用炉物理コードの構築 WP 資料
 WP02-1-8 共用炉物理プログラム基本システムの提案
 WP02-1-9 共用炉物理コードシステムの概念・開発手順（案）
 WP02-1-10 「共用炉物理コードシステム」へのご提案
 WP02-1-11 平成10年度計算機科学技術活用型特定研究開発推進事業（短期集中型）
 研究開発修了報告書（研究開発課題：地球惑星流体现象を念頭に置いた
 多次元数値データの構造化）
 WP02-1-12 VSOPコードの概要
 WP02-1-13 共用炉物理コードシステムの構築 WP の最終案のイメージ

<議事内容>

- (1) 前回議事録確認

飯島幹事より、WP02-1-1に基づき、前回議事録が報告された。

(2) コードシステム概念の取りまとめ方針

- ・松村リーダーより、WP02-1-2 (WP02-1-2-1) および WP02-1-3 に基づき、第 72 回炉物理研究委員会で共用炉物理コードシステムの構築 WP の平成 13 年度活動内容を報告したことが紹介された。これに対して、委員から検討項目に対する具体的な作業の進捗状況等の質疑応答とともに公募研究への応募等も視野に入れた今後の方向性への提言が求められるとのコメントがあったことが報告された。

JNC でも基本コードを作成しようという動きがある。モジュラーに対する意識に差があるようなので、JNC と本 WP メンバーで意識合わせが必要と感じているとの話があった。

これに対して、横山委員より JNC では、Small Talk を使ったオブジェクト指向のプログラム開発を基本方針としているとの話があった。

(3) 共用炉物理プログラム基本システムの提案

- ・池田幹事より、WP02-1-4 に基づき、共用炉物理コードシステムについて報告があった。コードシステム概念には、モジュール型、統合型、インターフェイス型の 3 つがあり、モジュール型の例として MOSRA、EURANOS、統合型の例として SRAC、VSOP、インターフェイス型の例として JOINT が紹介され、既存のコードを生かした共用炉物コードシステムについて提案があり、最小限の労力で既存コードを生かすには、インターフェイス型が適しているとの私案が示された。

これに対して、小林幹事より、FORTRAN90 を使ったモジュール型が良いとの提案があった。

- ・石井委員より、WP02-1-6 基づき、日立の BWR 核特性解析コードおよび共用コードの条件が示された。

日立では、燃料集合体コードとして HINES、VMONT、炉心シミュレータとして PANACEA、COSNEX、熱特性解析コードとして TRACG、STAND が使用されている。このうち、VMONT、COSNEX、STAND は日立の自社開発、他は GE からの導入コードである。

また、共用コードに必要な条件として、以下の 5 つが示された。

- ① 炉型に依存しない汎用性・高精度
- ② 燃料、炉心の高度化（形状・炉物理現象の複雑化）に対応できる近似の少ない手法
- ③ 十分な検証、高権威性
- ④ 高速
- ⑤ 高機能性、高い使い勝手
- ⑥ 高保守性

このうち、特に③が重要との意見が示された。

- ・横山委員より、WP02-1-7 基づき、第 3 回共用炉物理コードの構築 WP 資料の説明があった。第 2 回会合資料 (WP01-2-3, WP01-2-4) では、Python、Ruby 等のオブジェクト指向スクリプト言語を利用することを提案したが、これらのスクリプト言語は解析コードシステムへの適用実績が少ないとこれまでに作成された解析コードは、Fortran で書かれているものがほとんどであることから、Fortran により基本構築するのが現実的との提案がなされた。

これに対して、松村リーダーより両案の長所・短所の整理が必要とのコメントがあり、以下

の質疑があった。

小林幹事：大容量が必要な場合は、ハードディスクは利用可能か？

横山委員：ファイルと TYPE を対応させて、ファイルを介してやり取りが可能である。

飯島委員：RAM とデータをロック化するかどうかは、状況次第決めれば、融通性ができるのではないか

松村リーダー：松村の先の提案は簡易言語を必ずしも作ることを意図していた訳ではない。

- ・小林幹事より、WP02-1-8 に基づき、共用炉物理プログラム基本システムの提案があった。基本システムは、Fortran などの高級言語で書かれる必要があり、機種依存性のあるアセンブラー言語などの低級言語の使用は避ける。これを基本ルールとして、プログラム基本システム構築の原則、プログラムの基本システムが示された。小林提案は、Fortran90 を使用することを基本としており、周辺コードを開発して新しいことができるようになったら、コードシステムに登録するのが良いとの提案がなされた。示された実行例では、プログラム A とプログラム B を組み合わせてシステムを構築する際、各プログラムに同じサブルーチン名、コモン文があつても良いとの事である。

(4) コードシステムの今後の開発方針

- ・松村リーダーより、WP02-1-9 に基づき、共用炉物理コードシステムの概念・開発手順（案）が示された。

公開可能なコードの整理と共用コードシステムへの要望が、原研開発コードと JNC 開発コードをベースにまとめられた。

何が必要・問題か

- ・軽水炉、FBR の中間的なスペクトル炉心に対して、精度良い実効断面積が計算できない。
- ・FBR での 2 重非均質効果（非均質集合体など）の評価では、2 次元格子計算が必要
- ・解析コードの数が多く、各計算コードに入出力形式が不統一
- ・多様な集合体平均定数と不連続因子を計算できる集合体解析コードが必要（CASMO - 4 では不連続因子は 2 群まで）
- ・より高次の計算理論に基づいた近似の少ない、大容量・高速処理の機能を有する計算コード
- ・小型炉・高転換炉・ガス炉等に適用でき、汎用で精度が高く、ユーザーフレンドリーなコード
また、共用炉物理コードシステムの概念（案）、共用炉物理コードシステムの開発手順（案）が示された。

- ・池田幹事より、WP02-1-5 基づき、共用炉物理コードシステムの開発手順に対する意見が上記の松村案ベースに示された。

共用炉物理コードシステムについては、タスクフォースがなく、委員の意見の交換の場にあつているので、共用炉物理コードのコアに絞って目標を設定するのが良いとの意見が示された。
また、共用炉物理コードシステムの開発手順について私案が示された。

- ・松村リーダーより、WP02-1-10 に基づき、「共用炉物理コードシステム」へのご提案について説明があった。本提案は、日本総研の佐々木氏による提案で、汎用スクリプト言語を「ストリームプログラム」のベースとすること、データ交換のための共通データフォーマットの定義として XML を採用することから成っている。

なお、佐々木氏は、SE で MVP コードの作成者でもある。

しかしながら、本 WP の全体意見となっている Fortran をプログラム言語とすることに合致しない点があるので、今後検討が必要。

- ・横山委員より、WP02-1-11 に基づき、汎用スクリプト言語の利用に関連して、平成 10 年度計算機科学技術活用型特定研究開発推進事業（短期集中型）として実施された「地球惑星流体现象を念頭に置いた多次元数値データの構造化」について紹介があった。
- ・WP02-1-12 「VSOP コードの概要」（池田幹事）は、時間の関係で説明が省略した。

以上の議論の後、各委員から以下の意見が出された。

- ・池田幹事：ユーザーインターフェイスをどう作るか。基本コード（SRAC、SLAROM）をうまく利用することとユーザーインターフェイスコードの開発の具体化が必要。
- ・内藤委員：何が重要かはっきりさせることが重要。
- ・中川委員長：炉型によらずに 1 元管理できるシステムが重要。MVP の開発は、このようなシステムを目指したことを見出発点であった。共用コードに求められているものは、Deterministic にこれらが含まれることであろう。Up - Scattering の扱いは、熱炉コード特有の課題であり、この点で熱炉コードは、高速炉コードを包含できる。
- ・松村リーダーおよび別所幹事より、本 WP のタスクフォースを具体化するために、平成 15 年度の公募提案を行いたいとの意見が提案された。これは、炉物理委員会におけるコメントでもあり、来年 3 月までに提案書（案）を作成することになった。
- ・松村リーダーより、WP02-1-13 に基づき、共用炉物理コードシステムの構築 WP の最終案のイメージが示され、以下の担当で執筆することになった。
第 1 章（緒言）、第 2 章（活動報告）：松村リーダー
第 3 章（共用炉物理コードシステムの概念・方向性）
 - ・基本（ストリーム）プログラムの構造案：小林幹事
 - ・インターフェースのルール化：池田幹事
 - ・開発対象とすべきソルバー：松村リーダー（一例）
(開発対象とすべきソルバーについては、複数有っても良いため各委員から任意に提案を受け付けることになった)
 - ・先端計算機技術の活用：横山委員
- 第 4 章（今後の活動課題）、第 5 章（結言）：松村リーダー

これらは、次回の WP までに CCS メーリングリストでお互いに原案を送付する事になった。また、共用炉物理コードシステム開発に関する具体的な提案（波及効果を考慮した将来性のある提案）についても次回までの宿題（事前にメールで交換）となつた。

（5）その他

次回の WP は、12 月 6 日（金）に東京地区で開催する。

—以上—

付録3-4：第4回会合議事録

2002.12.9

(電中研 松村)

共用炉物理コードシステムの構築WP 平成14年度第二回会合議事録（案）

日時：平成14年12月6日（金） 13:00～17:30

場所：(財)電力中央研究所 大手町本部 第3会議室

出席者：中川正幸（炉物理研究委員会委員長、原電）、松村哲夫（リーダー、電中研）、小林啓祐（幹事、京都大学名誉教授）、森貴正（幹事、原研）、池田一三（幹事、三菱重工）、別所泰典（幹事、NUPEC）、飯島進（原研）、山岡光明（東芝）、横山賢治（サイクル機構）、内藤淑孝（NAIS）、日山伸行（原燃工）、杉暉夫（原研）（敬称省略）

議題：

- (1) 前回議事録確認
- (2) 委員会体制
- (3) 成果報告書の取りまとめ
- (4) 今後の進め方

配布資料：

WP02-2-1 共用炉物理コードシステムの構築WP 平成14年度第1回会合議事録詳細版
(案)

WP02-2-2 炉物理研究委員会原子炉システム専門部会活動について

WP02-2-3 共用炉物理コードシステムの構築WPの活動報告（叩き台）

WP02-2-3' 同上、資料

WP02-2-4 共用炉物理コードシステムの構築WPの活動報告OHP (案)

WP02-2-5 3.4 インターフェースのルール化について

議事内容：

- (1) 前回議事録確認

森幹事より、WP02-2-1に基づき、前回議事録詳細版が報告され、一部字句の修正を行う事で承認された。尚、議事録の簡易版は森幹事が担当することとなった。

- (2) 委員会体制

森幹事より、WP02-2-2に基づき、専門部会名簿の紹介があり、委員とオブザーバーの考え方方が代わったとの補足説明があった。

- (3) 成果報告書の取りまとめ

松村リーダーより、WP02-2-3 および WP02-2-3' に基づき、W P の成果報告書の叩き台が紹介され、以下の目次構成とすることが概ね了解された。

1. 緒言
2. 炉物理コードシステムの調査
 2. 1 諸機関の炉物理コードシステムの調査
 2. 2 先端計算機技術の活用
 2. 3 統合型物理コードの調査
 2. 4 共用炉物理コードシステム概念の検討
3. 共用炉物理コードシステムの基本概念
 3. 1 ストリームプログラムの構造
 3. 1. 1 スクリプトプログラムの利用
 3. 1. 2 F O R T R A N 9 0 の利用
 3. 1. 2 既存コードの連携
 3. 2 インターフェースのルール化
 3. 2. 1 C C C C のインターフェースのルール化
 3. 2. 2 热炉と高速炉に共用のインターフェース
 3. 2. 3 X L M を適用したインターフェース管理
 3. 3 開発すべきソルバー
4. 結言（今後の活動方針の提言）

資料

1. サイクル機構の高速炉炉心解析コードシステムの現状
2. 原研の計算コードシステム
3. 原燃工の炉物理コードの現状について
4. 電中研の軽水炉炉心解析コードシステム
5. 原子力安全解析所における炉心解析コードについて
6. 東芝のF B R 炉物理コードの現状
7. 三菱の核設計コードの現状
8. 日立のB W R 核特性解析コード
9. コマンド型の統合コード案
10. F O T R A N 9 0 のモジュール機能の利用案

成果報告書の各部分について以下の様な議論があった。

1) 報告書の発刊、タイトル

従来のWPの成果と同様に、原研の報告書して発刊するのが良い。この場合、報告書のタイトルはWPの活動報告書よりは、「共用炉物理コードシステムの基本概念の構築」などが良い。

2) 先端計算機技術の活用

横山委員より、WP02-2-3 の 2.2 節の資料をもとに紹介があり、以下の質問があった。

- E R A N O S の言語 (E S O P E) は公開されているのか。⇒公開されていない。
- 構造体とは何か。⇒データの固まりを記述できるようになっている。

また、横山委員より、フランス側から J N C 側に共通プラットフォームの開発について共同開発の打診があるとの補足説明があった。

3) 共用炉物理コードシステム概念の検討

計算プラットフォームで大型計算機を対象とするのは時代遅れではないか。既に U N I X が標準となっている。

4) 既存コードの連携

- 例 1 の高速炉計算と S R A C などの熱炉解析をエネルギー的に結合することは行わないのではないか。
- S R A C には 3 次元の拡散計算機能は統合されているが、炉心の輸送計算機能は無いため、この様な連携は有る。また、六角メッシュや三角メッシュの炉心計算との連携も必要。
- 廃止措置などで、炉心計算と遮蔽体の遮蔽計算を繋いだ計算を行っている。この様な統合的な使い方が考えられる。共用炉物理コードシステムでは臨界安全や遮蔽計算も統合して欲しい。
- 炉心計算と遮蔽計算の接合ではソースの与え方が難しい。
- 定常の炉心計算と動特性解析との結合も必要、

5) 開発すべきソルバー

- S R A C などの衝突確率法は小さな体系では有効であるが、複数の集合体や炉心計算への適用は困難。キャラクタリスト法の開発が必要ではないか。
- $r - \theta - z$ 体系のノード法のコードが無い。
- J E N D L 3. 3などをベースにした標準的な多群計算用のライブラリーのニーズもある。
- 不連続因子を計算出来る計算コードが少ない。

- ・ 三角メッシュ、六角メッシュの高速な計算コードが無い。

5) インターフェース

池田幹事より、WP02-2-2 に基づき、インターフェースのルール化の目次案について説明があり、以下の様な議論があった。

- ・ CCCCのインターフェースはかなり古いのではないか。フォーマットが複雑で利用が難しい。
- ・ CCCCをインターフェースにすると、共用炉物理コードシステムのサブコード全てにCCCC入出力のためのインターフェースを導入する必要があるのではないか。一方、必ずしも全てのコードに導入しなくも統合化は可能との意見があった。
- ・ XMLは柔軟性があるがテキスト形式でファイル容量が膨大になる可能性がある。
- ・ XMLの読み書きのためのDOM、SAXなどのインターフェース・アプリケーションが開発されている。DOMはメモリーに一度読みこむのでデータ処理の柔軟性が高い、SAXは部分的に読みこんでいくので大きなファイルの取り扱いに優れているが、柔軟性が少ない特徴がある。但し、FORTRANから直接DOM、SAXなどが呼び出されるかは不明である。
- ・ MVPの開発の際に、XMLそのものではないが、データをメモリーに読み込み、後で構造を定義するXMLの考え方を利用した。
- ・ 統合するサブ・システムが決まって居ない状況では、CCCCのようにインターフェースのフォーマットを予めておく事が出来ないのでないか。
- ・ 輸送計算でのP1の断面積もANISN系ライブラリーとそれ以外で(21+1)倍違うことなども考慮して置く必要がある。

6) 全般的意見

山岡委員より、WP02-2-3 の3.4.3節の資料を基に、以下の様な提案があった。

- ・ 決定論手法には何らかの近似が入るのが避けられないため、現状での決定論コードを作成しても陳腐化するのではないか。しかし、決定論コードのニーズはあり、熱炉、高速炉のコードシステムの融合化、熱炉、高速炉、遮蔽などのライブラリーの簡単な利用、最新手法の導入が容易、開発労力資金の最小化を目標としてはどうか。
- ・ インターフェースも全てのサブ・システムで統合化するのではなく、モジュール内で統合し、更に、モジュール間で変換コード・様式を定めてはどうか。

内藤委員より、WP02-2-3 の4章の資料を基に、臨界安全を目的としたJACSシステム

の紹介があり、臨界解析、遮蔽計算も共用炉物理コードシステムに統合は容易であるとの説明があった。また、杉委員の四因子計算などにも配慮して欲しいとの要望に答えて JACS システムの SIMCRI で四因子計算を行っているとの回答があった。

(4) 今後の進め方

議論の結果、以下の分担が決められた。

- ・ 報告書の先般は松村リーダーが取りまとめる。
- ・ 資料 1 0. F O T R A N 9 0 のモジュール機能の利用案は小林幹事がリバイスする。また、この概要を 3. 1. 2 F O T R A N 9 0 の利用として作成する。
- ・ 3. 2 インターフェースのルール化は W P での意見も参考にして、構成も含めて池田幹事が担当する。
- ・ 資料 1 ~ 8 の各機関の炉物理コードについては各機関で内容を確認・見直しを行う。
- ・ 資料 9 のコマンド型の統合コード案は松村リーダーがまとめ方も含めて見直す。
- ・ 春の原子力学会での炉物理部会セッションでは W P の報告を行わない予定（森幹事が確認する）であるため、松村リーダーが発表原稿を作成し、 W P メンバーの了解を得て予稿を提出する。

報告書に記載されるこれらの情報は修正がない場合でも、電子ファイル (Word, Power Point、これが不可能な場合は pdf も可) で 12 月末を目途に提出することとなった。

また、共用炉物理コードシステムの構築 W P は平成 14 年度で終了として、平成 15 年度は公募に応募して活動を継続することとする事が合意された。また、公募について以下の様な意見があった。

- ・ 経済産業省（エネ総研）と文部科学省の公募があるが、経済産業省の公募の方が実用化を標榜しているため近いのではないか。
- ・ 公募のメンバーとしては本 W P の委員（岩崎委員を含む）と大学の先生として阪大の竹田先生、京大の三澤先生、北大の島津先生などが候補である。特に、竹田先生については、本 W P の報告書(案)が纏まった段階で打診することになった。

最後に、小林幹事より、共用炉物理コードシステムについて 3 年間検討を行ったわけであり、是非コードシステムを実現して欲しいとの要望があった。

以上

付録4-1:原子力学会 2003年春の年会予稿

共用炉物理コードシステムの基本概念の構築

-共用炉物理コードシステムの構築ワーキングパーティー活動報告-

Development of basic concept of a common reactor physics code system

-Activity of common reactor physics code system working party-

○電中研 松村哲夫、京大 小林啓祐、原電 中川正幸、原研 森貴正 飯島進 杉暉夫、

三菱重工 池田一三、NUPEC 別所泰典 山本徹 中島鐵雄、サイクル機構 横山賢治、

東芝 山岡光明、日立 石井一弥、原燃工 日山信行、東北大 岩崎智彦、NAIS 内藤淑孝

MATSUMURA Tetsuo, KOBAYASHI Keisuke, NAKAGWA Masayuki, MORI Takamasa, IJIMA Susumu, SUGI Teruo,

IKEDA Kazumi, BESSHOU Hironori, YAMAMOTO Tooru, NAKAJIMA Tetsuo, YOKOYAMA Kenji,

YAMAOKA Mitsuaki, ISHII Kazuya, HIYAMA Nobuyuki, IWASAKI Tomohiko, NAITO Yasutaka

炉物理研究委員会の共用炉物理コードシステムの構築ワーキングパーティーの平成13, 14年度の活動として共用炉物理コードシステムの基本概念を構築した。システムはストリームプログラム、既存プログラムのサブシステム化、ソルバー、インターフェースからなり、望ましい開発言語などを集約した。

キーワード 炉物理コードシステム、基本概念、スクリプト言語、モジュール機能

1. 【緒言】 炉物理部会会員がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築することを目的に、16名のメンバーで平成13, 14年度の2カ年の期間で4回の会合を持ち望ましい共用炉物理コードシステムについて検討を実施した。

2. 【調査・課題抽出】 参加機関の炉物理コード、先端計算機技術、統合型炉物理コードなどの調査を行うとともに、参加者から、既存コード・システムの問題点、希望などを調査し、①プラットフォーム依存性の無いストリームプログラム、②既存コード・データの最少の労力での利用（サブシステム化）、③既存コードでカバー出来ない計算機能を補う新規のソルバー、④コード相互、また外部コード・データとも連携が容易なインターフェース、⑤検証、ベンチマーク問題の抽出を含め、十分な検証、⑥解法・機能を記述した十分な資料の作成や対外活動、⑦維持・発展させる仕組み、の必要性が抽出された。

3. 【コードシステムの基本概念】 コードシステムはストリームプログラム、既存プログラムのサブシステム化、必要な新規ソルバー、インターフェースからなる（図1）。ストリームプログラムの開発言語としては、先端計算機技術の活用の調査結果も踏まえ、Perl、Ruby、Pythonなどのスクリプト言語とモジュール機能を持つFORTRAN90を抽出し、得失を比較した。インターフェースとして拡張性の高いXML(extensible Markup Language)の利用やその考え方を利用した新規のインターフェース言語の設計が考えられる。また、必要なソルバーとして、標準多群計算用ライブラリー、キャラクタリスト法による輸送計算コードなどの開発課題が抽出された。

4. 【今後の課題】 開発のためのより具体的な手順・体制とともに、開発したコードシステムをその後、維持・発展させる仕組みの構築が必要である。

【文献】共用炉物理コードシステム特別専門委員会報告書、JNC TJ9420 2001-007

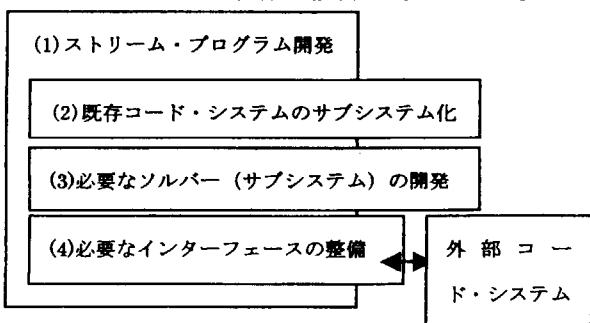


図1 共用炉物理コードシステムの開発手順案

付録4-2 原子力学会2003年春の年会発表

共用炉物理コードシステムの構築WP活動報告(案)

原子力学会2003年春の年会
(H20, 2003.3.28)
アルカスSASEBO

1

共用炉物理コードシステムの構築WPのメンバー構成

- 参加者数: 15名
- リーダー: 松村哲夫(電中研)
- 幹事: 小林 啓祐(京都大学名誉教授)
別所 泰典(NUPEC)
池田一三(三菱重工)
飯島 進(原研)*

*途中、森貴正氏に交代

2

活動概要

- ・ 第1回会合：平成13年11月26日（火）
リーダー・幹事の選出、活動計画
- ・ 第2回会合：平成14年 2月22日（金）
各機関のコードシステムの紹介
- ・ 第3回会合：平成14年 7月19日（金）
コードシステムの議論
- ・ 第4回会合：平成14年12月 6日（金）
取りまとめの議論

3

WP活動目的

- ・ 炉物理部会会員がその研究・開発活動を効率的に行えるように、幅広い研究分野で便利に使える炉物理コードシステムを構築する。

4

WP活動内容

- ・各機関各炉型毎のコードの現状・課題の整理
- ・先端計算機技術、統合型炉物理コードの調査
- ・共用炉物理コードシステムの全体構想検討
- ・ストリームプログラム、ソルバー、インターフェースなどの概念検討
- ・開発・整備の体制の検討

5

各機関各炉型毎のコードの 現状・課題の整理

- ・燃料棒、燃料集合体・要素、炉心、燃焼計算の他に、感度解析、炉定数調整、核設計精度評価、共分散処理、動特性、過渡解析、ガンマ線計算、装荷パターン作成、炉心安定性解析、遮蔽、臨界安全など多種の炉物理コード、コードシステムが利用されている。
- ・炉型を跨って利用されているコードは少ない
- ・各機関の役割に応じて、保有・開発しているコードに特徴がある。

6

統合型炉物理コード・ 先端計算機技術の調査

- 欧州炉物理解析システムERANOS
 - 独自のプログラミング言語の使用
 - オブジェクト指向的なデータ構造
- ドイツ・ユーリッヒ研究所のVSOP
- JOINT
- ORIGEN2のコマンド機能
- 計算機技術:スクリプト言語、
データ記述言語XML, FORTRAN90/95

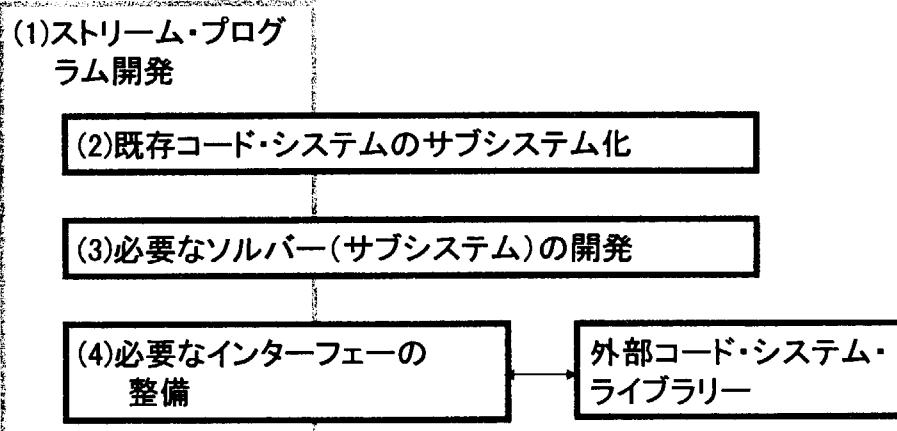
7

何が必要・何が問題

- 既存コード・データを結合して利用できるプラットフォーム依存性の無いストリームプログラム
- 既存コード・データは最小の労力でサブシステム化出来ること
- 既存コードでカバー出来ない計算機能を補う新規のソルバー
- コード相互、外部コード・データとも連携が容易なインターフェース
- 実用的で十分に検証されている必要
- 十分な資料の作成や対外活動も重要
- コードシステムを維持・発展させる仕組みが必要

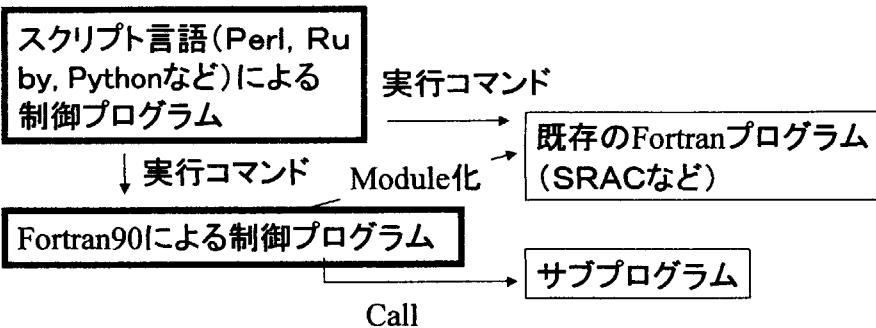
8

コードシステムの全体概念



9

ストリームプログラムの構造案



スクリプト言語やFortran90の利用により既存コードの容易な利用が可能である。

10

ストリームプログラムの構造案 —スクリプト言語の利用—

制御のイメージ

```
>                                     スクリプト言語のプロンプト  
>system("SRAC")      SRACの起動  
>open(in,"SRAC.out")  SRACの出力ファイルの操作  
>.....
```

- ・殆どのUNIX機種、Windowで稼動するフリーソフト
- ・既存のプログラムをそのまま利用
- ・他の計算機のプログラムも制御可能
- ・テキストファイルの処理が可能
- ・Webページ上でも稼動可能

11

ストリームプログラムの構造案 —FORTRAN90言語の利用—

制御プログラム
USE SRAC

既存プログラムのMODULE化

MODULE SRAC

A=B+C

- ・FORTRAN90は、FORTRAN77と比較して、MODULE化機能、メモリー領域の動的確保、マトリクス演算機能などが追加された。
- ・既存プログラムでMODULE宣言文を追加すれば殆ど修正不要
- ・FORTRANだけで制御可能

12

既存コードの連携のイメージ(1)

(例)

六角形状燃料要素を持つ革新炉での炉心解析

(問題点)

3次元六角形状解析は一般的でない

ストリーム・プログラム

SRACなどの燃料要素解析コード

高速炉の3次元六角形状輸送コード(NSHEX)など

13

既存コードの連携のイメージ(2)

(例)

熱炉での断面積等の感度解析

(問題点)

熱炉での断面積等の感度解析コードの整備が不充分

ストリーム・プログラム

SRACなどの燃料要素解析コード

高速炉用感度解析コード(SAGEP)

14

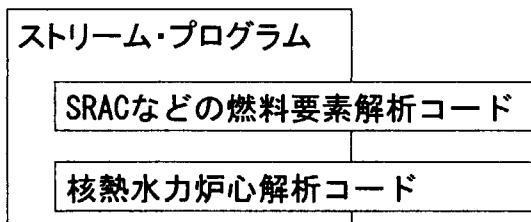
既存コードの連携のイメージ(3)

(例)

革新炉の核热水力炉心解析

(問題点)

- ・既存の核热水力解析コードは炉型依存性が強い
- ・MVPなどのモンテカルロ法で核热水力計算は困難



15

既存コードの連携方法

- ・標準ファイル形式の設定
XMLの様な汎用的な形式
(既存のPDSファイルなども利用)
- ・インターフェース・コードシステムの構築
既存プログラムの修正を最少化
XML(eXtensible Markup Language)の紹介
- ・自分自身のデータの構造を記述するDTD部 と
それに従って記述されたデータ部分からなるテ
キストデータからなる言語
- ・様々な形式のデータの保存および活用の方法と
して注目を集めている

16

その他の推進方法

新規に開発すべきソルバー案

- ・標準多群計算用ライブラリー
- ・キャラクタリスク法による輸送計算コード
- ・その他

コードシステムの検証・利用方法

- ・解法の公開性、十分な検証の必要性
- ・多くのユーザーが利用出来る体制作り
- ・計算結果(検証結果)のデータベース化
- ・海外との連携

17

共用炉物理コードシステムの 利用のイメージ

①自分のPC(LINUX), EWSなどで 利用環境

②WWWやTelenetを用いてネットワークで

①コマンド型で 入力方法

- Use JFS3-3
- Cal Pin-cell
- Save flux ...

②メニュー方式で

- Library
- JFS
- SRAC lib
-

出力方法

①デジタルデータ出力で

②グラフ、CGI出力で

18

まとめ

- ・各機関の炉物理コード、先端計算機技術、統合型炉物理コードなどを調査し、「炉型を跨って利用されているコードは少ない」などの問題点が抽出された
- ・「既存コード・データを結合して利用できるプラットフォーム依存性の無いストリームプログラム」など共用炉物理コードシステムに求められる用件、全体概念を明らかにした
- ・スクリプト言語やFortran90の利用によるストリームプログラムの構造を示した
- ・XMLの様な汎用的な形式の標準ファイル形式などインターフェースの構造を議論した
- ・解法の公開性、十分な検証の必要性、多くのユーザーが利用出来る体制作りなどの推進方法を提案した

19

提言

今後の発展のため、
「共用炉物理コードシステムの構築」
に多くの参加者を希望します



ご連絡をお願いします

20

国際単位系(SI)と換算表

表1 SI基本単位および補助単位

量	名称	記号
長さ	メートル	m
質量	キログラム	kg
時間	秒	s
電流	アンペア	A
熱力学温度	ケルビン	K
物質量	モル	mol
光度	カンデラ	cd
平面角	ラジアン	rad
立体角	ステラジアン	sr

表3 固有の名称をもつSI組立単位

量	名称	記号	他のSI単位による表現
周波数	ヘルツ	Hz	s^{-1}
力	ニュートン	N	$m \cdot kg/s^2$
圧力、応力	パスカル	Pa	N/m^2
エネルギー、仕事、熱量	ジュール	J	N·m
功率、放射束	ワット	W	J/s
電気量、電荷	クーロン	C	A·s
電位、電圧、起電力	ボルト	V	W/A
静電容量	ファラード	F	C/V
電気抵抗	オーム	Ω	V/A
コンダクタンス	ジーメンス	S	A/V
磁束	ウェーバ	Wb	V·s
磁束密度	テスラ	T	Wb/m ²
インダクタンス	ヘンリー	H	Wb/A
セルシウス温度	セルシウス度	°C	
光束	ルーメン	lm	$cd \cdot sr$
照度	ルクス	lx	lm/m^2
放射能	ベクレル	Bq	s^{-1}
吸収線量	グレイ	Gy	J/kg
線量当量	シーベルト	Sv	J/kg

表2 SIと併用される単位

名称	記号
分、時、日	min, h, d
度、分、秒	°, ', "
リットル	L, L
トン	t
電子ボルト	eV
原子質量単位	u

$$1 \text{ eV} = 1.60218 \times 10^{-19} \text{ J}$$

$$1 \text{ u} = 1.66054 \times 10^{-27} \text{ kg}$$

表5 SI接頭語

倍数	接頭語	記号
10^{18}	エクサ	E
10^{15}	ペタ	P
10^{12}	テラ	T
10^9	ギガ	G
10^6	メガ	M
10^3	キロ	k
10^2	ヘクト	h
10^1	デカ	da
10^{-1}	デシ	d
10^{-2}	センチ	c
10^{-3}	ミリ	m
10^{-6}	マイクロ	μ
10^{-9}	ナノ	n
10^{-12}	ピコ	p
10^{-15}	フェムト	f
10^{-18}	アト	a

表4 SIと共に暫定的に維持される単位

名称	記号
オングストローム	Å
バーン	b
バル	bar
ガル	Gal
キュリ	Ci
レントゲン	R
ラド	rad
レム	rem

$$1 \text{ Å} = 0.1 \text{ nm} = 10^{-10} \text{ m}$$

$$1 \text{ b} = 100 \text{ fm}^2 = 10^{-28} \text{ m}^2$$

$$1 \text{ bar} = 0.1 \text{ MPa} = 10^5 \text{ Pa}$$

$$1 \text{ Gal} = 1 \text{ cm/s}^2 = 10^{-2} \text{ m/s}^2$$

$$1 \text{ Ci} = 3.7 \times 10^{10} \text{ Bq}$$

$$1 \text{ R} = 2.58 \times 10^{-4} \text{ C/kg}$$

$$1 \text{ rad} = 1 \text{ cGy} = 10^{-2} \text{ Gy}$$

$$1 \text{ rem} = 1 \text{ cSv} = 10^{-2} \text{ Sv}$$

(注)

- 表1～5は「国際単位系」第5版、国際度量衡局1985年刊行による。ただし、1eVおよび1uの値はCODATAの1986年推奨値によった。
- 表4には海里、ノット、アール、ヘクトアルも含まれているが日常の単位なのでここでは省略した。
- barは、JISでは流体の圧力を表わす場合に限り表2のカテゴリーに分類されている。
- EC関係理事会指令ではbar、barnおよび「血圧の単位」mmHgを表2のカテゴリーに入れている。

換 算 表

力	N(=10 ⁵ dyn)	kgf	lbf
1	0.101972	0.224809	
9.80665	1	2.20462	
4.4822	0.453592	1	

$$\text{粘度 } 1 \text{ Pa} \cdot \text{s} = 10 \text{ P(ボアズ)} (\text{g}/(\text{cm} \cdot \text{s}))$$

$$\text{動粘度 } 1 \text{ m}^2/\text{s} = 10^4 \text{ St(ストークス)} (\text{cm}^2/\text{s})$$

圧力	MPa(=10 bar)	kgf/cm ²	atm	mmHg(Torr)	lbf/in ² (psi)
力	1	10.1972	9.86923	7.50062×10^3	145.038
0.0980665	0.0980665	1	0.967841	735.559	14.2233
0.101325	0.101325	1.03323	1	760	14.6959
	1.33322×10^{-4}	1.35951×10^{-3}	1.31579×10^{-3}	1	1.93368×10^{-2}
	6.89476×10^{-3}	7.03070×10^{-2}	6.80460×10^{-2}	51.7149	1

エネルギー・仕事・熱量	J(=10 ⁷ erg)	kgf·m		kW·h		cal(計量法)	Btu	ft · lbf	eV	1 cal = 4.18605 J(計量法)
		1	0.101972	2.77778×10^{-7}	0.238889					= 4.184 J(熱化学)
9.80665	1	2.72407 × 10 ⁻⁶	2.34270	9.29487×10^{-3}	7.23301	6.12082×10^{19}	= 4.1855 J(15 °C)			
3.6×10^6	3.67098×10^5	1	8.59999×10^5	3412.13	2.65522×10^6	2.24694×10^{25}	= 4.1868 J(国際蒸気表)			
4.18605	0.426858	1.16279×10^{-6}	1	3.96759×10^{-3}	3.08747	2.61272×10^{19}	仕事率 1 PS(仏馬力)			
1055.06	107.586	2.93072×10^{-4}	252.042	1	778.172	6.58515×10^{21}	= 75 kgf·m/s			
1.35582	0.138255	3.76616×10^{-7}	0.323890	1.28506×10^{-3}	1	8.46233×10^{18}	= 735.499 W			
1.60218×10^{-19}	1.63377×10^{-20}	4.45050×10^{-26}	3.82743×10^{-20}	1.51857×10^{-22}	1.18171×10^{-19}	1				

放射能	Bq	Ci	吸收線量	Gy	rad
	1	2.70270×10^{-11}		1	100
	3.7×10^{10}	1		0.01	1

照射線量	C/kg	R
	1	3876
	2.58×10^{-4}	1

線量当量	Sv	rem
	1	100
	0.01	1

共用炉物理コードシステムの基本概念の検討 —共用炉物理コードシステム(CCS)の構築WPの活動報告—

R100
古紙配合率100%
白色度70%再生紙を使用しています