JNC TN9400 2003-021

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(Ⅱ)

― プロトタイプ作成による検討 ―

(研究報告書)

2003年4月

核燃料サイクル開発機構
大洗工学センター

〒319-1184　茨城県那珂郡東海村村松４番地４９
核燃料サイクル開発機構
技術展開部　技術協力課
電話：029-282-1122（代表）
ファックス：029-282-7980
電子メール：jserv@jnc.go.jp

JNC TN9400 2003-021
2003年4月

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（II）

—プロトタイプ作成による検討—

（研究報告書）

横山 賢治[1]、細貝 広視[2]、宇都 成昭[3]、笠原 直人[4]、石川 眞[1]

要 旨

高速炉開発において、解析コードを用いた数値シミュレーションは、理論、実験を補足するための重要な役割を果たしている。研究開発に対するニーズが多様化し、解析対象がより複雑化している現在では、工学的なモデルや解析手法を柔軟に変更したり、新たなモデルや手法を開発して容易に従来のシステムを拡張したりできることが、解析コードに求められる重要な要素となってきている。また、原子力に関連する技術分野は多岐にわたるため、多くの異なる分野の物理データや工学的モデル・手法を、いかにうまく結合して利用できるようにするかという点に大きな課題がある。

本研究では、このような課題を解決できる次世代解析システムの実現を目標とし、3種類の方法、(1)多言語（SoftWIRE.NET、Visual Basic .NET、Fortran）、(2)Fortran90、(3)Python を採用してプロトタイプの作成・検討を実施した。

この結果、Visual Basic .NET を用いることで、Fortran で書かれた既存の解析コードの機能を新しい解析コードシステム上で部品化することや、Fortran90 の新しい機能を用いることにより従来の Fortran77 で書かれた解析コードの保守性や拡張性を向上させること、Python を用いてツールボックス型の解析コードシステムを構築することができることを示した。

---

1 大洗工学センター システム技術開発部　中性子工学グループ

2 常陽産業株式会社

3 大洗工学センター システム技術開発部　炉心・燃料システムグループ

4 大洗工学センター 要素技術開発部 構造信頼性研究グループ

JNC TN9400 2003-021
April, 2003

Development of the Next Generation Code System As an Engineering Modeling Language (II)

— Study with Prototyping —

Kenji YOKOYAMA[1], Hiromi HOSOGAI[2], Nariaki UTO[3],
Naoto KASAHARA[4] and Makoto ISHIKAWA[1]

Abstract

In the fast reactor development, numerical simulation using analytical codes plays an important role for complementing theory and experiment. It is necessary that the engineering models and analysis methods can be flexibly changed, because the phenomina to be investigated become more complicated due to the diversity of the needs for sresearch. And, there are large problems in combining physical propaties and engineering models in many different fields.

Aming to the realization of the next generation code system which can solve those problems, the authors adopted three methods, (1) Multi-language (SoftWIRE.NET, Visual Basic .NET and Fortran) (2) Fortran90 and (3) Python to make a prototype of the next generation code system.

As this result, the followings were comfirmed. (1)It is possible to reuse a function of the existing codes written in Fortran as an object of the next generation code system by using Visual Basic .NET. (2) The maintenanability of the existing code written by Fortran77 can be improved by using the new features of Fortran90. (3) The toolbox-type code system can be built by using Python.

---

1 Reactor Physics Research Gr.、 System Engineering and Technology Div.、 OEC、JNC

2 Joyo Industry Co. Ltd.

3 Core and Fuel System Gr.、 System Engineering and Technology Div.、 OEC、JNC

4 Structure and Material Research Group、 Advanced Technology Division、 OEC、 JNC

目　次

表リスト

図リスト

第1章 緒　　言

第1章　緒言

21 世紀においても、ますますの科学技術の発展が期待されるが、その発展の形態については、時代の流れや多様化するニーズに柔軟かつ適切に対応することがこれまで以上に求められている。このような傾向は、原子力分野においても例外ではなく、その研究開発においては、時代の流れや多様化するニーズに柔軟かつ適切に対応できることが必要となっている。

一方、現在、解析コードを用いた数値シミュレーションは、理論、実験を補足するための、科学技術活動の一環であることが一般に認められるようになっているが、このことは原子力分野においても例外ではない。原子力開発においても、数値シミュレーション技術はますます重要度を増している。

研究開発に対するニーズが多様化し、解析対象がより複雑化している現在では、解析対象に応じて、工学的なモデルや解析手法を柔軟に変更したり、新たなモデルや手法を開発して従来のシステムを拡張したりできることが、解析コードに求められる重要な要素となってきている。従来、原子力関連の解析コードに求められる性能は、計算精度や計算速度が主なものであったが、これらの性能に加えて、柔軟性、再利用性、拡張性、保守性といった性能が強く求められる。

一般に、個々の解析コードは多くの高度な計算機能を有しているが、従来のように、ある解析コードに異なる解析コードの機能を単に組み込んで、新たな解析コードを作成するという方法では、組み込まれた解析コードは肥大化・複雑化する一方である。また、組込まれた解析コードは、オリジナルとは少し異なったコピーが増え続けることにもなる。このような解析コードの構築方法は、工学システムを設計する上で考慮すべき要求事項が増えるに従い、必要となる新たなプログラム改修や保守・管理等の労力を著しく増大させるものであり、解析コード利用の柔軟性という観点からはもはや開発投資に見合うだけの利用価値がないものとなる恐れがある。

本研究では、このような課題を受けて、主に原子力に関連する様々な専門分野で開発されてきた解析コードを、ニーズにあわせてより柔軟にかつ迅速に組み合わせて利用できるような新しいタイプの解析システム（次世代解析システム）の開発を目的とする。ここで目標としている次世代解析システムは、基本的に原子力分野をターゲットとしているが、原子力に関連する技術分野は多岐にわたるため、多くの異なる分野の物理データや工学的モデル・手法を、いかにうまく結合して利用できるようにするかという点に大きな課題がある。

前作の報告書[1-1]では、次世代解析システムの基盤技術となりうる要素技術の調査結果をまとめた。本報告書では、この要素技術調査結果を受けて、高速炉の機器構造解析と炉心核特性解析の分野におけるプロトタイプを作成し、次世代解析システムの可能性について

検討した結果をまとめる。

第２章では、高速炉機器構造解析に関して、Visual Basic .NETとFortranを用いて行った検討結果について述べる。第３章、第４章では、それぞれ、Fortran90、Pythonを用いて、高速炉炉物理解析を対象に検討した結果をまとめる。第５章で、全体のまとめと今後の課題について述べ、結言とする。

第２章　多言語によるプロトタイプ（PARTS.NET）の開発

第2章　多言語によるプロトタイプ（PARTS.NET）の開発

2.1　概要

高速炉機器構造の支配過重である過渡熱応力を最小とする優れた構造形状および運転法を探索するため、熱過渡現象の計算機シミュレーションをベースとした、新しい高温構造設計体系の確立を目指している。

過渡熱応力は、1.冷却材過渡熱変化（流体）、2.構造温度応答（熱）、3.構造応力応答（構造）の複合現象により生じる。通常の方法で、系統動特性コード、多次元熱流動解析コード、構造非定常熱伝導コード、静的応力解析コード、および強度評価コードを逐次実行して計算すると、1ケースの評価に数ヶ月かかることもあり、数値シミュレーションによる最適構造探索は困難である。

このため、自律したサブシステムである 1. 動特性解析部品 PARTS-FLOW、2.構造計算部品 PARTS-STRESS を、分散オブジェクト技術によって協調させながら高速で実行する過渡熱応力リアルタイムシミュレータの開発を行っている。

これまで Smalltalk 言語でプロトタイプを開発してきたが、以下のようなニーズが新たに生じた。

1. Smalltalk 言語がサポートされなくなったため言語の変更が必要となった。
2. 実用化戦略調査研究に対応するため、新しい計算部品の追加が必要となった。
3. 水・蒸気系等の計算精度を向上させるため、既存の Fortran コードを利用する必要が生じた。

上記ニーズを満たすため、多言語（マルチランゲージ）によるプロトタイプ（PARTS.NET）への全面的な書き換えを以下の手順で実施した。

1. オブジェクト指向設計
   Model View Control モデル化（MVC という）に基づく設計
   設計ツールとして UML（Unified Modeling Language）を利用

2. MVC の各計算部品に対する最適な言語を使用した（マルチランゲージ）
   - 制御部（Control）　→ VisualBasic.NET 言語
   - モデル部（Model）　→ Fortran90 言語　及び　VisualBasic.NET 言語
   - インターフェース部(View) →　SoftWIRE.NET 言語ツール

3. データ入力に加えプログラムの制御方法の変更も可能な先進の GUI の実現

更にモデル部（Model）については、以下の作業も実施した。

(1) 熱流動計算部品のアルゴリズム変更
独自開発部品 → 既存コード（Super-COPD 部品）※1 に変更

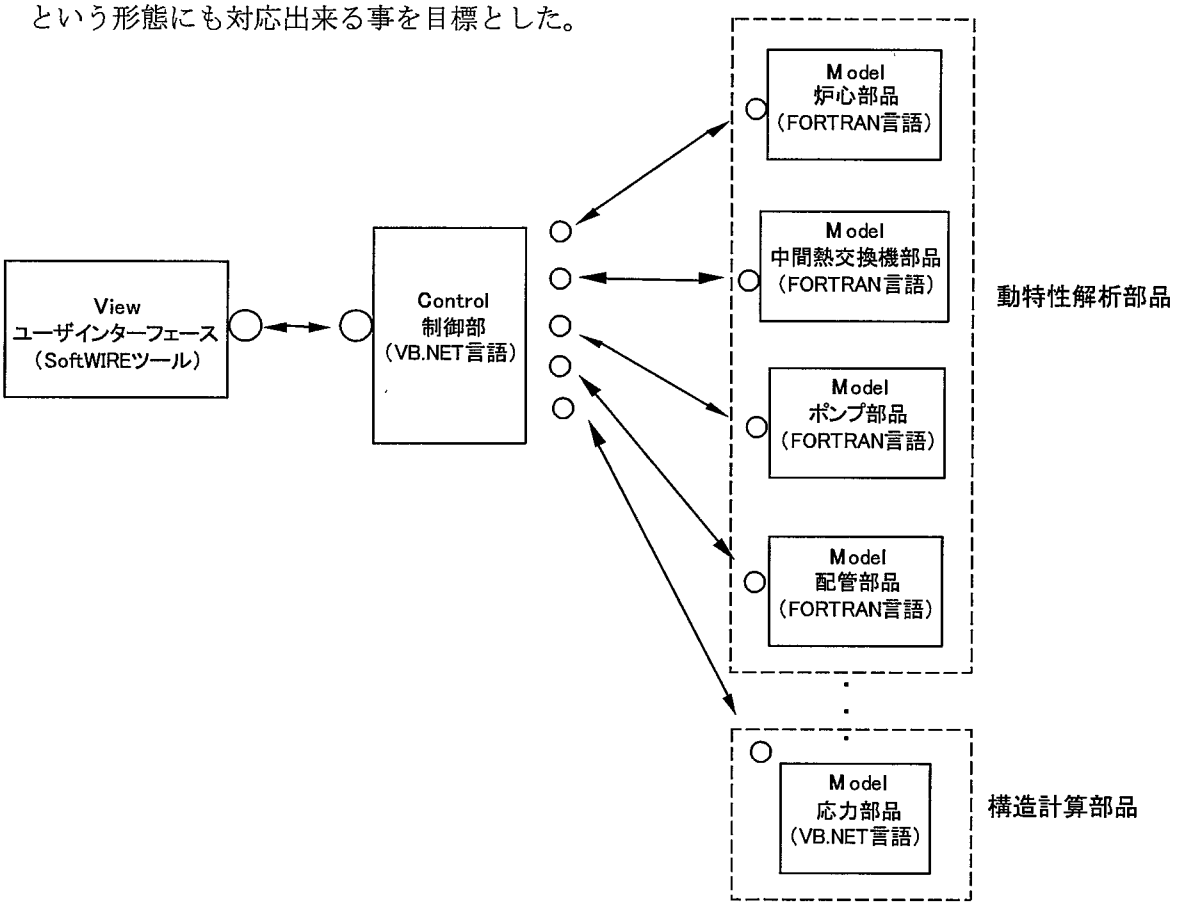
(2) 応力計算部品の変更
複数の冷却材温度変化と熱伝達率の切り替えに対応

システム開発に際しては、オブジェクト指向による分析 / 設計 / プログラミング開発手法である、OOA（Object Oriented Analysis） / OOP（Object Oriented Programming）を実施した。分析段階から UML の静的クラス図・シーケンス図等を作成し、これらに基づいてプログラムレベルのクラスコードを自動生成し開発を行う手法である。

2.2 オブジェクト指向設計方法論（OOAD）による開発

システム開発は、オブジェクト指向による分析 / 設計 / プログラミング開発手法である、OOA（Object Oriented Analysis）/ OOD（Object Oriented Development）を実施した。分析段階からUMLの静的クラス図・シーケンス図等を作成し、これらに基づいてプログラムレベルのクラスコードを生成し開発を行う手法である。

更にオブジェクト指向設計において「機能の明確なる分離」を行うための技術であるMVCモデル（Model / View / Control）を導入することにより、システムを以下の3つの機能要素に分割し各要素の独立性・汎用性を高め、計算処理部（Model）を別計算機で実行させるという形態にも対応出来る事を目標とした。



※ 実行環境に依存しないシステム構成方式（機能毎に異なる計算機で動作する）

図 2.2.1 PARTS システム 部品構成図（MVC モデル化）

Model = 各機器部品（IHX：中間熱交換機等）
View = ユーザインターフェース
Control = 制御部

OOAD 自体は、システム全体の要求を一度に聞いて、一遍に設計し、更に一遍に実装するのは無理があるため、少しずつ分析し、少しずつ設計し、少しずつ実装し、というサイクルを段階的に繰り返していこうというプロセスである。こうすることで、理解できたところ、要求が固まったところから作業を前に進められ、誤りの修正も 1 サイクル以内に行えるという柔軟性が手に入る訳である。

反復的でインクリメント

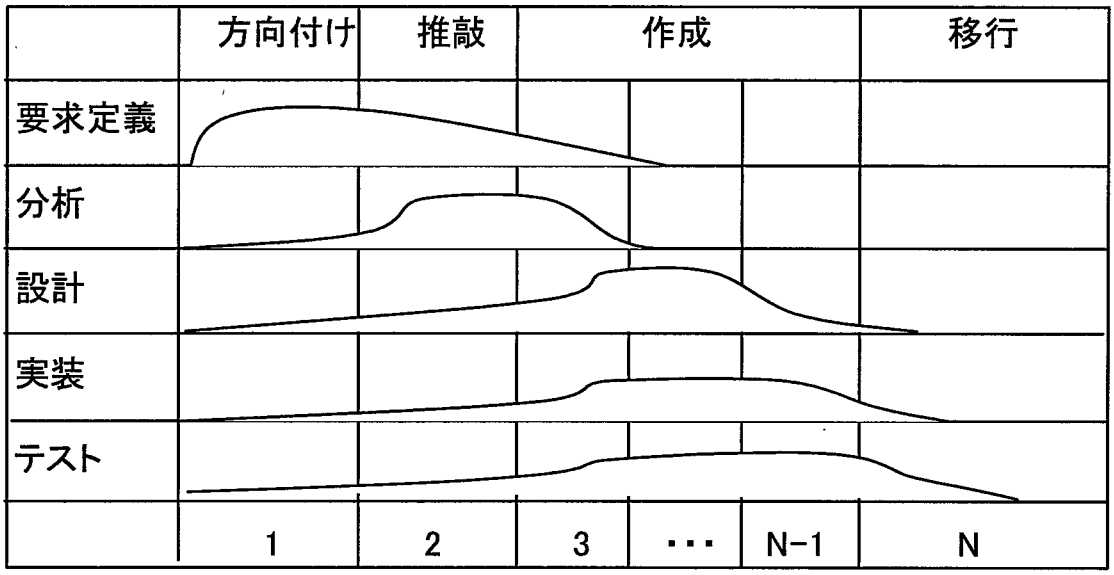


図 2.2.2　反復的でインクリメントな開発サイクル

UML（Unified Modeling Language：統一モデリング言語）は、オブジェクト指向に基づいて 4 つの観点から対象問題を分析し、その結果を 10 の図表として記述する方法であくまで表記法であり、重要なのは 4 つの観点から問題を分析するそのやり方である。

4 つの観点と UML ダイアグラムの対応を以上に示す。

(1)　モデル化対象に対する要求を表す。
　　UseCase 図
(2)　モデル化対象の静的な振舞いを表す。
　　Class 図、Object 図、Package 図
(3)　モデル化対象の動的な振舞いを表す。
　　Sequence 図、Collaboration 図、Activity 図、StateChart 図
(4)　モデル化対象の物理的な実装を表す。
　　Component 図、Deploy 図

OOAD は、UML の各種ダイアグラムを作成及び修正することで詳細化を推し進め、やが

てプログラムコードレベルまで落とし込むやり方である。

OOAD の成果物である UML ダイアグラムを以降に示す。

- PARTS システムの概念モデル図（システム全体）図 2.3.3
- PARTS システムの静的クラス図（システム全体）図 2.3.4
- PARTS システムの静的クラス図（設備部品）　図 2.3.5
- PARTS システムの静的クラス図（材料部品）　図 2.3.6
- PARTS システムの静的クラス図（応力計算部品）図 2.3.7

等

尚、開発言語としては、以下のものを使用した。

開発言語ツール ： Microsoft VisualStudio.NET Enterprise Architect
Microsoft VisualBasic.NET ＋ Fortran90 ＋ SoftWIRE.NET

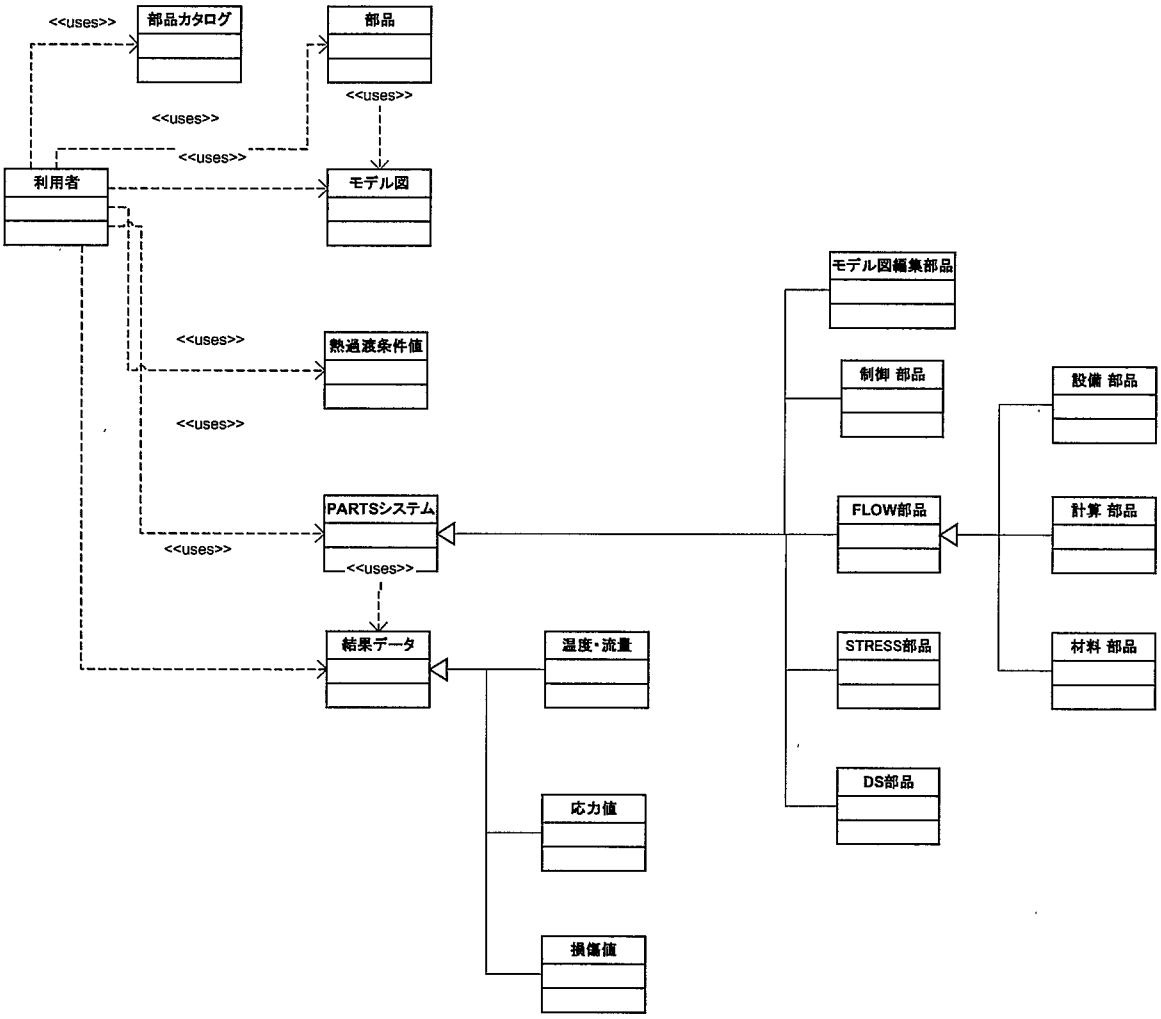


図 2.2.3　PARTS システムの概念モデル図（システム全体）

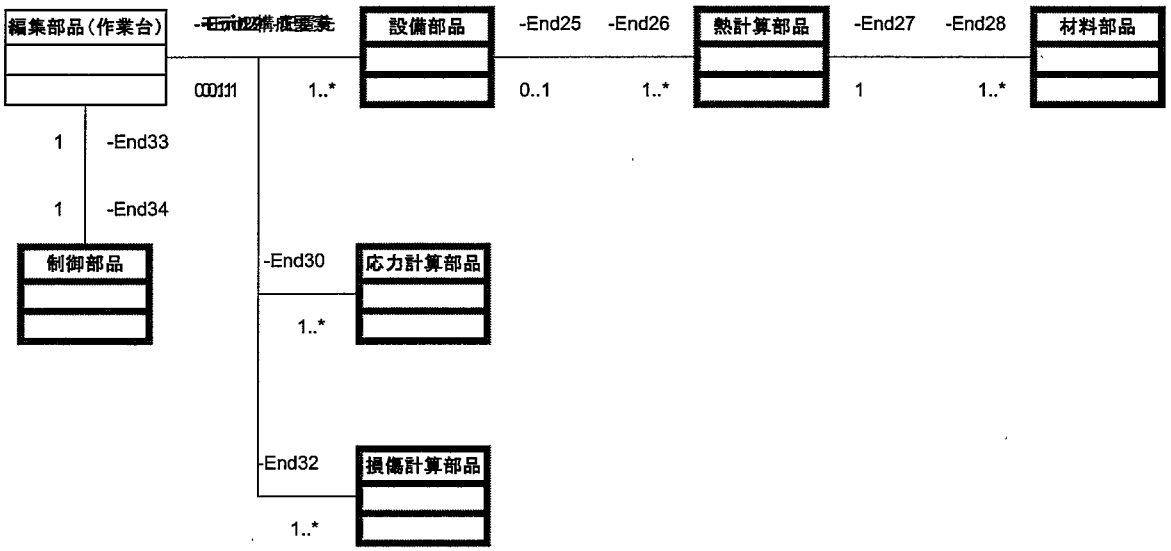


図 2.2.4　PARTS システムの静的クラス図（システム全体）

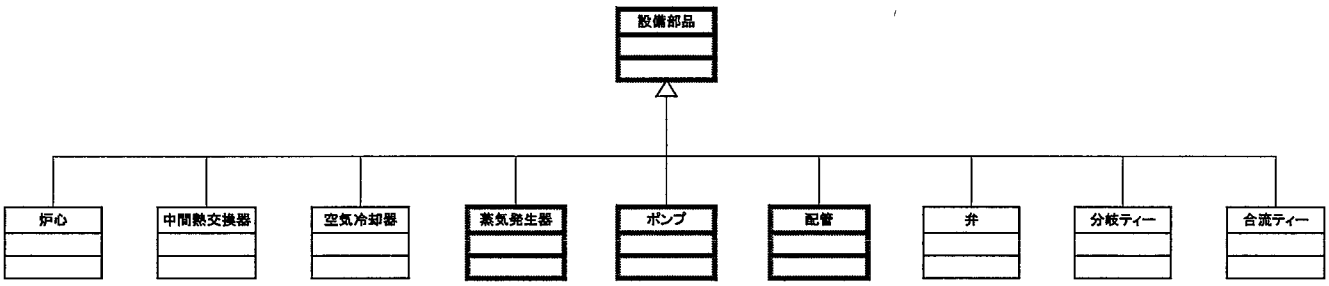


図 2.2.5　PARTS システムの静的クラス図（設備部品）

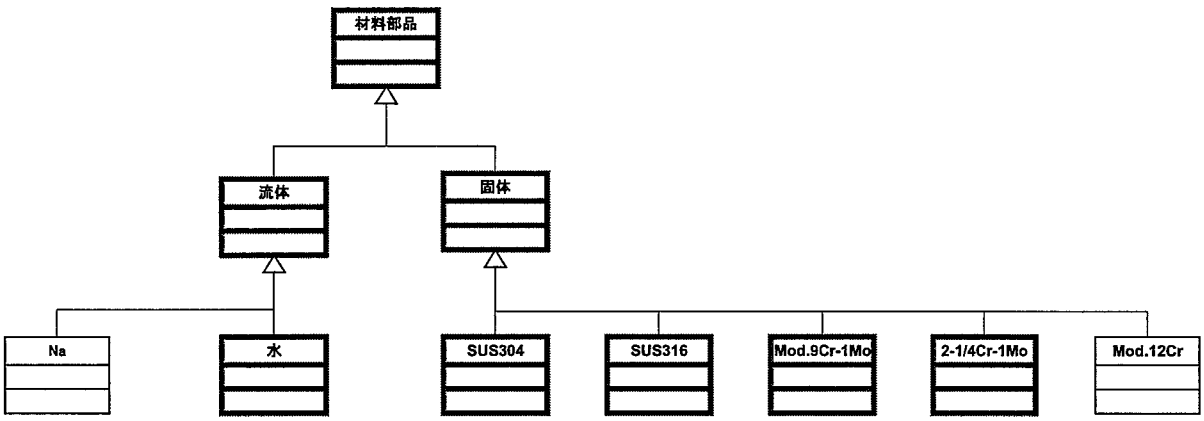


図 2.2.6　PARTS システムの静的クラス図（材料部品）

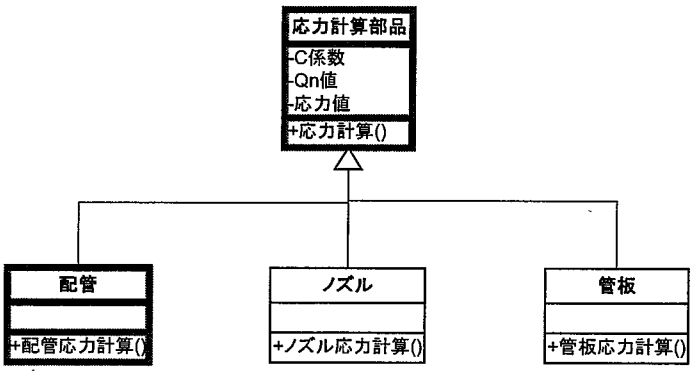


図 2.2.7　PARTS システムの静的クラス図（応力計算部品）

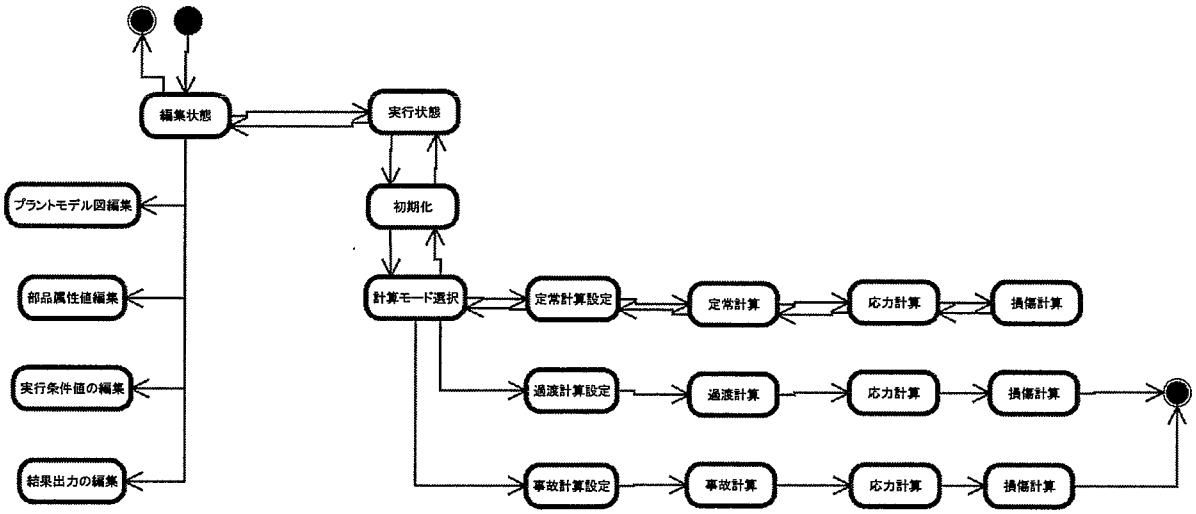


図 2.2.8　PARTS システムの StateChart 図－システム全体

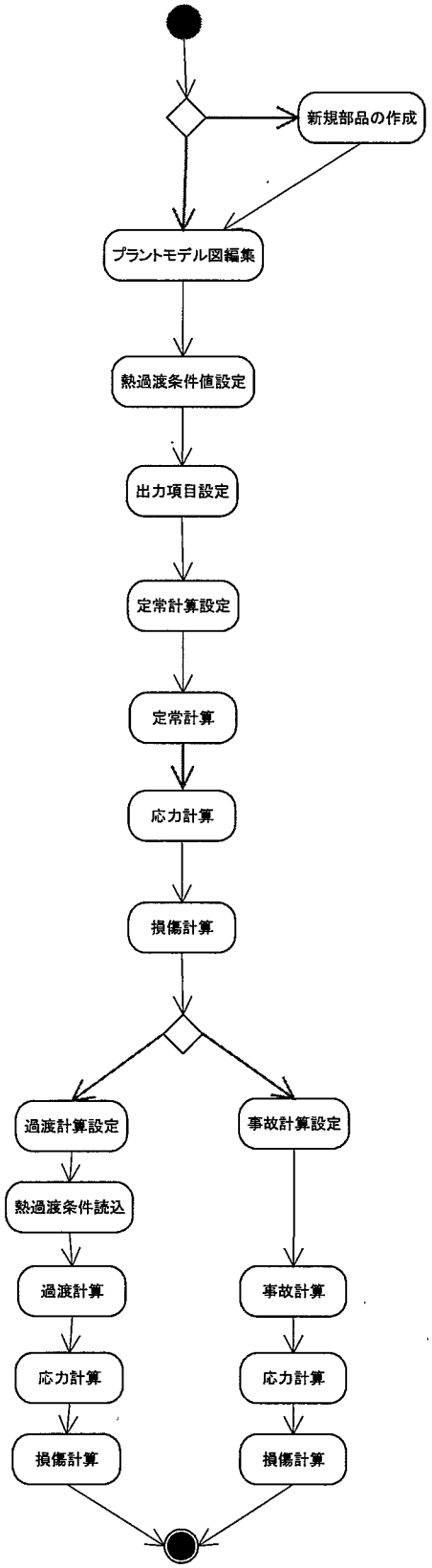


図 2.2.9　Activity 図－システム全体

2.3 GUI 部及び制御部

PARTS.NET システムの GUI（ユーザインターフェース）部は、ビジュアルプログラムツール「SoftWIRE.NET」により実現している。

ビジュアルプログラミング（ビジュアルインターフェース）とは、複雑な事象を単純化・抽象化したシンボルで表わすなどして視覚化し、これらを直接操作することによってプログラミングを行うことであり、近年急速に注目が集まっている。単にプログラムがグラフィクスを使うだけではなく、プログラミングやデバッグまでもビジュアルかつインタラクティブに行えるような「ビジュアルプログラミング・システム」も一部の開発言語用に製品化されている。

ビジュアルプログラミング・システムにおいては、従来のようなテキスト表示でなく、抽象化された視覚的表現によりプログラミングを行い、それをアニメーションさせることによって実行を表現する。これにより、テキストベースの言語に比べてプログラムの動作がわかりやすくなることが期待できる。

更に図的な表現を導入することによって、テキストのみで行なう現在のプログラミングより直観的能力を有効に活用することができるようになる。しかし、現状では、複雑なプログラムを記述するには、画面上が複雑になり過ぎるという問題が出てきている。

こうした問題に対しては、プログラムをシンポル（アイコン）化する際、従来の開発言語の命令と 1 体 1 で対応させるのではなく、関数単位でシンボル化することにより、制御の流れをよりシンプルにする事が可能となる。

ビジュアルプログラミング技術が製品化された例としては、Smalltalk 言語上で動作する製品が一部存在した。しかし、Smalltalk の言語仕様は非常に独特なものであり、習得にかなりの時間を要するという難点があった。

もっと身近で理解しやすい開発言語上で動作するビジュアルプログラミング製品はないかということで登場したのが、SoftWIRE.NET である。最大の特徴は VisualBasic 上で動作することであり、VisualBasic であれば比較的容易な言語仕様であるため、習得期間を必要とせずシステムを構築する事が可能である。

操作方法は、到って簡単でシンプルなメニューから必要なファンクションを選び、オブジェクトを画面の好きな場所に配置し、ドラッグ・アンド・ドロップでさっとワイヤを描いてオブジェクトをつないでゆけば、あとはプログラムを実行するだけである。それだけでプログラムは完成する。

アイコンの 1 つ 1 つが実機（例：中間熱交換機）に対応しており、任意に組み合わせることでプラント解析モデルが作成出来る。部品としては 動特性解析部品、構造計算部品 等がある。

図 2.3.1 のプラントを SoftWIRE.NET で記述したものが図 2.3.2 である。

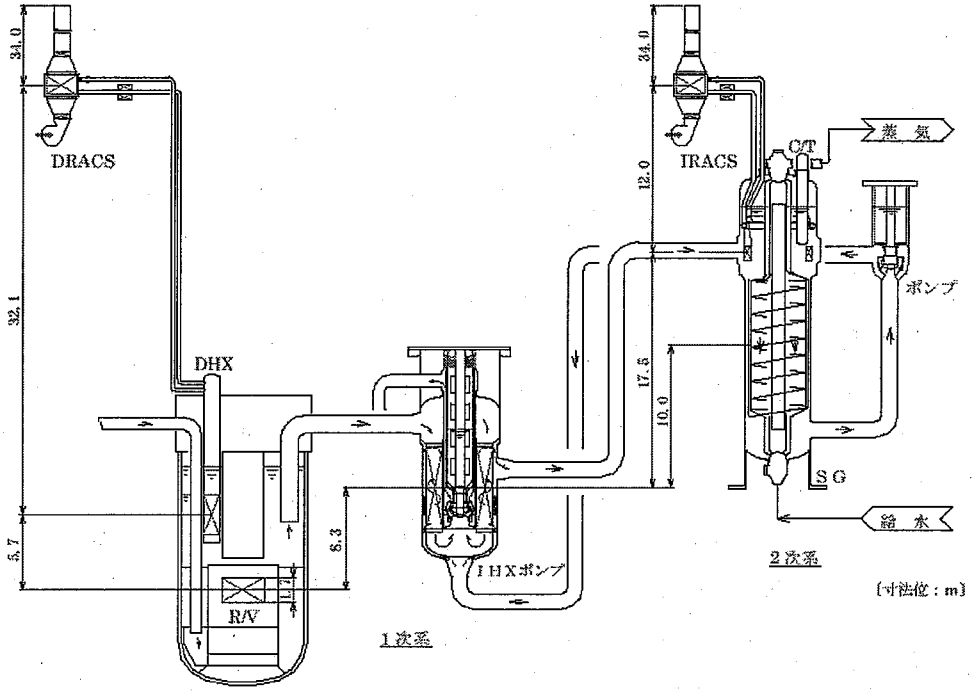


図 2.3.1　解析対象のプラント図

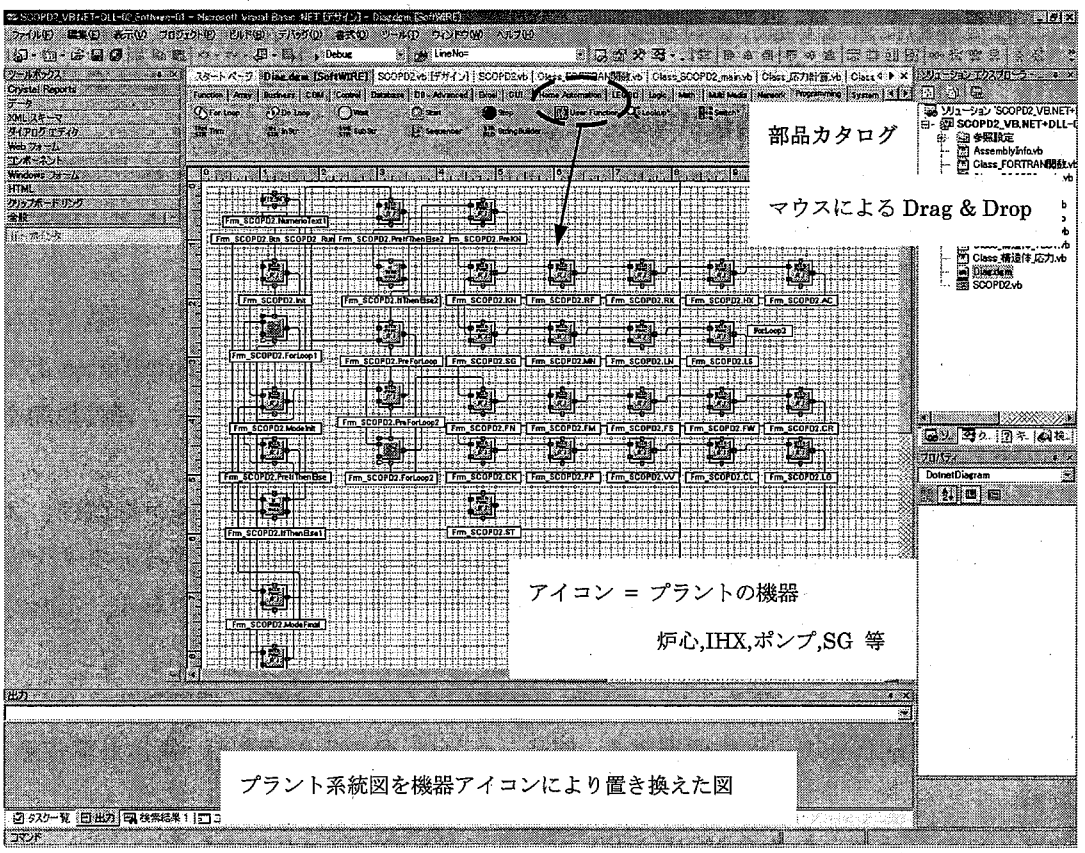


図 2.3.2　部品編集時画面（SoftWIRE.NET）

図 2.3.2 の部品編集画面の個々のアイコンはプラントの各機器を表しており、更に当該機器に対する Super-COPD の Fortran モジュールと対応している。プログラム起動時には、VB.NET で記述された制御プラグラムがアイコンの連結関係に応じて Fortran モジュールとデータ通信を行いながら、シミュレーションを実行する。

Fortran モジュールについては、そのままでは VB.NET 言語から呼び出せないため、I/F プログラムを付加することにより外部呼出し可能な形式のライブラリを構築した。

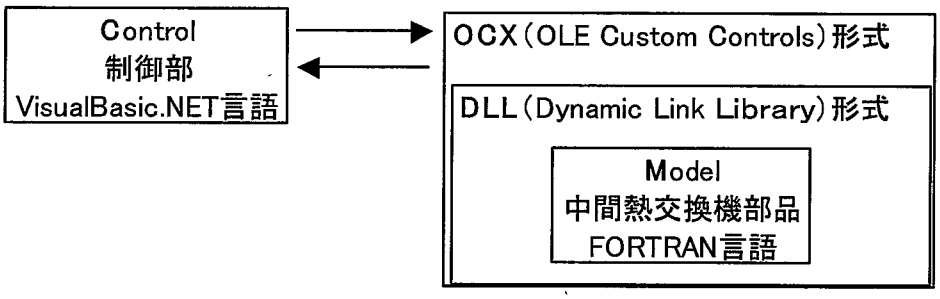


図 2.3.3　OCX 形式の Model 部品

実際のコードを図 2.3.4 に示す。

また、各アイコン（部品）間で受け渡されるパラメータを図 2.3.5 に示す。

膨大な数の項目については、幾つかのカテゴリに分類し、構造体として定義した。構造体を利用するため、Fortran77 コードに対しては若干のデータ形式変換プログラムを付加して上で Fortran90 にてリコンパイリングした。構造体を利用すると、その名称を受け渡すことでパラメータ送受信が完了するため、プログラム上はかなり整理される。

```
'-----------------------------------------------------------------------
' Module      : RXCONT
' Description : 原子炉熱計算詳細モデル
' SystemName  : モジュール型プラント動特性解析コード Super-COPD v2
' Dev-Language: Microsoft VisualBasic.NET
' Update      : 2002.10.03 (Thu)
'-----------------------------------------------------------------------
Private Sub RX_PostRunBlock(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles RX.PostRunBlock
    '-- MODULE RX -
    Call UserControl_RXCONT01(J1,aStruct_Local.NINT,aStruct_Local.TIME, Struct_AAC1.ETMIN, aStruct_Local.IRSTAT)
End Sub
'-----------------------------------------------------------------------
' Module      : HXCONT
' Description : 中間熱交換器（ＩＨＸ）熱計算
' SystemName  : モジュール型プラント動特性解析コード Super-COPD v2
' Dev-Language: Microsoft VisualBasic.NET
' Update      : 2002.10.03 (Thu)
'-----------------------------------------------------------------------
Private Sub HX_PostRunBlock(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles HX.PostRunBlock
    '  -MODULE HX-
    Call UserControl_HXCONT(J1, aStruct_Local.NINT, aStruct_Local.TIME, aStruct_Local.ETMIN, aStruct_Local.IRSTAT)
End Sub
'-----------------------------------------------------------------------
' Module      : ACCONT
' Description : 空気冷却器熱計算
' SystemName  : モジュール型プラント動特性解析コード Super-COPD v2
' Dev-Language: Microsoft VisualBasic.NET
' Update      : 2002.10.03 (Thu)
'-----------------------------------------------------------------------
Private Sub AC_PostRunBlock(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles AC.PostRunBlock
    '-- MODULE AC -
    Call UserControl_ACCONT(J1, aStruct_Local.NINT, aStruct_Local.TIME, aStruct_Local.ETMIN, aStruct_Local.IRSTAT)
End Sub
'-----------------------------------------------------------------------
' Module      : SGCONT
' Description : 蒸気発生器（ＳＧ）熱計算
```

図 2.3.4　SoftWIRE アイコンに対応した Fortran モジュール

```
'-------------------------------------------------------------------
' クラス名    = Struct_AAP
' 概 要       = 制御ファイル（入力）に関する構造体クラス
'-------------------------------------------------------------------
Public Structure Struct_AAP
    '---------------------------------------------------------------
    ' Class Variable
    '---------------------------------------------------------------
    'Public Const KNTP = 5000          'プール変数ＴＥＭＰ 最大数
    Public Const KNTP = 2500          'プール変数ＴＥＭＰ 最大数
    Public Const KNFF = 100           'プール変数ＦＬＷＮ 流路最大数
    Public Const KNWF = 20            'プール変数ＦＬＷＮ 流路網最大数
    Public Const KNOP = 50            'プール変数ＰＲＳＮ 圧力ノード最大数
    Public Const KNWP = 20            'プール変数ＰＲＳＮ 流路網最大数
    Public Const KNPM = 16            'プール変数ＰＭＰＮ 関連プロセス量最大数
    Public Const KNNM = 16            'プール変数ＰＭＰＮ ポンプ最大数
    Public Const KNPV = 5             'プール変数ＶＬＶＮ 関連プロセス量最大数
    Public Const KNNV = 120           'プール変数ＶＬＶＮ バルブ最大数
    Public TEMP() As Double           'プール変数 : 出力，温度，反応度など
    Public FLWN(,) As Double          'プール変数 : 流路網ｊ，流路ｉの流量
    Public PRSN(,) As Double          'プール変数 : 流路網ｊ，圧力ノードｉの圧力
    Public PMPN(,) As Double          'プール変数 : ｊ番目ポンプ，ｉ番目プロセス量
    Public VLVN(,) As Double          'プール変数 : ｊ番目バルブ，ｉ番目プロセス量
    Public Sub ReDim_AAP()
        ReDim TEMP(KNTP)
        ReDim FLWN(KNFF, KNWF)
        ReDim PRSN(KNOP, KNWP)
        ReDim PMPN(KNPM, KNNM)
        ReDim VLVN(KNPV, KNNV)
    End Sub
End Structure
```

図 2.3.5　各部品間で受け渡されるパラメータ（構造体）

2.4 動特性解析部品（PARTS-FLOW）

以前のシステムでは、Smalltalk 言語による独自開発の系統温度応答部品を使用していたが、水／蒸気系の計算精度・機能拡張性等の基準を検討すると、設計作業に使用された実績のある既存コードを流用することが優れていると判断し、プラント動特性解析コードである「Super-COPD」を部品化して使用することにした。
Super-COPD コードは、タンク型炉を含む多様な高速増殖炉概念のプラント動特性評価が可能な動特性解析コードである。

Super-COPD コードは、蒸気モデル及び対象についてはすべて解析可能であると共に配管破断等についても、配管よりの消滅モデルを含んでおり、標準的な計算は可能である。また、Super-COPD は、汎用フローネットワーク及びモジュール構造の採用により、次の FBR システムに適用できる。

1) 系統構成…… ループ、タンク、2 次系削除型
2) 熱交換器型式…… 向流型、並流型、直交流型、直管型、ヘリカルコイル型、
有液面、無液面
3) ＳＧ系…… 一体貫流、再循環、分離貫流、ズルツァー
4) 崩壊熱除去系…… ＤＲＡＣＳ、ＩＲＡＣＳ、ＳＧＡＨＲＳ
5) 冷却材…… ナトリウム／鉛／鉛ビスマス、水等

また、解析効率の向上のため各モジュール単体で実行可能とすると共に、リスタート機能も有している。

この Super-COPD をオブジェクト指向言語（VisualBasic.NET）から呼び出せるように部品化した訳だが、以下の手順で実施した。

動特性解析部品の開発手順

- 機能単位で Fortran モジュールを切り出す
- DLL（Dynamic Link Library）形式のライブラリを構築
  部品の実体は Fortran 関数であり、インターフェース（及びパラメータ）のみ .NET プラットフォームに対応した形式にし、VisualBasic.NET 言語から呼び出している。
- VisualBasic.NET 言語から DLL ライブラリ内の部品を呼び出し実行する

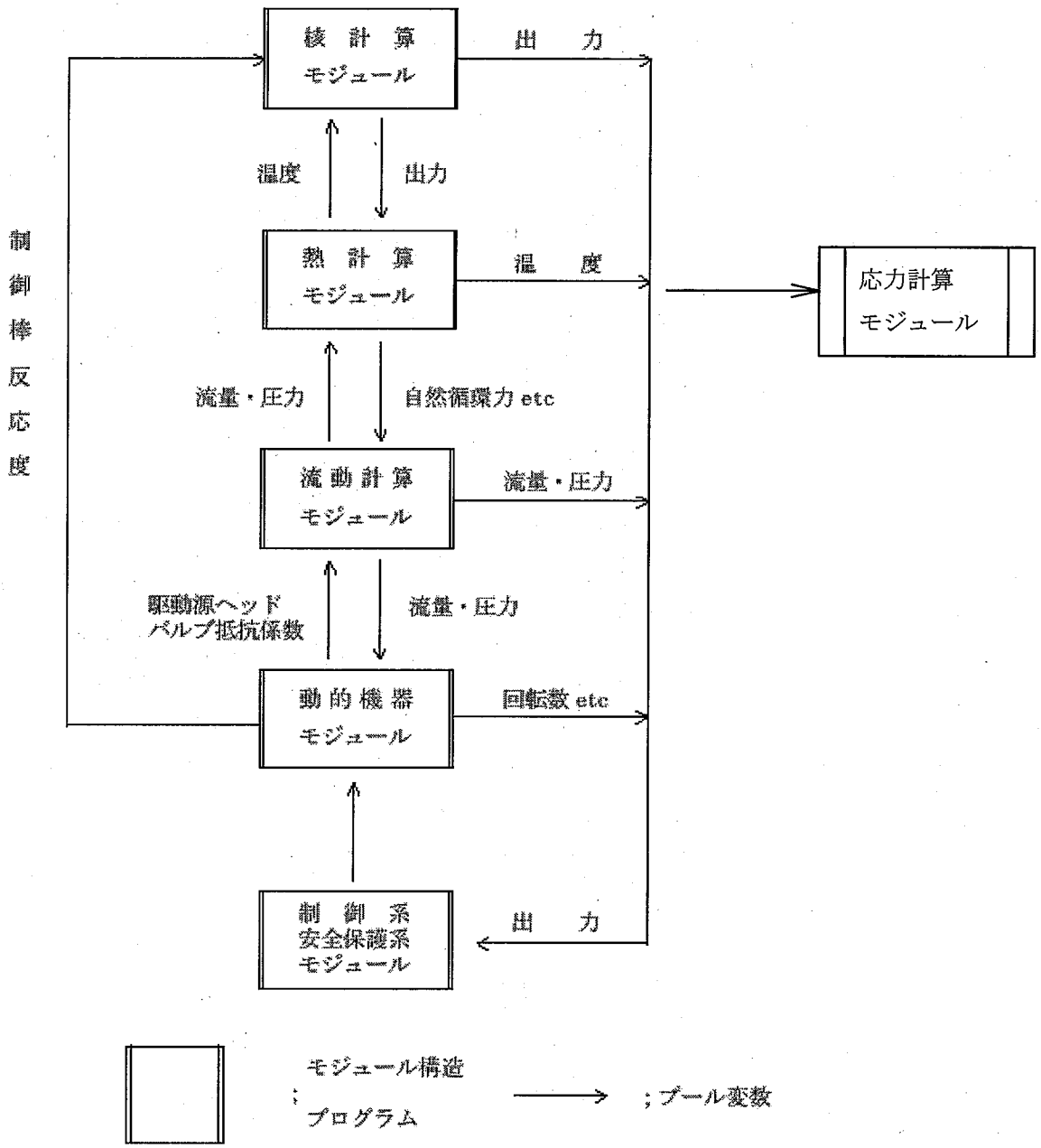


図 2.4.1　Super-COPD のモジュール構成及び処理の流れ

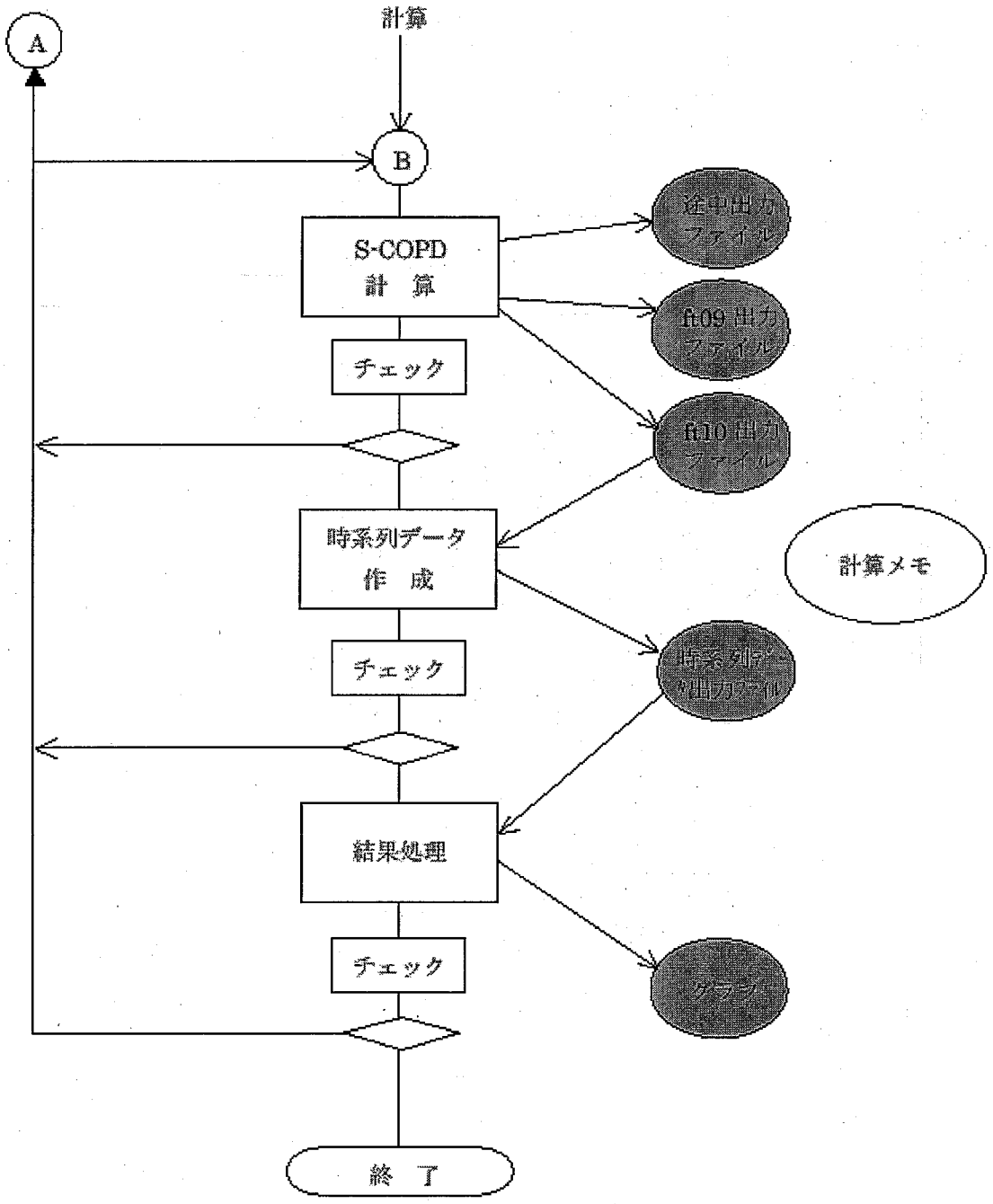


図 2.4.2 Super-COPD 計算作業の流れ

名　称	機　　能	備　　考
KN	原子炉核計算	核計算モジュール
RX	原子炉詳細熱計算	熱計算モジュール
HX	中間熱交換器熱計算	
AC	空気冷却器熱計算	
SG	蒸気発生器熱計算	
MN	完全混合，ミキシング・ティ要素熱計算	
LN	冷却系配管要素熱計算	
LS	水・蒸気系配管要素熱計算	
FW	水・蒸気系熱流動計算	熱流動計算モジュール
FN	非圧縮性流体流動計算	流動計算モジュール
FS	非圧縮性流体流動計算（分岐・合流なし）	
CR	制御棒の計算	動的機器モジュール
PP	循環ポンプの計算	
VV	各種弁の計算	
CL	制御系の計算	制御系・安全保護系モジュール
LG	安全保護系の計算	

表 2.4.1　SUPER-COPD 計算モジュール一覧
（Fortran モジュール）

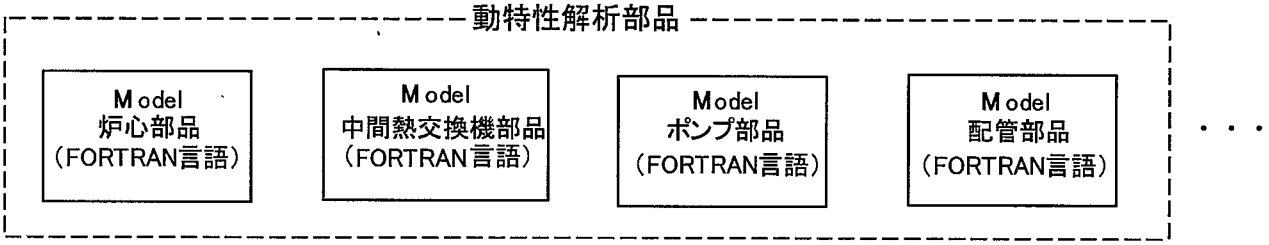


図 2.4.3　部品化された Super-COPD モジュール

従来の Fortran モジュールを機能毎に切り出し、オブジェクト指向言語から呼び出し実行出来るように独立した部品として再構築した。単体実行あるいはどのような組み合わせでも実行可能である。

変数名	ft番号	意　味	備考
td07	ft07	入力データファイル名（基本データ）	(注1)
td05	ft05	入力データファイル名（変更データ）	(注2)
s06c	ft06	途中出力データファイル名	
s09c	ft09	プール変数情報出力データファイル名	
s10c	ft10	プール変数データ出力データファイル名	
s40c	ft40	コモン変数データ出力データファイル名	
s11c	ft11	リスタート時入力データファイル名 （リスタート前ft09出力ファイル名を入力）	(注3)
s41c	ft41	リスタート時入力データファイル名 （リスタート前ft40出力ファイル名を入力）	(注3)

(注1)　基本データ　通常外乱の入らないデータ

(注2)　変更データ　通常外乱、パラメータ、タイムメッシュ、出力コントロール等のデータ

(注3)　通常ランの場合は、計算では、使用しない。

表 2.4.2　Super-COPD 入出力ファイル一覧

```
&NAMMN2
      VMIX(1)  =  2*9.350E3,  2*2.464E4, 2*1.0,   2*1.578E4,  2*5.987E4,
                  6*0.0,      3*1.241E3,  3*1.0,
      VMIX(11) =  2*2.260E4,  2*6.59E3,  2*0.001,
      VMIX(23) =  2*1.700E4,  2*1.986E4,
      VMIX(27) =  2*4.901E4,  1.39E4,  5.42E4,  7.53E4,  9.68E4,
                  6.32E4,     1.86E5,
      ZNFM(1)  =  2*0.4,  2*1.5,  2*1.6,  2*2.2,  2*2.0,  2*0.0,  2*1.0,
      ZNFM(20)=  0.305, 0.0, 0.0,
      ZNFM(23)=  2*3.135,  2*2.3,  2*1.355,  1.3,  0.0,  2.0,  2.305,  1.5,  0.0,
      WGPM(1)  =  2*90.0,  2*-90.0,  2*-90.,  2*90.,  2*-90.0,  2*0.0,  2*90.0,
      WGPM(20) =  -90.0, 0.0, 0.0,
      WGPM(23) =  2*90.,   2*-90.,  2*-90.0,  6*90.0,
&END
&NAMKN1
      IMDLK  = 1,   IOPTK  =1, IRCF   = 12,
      KLDK   = 0,   PFO    =0.0,
      NINTK  = 3,  TMAXK  = 1.0E+10,
      DDELK(1) = 5*0.0001,
      IOTIMK = 0,
      ISTAK  = 1,   IDBGK  = 2,
      OUTK(1)=0.01, 1.0, 5.0, 10.,
      NRTOT    = 9,     NRD    = 4,    NRC(1)  = 5, 6, 7, 8, 42, 43, 44, 45,
      NRR      = 100,   NRS    = 2,    NRG     = 3,
      NPF      = 11,    NTAV(1) = 31,  32,  33,  34, 38, 39, 40, 41,
                        NTAV(9) = 61, 62, 63, 64,
&END
&NAMKN2
      PNLT     = 3.85E-7,
      BE(1)    = 7.81E-5,  7.31E-4,  6.52E-4,  1.30E-3,  6.11E-4,  1.77E-4,
      RA(1)    = 0.0130,  0.0313,  0.134,  0.347,  1.40,  3.81,
      NFTBL(1)   = 12*2,
      TFTBL(1,1) = -10000.0, 10000.0,
      TFTBL(1,2) = -10000.0, 10000.0,
      TFTBL(1,3) = -10000.0, 10000.0,
      TFTBL(1,4) = -10000.0, 10000.0
```

図 2.4.4　SUPER-COPD 用入力データ（td07）

```
      SUBROUTINE HXCONT(J1      ,NINT ,TIME ,ETMIN ,IRSTAT)
      !DEC$ATTRIBUTES DLLEXPORT, ALIAS:'HXCONT' :: HXCONT
      INTEGER*4 J1        [VALUE]
          INTEGER*4 NINT      [VALUE]
          REAL*8    TIME      [REFERENCE]
          REAL*8    ETMIN     [VALUE]
          INTEGER*4 IRSTAT    [VALUE]
      INCLUDE 'COMMON/HXA.H'                                    00003400
      INCLUDE 'COMMON/HXM.H'                                    00003500
      IF( J1 .EQ. 1 .OR. J1 .EQ. 3 ) THEN                       00004200
      CALL HXREAD( J1 )                                         00004300
      ELSE IF( J1 .EQ. 2 ) THEN                                 00004700
      JR  =  2                                                  00004800
      CALL HXRSTA(JR)                                           00004900
      ELSE IF( J1 .EQ. 4 ) THEN                                 00005300
      DO 400 II= 1,NNHX                                         00005500
      IF( IRSTAT    .EQ. 1    ) GO TO 400                       00005700
      IF( IMDLHX(II) .EQ. 0   ) GO TO 400                       00005800
      IF( NINTHX(II) .NE. NINT ) GO TO 400                      00005900
      CALL HXSTAT(II)                                           00006100
  400 CONTINUE                                                  00006300
      ELSE IF( J1 .EQ. 5 ) THEN                                 00006700
      IF( IRSTAT .EQ. 2 ) THEN                                  00006900
      JR    = 1                                                 00007200
      CALL HXRSTA(JR)                                           00007300
      ELSE                                                      00007400
      DO 500 II= 1,NNHX                                         00007600
      IF( IMDLHX(II) .EQ. 0       ) GO TO 500                   00007800
      IF( ETHX(II)  .GT. TMAXHX(II)) GO TO 500                  00007900
      IF( ETHX(II)  .EQ. ETMIN    ) CALL HXTRNS(II)             00008000
      IF( TIME .GE. ETHX(II) ) TIME = ETHX(II)                  00008100
  500 CONTINUE                                                  00008300
      END IF                                                    00008500
      END IF                                                    00008800
      RETURN                                                    00009000
      END                                                       00009100
```

図 2.4.5　DLL 用に Interface 部を修正した Fortran プログラム（中間熱交換機：IHX）

```
      SUBROUTINE HXTRNS( II )
C     MODULE HX TRANSIENT CALCULATION CONTROL
C ====MODULE SUBROUTINE====
      INCLUDE 'COMMON/AAA.H'
      INCLUDE 'COMMON/AAP.H'
      INCLUDE 'COMMON/HXA.H'
      INCLUDE 'COMMON/HXM.H'
      DIMENSION RO(KMHX)
      N             = MHX2(II)
      NP1           = MHX1(II)
C ----INPUT OF THIS MODULE----
      GPHX(N)       = FLWN(NFPHX(II),NWPHX(II))
      GSHX(N)       = FLWN(NFSHX(II),NWSHX(II))
      IF( GPHX(N) .LT. 0.0 ) THEN
      TPOTHX(N)     = TEMP(NTPOHX(II))
      ELSE
      TPINHX(N)     = TEMP(NTPIHX(II))
      END IF
      IF( GSHX(N) .LT. 0.0 ) THEN
      TSOTHX(N)     = TEMP(NTSOHX(II))
      ELSE
      TSINHX(N)     = TEMP(NTSIHX(II))
      END IF
      TARHX(N)            = TEMP(NTARHX(II))
      TEXHX(N)            = TEMP(NTEXHX(II))
        IF( KPR1HX(II) .EQ. 1)  THEN
          DO 231 K=1,NENDHX(II)
          CALL ZFLUID( KPR5HX(II),1,TPHX(NP1,K),CPPHX(II,K) )
          CALL ZFLUID( KPR5HX(II),2,TPHX(NP1,K),ROPHX(II,K) )
          CALL ZFLUID( KPR5HX(II),3,TPHX(NP1,K),PRPHX(II,K) )
          CALL ZFLUID( KPR5HX(II),4,TPHX(NP1,K),VKPHX(II,K) )
          CALL ZFLUID( KPR5HX(II),5,TPHX(NP1,K),TKPHX(II,K) )
          IF( KCOWHX(II,1) .EQ. 1)
     1    CALL ZFLUID( KPR5HX(II),2,TPTOHX(NP1,K),RPTOHX(II,K) )
          IF( IOP1HX(II).EQ.1 .AND. KCOWHX(II,3) .EQ. 1)
     1    CALL ZFLUID( KPR5HX(II),2,TPVIHX(NP1,K),RPVIHX(II,K) )
```

図 2.4.6 Fortran プログラム（中間熱交換機：IHX）

```
C-----------------------------------------------------------------------
C     STRUCTURE : Stress 1/3
C-----------------------------------------------------------------------
      STRUCTURE /Stress/
        REAL*8  Time(2000)
        REAL*8  Stress(2000)
        REAL*8  CActTime(2)
        REAL*8  DeltTime
      END STRUCTURE
C-----------------------------------------------------------------------
C     STRUCTURE : Stress ( CaseNo, StressElementNo ) EVALUATION POINT 2/3
C-----------------------------------------------------------------------
      STRUCTURE /StressCaseElm/
        RECORD /Stress/ S(3, 4)
      END STRUCTURE
C-----------------------------------------------------------------------
C     STRUCTURE : Stress( 2 ) ... 1st Or 2nd
C-----------------------------------------------------------------------
      STRUCTURE /StressEval/
        RECORD /StressCaseElm/ S12(2)
      END STRUCTURE
C-----------------------------------------------------------------------
C     STRUCTURE : Stress ( EvaluateNo )
C-----------------------------------------------------------------------
      STRUCTURE /StressModule/
        RECORD /StressEval/ S(10)
      END STRUCTURE
C---
      STRUCTURE /CTRL/
        CHARACTER*16 ControlFileName
        INTEGER*4    EvalPointNo
        INTEGER*4    HeatTransferRateNo
        INTEGER*4    StressElementNo
        INTEGER*4    ScopdModuleNo
        INTEGER*4    ModuleNum1(30)
        INTEGER*4    ModuleNum2(30)
```

図 2.4.7　Fortran プログラム構造体

2.5 構造計算部品（PARTS-STRESS）

PARTS-STRESS は、構造物の応力計算を行うものであるが、任意の冷却材温度変化に対して構造物中に生ずる温度や応力の履歴を計算する手法として Green 関数法を採用している。

Green 関数法は、は有限要素解析等にて求めた構造物の冷却材ステップ温度変化に対する温度及び応力応答（Green 関数）を予めプログラム内部に設定しておき、設計条件から決まる任意の温度変化に対する応答を畳み込み積分によって、詳細な有限要素法を行うことなく評価する手法ある。

PARTS-STRESS は、中間熱交換機（IHX）管板のような複雑な形状で、更に独立した 2 系統の冷却材の熱過渡を同時に受け、熱伝達率が経時的に変化する条件のプラント機器の構造設計にも適用可能である。

PARTS-STRESS には、以下の 2 つの機能を追加した。

1) 構造が接する流体が 2 系統の場合の熱応力解析機能
2) 流体の熱伝達率が変化する場合の熱応力解析機能

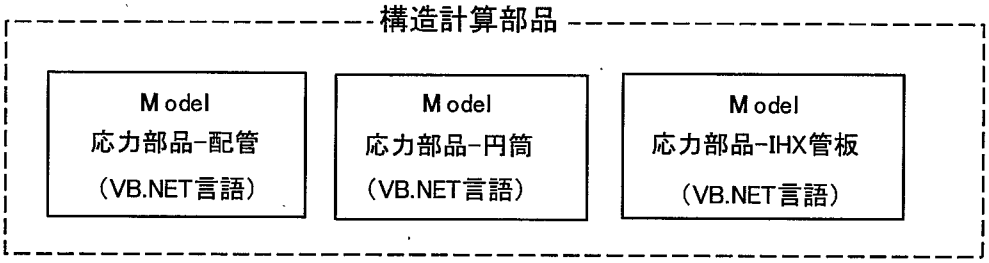


図 2.5.1 構造計算モジュールの部品化

Green 関数法には、C 係数（各構造のステップ温度応答に対する応力履歴の係数）と Qn 値（減衰の時定数を表す、評価時間に影響する）という 2 つパラメータが必要になる。有限要素法の結果を基にこの 2 つのパラメータを設定する訳であるが、表 2.5.1 がその例である。

基本的に 1 つの応力要素に対し、1 つの C 係数（及び Qn 値）が対応している。項目の意味付け、データの階層構造を定義しているのが特徴である。

EvaluatePointNo	1	CaseNo	3	StressElementNo	4							
EvaluatePoint	1	Case	1	StressElement	σR	M-No	16	ThermalTransferRateTime(sec)	0	25	DeltaTime(sec)	0
1	1.27E+02	0										
2	1.13E+01	0.001										
3	-6.12E+00	0.002										
4	3.54E+00	0.005										
5	-4.31E+01	0.01										
6	4.47E+00	0.02										
7	-7.59E+00	0.05										
8	2.76E+01	0.1										
9	-3.50E+01	0.2										
10	-8.21E+01	0.5										
11	1.32E+01	1										
12	-3.42E+00	2										
13	-1.09E+01	5										
14	5.03E+00	10										
15	-7.83E+00	20										
16	4.09E+00	50										
EvaluatePoint	1	Case	1	StressElement	σZ	M-No	16	ThermalTransferRateTime(sec)	0	25	DeltaTime(sec)	0
1	2.11E+02	0										
2	1.60E+01	0.001										
3	-6.41E+00	0.002										
4	3.10E+00	0.005										
5	-6.85E+01	0.01										
6	7.00E+00	0.02										
7	-1.10E+01	0.05										
8	4.13E+01	0.1										
9	-5.67E+01	0.2										
10	-1.36E+02	0.5										
11	2.12E+01	1										
12	-5.20E+00	2										
13	-1.85E+01	5										
14	8.55E+00	10										
15	-1.31E+01	20										
16	6.83E+00	50										
EvaluatePoint	1	Case	1	StressElement	σθ	M-No	16	ThermalTransferRateTime(sec)	0	25	DeltaTime(sec)	0
1	2.64E+02	0										
2	2.95E+01	0.001										
3	-2.61E+01	0.002										
4	1.85E+01	0.005										
5	-7.81E+01	0.01										
6	8.12E+00	0.02										
7	-2.64E+01	0.05										
8	3.93E+01	0.1										
9	-6.51E+01	0.2										
10	-1.64E+02	0.5										
11	2.71E+01	1										
12	-6.94E+00	2										
13	-2.16E+01	5										
14	9.83E+00	10										
15	-1.55E+01	20										
16	8.16E+00	50										
EvaluatePoint	1	Case	1	StressElement	τRZ	M-No	16	ThermalTransferRateTime(sec)	0	25	DeltaTime(sec)	0
1	1.48E+02	0										
2	1.28E+01	0.001										
3	-6.54E+00	0.002										
4	3.66E+00	0.005										
5	-4.94E+01	0.01										
6	5.61E+00	0.02										
7	-9.32E+00	0.05										
8	2.96E+01	0.1										
9	-3.84E+01	0.2										
10	-9.63E+01	0.5										
11	1.56E+01	1										
12	-4.00E+00	2										
13	-1.28E+01	5										
14	5.88E+00	10										
15	-9.14E+00	20										
16	4.773477426	50										

表 2.5.1 Green関数 C係数 及び Qn値 （ 各応力要素用：σR, σZ, σθ, τRZ ）

	一次側温度		二次側温度		σR		σZ		σθ		τRZ
0	550	0	520	0	−0.427055142	0	−0.505612352	0	−0.606338526	0	0.001177125
0.11	550.0000762	0.11	520.4357132	0.11	−0.427055142	0.11	−0.505612352	0.11	−0.606338526	0.11	0.001177125
0.2	550.0004017	0.2	520.4376248	0.2	−0.794478869	0.2	−0.971086188	0.2	−1.211497959	0.2	0.003181206
0.3	550.0004017	0.3	520.4394629	0.3	−0.737290139	0.3	−0.945989307	0.3	−1.220539081	0.3	0.004413619
0.4	550.0011282	0.4	520.441647	0.4	−0.690625187	0.4	−0.924088101	0.4	−1.221749293	0.4	0.005554835
0.5	550.0011282	0.5	520.4443676	0.5	−0.673911425	0.5	−0.928795387	0.5	−1.246761844	0.5	0.006676459
0.6	550.0022946	0.6	520.4466947	0.6	−0.666528578	0.6	−0.938347395	0.6	−1.272151736	0.6	0.00776533
0.7	550.0022946	0.7	520.4474733	0.7	−0.658378112	0.7	−0.942821981	0.7	−1.287806459	0.7	0.008831843
0.8	550.003852	0.8	520.4456684	0.8	−0.646031435	0.8	−0.939443835	0.8	−1.291383261	0.8	0.009866231
0.9	550.003852	0.9	520.4404657	0.9	−0.628465172	0.9	−0.927642505	0.9	−1.282763991	0.9	0.010888709
1	550.0054863	1	520.4312695	1	−0.605368285	1	−0.907377076	1	−1.262086246	1	0.011862909
2	549.9919559	2	520.077256	2	−0.029293483	2	−0.148630083	2	−0.247868185	2	0.010134425
3.01	549.8613249	3.01	519.2683055	3.01	1.08810211	3.01	1.463302687	3.01	1.980524264	3.01	−0.047118104
4.01	549.3966085	4.01	518.0867885	4.01	2.553518168	4.01	3.655229237	4.01	5.065013666	4.01	−0.263188066
5.01	548.5312469	5.01	516.6104032	5.01	4.164405754	5.01	6.118797106	5.01	8.59656332	5.01	−0.748448426
6.01	547.2865496	6.01	514.8987388	6.01	5.784510771	6.01	8.63182764	6.01	12.27026289	6.01	−1.573881848
7.01	545.718189	7.01	513.0048319	7.01	7.323586072	7.01	11.04657612	7.01	15.87686721	7.01	−2.766613719
8.01	543.9045451	8.01	510.9793621	8.01	8.72805498	8.01	13.275292	8.01	19.28320508	8.01	−4.314730909
9.01	541.9201818	9.01	508.8695965	9.01	9.972406467	9.01	15.27521678	9.01	22.41379712	9.01	−6.179533537
10.01	539.8266874	10.01	506.7191237	10.01	11.04715036	10.01	17.02910545	10.01	25.22728198	10.01	−8.309088877
11.01	537.6878151	11.01	504.565514	11.01	11.95751821	11.01	18.54249224	11.01	27.71380507	11.01	−10.64334708
12.01	535.5908434	12.01	502.4459349	12.01	12.71856103	12.01	19.83852398	12.01	29.88734396	12.01	−13.10932113
13.01	533.5080438	13.01	500.3942104	13.01	13.32846739	13.01	20.91215579	13.01	31.72911208	13.01	−15.65825493
14.01	531.4037738	14.01	498.4400678	14.01	13.76452967	14.01	21.72296039	14.01	33.17990557	14.01	−18.27138741
16.01	527.2450397	16.01	494.923869	16.01	13.91039885	16.01	22.33460781	16.01	34.70306975	16.01	−23.68704052
17.01	525.240996	17.01	493.4037915	17.01	13.8348449	17.01	22.24883929	17.01	34.73411535	17.01	−26.32110151
18	523.3016663	18	492.0613576	18	13.52301496	18	21.85894227	18	34.37640099	18	−28.93241215
19	521.4317202	19	490.9161739	19	12.94114542	19	21.06249162	19	33.43827818	19	−31.52821685
20	519.6344822	20	489.9763985	20	12.14518373	20	19.91709751	20	31.98267596	20	−34.0631674
21	517.9114197	21	489.2560374	21	11.12214203	21	18.40183358	21	29.98438048	21	−36.52657525
22	516.2631036	22	488.7711978	22	9.854182407	22	16.48746861	22	27.40603997	22	−38.90961004
23	514.6892813	23	488.5266542	23	8.340075847	23	14.16713584	23	24.24044746	23	−41.2055952
24	513.1889752	24	488.5267544	24	6.576080889	24	11.4324235	24	20.47967209	24	−43.40967512
25	511.7606693	25	488.7742062	25	4.182140393	25	6.540251168	25	15.6740619	25	4.650842437
26	510.4024831	26	489.2694023	26	1.879419604	26	2.890256235	26	10.69443698	26	2.305840893
27	509.1122892	27	490.0098199	27	−0.667789155	27	−1.166259707	27	5.133987419	27	−0.321485297
28	507.8877896	28	490.989342	28	−3.452010503	28	−5.619547864	28	−0.990300336	28	−3.226826891
29	506.7265713	29	492.197474	29	−6.460515622	29	−10.4515854	29	−7.649937931	29	−6.400582065
30	505.6256169	30	493.6187528	30	−9.674610915	30	−15.63487131	30	−14.80322949	30	−9.826988029
31	504.5699996	31	495.2220396	31	−13.05677324	31	−21.11521413	31	−22.37112202	31	−13.47611254
32	503.5430156	32	496.81968	32	−16.3490464	32	−26.53035328	32	−29.85391876	32	−17.16160938
33	502.5385702	33	498.2997703	33	−19.42835383	33	−31.67150743	33	−36.944195	33	−20.73313773
34	501.5552219	34	499.6231131	34	−22.26741311	34	−36.4669758	34	−43.52894447	34	−24.11429025
35	500.5934932	35	500.7763149	35	−24.86197614	35	−40.89090876	35	−49.56831023	35	−27.2685006
36	499.654909	36	501.7599745	36	−27.22041077	36	−44.94339562	36	−55.06315079	36	−30.18233307
37	498.7415402	37	502.582656	37	−29.357647	37	−48.63901978	37	−60.03691685	37	−32.85616736
38	497.8556034	38	503.2570329	38	−31.29106004	38	−51.9993158	38	−64.52395001	38	−35.29825902
39	496.9991023	39	503.7979076	39	−33.03885478	39	−55.04929272	39	−68.56381853	39	−37.52147699
41	495.3788928	41	504.5357462	41	−36.05105167	41	−60.3468213	41	−75.50707093	41	−41.41028748
42	494.6136924	42	504.7382797	42	−37.30209393	42	−62.54439602	42	−78.33423502	42	−43.01445216
43	494.0216169	43	504.8879604	43	−38.4436613	43	−64.54595787	43	−80.90271194	43	−44.47164412
44.01	493.1679646	44.01	504.9718756	44.01	−39.50799998	44.01	−66.42005821	44.01	−83.28570266	44.01	−45.83814394
45.01	492.4908598	45.01	504.9799738	45.01	−40.45241993	45.01	−68.09165178	45.01	−85.39256683	45.01	−47.06576643
46.01	491.8463816	46.01	504.9334631	46.01	−41.29867684	46.01	−69.58967932	46.01	−87.26362079	46.01	−48.16454375
47.01	491.2334524	47.01	504.839222	47.01	−42.05459926	47.01	−70.92919066	47.01	−88.92148913	47.01	−49.14729985
48.01	490.6493	48.01	504.7032623	48.01	−42.72773037	48.01	−72.1234811	48.01	−90.38539087	48.01	−50.02408479
49.01	490.0906103	49.01	504.5308285	49.01	−43.32582828	49.01	−73.18599524	49.01	−91.67417412	49.01	−50.8047767
50.01	489.5542231	50.01	504.3264909	50.01	−44.55555463	50.01	−75.14784168	50.01	−93.13205932	50.01	−52.22857534
52.01	488.5373769	52.01	503.8375251	52.01	−45.47003314	52.01	−76.74280267	52.01	−94.99589969	52.01	−53.38129514
54.01	487.5792495	54.01	503.2624719	54.01	−46.17879915	54.01	−77.9949125	54.01	−96.42108674	54.01	−54.30320513
56.01	486.6630314	56.01	502.6212067	56.01	−46.74166547	56.01	−78.98919255	56.01	−97.50769249	56.01	−55.03821969
58.01	485.7739798	58.01	501.9288952	58.01	−47.19523037	58.01	−79.78848329	58.01	−98.33789339	58.01	−55.63093786
60.01	484.8993148	60.01	501.1971038	60.01	−47.56941958	60.01	−80.44478489	60.01	−98.97968231	60.01	−56.11861453
62.01	484.0281083	62.01	500.4346633	62.01	−47.88854856	62.01	−81.00054739	62.01	−99.48847199	62.01	−56.53170458
64.01	483.1510841	64.01	499.6482642	64.01	−48.17227372	64.01	−81.49023328	64.01	−99.90919302	64.01	−56.89496561
66.01	482.2603534	66.01	498.8429885	66.01	−48.4365305	66.01	−81.94186826	66.01	−100.2783659	66.01	−57.22853778
68.01	481.3491692	68.01	498.022642	68.01	−48.69420748	68.01	−82.37819817	68.01	−100.6256235	68.01	−57.54878494
70.01	480.4116554	70.01	497.1899632	70.01	−48.95567622	70.01	−82.81760631	70.01	−100.9748928	70.01	−57.86896371
72.01	479.4424961	72.01	496.3469171	72.01	−49.22946435	72.01	−83.27522549	72.01	−101.345841	72.01	−58.19998151

表 2.5.2　構造計算部品の出力（4 応力要素）

```
    Public Sub CalcStress(ByVal aStruct_Ctrl As Struct_Control, _
                          ByVal aStruct_Temp1 As Struct_TempModule, _
                          ByVal aStruct_Temp2 As Struct_TempModule, _
                          ByVal aStruct_C1 As Struct_CDataModule, _
                          ByVal aStruct_C2 As Struct_CDataModule, _
                          ByRef aStruct_StressInit As Struct_StressModule, _
                          ByRef aStruct_StressElm1 As Struct_StressModule, _
                          ByRef aStruct_StressElm2 As Struct_StressModule, _
                          ByRef aStruct_StressCase1 As Struct_StressModule, _
                          ByRef aStruct_StressCase2 As Struct_StressModule, _
                          ByRef aStruct_StressFin As Struct_StressModule, _
                          ByVal aClass_応力計算 As Class_応力計算)
        On Error GoTo ErrHandler_CalcStress
        '-- SCOPDモジュール数
        Dim i As Integer
        For i = 0 To (aStruct_Ctrl.ScopdModuleNo - 1)
            Dim j As Integer
            '-- 応力値計算（個別要素）1/2
            Call aClass_応力計算.CalcStressElm(aStruct_Ctrl, aStruct_Temp1,
                                aStruct_C1, aStruct_StressElm1)
            '-- 応力値計算（個別要素）2/2
            Call aClass_応力計算.CalcStressElm(aStruct_Ctrl, aStruct_Temp2,
                                aStruct_C2, aStruct_StressElm2)
            '-- 応力値合成（Case間）2/3 ： 1次側温度変化 応力値
            Call aClass_応力計算.CalcStressCase(aStruct_Ctrl, aStruct_Temp1,
                                aStruct_StressElm1, aStruct_StressCase1)
            '-- 応力値合成（Case間）3/3 ： 2次側温度変化 応力値
            Call aClass_応力計算.CalcStressCase(aStruct_Ctrl, aStruct_Temp2,
                                aStruct_StressElm2, aStruct_StressCase2)
            '-- 応力値合成（最終）
            Call aClass_応力計算.CalcStressFin(aStruct_Ctrl, aStruct_Temp1,
                                aStruct_StressCase1, aStruct_StressCase2,
                                aStruct_StressInit, aStruct_StressFin)
        Next i
        Exit Sub
ErrHandler_CalcStress:
    End Sub
```

図 2.5.2　構造計算部品プログラム（VisualBasic.NET）

2.6 試計算

今回開発したVB.NET版PARTSシステムによる試計算を実施した。解析仕様については、以下の通りである。

2.6.1 動特性解析

1) 解析対象

図2.6.1.1.1プラント全体図に示すような2ループのFBRナトリウムループ型炉の解析を行う。

2) 解析事象

代表事象として、(a) 手動トリップ (b) 外部電源喪失 の解析を行う。

3) 解析条件

手動トリップの解析条件を図2.6.1.2.1 に、外部電源喪失の解析条件を図2.6.1.2.2に示す。

2.6.1 動特性解析

2.6.1.1 解析対象

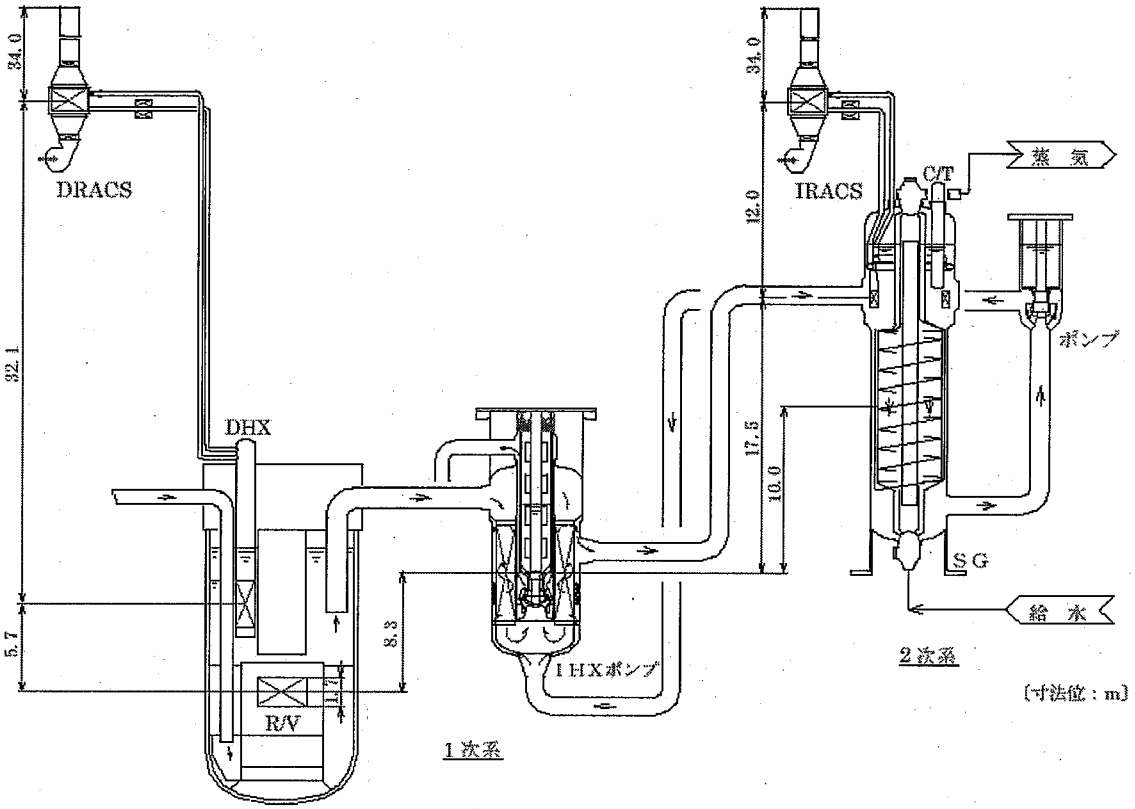


図 2.6.1.1.1　プラント系統構成

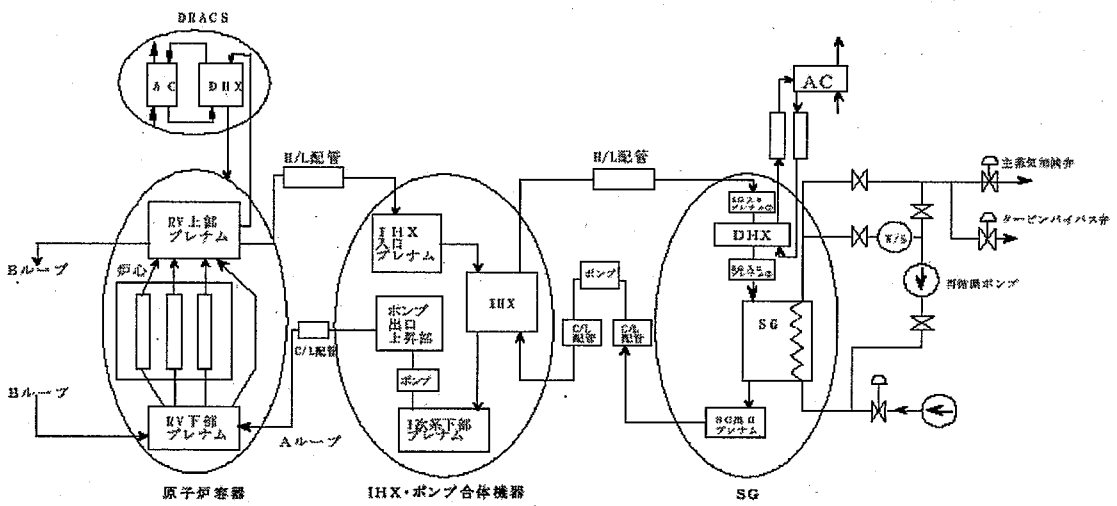


図 2.6.1.1.2　解析用動特性モデル図

2.6.1.2 解析事象

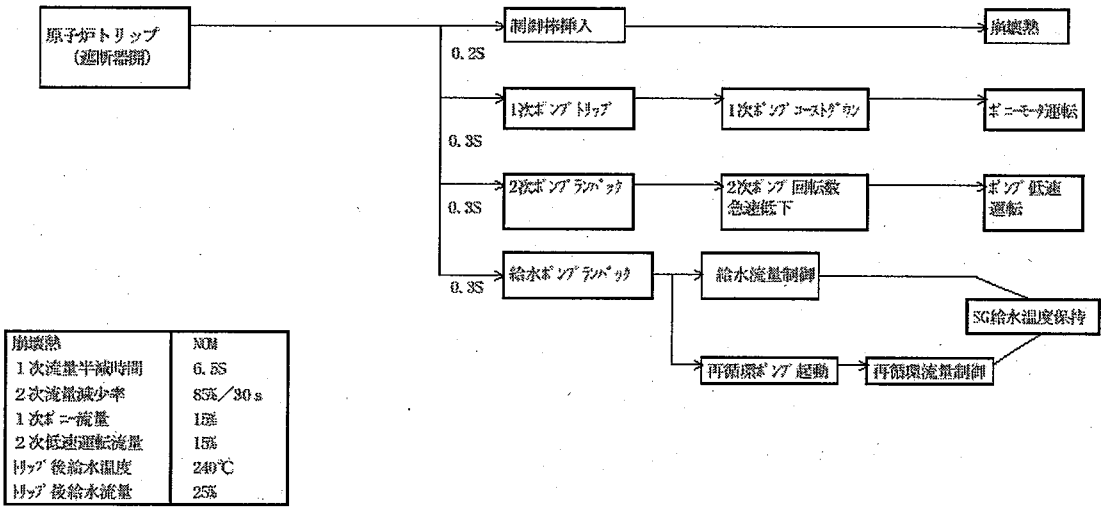


図 2.6.1.2.1　手動トリップシーケンス図

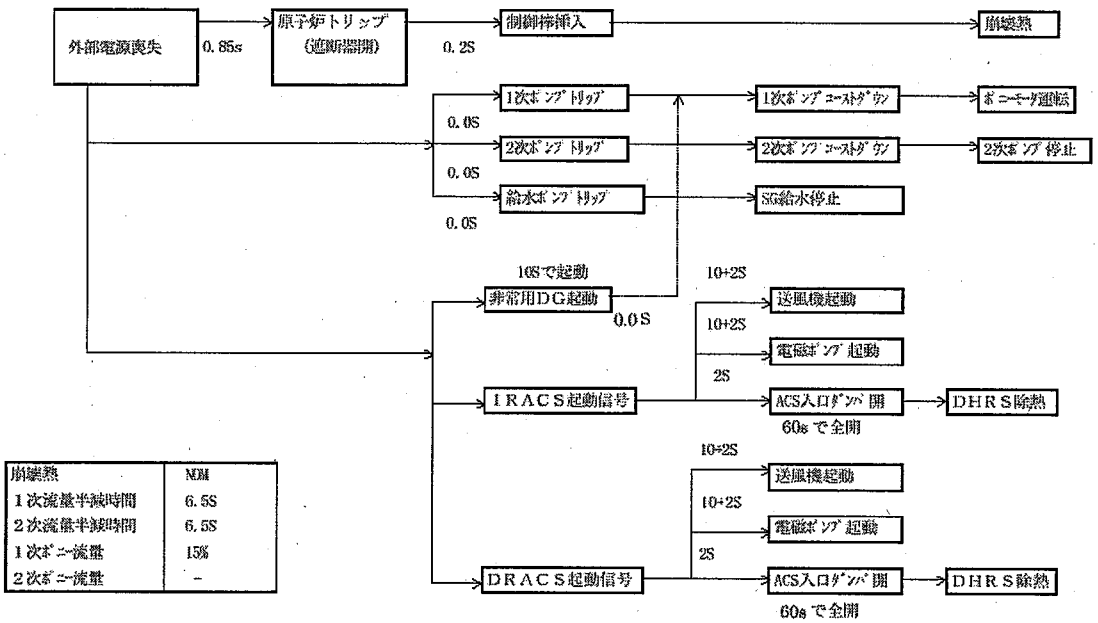


図 2.6.1.2.2　電源喪失シーケンス図

2.6.2 構造計算

(1) 解析対象

FBRナトリウムループ型炉の中間熱交換機の内部構造を検討対象とし特にIHX管板について計算を実施した。図 2.6.2.1.1 参照。

(2) 解析事象

熱過渡事象としては、構造健全性への影響が大きい「手動トリップ」を選定した。

(3) 熱過渡条件

IHX 管板に関する以下の条件

一次ナトリウム（入口側）の流量、温度

二次ナトリウム（出口側）の流量・温度

2.6.2.1 解析対象

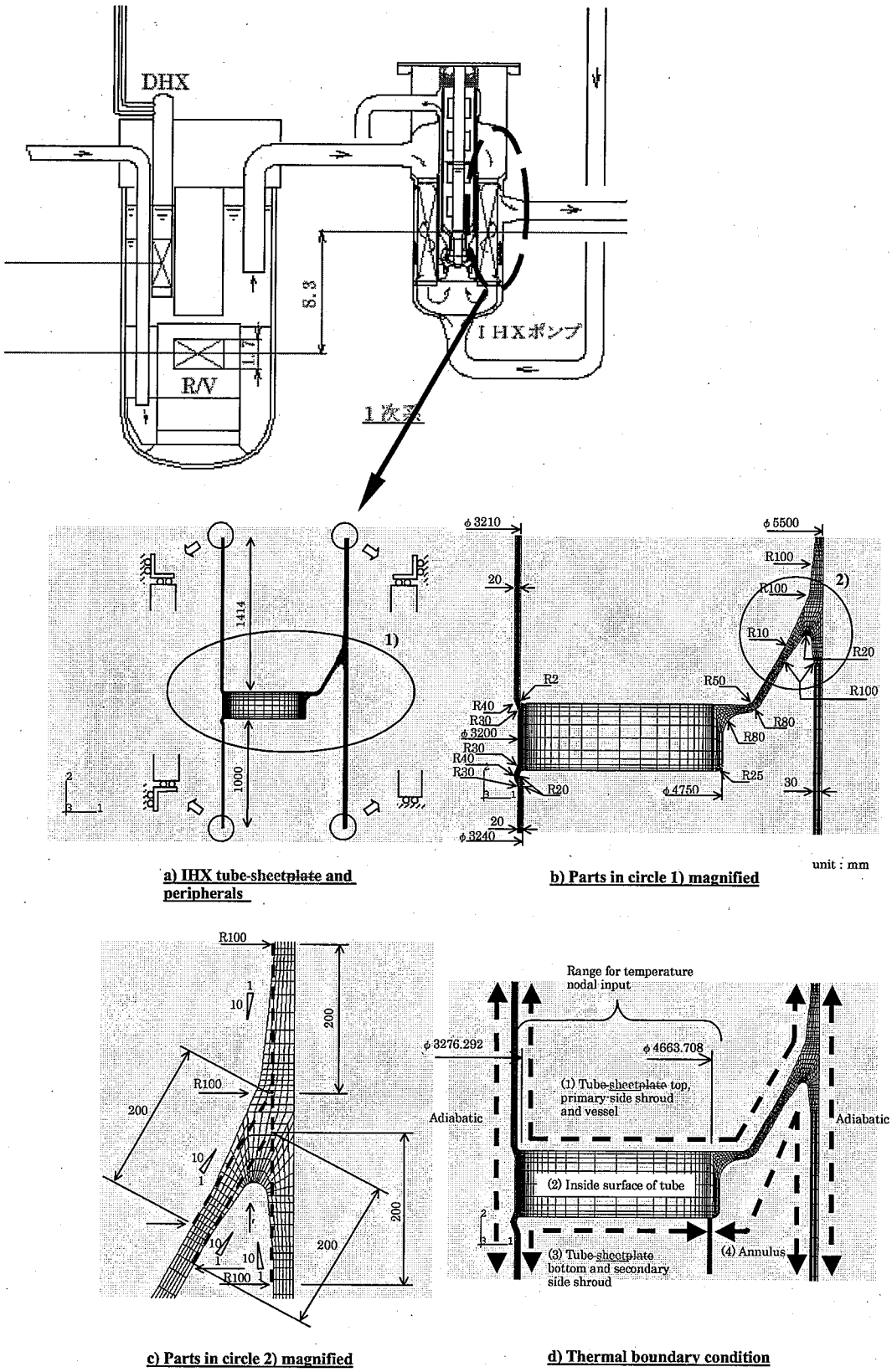


図 2.6.2.1.1　応力計算部品の解析対象部位（IHX 管板）

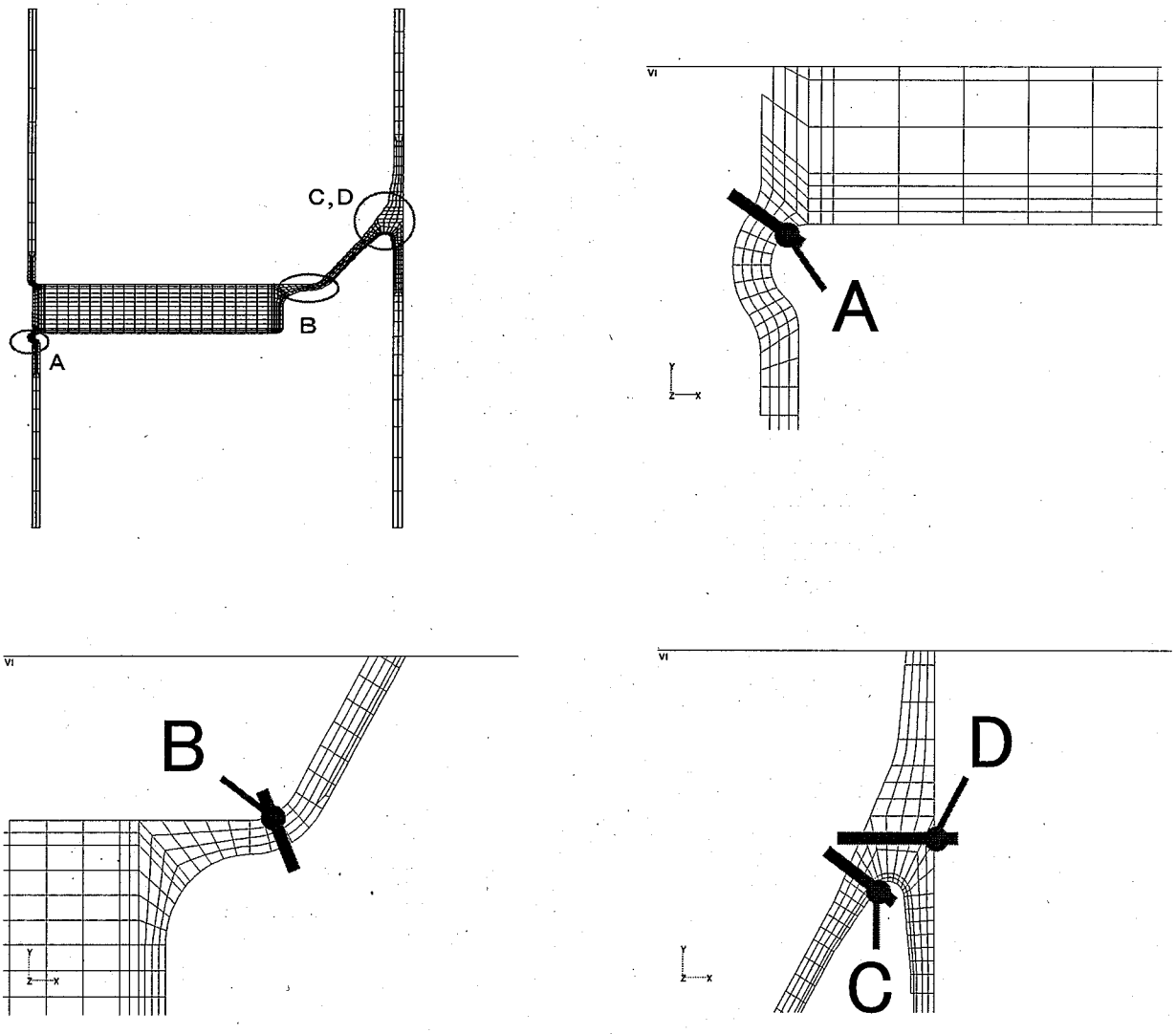


図 2.6.2.1.2　IHX 管板における評価点（4 つ）

今回のシステムで用いている中間熱交換機（IHX）管板に設定した評価点 A の応力成分（σR, σZ, σθ, τRZ）の C 係数及び Qn 値を表 2.7.2.3.1 ～ 表 2.7.2.3.4 に示す。

2.6.2.2 熱伝達率の切り替え

一次系冷却材の熱伝達率を 3 つ切り替えており、Case1～3 がそれぞれの熱伝達率に対応している。Case 4 ～ 6 も同様に二次系冷却材の熱伝達率がそれぞれに対応している。

Section / Time (sec)	Primary coolant side		Secondary coolant side	
	a) and b)	c)	d) and e)	f)
0 - 25	2.09	43.05	92.80	3.95
25 - 49.01	1.28	27.88	69.68	2.73
49.01 -			62.70	1.57

表 2.6.2.2.1　IHX の対象部位における熱伝達率の切り替え

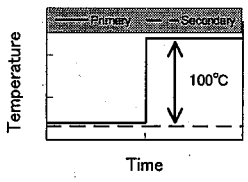
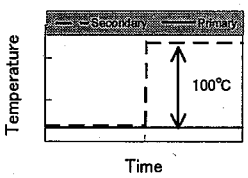
Case	Step increment of coolant temperature	Heat Transfer rate kW/m^2 ℃			
		Primary coolant side		Secondary coolant side	
		(1) Tubesheet top *	(2) Inside surface of tube *	(3) Tubesheet bottom *	(4) Annulus *
Case 1		2.09	43.05	92.80	3.95
Case 2		1.28	27.88	69.68	2.73
Case 3	Primary: +100℃ Secondary: Constant	1.28	27.88	62.70	1.57
Case 4		2.09	43.05	92.80	3.95
Case 5		1.28	27.88	69.68	2.73
Case 6	Primary: Constant Secondary: +100℃	1.28	27.88	62.70	1.57

表 2.6.2.2.2　熱伝達率と解析ケースの対応

2.6.2.3 IHX 評価点 A における応力成分の C 係数及び Qn 値

Case	I	C(i)	Qn(i)	Case	I	C(i)	Qn(i)
Case1	1	1.27E+02	0	Case4	1	-1.19E+02	0
	2	1.13E+01	0.001		2	-4.83E+01	0.001
	3	-6.12E+00	0.002		3	6.51E+01	0.002
	4	3.54E+00	0.005		4	-3.84E+01	0.005
	5	-4.31E+01	0.01		5	4.37E+01	0.01
	6	4.47E+00	0.02		6	-5.25E+01	0.02
	7	-7.59E+00	0.05		7	3.50E+01	0.05
	8	2.76E+01	0.1		8	-3.50E+01	0.1
	9	-3.50E+01	0.2		9	4.74E+01	0.2
	10	-8.21E+01	0.5		10	-7.44E+01	0.5
	11	1.32E+01	1		11	1.02E+02	1
	12	-3.42E+00	2		12	-1.82E+02	2
	13	-1.09E+01	5		13	2.69E+02	5
	14	5.03E+00	10		14	-6.14E+02	10
	15	-7.83E+00	20		15	9.47E+02	20
	16	4.09E+00	50		16	-3.45E+02	50
Case2	1	1.20E+02	0	Case5	1	-1.23E+02	0
	2	4.53E+01	0.001		2	-3.03E+01	0.001
	3	-5.07E+01	0.002		3	4.28E+01	0.002
	4	4.25E+01	0.005		4	-1.93E+01	0.005
	5	-6.06E+01	0.01		5	3.21E+01	0.01
	6	1.76E+00	0.02		6	-4.68E+01	0.02
	7	-1.46E+01	0.05		7	2.88E+01	0.05
	8	4.08E+01	0.1		8	-2.38E+01	0.1
	9	-5.96E+01	0.2		9	3.26E+01	0.2
	10	-7.25E+01	0.5		10	-5.13E+01	0.5
	11	2.10E+01	1		11	6.50E+01	1
	12	-8.32E+00	2		12	-1.27E+02	2
	13	-6.08E+00	5		13	1.77E+02	5
	14	2.54E+00	10		14	-4.24E+02	10
	15	-4.61E+00	20		15	7.74E+02	20
	16	2.88E+00	50		16	-3.07E+02	50
Case3	1	1.17E+02	0	Case6	1	-1.24E+02	0
	2	5.34E+01	0.001		2	-1.96E+01	0.001
	3	-6.08E+01	0.002		3	2.98E+01	0.002
	4	5.16E+01	0.005		4	-2.79E+00	0.005
	5	-6.52E+01	0.01		5	1.71E+01	0.01
	6	2.79E+00	0.02		6	-4.14E+01	0.02
	7	-1.44E+01	0.05		7	2.78E+01	0.05
	8	3.98E+01	0.1		8	-2.34E+01	0.1
	9	-5.89E+01	0.2		9	2.78E+01	0.2
	10	-7.29E+01	0.5		10	-4.31E+01	0.5
	11	2.13E+01	1		11	5.16E+01	1
	12	-8.41E+00	2		12	-1.08E+02	2
	13	-6.06E+00	5		13	1.44E+02	5
	14	2.53E+00	10		14	-3.51E+02	10
	15	-4.61E+00	20		15	7.02E+02	20
	16	2.88E+00	50		16	-2.87E+02	50

表 2.6.2.3.1 Green 関数 C 係数 及び Qn 値 (IHX 評価点 A - σR)

Case	I	C(i)	Qn(i)	Case	I	C(i)	Qn(i)
	1	2.11E+02	0		1	-2.00E+02	0
	2	1.60E+01	0.001		2	-7.63E+01	0.001
	3	-6.41E+00	0.002		3	1.03E+02	0.002
	4	3.10E+00	0.005		4	-6.05E+01	0.005
	5	-6.85E+01	0.01		5	7.14E+01	0.01
	6	7.00E+00	0.02		6	-8.86E+01	0.02
	7	-1.10E+01	0.05		7	5.20E+01	0.05
Case1	8	4.13E+01	0.1	Case4	8	-4.69E+01	0.1
	9	-5.67E+01	0.2		9	7.79E+01	0.2
	10	-1.36E+02	0.5		10	-1.01E+02	0.5
	11	2.12E+01	1		11	1.63E+02	1
	12	-5.20E+00	2		12	-2.24E+02	2
	13	-1.85E+01	5		13	3.74E+02	5
	14	8.55E+00	10		14	-7.40E+02	10
	15	-1.31E+01	20		15	1.10E+03	20
	16	6.83E+00	50		16	-4.07E+02	50
	1	2.01E+02	0		1	-2.05E+02	0
	2	7.24E+01	0.001		2	-4.81E+01	0.001
	3	-8.04E+01	0.002		3	6.84E+01	0.002
	4	6.80E+01	0.005		4	-3.02E+01	0.005
	5	-9.74E+01	0.01		5	5.35E+01	0.01
	6	1.97E+00	0.02		6	-8.02E+01	0.02
	7	-2.25E+01	0.05		7	4.29E+01	0.05
Case2	8	6.27E+01	0.1	Case5	8	-2.97E+01	0.1
	9	-9.72E+01	0.2		9	5.49E+01	0.2
	10	-1.20E+02	0.5		10	-6.58E+01	0.5
	11	3.43E+01	1		11	1.08E+02	1
	12	-1.34E+01	2		12	-1.47E+02	2
	13	-1.03E+01	5		13	2.49E+02	5
	14	4.37E+00	10		14	-4.99E+02	10
	15	-7.74E+00	20		15	8.84E+02	20
	16	4.81E+00	50		16	-3.55E+02	50
	1	1.95E+02	0		1	-2.06E+02	0
	2	8.64E+01	0.001		2	-3.13E+01	0.001
	3	-9.77E+01	0.002		3	4.81E+01	0.002
	4	8.37E+01	0.005		4	-3.64E+00	0.005
	5	-1.05E+02	0.01		5	2.92E+01	0.01
	6	3.80E+00	0.02		6	-7.20E+01	0.02
	7	-2.21E+01	0.05		7	4.19E+01	0.05
Case3	8	6.11E+01	0.1	Case6	8	-3.00E+01	0.1
	9	-9.60E+01	0.2		9	4.78E+01	0.2
	10	-1.21E+02	0.5		10	-5.36E+01	0.5
	11	3.47E+01	1		11	8.90E+01	1
	12	-1.36E+01	2		12	-1.20E+02	2
	13	-1.03E+01	5		13	2.05E+02	5
	14	4.36E+00	10		14	-4.10E+02	10
	15	-7.75E+00	20		15	7.95E+02	20
	16	4.81E+00	50		16	-3.30E+02	50

表 2.6.2.3.2 Green 関数 C 係数 及び Qn 値 (IHX 評価点 A - σZ)

Case	I	C(i)	Qn(i)	Case	I	C(i)	Qn(i)
Case1	1	2.64E+02	0	Case4	1	-2.47E+02	0
	2	2.95E+01	0.001		2	-1.13E+02	0.001
	3	-2.61E+01	0.002		3	1.51E+02	0.002
	4	1.85E+01	0.005		4	-9.98E+01	0.005
	5	-7.81E+01	0.01		5	1.01E+02	0.01
	6	8.12E+00	0.02		6	-1.12E+02	0.02
	7	-2.64E+01	0.05		7	3.74E+01	0.05
	8	3.93E+01	0.1		8	-5.07E+01	0.1
	9	-6.51E+01	0.2		9	8.06E+01	0.2
	10	-1.64E+02	0.5		10	-1.20E+02	0.5
	11	2.71E+01	1		11	2.13E+02	1
	12	-6.94E+00	2		12	-2.62E+02	2
	13	-2.16E+01	5		13	4.91E+02	5
	14	9.83E+00	10		14	-8.96E+02	10
	15	-1.55E+01	20		15	1.32E+03	20
	16	8.16E+00	50		16	-4.94E+02	50
Case2	1	2.53E+02	0	Case5	1	-2.54E+02	0
	2	8.55E+01	0.001		2	-7.98E+01	0.001
	3	-9.89E+01	0.002		3	1.09E+02	0.002
	4	7.96E+01	0.005		4	-6.50E+01	0.005
	5	-1.11E+02	0.01		5	7.62E+01	0.01
	6	1.18E+01	0.02		6	-9.73E+01	0.02
	7	-4.12E+01	0.05		7	2.65E+01	0.05
	8	6.48E+01	0.1		8	-2.97E+01	0.1
	9	-1.14E+02	0.2		9	5.41E+01	0.2
	10	-1.45E+02	0.5		10	-7.52E+01	0.5
	11	4.21E+01	1		11	1.44E+02	1
	12	-1.64E+01	2		12	-1.65E+02	2
	13	-1.21E+01	5		13	3.32E+02	5
	14	5.06E+00	10		14	-5.96E+02	10
	15	-9.19E+00	20		15	1.05E+03	20
	16	5.75E+00	50		16	-4.25E+02	50
Case3	1	2.49E+02	0	Case6	1	-2.56E+02	0
	2	9.75E+01	0.001		2	-6.26E+01	0.001
	3	-1.14E+02	0.002		3	8.77E+01	0.002
	4	9.26E+01	0.005		4	-4.03E+01	0.005
	5	-1.21E+02	0.01		5	5.17E+01	0.01
	6	1.61E+01	0.02		6	-8.48E+01	0.02
	7	-4.10E+01	0.05		7	2.48E+01	0.05
	8	6.31E+01	0.1		8	-2.80E+01	0.1
	9	-1.12E+02	0.2		9	4.61E+01	0.2
	10	-1.45E+02	0.5		10	-6.00E+01	0.5
	11	4.25E+01	1		11	1.20E+02	1
	12	-1.66E+01	2		12	-1.31E+02	2
	13	-1.21E+01	5		13	2.78E+02	5
	14	5.01E+00	10		14	-4.87E+02	10
	15	-9.17E+00	20		15	9.36E+02	20
	16	5.75E+00	50		16	-3.94E+02	50

表 2.6.2.3.3 Green 関数 C 係数 及び Qn 値（ IHX 評価点 A- $\sigma\theta$ ）

Case	I	C(i)	Qn(i)	Case	I	C(i)	Qn(i)
Case1	1	1.48E+02	0	Case4	1	3.89E+01	0
	2	1.28E+01	0.001		2	−1.23E+01	0.001
	3	−6.54E+00	0.002		3	1.33E+01	0.002
	4	3.66E+00	0.005		4	−1.69E+01	0.005
	5	−4.94E+01	0.01		5	5.65E+00	0.01
	6	5.61E+00	0.02		6	−1.57E+01	0.02
	7	−9.32E+00	0.05		7	−1.24E+00	0.05
	8	2.96E+01	0.1		8	−1.36E+01	0.1
	9	−3.84E+01	0.2		9	4.87E+00	0.2
	10	−9.63E+01	0.5		10	−1.32E+00	0.5
	11	1.56E+01	1		11	−1.36E+00	1
	12	−4.00E+00	2		12	3.93E−01	2
	13	−1.28E+01	5		13	−9.69E−01	5
	14	5.88E+00	10		14	1.33E+00	10
	15	−9.14E+00	20		15	−1.85E+00	20
	16	4.77E+00	50		16	7.89E−01	50
Case2	1	1.40E+02	0	Case5	1	−1.47E+02	0
	2	5.25E+01	0.001		2	−1.82E+01	0.001
	3	−5.85E+01	0.002		3	2.78E+01	0.002
	4	4.92E+01	0.005		4	−6.40E+00	0.005
	5	−6.98E+01	0.01		5	2.51E+01	0.01
	6	2.15E+00	0.02		6	−4.67E+01	0.02
	7	−1.75E+01	0.05		7	2.36E+01	0.05
	8	4.50E+01	0.1		8	−1.29E+01	0.1
	9	−6.71E+01	0.2		9	3.26E+01	0.2
	10	−8.52E+01	0.5		10	−1.38E+01	0.5
	11	2.47E+01	1		11	5.93E+01	1
	12	−9.74E+00	2		12	−2.12E+01	2
	13	−7.09E+00	5		13	1.05E+02	5
	14	2.97E+00	10		14	−1.19E+02	10
	15	−5.38E+00	20		15	2.00E+02	20
	16	3.36E+00	50		16	−8.84E+01	50
Case3	1	1.36E+02	0	Case6	1	−1.47E+02	0
	2	6.25E+01	0.001		2	−9.21E+00	0.001
	3	−7.10E+01	0.002		3	1.72E+01	0.002
	4	6.04E+01	0.005		4	9.61E+00	0.005
	5	−7.55E+01	0.01		5	1.01E+01	0.01
	6	3.57E+00	0.02		6	−4.22E+01	0.02
	7	−1.73E+01	0.05		7	2.43E+01	0.05
	8	4.39E+01	0.1		8	−1.51E+01	0.1
	9	−6.63E+01	0.2		9	2.99E+01	0.2
	10	−8.57E+01	0.5		10	−8.69E+00	0.5
	11	2.50E+01	1		11	5.20E+01	1
	12	−9.86E+00	2		12	−1.17E+01	2
	13	−7.06E+00	5		13	9.07E+01	5
	14	2.96E+00	10		14	−9.43E+01	10
	15	−5.38E+00	20		15	1.75E+02	20
	16	3.36E+00	50		16	−8.04E+01	50

表 2.6.2.3.4 Green 関数 C 係数 及び Qn 値 (IHX 評価点 A - τRZ)

2.7 PARTS.NET システムによる熱過渡応力解析値

応力解析対象としたアドバンス型炉の中間熱交換機内部構造に熱過渡を与える事象として手動トリップを想定したが、その時の中間熱交換機（IHX）評価点 A の各応力成分の Green 関数法による計算値を図 2.7.1～図 2.7.6 に示す。

更に IHX 一次系及び二次系冷却材の温度変化を図 2.7.7、図 2.7.8 に示す。

今回の PARTS.NET システムは、詳細設計時に使用することを想定している。

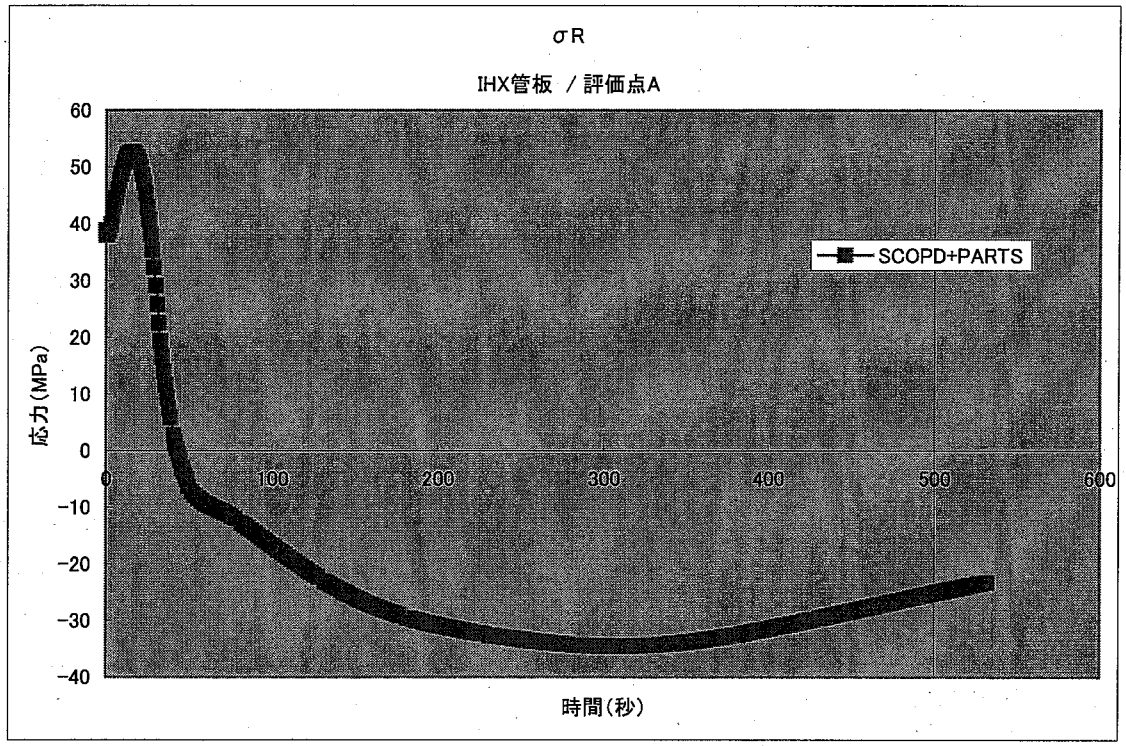


図 2.7.1　IHX 評価点 A における応力成分（σR）値 その 1

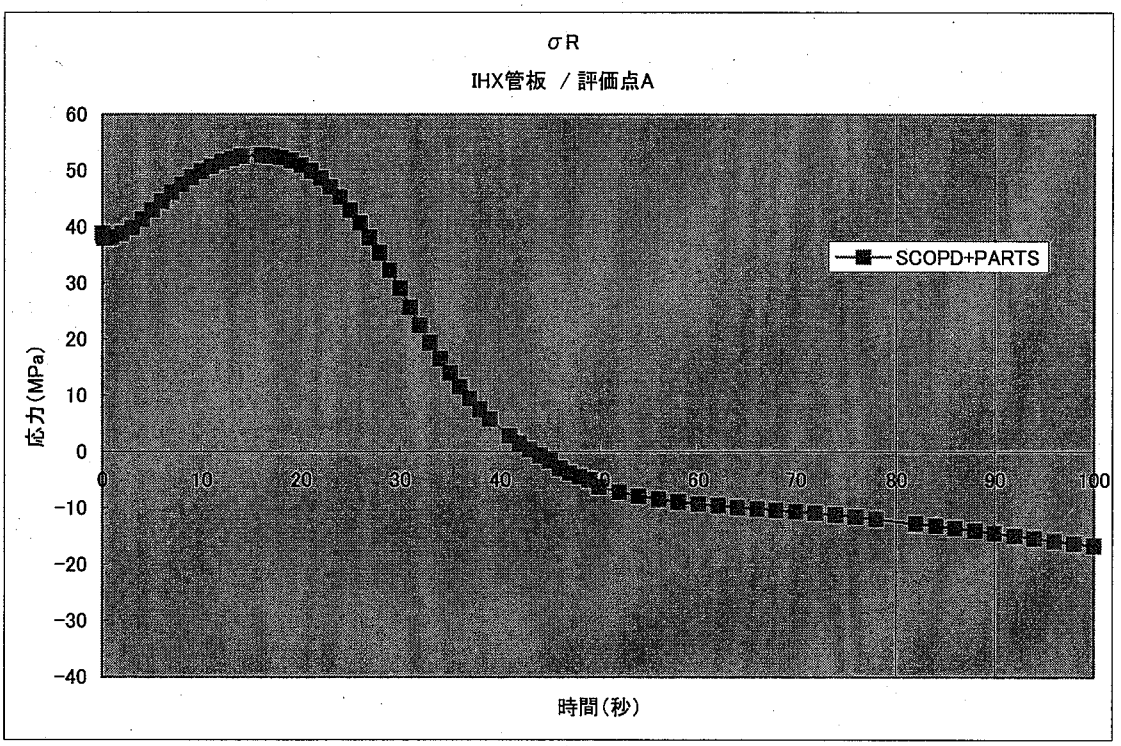


図 2.7.2　IHX 評価点 A における応力成分（σR）値 その 2

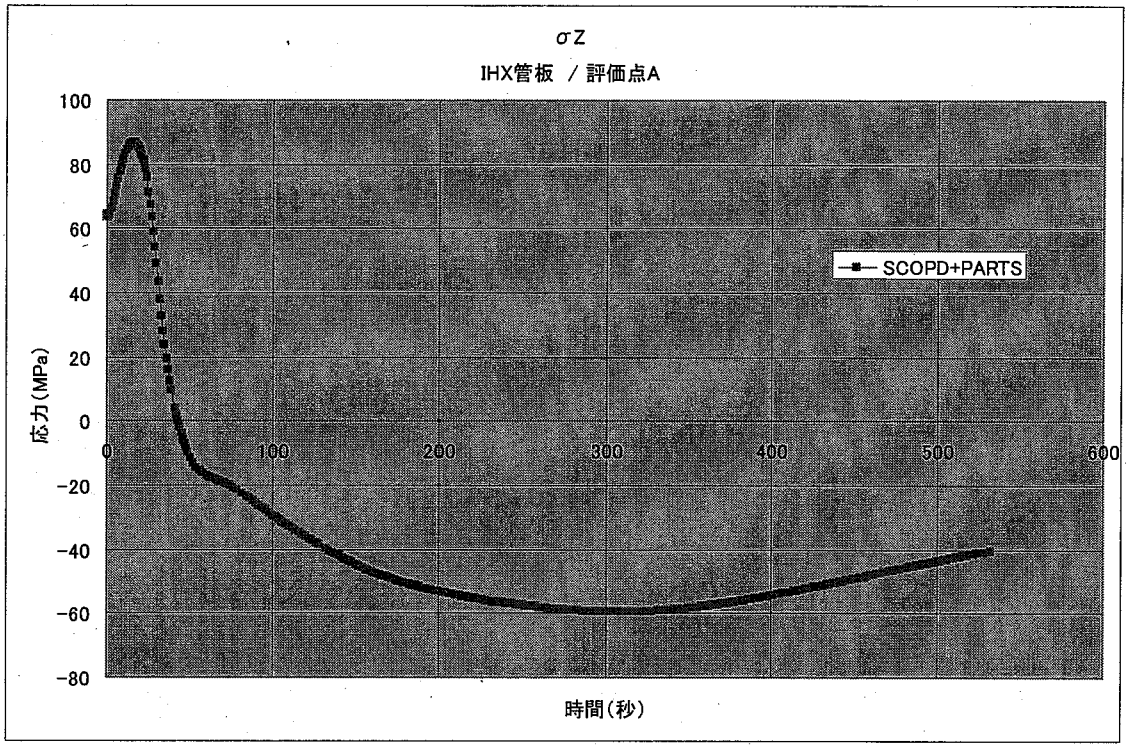


図 2.7.3 IHX 評価点 A における応力成分（σZ）値 その 1

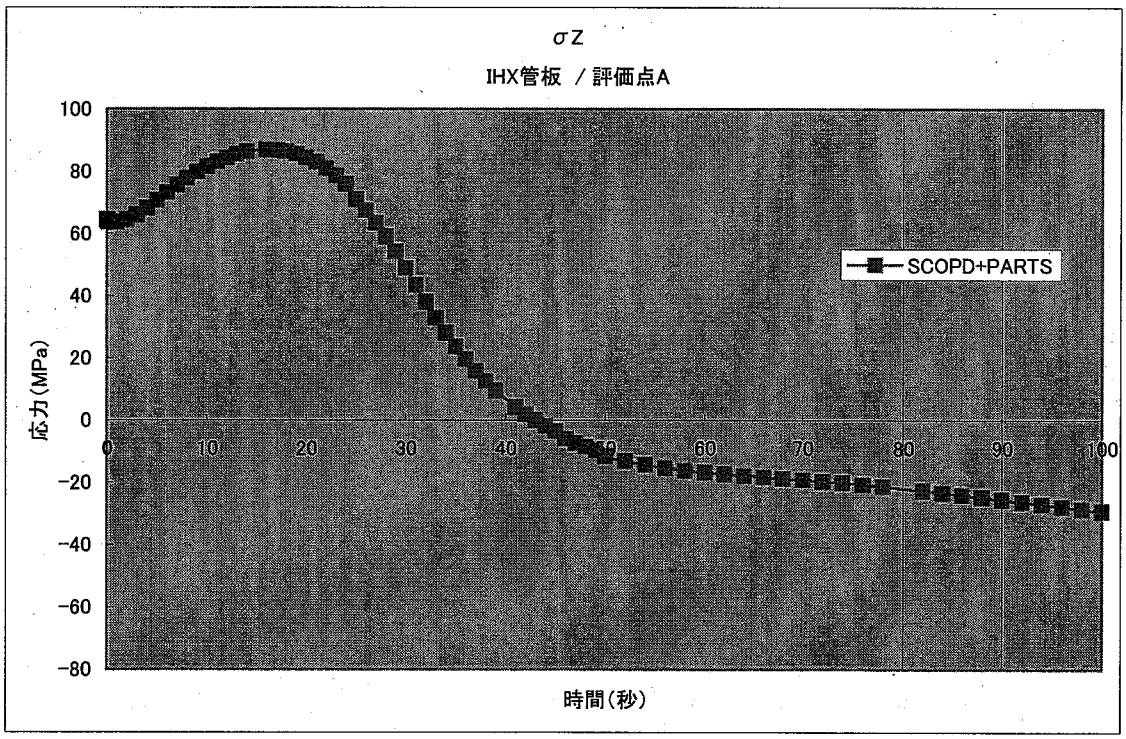


図 2.7.4 IHX 評価点 A における応力成分（σZ）値 その 2

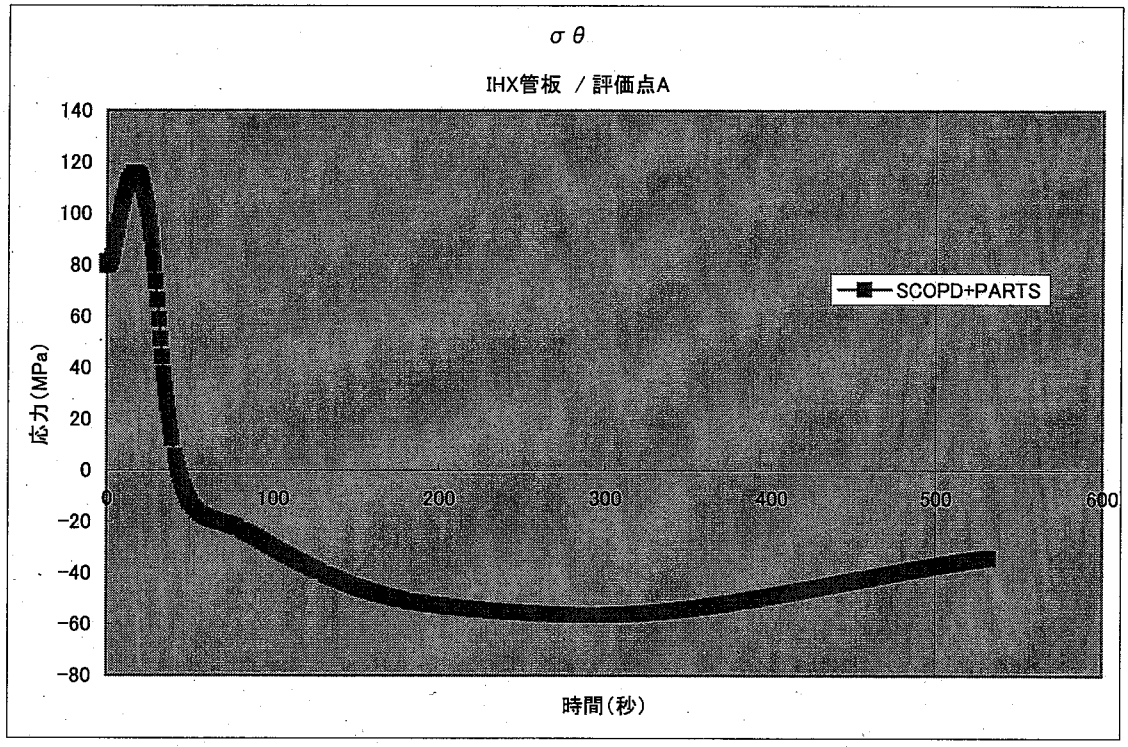


図 2.7.5 IHX 評価点 A における応力成分（σθ）値 その 1

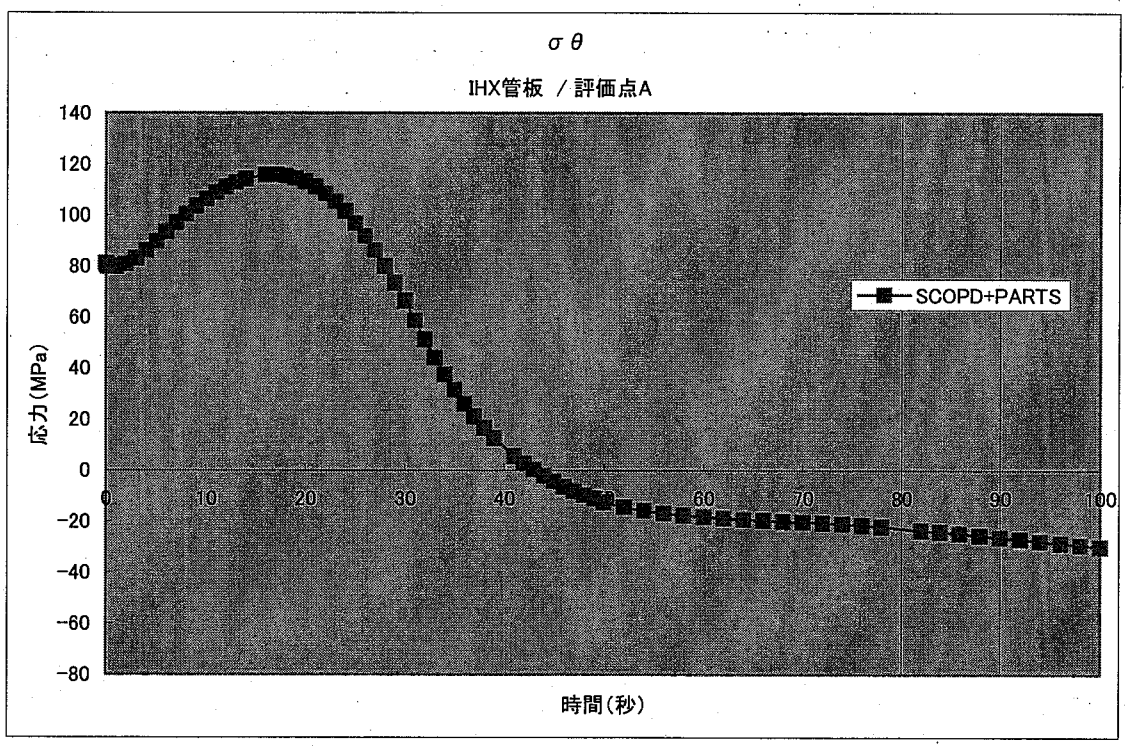


図 2.7.6 IHX 評価点 A における応力成分（σθ）値 その 2

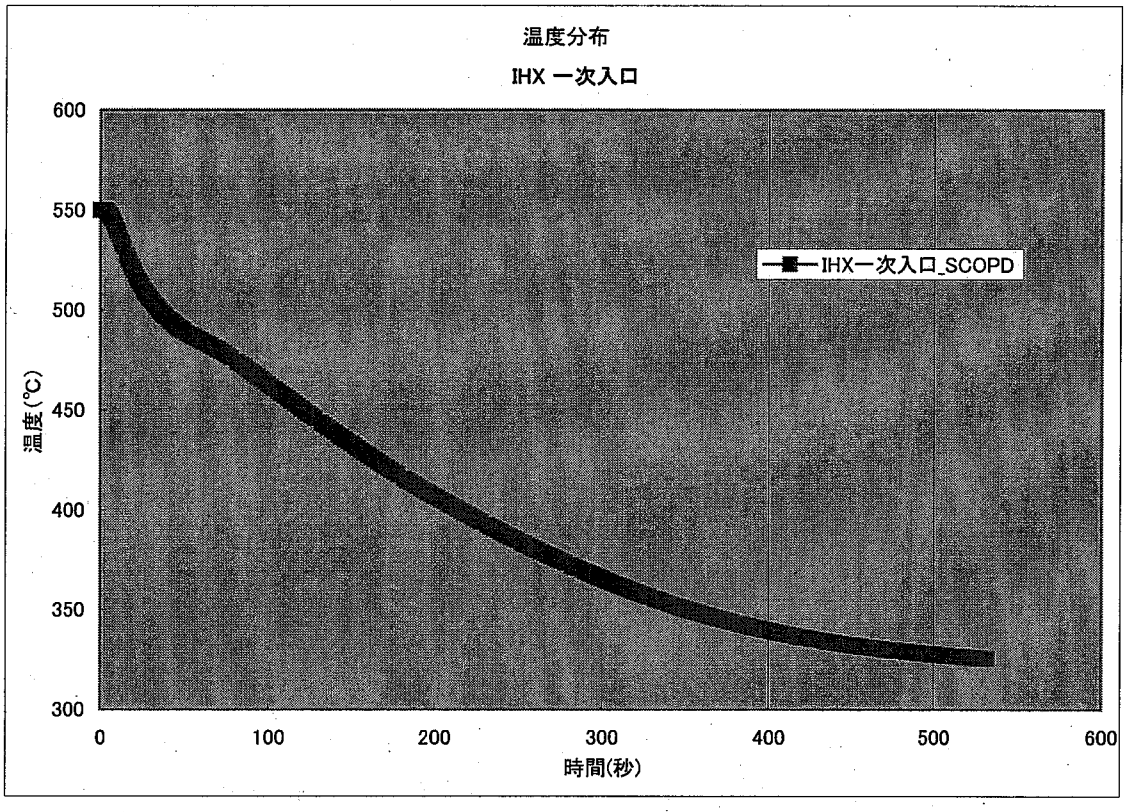


図 2.7.7　手動トリップ時における IHX 一次系冷却材の温度変化

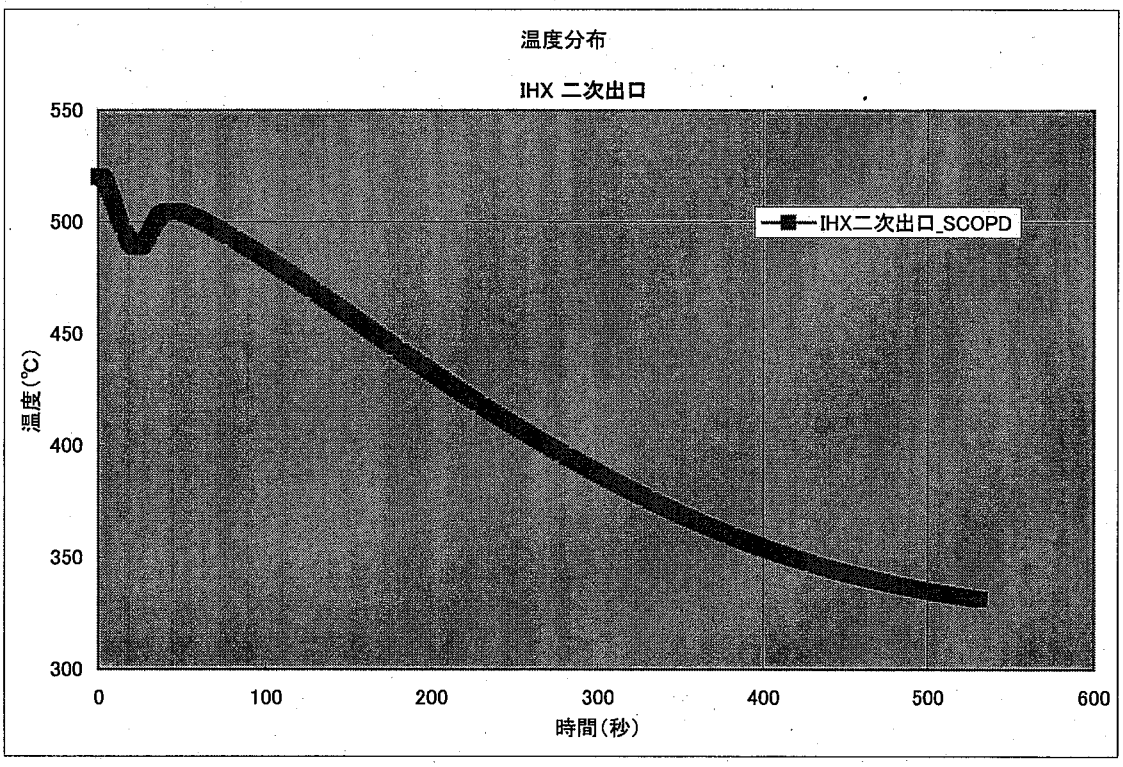


図 2.7.8　手動トリップ時における IHX 二次系冷却材の温度変化

第３章　Fortran90 によるプロトタイプ作成

第3章　Fortran90によるプロトタイプ作成

3.1　炉物理分野への適用検討

著者らは文献（3-1）で、Fortran90によるオブジェクト指向プログラミングの可能性について調査し、その結果を報告した。その主な内容は以下の通りである。

(1) Fortran90は言語分類上、オブジェクト指向言語ではないと称されるものの、Fortran77からの拡張機能であるMODULE文、USE文、TYPE文、INTERFACE文等を活用することで、クラス生成、継承、多様性（多態性、多相性）といったオブジェクト指向のキーとなる特徴を示すことが可能である。
(2) (1)により、オブジェクト指向言語の一つであるC++と同等の機能を持つプログラミングが原理的には可能である。
(3) Fortran77におけるCOMMON変数はそれを使用するすべての副プログラムに共通して明記する必要があるのに対し、Fortran90におけるクラスで使用するデータ構造は他のクラスとは独立した存在であるため、異なるクラス同士でのデータ構造の共有といったことに留意する必要は基本的にない。従って、プログラム構成要素の増加に伴う保守・管理作業に要する負荷は、Fortran90を利用することによりFortran77の利用に比べてより小さくなることが期待される。

そこで本報では、上記のようなFortran90が有するオブジェクト指向相当の特徴を炉物理分野に適用し、Fortran90を用いたプロトタイププログラムの概念設計を行うとともに、Fortran77と比較してプログラムの保守・管理の観点からその優位性について考察した。

3.2 概念設計

3.2.1 設計対象

一般に、炉心核設計や臨界実験解析等の核特性解析では、燃料ピンや燃料集合体等の 1 つの単位を格子系とみなし、その系を炉心の各領域（炉心燃料、ブランケット燃料、遮蔽体等）に適用して領域ごとに実効断面積を作成する。

実効断面積は中性子と炉心を構成する各核種（^{239}Pu、Fe 等）との反応確率を表すものに相当し、実効ミクロ断面積と実効マクロ断面積の 2 種類がある。前者は核種 1 個ごとの断面積（単位：barn=10^{-24}cm^2）を表し、後者はその核種の原子数密度分の断面積（単位：cm^{-1}）、すなわち実効ミクロ断面積と原子数密度の積で定義される。

通常、炉心の各領域に存在するどの核種をみても、それ自身が周囲の媒質と全く孤立して存在することはなく、複数の自分自身や他の核種に囲まれながら存在する。多くの核種の場合、それに入射してくる中性子のある特定のエネルギー範囲において非常に大きな確率で中性子と核反応を起こそうとするが（共鳴核反応）、核種数が複数にのぼるとこの共鳴核反応は核種間で互いに干渉し合うことになる。そのときの反応確率は、その核種が着目する系内で孤立している場合に比べ小さくなる。これを炉物理分野では自己遮蔽効果と称し、その効果の大きさを通常は自己遮蔽因子という炉物理量で評価する。この自己遮蔽因子に、孤立したと仮定した場合のミクロ断面積（無限希釈断面積という）を掛けたものが実効ミクロ断面積である。なお、自己遮蔽因子は着目核種の周囲に存在する核種の種類や存在量（これらを“バックグランド断面積”と定義される炉物理量で評価）と、その核種の温度との関数で表され、通常解析コード内では、自己遮蔽因子はバックグランド断面積と温度をパラメータとして数値テーブル化されている。これらのデータに対して内外挿補間することにより、解析対象におけるバックグランド断面積および温度に対応する自己遮蔽因子を求める。

こうして作成された実効ミクロ断面積を基に実効マクロ断面積が作成され、それが炉心全域にわたる中性子束分布等を求める計算に用いられる。

以上の背景に基づき、本報では高速炉核特性解析において実効ミクロおよびマクロ断面積を作成する作業を検討例に取り上げ、その作業に必要なプログラムの概念設計を行った。なお、問題を単純化するため、ここでは取り扱う格子系を均質体系（着目する領域内に存在する全核種がその領域内に均質に存在していると仮定）とし、実効断面積はミクロに限

定した。

3.2.2 設計方針

オブジェクト指向プログラミングでは、情報授受の主役を担うオブジェクトの生成元であるクラスをどのように定義するかが重要である。

実効断面積の作成過程において現れる主な物理量、パラメータを挙げると以下のようになる。

- 中性子エネルギー群数
- 核種 ID（番号または記号）
- 原子数密度
- 温度
- 無限希釈断面積
- バックグラウンド断面積
- 自己遮蔽因子
- 実効ミクロ断面積
- 実効マクロ断面積
- 核分裂スペクトル
- 反応の種類（捕獲、核分裂、生成、輸送、弾性散乱、非弾性散乱、(n,2n) 反応、(n, α) 反応、等）

さらに、上記の自己遮蔽因子や実効断面積等を求めるため、データの初期化や算出機能も必要となる。

以上を踏まえ、以下の方針でプログラム設計を行うこととした。

(1) 上記の物理量は基本的にある各種が有する属性であることから、ここでは“核種”をクラスとして定義し、これを基に計算対象とする具体的な核種（^{239}Pu、Na 等）をオブジェクトとして派生することとした。

(2) 上記(1)の“核種”クラスのデータ構造には以下のようなデータを持たせることと

した。

- 中性子エネルギー群数
- 核種 ID 番号
- 原子数密度
- 温度
- 無限希釈断面積（捕獲、核分裂、生成、輸送、散乱）*
- バックグラウンド断面積
- 自己遮蔽因子（捕獲、核分裂、生成、輸送、散乱）
- 実効ミクロ断面積（捕獲、核分裂、生成、輸送、散乱）
- 核分裂スペクトル

(3) 上記（1）の“核種”クラスのメソッドは以下に示す通りである。

- 核種 ID 番号、原子数密度、温度の各データの読み込み
- 無限希釈断面積の初期化
- バックグラウンド断面積の初期化
- 自己遮蔽因子の初期化
- 自己遮蔽因子の計算
- 実効ミクロ断面積の計算

以上の設計方針に基づき、作成したプログラムを図 3.1、3.2 に示す。図 3.1 はクラス“核種”を示したもので、プログラム中では MODULE 文で定義されている NUCLIDE_CLASS という名がクラス名になる。このクラスは、TYPE 文で宣言された NUCLIDE という名のデータ構造を有する。図 3.2 はメインプログラムで、中性子エネルギー群数、核種総数をパラメータで与え、クラス NUCLIDE_CLASS から必要な数だけ（図 3.2 ではデフォルト値として 15 個）オブジェクトを派生させる。これらオブジェクトに属性を与えるために、クラス NUCLIDE_CLASS で定義したメソッドを適宜 Call する。なお、オブジェクトは ALLOCATE 文で割り当てることとし、実行終了の際は DEALLOCATE で解放することとしている。

---

* 実際には（n,α）反応や（n,2n）反応など、さらに多くの反応の種類が存在するが、ここでは簡単のために 5 種類の反応に限定した。

3.3 Fortran77との比較

3.2節で作成したプログラムがFortran77に基づくプログラムに比べて保守・管理に要する作業負荷を軽減できる可能性について検討した。

図3.1、3.2のFortran90を用いたプログラムでは、1つのクラス内で複数のメソッド、すなわちFortran77のプログラムにおいても用いられる副プログラムがすべて定義されているため、一度そのクラス内で変数をデータ構造の中に取り込んでおけば、メソッド（副プログラム）ごとにそれら変数を改めて定義する必要がなくなっていることがわかる。

このような特徴は、オブジェクト指向の重要な概念の一つである「継承」にみられる、スーパークラスとサブクラスの間におけるデータ構造の定義において一層顕著に現れる。図3.3～図3.8は、図3.1に示したクラスNUCLIDE_CLASSクラスを、NUCLIDE_CLASSをスーパークラスに、その配下に設けるND_TEMP_CLASS、XINF_CLASS、XEFF_CLASS、SIG0_CLASS、FACTOR_CLASSの5つをサブクラスとして階層構造化したものである。図3.9にNUCLIDE_CLASSに関連するクラスのクラス図を示す。NUCLIDE_CLASSは2つのREAL変数からなるデータ構造NUCLIDEを有している。NUCLIDEは5つの各サブクラスに継承されるが、各サブクラスのTYPE文をみると、各サブクラスにおいてはNUCLIDEを構成する2つのREAL変数について型宣言がなされていないことがわかる。

一方、このような機能をFortran77で表現した場合を考えると、Fortran77では副プログラム間でのデータの共有化は、多くの場合CALL文を介するか、COMMON文で宣言するかによって行われる。CALL文を用いた場合、Callする側の副プログラムとCallされる側の副プログラムの両方に対して、それらが共通に用いる変数の型宣言を行わなければならない。すなわち、例えばCallする側の副プログラムで宣言している型をINTEGERからREALに変えたい場合、Callされる側の副プログラムについても同様の型変更を行う必要がある。また、このような作業負荷への影響は、複数の副プログラム間をCOMMON文で結合しているときはさらに大きくなり、同一のCOMMON文を用いている副プログラムの数だけ変数名や数を変更しなければならない。

以上のことから、プログラムを構成する各モジュール（クラスまたは副プログラム）において、例えばある一箇所の変数の次元を変更する必要が生じた場合でも、Fortran77ではそれらを共通に用いる副プログラムすべてに修正をかけなければならないのに対し、Fortran90ではその変数が属するデータ構造を有するクラスのみの修正で済み、継承等でそ

のデータ構造を使用する他のクラスについては修正を要しない。

さらに、例えば ^{239}Pu の 3 群核分裂断面積というようなある変数を表記したい場合、Fortran77 では XEFF_FIS (4,3) というように変数名の表記には 1 つの配列名しか使用できないのに対し、Fortran90 では PU239%XEFF_FIS (3) というように構造体の概念を用いた表記が可能であるため、核種名と核分裂断面積という反応の種類との関係が Fortran77 に比べて容易に理解しやすい。これはプログラムを修正する際の作業負荷に差をもたらす要因となり、この点においても Fortran90 の方が優れているといえる。

3.4 Fortran90 適用性検討結果のまとめ

Fortran90 を簡単な炉物理計算問題に適用するためのプログラムを概念設計し、Fortran77 との比較検討を行なった。その結果、Fortran90 は Fortran77 に比べ、メソッド間で共通変数の冗長定義をせずに済むこと、構造体概念を用いた変数表記ができるため、変数の意味がそれを見ただけで容易に理解しやすい、などの利点を有し、プログラムの保守・管理作業に要する負荷を低減するのに有効であることがわかった。

今後、本プログラムに自己遮蔽因子を始めとする計算機能（メソッド）を記述してプログラムとしての動作・機能を確認することにより、Fortran90 のもつ特長をさらに具体的に示すことが期待される。

図 3.1 モジュール NUCLIDE_CLASS (FORTRAN90)

```
      MODULE NUCLIDE_CLASS
       IMPLICIT NONE
       TYPE NUCLIDE
        INTEGER :: NGROUP                        ! NUMBER OF ENERGY GROUPS
        INTEGER :: IDNUM                         ! ID NO. OF NUCLIDE
        REAL :: NDENS                            ! NUMBER DENSITY OF NUCLIDE
        REAL :: TEMP                             ! TEMPERATURE OF NUCLIDE
        REAL, POINTER, DIMENSION(:) :: XINF_CAP,  ! INFINITIVE CAPTURE X-SEC
     1                                 XINF_FIS,  ! INFINITIVE FISSION X-SEC
     2                                 XINF_PRO,  ! INFINITIVE PRODUCTION X-SEC
     3                                 XINF_TRA   ! INFINITIVE TRANSPORT X-SEC
        REAL, POINTER, DIMENSION(:,:) :: XINF_SCA ! INFINITIVE SCATTERING X-SEC
        REAL, POINTER, DIMENSION(:) :: SIGO       ! BACKGROUND X-SEC
        REAL, POINTER, DIMENSION(:) :: F_CAP,     ! F FACTOR FOR CAPTURE X-SEC
     1                                 F_FIS,     ! F FACTOR FOR FISSION X-SEC
     2                                 F_PRO,     ! F FACTOR FOR PRODUCTION X-SEC
     3                                   F_TRA    ! F FACTOR FOR TRANSPORT X-SEC
        REAL, POINTER, DIMENSION(:,:) :: F_SCA    ! F FACTOR FOR SCATTERING X-SEC
        REAL, POINTER, DIMENSION(:) :: XEFF_CAP,  ! EFFECTIVE CAPTURE X-SEC
     1                                 XEFF_FIS,  ! EFFECTIVE FISSION X-SEC
     2                                 XEFF_PRO,  ! EFFECTIVE PRODUCTION X-SEC
     3                                 XEFF_TRA   ! EFFECTIVE TRANSPORT X-SEC
        REAL, POINTER, DIMENSION(:,:) :: XEFF_SCA ! EFFECTIVE SCATTERING X-SEC
        REAL, POINTER, DIMENSION(:) :: CHAI       ! FISSION SPECTRUM
       END TYPE
C
       CONTAINS
C
       SUBROUTINE NUCLIDE_DATA(THIS, NG)
        TYPE(NUCLIDE), INTENT(INOUT) :: THIS
        INTEGER, INTENT(IN) :: NG
        INTEGER :: ID
        REAL :: ND, T
C
        WRITE(*,*) '<NUCLIDE_DATA>'
        THIS%NGROUP = NG
        OPEN(UNIT=5, FILE=DATA_NUC)
        READ(5,*) ID, ND, T
        THIS%IDNUM = ID
        THIS%NDENS = ND
        THIS%TEMP = T
       END SUBROUTINE
C
       SUBROUTINE INIT_XINF(THIS)
        TYPE(NUCLIDE), INTENT(INOUT) :: THIS
        INTEGER :: N, NN
C
        WRITE(*,*) '<INIT_XINF>'
        ALLOCATE (THIS%XINF_CAP(THIS%NGROUP),
     1            THIS%XINF_FIS(THIS%NGROUP),
     2            THIS%XINF_PRO(THIS%NGROUP),
     3            THIS%XINF_TRA(THIS%NGROUP),
     4            THIS%XINF_SCA(THIS%NGROUP, THIS%NGROUP))
        DO N = 1, THIS%NGROUP
         THIS%XINF_CAP(N) = 0.0
         THIS%XINF_FIS(N) = 0.0
         THIS%XINF_PRO(N) = 0.0
         THIS%XINF_TRA(N) = 0.0
         DO NN = 1, THIS%NGROUP
          THIS%XINF_SCA(NN, N) = 0.0
```

(次ページへ続く)

（前ページからの続き）

```
          END DO
         END DO
        END SUBROUTINE
C
        SUBROUTINE INIT_SIGO(THIS)
         TYPE(NUCLIDE), INTENT(INOUT) :: THIS
         INTEGER :: N
C
         WRITE(*,*) '<INIT_SIGO>'
         ALLOCATE (THIS%SIGO(THIS%NGROUP))
         DO N = 1,THIS%NGROUP
          THIS%SIGO(N) = 0.0
         END DO
        END SUBROUTINE
C
        SUBROUTINE INIT_F(THIS)
         TYPE(NUCLIDE), INTENT(INOUT) :: THIS
         INTEGER :: N,NN
C
         WRITE(*,*) '<INIT_F>'
         ALLOCATE (THIS%F_CAP(THIS%NGROUP),
     1             THIS%F_FIS(THIS%NGROUP),
     2             THIS%F_PRO(THIS%NGROUP),
     3             THIS%F_TRA(THIS%NGROUP),
     4             THIS%F_SCA(THIS%NGROUP,THIS%NGROUP))
         DO N = 1,THIS%NGROUP
          THIS%F_CAP(N) = 0.0
          THIS%F_FIS(N) = 0.0
          THIS%F_PRO(N) = 0.0
          THIS%F_TRA(N) = 0.0
          DO NN = 1,THIS%NGROUP
           THIS%F_SCA(NN,N) = 0.0
          END DO
         END DO
        END SUBROUTINE
C ***
C
        SUBROUTINE INIT_XEFF(THIS)
         TYPE(NUCLIDE), INTENT(INOUT) :: THIS
         INTEGER :: N,NN
C
         WRITE(*,*) '<INIT_XEFF>'
         ALLOCATE (THIS%XEFF_CAP(THIS%NGROUP),
     1             THIS%XEFF_FIS(THIS%NGROUP),
     2             THIS%XEFF_PRO(THIS%NGROUP),
     3             THIS%XEFF_TRA(THIS%NGROUP),
     4             THIS%XEFF_SCA(THIS%NGROUP,THIS%NGROUP))
         DO N = 1,THIS%NGROUP
          THIS%XEFF_CAP(N) = 0.0
          THIS%XEFF_FIS(N) = 0.0
          THIS%XEFF_PRO(N) = 0.0
          THIS%XEFF_TRA(N) = 0.0
          DO NN = 1,THIS%NGROUP
           THIS%XEFF_SCA(NN,N) = 0.0
          END DO
         END DO
        END SUBROUTINE
C
        SUBROUTINE INIT_CHAI(THIS)
         TYPE(NUCLIDE), INTENT(INOUT) :: THIS
         INTEGER :: N
C
         WRITE(*,*) '<INIT_CHAI>'
```

（次ページへ続く）

（前ページからの続き）

```
            ALLOCATE (THIS%CHAI(THIS%NGROUP))
            DO N = 1,THIS%NGROUP
             THIS%CHAI(N) = 0.0
            END DO
           END SUBROUTINE
C
           SUBROUTINE CALC_F(THIS)
C
C      自己遮蔽因子の計算機能
C
           END SUBROUTINE
C
           SUBROUTINE CALC_XEFF(THIS)
C
C      実効ミクロ断面積の作成機能
C
           END SUBROUTINE
C

           SUBROUTINE DELETE_XINF(THIS)
            TYPE(NUCLIDE), INTENT(INOUT) :: THIS
C
            WRITE(*,*) '<DELETE_XINF>'
            DEALLOCATE (THIS%XINF_CAP,
          1             THIS%XINF_FIS,
          2             THIS%XINF_PRO,
          3             THIS%XINF_TRA,
          4             THIS%XINF_SCA)
           END SUBROUTINE
C
           SUBROUTINE DELETE_SIGO(THIS)
            TYPE(NUCLIDE), INTENT(INOUT) :: THIS
C
            WRITE(*,*) '<DELETE_SIGO>'
            DEALLOCATE (THIS%SIGO)
           END SUBROUTINE
C
           SUBROUTINE DELETE_F(THIS)
            TYPE(NUCLIDE), INTENT(INOUT) :: THIS
C
            WRITE(*,*) '<DELETE_F>'
            DEALLOCATE (THIS%F_CAP,
          1             THIS%F_FIS,
          2             THIS%F_PRO,
          3             THIS%F_TRA,
          4             THIS%F_SCA)
           END SUBROUTINE
C ***
C
```

図 3.2 メインプログラム MAIN (FORTRAN90)

```
      PROGRAM MAIN
C
C     THIS SAMPLE PROGRAM IS TO CALCULATE EFFECTIVE MICROSCOPOC
C     CROSS SECTIONS OF VARIOUS NUCLIDES IN A HOMOGENEOUS MEDIUM.
C
      USE NUCLIDE_CLASS
C
      TYPE (NUCLIDE),ALLOCATABLE :: NUC(:)
      INTEGER, PARAMETER :: NG=70,NUM_NUCIDE=15
      INTEGER :: I
C
      ALLOCATE(NUC(NUM_NUCIDE),STAT=ERR)
C
      DO I =1,NUM_NUCLIDE
       CALL NUCLIDE_DATA(NUC(I),NG)
       CALL INIT_XINF(NUC(I))
       CALL INIT_SIGO(NUC(I))
       CALL INIT_F(NUC(I))
       CALL INIT_XEFF(NUC(I))
       CALL INIT_CHAI(NUC(I))
      END DO
C
      DO I = 1,NUM_NUCLIDE
       CALL CALC_F(NUC(I))
       CALL CALC_XEFF(NUC(I))
      END DO
C
      DO I = 1,NUM_NUCLIDE
       CALL DELETE_XINF(NUC(I))
       CALL DELETE_SIGO(NUC(I))
       CALL DELETE_F(NUC(I))
      END DO
C
      DEALLOCATE(NUC)
C
      END PROGRAM
```

図 3.3 スーパークラス NUCLIDE_CLASS (FORTRAN90)

```
      MODULE NUCLIDE_CLASS
       IMPLICIT NONE
       TYPE NUCLIDE
        CHARACTER :: NUCLIDE_NAME*6            ! NAME OF NUCLIDE
        INTEGER :: NUCLIDE_IDNUMBER            ! IDNUMBER OF NUCLIDE
       END TYPE
C
      CONTAINS
C
       SUBROUTINE NUCLIDE_INPUT(THIS)
        TYPE(NUCLIDE), INTENT(INOUT) :: THIS
        CHARACTER :: NAME*6
        INTEGER :: NO
C
        WRITE(*,*) '<NUCLIDE_DATA>'
        OPEN(UNIT=5,FILE=DATA_NUC)
        READ(5,*) NAME,NO
        THIS%NUCLIDE_NAME = NAME
        THIS%NUCLIDE_IDNUMBER = NO
       END SUBROUTINE
C
      END MODULE NUCLIDE_CLASS
```

図 3.4 サブクラス ND_TEMP_CLASS (FORTRAN90)

```
      MODULE ND_TEMP_CLASS
C
C *** BRING NUCLIDE_CLASS INTO SCOPE
C
       USE NUCLIDE_CLASS
C
       IMPLICIT NONE
       TYPE ND_TEMP
        REAL :: NUMBER_DENSITY             ! NUMBER DENSITY OF NUCLIDE
        REAL :: TEMPERATURE                ! TEMPERATURE OF NUCLIDE
       END TYPE
C
      CONTAINS
C
       SUBROUTINE ND_TEMP_INPUT(THIS)
        TYPE(ND_TEMP), INTENT(INOUT) :: THIS
        REAL :: ND
        REAL :: TEMP
C
        WRITE(*,*) '<NUMBER DENSITY & TEMPARATURE>'
        OPEN(UNIT=5,FILE=ND_TEMP)
        READ(5,*) ND,TEMP
        THIS%NUMBER_DENSITY = ND
        THIS%TEMPERATURE = TEMP
       END SUBROUTINE
C
      END MODULE ND_TEMP_CLASS
```

図 3.5 サブクラス XINF_CLASS (FORTRAN90)

```
      MODULE XINF_CLASS
C
C *** BRING NUCLIDE_CLASS INTO SCOPE
C
       USE NUCLIDE_CLASS
C
       IMPLICIT NONE
       TYPE XINF
        TYPE(NUCLIDE) NUC
        REAL,POINTER,DIMENSION(:) :: XINF_CAP,   ! INFINITIVE CAPTURE X-SEC
     1                               XINF_FIS,   ! INFINITIVE FISSION X-SEC
     2                               XINF_PRO,   ! INFINITIVE PRODUCTION X-SEC
     3                               XINF_TRA    ! INFINITIVE TRANSPORT X-SEC
        REAL,POINTER,DIMENSION(:,:) :: XINF_SCA ! INFINITIVE SCATTERING X-SEC
       END TYPE
C
      CONTAINS
C
       SUBROUTINE INIT_XINF(THIS,NGROUP)
        TYPE(XINF), INTENT(INOUT) :: THIS
        INTEGER, INTENT(IN) :: NGROUP
C
        WRITE(*,*) '<INIT_XINF>'
        ALLOCATE (THIS%XINF_CAP(THIS%NGROUP),
     1            THIS%XINF_FIS(THIS%NGROUP),
     2            THIS%XINF_PRO(THIS%NGROUP),
     3            THIS%XINF_TRA(THIS%NGROUP),
     4            THIS%XINF_SCA(THIS%NGROUP,THIS%NGROUP))
        THIS%XINF_CAP(1:THIS%NGROUP) = 0.0
        THIS%XINF_FIS(1:THIS%NGROUP) = 0.0
        THIS%XINF_PRO(1:THIS%NGROUP) = 0.0
        THIS%XINF_TRA(1:THIS%NGROUP) = 0.0
     THIS%XINF_SCA(1:THIS%NGROUP,1:THIS%NGROUP) = 0.0
       END SUBROUTINE
C
      END MODULE XINF_CLASS
```

図 3.6 サブクラス XEFF_CLASS (FORTRAN90)

```
      MODULE XEFF_CLASS
C
C *** BRING NUCLIDE_CLASS INTO SCOPE
C
       USE NUCLIDE_CLASS
C
       IMPLICIT NONE
       TYPE XEFF
        TYPE(NUCLIDE) NUC
        REAL, POINTER, DIMENSION(:) :: XEFF_CAP,   ! EFFECTIVE CAPTURE X-SEC
     1                                 XEFF_FIS,   ! EFFECTIVE FISSION X-SEC
     2                                 XEFF_PRO,   ! EFFECTIVE PRODUCTION X-SEC
     3                                 XEFF_TRA    ! EFFECTIVE TRANSPORT X-SEC
        REAL, POINTER, DIMENSION(:,:) :: XEFF_SCA ! EFFECTIVE SCATTERING X-SEC
       END TYPE
C
      CONTAINS
C
       SUBROUTINE INIT_XEFF(THIS, NGROUP)
        TYPE(XEFF), INTENT(INOUT) :: THIS
        INTEGER, INTENT(IN) :: NGROUP
C
        WRITE(*,*) '<INIT_XEFF>'
        ALLOCATE (THIS%XEFF_CAP(THIS%NGROUP),
     1            THIS%XEFF_FIS(THIS%NGROUP),
     2            THIS%XEFF_PRO(THIS%NGROUP),
     3            THIS%XEFF_TRA(THIS%NGROUP),
     4            THIS%XEFF_SCA(THIS%NGROUP, THIS%NGROUP))
        THIS%XEFF_CAP(1:THIS%NGROUP) = 0.0
        THIS%XEFF_FIS(1:THIS%NGROUP) = 0.0
        THIS%XEFF_PRO(1:THIS%NGROUP) = 0.0
        THIS%XEFF_TRA(1:THIS%NGROUP) = 0.0
      THIS%XEFF_SCA(1:THIS%NGROUP, 1:THIS%NGROUP) = 0.0
       END SUBROUTINE
C
      END MODULE XEFF_CLASS
```

図 3.7 サブクラス SIGO_CLASS (FORTRAN90)

```
      MODULE SIGO_CLASS
C
C *** BRING NUCLIDE_CLASS INTO SCOPE
C
       USE NUCLIDE_CLASS
C
       IMPLICIT NONE
       TYPE SIGO
        TYPE(NUCLIDE) NUC
        REAL,POINTER,DIMENSION(:) :: SIGO       ! BACKGROUND X-SEC
       END TYPE
C
      CONTAINS
C
       SUBROUTINE INIT_SIGO(THIS,NGROUP)
        TYPE(SIGO), INTENT(INOUT) :: THIS
        INTEGER, INTENT(IN) :: NGROUP
C
        WRITE(*,*) '<INIT_SIGO>'
        ALLOCATE (THIS%SIGO(THIS%NGROUP))
        THIS%SIGO(1:THIS%NGROUP) = 0.0
       END SUBROUTINE
C
      END MODULE SIGO_CLASS
```

図 3.8 サブクラス FACTOR_CLASS (FORTRAN90)

```
      MODULE FACTOR_CLASS
C
C *** BRING NUCLIDE_CLASS INTO SCOPE
C
       USE NUCLIDE_CLASS
C
       IMPLICIT NONE
       TYPE FACTOR
        TYPE(NUCLIDE) NUC
        REAL,POINTER,DIMENSION(:) :: FACTOR_CAP,   ! EFFECTIVE CAPTURE X-SEC
     1                               FACTOR_FIS,   ! EFFECTIVE FISSION X-SEC
     2                               FACTOR_PRO,   ! EFFECTIVE PRODUCTION X-SEC
     3                               FACTOR_TRA    ! EFFECTIVE TRANSPORT X-SEC
        REAL,POINTER,DIMENSION(:,:) :: FACTOR_SCA ! EFFECTIVE SCATTERING X-SEC
       END TYPE
C
      CONTAINS
C
       SUBROUTINE INIT_FACTOR(THIS,NGROUP)
        TYPE(FACTOR), INTENT(INOUT) :: THIS
        INTEGER, INTENT(IN) :: NGROUP
C
        WRITE(*,*) '<INIT_FACTOR>'
        ALLOCATE (THIS%FACTOR_CAP(THIS%NGROUP),
     1            THIS%FACTOR_FIS(THIS%NGROUP),
     2            THIS%FACTOR_PRO(THIS%NGROUP),
     3            THIS%FACTOR_TRA(THIS%NGROUP),
     4            THIS%FACTOR_SCA(THIS%NGROUP,THIS%NGROUP))
        THIS%FACTOR_CAP(1:THIS%NGROUP) = 0.0
        THIS%FACTOR_FIS(1:THIS%NGROUP) = 0.0
        THIS%FACTOR_PRO(1:THIS%NGROUP) = 0.0
        THIS%FACTOR_TRA(1:THIS%NGROUP) = 0.0
      THIS%FACTOR_SCA(1:THIS%NGROUP,1:THIS%NGROUP) = 0.0
       END SUBROUTINE
C
      END MODULE FACTOR_CLASS
```

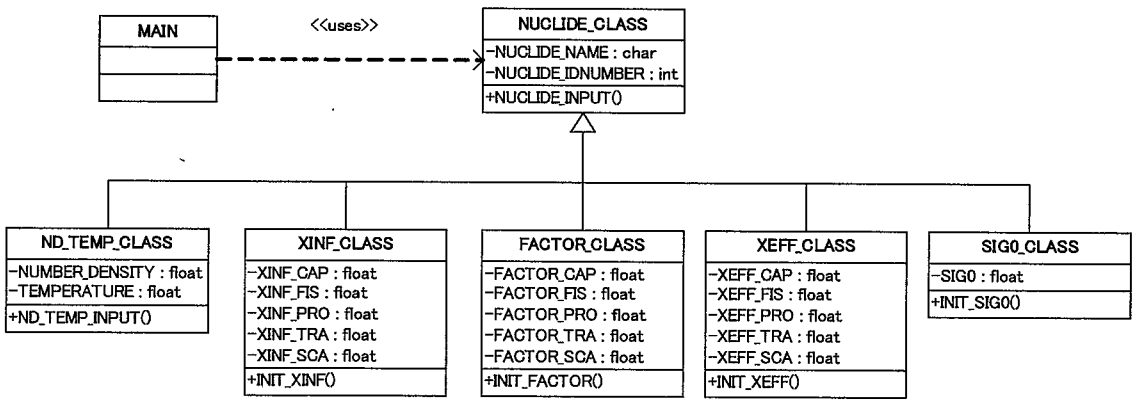


図 3.9 NUCLIDE_CLASS 関連のクラス図

第４章　Pythonによるプロトタイプ作成

第４章　Pythonによるプロトタイプ作成

本章では、Python[4-1]をベースとした次世代解析システムのプロトタイプ作成による検討結果をまとめる。ここでの対象は高速炉の炉物理（炉心核特性）計算とした。なお、フランスの次世代シミュレーションコードシステムの開発計画の中では、SALOME[4-2]と呼ばれる統合プラットフォームの開発を進めており、SALOME のグルー言語として、Python が採用されている。

4.1　プロトタイプ作成による検討

Python のモジュール機能を用いたツールボックス型の解析システム、ソルバーや体系データ、核反応断面積の取り扱い方法、演算子オーバーライドによる核反応断面積セットに対する四則演算の利用等について検討した結果を以下にまとめる。

4.1.1　Python のモジュール機能と解析用ツールボックス

フランスの次世代シミュレーションコードシステムの開発計画では、ツールボックスという概念が考えられており、具体的には、Neutronics ツールボックス、Thermal-Hydraulics ツールボックス、Fuel ツールボックスなどが挙げられている。

Python では、モジュール（module）とインポート（import）という概念があり、コンピュータ上のディレクトリとファイルを、Python プログラム上のモジュールとして取り扱うことが可能である。Python では、言語自体の機能はなるべく最小限にするように設計されており、多くの機能はモジュールの形でまとめられている。これらはライブラリと呼ばれている。Python には標準のライブラリが数多く添付されており、標準ライブラリが一体となってプログラミング言語の機能を果たしている。

Python では、ファイルとモジュールが同一視されているため、Python のプログラムを一旦書けば、そのプログラムを import して使えば、モジュールとして見ることができる。このため、独自に開発したプログラムをライブラリとして登録することも可能であり、標準ライブラリ以外に、多くの拡張ライブラリがインターネット上で配付されている。

Python のモジュール機能を使って、Python のライブラリとして、炉物理関連のツールをまとめれば、「Neutronics ツールボックス」に相当するものができあがると考えられる。このため、Neutronics という Python の拡張ライブラリとして、プロトタイプを作成した。

図 4.1.1-1 に、Neutronics ライブラリのディレクトリ構造を示す。このディレクトリ構造

が、そのまま、Python のモジュール構造として認識される。各ディレクトリに含まれる「__init__.py」というファイルは、ディレクトリ構造をモジュールとして認識させるためにある。このファイルに、モジュールを読み込んだ際に自動的にさせたい操作や、モジュールについてのドキュメントを記述することも可能であるが、モジュールとして認識させたいだけであれば、空のファイルで問題ない。現在のところ、すべて「__init__.py」は空ファイルとなっている。

このようなディレクトリ構造を決定することがシステム設計につながるので、炉物理の専門家が見て不自然でない分類が必要である。今回、暫定的に設定したものであるが、今後、このような分類方法について検討していく必要がある。例えば、今回、基本物理量データ構造・ツールや基本システムデータ構造・ツールも、Neutronics ライブラリに入っているが、本来 Thermal-Hydraulics ライブラリ、Fuel ライブラリ等が存在するはずであるので、Neutronics ライブラリではなく、もうひとつ上の階層にあるべきである。

なお、輸送方程式や燃焼方程式のソルバー（アルゴリズム）と、核データや体系データなどのデータ（データ構造）を同列に取り扱い、分離されていることに注目する必要がある。通常、「解析コード」といった場合、どのようなモデル化や手法が用いられているかというソルバーの特徴（アルゴリズム）のみが強調される傾向がある。このため、データの取り扱いは、解析コードの一部とみなされ、アルゴリズムとデータが不可分となっていることが多い。アルゴリズムとデータをセットとするのは、オブジェクト指向のクラスの概念であり、アルゴリズムとデータに密接な関係があるのは当然である。しかし、複数のアルゴリズムを自由に結合して使うためには、アルゴリズムから独立した形でデータ構造を抜き出し定義する必要がある。そして、特定のアルゴリズムに依存しない、そのデータ構造に対する処理（メソッド）のみをまとめておく必要がある。

ここでは、アルゴリズムから独立したデータ構造として、Geometry、NuclearData、GroupData、Nuclide 等を定義した。例えば、GroupData は、炉物理解析では非常に基礎的なデータ構造であり、輸送方程式解法、燃焼方程式解法、核反応データ、感度解析等、すべての過程で利用されるデータである。

- 炉物理・中性子工学関連データ構造&ツール
  - Boltzmann　：輸送方程式ソルバー&データ構造
  - Burnup　　：燃焼方程式ソルバー&データ構造
  - Geometry　：計算体系関連データ&ツール
  - NuclearData：核反応断面積関連データ&ツール
  - Sensitivity　：感度係数関連データ&ツール

- 基本物理量データ構造・ツール
  - GroupData　：エネルギー群構造データ&ツール

- Nuclide　　：核種関連データ＆ツール

・ 基本システムデータ構造・ツール
  - Mixin　　：オブジェクトが持つ基本性質を定義する Mix-in
  - Wrappers　：サブシステム結合用 Wrapper（Adapter）

4.1.2 Neutronics ライブラリの各モジュール

本節では、Neutronics ライブラリの各モジュールについて、ソルバー関連とデータ関連に分けて説明する。

4.1.2.1 中性子輸送計算（Boltzmann モジュール）

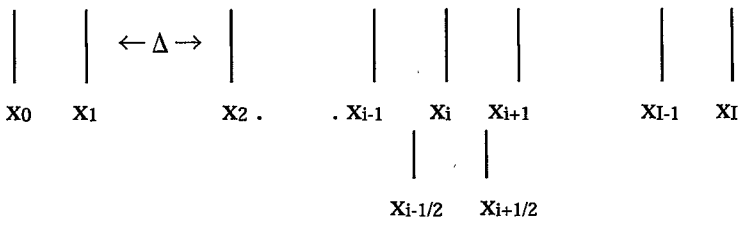
ここでは、中性子輸送方程式の数値解法として最も簡単なモデルである１次元スラブ体系１群拡散計算を行うモジュール（Diffusion1D.py）を作成した。このモジュールは Python のみで書かれている。以下に、数値解法を簡単にまとめる[4-3]。

(1) 中性子拡散方程式の数値解法

１次元スラブ体系の中性子拡散方程式は以下のように表すことができる。

$$-\frac{d}{dx}D(x)\frac{d\phi(x)}{dx}+\Sigma_a(x)\phi(x)=\frac{1}{\lambda}\nu\Sigma_f(x)\phi(x)$$

ここで、以下のような１次元スラブ体系を考える。



このとき、中性子拡散方程式は、以下のように書き換えることができる。

$$a_{i,i-1}\phi_{i-1}+a_{i,i}\phi_i+a_{i,i+1}\phi_{i+1}=\frac{1}{\lambda}f_i\phi_i\equiv S_i,\qquad i=1,2,\mathrm{K}\ ,I-1$$

ただし、ここで以下の式が成り立つとする。

$$a_{i,i-1} = -\frac{1}{2}\left(\frac{D_i + D_{i-1}}{\Delta^2}\right)\left(1 - \frac{c}{2i-1}\right)$$

$$a_{i,i} = \Sigma_{ai} + \frac{1}{2}\left(\frac{D_{i-1} + 2D_i + D_{i+1}}{\Delta^2}\right)$$

$$a_{i,i+1} = -\frac{1}{2}\left(\frac{D_{i+1} + D_i}{\Delta^2}\right)\left(1 + \frac{c}{2i-1}\right)$$

$$f_i = \nu\Sigma_{fi}$$

なお、Σ_{ai}、D_iは、$x_{i-1/2} \le x \le x_{i+1/2}$の区間における巨視的吸収断面積、拡散係数を表す。

ここで、核分裂源S_iが既知であるとすると、以下のようにして、各メッシュiの中性子束を求めることができる。この計算は内側反復と呼ばれる。

$$\phi_{I-1} = \alpha_{I-1}$$

$$\phi_{I-2} = -A_{I-2}\phi_{I-1} + \alpha_{I-2}$$

M

$$\phi_i = -A_i\phi_{i-1} + \alpha_i$$

ただし、ここで、以下の式が成り立つ。

$$A_1 = \frac{a_{1,2}}{a_{1,1}}, A_i = \frac{a_{i,i+1}}{a_{i,i} - a_{i,i-1}A_{i-1}}$$

$$\alpha_1 = \frac{S_1}{a_{1,1}}, \alpha_i = \frac{S_i - a_{i,i+1}\alpha_{i-1}}{a_{i,i} - a_{i,i-1}A_{i-1}}$$

実際には、核分裂源は既知ではないので、中性子束の初期値$\phi_i^{(0)}$と固有値の初期値$\lambda_i^{(0)}$を仮定して、核分裂源を以下の式

$$S_i^{(0)} = \frac{1}{\lambda^{(0)}}\nu\Sigma_{fi}\phi_i^{(0)}$$

で求め、内側反復の計算を繰り返し行う必要がある。この繰り返し計算は外側反復と呼ばれる。通常、以下のような条件を満たすところで、繰り返し計算を終了させる。

$$\frac{\lambda^{(n)} - \lambda^{(n-1)}}{\lambda^{(n-1)}} < \varepsilon$$

上記の内側反復、外側反復といった計算の大きな流れは、体系や中性子のエネルギー群数が変わった場合であっても、拡散近似を使わない場合でも同じである。

(2) Pythonによる実装

ここでは、上記の計算を行うために、このモジュールの中に計算を行うための**Diffusion1D**、**Geometry1D**、**MacroscopicCrossSection**という３つのクラスを定義した。

図 4.1.2-1 にこのモジュールのクラス図を示す。

、内側反復の計算は、上述のように数学的には再帰関数で定義されているが、Python では、再帰関数を直接取り扱えるので、数学的表記とほとんど同じであり簡潔に記述できる。Diffusion1D.py のソースプログラムは、付録 A の図 A-5 に示した。なお、再帰関数は、Fortran90、C、C++等でも利用可能であるが、Fortran77 では利用できない。

Diffusion1D クラスは、内側反復の計算のみを表しており、実際に使う場合は、外側反復の計算部分から Diffusion1D オブジェクトを呼び出すことになる。図 4.2.1.2-2 に、この計算モジュールを使った例を示す。この例に示される、for 文によるループが外側反復の計算に相当している。この繰り返し計算部分は、デザインパターン[4-4]の Iterator パターンの一種となっている。外側反復の計算では、内側反復に相当するオブジェクト（ここは、d1d となっている）に、次の世代を計算する nextgen()というメッセージ（メソッド）を送っているだけであるので、内側反復に、異なったアルゴリズムの内側反復オブジェクトがきても、外側反復計算部分を全く変更する必要はない。

炉物理上でも「次の中性子世代の計算をする」という概念があり、上記の考え方は、デザインパターン（Iterator パターン）を知らなくても、炉物理の知識があれば、比較的容易に理解できると思われる。炉物理の知識を理解する上で、「内側反復」、「外側反復」はあるまとまった概念（オブジェクト）として認識しているからである。

図 4.2.1.2-3 に実行結果を示す。Diffusion1D クラスは、可視化のためのメソッド（visualize()）も持っており、中性子束分布の収束状況をリアルタイムで見ることも可能である。なお、実行速度は、今回の計算問題は小さいので特に問題にならなかった（Pentium4, 2.0GHz, Redhat Linux 7.2：数十秒）。

4.1.2.2 燃焼計算（Brunup モジュール）

ここでは、行列指数法を用いた燃焼計算行うためのモジュール(Burnup.py)を作成した。燃焼計算コードとして有名な ORIGEN2[4-5]でも行列指数法が用いられており、燃焼計算の解法としては一般的なものである。

(1) 燃焼方程式の数値解法

燃焼方程式は以下のように表される。

$$\frac{dN_i}{dt} = \sum_j \gamma_{ji}\sigma_{f,j}N_j\phi + \sigma_{c,i-1}N_{i-1}\phi + \lambda_{i'}N_{i'} - \sigma_{f,i}N_i\phi - \sigma_{c,i}N_i\phi - \lambda_i N_i$$

ここで、右辺の各項は以下を表す。

第１項：j 核種の核分裂による寄与

第２項：i-1 核種が中性子捕獲により i 核種になる寄与

第３項：I'核種がβ崩壊によりｉ核種になる寄与

第４、５項：ｉ核種の中性子吸収による消滅

第６項：ｉ核種のβ崩壊に依る消滅

なお、ここで(n,2n)、(n3,n)、(γ,n)反応などは考慮していない。これらの反応が重要な場合には考慮する必要がある。

燃焼方程式の数値解法には、Euler 法、Runge-Kutta 法、行列指数法などがあるが、ORIGEN2 でも用いられている行列指数法による燃焼方程式の解法について、以下に簡単にまとめる。

上述の燃焼方程式は、一般に以下のような行列式に書き換えることができる。

$$\frac{d\mathbf{N}}{dt} = \mathbf{M}\mathbf{N}$$

この方程式の解は、形式的に以下のように書ける。

$$\mathbf{N}(t) = \exp(\mathbf{M}t)\mathbf{N}(0)$$

ここで、$\exp(\mathbf{M}t)$は以下の式により定義される行列である。

$$\exp(\mathbf{M}t) = \sum_{n=0}^{\infty} \frac{1}{n!}(\mathbf{M}t)^n$$

(2) Python による実装

Python では、SciPy[4-6]という現在開発中のライブラリがあり、この中に行列指数を計算する関数 expm がある。SciPy は現段階ではまだα版であるが、expm については動作することが確認できたので、これを利用して、簡単な燃焼計算モジュール（Burnup.py）を作成した。

図 4.2.1.2-1 に Burnup 関連のクラス図を示す。クラスは BurnupCalculation のひとつだけである。燃焼行列と初期ベクトル、燃焼期間を設定するためのメソッドと、燃焼計算を行う solve メソッドからなる。図 4.2.1.2-2 に、Burnup モジュールの利用例を示す。ここでは、U-238、U-239、Np-239、Pu-239 のチェーンのみを考慮して、簡単な燃焼・崩壊計算を行っている。ユーザーは、BurnupCalculation のインスタンスを生成し、燃焼行列、初期ベクトル、燃焼期間を設定して、計算を実行させるという操作が明確に読みとれる。実行結果は、図.2.1.2.2-3 に示した。

ここでは、直接、燃焼行列を作成しているが、燃焼計算を行う場合の燃焼チェーンを、直接、燃焼行列の形で認識している人はいないと思われる。通常は、燃焼チェーンの図をイメージしている。したがって、本来、燃焼チェーンを図の形に近いイメージで表現できるクラスを作成して、燃焼行列を生成するメソッドを持たせるのが良いと思われる。この

ようにすることで、燃焼チェーンというデータ構造と、燃焼計算のアルゴリズム（ここでは、行列指数法）を分離することが可能となる。

4.1.2.3 炉心体系データ（Geometry モジュール）

炉物理計算を行う際、炉心体系データは、燃料集合体や燃料ドロワなどの単位（あるいはそれを更にいくつか分割した単位）を計算メッシュとして取り扱われる。通常、コードの中で、この計算メッシュは Fortran の配列として表現されるため、炉心計算コードの体系入力データは、計算メッシュの配列を直接指定するような形式のものが多い。

計算機の容量が向上したため、計算メッシュは詳細化される傾向にあり、計算メッシュを直接取り扱うのは煩雑になってきている。

(1) 高速炉炉心体系データの取り扱い

高速炉実機で燃焼を考慮した解析する際には、時間依存のない臨界実験解析などに比べて、炉心体系の取り扱いが複雑になる。臨界実験体系では考慮する必要のない原子数密度の時間変化を取り扱う必要があるだけでなく、燃料交換や制御棒の引き抜きなどを考慮しなければならないからである。現在の炉心計算コードでは、炉心体系データを計算メッシュとして入力しなければならないため、燃料集合体を一本だけ取り出して違うものに交換したり、制御棒を少しだけ引き抜いたりする操作を、すべて計算メッシュとして考えなければいけないため、このような操作は非常に複雑になる。

ここでは、通常、解析する人のイメージに近い形で、炉心体系データを取り扱う方法を考える。

実際の炉心は、燃料集合体が集まって構成されている。更に、その燃料集合体は、燃料ピンが集まって構成される。オブジェクト指向の技術を使えば、こういう状態を計算機上で表現するのは簡単である。更に、炉物理解析の炉心計算では、通常、燃料集合体は均質化してしまうので、以下のように単純化して考えることができる。

「燃料ペレット、被覆管、減速材、ラッパ管などの単一組成の物質（SimpleMaterial）が集まって媒質（Medium）を構成される。この媒質が集まって、ひとつの燃料集合体（HexagonalAssembly）が構成される。更に燃料集合体が集まって、高速炉の炉心が構成される（ConventionalFastReactorCore）。」

このような考え方をクラス図にすると図 4.1.2.3-1 のようになる。これを実装した Python プログラムは付録 A の図 A-7 に示した。このモジュールを使うと、図 4.1.2-3-2 のような形で、炉心の体系データを記述することができるようになる。この利用例の最後に出てくる

「joyocore」という変数（オブジェクト）の形として、炉心体系データをひとまとめにすることができる。

ここで示した例のように、計算メッシュを直接取り扱う方法に比べて、解析する人がイメージする炉心体系データにより近い形で表現できることが分かる。ただし、現段階では全く実用性は内ので、今後、この体系データに対する処理（例えば計算メッシュ生成等）を付加していくことにより、炉心計算とは完全に分離したまま、燃料交換や制御棒操作を取り扱う機能を追加していくことが可能になる。

(2) 炉心体系データの可視化

続いて、炉心体系データ取り扱いに関して、Python を使った可視化のサンプルを作成した。ここでは、炉心計算コード CITATION[4-7]の入力データを読み込むツール、及び、内容を確認するために、体系データを表示するツールを作成した。

図 4.1.2.3-3 にクラス図を示す。この構造はもう少し簡単化できると思われるので、今後検討が必要である。これらのクラスを直接ユーザーが使うのは煩雑であるので、citview という関数にまとめた（デザインパターンの Façade パターンの一種）。図 4.1.2.3-4 は、これらのクラスを使って、citview.py というコマンドにまとめたものである。ユーザーはこのコマンドを利用することにより、図 4.1.2.3-5 のような CITATION の体系入力データを、図 4.1.2.3-6 のような図に変換することができる。

なお、作図には、Python の標準ライブラリである Tkinter（Tcl/Tk の Tk へのバインド）を利用した。この Tkinter は、Windows 上でも、Unix 上でも利用することが可能であり、OS に依存しない GUI を作成することが可能である。また、配列データの処理には Numerical Python[4-8]を、Fortran のフォーマットを読みとるためには Scientific Python[4-9]というパッケージを利用した。なお、Tkinter は GUI 作成ツールとしては、それほど強力なものではないが、より強力な機能を持つ Qt[4-10]等を Python から利用するためのライブラリも開発されており、Python を利用した本格的な GUI 作成も可能である。また、可視化という観点では、VTK、OpenGL 等も利用可能である。

ちなみに、フランスが開発を進めている SALOME 統合プラットフォームでは、GUI 作成には Qt、可視化には VTK[4-11]が採用されている。また、3 次元形状を取り扱うためには、OpenCASCADE[4-12]という CAD ツールが用いられるようである。

4.1.2.4 中性子核反応データ（NuclearData モジュール）

高速炉核特性計算で必要となる中性子核反応データは、JFS-3 ライブラリ[4-13]が主に用いられている。JFS-3 ライブラリは独自のバイナリ形式を採用している。ここでは、JFS-3 ライブラリを Python から取り扱う方法について検討した結果をまとめる。

(1) JFS-3 ライブラリの XML 形式への変換

Python から直接バイナリデータを扱うことも可能であるが、JFS-3 のバイナリ形式のデータを直接 Python から取り扱うのは煩雑であり、機種依存性を発生させる恐れもあるので、JFS-3 ライブラリを、一旦、XML 形式に変換し、Python から XML 形式の JFS-3 ライブラリを取り扱うことにした。

バイナリ形式の JFS-3 ライブラリをテキスト形式に変換するための既存の Fortran プログラムを少し変更して、XML 形式に変換するプログラムを作成した（付録 A の図 A-10)。図 4.1.2.4-1 に Fortran プログラムにより変換された XML 形式の JFS-3 ライブラリを示す。

Python での XML の処理には SAX を用いた。当初、XML の処理に DOM を使っていたが、DOM を用いた場合、データ量が多いため負荷がかかり、現在の 1 核種分の XML のデータを処理するのに 5 分程度かかった。このため、XML 処理部分は、SAX で実装することにした。SAX を使うことによって、一核種あたり、最大でも数 10 秒程度で処理できるようになった。

エネルギー群数が大きくなることにより、XML の処理時間が長くなると思われるが、この結果から、核反応データを XML 形式で取り扱うことは、十分可能であると考える。また、常に XML を処理して読み込むのは負荷が大きいため、Python のオブジェクトを保存するための Pickle 形式を利用することも可能である。

ここでは、XML から Python のオブジェクトにして、Python オブジェクトを Pickle 形式に保存できるようにしてある。Pickle 形式で保存するのも呼び出す場合、一核種あたり数秒であり、SAX よりも更に高速に処理が可能となる。

更に、Pickle にはテキストファイルではなく、バイナリファイルで取り扱うオプションもあり、データ量も小さくできることを確認した。Pickle 化されたファイルは、1 核種あたり 150kB～550kB である。なお、XML ファイルは、1 核種あたり 150kB～1.1MB である。

(2) Python による JFS3 ライブラリの実装

図 4.1.24-2 に JFS3utils.py のクラス図を示す（ソースプログラムは付録 A の図 A-9)。この図から分かるように、JFS-3 ライブラリは、最終的に、GroupConstant の集約された GroupConstantSet というオブジェクトになる。GroupConstant クラスの派生クラスとして、1 次元配列の断面積データ（GroupMicroscopicCrossSection1D)、2 次元配列の断面積データ（GroupMicroscopicCrossSection2D）である弾性散乱、非弾性散乱、(n,2n)反応断面積、f-table（Ftable）が定義されている。また、核種を識別するために、Nuclide クラスを利用している。

通常、JFS-3 ライブラリの全核種のデータをメモリ上に常駐させておく必要はないので、これらのクラスを使って作成された makeAllPickles という関数で、すべての核種を Pickle 形式に変換できるようになっている。この変換操作は一度だけ行い、その後は、Pickle 形式に変換されたものを利用する。

現在の高速炉核特性解析システムでは、JFS-3 ライブラリを取り扱うコードは、CASUP、SLAROM 等がある。これは、二つのコードにほとんど同じ機能、同じデータ構造を取り扱うプログラムが重複して組み込まれているということである。

今回、JFS-3 ライブラリは複合オブジェクトとして実装しており、JFS-3 ライブラリだけでも、かなり巨大なデータ構造である。CASUP、SLAROM コードから、JFS-3 ライブラリの取り扱いに関連する部分だけ取り出して部品化することで、複雑さを小さくすることができる。

CASUP や SLAROM をひとつのクラスとしてみた場合、衝突確率の計算やバックグラウンド断面積の計算等もメソッドとして取り込まれていることになるが、JFS-3 ライブラリにこれらのメソッドを持たせるのは責任範囲が広すぎると考えられる。適宜、分解してクラスを定義していく必要があると考えられる。一方、JFS-3 ライブラリのバイナリ形式のデータをテキストに変換する場合には、現在、DT1OUT 等の独立したユーティリティプログラムを利用しているが、このような機能は JFS-3 ライブラリが持つべき機能（メソッド）である。また、f-table の内挿計算等は、JFS-3 ライブラリのデータ構造に強く依存しているので、JFS-3 ライブラリの機能（メソッド）とするべきである。

上のような分析から、f-table の内挿と可視化のメソッドを実装してみた。図 4.1.2.4-3 に、f-table を内挿して任意のエネルギー群、バックグラウンド断面積、温度、R パラメータに対する自己遮蔽因子を計算するメソッド（getffactor）の利用例を示す。f-table の内挿計算は、f-table のみに依存するものであるので、Ftable クラスのメソッドとしている。

ここでは、最初に help コマンドを使って、getffactor メソッドの引数を確認している。Python の help コマンドを使えば、ソースプログラムから必要最低限の情報を取り出してくれるので、インタラクティブにメソッドの内容を確認することも可能である。特にコメントは入れていないが、物理的な内容を知っている人であれば、引数の意味は簡単に想像できるはずである。最後に、visualize メソッドを使って、一次元配列の断面積データを可視化している。

なお、パラメータの補完には、SciPy ライブラリのスプライン関数を利用した。今回の実装の目的はプロトタイピングであるので、正確な結果が出ているということは確認していない。

4.1.2.5 感度解析（Sensitivty モジュール）

前節に示したとおり、JFS ライブラリから無限希釈断面積等が取り出せるようになった

ので、感度解析への応用を例にして、クラスの抽象化を検討した。

感度解析に関して知識のある人であれば、以下のような表現で感度係数を利用したライブラリ効果の計算方法を理解することができる。

「ライブラリ効果を求めるためには、断面積の差に感度係数を乗じればよい。」
→ JENDL-3.3 と-3.2 のライブラリ効果= (j33−j32)／j32×sens

実際、このような表現は日常的に使われる。暗黙のうちに、各反応断面積（核分裂断面積、捕獲反応断面積等)、各エネルギー群に対して計算する必要があることを含んでいる。かなり抽象化された表現であるが、対象のデータが決定されれば、加減乗除の計算方法は自ずと決まることが多いため、物理的な内容を知っている人どうしであれば、意味は正確に通じる。逆に考えると、このような物理的な内容を実装をすることは、専門家どうしが持っている暗黙の了解（技術ノウハウ）をプログラムとして正確に記述することになり、技術の継承にも貢献できる可能性がある。

(1) 群定数（GroupData、GroupDataSet）クラスの実装

図 4.1.2.5-1 に GroupData 関連のクラス図を示す。GroupData クラスと GroupDataSet クラスは、それぞれ、JFS3 ライブラリの実装で作成した GroupConstant クラスと GroupConstantSet クラスによく似ているが、ここでは、演算子オーバーロードのテストが主な目的であるので、問題を単純化するために、新たに別のクラスを定義した（本来は、共有すべきものである)。ここでは、Wrapper モジュールを使って、GroupConstant オブジェクトから GroupData オブジェクトを生成するようにした。

GroupData クラス、及び、GroupDataSet クラスには、__add__、__sub__、__mul__、__div__メソッド（それぞれ、+、-、*、/演算子に対する処理を定義するメソッド）を定義して、GroupData、GroupDataSet オブジェクトに対して、直接、加減乗除ができるようにした。また、今回の計算では、群縮約計算やレサジーの計算が必要であったが、これらの計算機能は、GroupData、GroupDataSet クラスに持たせるのが妥当と考え、関連するメソッドや属性を追加した。

今回使う群データである、無限希釈断面積、感度係数ともに GroupDataSet オブジェクトとして取り扱う。

感度係数、断面積等の物理量をひとつの「変数」として扱うこと、「感度係数と断面積の四則演算」という概念を一般的な「+、-、*、/」という演算子で表現することができる。これは、Python の演算子のオーバーライドの機能を使って実現しており、GroupDataSet に対して、+、-、*、/を定義している。オブジェクト指向言語では、通常、このような機能を

持っている。

(2) SAGEP コード関連（SAGEPutils）モジュールの実装

感度解析コード SAGEP[4-14]の計算結果を処理して、Python に取り込めるようなツールを作成し、SAGEPutlis モジュールにまとめた。図 4.1.2.5-2 に SAGEPutils 関連のクラス図を示す。このモジュールで定義したクラスの構造は少々複雑であるが、makePickle 関数を呼び出すことで、SAGEP の計算結果（テキスト形式）を Pickle 形式に変換して保存できるようにした。

(3) 感度解析の実行例

図 4.1.2.5-3 に感度解析の実行のためのプログラム (sensana.py) を示す。ここでは、JFS3 ライブラリの断面積差と感度係数から、JENDL-3.3 と JENDL-3.2 のライブラリ効果を求めている。

実際にはデータの読み込み、群縮約などの操作が含まれているので少し複雑になるが、重要なポイントは、感度解析の計算の部分である。sensana.py では、以下のような式で書かれている。

```
diff = (j33_18g - j32_18g) / j32_18g
effect = diff * sens_18g
```

これは、前述の抽象化された式と全く同じである。

実行結果を図 4.1.2.5-2 に示す。結果は大量の数値データとなるので、グラフ化して出力するようにした。グラフ化には、GroupData、GroupDataSet の visualize メソッドを利用している。断面積、感度係数、ライブラリ効果にかかわらず、GroupData、GroupDataSet クラスを使っているので、同じ visualize メソッドを再利用して、グラフ化できている点も注目に値する。

なお、グラフ化には、グラフ作成ツール Gnuplot と、Python から Gnuplot を利用するためのモジュールである gnuplot.py を用いた。

(4) 今後の課題

GroupDataSet と GroupData は重複した機能（コード）を持っているので、デザインパターンの適用等により、もっとうまくまとめる方法があると思われる。また、GroupData、

GroupDataSetの可視化に関して、物理量の単位に関するクラスをうまく定義できれば、演算後のデータに対しても、きちんと単位を変換することができるのではないかと感じた。また、物理量の単位については、解析システム全般に亘って共有化できるはずである。Martin Fowler著の「アナリシスパターン」でも、対象は通貨をであるが、量と単位に関する分析例が取り上げられている。物理量とその単位に関して、「量・単位」に関するアナリシスパターンを適用できる可能性があると思われる。これらは、今後の課題である。

4.1.2.6 基本システムデータ構造

これまでに示してきたクラス図には、VisualizeAndSerializeというクラスが含まれており、多くのクラスがこのクラスの派生クラスとなっていることが示されている。このVisualizeAndSerializeクラスは、Mixinモジュールで定義されている。

最初の実装では、各クラスに、インスタンスの内容を確認するためのshowメソッドや、データをセーブ・ロードするためのdump、loadメソッドを書いていたが、非常に面倒に感じた。よく考えてみると、インスタンスの中味を確認したり、ハードディスクに保存したり・読み出したりする機能は、すべてのオブジェクトにあってしかるべきものである。

このため、すべてのクラスは、Mixinモジュールの VisibleAndSerializableクラスを継承するようにした。こうすることで、すべてのクラスで、全く同じメソッド名を使って、中味の確認（show）、セーブ（dump）・ロード（load）できるようになる。

showメソッドの実装には、Pythonのイントロスペクションの機能を利用している。きれいな表示は期待できないが、イントロスペクションを利用しているので、今後、定義されるクラスすべてに対しても、とにかく中味を表示することが可能である。また、dump、loadメソッドの定義には、Pickleの機能を利用している。Pickleを使えば、どのようなオブジェクトでもPickle形式に変換してくれるので、dump、loadメソッドも、今後、定義されるクラスすべてに対して有効である。

なお、メソッドだけ取り込む、今回のような使い方は、Mixinと呼ばれており、今回の継承の使い方はMixinと呼べる。ちなみに、Ruby等では、多重継承を許しておらず、Mixinの機能のみを提供している。Pythonでは、多重継承を使うことでMixinと同様のことが可能である。

4.1.3 まとめと今後の課題

高速炉の炉物理分野を対象として、Pythonを用いた次世代解析システムのプロトタイプ作成による検討を実施した。Pythonのモジュール化機能を用いることにより、必要な機能やデータを構造化して分類し、必要に応じてインポートして使うという「ツールボックス」のイメージでシステム化できる見込みを得た。また、各ツール（部品）のサンプルとして、解析手法（炉心計算、燃焼計算）や解析データ（体系データ、核反応断面積）を取り扱うモジュールも作成した。

今回の検討結果から、解析上必要となる体系データや核反応断面積データ等に対する作成、保存、利用、可視化といった操作を一貫してPythonから制御できると考えられる。

一方で、炉物理解析で重要、かつ、量の多くなる核反応断面積データを、XML形式やPickle形式に変換して取り扱うことが可能であることも確認した。また、Pythonの演算子オーバーロードの機能を用いることにより、断面積データや感度係数に対する演算を、従来に比べて高度に抽象化できることも示した。

なお、今回のプロトタイプ作成では、JFS-3ライブラリをXML形式に変換するプログラム以外は、すべてPythonを使っているが、前作の報告書で示したとおり、SWIG、f2py等を利用することで、PythonからC言語やFortranで作成されたプログラムを、あたかもPythonで書かれているかのように利用することが可能である。今後、第2章で示されたVisual Basic .NETとFortranの結合のように、Pythonでも既存のFortranで書かれたコードの機能を、実用レベルで活用できることを確認する予定である。

図 4.1.1-1 Neutronics モジュールのディレクトリ構造

```
Neutronics/
        Boltzmann/
                __init__.py
                Diffusion1D.py
        Burnup/
                __init__.py
                Burnup.py
        Examples/
                burnup.py
                citview.py
                diff1d.py
                geom.py
                sensana.py
        Geometry/
                 CITATION/
                          __init__.py
                          CITATIONutils.py
                 __init__.py
                 Core.py
        NuclearData/
                JFS3/
                         __init__.py
                         JFS3utils.py
                         jfs2xml.f
                __init__.py
        Sensitivity/
                SAGEP/
                         __init__.py
                         SAGEPutils.py
                __init__.py
        utils/
                __init__.py
                LISTutils.py
        __init__.py
        GroupData.py
        Mixin.py
        Nuclide.py
        Wrappers.py
```

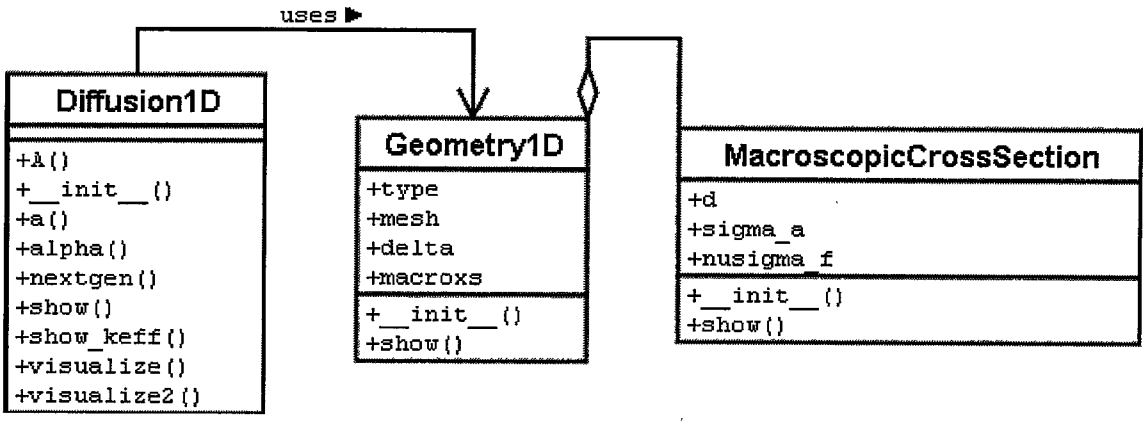


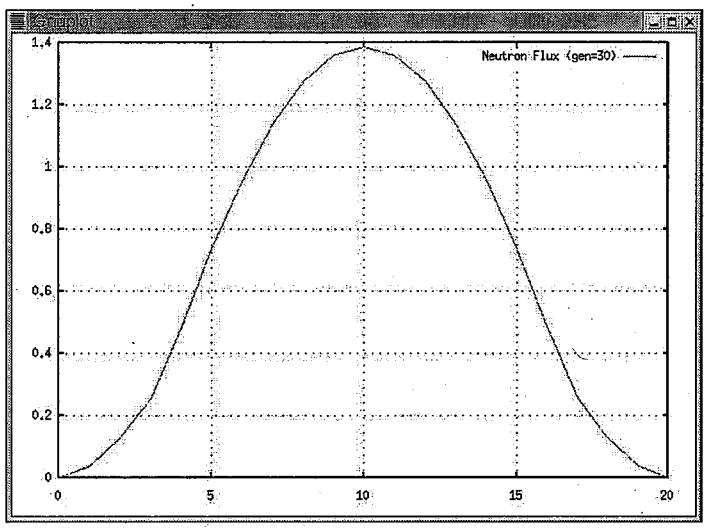
図 4.1.2.1-1 炉心計算 (Neutronics.Diffusion1D) モジュールのクラス図

図 4.1.2.1-2 炉心計算モジュールの利用例 (diff1d.py)

```
#!/usr/local/bin/python

from Neutronics.Boltzmann.Diffusion1D import Diffusion1D, MacroscopicCrossSection, Geometry1D

xs_u5 = MacroscopicCrossSection(1/(3*0.765), 9.24E-2, 0.145)
xs_fe = MacroscopicCrossSection(1/(3*1.15),  0.222,   0.0  )
xs_u5.show()
xs_fe.show()

geom1d = Geometry1D("slab", mesh=19, delta=0.6,
                    macroxs = [xs_fe for i in range(3)]
                    + [xs_u5 for i in range(13)]
                    + [xs_fe for i in range(3)])
geom1d.show()

#Inner Iteration
inner = Diffusion1D(geom1d)
inner.show()

#Outer Iteration
for i in xrange(50):
    inner.nextgen()
    inner.show_keff()
    inner.visualize()
    if d1d.eps < 1.0E-5:
        print "Converged!"
        break
```

図4.1.2.1-3　diff1d.pyの実行結果

```
$ python diff1d.py
gen= 1 keff= 1.00550654865 eps= 0.00550654865177
gen= 2 keff= 1.02038855967 eps= 0.0148005112828
gen= 3 keff= 1.03042990605 eps= 0.00984070850823
gen= 4 keff= 1.01832501749 eps= 0.0117474158006
gen= 5 keff= 1.01740090749 eps= 0.000907480404796
gen= 6 keff= 1.02068471693 eps= 0.00322764547833
gen= 7 keff= 1.02533879849 eps= 0.00455976412842
gen= 8 keff= 1.03009922208 eps= 0.00464278109464
gen= 9 keff= 1.03424873331 eps= 0.00402826363083
gen= 10 keff= 1.0373640022 eps= 0.00301210800589
gen= 11 keff= 1.03923821051 eps= 0.00180670266362
gen= 12 keff= 1.03981300028 eps= 0.000553087609241
gen= 13 keff= 1.0391487967 eps= 0.000638772150019
gen= 14 keff= 1.0374297344 eps= 0.00165429850431
gen= 15 keff= 1.03506481906 eps= 0.00227959085388
gen= 16 keff= 1.03248985221 eps= 0.00248773487919
gen= 17 keff= 1.03022236773 eps= 0.00219613246549
gen= 18 keff= 1.02828257371 eps= 0.00188288866349
gen= 19 keff= 1.02668469751 eps= 0.00155392713977
gen= 20 keff= 1.02540951351 eps= 0.00124204052343
gen= 21 keff= 1.02442039774 eps= 0.000964605612979
gen= 22 keff= 1.02367448786 eps= 0.000728128684453
gen= 23 keff= 1.02312890383 eps= 0.000532966330046
gen= 24 keff= 1.0227438041 eps= 0.000376394144172
gen= 25 keff= 1.02248387055 eps= 0.000254153137543
gen= 26 keff= 1.02231886172 eps= 0.000161380373919
gen= 27 keff= 1.02222363254 eps= 9.3150175425e-05
gen= 28 keff= 1.02217785312 eps= 4.47841572421e-05
gen= 29 keff= 1.02216555446 eps= 1.20318138359e-05
gen= 30 keff= 1.02217457526 eps= 8.82517802921e-06
Converged!
```



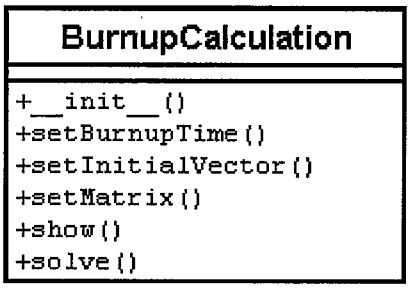


図 4.1.2.2-1　燃焼計算（Neutronics.Burnup）モジュールのクラス図

図 4.1.2.2-2 燃焼計算モジュールの利用例 (burnup.py)

```
#!/usr/local/bin/python

from math import log
from Neutronics.Burnup.Burnup import BurnupCalculation

###
### Data Preparation
###
lambda_28 = log(2)/(4.468E6*365*24*60*60)
lambda_29 = log(2)/(23.0*60)
lambda_39 = log(2)/(56.0*60*60)
lambda_49 = log(2)/(2.412E4*365*24*60*60)

phi = 1.0E15
## phi = 0.0
sigma_c_28 = 2.73 * 1E-24
sigma_f_29 = 14.0 * 1E-24
sigma_f_49 = 740.6 * 1E-24
sigma_c_49 = 273.9 * 1E-24

delta_t = 300*24*60*60

N0 = [ 10000.0, 0.0, 0.0, 0.0 ]
M = [[ -lambda_28-sigma_c_28*phi, 0.0, 0.0, 0.0 ],
     [ +lambda_28, -lambda_29-sigma_f_29*phi, 0.0, 0.0 ],
     [ 0.0, +lambda_29, -lambda_39, 0.0 ],
     [ 0.0, 0.0, +lambda_39, -lambda_49-(sigma_f_49-sigma_c_49)*phi]]

###
### Calculation
###
burn = BurnupCalculation()
burn.setMatrix(M)
burn.setInitialVector(N0)
burn.setBurnupTime(delta_t)
burn.show("initial vector:")
burn.solve()
burn.show("result vector:")
```

図 4.1.2.2-3 burnup.py の実行結果

```
$ python burnup.py
initial vector:
[ 10000.       0.       0.       0.]
result vector:
[  9.31683860e+03    9.12469050e-08    1.33405753e-05    9.88588909e-05]
```

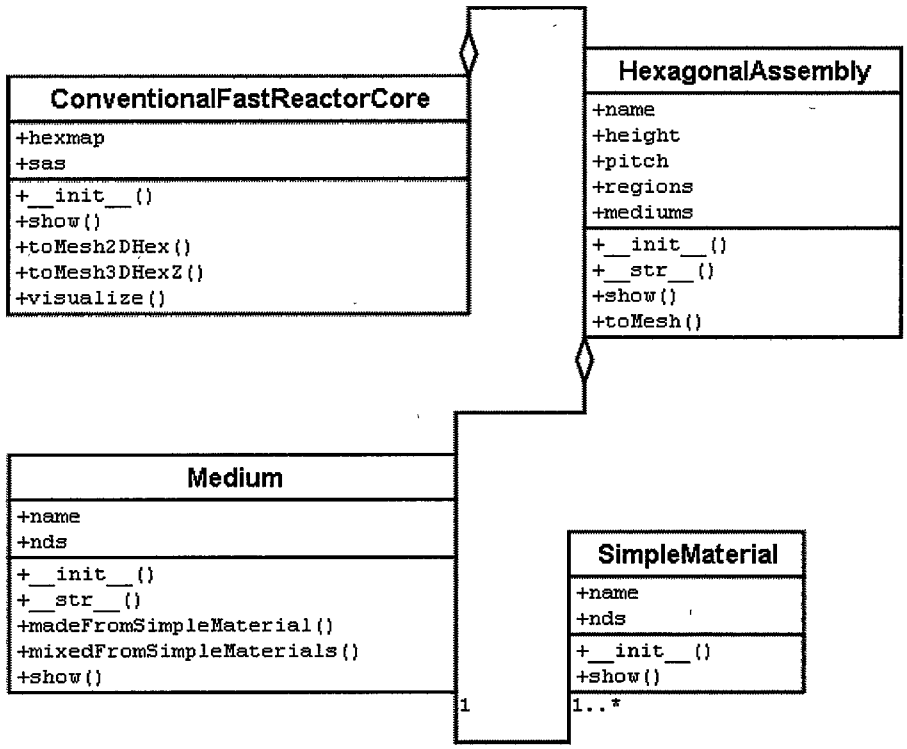


図 4.1.2.3-1　炉心体系データ（Neutronics.Geometory.Core）モジュールのクラス図

図 4.1.2.3-2 炉心体系データモジュールの利用例（core.py）

```
#!/usr/local/bin/python

from Neutronics.Geometry.Core import *

CR50=0.0435; CR52=0.8379; CR53=0.0950; CR54=0.0236;
FE54=0.0580; FE56=0.9172; FE57=0.0220; FE58=0.0028;
NI58=0.6827; NI60=0.2610; NI61=0.0113; NI62=0.0359; NI64=0.0091;

mat_fuel = SimpleMaterial('FUEL',
    {'U235':  1.57990E-03,       'U238':  5.22530E-03,
     'PU239': 1.11330E-03,       'PU240': 2.80660E-04,
     'PU241': 3.74390E-05,       'PU242': 7.96050E-06,
     'O':     1.65360E-02,       'NA':    9.34840E-03,
     'CR50':  3.26720E-03*CR50,  'CR52':  3.26720E-03*CR52,
     'CR53':  3.26720E-03*CR53,  'CR54':  3.26720E-03*CR54,
     'FE54':  1.18960E-02*FE54,  'FE56':  1.18960E-02*FE56,
     'FE57':  1.18960E-02*FE57,  'FE58':  1.18960E-02*FE58,
     'NI58':  2.09350E-03*NI58,  'NI60':  2.09350E-03*NI60,
     'NI61':  2.09350E-03*NI61,  'NI62':  2.09350E-03*NI62,
     'NI64':  2.09350E-03*NI64,
     'MO':    2.35390E-04})

mat_ab = SimpleMaterial('AB',
    {'U235': 1.66530E-05,       'U238': 8.20490E-03,
     'O':    1.64130E-02,       'NA':   9.34840E-03,
     'CR50': 3.26720E-03*CR50,  'CR52': 3.26720E-03*CR52,
     'CR53': 3.26720E-03*CR53,  'CR54': 3.26720E-03*CR54,
     'FE54': 1.18960E-02*FE54,  'FE56': 1.18960E-02*FE56,
     'FE57': 1.18960E-02*FE57,  'FE58': 1.18960E-02*FE58,
     'NI58': 2.09350E-03*NI58,  'NI60': 2.09350E-03*NI60,
     'NI61': 2.09350E-03*NI61,  'NI62': 2.09350E-03*NI62,
     'NI64': 2.09350E-03*NI64,
     'MO':   2.35390E-04})

mat_rb = SimpleMaterial('RB',
    {'U235': 2.21490E-05,       'U238': 1.09130E-02,
     'O':    2.19760E-02,       'NA':   7.54820E-03,
     'CR50': 2.84210E-03*CR50,  'CR52': 2.84210E-03*CR52,
     'CR53': 2.84210E-03*CR53,  'CR54': 2.84210E-03*CR54,
     'FE54': 1.03480E-02*FE54,  'FE56': 1.03480E-02*FE56,
     'FE57': 1.03480E-02*FE57,  'FE58': 1.03480E-02*FE58,
     'NI58': 1.82120E-03*NI58,  'NI60': 1.82120E-03*NI60,
     'NI61': 1.82120E-03*NI61,  'NI62': 1.82120E-03*NI62,
     'NI64': 1.82120E-03*NI64,
     'MO':   2.04770E-04})

mat_b4chom = SimpleMaterial('B4CHOM',
    {'CR50': 3.43600E-03*CR50,
      (途中省略)
     'NA':   1.30400E-02})

## JOYO MK-I Movable Reflector
mat_mrf = SimpleMaterial('MRF',
    {'NA':   0.004561, 'CR50':
      (途中省略)
     'NI64': 0.000056})

## JOYO MK-I Fixed Reflector
mat_frf = SimpleMaterial('FRF',
    {'NA':   0.015346,
      (途中省略)
```

（次ページへ続く）

（前ページからの続き）

```
    'NI64': 0.000026})

## JOYO MK-I Axial Reflector
mat_arf = SimpleMaterial('ARF',
    {'NA':   0.009348,
      (途中省略)
     'MO':   0.000235})

## JOYO MK-I NUTORON SOURCE
mat_nus = SimpleMaterial('NUS',
    {'NA':   0.005830,
      (途中省略)
     'MO':   0.000229})

## JOYO MK-I Control Rod PLENUM
mat_plnm = SimpleMaterial('PLNM',
    {'NA':   0.013040,
      (途中省略)
     'MO':   0.000248})

## !JOYO MK-I Control Rod DUSHRAM
mat_dsrm = SimpleMaterial('DSRM',
    {'NA':   0.019404,
      (途中省略)
     'MO':   0.000190})

## JOYO MK-I Control Rod NA
mat_nafl = SimpleMaterial('NAFL',
    {'NA':   0.021507,
      (途中省略)
     'MO':   0.000089})

### Dummy
mat_dum = SimpleMaterial('DUMMY',{'DUM': 0.0})

###
med_fuel   = Medium().madeFromSimpleMaterial('FUEL',   mat_fuel)
med_ab     = Medium().madeFromSimpleMaterial('AB',     mat_ab)
med_rb     = Medium().madeFromSimpleMaterial('RB',     mat_rb)
med_b4chom = Medium().madeFromSimpleMaterial('B4CHOM', mat_ab)
med_mrf    = Medium().madeFromSimpleMaterial('MRF',    mat_mrf)
med_frf    = Medium().madeFromSimpleMaterial('FRF',    mat_frf)
med_arf    = Medium().madeFromSimpleMaterial('ARF',    mat_arf)
med_nus    = Medium().madeFromSimpleMaterial('NUS',    mat_nus)
med_plnm   = Medium().madeFromSimpleMaterial('PLNM',   mat_plnm)
med_dsrm   = Medium().madeFromSimpleMaterial('DSRM',   mat_dsrm)
med_nafl   = Medium().madeFromSimpleMaterial('NAFL',   mat_nafl)
med_dum    = Medium().madeFromSimpleMaterial('DUMMY',  mat_dum)

###
sa_fuel = HexagonalAssembly('FUEL_SA',
    8.15,
    [0.0, 20.0, 60.0, 120.0, 160.0, 180.0],
    [med_arf, med_ab, med_fuel, med_ab, med_arf])

sa_rb = HexagonalAssembly('AB_SA',
    8.15,
    [0.0, 20.0, 160.0, 180.0],
    [med_arf, med_rb, med_arf])

sa_cr = HexagonalAssembly('CR_SA',
    8.15,
    [0.0, 60.0, 95.0, 115.0, 180.0],
```

（次ページへ続く）

（前ページからの続き）

```
    [med_b4chom, med_plnm, med_dsrm, med_nafl])

sa_mrf = HexagonalAssembly('MRF_SA',
    8.15,
    [0.0, 20.0, 60.0, 120.0, 160.0, 180.0],
    [med_arf, med_ab, med_rb, med_ab, med_arf])

sa_frf = HexagonalAssembly('FRF_SA',
    8.15,
    [0.0, 180.0],
    [med_mrf])

sa_nus = HexagonalAssembly('NUS_SA',
    8.15,
    [0.0, 20.0, 60.0, 120.0, 160.0, 180.0],
    [med_arf, med_ab, med_rb, med_ab, med_arf])

sa_dum = HexagonalAssembly('DUMMY_SA',
    8.15,
    [0.0, 180.0],
    [med_dum])

sas = [sa_fuel, sa_rb, sa_cr, sa_mrf, sa_frf, sa_nus]

input_txt = '''  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  1  1
  1  1  1  1  1  1  1  1  1  2  3  3  3  3  3  3  3  3  2  1
  1  1  1  1  1  1  1  1  2  3  3  3  3  3  3  3  3  3  3  2
  1  1  1  1  1  1  1  2  3  3  3  3  3  3  3  3  3  3  3  2
  1  1  1  1  1  1  2  3  3  3  3  4  4  4  4  3  3  3  3  3  2
  1  1  1  1  1  2  3  3  3  4  4  5  5  6  5  5  4  3  3  3  2
  1  1  1  1  2  3  3  3  4  5  6  7  7  7  7  5  4  3  3  3  2
  1  1  1  2  3  3  3  4  6  7  8  9 10  8  7  5  4  3  3  3  2
  1  1  2  3  3  3  4  5  7 10 11 11 11  9  7  6  4  3  3  3  2
  1  1  3  3  3  4  5  7  9 11 12 12 11 10  7  5  4  3  3  3  2
  1  2  3  3  3  4  6  8 11 12 13 12 11  8  6  4  3  3  3  2  1
  1  3  3  3  4  5  7 10 11 12 12 11  9  7  5  4  3  3  3  1  1
  2  3  3  3  4  6  7  9 11 11 11 10  7  5  4  3  3  3  2  1  1
  2  3  3  3  4  5  7  8 10  9  8  7  6  4  3  3  3  2  1  1  1
  2  3  3  3  4  5 14  7  7  7  6  5  4  3  3  3  2  1  1  1  1
  2  3  3  3  4  4  5  6  5  5  4  4  3  3  3  2  1  1  1  1  1
  2  3  3  3  3  4  4  4  4  4  3  3  3  3  2  1  1  1  1  1  1
  2  3  3  3  3  3  3  3  3  3  3  3  3  2  1  1  1  1  1  1  1
  2  3  3  3  3  3  3  3  3  3  3  3  2  1  1  1  1  1  1  1  1
  1  2  3  3  3  3  3  3  3  3  2  1  1  1  1  1  1  1  1  1  1
  1  1  2  2  2  2  2  2  2  1  1  1  1  1  1  1  1  1  1  1  1'''

## Making Map Data from Text Data
hexmap = input_txt.splitlines()
hexmap2 = []
for line in hexmap:
    hexmap2.append(map(int, line.split()))

sas_core = [sa_dum,  sa_frf,  sa_mrf,  sa_rb,   sa_rb, sa_rb,
            sa_fuel, sa_fuel, sa_fuel, sa_fuel, sa_cr, sa_fuel,
            sa_fuel, sa_fuel]
hexmap = hexmap2

joyocore = ConventionalFastReactorCore(hexmap, sas)
joyocore.show()
```

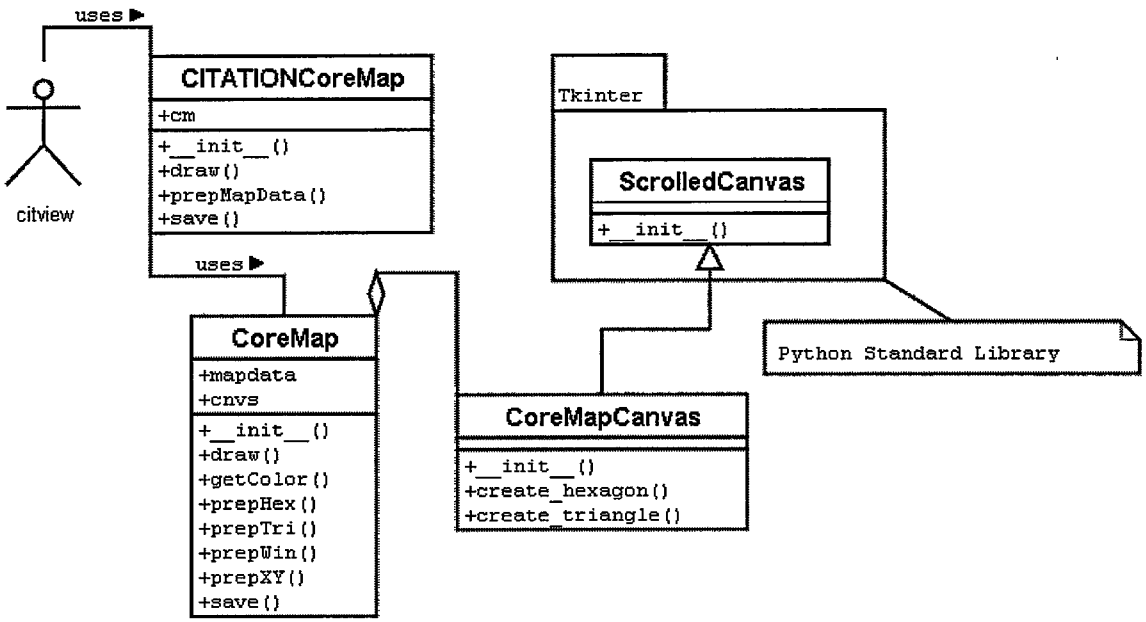


図 4.1.2.3-3 CITATION コード関連 (Neutronics.Geometry.CITAITON.CITATIONutils) モジュールのクラス図

図 4.1.2.3-4 CITATION コード関連モジュールの利用例 (citview.py)

```
#!/usr/local/bin/python

import sys
from Neutronics.Geometry.CITATION.CITATIONutils import citview

filename = sys.argv[1]
try:
    k = int(sys.argv[2]) - 1
except:
    k = 0
try:
    pixelsize = int(sys.argv[3])
except:
    pixelsize = 10

citview(filename, k, pixelsize)
```

図 4.1.2.3-5 CITATION コードの入力データ例（常陽 Mk-I 炉心）

```
（以上省略）
005
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10  9  9  9 10 10 10  9  9  9 10 10 10  9  9  9 10 10 10 10 10 10 10 10
 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  9  9  9
  9  9  9 14 14 14  9  9  9 14 14 14  9  9  9  9  9  9 10 10 10 10 10 10
 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  9  9  9  9  9  9 14
 14 14 14 14 14 14 14 14 14 14 14 14 14 14  9  9  9  9  9  9 10 10 10 10
 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10  9  9  9  9  9  9 14 14 14 14 14
 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  9  9  9  9  9  9 10 10
 10 10 10 10
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10
 10 10 10 10 10 10 10 10 10  9  9  9  9  9  9 14 14 14 14 14 14 14 14 14
 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14  9  9  9  9  9  9
 10 10 10 10
（以下省略）
```

このケースでは、同様のデータが約 2,000 行続く。

図 4.1.2.3-6　citview.py の実行結果

```
$ python citview.py input/cit_joyomk1.inp
filename = input/cit_joyomk1.inp
k = 0
pixelsize = 10
Now trying to draw on the display...
```

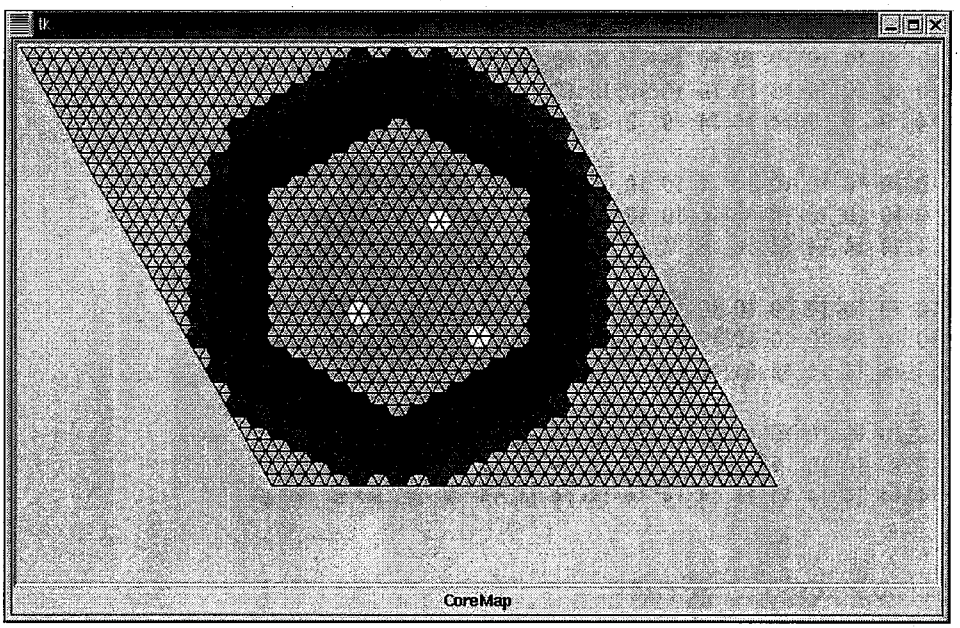


図 4.1.2.4-1 XML 形式に変換された JFS-3 ライブラリ

```
<?xml version="1.0" encoding="iso-8859-1"?>
 <jfs3>
 <record1>
   <no_of_nuclide varname="lnmax">
 383
   </no_of_nuclide>
   <no_of_energy_group varname="imax">
 70
   </no_of_energy_group>
   <no_of_fission_spectrum varname="mxchi">
 5
   </no_of_fission_spectrum>
   <no_of_reaction_1d_data varname="mxr1d">
 8
   </no_of_reaction_1d_data>
   <matrix_specification varname="mtxr23">
 2
   </matrix_specification>
   <max_LD_el varname="mxdnse">
 10
   </max_LD_el>
   <max_LD_in varname="mxdnsi">
 70
   </max_LD_in>
   <max_LD_n2n varname="mxdns2">
 70
   </max_LD_n2n>
   <max_LA_el varname="mxdwne">
 70
   </max_LA_el>
   <max_LA_in varname="mxdwni">
 47
   </max_LA_in>
   <max_LA_n2n varname="mxdwn2">
 7
   </max_LA_n2n>
   <no_of_reaction_ftable varname="mxreac">
 6
   </no_of_reaction_ftable>
   <no_of_sig0values varname="mxsigo">
 8
   </no_of_sig0values>
   <max_of_temperatures varname="mxtemp">
 4
   </max_of_temperatures>
   <max_of_rvalues varname="mxr">
 4
   </max_of_rvalues>
   <position_of_hydrogen varname="iswh">
 1
   </position_of_hydrogen>
 </record1>
 <record2>
```

（次ページへ続く）

（前ページからの続き）

```
  <nuclide_code varname="ncodel">
928
  </nuclide_code>
  <energy_boundary varname="enbnd">
 10000000.   7788010.   6065310.   4723670.   3678790.   2865050.   2231300.
 1737740.   1353350.   1053990.   820850.   639279.   497871.   387742.   301974.
 235178.   183156.   142642.   111090.   86517.   67379.5   52475.1992   40867.6992
 31827.8008   24787.5   19304.5   15034.4004   11708.7998   9118.82031   7101.74023
 5530.83984   4307.41992   3354.62988   2612.59009   2034.68005   1584.60999
 1234.09998   961.117004   748.518005   582.947021   453.998993   353.575012
 275.36499   214.453995   167.016998   130.072998   101.301003   78.8932037
 61.4421005   47.8512001   37.2664986   29.0231991   22.6033001   17.6035004
 13.7096004   10.677   8.31529045   6.47594976   5.04347992   3.92786002
 3.05902004   2.38237   1.85538995   1.44498003   1.12535   0.876425028
 0.682560027   0.531578004   0.413994014   0.322418988   9.99999975E-06
  </energy_boundary>
  <code_given_fission_spectrum varname="ncodx">
928
  </code_given_fission_spectrum>
  <fission_spectrum varname="chi">
 0.00399857014   0.0142556001   0.0347128995   0.0633480027   0.0932677016
 0.115322001   0.122942999   0.11654   0.101461001   0.0846792981   0.0672708005
 0.0515503995   0.0382247008   0.0276085995   0.0196286999   0.0138100004
 0.00964485016   0.00667804992   0.00466354983   0.00323297991   0.00223293994
 0.00153990998   0.00106087001   0.000730325992   0.000502320996   0.000345006003
 0.000236533   0.000163182005   0.000112640002   7.74387008E-05   5.32295016E-05
 3.6580801E-05   2.51284E-05   1.72618002E-05   1.18451999E-05   8.12566032E-06
 5.58016018E-06   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
  </fission_spectrum>
  <atomic_weight varname="aw">
 238.05098
  </atomic_weight>
  <max_sink_group varname="ld">
2 51 69
  </max_sink_group>
  <lowest_energy_group varname="la">
70 22 2
  </lowest_energy_group>
  <ftable_specification varname="tab">
 0.100000001   1.   10.   100.   1000.   10000.   100000.   1000000.
  </ftable_specification>
</record2>
<record3>
  <temperature varname="ft">
 300.   800.   2100.   4500.
```

（以下省略）

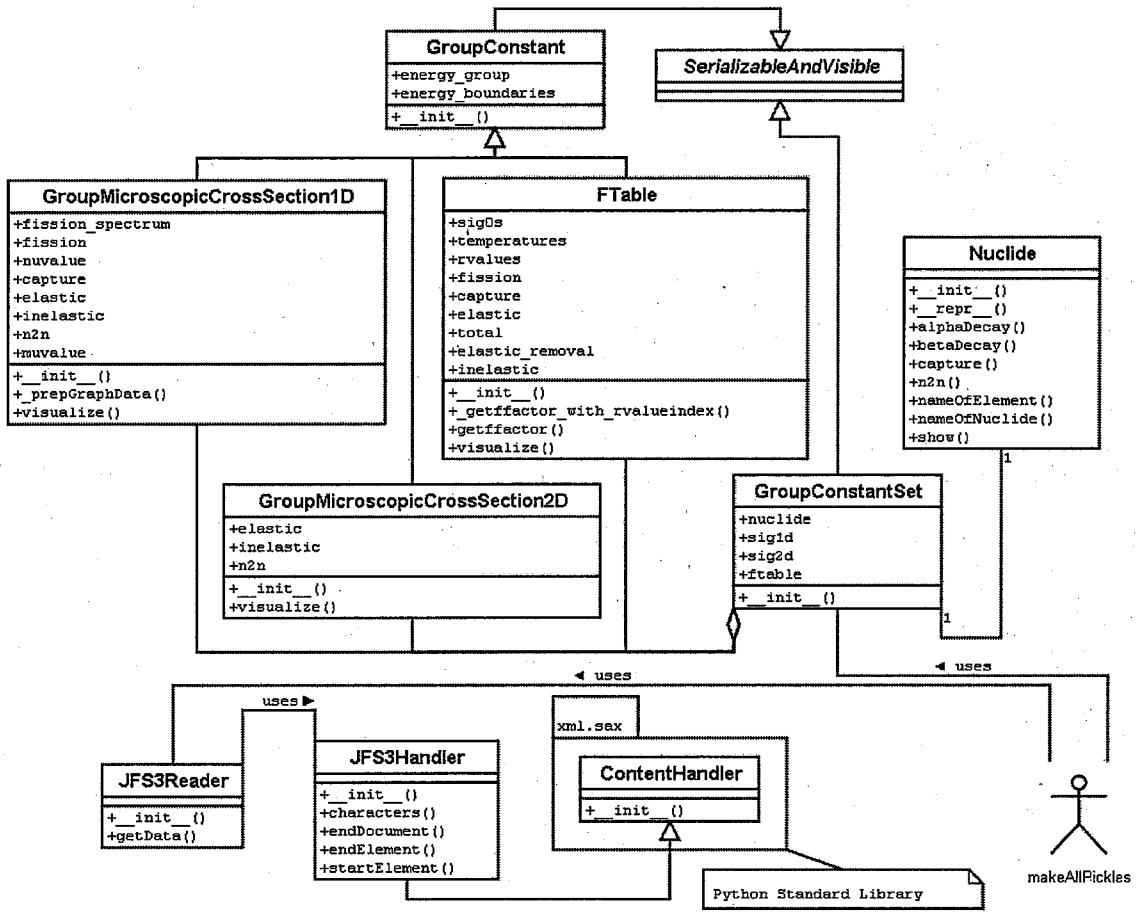


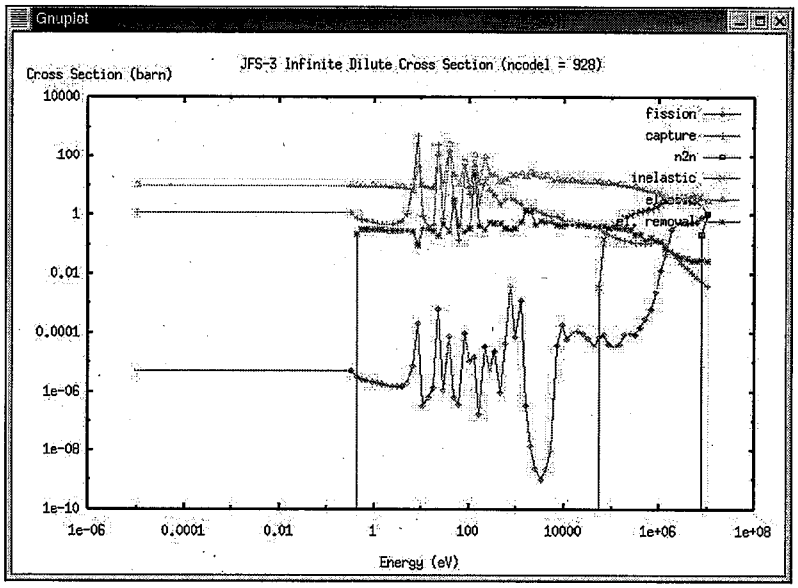
図 4.1.2.4-2 JFS-3 ライブラリ関連 (Neutronics.NuclearData.JFS3.JFS3utils) モジュールのクラス図

図 4.1.2.4-3　JFS-3 ライブラリの読み込みと自己遮蔽因子の計算（jfs3.py）

```
#!/usr/bin/env python

from Neutronics.NuclearData.JFS3.JFS3utils import *

# Pickle 形式の JFS-3 ライブラリの読み込み
gcset = GroupConstantSet().load("../../NeutronicsData/PKL_JFS3J33.ORG/Pu-239.pkl")

# 引数の確認
help(gcset.ftable.getffactor)

# f-table の内挿
print gcset.ftable.getffactor(30, 100.0, 293.0, 1.0)
print gcset.ftable.getffactor(30, 100.0, 300.0, 1.0)
print gcset.ftable.getffactor(30, 100.0, 4450.0, 1.0)
print gcset.ftable.getffactor(30, 100.0, 4500.0, 1.0)
print gcset.ftable.getffactor(30, 100.0, 4550.0, 1.0)

# 一次元断面積データの可視化
gcset.sig1d.visualize()
```

図 4.1.2.4-4　jfs3.py の実行結果

```
Loading GroupConstantSet object from Pickled file...
Help on method getffactor in module Neutronics.NuclearData.JFS3.JFS3utils:

getffactor(self, group, sig0, temp, rvalue=None) method of
 Neutronics.NuclearData.JFS3.JFS3utils.FTable instance

0.968286846626
0.97832799
0.998006721766
0.998021007
0.983099434141
```



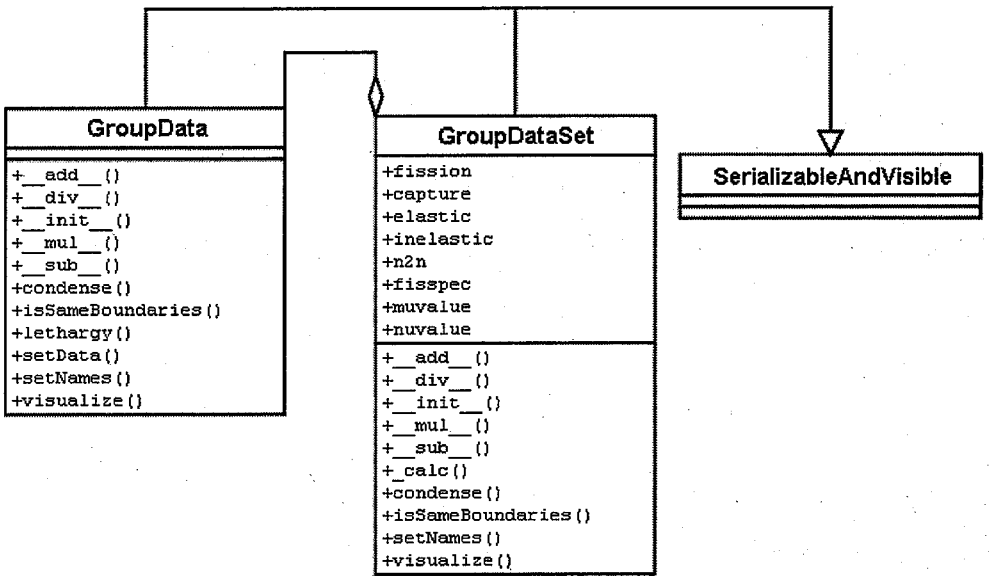


図 4.1.2.5-1　群定数（Neutronics.GroupData）モジュールのクラス図

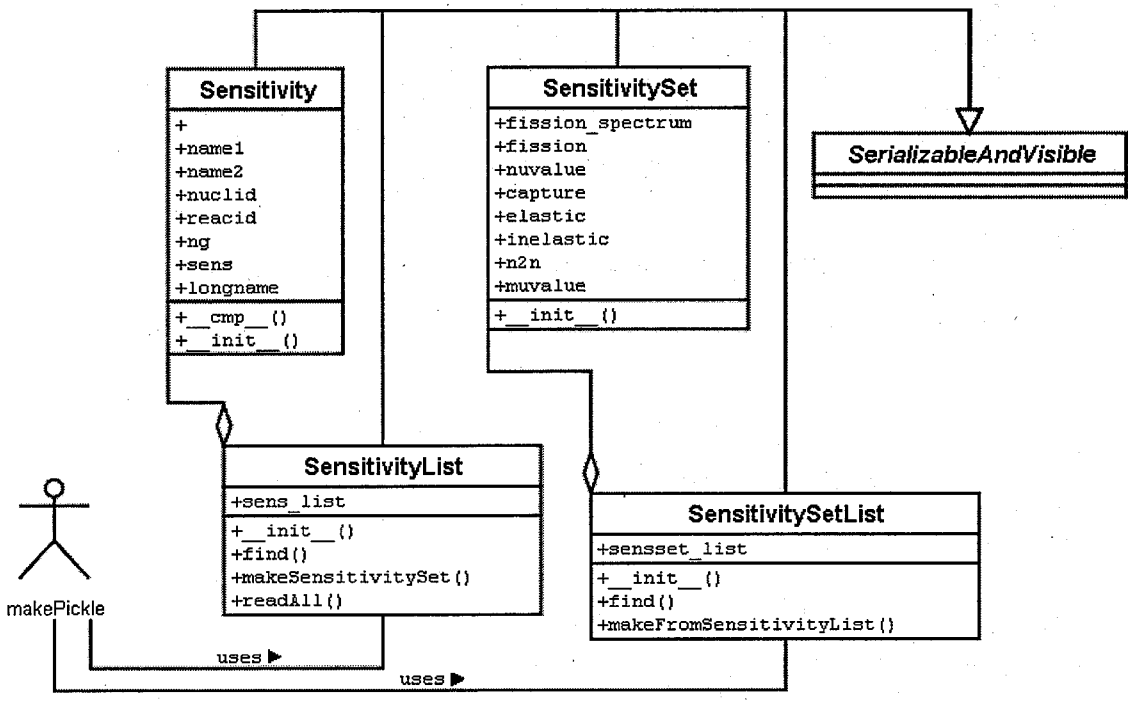


図 4.1.2.5-2　SAGEP コード関連（Neutronics.Sensitivity.SAGEP.SAGEPutils）モジュールのクラス図

図 4.1.2.5-3　感度解析用モジュールの利用例 (sensana.py)

```
#!/usr/local/bin/python

from Numeric import *

from Neutronics.NuclearData.JFS3.JFS3utils    import *
from Neutronics.Sensitivity.SAGEP.SAGEPutils import *
from Neutronics.GroupData import GroupData, GroupDataSet
from Neutronics.Nuclide   import Nuclide
from Neutronics.Wrappers  import JFS3toGroupDataSet, SAGEPtoGroupDataSet
from Neutronics.GroupData import GroupData
from Neutronics.NuclearData.JFS3.JFS3utils import jfs70g

import os

def prepare_weight():
    flux = [
     4.0103E-08,1.3504E-07,3.0969E-07,5.0693E-07,7.8035E-07,
     1.1850E-06,1.2671E-06,1.5619E-06,1.6218E-06,1.7835E-06,
     2.8248E-06,3.7527E-06,2.6168E-06,3.6886E-06,3.7234E-06,
     3.6925E-06,3.8310E-06,4.4528E-06,3.7315E-06,3.9523E-06,
     3.0603E-06,3.3333E-06,2.7226E-06,2.7866E-06,2.5821E-06,
     1.7800E-06,1.9029E-06,1.6076E-06,1.0310E-06,1.0244E-06,
     8.2157E-07,4.9685E-07,1.1921E-07,4.0968E-07,7.0067E-07,
     6.6098E-07,4.8942E-07,3.6947E-07,2.7122E-07,1.7440E-07,
     1.0442E-07,7.5198E-08,4.7864E-08,2.8005E-08,1.6257E-08,
     8.5142E-09,4.6061E-09,2.0282E-09,1.0516E-09,4.8364E-10,
     2.6648E-10,2.7149E-10,7.0645E-11,4.8801E-11,3.6876E-11,
     4.8575E-11,1.5577E-11,1.5155E-11,2.2814E-11,2.6987E-11,
     2.3478E-11,1.9939E-11,1.4593E-11,5.0716E-12,1.6828E-13,
     9.2206E-13,7.9237E-13,3.3291E-13,5.2236E-14,7.0462E-14,
    ]
    weight = GroupData(70, jfs70g,
                       "Neutoron flux in ZPPR-9 inner core clculated by JENDL-3.2")
    weight.setData(flux)
    return weight

def main(nucl):
    ### Preparing Weighting Function
    weight = prepare_weight()
    weight = weight * weight.lethargy()
    condensed = GroupData(18, jfs18g)
    ## weight.visualize()

    ### Loading JFS3 Libraries
    datadir = "../../NeutronicsData/"
    jfs3j32 = GroupMicroscopicCrossSection1D().load(
        datadir + "PKL_JFS3J32R.Y0202.N295/"+nucl+".pkl")
    jfs3j33 = GroupMicroscopicCrossSection1D().load(
        datadir + "PKL_JFS3J33.ORG/"+nucl+".pkl")

    j32_70g = JFS3toGroupDataSet(jfs3j32)
    j33_70g = JFS3toGroupDataSet(jfs3j33)

    j32_70g.setNames(nucl+" of JFS3-J3.2",
```

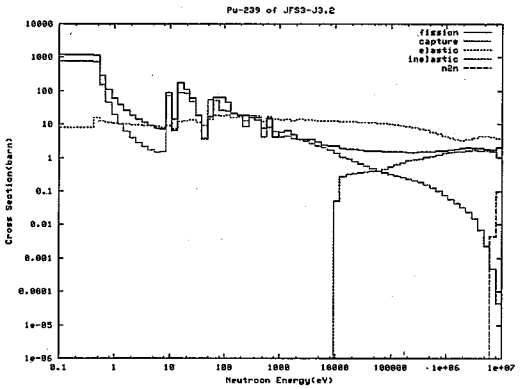
（次ページへ続く）

（前ページからの続き）

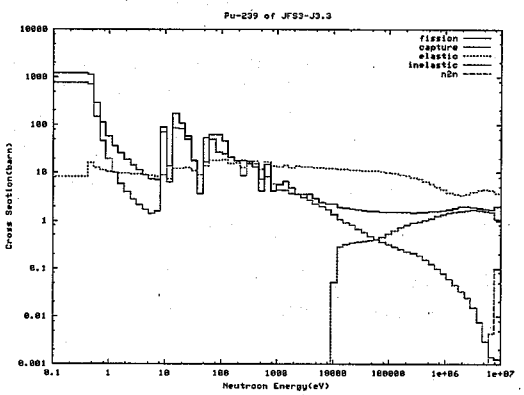
```
                    "Neutroon Energy", "Cross Section",
                    "eV",              "barn")
    j33_70g.setNames(nucl+" of JFS3-J3.3",
                    "Neutroon Energy", "Cross Section",
                    "eV",              "barn")

    ### Group Conden(前ページからの続き)
    j32_18g = j32_70g.condense(weight, condensed)
    j33_18g = j33_70g.condense(weight, condensed)

    ### Loading Sensitivity Coefficients
    sensset_list = SensitivitySetList().load(datadir+"PKL_SNS/ZPPR09.SNS.pkl")
    sensset = sensset_list.find(
                "ZPPR-09", "KEFF", idOfJFS(Nuclide(nucl))
              )
    sens_18g = SAGEPtoGroupDataSet(sensset)

    sens_18g.setNames("ZPPR-9 Keff Sensitivity of "+nucl,
                     "Neutroon Energy", "Sensitivity Coefficient",
                     "eV",              "abs.")

    ### Calculation of Library Difference
    diff = (j33_18g - j32_18g) / j32_18g

    ### Calculation of Library Effect
    effect = diff * sens_18g

    ### Visualization
    diff.setNames("Difference of "+nucl+" between JENDL3.3 and JENDL3.2",
                  "Neutroon Energy", "Difference",
                  "eV",              "abs.")
    effect.setNames("Library Effect of "+nucl+" between JENDL3.3 and JENDL3.2",
                    "Neutroon Energy", "Library Effect",
                    "eV",              "abs.")
    outputdir = "./output/"

    try:
        os.mkdir(outputdir)
    except OSError:
        pass

    j32_70g.visualize(filename=outputdir+"j32_70g.png", terminal="png")
    j33_70g.visualize(filename=outputdir+"j33_70g.png", terminal="png")
    j32_18g.visualize(filename=outputdir+"j32_18g.png", terminal="png")
    j33_18g.visualize(filename=outputdir+"j33_18g.png", terminal="png")
    sens_18g.visualize(logscaley=0,
                       filename=outputdir+"sens_18g.png", terminal="png")
    diff.visualize(logscaley=0, yrange="[-0.2:0.2]",
                   filename=outputdir+"diff.png", terminal="png")
    effect.visualize(logscaley=0,
                     filename=outputdir+"effect.png", terminal="png")

if __name__ == "__main__":
    main("Pu-239")
```

図 4.1.2.5-4　sensana.py の実行結果

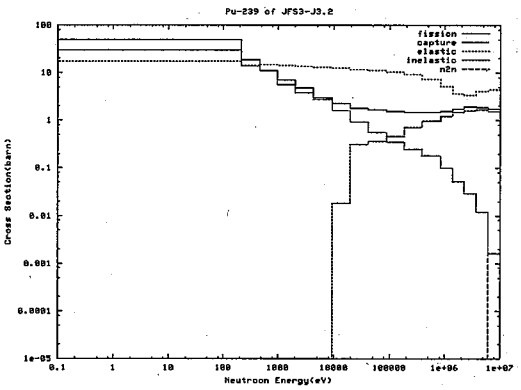
```
$ python sensana.py
Loading GroupMicroscopicCrossSection1D object from Pickled file...
Loading GroupMicroscopicCrossSection1D object from Pickled file...
Loading SensitivitySetList object from Pickled file...
```



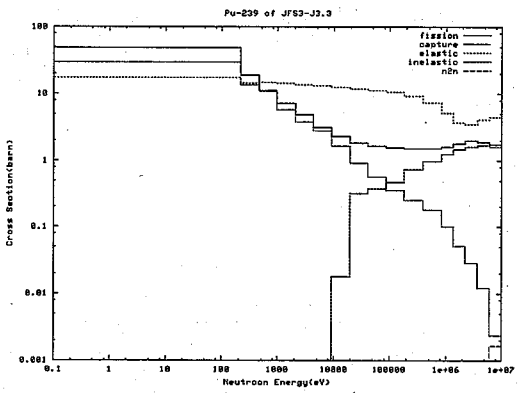
JFS-3-J3.2 の Pu-239 核反応断面積（70 群）



JFS-3-J3.3 の Pu-239 核反応断面積（70 群）



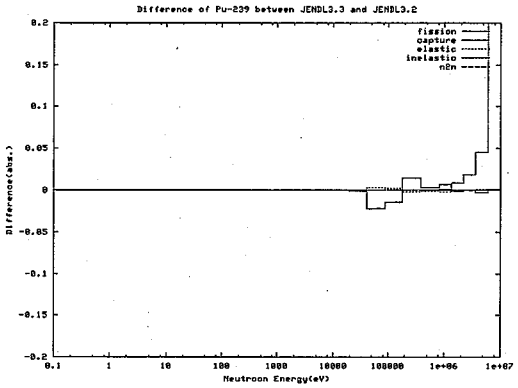
JFS-3-J3.2 の Pu-239 核反応断面積（18 群）



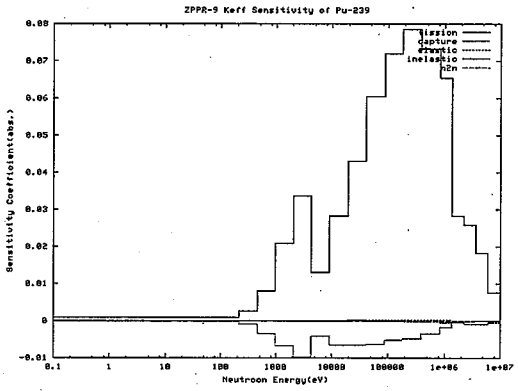
JFS-3-J3.3 の Pu-239 核反応断面積（18 群）

（次ページへ続く）

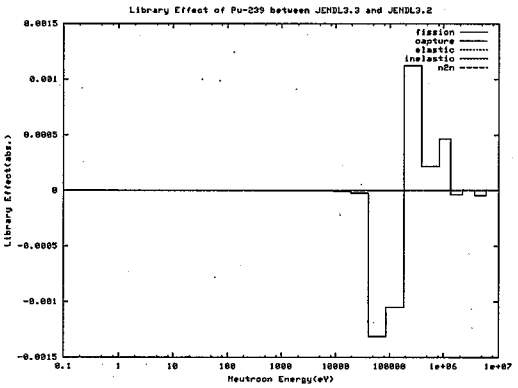
（前ページからの続き）



JFS-3-J3.2 と JFS-3-3.3 の断面積差（18 群）



Pu-239 の ZPPR-9 臨界性に対する感度係数（18 群）



ライブラリ効果（JFS-3-J3.2 と JFS-3-J3.3）

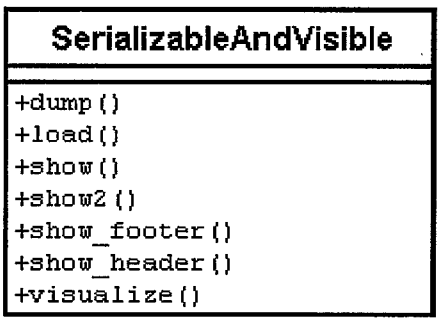


図 4.1.2.6-1　基本データ用 Mixin (Neutronics.Mixin) のクラス図

4.2 Pythonによるデザインパターン

オブジェクト指向設計を行う際のパターンを、Eric Gamma, Richard Helm, Ralph Johnson, John Vlissidesの4人（GoF: Gang of Fourと呼ばれる）が、23の「デザインパターン」という形でまとめた[4-4]。これはGoFの「デザインパターン」として有名であり、オブジェクト指向プログラミングをする際の基本的な技術となっている。もともとはC++での経験がまとめられたものであるが、オブジェクト指向言語で一般的に利用可能な知見である。

オブジェクト指向プログラミングの特徴を知る上でも、デザインパターンがどのように実装されるかは良いサンプルとなると考えられる。Pythonによるデザインパターンについてはいろいろな文献[4-15, 4-16]で取り上げられているが、23パターンすべてをPythonで説明しているものは存在しない。

一方、Javaを用いたデザインパターンの実装は数多くの文献が存在するが、ここでは、より入門的な書籍として、結城浩氏により日本語で書かれた「Java言語で学ぶデザインパターン入門」[4-17]を取り上げ、ここで紹介されているサンプルプログラムをPythonで書き直してみることで、Pythonによるオブジェクト指向プログラミングの詳細と、次世代解析システムへの応用の可能性について検討する。なお、この文献には、23のパターンすべてについて、Javaで書かれた実行可能なサンプルプログラムが紹介されている。この内、ここでは、前半の9パターンについて検討した結果を以下にまとめる。

4.2.1 Iteratorパターン

Iteratorパターンとは、「何かがたくさん集まっているときに、それを順番に指し示していき、全体をスキャンしていく処理を行うもの」である。

図4.2.1-1に、Javaのサンプルプログラムを Pythonに直接翻訳したものを示す。なお、Pythonでは言語仕様として、Iteratorが装備されているので、図4.2.1-1のような書き方はPythonでは冗長であり、もっと簡単に書くことができる。Pythonのイテレータ機能を使って書き直すと図4.2.1-2のようになる。Pythonの組込型であるリストを使っているので、BookShelfの長さを可変にするのも簡単である。

応用例：

Iteratorという名前からも、中性子輸送方程式の反復解法などにも適用できると考えられる。拡散近似、Sn法、ノード法などに依らず、内側反復、外側反復という概念をIteratorパターンで一般的に表現できると考えられる。

例えば、プロトタイプ中のDiffusion1D.pyで、nextgen()というメソッドを用意している

が、これは一種の Iterator パターンである。

4.2.2 Adapter パターン

Adapter パターンとは、「既に提供されているものがそのまま使えない時に、必要な形に変換してから利用するために、『既に提供されているもの』と『必要なもの』の間のずれを埋める」ようなデザインパターンである。

文献 4-17 には、継承を使ったものと委譲を使ったものの２種類のサンプルプログラムが示されており、両方とも Python に書き換えてみた。図 4.2.2-1 に継承を用いたもの、図 4.2.2-2 に委譲を用いたものを示す。両者の差は、PrintBanner クラスの__init__()メソッドにある。委譲の方では、このクラスが、self.banner という Banner クラスのインスタンスを保持しておき、すべて、この self.banner に操作をさせているのが分かる。

応用例：

既存の解析コードに変更を加えずに次世代解析システムに組み込む場合、Adapter パターンが適用できると考えられる。

4.2.3 Template Method パターン

スーパークラスで処理の枠組みを定め、サブクラスでその具体的内容を定めるようなデザインパターンを、Template Method パターンという。このようにすることで、アルゴリズムの共有化が可能となる。また、スーパークラスとサブクラスを同一視することが可能となり、サブクラスの型を特定せずにプログラムを記述することが可能となる。

図 4.2.3-1 に Python による実装を示す。AbstractDisplay クラスでは、「open して、print_を５回繰り返した後に、close する」という抽象化された手続きを定義してあるだけであり、CharDisplay、StringDisplay クラスで、open、print_、close の具体的な操作が定義されている。

なお、Python では、print は予約語であるため、print というメソッド名を使うことはできない。このため、「print_」というメソッド名にした。図 4.2.3-2 に実行結果を示す。「open して、print_を５回繰り返した後に、close する」という手続きを使って、CharDisplay クラスは「<<」と「>>」で囲まれた文字列を、StringDisplay クラスは矩形で囲まれた文字列を表示する。

応用例：

アルゴリズムの共有化などに使える可能性がある。例えば、外側反復と内側反復のアルゴリズムを記述したスーパークラスと、拡散、輸送 Sn、ノード法などのサブクラスといっ

た形で利用できる可能性がある。

4.2.4 Factory Method パターン

Template Method パターンを、インスタンス生成の場面に適用したものを、Factory Method パターンという。Factory Method パターンでは、インスタンスの生成方法をスーパークラスで定めるが、具体的なクラス名までは定めない。これにより、インスタンス生成のための枠組みと実際のインスタンス生成のクラスを分けることが可能となる。

図 4.2.4-1、図 4.2.4-2 に Python による実装を示す。このパターンの特徴は、フレームワーク（ここでは、framework.py で定義されている）を全く変更することなく、全く別の製品（IDCard、IDCardFactory クラスに相当するもの）を作ることができる。言い換えれば、この例の場合、「作成して（createProduct）登録して（registerProduct）、使う（use）もの」であれば、どのようなものにでも、このフレームワークを利用することができるということである。

このようなことが可能になるのは、framework.py では、idcard.py を import しておらず、idcard.py に関する情報は全く記述されていないからである。

応用例：

群定数化された核分裂断面積、平均散乱角余弦、感度係数、断面積誤差などは、すべて、中性子エネルギーで離散化された物理量という特徴を持つ。例えば、これらのデータに関して、エネルギー群数を少なくするための群縮約の操作が考えられるが、それぞれ、群縮約の方法は異なる。このようなデータのインスタンスを作成する場合に、Factory Method パターンが利用できると考えられる。

4.2.5 Singleton パターン

Singleton パターンとは、指定したクラスのインスタンスが絶対に１つしか存在しないことを保証するためのパターンである。インスタンスが１つしか必要ない場合、インスタンスが１つしか存在しないようにすることで、インスタンスどうしが影響しあって、プログラムエラーを発生させることを避けることができる。

Java のサンプルプログラムでは、クラスの静的なインスタンスを使って Singleton パターンを実現しているが、Python は動的な言語であるため、同じ方法で直接翻訳しようとしてもうまく動作しない。このため、文献 4-16 で紹介されていた Nested Scope（この例では、クラスの中にクラスがあるが、Python ではこのようなことが許されており、Nested Scope と呼ばれている）を使う方法で、Java のサンプルプログラムを Python に変換した。

図 4.2.5-1、図 4.2.5-2 に、それぞれ、Python による実装、実行結果を示す。実行結果の

「<__main__.__Singleton instance at 0x8157eec>」は Singleton クラスのインスタンスの情報を示しており、最後の数字が一致していることから、生成された２つの Singleton インスタンスは同一であることが確認できる。

応用例：

Singleton パターンは、実際のシステムにひとつしかないものを表す場合に利用される。例えば、遠隔地にある計算機を表すインスタンスはひとつあれば十分である。逆に、実際にひとつしか存在しないものを表すインスタンスが複数存在しうるのは問題の原因になると考えられる。このような場合に Singleton パターンが利用できる。

4.2.6 Prototype パターン

Prototype パターンとは、(1)種類が多すぎてクラスにまとめられない場合、(2)クラスからのインスタンス生成が難しい場合、(3)フレームワークと生成するインスタンスを分けたい場合などに、クラスからインスタンスを生成するのではなく、インスタンスから別のインスタンスを作成するデザインパターンである。

図 4.2.6-1、図 4.2.6-2 に Python による実装を示す。この例では、文字列に、「~」、「*」、「/」等の文字で下線を引いたり、枠を付けたりして装飾している。装飾するための文字ごとに、新たなクラスを定義することも可能であるが、装飾するための文字が多い場合、ソースプログラムの管理が難しくなると考えられる。

この例では、新たなクラスを定義することなく、「装飾するための文字」、「下線付け（UnderlinePen）」、「枠付け（MessageBox）」を自由に組み合わせた製品（Product）を作ることが可能となっている。

応用例：

複雑な炉心体系データなどを取り扱う場合、繰り返し同じ形状を利用する可能性は高い。例えば、燃料集合体の形状は同じものとして取り扱える。このような場合に、Prototype パターンを使うことが可能である。

4.2.7 Builder パターン

Builder パターンとは、複雑な構造を持ったインスタンスを一気に完成させるのが難しい場合、まず全体を構成している各部分を作り、段階を踏んでくみ上げていくデザインパターンである。このパターンで利用される Director クラスは、Builder クラスのサブクラスについて全く知らないため、Builder クラスのサブクラスのインスタンスを自由に入れ替えることが可能となっている。

図 4.2.7-1 に Python による実装を示す。この例では、以下のような構造を持った文書を、プレーンテキスト形式、及び、HTML 形式に整形している。

- タイトルをひとつ含む
- 文字列をいくつか含む
- 箇条書きの項目をいくつか含む

Director クラスは Builder クラスのメソッドを使って文書を整形する。しかしながら、Director クラスは、自分が使っているのは、TextBuilder なのか、HTMLBuilder なのかは全く意識する必要がない（判定文がない）。このため、他の形式に対応した Builder を作る場合でも、既存のソースには全く手を入れる必要がなくなる。また、形式を変更したいと思った場合には、Director に渡す Builder インスタンスを交換するだけで良い。

応用例：

Prototype パターンと同様に、複雑な体系データのインスタンスの作成などに使える可能性がある。また、核反応断面積データのセットは、多くの反応の組み合わせで成り立っているので、複雑な構造を持ったインスタンスになると考えられる。また、核種によっては、核分裂断面積が不要であったり、解析対象によって必要な反応が追加されたりするので、Builder パターンを適用することで柔軟性を持たせることができる。

4.2.8 Abstract Factory パターン

Abstract Factory パターンは名前のとおり、「抽象的な工場」であり、抽象的な部品を組み合わせて抽象的な製品を作るパターンである。部品の具体的な実装には注目せず、インタフェース（API）に注目して、部品を組み立て、製品をまとめるパターンである。

図 4.2.8-1～図 4.2.8-4 に Python による実装を示す。ここでは、ホームページへのリンクを示すホームページ（HTML ファイル）を、リスト形式と表形式で生成するプログラムを例にしている。実行結果を図 4.2.8-5、図 4.2.8-6 に示す。

この例では、ホームページへのリンクは、Item、Link、Tray クラスで表現されており、これらをまとめた HTML ファイル全体を Page クラスとなっている。ホームページ（HTML ファイル）の生成という作業が、「工場（Factory クラス）が、部品（Item、Link、Tray クラス）を使って、製品（Page クラス）を作成する」という形で実現されている。更に、工場が作成するのは、ホームページということであって、どのような形式（リスト形式、表形式）かは定義されていない。このような意味で、抽象化された工場と呼ばれる。

これに対して、ここでは、リスト形式のホームページを作成する工場（ListFactory クラス）と表形式のホームページを作成する工場（TableFactory クラス）が、具体的な工場と

いうことになる。

このようにすることで、具体的な工場を追加したり、変更したりする場合、他の部分（抽象的な工場、main 等）を変更する必要がなくなる。一方で、具体的な工場を追加するのは簡単であるが、部品を追加する場合は修正が困難になる点には注意する必要がある。

応用例：

このサンプルと同じように、解析結果を表示する方法に対してこの Abstract Factory パターンが使える可能性がある。同じ解析結果であっても、ユーザーによって見たい部分や方法が異なるということはよくあることである。解析結果とは、完全に分離した形で、テキスト、HTML、PDF、３次元画像等に対応することが可能となる。また、新たな形式に対応させたい場合、新しい工場を作成するだけで良い。

4.2.9 Bridge パターン

「機能のクラス階層」と「実装のクラス階層」を分け、２つの階層を橋渡しするパターンを Bridge パターンという。２つのクラス階層に分けておくことで、それぞれのクラス階層を独立に拡張することが可能となる。

図 4.2.9-1 に Python による実装例を示す。ここでは、Display と CountDisplay が機能のクラス階層を、DisplayImp と StringDisplayImp が実装のクラス階層を形成している。

このようにすることで得られる利点は、機能のクラスや実装のクラスを追加した場合のことを考えるとよく分かる。例えば、Display クラスしか存在しない状態から、新たな機能が必要となり CountDisplay クラスを追加した場合のことを考えると、CountDisplay クラスは、既存の StringDisplayImpl の実装をそのまま利用できることが分かる。他に多くの実装が存在する場合、この利点は非常に優れたものとなる。

応用例：

例えば、Python では、OS 依存性の高いファイル操作なども、Windows、Unix、Macintosh 等の OS に関係なく統一されたインターフェイスが用意されているが、これは Bridge パターンを利用したものである。次世代解析システムは、計算機に依存しないことが望まれており、機種依存性のある実装が必要な場合などに Bridge パターンが利用できると考えられる。

4.2.10 まとめ

文献「Java 言語で学ぶデザインパターン入門」[4-17]を取り上げ、ここで紹介されているJavaのサンプルプログラムをPythonで書き直してみることで、Pythonによるオブジェクト指向プログラミングの詳細と、次世代解析システムへの応用の可能性について検討した。Pythonを使って、デザインパターンを応用した、実際に動作するプログラムを作成することにより、オブジェクト指向プログラミング言語を利用することによって得られる再利用性、拡張性等を示すことができた。

また、Javaのプログラムを Pythonに変換することで、両者の比較をすることができた。Java、Pythonはともにオブジェクト指向言語であり、ある程度、両言語の定型句が分かれば、JavaからPythonへの変換はそれほど難しいものではなかった。著者の印象としては、デザインパターンを実装する上では、両者の間に、機能的な大きな差はないと考える。

ただし、一般的にいって、Python では、型付けが弱く型宣言が不要であるので、Javaに比べてPythonの方がプログラムが短くなり、すっきりとするという印象を受けた。逆に、Python には abstract、private、public、protected 等のキーワードがないので、プログラマの意図を明確にできるという点では型付けの強い Java の方が良い場合もあると感じた。ただし、Python でも、２つのアンダーバーから始まるメソッド名や変数名を使うことで、private なメソッド・変数として取り扱われる（外部からアクセスしようとするとエラーになる)。ここに示したPythonのプログラムでは、privateであることを示すために、２つのアンダーバーから始まるメソッド名・変数名を使った。なお、Python には protected に相当する機能はないため、便宜的に、１つのアンダーバーで始まるメソッド名・変数名を使っている。

今回、23 種類あるデザインパターンの内、9 種類について検討したが、この検討結果から、解析コードの設計においてもデザインパターンの適用が可能であると考える。今後、残りの 14 種類についても調査し、次世代解析システムの設計への適用可能性についての検討を継続する予定である。

―― 図 4.2.1-1 Java サンプルの直訳的な Iterator パターンの実装 (Iterator.py) ―

```
#!/usr/bin/env python

class Aggregate:
    def iterator(self):
        raise NotImplementedError

class Iterator:
    def hasNext(self):
        raise NotImplementedError

    def next(self):
        raise NotImplementedError

class Book:
    def __init__(self, name=""):
        self.__name = name

    def getName(self):
        return self.__name

class BookShelf(Aggregate):
    def __init__(self, maxsize):
        self.__books = [Book() for i in range(maxsize)]
        self.__last = 0

    def getBookAt(self, index):
        return self.__books[index]

    def appendBook(self, book):
        self.__books[self.__last] = book
        self.__last += 1

    def getLength(self):
        return self.__last

    def iterator(self):
        return BookShelfIterator(self)

class BookShelfIterator(Iterator):
    def __init__(self, bookShelf):
        self.__bookShelf = bookShelf
        self.__index = 0

    def hasNext(self):
        if self.__index < self.__bookShelf.getLength():
            return 1
        else:
            return 0

    def next(self):
        book = self.__bookShelf.getBookAt(self.__index)
        self.__index += 1
        return book
```

(次ページへ続く)

（前ページからの続き）

```
def main():
    bookShelf = BookShelf(4)
    bookShelf.appendBook(Book("Around the World in 80 Days"))
    bookShelf.appendBook(Book("Bible"))
    bookShelf.appendBook(Book("Cinderella"))
    bookShelf.appendBook(Book("Daddy-Long-Legs"))
    it = bookShelf.iterator()
    while (it.hasNext()):
        book = it.next()
        print "" +  book.getName()


if __name__ == "__main__":
    main()
```

図 4.2.1-2 Python の機能を用いた Iterator パターンの実装(Iterator2.py)

```
#!/usr/bin/env python

class Book:
    def __init__(self, name=""):
        self.__name = name

    def getName(self):
        return self.__name

class BookShelf:
    def __init__(self):
        self.__books = []

    def __len__(self):
        return len(self.__books)

    def __iter__(self):
        return iter(self.__books)

    def appendBook(self, book):
        if isinstance(book, Book):
            self.__books.append(book)
        else:
            raise TypeError

def main():
    bookShelf = BookShelf()
    print "bookShelf's length is", len(bookShelf)
    bookShelf.appendBook(Book("Around the World in 80 Days"))
    bookShelf.appendBook(Book("Bible"))
    bookShelf.appendBook(Book("Cinderella"))
    bookShelf.appendBook(Book("Daddy-Long-Legs"))
    print "bookShelf's length is", len(bookShelf)
    for book in bookShelf:
        print "" +  book.getName()

if __name__ == "__main__":
    main()
```

図 4.2.1-3　Iterator.py の実行結果

```
$ python Iterator.py
Around the World in 80 Days
Bible
Cinderella
Daddy-Long-Legs
```

図 4.2.1-4　Iterator2.py の実行結果

```
$ python Iterator2.py
bookShelf's length is 0
bookShelf's length is 4
Around the World in 80 Days
Bible
Cinderella
Daddy-Long-Legs
```

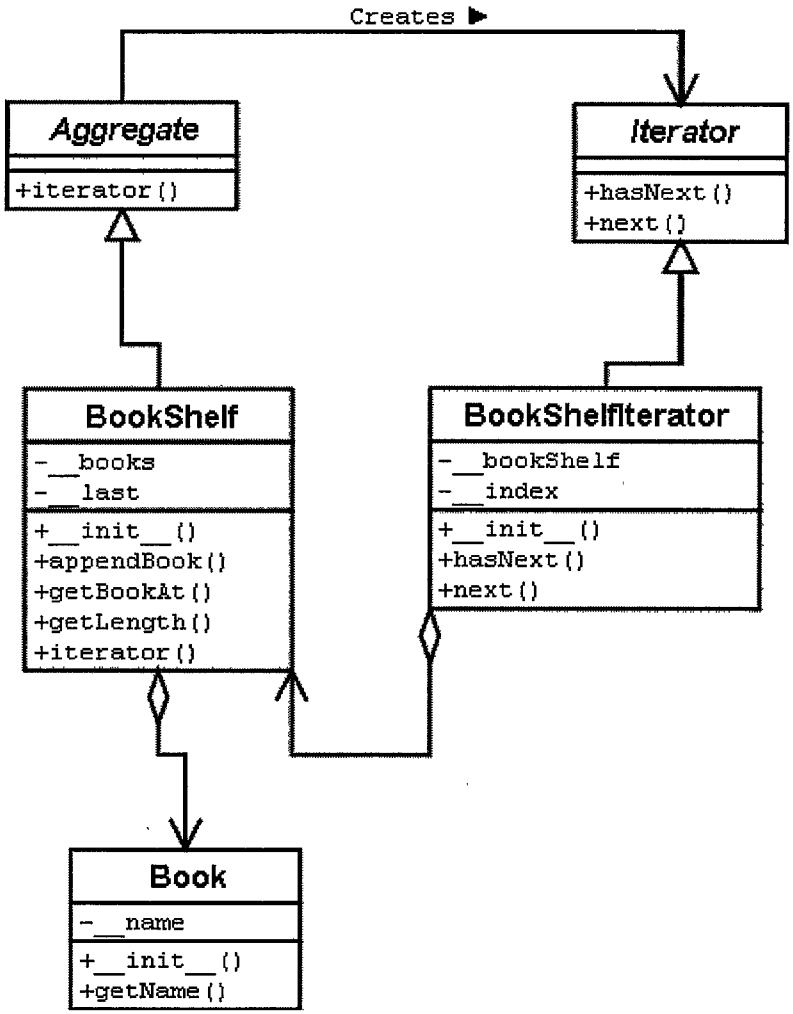


図 4.2.1-5　Iterator パターンのクラス図

図 4.2.2-1 継承を用いた Adapter パターンの実装 (AdapterInherit.py)

```
#!/usr/bin/env python

class Banner:
    def __init__(self, string):
        self.__string = string

    def showWithParen(self):
        print "(" + self.__string + ")"

    def showWithAster(self):
        print "*" + self.__string + "*"

class Print:
    def printWeak(self):
        raise NotImplementedError

    def printStrong(self):
        raise NotImplementedError

class PrintBanner(Banner, Print):
    def __init__(self, string):
        Banner.__init__(self, string)

    def printWeak(self):
        Banner.showWithParen(self)

    def printStrong(self):
        Banner.showWithAster(self)

def main():
    p = PrintBanner("Hello")
    p.printWeak()
    p.printStrong()

if __name__ == "__main__":
    main()
```

図 4.2.2-2　委譲を用いた Adapter パターンの実装 (AdapterDelegate.py)

```
#!/usr/bin/env python

class Banner:
    def __init__(self, string):
        self.__string = string

    def showWithParen(self):
        print "(" + self.__string + ")"

    def showWithAster(self):
        print "*" + self.__string + "*"

class Print:
    def printWeak(self):
        raise NotImplementedError

    def printStrong(self):
        raise NotImplementedError

class PrintBanner(Print):
    def __init__(self, string):
        self.banner = Banner(string)

    def printWeak(self):
        self.banner.showWithParen()

    def printStrong(self):
        self.banner.showWithAster()

def main():
    p = PrintBanner("Hello")
    p.printWeak()
    p.printStrong()

if __name__ == "__main__":
    main()
```

図 4.2.2-3　AdapterInherit.py 及び AdapterDelegate.py の実行結果

```
$ python AdapterInherit.py
(Hello)
*Hello*
```

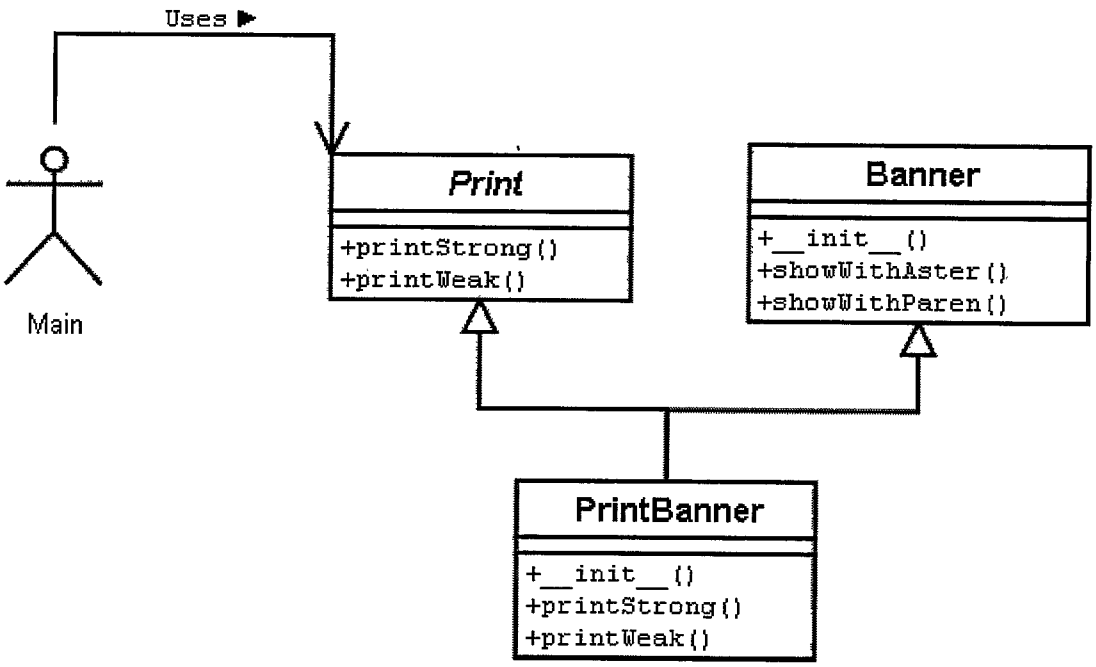


図 4.2.2-4　継承を用いた Adapter パターンのクラス図

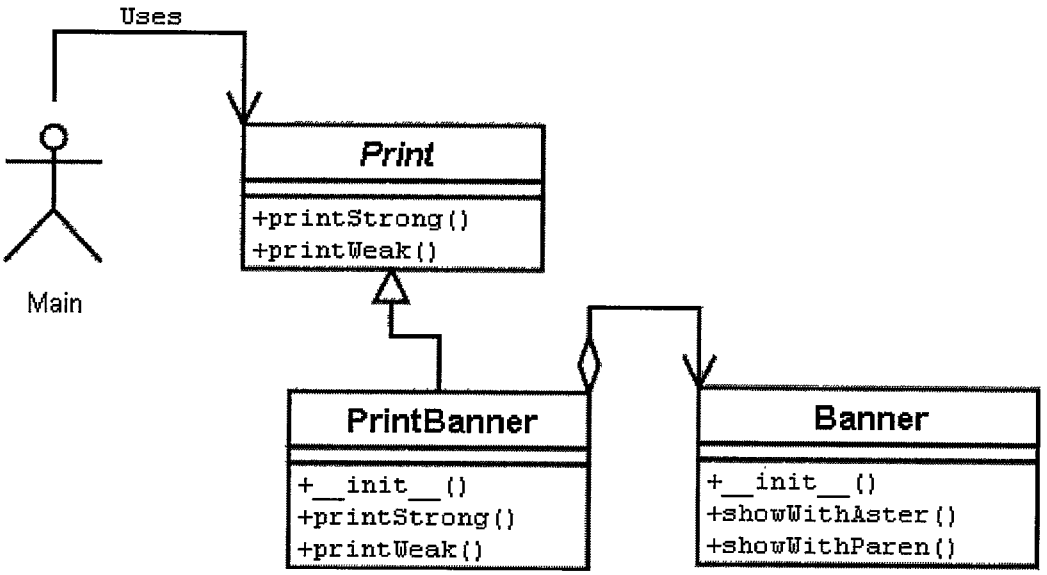


図 4.2.2-5　委譲を用いた Adapter パターンのクラス図

図 4.2.3-1 Tenmplate Method パターンの実装 (TemplateMethod.py)

```
#!/usr/bin/env python

import sys

class AbstractDisplay:
    def open(self):
        raise NotImplementedError

    def print_(self):
        raise NotImplementedError

    def close(self):
        raise NotImplementedError

    def display(self):
        self.open()
        for i in range(5):
            self.print_()
        self.close()

class CharDisplay(AbstractDisplay):
    def __init__(self, ch):
        self.__ch = ch

    def open(self):
        sys.stdout.write("<<")

    def print_(self):
        sys.stdout.write(self.__ch)

    def close(self):
        sys.stdout.write(">>" + "¥n")

class StringDisplay(AbstractDisplay):
    def __init__(self, string):
        self.__string = string
        self.__width = len(string)

    def open(self):
        self.__printLine()

    def print_(self):
        print "|" + self.__string + "|"

    def close(self):
        self.__printLine()

    def __printLine(self):
        sys.stdout.write("+")
        sys.stdout.write("-" * self.__width)
        sys.stdout.write("+" + "¥n")

def main():
    d1 = CharDisplay("H")
```

(次ページへ続く)

（前ページからの続き）

```
    d2 = StringDisplay("Hello, world.")
    d3 = StringDisplay("こんにちは。")
    d1.display()
    d2.display()
    d3.display()

if __name__ == "__main__":
    main()
```

図 4.2.3-2　TemplateMethod.py の実行結果）

```
$ python TemplateMethod.py
<<HHHHH>>
+-------------+
|Hello, world.|
|Hello, world.|
|Hello, world.|
|Hello, world.|
|Hello, world.|
+-------------+
+------------+
|こんにちは。|
|こんにちは。|
|こんにちは。|
|こんにちは。|
|こんにちは。|
+------------+
```

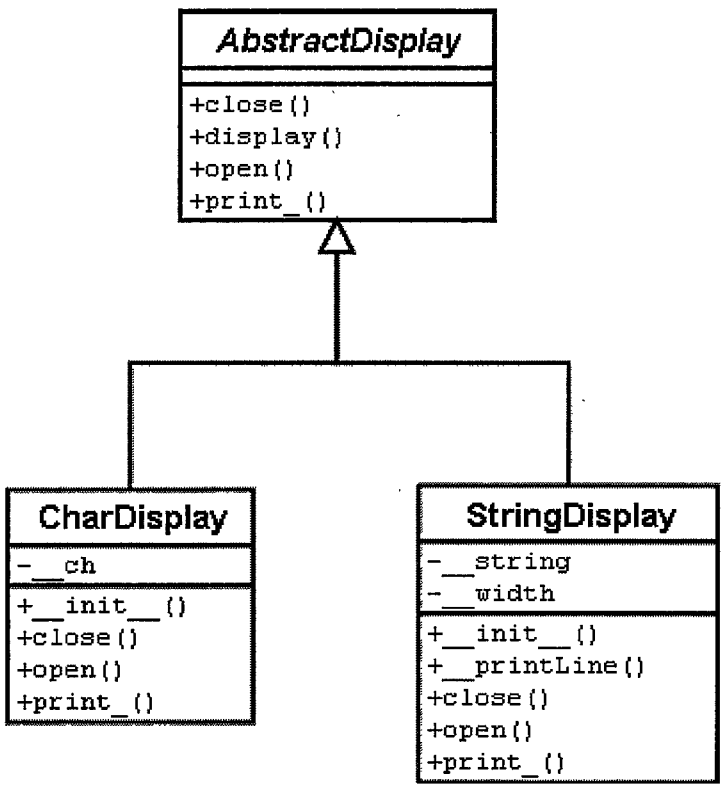


図 4.2.3-3　TemplateMethod パターンのクラス図

図 4.2.4-1 Factory Method パターンの実装 (framework.py)

```
#!/usr/bin/env python

class Product:
    def use(self):
        raise NotImplementedError

class Factory:
    def create(self, owner):
        p = self.createProduct(owner)
        self.registerProduct(p)
        return p

    def createProduct(self, owner):
        raise NotImplementedError

    def registerProduct(self, product):
        raise NotImplementedError
```

図 4.2.4-2 Factory Method パターンの実装（idcard.py）

```
#!/usr/bin/env python

from framework import Product, Factory

class IDCard(Product):
    def __init__(self, owner):
        print owner + "のカードを作ります。"
        self.__owner = owner

    def use(self):
        print self.__owner + "のカードを使います。"

    def getOwner(self):
        return self.__owner

class IDCardFactory(Factory):
    def createProduct(self, owner):
        self.__owners = []
        return IDCard(owner)

    def registerProduct(self, product):
        self.__owners.append(product.getOwner())

    def getOwners(self):
        return self.__owners

def main():
    factory = IDCardFactory()
    card1 = factory.create("結城浩")
    card2 = factory.create("とむら")
    card3 = factory.create("佐藤花子")
    card1.use()
    card2.use()
    card3.use()

if __name__ == "__main__":
    main()
```

図 4.2.4-3 idcard.py の実行結果

```
$ python idcard.py
結城浩のカードを作ります。
とむらのカードを作ります。
佐藤花子のカードを作ります。
結城浩のカードを使います。
とむらのカードを使います。
佐藤花子のカードを使います。
```

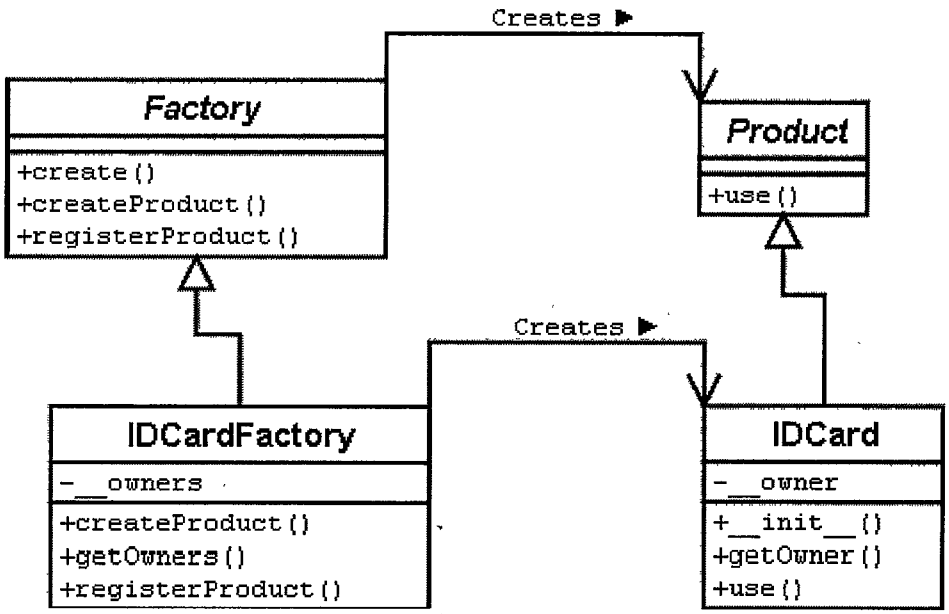


図 4.2.4-4　Factory Method パターンのクラス図

図 4.2.5-1 Singleton パターン (Singleton.py)

```
#!/usr/bin/env python

class Singleton:

    class __Singleton:
        def __init__(self, val):
            self.val = val

        def __str__(self):
            return `self` + str(self.val)

    instance = None

    def __init__(self, val=""):
        if Singleton.instance is None:
            Singleton.instance = Singleton.__Singleton(val)
            print "インスタンスを生成しました。"
        else:
            Singleton.instance.val = val
            print "すでに生成されているので、新たなインスタンスは生成しませんでした。"

    def __getattr__(self, name):
        return getattr(self.instance, name)

    def getInstance(self):
        return Singleton.instance

def main():
    print "Start."
    obj1 = Singleton().getInstance()
    obj2 = Singleton().getInstance()
    print obj1
    print obj2
    if obj1 is obj2:
        print "obj1 と obj2 は同じインスタンスです。"
    else:
        print "obj1 と obj2 は同じインスタンスではありません。"

if __name__ == "__main__":
    main()
```

図 4.2.5-2　Singleton.py の実行結果結果

```
$ python Singleton.py
Start.
インスタンスを生成しました。
すでに生成されているので、新たなインスタンスは生成しませんでした。
<__main__.__Singleton instance at 0x8157eec>
<__main__.__Singleton instance at 0x8157eec>
obj1 と obj2 は同じインスタンスです。
```

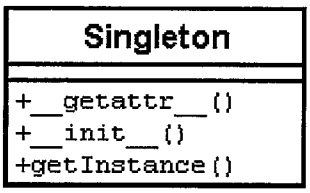


図 4.2.5-3　Singleton パターンのクラス図

図 4.2.6-1 Prototype パターンの実装 (framework.py)

```
#!/usr/bin/env python

import copy

class Product:
    def use(self, s):
        raise NotImplementedError

    def createClone(self):
        raise NotImplementedError

class Manager:
    def __init__(self):
        self.__showcase = {}

    def register(self, name, proto):
        self.__showcase[name] = proto

    def create(self, protoname):
        p = self.__showcase[protoname]
        return copy.copy(p)
```

図 4.2.6-2 Prototype パターンの実装 (main.py)

```
#!/usr/bin/env python

import sys
from framework import Product, Manager

class MessageBox(Product):
    def __init__(self, decochar):
        self.__decochar = decochar

    def use(self, s):
        length = len(s)
        print self.__decochar * (length+4)
        print self.__decochar + " " + s + " " + self.__decochar
        print self.__decochar * (length+4)

    def createClone(self):
        p = None
        p = self.clone()
        return p

class UnderlinePen(Product):
    def __init__(self, ulchar):
        self.__ulchar = ulchar

    def use(self, s):
        length = len(s)
        print '"' + s + '"'
        print ' ' + self.__ulchar * length + ' '

    def createClone(self):
        p = None
        p = self.clone()
        return p

def main():
    manager = Manager()
    upen = UnderlinePen("~")
    mbox = MessageBox("*")
    sbox = MessageBox("/")
    manager.register("strong message", upen)
    manager.register("warning box", mbox)
    manager.register("slash box", sbox)

    p1 = manager.create("strong message")
    p1.use("Hello, world.")
    p2 = manager.create("warning box")
    p2.use("Hello, world.")
    p3 = manager.create("slash box")
    p3.use("Hello, world.")

if __name__ == "__main__":
    main()
```

図 4.2.6-3　main.py の実行結果

```
$ python main.py
"Hello, world."
 ~~~~~~~~~~~~~
*****************
* Hello, world. *
*****************
/////////////////
/ Hello, world. /
/////////////////
```

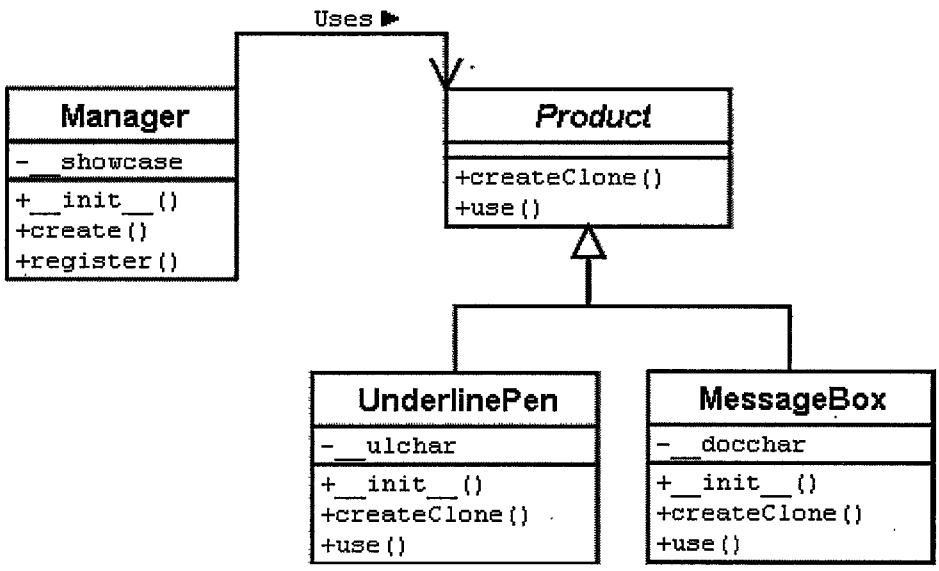


図 4.2.6-4　Prototype パターンのクラス図

図 4.2.7-1 Builder パターンの実装 (Builder.py)

```
#!/usr/bin/env python

import sys
import cStringIO

class Builder:
    def makeTitle(self, title):
        raise NotImplementedError

    def makeString(self, str):
        raise NotImplementedError

    def makeItems(self, items):
        raise NotImplementedError

    def close(self):
        raise NotImplementedError

class Director:
    def __init__(self, builder):
        self.__builder = builder

    def construct(self):
        self.__builder.makeTitle("Greeting");
        self.__builder.makeString("朝から昼にかけて")
        self.__builder.makeItems(["おはようございます。",
                                  "こんにちは。"])
        self.__builder.makeString("夜に")
        self.__builder.makeItems(["こんばんは。",
                                  "おやすみなさい。",
                                  "さようなら。"])
        self.__builder.close()

class TextBuilder(Builder):
    def __init__(self):
        self.__buffer = []

    def makeTitle(self, title):
        self.__buffer.append("==========================¥n")
        self.__buffer.append("「" + title + "」¥n")
        self.__buffer.append("")

    def makeString(self, str):
        self.__buffer.append("■" + str + "¥n")
        self.__buffer.append("¥n")

    def makeItems(self, items):
        for i in range(len(items)):
            self.__buffer.append("  ・" + items[i] + "¥n")
        self.__buffer.append("¥n")

    def close(self):
        self.__buffer.append("==========================¥n")
```

(次ページへ続く)

（前ページからの続き）

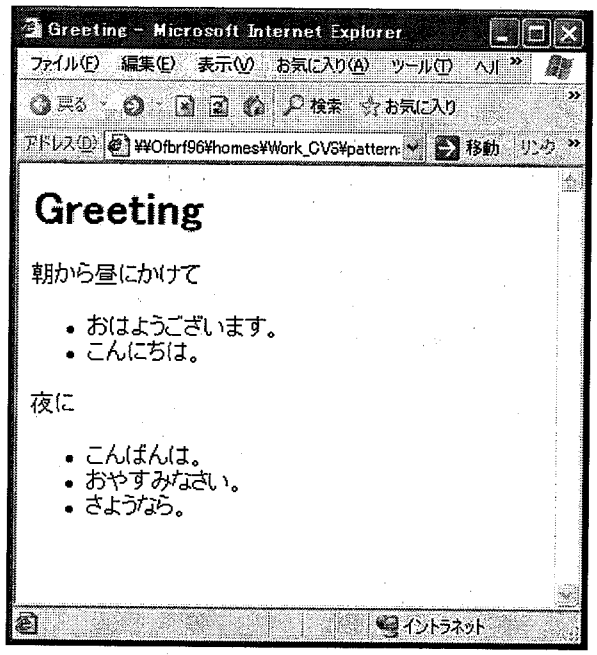
```
    def getResult(self):
        ret = cStringIO.StringIO()
        ret.writelines(self.__buffer)
        ret.seek(0)
        return ret.read()

class HTMLBuilder(Builder):
    def makeTitle(self, title):
        self.__filename = title + ".html"
        sys.stdout = open(self.__filename, "w")
        print "<html><head><title>" + title + "</title></head><body>"
        print "<h1>" + title + "</h1>"

    def makeString(self, str):
        print "<p>" + str + "</p>"

    def makeItems(self, items):
        print "<ul>"
        for i in range(len(items)):
            print "<li>" + items[i] + "</li>"
        print "</ul>"

    def close(self):
        print "</body></html>"
        sys.stdout = sys.__stdout__

    def getResult(self):
        return self.__filename

def main():
    if len(sys.argv) != 2:
        usage(); sys.exit(0)

    if sys.argv[1] == "plain":
        textbuilder = TextBuilder()
        director = Director(textbuilder)
        director.construct()
        result = textbuilder.getResult()
        print result
    elif sys.argv[1] == "html":
        htmlbuilder = HTMLBuilder()
        director = Director(htmlbuilder)
        director.construct()
        filename = htmlbuilder.getResult()
        print filename + "が作成されました。"
    else:
        usage(); sys.exit(0)

def usage():
    print "Usage: python Builder.py plain  プレーンテキストで文書作成"
    print "       python Builder.py html   HTML ファイルで文書作成"

if __name__ == "__main__":
    main()
```

図 4.2.7-2 Builder.py の実行結果（テキスト形式）

```
$ python Builder.py plain
==========================
「Greeting」
■朝から昼にかけて

   ・おはようございます。
   ・こんにちは。

■夜に

   ・こんばんは。
   ・おやすみなさい。
   ・さようなら。

==========================
```

図 4.2.7-3 Builder.py の実行結果（HTML 形式）

```
$ python Builder.py html
Greeting.html が作成されました。
```



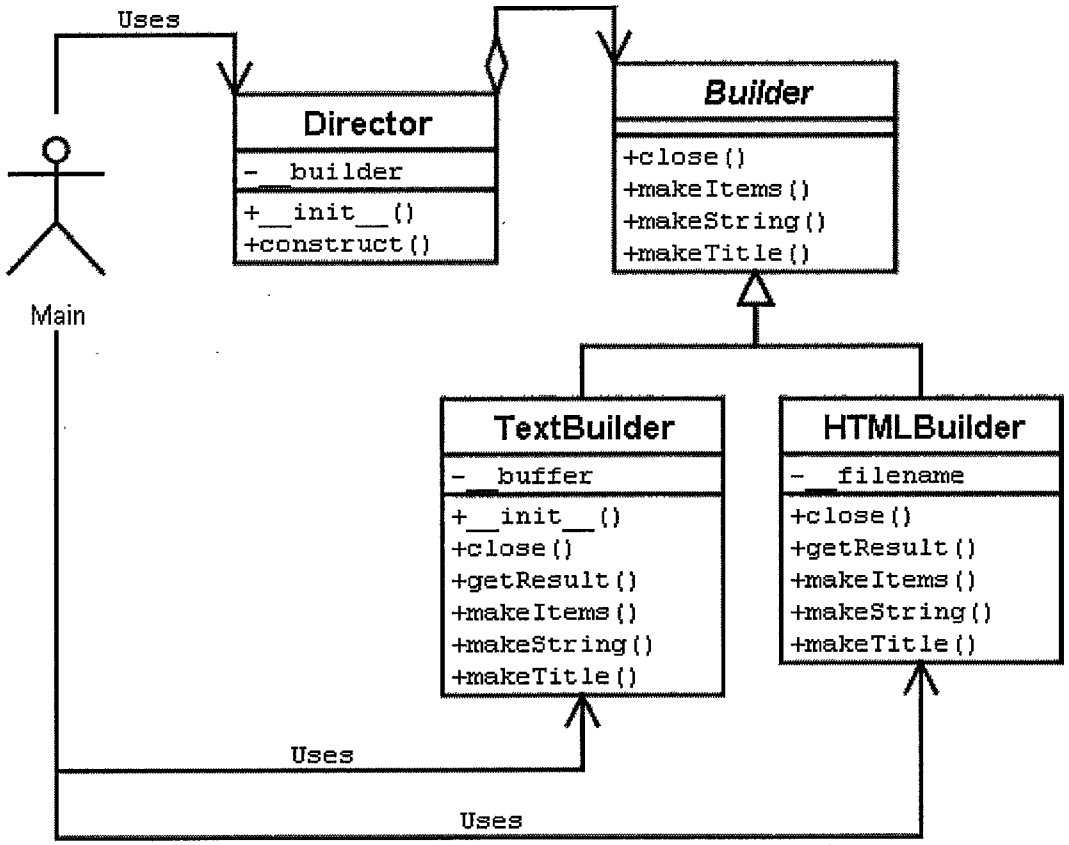


図 4.2.7-4 Builder パターンのクラス図

図 4.2.8-1 Abstract Factory パターンの実装 (factory.py)

```
#!/usr/bin/env python

class Item:
    def __init__(self, caption):
        self._caption = caption

    def makeHTML(self):
        raise NotImplementedError

class Link(Item):
    def __init__(self, caption, url):
        Item.__init__(self, caption)
        self._url = url

class Tray(Item):
    def __init__(self, caption):
        Item.__init__(self, caption)
        self._tray = []

    def add(self, item):
        self._tray.append(item)

class Page:
    def __init__(self, title, author):
        self._title = title
        self._author = author
        self._content = []

    def add(self, item):
        self._content.append(item)

    def output(self):
        filename = self._title + ".html"
        writer = open(filename, "w")
        writer.write(self.makeHTML())
        writer.close()
        print filename + "を作成しました。"

    def makeHTML(self):
        raise NotImplementedError

class Factory(object):
    def getFactory(self, modulename, classname):
        exec("import " + modulename)
        exec("factory = " + modulename + "." + classname + "()")
        return factory

    def createLink(self, caption, url):
        raise NotImplementedError

    def createTray(self, caption):
        raise NotImplementedError

    def createPage(self, title, author):
```

(次ページへ続く)

（前ページからの続き）

```
        raise NotImplementedError

    getFactory = classmethod(getFactory)
```

図 4.2.8-2 Abstract Factory パターンの実装（listfactory.py）

```
#!/usr/bin/env python

from factory import *

class ListFactory(Factory):
    def createLink(self, caption, url):
        return ListLink(caption ,url)

    def createTray(self, caption):
        return ListTray(caption)

    def createPage(self, title, author):
        return ListPage(title, author)

class ListLink(Link):
    def __init__(self, caption, url):
        Link.__init__(self, caption, url)

    def makeHTML(self):
        return "  <li><a href='" + self._url + "'>" + self._caption + "</a></li>¥n"

class ListTray(Tray):
    def __init__(self, caption):
        Tray.__init__(self, caption)

    def makeHTML(self):
        buffer = []
        buffer.append("<li>¥n")
        buffer.append(self._caption + "¥n")
        buffer.append("<ul>¥n")
        for item in self._tray:
            buffer.append(item.makeHTML())
        buffer.append("</ul>¥n")
        buffer.append("</li>¥n")
        return toString(buffer)

class ListPage(Page):
    def __init__(self, title, author):
        Page.__init__(self, title, author)

    def makeHTML(self):
        buffer = []
        buffer.append("<html><head><title>" + self._title + "</title></head>¥n")
        buffer.append("<body>¥n")
        buffer.append("<h1>" + self._title + "</h1>¥n")
        buffer.append("<ul>¥n")
        for item in self._content:
            buffer.append(item.makeHTML())
        buffer.append("</ul>¥n")
        buffer.append("<hr><address>" + self._author + "</address>")
        buffer.append("</body></html>¥n")
        return toString(buffer)
```

（次ページへ続く）

（前ページからの続き）

```
def toString(strlist):
    import cStringIO
    ret = cStringIO.StringIO()
    ret.writelines(strlist)
    ret.seek(0)
    return ret.read()
```

図 4.2.8-3 Abstract Factory パターンの実装 (tablefactory.py)

```
#!/usr/bin/env python

from factory import *
from listfactory import toString

class TableFactory(Factory):
    def createLink(self, catption, url):
        return TableLink(catption, url)

    def createTray(self, caption):
        return TableTray(caption)

    def createPage(self, title, author):
        return TablePage(title, author)

class TableLink(Link):
    def __init__(self, caption, url):
        Link.__init__(self, caption, url)

    def makeHTML(self):
        return "<td><a href='" + self._url + "'>" + self._caption + "</a></td>¥n"

class TableTray(Tray):
    def __init__(self, caption):
        Tray.__init__(self, caption)

    def makeHTML(self):
        buffer = []
        buffer.append("<td>")
        buffer.append("<table width='100%' border='1'><tr>")
        buffer.append("<td bgcolor='#cccccc' align='center' colspan='"
                      + str(len(self._tray))
                      + "'><b>" + self._caption + "</b></td>")
        buffer.append("</tr>¥n")
        buffer.append("<tr>¥n")
        for item in self._tray:
            buffer.append(item.makeHTML())
        buffer.append("</tr></table>")
        buffer.append("</td>")
        return toString(buffer)

class TablePage(Page):
    def __init__(self, title, author):
        Page.__init__(self, title, author)

    def makeHTML(self):
        buffer = []
        buffer.append("<html><head><title>" + self._title + "</title></head>¥n")
        buffer.append("<body>¥n")
        buffer.append("<h1>" + self._title + "</h1>¥n")
        buffer.append("<table width='80%' border='3'>¥n")
        for item in self._content:
            buffer.append("<tr>" + item.makeHTML() + "</tr>")
```

(次ページへ続く)

（前ページからの続き）

```
        buffer.append("</table>¥n")
        buffer.append("<hr><address>" + self._author + "</address>")
        buffer.append("</body></html>¥n")
        return toString(buffer)
```

図 4.2.8-4 Abstract Factory パターン (main.py)

```
#!/usr/bin/env python

import sys
from factory import *
from listfactory import *

def main():
    if len(sys.argv) != 3:
        print "Usage: python main.py module_name class_name"
        print "Example 1: python main.py listfactory ListFactory"
        print "Example 2: python main.py tablefactory TableFactory"
        sys.exit(0)

    factory = Factory.getFactory(sys.argv[1], sys.argv[2])

    asahi   = factory.createLink("朝日新聞", "http://www.asahi.com/")
    yomiuri = factory.createLink("読売新聞", "http://www.yomiuri.co.jp/")

    us_yahoo = factory.createLink("Yahoo!", "http://www.yahoo.com/")
    jp_yahoo = factory.createLink("Yahoo!Japan", "http://www.yahoo.co.jp/")
    excite   = factory.createLink("Excite", "http://www.excite.com/")
    google   = factory.createLink("Google", "http://www.google.com/")

    traynews = factory.createTray("新聞")
    traynews.add(asahi)
    traynews.add(yomiuri)

    trayyahoo = factory.createTray("Yahoo!")
    trayyahoo.add(us_yahoo)
    trayyahoo.add(jp_yahoo)

    traysearch = factory.createTray("サーチエンジン")
    traysearch.add(trayyahoo)
    traysearch.add(excite)
    traysearch.add(google)

    page = factory.createPage("LinkPage", "結城 浩")
    page.add(traynews)
    page.add(traysearch)
    page.output()

if __name__ == "__main__":
    main()
```

図 4.2.8-5　main.py の実行結果（ListFactory の利用）

```
$ python main.py listfactory ListFactory
LinkPage.html を作成しました。
```

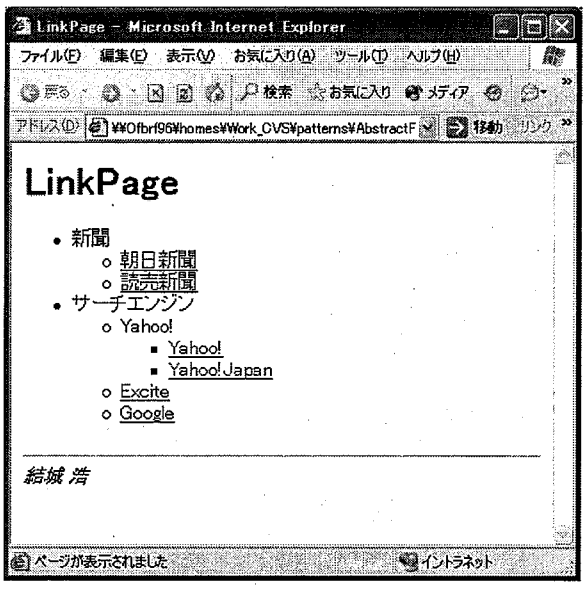
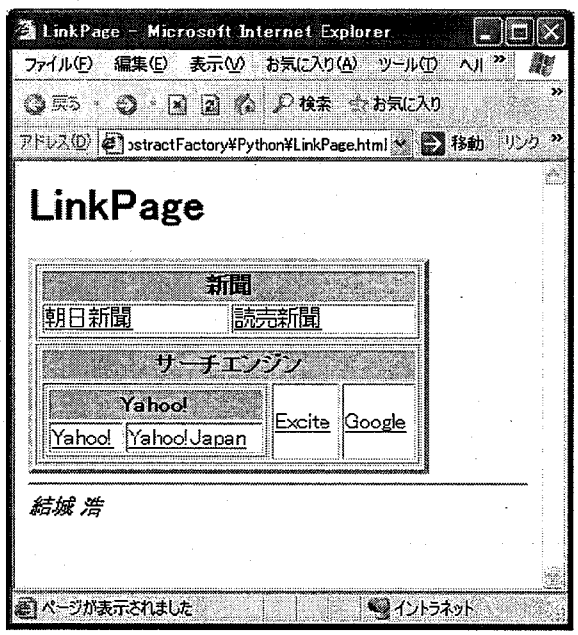


図 4.2.8-6　main.py の実行結果（TableFactory の利用）

```
$ python main.py tablefactory TableFactory
LinkPage.html を作成しました。
```



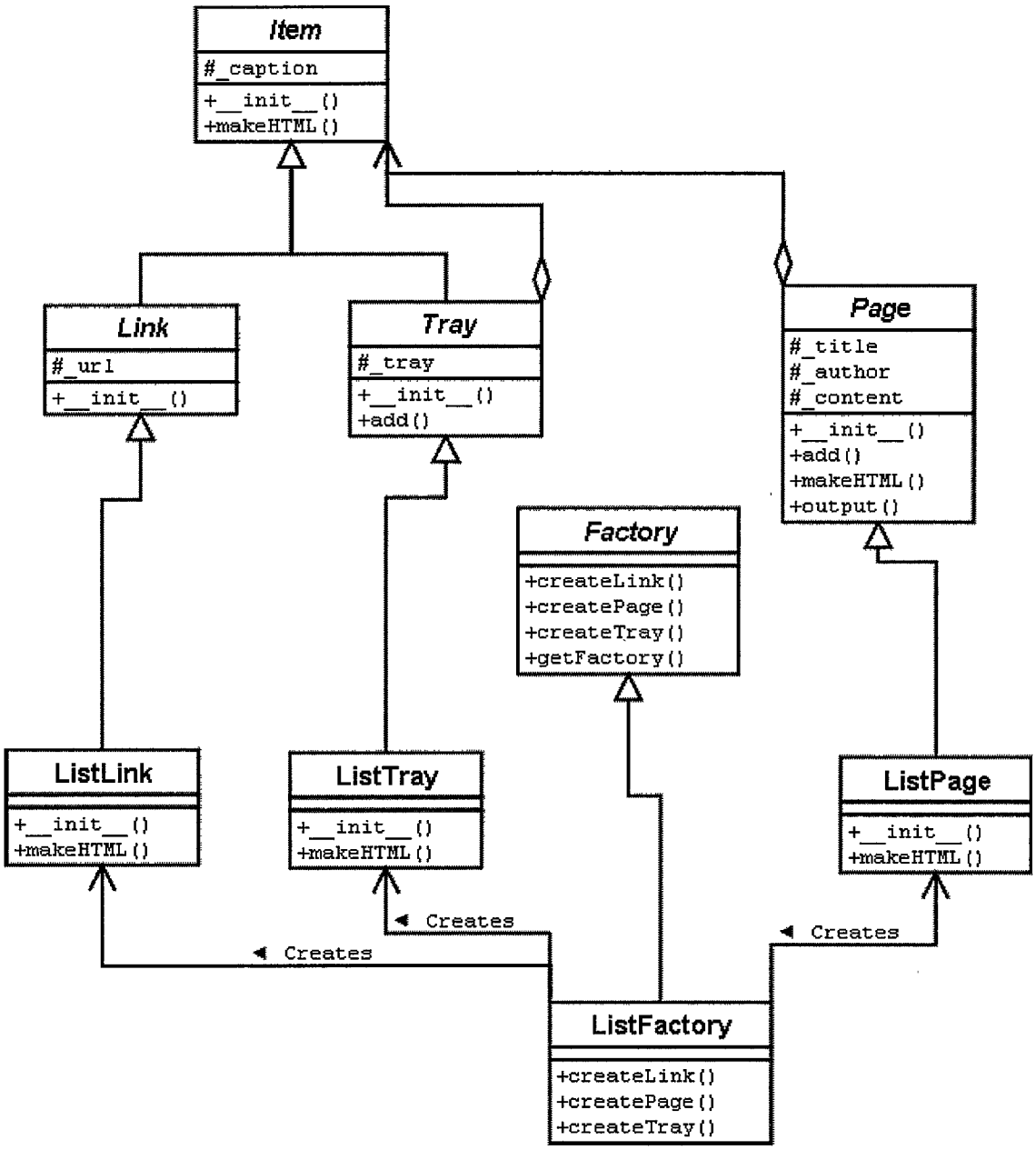


図 4.2.8-7　Abstract Factory パターンのクラス図

図 4.2.9-1 Bridge パターンの実装 (Bridge.py)

```
#!/usr/bin/env python

class Display:
    def __init__(self, impl):
        self.__impl = impl

    def open(self):
        self.__impl.rawOpen()

    def print_(self):
        self.__impl.rawPrint()

    def close(self):
        self.__impl.rawClose()

    def display(self):
        self.open()
        self.print_()
        self.close()

class CountDisplay(Display):
    def __init__(self, impl):
        Display.__init__(self, impl)

    def multiDisplay(self, times):
        self.open()
        for i in range(times):
            self.print_()
        self.close()

class DisplayImpl:
    def rawOpen(self):
        raise NotImplementedError

    def rawPrint(self):
        raise NotImplementedError

    def rawClose(self):
        raise NotImplementedError

class StringDisplayImpl(DisplayImpl):
    def __init__(self, string):
        self.__string = string
        self.__width = len(string)

    def rawOpen(self):
        self.__printLine()

    def rawPrint(self):
        print("|" + self.__string + "|")

    def rawClose(self):
        self.__printLine()
```

(次ページへ続く)

（前ページからの続き）

```
    def __printLine(self):
        print "+" + "-" * self.__width + "+"

def main():
    d1 = Display(StringDisplayImpl("Hello, Japan"))
    d2 = CountDisplay(StringDisplayImpl("Hello, World."))
    d3 = CountDisplay(StringDisplayImpl("Hello, Universe."))
    d1.display()
    d2.display()
    d3.display()
    d3.multiDisplay(5)

if __name__ == "__main__":
    main()
```

図 4.2.9-2　Bridge.py の実行結果

```
$ python Bridge.py
+------------+
|Hello, Japan|
+------------+
+--------------+
|Hello, World.|
+--------------+
+----------------+
|Hello, Universe.|
+----------------+
+----------------+
|Hello, Universe.|
|Hello, Universe.|
|Hello, Universe.|
|Hello, Universe.|
|Hello, Universe.|
+----------------+
```

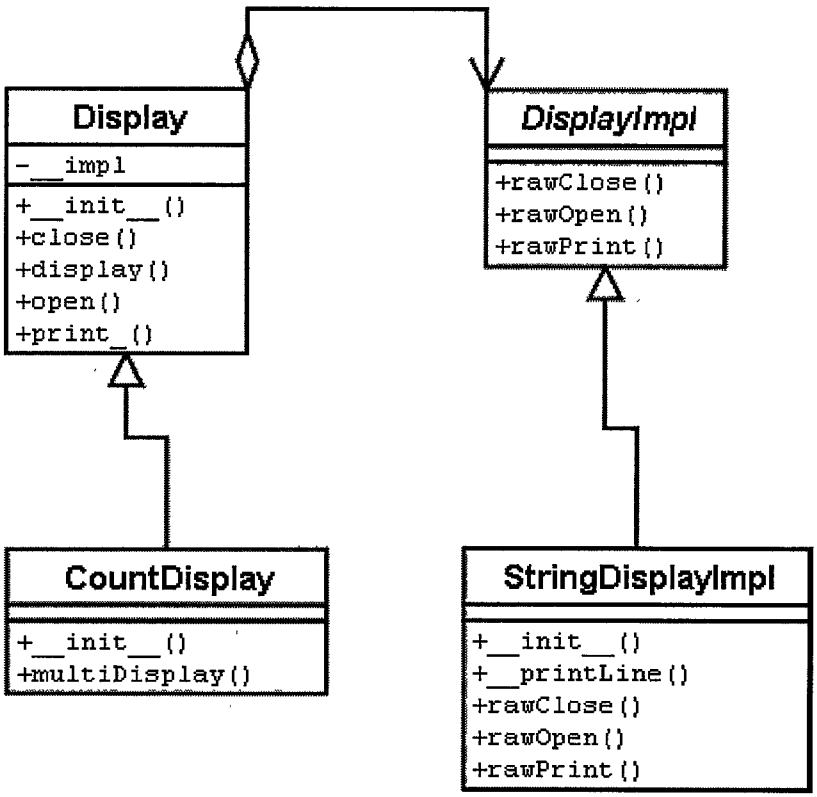


図 4.2.9-3　Bridge パターンのクラス図

4.3 オブジェクト指向と実行効率

オブジェクト指向プログラミング言語の実行速度は、一般に、従来の手続き型のプログラミング言語に比べて遅くなるといわれている。これは、一般に、オブジェクトを取り扱う際のオーバーヘッドが大きくなるからである。一方で、オブジェクト指向プログラミング言語を使うことにより、プログラムの拡張や変更が容易になり、プログラム自体を理解しやすくなるという利点がある。オブジェクト指向と実行効率はトレードオフの関係にある。

本章では、Python を例として、ポリモルフィズムの利用が実行効率に与える影響を評価した結果を示す。また、Python から MPI ライブラリを利用するための、PyMPI[4-18]と呼ばれる拡張ライブラリを使い、Python による並列計算のテストを行ったので、この結果も示す。

4.3.1 条件分岐とポリモルフィズム

オブジェクト指向プログラミングでは、ポリモルフィズムの機能を使うことにより、条件分岐を少なくすることが可能である。よく似た条件分岐が、システムの各所に分散されていると、プログラムの修正を困難にする。このため、ポリモルフィズムを利用して、条件分岐を少なくすることは、プログラムの拡張性・保守性を上げることにつながる。

ポリモルフィズムはオブジェクト指向プログラミングでは重要な概念であるが、一方で、実行効率が落ちるという考え方が強く、ポリモルフィズムを使いたがらないプログラマが多いのも現実である。しかしながら、最近は、コンパイラの向上により、通常の条件分岐よりもポリモルフィズムを利用した方が、実行効率が良くなる、という報告もある。

ここでは、Martin Fowler 著「リファクタリング」[4-19]で紹介されているポリモルフィズムと条件分岐のプログラムの例を Python に移植することで、ポリモルフィズムによる条件分岐の置き換えの例を示すとともに、Python のプロファイラ機能を用いた実行効率のベンチマークテストを行う。

4.3.1.1 Java の例

図 4.3.1-11 に、文献 4-19 で紹介されている Java の switch 文を使ったプログラム例を示す。上述の文献の中では、図 4.3.1-1 に示されるようなプログラムを図 4.3.1-2 のように置き換えることができると説明されている。この例では、payAmount という関数の中の switch 文を、Engineer、Salesman、Manager クラスのポリモルフィックなメソッド payAmount に置き換えている。

4.3.1.2 Pythonにおける実装

前述のJavaの例を参考にして、実際に動作するPythonプログラムを作成した。図4.3.1-3に条件分岐版のベンチマーク用プログラム（bench.py）を示す。図4.3.1-4に、同じくポリモルフィズム版のベンチマーク用プログラム（bench2.py）を示す。

このプログラムはPython2.2以上で動作する。Python2.2以前のバージョンでは、JavaやC++のstaticメソッドやクラス変数に相当する概念がなかったが、Python2.2ではこれらの概念が導入されており、ほぼ直接翻訳する形で、JavaからPythonに変換することができた。ただし、Pythonにはswitch文がないので、if-elif-else文によりswitch文相当のプログラムとしている。

なお、Pythonでは設計思想上、switch文はif-elif-else文で全く同じことができるので、不要と考えられている。switch文に相当する機能が必要な場合はif-elif-else文を使うことになっている。

4.3.1.3 Pythonによる実行速度のベンチマーク

Pythonはインタプリタ言語であるため、一般に、C、C++、Fortran等のコンパイラ言語に比べると実行速度は、数倍から数十倍遅いといわれている。このため、Pythonどうしで実行速度のベンチマークを行うことは、実用上あまり意味はないが、上で作成した2つのプログラムの実行速度を比較した。

図4.3.1-5に条件分岐版、図4.3.1-6にポリモルフィズム版の実行結果を示す。なお、実行速度について詳細に調べるため、Pythonの標準モジュールであるプロファイラ profileを使用している。

これらの結果から分かるように、全体の実行時間は、bench.pyが9.200秒、bench2.pyが8.630秒となる。計算機の負荷状況により、結果は少し変化するが、数回実行して有意な変化が起こらないことを確認した。この結果から、わずかであるが条件分岐版の方が遅くなることが分かった。

プロファイラの結果を詳細に見てみると、条件分岐版の方が、関数呼び出し回数がbench.pyの方が多くなっていることが分かる（特にgetType）。このベンチマークテストで、条件分岐版の方が遅くなったのは、これが原因と考えられる。一方、ポリモルフィズム版では、payAmountメソッドをポリモルフィックに呼び出す時のオーバーヘッドというよりは、インスタンス生成（create）にかかるオーバーヘッドが大きくなって実行速度が落ちていることが分かる。

このため、条件分岐版のプログラムを、getTypeメソッドを呼ばずに直接クラス変数_typeを参照するように変更（コメント部分）してみた。図4.3.1-7に、この修正を施した条件分

岐版の実行結果を示す。このようにすることで、条件分岐版の方が有意に速くなった。

4.3.1.4 まとめ

Python を使って、条件分岐をポリモルフィズムで置き換えるサンプルプログラムを作成した。ポリモルフィズム版の方がプログラムは長くなり、このような小さなプログラムでは、ポリモルフィズムを使う利点は分かりにくいが、このようにすることで、条件分岐を一カ所にまとめることが可能とある。この例では、Employee クラスの create メソッドにのみ条件分岐があれば良いことになる。

また、作成した Python プログラムを使って、実行速度のベンチマークテストを行った。この結果、ポリモルフィズムを使うと、クラスが増えてクラス生成に時間がかかるため、実行速度が遅くなるのが確認されたが、ポリモルフィックにメソッドを呼び出すこと自体は、それほど実行速度が遅くなる原因とはならないことが分かった。

この結果は、プログラミング言語やプログラムの書き方によって状況は変化すると考えられるが、オブジェクト指向プログラミングの適用により実行速度が遅くなる効果を、手続き型言語における経験に基づく推測だけで過大評価しないように注意する必要がある。実行速度に関する誤った推測で、不必要にコードの拡張性、可読性を下げないようにするためには、プロファイラの活用等により、実際にどの処理に時間がかかっているのかを正確に把握した上で、解決策を考えるべきである。

4.3.2 PyMPI を用いた Python による並列計算

Python から MPI ライブラリを呼び出すことで、Python で並列計算することを可能とする、PyMPI と呼ばれる拡張ライブラリ[4-18]の開発が、米国のカリフォルニア大学とローレンス・リバモア国立研究所の共同研究として進められている。現在、まだα版であるが、添付されていたサンプルプログラムが正常に動作することを確認できたので、その結果を示す。

4.3.2.1 動作環境

すぐに利用できる PC クラスタが手元になかったので、LAN ケーブル（100Base/T）で繋がっているワークステーション４台を使って、並列計算を行った。４台の内１台は 2CPU マシンであるので、CPU 数としては５である。並列計算専用ではないため、各ノード間の通信や各ワークステーションに、他の負荷がかかっている状況下でテストしたため、並列計算専用機であれば、もう少し速くなる可能性がある。

計算に使ったワークステーションの構成は以下の通りである。なお、計算の実行は ofbrf96 から行った。

- ホスト名 ofbrf86 : Redhat Linux 8.0, Pentium4 CPU 2.4GHz×2
- ホスト名 ofbrf93 : Redhat Linux 7.2, Pentium4 CPU 2.4GHz×1
- ホスト名 ofbrf94 : Redhat Linux 7.2, Pentium4 CPU 1.8GHz×1
- ホスト名 ofbrf96 : Redhat Linux 7.2, Pentium4 CPU 2.0GHz×1

4.3.2.2 テスト問題

図 4.3.2-1 に、PyMPI のパッケージに添付されていたサンプルプログラム（fractal.py）を転載する。このプログラムは、フラクタルを計算して Bitmap 形式の画像を出力する。プログラム中の mpi.rank、mpi.procs、mpi.reduce などで、Python から並列計算を制御していることが分かる。

4.3.2.3 実行結果

図 4.3.2-2 に実行結果を示す。ノード数を 1～5 に変更して実行している。結果は、以下のとおりである。

- ノード数 1 : 50sec

- ノード数 2 : 36sec
- ノード数 3 : 22sec
- ノード数 4 : 20sec
- ノード数 5 : 15sec

サンプルプログラムとして、線形性の良い問題が選ばれているものと思われるが、Python からの MPI 利用でも十分に並列計算の性能が出ていることが確認できる。

4.3.2.4 まとめ

PyMPI を使って、Python による並列計算をテストした。Python の実行速度自体が遅いので、実行速度を上げたい場合は、並列計算を導入する以前に、計算時間のかかる部分を C や C++、Fortan などで書き換えるべきであろうが、PyMPI のようなものが可能であれば、アルゴリズムに密接した形で並列計算を組み込まずに、Python のレベルの大きな部品に対して、並列計算のアルゴリズムを適用するのを容易にする可能性がある。

図 4.3.1-1　Java における switch 文の例

```
class Employee...
  int payAmount() {
    switch (getType)) {
      case EmployeeType.ENGINEER:
        return _monthlySalary;
      case EmployeeType.SALESMAN:
        return _monthlySalary + _commission;
      case EmployeeType.MANAGER:
        return _monthlySalary + _bonus;
      default:
        throw new RuntimeException("不正な従業員");
    }
  }
```

図 4.3.1-2　Java におけるポリモルフィズムによる switch 文の置換

```
class EmployeeType...
  abstract int payAmount(Employee emp);

class Engineer...
  int payAmount(Employee emp) {
    return emp.getMonthlySalary();
  }

class Salesman...
  int payAmount(Employee emp) {
    return emp.getMonthlySalary() + emp.getCommission();
  }

class Manager...
  int payAmount(Employee emp) {
    return emp.getMonthlySalary() + emp.getBonus();
  }
```

図 4.3.1-3 Python によるベンチマーク（条件分岐版：bench.py）

```
#!/usr/bin/env python

"""ポリモルフィズムのベンチマーク
 （条件分岐版 payAmount）
"""
class Employee(object):

    ENGINEER = 0
    SALESMAN = 1
    MANAGER  = 2
    _monthlySalary = 100
    _commission    = 20
    _bonus         = 50

    def __init__(self, code):
        self.setType(code)

    def getType(self):
        return self._type

    def setType(self, code):
        self._type = code

    def payAmount(self):
        if   self.getType() == Employee.ENGINEER:
        ## if   self._type == Employee.ENGINEER:
            return Employee._monthlySalary
        elif self.getType() == Employee.SALESMAN:
        ## elif self._type == Employee.SALESMAN:
            return Employee._monthlySalary + Employee._commission
        elif self.getType() == Employee.MANAGER:
        ## elif self._type == Employee.MANAGER:
            return Employee._monthlySalary + Employee._bonus;
        else:
            raise RuntimeError, "Illegal Employee Code"

def main():
    MAX = 100000
    for i in range(MAX):
        emp = Employee(i%3)
        if i < 3:
            print emp
            print emp.getType()
            print emp.payAmount()
        else:
            emp.getType()
            emp.payAmount()

if __name__ == "__main__":
    import profile
    profile.run("main()")
```

図 4.3.1-4 Python によるベンチマーク（ポリモルフィズム版：bench2.py）

```
#!/usr/bin/env python

"""ポリモルフィズムのベンチマーク
（ポリモルフィズム版 payAmount）
"""
class Employee(object):

    ENGINEER = 0
    SALESMAN = 1
    MANAGER  = 2
    _monthlySalary = 100
    _commission    = 20
    _bonus         = 50

    def __init__(self, code):
        self.setType(code)

    def getType(self):
        return self._type

    def setType(self, code):
        self._type = code

    def payAmount(self):
        raise NotImplementedError, "Abstract Method"

    def create(code):
        if   code == Employee.ENGINEER:
            return Engineer()
        elif code == Employee.SALESMAN:
            return Salesman()
        elif code == Employee.MANAGER:
            return Manager()
        else:
            raise RuntimeError, "Illegal Employee Code"
    create = staticmethod(create)

class Engineer(Employee):
    def __init__(self):
        self.setType(Employee.ENGINEER)

    def payAmount(self):
        return Employee._monthlySalary

class Salesman(Employee):
    def __init__(self):
        self.setType(Employee.SALESMAN)

    def payAmount(self):
        return Employee._monthlySalary + Employee._commission

class Manager(Employee):
    def __init__(self):
        self.setType(Employee.MANAGER)
```

（次ページへ続く）

（前ページからの続き）

```
    def payAmount(self):
        return Employee._monthlySalary + Employee._bonus;

def main():
    MAX = 100000
    for i in range(MAX):
        emp = Employee.create(i%3)
        if i < 3:
            print emp
            print emp.getType()
            print emp.payAmount()
        else:
            emp.getType()
            emp.payAmount()

if __name__ == "__main__":
    import profile
    profile.run("main()")
```

図 4.3.1-5　ポリモルフィズムのベンチマーク実行結果（条件分岐版）

```
$ python bench.py
<__main__.Employee object at 0x8155e44>
0
100
<__main__.Employee object at 0x81561dc>
1
120
<__main__.Employee object at 0x815477c>
2
150
         600002 function calls in 9.200 CPU seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    9.180    9.180 <string>:1(?)
   100000    1.490    0.000    2.340    0.000 bench.py:15(__init__)
   299999    1.770    0.000    1.770    0.000 bench.py:18(getType)
   100000    0.850    0.000    0.850    0.000 bench.py:21(setType)
   100000    2.330    0.000    3.520    0.000 bench.py:24(payAmount)
        1    2.740    2.740    9.180    9.180 bench.py:37(main)
        1    0.020    0.020    9.200    9.200 profile:0(main())
        0    0.000             0.000          profile:0(profiler)
```

―― 図 4.3.1-6 ポリモルフィズムのベンチマーク実行結果（ポリモルフィズム版） ―

```
$ python bench2.py
<__main__.Engineer object at 0x81522f4>
0
100
<__main__.Salesman object at 0x815df3c>
1
120
<__main__.Manager object at 0x8154784>
2
150
         500003 function calls in 8.630 CPU seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    8.620    8.620 <string>:1(?)
   100000    0.700    0.000    0.700    0.000 bench2.py:18(getType)
   100000    0.830    0.000    0.830    0.000 bench2.py:21(setType)
   100000    2.000    0.000    4.340    0.000 bench2.py:27(create)
    33334    0.500    0.000    0.740    0.000 bench2.py:39(__init__)
    33334    0.170    0.000    0.170    0.000 bench2.py:42(payAmount)
    33333    0.570    0.000    0.790    0.000 bench2.py:46(__init__)
    33333    0.200    0.000    0.200    0.000 bench2.py:49(payAmount)
    33333    0.440    0.000    0.810    0.000 bench2.py:53(__init__)
    33333    0.330    0.000    0.330    0.000 bench2.py:56(payAmount)
        1    2.880    2.880    8.620    8.620 bench2.py:59(main)
        1    0.010    0.010    8.630    8.630 profile:0(main())
        0    0.000             0.000          profile:0(profiler)
```

図 4.3.1-7　条件分岐版 (bench.py) で getType を使わない場合の実行結果

```
$ python bench.py
<__main__.Employee object at 0x8153c54>
0
100
<__main__.Employee object at 0x814f61c>
1
120
<__main__.Employee object at 0x8153c54>
2
150
         400003 function calls in 6.860 CPU seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000    6.850    6.850 <string>:1(?)
   100000    1.630    0.000    2.310    0.000 bench.py:15(__init__)
   100000    0.460    0.000    0.460    0.000 bench.py:18(getType)
   100000    0.680    0.000    0.680    0.000 bench.py:21(setType)
   100000    0.980    0.000    0.980    0.000 bench.py:24(payAmount)
        1    3.100    3.100    6.850    6.850 bench.py:37(main)
        1    0.010    0.010    6.860    6.860 profile:0(main())
        0    0.000             0.000          profile:0(profiler)
```

図 4.3.2-1 PyMPI による並列計算サンプルプログラム (fractal.py) (4-???)

```
import mpi,sys
from array import *
from struct import *

#Function to return BMP header
def makeBMPHeader( width, height ):

    #Set up the bytes in the BMP header
    headerBytes = [ 66, 77, 28, 88, 0, 0, 0, 0, 0, 0, 54, 0, 0, 0]
    headerBytes += [40] + 3*[0] + [100] + 3*[0] + [ 75,0,0,0,1,0,24] + [0]*9 + [18,11,0,0,18,11]
    headerBytes += [0]*10

    #Pack this data sa a string
    data =""
    for x in range(54):
        data += pack( 'B', headerBytes[x] )

    #Create a string to overwrite the width and height in the BMP header
    replaceString = pack( 'l', width )
    replaceString += pack( 'l', height)

    #Return a 54-byte string that will be the new BMP header
    newString = data[0:18] + replaceString + data[26:54]
    return newString

#Define our fractal parameters
c = 0.4 + 0.3j
maxIterationsPerPoint = 64
distanceWhenUnbounded = 3.0

#define our function to iterate
def f( x ):
    return x*x + c

#Define the bounds of the xy plane we will work in
globalBounds = (-0.6, -0.6, 0.4, 0.4 ) #x1, y1, x2, y2

#define the size of the BMP to output
#For now this must be divisible by the # of processes
bmpSize = (400,400)

#Define the range of y-pixels in the BMP this process works on
myYPixelRange = [ 0,0]
myYPixelRange[0] = mpi.rank*bmpSize[1]/mpi.procs
myYPixelRange[1] = (mpi.rank+1)*bmpSize[1]/mpi.procs

if mpi.rank == 0:
    print "Starting computation (groan)¥n"

#Now we can start to iterate over pixels!!
myString = ""
myArray = array('B')
for y in range( myYPixelRange[0], myYPixelRange[1]):
    for x in range( 0, bmpSize[0] ):
```

(次ページへ続く)

（前ページからの続き）

```
        #Calculate the (x,y) in the plane from the (x,y) in the BMP
        thisX = globalBounds[0] + (float(x)/bmpSize[0])*(globalBounds[2]-globalBounds[0])
        thisY = (float(y)/bmpSize[1])*(globalBounds[3] - globalBounds[1])
        thisY += globalBounds[1]

        #Create a complex # representation of this point
        thisPoint = complex(thisX, thisY)

        #Iterate the function f until it grows unbounded
        nxt = f( thisPoint )
        numIters = 0
        while 1:
            dif = nxt-thisPoint
            if abs(nxt - thisPoint) > distanceWhenUnbounded:
                break;
            if numIters >= maxIterationsPerPoint:
                break;
            nxt = f(nxt)
            numIters = numIters+1

        #Convert the number of iterations to a color value
        colorFac = 255.0*float(numIters)/float(maxIterationsPerPoint)
        myRGB = ( colorFac*0.8 + 32, 24+0.1*colorFac, 0.5*colorFac )

        #append this color value to a running list
        myArray.append( int(myRGB[2]) ) #blue first
        myArray.append( int(myRGB[1]) )    #The green
        myArray.append( int(myRGB[0]) )  #Red is last

#Now I reduce the lists to process 0!!
masterString = mpi.reduce( myArray.tostring(), mpi.SUM, 0 )

#Tell user that we're done
message = "process " + str(mpi.rank) + " done with computation!!"
print message

#Process zero does the file writing
if mpi.rank == 0:
    masterArray = array('B')
    masterArray.fromstring(masterString)

    #Write a BMP header
    myBMPHeader = makeBMPHeader( bmpSize[0], bmpSize[1] )
    print "Header length is ", len(myBMPHeader)
    print "BMP size is ", bmpSize
    print "Data length is ", len(masterString)

    #Open the output file and write to the BMP
    outFile = open( 'output.bmp', 'w' )
    outFile.write( myBMPHeader )
    outFile.write( masterString )
    outFile.close()
```

図4.3.2-2 PyMPIによる並列計算の実行結果

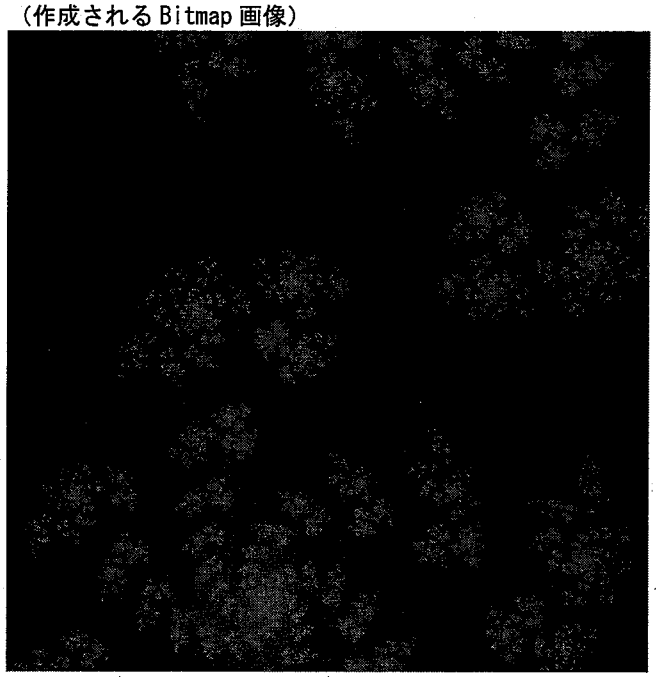
```
[yokoyama@ofbrf96 examples]$ date; mpirun -np 1 /usr/local/bin/pyMPI fractal.py; date
2003年  2月 27日 木曜日 10:41:29 JST
Starting computation (groan)

process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
2003年  2月 27日 木曜日 10:42:19 JST
[yokoyama@ofbrf96 examples]$ date ; mpirun -np 2 /usr/local/bin/pyMPI fractal.py ; date
2003年  2月 27日 木曜日 10:42:27 JST
Starting computation (groan)

process 1 done with computation!!
process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
2003年  2月 27日 木曜日 10:42:56 JST
[yokoyama@ofbrf96 examples]$ date ; mpirun -np 3 /usr/local/bin/pyMPI fractal.py ; date
2003年  2月 27日 木曜日 10:43:03 JST
Starting computation (groan)

process 1 done with computation!!
process 2 done with computation!!
process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
2003年  2月 27日 木曜日 10:43:25 JST
[yokoyama@ofbrf96 examples]$ date ; mpirun -np 4 /usr/local/bin/pyMPI fractal.py ; date
2003年  2月 27日 木曜日 10:43:31 JST
Starting computation (groan)

process 1 done with computation!!
process 2 done with computation!!
process 3 done with computation!!
process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
2003年  2月 27日 木曜日 10:43:51 JST
[yokoyama@ofbrf96 examples]$ date ; mpirun -np 5 /usr/local/bin/pyMPI fractal.py ; date
2003年  2月 27日 木曜日 10:43:55 JST
Starting computation (groan)

process 1 done with computation!!
process 2 done with computation!!
process 3 done with computation!!
process 4 done with computation!!
process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
```

（次ページへ続く）

（前ページからの続き）

```
2003年  2月 27日 木曜日 10:44:10 JST
[yokoyama@ofbrf96 examples]$ date ; mpirun -np 5 /usr/local/bin/pyMPI fractal.py ; date
2003年  2月 27日 木曜日 10:44:18 JST
Starting computation (groan)

process 1 done with computation!!
process 2 done with computation!!
process 3 done with computation!!
process 4 done with computation!!
process 0 done with computation!!
Header length is  54
BMP size is  (400, 400)
Data length is  480000
2003年  2月 27日 木曜日 10:44:33 JST
```

（作成されるBitmap画像）



第5章　結　　言

第５章　結言

高速炉開発に関連する機器構造解析、炉物理解析の分野を対象として、次世代解析システムのプロトタイプ作成による検討を実施した。これまでに行った要素技術の調査結果から、３種類の方法で、(1)多言語（SoftWIRE.NET 、Visual Basic .NET、Fortran）、(2)Fortran90、(3)Python を採用してプロトタイプを作成し、課題の検討を実施した。以下に、各プログラミング言語を用いたプロトタイプ作成の検討結果をまとめる。

(1)　多言語によるプロトタイプ作成

高速炉の熱過渡応力解析に関しては、従来、すべて Smalltalk で開発されていた解析コードを Visual Basic .NET とビジュアルプログラミングツールである SoftWIRE の組み合わせた開発環境を用いて再構築を行った。更に、動特性解析部分については、Fortran で書かれた既存の解析コードを部品化し、Visual Basic .NET と Fortran の複合言語によるシステム化の可能性について検討した。この結果、Fortran で書かれた既存の解析コードは、新しい解析コードシステム上で部品化することに成功した。既存の解析コードの機能は、ひとつのアイコンとして表現され、Visual Basic .NET で作成された他の部品との結合も可能であることを確認することができた。また、従来の解析コードとの計算結果の比較も実施し、プロトタイプコードが物理的にも正しい結果が得られていることも確認できた。

(2)　Fortran90 によるプロトタイプ作成

Fortran90 を簡単な炉物理計算問題に適用するためのプログラムを作成し、Fortran77 に基づくプログラムと比較した。その結果、Fortran90 は Fortran77 に比べ、メソッド間で共通変数の冗長定義をせずに済むこと、構造体概念を用いた変数表記ができるため、変数の意味がそれを見ただけで容易に理解しやすい、などの利点を有し、プログラムの保守・管理作業に要する負荷を低減するのに有効であることがわかった。

今後、本プログラムに自己遮蔽因子を始めとする計算機能（メソッド）を記述してプログラムとしての動作・機能を確認するとともに、本報で示したクラスに「継承」の概念を適用して階層化を図ることで、Fortran90 のもつ特長をさらに具体的に示すことが期待される。

(3)　Python によるプロトタイプ作成

(2)と同様に、高速炉の炉物理計算を対象として、Python を用いた次世代解析システムのプロトタイプ作成による検討を実施した。Python のモジュール化機能を用いることにより、必要な機能やデータを構造化して分類し、必要に応じてインポートして使うという「ツールボックス」のイメージでシステム化できる見込みを得た。また、各ツール（部品）のサ

ンプルとして、解析手法（炉心計算、燃焼計算）や解析データ（体系データ、核反応断面積）を取り扱うモジュールも作成した。この結果から、解析上必要となる体系データや核反応断面積データ等に対する、作成、保存、利用、可視化といった操作を、一貫してPython上で制御できる見込みを得た。

また、一方で、炉物理解析で重要、かつ、量の多くなる核反応断面積データを、XML形式やPickle形式に変換して取り扱うことが現実的な範囲で可能であることも確認した。また、Pythonの演算子オーバーロードの機能を用いることにより、断面積データや感度係数に対する演算を、従来に比べて高度に抽象化できることも示した。

なお、本報告書に記載した内容は、平成14年度の作業分であり、今後もプロトタイプ作成による検討を継続する予定である。構造解析の分野は、従来のコードがオブジェクト指向設計に基づいていることもあり、プロトタイプであるものの、既にある程度実用性のあるレベルに達しているが、今後の機能の拡充とともに必要となる分析・設計の反復を行う予定である。一方、炉物理分野については、オブジェクト指向分析・設計の適用のが初めての試みであり、今後、デザインパターンの適用なども考慮に入れて、様々な角度から分析・設計を継続していく必要がある。

参 考 文 献

参考文献

2-1 仲井 悟、岩崎 隆 他、「モジュール型プラント動特性解析コード Super-COPD コード使用説明書」、PNC TN9520 89-001 (1989)

2-2 堺 公明、「モジュール型プラント動特性解析コード MSG-COPD コード説明書マニュアル」、JNC TN9520 2002-003 (2002)

2-3 笠原 直人 他、「過渡熱応力リアルタイムシミュレーションコード PARTS の開発 (3) 構造計算部品 PARTS-STRESS の開発」、PNC TN9410 96-100 (1996)

2-4 井上 正明、神保 雅一 他、「系統熱過渡設計用熱流動・構造統合解析手法の開発 調査報告書」、JNC TJ9420 2001-009 (2002)

2-5 田中 良彦、笠原 直人 他、「熱過渡応力解析への Green 関数法の適用範囲の拡張 (1)流速変化を伴う 2 流体に対する応答の Green 関数による記述」、JNC TN9400 2001-121 (2002)

2-6 田中 良彦、笠原 直人、「熱過渡応力解析への Green 関数法の適用範囲の拡張 (2) ナトリウム冷却炉の中間熱交換機内部構造に生ずる熱過渡応力の記述」、JNC TN9400 2002-038 (2002)

2-7 横山 賢治、細貝 広視 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) 課題及び要素技術の調査」、JNC TN9420 2002-004 (2002)

3-1 横山 賢治、細貝 広視 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) 課題及び要素技術の調査」、JNC TN9420 2002-004 (2002)

4-1 Python Programming Language, http://www.python.org/

4-2 SLOME, http://www.opencascade.org/SALOME/

4-3 Weston M. Stacey, "Nuclear Reactor Physics", John Wiley & Sons, Inc., ISBN 0-471-39127-1 (2001)

4-4 Eric Gamma, Richard Helm 他、本位田 真一、吉田 和樹 監訳、「オブジェクト指向における再利用のためのデザインパターン (改訂版)」、ソフトバンク パブリッシング (1999)

4-5 A. G. Croff, "A User Manual for the ORIGEN2 Computer Code", ORNL/TM-7175 (1980)

4-6 ScyPy - Scientific tools for Python, http://www.scipy.org/

4-7 T. B. Fowler, et al., "Nuclear Reactor Analysis Code: CITATION," ORNL TM-2496, Rev.2 (1971)

4-8 Numerical Python, http://www.pfdubois.com/numpy/

4-9 Scientific Python, http://starship.python.net/crew/hinsen/scientific.html

4-10 Qt, http://www.trolltech.com/products/qt/

4-11 VTK, http://public.kitware.com/VTK/

4-12 OpenCASCADE, http://www.opencascade.com/

4-13 H. Takano and Y. Ishiguro, "Production and Benchmark Tests of Fast Reactor Group Constant Set JFS-3-J2" JAERI-M 82-135 (1982)

4-14 原 昭浩、竹田 敏一 他、「SAGEP：一般化摂動理論に基づく二次元感度解析コード」、JAERI-M 84-027（1984）

4-15 V. Savikko, "Design Patterns in Python," Proc of the 6th International Python Confference, San Jose, California (1997) http://www.python.org/workshops/1997-10/proceedings/

4-16 B. Eckel, "Thinking in Python - Rev. 0.1.2 -," MindView, Inc. (2001) http://www.mindview.net/Books/TIPython/

4-17 結城 浩、「Java 言語で学ぶデザインパターン入門」、ソフトバンク パブリッシング、ISBN4-7973-1646-2（2001）

4-18 P. Miller, "pyMPI – An introduction to parallel Python using MPI," UCRL-WEB-150152 (2002) http://pympi.sourceforge.net/

4-19 M. Fowler 著、児玉 公信 他 訳、「リファクタリング―プログラムの体質改善テクニック」、ピアソン・エデュケーション、ISBN4-89471-228-8（2000）

付録A　Pythonによるプロトタイプ作成の詳細

付録A　Pythonによるプロトタイプ作成の詳細

図 A-1 GroupData.py

```
#!/usr/local/bin/python

"""
Definition of the energy group data
"""

from Numeric import *
from Nuclide import *
import cPickle
import math
import Gnuplot
import Mixin

def indexWithinEps(list, value, eps=1.0E-6):
    index = 0
    for element in list:
        if abs(element-value)/value < eps:
            return index
        index += 1

def prepareGraphData(data):
    ret = []
    list = data[:]
    list.append(data[-1])
    for i in list:
        if (i.__str__() == "nan" or
            i.__str__() == "inf"):
            ret.append(0.0)
        else:
            ret.append(i)
    return ret

class GroupData(Mixin.SerializableAndVisible):
    def __init__(self,
                 ngroup          = 1,
                 boundaries      = [1E+6, 1E-5],
                 name            = "Undefined",
                 name_boundaries = "Undefined",
                 name_data       = "Undefined",
                 unit_boundaries = "arbitrary",
                 unit_data       = "arbitrary"):
        self.ngroup     = int(ngroup)
        self.boundaries = array(boundaries, Float)
        self.data       = zeros(ngroup, Float)

        self.name            = str(name)
        self.name_boundaries = str(name_boundaries)
        self.name_data       = str(name_data)
        self.unit_boundaries = str(unit_boundaries)
        self.unit_data       = str(unit_data)

        assert self.boundaries.shape == (self.ngroup+1,)
        assert self.data.shape       == (self.ngroup,)

    def setData(self, data):
        if len(data) == self.ngroup:
            self.data = array(data, Float)
        else:
            raise TypeError

    def setNames(self, name,
                 name_boundaries, name_data,
                 unit_boundaries, unit_data):
        self.name            = str(name)
        self.name_boundaries = str(name_boundaries)
        self.name_data       = str(name_data)
        self.unit_boundaries = str(unit_boundaries)
        self.unit_data       = str(unit_data)

    def __add__(self, other):
        if self.isSameBoundaries(other):
            ret = GroupData(self.ngroup, self.boundaries)
            ret.data = self.data + other.data
            ret.setNames(self.name,
                         self.name_boundaries, self.name_data,
                         self.unit_boundaries, self.unit_boundaries)
            return ret
        else:
            raise TypeError

    def __sub__(self, other):
        if self.isSameBoundaries(other):
            ret = GroupData(self.ngroup, self.boundaries)
```

```
        ret.data = self.data - other.data
        ret.setNames(self.name,
                     self.name_boundaries, self.name_data,
                     self.unit_boundaries, self.unit_boundaries)
        return ret
    else:
        raise TypeError

def __mul__(self, other):
    if self.isSameBoundaries(other):
        ret = GroupData(self.ngroup, self.boundaries)
        ret.data = self.data * other.data
        return ret
    else:
        raise TypeError

def __div__(self, other):
    if self.isSameBoundaries(other):
        ret = GroupData(self.ngroup, self.boundaries)
        ret.data = self.data / other.data
        return ret
    else:
        raise TypeError

def condense(self, weight, condensed):
    if (self.isSameBoundaries(weight) and
        self.boundaries[0]  == condensed.boundaries[0] and
        self.boundaries[-1] == condensed.boundaries[-1]):

        indices = []
        for boundary in condensed.boundaries.tolist():
            indices.append(indexWithinEps(self.boundaries.tolist(),
                                          boundary, eps=1.0E-5))
        indices.reverse()

        newGroupData = [0.0 for i in range(condensed.ngroup)]
        jmax = indices.pop()
        for i in range(condensed.ngroup):
            jmin = jmax
            jmax = indices.pop()
            sum = 0.0
            for j in range(jmin, jmax):
                newGroupData[i] += self.data[j] * weight.data[j]
                sum += weight.data[j]
            newGroupData[i] /= sum
        ret = GroupData(condensed.ngroup, condensed.boundaries,
                        self.name,
                        self.name_boundaries, self.name_data,
                        self.unit_boundaries, self.unit_boundaries)
        ret.setData(newGroupData)
        return ret
    else:
        raise TypeError

def lethargy(self):
    ret = GroupData(self.ngroup, self.boundaries)
    for i in xrange(ret.ngroup):
        ret.data[i] = math.log(ret.boundaries[i]/ret.boundaries[i+1])
    return ret

def isSameBoundaries(self, other):
    return (type(self) == type(other) and
            self.ngroup == other.ngroup and
            allclose(self.boundaries, other.boundaries))

def visualize(self, logscalex=1, logscaley=1,
              xrange=None, yrange=None):

    x = self.boundaries.tolist()
    y = prepareGraphData(self.data.tolist())

    g = Gnuplot.Gnuplot(persist=1)
    g.title(self.name)
    g.xlabel(self.name_boundaries+"("+self.unit_boundaries+")")
    g.ylabel(self.name_data+"("+self.unit_data+")")
    if logscalex: g("set logscale x")
    if logscaley: g("set logscale y")
    if xrange is not None: g("set xrange "+xrange)
    if yrange is not None: g("set yrange "+yrange)
    d = Gnuplot.Data(x, y,
                     title = self.name,
                     with = "fsteps")
    g.clear()
    g.plot(d)
    wait_a_bit()
```

```
class GroupDataSet(Mixin.SerializableAndVisible):
    def __init__(self,
                 ngroup = 1,
                 boundaries = [1E+6, 1E-5],
                 name            = "Undefined",
                 name_boundaries = "Undefined",
                 name_data       = "Undefined",
                 unit_boundaries = "arbitrary",
                 unit_data       = "arbitrary"):

        self.name            = str(name)
        self.name_boundaries = str(name_boundaries)
        self.name_data       = str(name_data)
        self.unit_boundaries = str(unit_boundaries)
        self.unit_data       = str(unit_data)

        self.ngroup      = int(ngroup)
        self.boundaries  = array(boundaries)
        self.fission     = GroupData(self.ngroup, self.boundaries)
        self.capture     = GroupData(self.ngroup, self.boundaries)
        self.elastic     = GroupData(self.ngroup, self.boundaries)
        self.inelastic   = GroupData(self.ngroup, self.boundaries)
        self.n2n         = GroupData(self.ngroup, self.boundaries)
        self.fisspec     = GroupData(self.ngroup, self.boundaries)
        self.muvalue     = GroupData(self.ngroup, self.boundaries)
        self.nuvalue     = GroupData(self.ngroup, self.boundaries)

    def _calc(self, other, operator=""):
        ret = GroupDataSet(self.ngroup, self.boundaries,
                           "("+self.name+operator+other.name+")")
        ret.fission   = eval("self.fission"+operator+"other.fission")
        ret.capture   = eval("self.capture"+operator+"other.capture")
        ret.elastic   = eval("self.elastic"+operator+"other.elastic")
        ret.inelastic = eval("self.inelastic"+operator+"other.inelastic")
        ret.n2n       = eval("self.n2n"+operator+"other.n2n")
        ret.fisspec   = eval("self.fisspec"+operator+"other.fisspec")
        ret.muvalue   = eval("self.muvalue"+operator+"other.muvalue")
        ret.nuvalue   = eval("self.nuvalue"+operator+"other.nuvalue")
        return ret

    def setNames(self, name,
                 name_boundaries, name_data,
                 unit_boundaries, unit_data):
        self.name            = str(name)
        self.name_boundaries = str(name_boundaries)
        self.name_data       = str(name_data)
        self.unit_boundaries = str(unit_boundaries)
        self.unit_data       = str(unit_data)

    def __add__(self, other):
        if self.isSameBoundaries(other):
            ret = self._calc(other, "+")
            return ret
        else:
            raise TypeError

    def __sub__(self, other):
        if self.isSameBoundaries(other):
            ret = self._calc(other, "-")
            return ret
        else:
            raise TypeError

    def __mul__(self, other):
        if self.isSameBoundaries(other):
            ret = self._calc(other, "*")
            return ret
        else:
            raise TypeError

    def __div__(self, other):
        if self.isSameBoundaries(other):
            ret = self._calc(other, "/")
            return ret
        else:
            raise TypeError

    def condense(self, weight, condensed):
        if (self.isSameBoundaries(weight) and
            self.boundaries[0]  == condensed.boundaries[0] and
            self.boundaries[-1] == condensed.boundaries[-1]):

            weight_one = GroupData(self.ngroup, self.boundaries)
            weight_one.setData(ones(self.ngroup))

            ret = GroupDataSet(condensed.ngroup, condensed.boundaries, self.name)
```

```
        ret.setNames(self.name,
                     self.name_boundaries, self.name_data,
                     self.unit_boundaries, self.unit_data)
        ret.fission   = self.fission.condense(weight, condensed)
        ret.capture   = self.capture.condense(weight, condensed)
        ret.elastic   = self.elastic.condense(weight, condensed)
        ret.inelastic = self.inelastic.condense(weight, condensed)
        ret.n2n       = self.n2n.condense(weight, condensed)
        ret.fisscpec  = self.fisspec.condense(weight_one, condensed)
        ret.muvalue   = self.muvalue.condense(weight_one, condensed)
        ret.nuvalue   = self.nuvalue.condense(weight_one, condensed)
        return ret
    else:
        raise TypeError

def isSameBoundaries(self, other):
    return (type(self) == type(other) and
            self.ngroup == other.ngroup and
            allclose(self.boundaries, other.boundaries))
```


図A-2 Mixin.py

```
"""
Definitions for the basic Mix-in Classes
"""

import cPickle

class SerializableAndVisible:
    def show_header(self):
        print "=====", self.__class__.__name__, "(BEGIN) ====="

    def show_footer(self):
        print "=====", self.__class__.__name__, "(END) ====="

    def show(self, verbose=0, header=1, footer=0):
        if header: self.show_header()
        keys = self.__dict__.keys()
        keys.sort()
        for key in keys:
            print key, "=", self.__dict__[key]
        if footer: self.show_header()

    def show2(self, verbose=0):
        self.show_header()
        keys = self.__dict__.keys()
        keys.sort()
        for key in keys:
            if hasattr(self.__dict__[key], "show"):
                print key, "=", self.__dict__[key]
                print self.__dict__[key].show(header=0, footer=0)
            else:
                print key, "=", self.__dict__[key]
        self.show_footer()

    def visualize(self):
        raise NotImplementedError

    def dump(self, filename, binary=1):
        print "Dumping", self.__class__.__name__, "object to Pickled file..."
        fh = open(filename, "w")
        cPickle.dump(self, fh, binary)
```

```
def load(self, filename, binary=1):
    print "Loading", self.__class__.__name__, "object from Pickled file..."
    fh = open(filename, "rb")
    if not binary: fh = open(filename, "r")
    return cPickle.load(fh)
```


図A-3 Nuclide.py

```
#!/usr/local/bin/python

"""
Definition of nuclides
"""


import exceptions

elements = [
          'H' ,'He','Li','Be','B' ,'C' ,'N' ,'O' ,'F' ,'Ne',
          'Na','Mg','Al','Si','P' ,'S' ,'Cl','Ar','K' ,'Ca',
          'Sc','Ti','V' ,'Cr','Mn','Fe','Co','Ni','Cu','Zn',
          'Ga','Ge','As','Se','Br','Kr','Rb','Sr','Y' ,'Zr',
          'Nb','Mo','Tc','Ru','Rh','Pd','Ag','Cd','In','Sn',
          'Sb','Te','I' ,'Xe','Cs','Ba','La','Ce','Pr','Nd',
          'Pm','Sm','Eu','Gd','Tb','Dy','Ho','Er','Tm','Yb',
          'Lu','Hf','Ta','W' ,'Re','Os','Ir','Pt','Au','Hg',
          'Tl','Pb','Bi','Po','At','Rn','Fr','Ra','Ac','Th',
          'Pa','U' ,'Np','Pu','Am','Cm','Bk','Cf','Es','Fm',
          'Md','No','Lr']

#
#  Nuclide
#
class Nuclide:
    """このクラスは核種を表します。引数には、原子番号（第１引数）と質量数
    （第２引数）を渡します。¥n

    例：Pu-239 のインスタンス生成と名前の表示/n
        >>> p9 = Nuclide(94,239)¥n
        >>> print p9/n
        Pu-239/n

    核種が核変換した場合や中性子と反応した場合に、どの核種になるかと
    いう情報を持っています。¥n

    例：U-238 から Pu-239 への核変換/n
        >>> u8 = nuclide.Nuclide(92,238)¥n
        >>> print u8/n
        U-238/n
```

```
    >>> print u8.capture().betaDecay().betaDecay()¥n
    Pu-239/n

将来、各解析コードで利用される ID 番号を返すメソッドを実装する予定です。¥n
"""

def __init__(self, name = None, anumber = 0, amass = 0,
             meta = 0, nat = 0, fp = 0):
    if name != None:
        self.nat  = 0
        self.meta = 0
        self.fp   = 0

        if name.count("-nat."):
            self.nat = 1
            name = name.replace("-nat.", "-0")
        if name[-1] == "m":
            self.meta = 1
            name = name[:-1]
        if name.count("FP"):
            self.fp = 1
            name = name[:-2]

        self.anumber = elements.index(name.split('-')[0])+1
        self.amass   = int(name.split('-')[1])

    else:
        self.anumber = anumber
        self.amass   = amass
        self.meta    = meta
        self.nat     = nat
        self.fp      = fp

def __repr__(self):
    return self.nameOfNuclide()

def nameOfElement(self):
    "元素名（例：Pu-239 の場合、Pu）を返します"
    return elements[self.anumber-1]

def nameOfNuclide(self):
    "核種の名前（例：Pu-239）を返します"
    ret = self.nameOfElement() + '-' + str(self.amass)
    if self.meta:
        ret = ret + "m"
    if self.nat:
        ret = ret.replace("-0", "-nat.")
    if self.fp:
        ret = ret + "FP"
    return ret

def show(self, verbose=0):
    return self.nameOfNuclide()

def alphaDecay(self):
    "α崩壊後の核種（Nuclide）のインスタンスを返します。"
    return Nuclide(anumber=self.anumber-2, amass=self.amass-4)

def betaDecay(self):
    "β崩壊後の核種（Nuclide）のインスタンスを返します。"
    return Nuclide(anumber=self.anumber+1, amass=self.amass)

def n2n(self):
    "(n,2n)反応後生成する核種（Nuclide）のインスタンスを返します。"
    return Nuclide(anumber=self.anumber-1, amass=self.amass-2)

def capture(self):
    "中性子捕獲反応後生成する核種（Nuclide）のインスタンスを返します。"
    return Nuclide(anumber=self.anumber, amass=self.amass+1)
```

図 A-4 Wrappers.py

```
#!/usr/local/bin/python

"""
Wrapper utilities
"""

from Numeric import *

import Neutronics.NuclearData.JFS3.JFS3utils as JFS3utils
from Neutronics.Sensitivity.SAGEP.SAGEPutils import *

from GroupData import GroupData, GroupDataSet

###
### Functions
###
def JFS3toGroupDataSet(jfs, name=""):
    assert isinstance(jfs, JFS3utils.GroupConstantSet)
    ret = GroupDataSet(jfs.sig1d.energy_group,
                       jfs.sig1d.energy_boundaries)
    ret.fission.setData(jfs.sig1d.fission)
    ret.capture.setData(jfs.sig1d.capture)
    ret.elastic.setData(jfs.sig1d.elastic)
    ret.inelastic.setData(jfs.sig1d.inelastic)
    ret.n2n.setData(jfs.sig1d.n2n)
    ret.muvalue.setData(jfs.sig1d.muvalue)
    ret.nuvalue.setData(jfs.sig1d.muvalue)

    ret.setNames            ("JFS3 Library",              "Neutron Energy", "Cross Section",
"eV", "barn")
    ret.fission.setNames    ("Fission",             "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.capture.setNames    ("Capture",             "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.elastic.setNames    ("Elastic Scatering",   "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.inelastic.setNames("Inelastic Scattering", "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.n2n.setNames        ("(n,2n) Reaction",     "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.muvalue.setNames    ("Mu Value",            "Neutron Energy", "Cross Section", "eV",
"barn")
    ret.nuvalue.setNames    ("Nu Value",            "Neutron Energy", "Cross Section", "eV",
"barn")

    return ret

def SAGEPtoGroupDataSet(senssset, name=""):
##     assert isinstance(sensset, SAGEPutils.SensitivitySet)
    assert isinstance(senssset, SensitivitySet)
    ret = GroupDataSet(18,
                       JFS3utils.jfs18g,
                       name)
    ret.fission.setData(senssset.fission.sens[:])
    ret.capture.setData(senssset.capture.sens[:])
    ret.elastic.setData(senssset.elastic.sens[:])
    ret.inelastic.setData(senssset.inelastic.sens[:])
    ret.n2n.setData(senssset.n2n.sens[:])
    ret.muvalue.setData(senssset.muvalue.sens[:])
    ret.nuvalue.setData(senssset.nuvalue.sens[:])

    ret.fission.setNames    ("Fission",             "Neutron Energy", "Sensitivity
Coefficient", "eV", "barn")
    ret.capture.setNames    ("Capture", "Neutron Energy", "Sensitivity Coefficient", "eV",
"barn")
    ret.elastic.setNames    ("Elastic Scattering", "Neutron Energy", "Sensitivity
Coefficient", "eV", "barn")
    ret.inelastic.setNames("Inlastic Scattering", "Neutron Energy", "Sensitivity
Coefficient", "eV", "barn")
    ret.n2n.setNames        ("(n,2n) Reaction", "Neutron Energy", "Sensitivity Coefficient",
"eV", "barn")
    ret.muvalue.setNames    ("Mu Value", "Neutron Energy", "Sensitivity Coefficient", "eV",
"barn")
    ret.nuvalue.setNames    ("Nu Value", "Neutron Energy", "Sensitivity Coefficient", "eV",
"barn")

    return ret

if __name__ == "__main__":
    pass
```

図 A-5 Diffusion1D.py

```
#!/usr/local/bin/python

import exceptions
## from Numeric import *
## import Gnuplot
from scipy import *

class MacroscopicCrossSection:
    def __init__(self, d, sigma_a, nusigma_f):
        self.d         = d
        self.sigma_a   = sigma_a
        self.nusigma_f = nusigma_f

    def show(self):
        print "***** MacroscopicCrossSection *****"
        print "Diffusion Coefficient:", self.d
        print "Sigma_absorption:     ", self.sigma_a
        print "Nu * Sigma_fission:   ", self.nusigma_f

class Geometry1D:
    def __init__(self, type, mesh, delta, macroxs):
        types = {"slab": 0, "cylindrical": 1, "spherical": 2}
        self.type    = types[type]
        self.mesh    = mesh
        self.delta   = delta
        self.macroxs = macroxs

    def show(self):
        print "***** Geometry1D *****"
        print "No. of Meshes:       ", self.mesh
        print "Intervals of Meshes: ", self.delta
        for i in self.macroxs:
            i.show()

class Diffusion1D:
    def __init__(self, geom1d):
        self.c       = geom1d.type
        self.Delta   = geom1d.delta
        self.I       = geom1d.mesh + 2
        self.D       = [0.0]+[geom1d.macroxs[i].d
                              for i in range(geom1d.mesh)]+[0.0]
        self.Sigma_a = [0.0]+[geom1d.macroxs[i].sigma_a
                              for i in range(geom1d.mesh)]+[0.0]
        self.f       = [0.0]+[geom1d.macroxs[i].nusigma_f
                              for i in range(geom1d.mesh)]+[0.0]

        self.lmbd = 1.0
        self.eps  = 1.0E+99
        self.gen  = 0

        self.phi  = [1.0 for i in range(self.I)];
        self.phi[0] = 0.0
        self.phi[-1] = 0.0

        self.S    = [self.f[i]*self.phi[i] for i in range(self.I)]

    def a(self, i, j):
        if i == j-1:
            ret = -1.0/2.0 * (self.D[i] + self.D[i-1]) ¥
                  / self.Delta**2.0 * (1 - self.c/(2*i-1))
            return ret
        elif i == j:
            ret =  self.Sigma_a[i] + 1.0/2.0 * ¥
                  ((self.D[i-1] + 2*self.D[i] + self.D[i+1]) / self.Delta**2.0)
            return ret
        elif i == j+1:
            ret = -1.0/2.0 * ((self.D[i+1] + self.D[i]) ¥
                              / self.Delta**2.0) * (1 + self.c/(2*i-1))
            return ret
        else:
            raise exeptions.Exception, 'a(i,j): Not Defined'

    def A(self, i):
        if i == 1:
            ret = self.a(1,2) / self.a(1,1)
            return ret
        elif i > 1:
            ret = self.a(i,i+1)/(self.a(i,i) - self.a(i,i-1) * self.A(i-1))
            return ret
        else:
            raise exceptions.Exception, 'A(i): Not Defined'

    def alpha(self, i):
        if i == 1:
```

```
        ret = self.S[1] / self.a(1,1)
        return ret
    elif i > 1:
        ret = (self.S[i] - self.a(i,i-1)*self.alpha(i-1)) ¥
              / (self.a(i,i) - self.a(i,i-1)*self.A(i-1))
        return ret
    else:
        raise exceptions.Exception, 'alpha(i): Not Defined'

def nextgen(self):
    phi2 = self.phi[:]
    S2   = self.S[:]
    ret1 = 0.0
    ret2 = 0.0
    for i in range(1, self.l-1):
       phi2[i] = -self.A(i)*self.phi[i+1] + self.alpha(i)
       ret1 += self.f[i] *     phi2[i] * self.Delta
       ret2 += self.f[i] * self.phi[i] * self.Delta / self.lmbd
    lmbd2 = ret1/ret2
    eps2  = abs((lmbd2 - self.lmbd)/self.lmbd)

    for i in range(1, self.l-1):
        S2[i] = 1.0/lmbd2 * self.f[i] * phi2[i]

    self.phi  = phi2
    self.S    = S2
    self.lmbd = lmbd2
    self.eps  = eps2
    self.gen  += 1

def show(self):
    print "***** Diffusion1D *****"
    print "Neutron Flux,          phi =", self.phi
    print "Neutron Source,          S =", self.S
    print "Mutiplication Factor, keff =", self.lmbd
    print "Convergence Condition, eps =", self.eps
    print "Generation,            gen =", self.gen

def show_keff(self):
    print "gen=", self.gen, "keff=", self.lmbd, "eps=", self.eps

def visualize(self):
    x = arange(0, self.l)
    y = array(self.phi)
    gplt.plot(x, y,
              "title 'Neutron Flux (gen=" + str(self.gen) +")' with lines")

def visualize2(self):
    x = arange(0, self.l)
    y = array(self.phi)
    g = Gnuplot.Gnuplot()
    d1 = Gnuplot.Data(x, y,
                      title = 'Neutron Flux',
                      with = 'line')
    g.title('Neutron Flux')
    g.plot(d1)

if __name__ == '__main__':

    xs_u5 = MacroscopicCrossSection(1/(3*0.765), 9.24E-2, 0.145)
    xs_fe = MacroscopicCrossSection(1/(3*1.15),  0.222,   0.0  )
    ## xs_u5.show()
    ## xs_fe.show()

    geom1d = Geometry1D("slab", mesh=19, delta=0.6,
                        macroxs = [xs_fe for i in range(3)]
                        + [xs_u5 for i in range(13)]
                        + [xs_fe for i in range(3)])
    ## geom1d.show()

    d1d = Diffusion1D(geom1d)
    ## d1d.show()

    for i in range(50):
        d1d.nextgen()
        ## d1d.show()
        d1d.show_keff()
        d1d.visualize()
        if d1d.eps < 1.0E-6:
            print "Converged!"
            break
```

図 A-6 Burnup.py

```
#!/usr/local/bin/python

"""Solvers for Burnup equation"""

from scipy import *
from math import log

class BurnupCalculation:
    def __init__(self):
        pass

    def setMatrix(self, matrix):
        self.matrix = array(matrix)

    def setInitialVector(self, initvec):
        self.initvec = array(initvec)
        self.retvec = array(initvec)

    def setBurnupTime(self, dtime):
        self.dtime = dtime

    def solve(self):
        ## print transpose(N0)
        ## print M
        ## print M*delta_t
        ## print linalg.expm(M*delta_t)
        self.retvec = matrixmultiply(
            linalg.expm(self.matrix * self.dtime),
            transpose(self.initvec)
            )

    def show(self, comments):
        print comments
        print self.retvec
```

図 A-7 Core.py

```
#!/usr/local/bin/python

from Numeric import *
import copy

class SimpleMaterial:
    """
    Simplematerial is a class representing the number desities
    like ERANOS's MATERIAU_SIMPLE
    """
    def __init__(self, name, numberDensities):
        self.name = name
        self.nds  = numberDensities

    def show(self):
        print self.name
        print self.nd

class Medium:
    """
    Medium is a class representing the medium
    (for instans, homogenized region)
    like ERANOS's MILIEU
    """
    def __init__(self):
        self.name = ''
        self.nds   = []

    def __str__(self):
        return self.name

    def madeFromSimpleMaterial(self, name, simpleMaterial):
        self.name = name
        self.nds  = copy.copy(simpleMaterial.nds)

    def mixedFromSimpleMaterials(self, name, simpleMaterials, weights):
        self.name = name
        if len(simpleMaterials) == len(weights):
            raise Exception
        pass
```

```
    def show(self):
        print "Name of Material:", self.name
        ## print self.nd
        for nd in self.nds:
            print "    ", nd, self.nds[nd]

class HexagonalAssembly:
    def __init__(self, name, pitch, regions, mediums):
        self.name      = name
        self.height    = max(regions)
        self.pitch     = pitch
        self.regions   = regions
        self.mediums   = mediums

    def __str__(self):
        return self.name

    def toMesh(self):
        ret = Mesh1DX(len(self.regions))
        ret.sizes = []
        for index in range(len(self.regions)-1):
            ret.sizes.append(self.regions[index+1]-self.regions[index])
        ret.names = addUnique(ret.names, [medium.name for medium in self.mediums])
        i = 0; adict = {}
        for name in ret.names:
            if name not in adict.keys(): adict[name] = i
            i += 1
        ret.mapdata = [adict[medium.name] for medium in self.mediums]
        return ret

    def show(self):
        print '*** HexagonalAssembly ***'
        print 'Name:',     self.name
        print 'Height:',   self.height
        print 'Pitch:',    self.pitch
        print 'Regions:',  self.regions
        print 'Mediums:',  self.mediums

class ConventionalFastReactorCore:
    def __init__(self, hexmap, sas):
        self.hexmap = hexmap
        self.sas = sas

    def toMesh2DHex(self):
        ret = Mesh2DHex(len(self.hexmap), len(self.hexmap[0]))

    def toMesh3DHexZ(self):
        regs = []
        for sa in self.sas:
            addUnique(regs, sa.regions)
        regs.sort()

    def show(self):
        print "*** ConventionalFastReactorCore ***"
        print array(self.hexmap)
        print self.sas

    def visualize(self):
        pass

if __name__== '__main__':
    pass
```

図 A-8 CITATIONutils.py

```
#!/usr/local/bin/python

import re
import os

from Numeric import *
from Scientific.IO.FortranFormat import FortranFormat, FortranLine

from Tkinter import *
from Pmw import ScrolledCanvas

GeometryType = [( 1, "1D-Slab"),
                ( 2, "1D-Pole"),
                ( 3, "1D-Sphere"),
                ( 6, "2D-XY"),
                ( 7, "2D-RZ"),
                ( 8, "2D-RTheta"),
                ( 9, "2D-Hex"),
                (10, "2D-Tri"),
                (11, "3D-XYZ"),
                (12, "3D-RZTheta"),
                (13, "3D-HexZ"),
                (14, "3D-TriZ")]

def readSec003(filename):
    f = open(filename, "r")
    inSec003 = 0
    nuac5 = 0
    geomtype = None
    for line in f.readlines():
        if inSec003:
           nuac5 = FortranLine(line, FortranFormat("(24I3)"))[4]
           break
        if line[:3] == "003":
            inSec003 = 1

    for num, name in GeometryType:
        if num == nuac5:
            geomtype = name
    return geomtype

def readSec004(filename):
    f = open(filename, "r")
    inSec004 = 0
    numi = []; numj = []; numk = []
    widthi = []; widthj = []; widthk = []
    inI = 0; inJ = 0; inK = 0
    for line in f.readlines():
        if inSec004:
            if inI:
                data = FortranLine(line, FortranFormat("6(I3,E9.0)"))
                numi.extend  ([data[0],data[2],data[4],data[6],data[8],data[10]])
                widthi.extend([data[1],data[3],data[5],data[7],data[9],data[11]])
                idx = -1
                try:
                    idx = numi.index(0)
                except:
                    idx = -1
                if idx != -1:
                    numi   = numi[:idx]
                    widthi = widthi[:idx]
                    inI = 0
                    inJ = 1
            elif inJ:
                data = FortranLine(line, FortranFormat("6(I3,E9.0)"))
                numj.extend  ([data[0],data[2],data[4],data[6],data[8],data[10]])
                widthj.extend([data[1],data[3],data[5],data[7],data[9],data[11]])
                idx = -1
                try:
                    idx = numj.index(0)
                except:
                    idx = -1
                if idx != -1:
                    numj   = numj[:idx]
                    widthj = widthj[:idx]
                    inJ = 0
                    inK = 1
            elif inK:
                data = FortranLine(line, FortranFormat("6(I3,E9.0)"))
                numk.extend  ([data[0],data[2],data[4],data[6],data[8],data[10]])
                widthk.extend([data[1],data[3],data[5],data[7],data[9],data[11]])
                idx = -1
                try:
                    idx = numk.index(0)
```

```
                except:
                    idx = -1
                if idx != -1:
                    numk   = numk[:idx]
                    widthk = widthk[:idx]
                    inK = 0
            if FortranLine(line, FortranFormat("A3"))[0] == "004":
                inSec004 = 1
                inI = 1
            if FortranLine(line, FortranFormat("A3"))[0] == "005":
                inSec004 = 0
        return numi, widthi, numj, widthj, numk, widthk

    def readSec005(filename, ni, nj, nk):
        f = open(filename, "r")
        inSec005 = 0
        meshdata = []
        lines = f.readlines()
        line = 0
        while 1:
            if inSec005:
                for i in range(ni/24+1):
                    data = FortranLine(lines[line], FortranFormat("24I3"))
                    meshdata.extend([data[index] for index in range(24)])
                    line += 1
                idx = -1
                try:
                    idx = meshdata.index(0)
                    meshdata = meshdata[:idx]
                    idx = -1
                except:
                    idx = -1
                line -= 1
            if FortranLine(lines[line], FortranFormat("A3"))[0] == "005":
                inSec005 = 1
            if len(meshdata) == ni*nj*nk:
                meshdata = reshape(array(meshdata), (nk, nj, ni)).tolist()
                return meshdata
                break
            line += 1


    class CoreMapCanvas(ScrolledCanvas):
        def __init__(self, parent=None, **kwargs):
            ScrolledCanvas.__init__(self, parent, **kwargs)

        def create_hexagon(self, x, y, pitch, **kwargs):
            a = pitch/2.0/sqrt(3.0)
            b = pitch/2.0/sqrt(3.0)*2.0
            ScrolledCanvas.create_polygon(self,
                                          x  , y-b,
                                          x+b, y-a,
                                          x+b, y+a,
                                          x  , y+b,
                                          x-b, y+a,
                                          x-b, y-a, **kwargs)

        def create_triangle(self, x, y, base, reverse=0, **kwargs):
            a = base/2.0
            b = base/2.0*sqrt(3.0)/2.0
            if reverse:
                b = -b
            ScrolledCanvas.create_polygon(self,
                                          x-a, y+b,
                                          x,   y-b,
                                          x+a, y+b, **kwargs)

    class CoreMap:
        def __init__(self, type, mapdata, label_text="CoreMap", pixelsize=10):
            self.root = Tk()
            self.mapdata = array(mapdata)
            self.canvasmargin = 0
            self.xmax = 300
            self.ymax = 300
            self.pixelsize = pixelsize
            self.cnvs = CoreMapCanvas(self.root,
                                      labelpos=S,
                                      label_text=label_text,
                                      canvasmargin=self.canvasmargin)

            if type == "XY":  self.prepXY()
            if type == "Hex": self.prepHex()
            if type == "Tri": self.prepTri()
            self.prepWin()

        def prepXY(self):
```

```
        xwidth = self.pixelsize
        ywidth = self.pixelsize
        for y in xrange(self.mapdata.shape[-2]):
            for x in xrange(self.mapdata.shape[-1]):
                self.cnvs.create_rectangle(x*xwidth,
                                           y*ywidth,
                                           (x+1)*xwidth,
                                           (y+1)*ywidth,
                                           fill=self.getColor(self.mapdata[y,x]))
        self.xmax = xwidth*self.mapdata.shape[-1]
        self.ymax = ywidth*self.mapdata.shape[-2]

    def prepHex(self):
        pitch = self.pixelsize
        for y in xrange(self.mapdata.shape[-2]):
            for x in xrange(self.mapdata.shape[-1]):
                self.cnvs.create_hexagon(pitch + x*pitch+y*pitch/2.0,
                                         pitch + y*pitch,
                                         pitch,
                                         fill=self.getColor(self.mapdata[y,x]),
                                         outline='black')

        self.xmax = pitch + self.mapdata.shape[-1]*pitch + self.mapdata.shape[-2]*pitch
        self.ymax = pitch + self.mapdata.shape[-2]*pitch

    def prepTri(self):
        base = self.pixelsize
        for y in xrange(self.mapdata.shape[-2]):
            for x in xrange(self.mapdata.shape[-1]):
                self.cnvs.create_triangle(base + x*base/2.0 + y*base/2.0,
                                          base + y*base*sqrt(3.0)/2.0,
                                          base,
                                          reverse=(x-1)%2,
                                          fill=self.getColor(self.mapdata[y,x]),
                                          outline='black')

        self.xmax = base + self.mapdata.shape[-1]*base/2.0 +
self.mapdata.shape[-2]*base/2.0
        self.ymax = base + self.mapdata.shape[-2]*base*sqrt(3.0)/2.0

    def prepWin(self):
        self.cnvs.pack(expand=YES, fill='both')
        self.cnvs.resizescrollregion()

    def draw(self):
        self.root.mainloop()

    def save(self, filename, **kwargs):
        self.cnvs.postscript(file=filename,
                             width=self.canvasmargin*2.0 + self.xmax,
                             height=self.canvasmargin+2.0 + self.ymax,
                             **kwargs)

    def getColor(self, index):
        colors = ['black', 'blue', 'yellow', 'red', 'green', 'brown',
                  'gray', 'darkblue', 'darkred', 'darkgreen',
                  'gold', 'skyblue', 'yellowgreen', 'pink', 'lightgreen',
                  'ivory', 'violet', 'lightcyan', 'lightpink', 'linen',
                  'violetred', 'aliceblue', 'snow']
        return colors[index%len(colors)]

class CITATIONCoreMap:
    def __init__(self, inputfilename):
        self.inputfilename = inputfilename
        self.mapdata = []
        self.geomtype = ""
        self.cm = None

        self.prepMapData()

    def draw(self, k=0, pixelsize=10):
        self.cm = CoreMap(self.geomtype, self.mapdata[k][:][:], pixelsize=pixelsize)
        self.cm.draw()

    def save(self, k=0, pixelsize=10, outputfilename="output.ps"):
        self.cm = CoreMap(self.geomtype, self.mapdata[k][:][:], pixelsize=pixelsize)
        self.cm.save(outputfilename)

    def prepMapData(self):
        geomtype = readSec003(self.inputfilename)
        if geomtype == "3D-XYZ": self.geomtype = "XY"
        if geomtype == "3D-HexZ": self.geomtype = "Hex"
        if geomtype == "3D-TriZ": self.geomtype = "Tri"
        ni, wi, nj, wj, nk, wk = readSec004(self.inputfilename)
        self.mapdata = readSec005(self.inputfilename, len(ni), len(nj), len(nk))
```

```
###
def citview(filename, k, pixelsize):

    print "filename =", filename
    print "k =", k
    print "pixelsize =", pixelsize

    citview = CITATIONCoreMap(filename)

    print "Now trying to draw on the display..."
    citview.draw(k, pixelsize=pixelsize)

    print "Now trying to convert into a Postscript file..."
    outputdir = "./output/"
    try:
        os.mkdir(outputdir)
    except OSError:
        pass
    citview.save(k, pixelsize=pixelsize, outputfilename=outputdir+"output.ps")


if __name__ == "__main__":
    ni, wi, nj, wj, nk, wk = readSec004("XYZ18G.dat")
    mesh = readSec005("XYZ18G.dat", len(ni), len(nj), len(nk))
    print len(ni)*len(nj)*len(nk)
    print array(mesh)
```

図 A-9　JFS3utils.py

```
#!/usr/local/bin/python

from xml.sax import make_parser
from xml.sax.handler import ContentHandler
from xml.sax._exceptions import SAXParseException
## from Numeric import *
from scipy import *

from Neutronics.Nuclide import *
import Neutronics.Mixin as Mixin

###
### DataBase
###
jfsids ={
    'H-1':1,        'H-2':2,        'He-3':203,     'He-4':204,
    'Li-6':306,     'Li-7':307,     'Be-9':4,       'B-10':105,
    'B-11':115,     'C-12':6,       'N-14':147,     'N-15':157,
    'O-16':8,       'F-19':9,       'Na-23':11,     'Mg-nat.':12,
    'Al-27':13,     'Si-nat.':14,   'P-31':15,      'S-nat.':16,
    'Cl-nat.':17,   'Ar-40':180,    'K-nat.':19,    'Ca-nat.':20,
    'Sc-45':215,    'Ti-nat.':22,   'V-51':23,      'Cr-nat.':24,
    'Mn-55':25,     'Fe-nat.':26,   'Co-59':279,    'Ni-nat.':28,
    'Cu-nat.':29,   'Ga-nat.':31,   'Ga-69':319,    'Ga-71':311,
    'Ge-nat.':32,   'Ge-70':320,    'Ge-72':322,    'Ge-73':323,
    'Ge-74':324,    'Ge-76':326,    'As-75':335,    'Se-74':344,
    'Se-76':346,    'Se-77':347,    'Se-78':348,    'Se-79':349,
    'Se-80':340,    'Se-82':342,    'Br-79':359,    'Br-81':351,
    'Kr-78':368,    'Kr-80':360,    'Kr-82':362,    'Kr-83':363,
    'Kr-84':364,    'Kr-85':365,    'Kr-86':366,    'Rb-85':375,
    'Rb-86':376,    'Rb-87':377,    'Sr-84':384,    'Sr-86':386,
    'Sr-87':387,    'Sr-88':388,    'Sr-89':389,    'Sr-90':380,
    'Y-89':399,     'Y-90':390,     'Y-91':391,     'Zr-nat.':40,
    'Zr-90':400,    'Zr-91':401,    'Zr-92':402,    'Zr-93':403,
    'Zr-94':404,    'Zr-95':405,    'Zr-96':406,    'Nb-93':413,
    'Nb-94':414,    'Nb-95':415,    'Mo-nat.':42,   'Mo-92':422,
    'Mo-94':424,    'Mo-95':425,    'Mo-96':426,    'Mo-97':427,
    'Mo-98':428,    'Mo-99':429,    'Mo-100':420,   'Tc-99':439,
    'Ru-96':446,    'Ru-98':448,    'Ru-99':449,    'Ru-100':440,
    'Ru-101':441,   'Ru-102':442,   'Ru-103':443,   'Ru-104':444,
    'Ru-105':447,   'Ru-106':445,   'Rh-103':453,   'Rh-105':455,
```

```
'Pd-102':462,   'Pd-104':464,   'Pd-105':465,   'Pd-106':466,
'Pd-107':467,   'Pd-108':468,   'Pd-110':460,   'Ag-nat.':47,
'Ag-107':477,   'Ag-109':479,   'Ag-110m':470,  'Cd-nat.':48,
'Cd-106':486,   'Cd-108':488,   'Cd-110':480,   'Cd-111':481,
'Cd-112':482,   'Cd-113':483,   'Cd-114':484,   'Cd-116':485,
'In-113':493,   'In-115':495,   'Sn-nat.':50,   'Sn-112':492,
'Sn-114':494,   'Sn-115':505,   'Sn-116':501,   'Sn-117':507,
'Sn-118':508,   'Sn-119':509,   'Sn-120':500,   'Sn-122':502,
'Sn-123':503,   'Sn-124':504,   'Sn-126':506,   'Sb-nat.':51,
'Sb-121':511,   'Sb-123':513,   'Sb-124':514,   'Sb-125':515,
'Sb-126':516,   'Te-120':520,   'Te-122':522,   'Te-123':523,
'Te-124':524,   'Te-125':525,   'Te-126':526,   'Te-127m':527,
'Te-128':528,   'Te-129m':529,  'Te-130':533,   'Te-132':532,
'I-127':537,    'I-129':539,    'I-130':530,    'I-131':531,
'I-135':535,    'Xe-124':534,   'Xe-126':536,   'Xe-128':548,
'Xe-129':549,   'Xe-130':540,   'Xe-131':541,   'Xe-132':542,
'Xe-133':543,   'Xe-134':544,   'Xe-135':545,   'Xe-136':546,
'Cs-133':553,   'Cs-134':554,   'Cs-135':555,   'Cs-136':556,
'Cs-137':557,   'Ba-130':560,   'Ba-132':562,   'Ba-134':564,
'Ba-135':565,   'Ba-136':566,   'Ba-137':567,   'Ba-138':568,
'Ba-140':569,   'La-138':578,   'La-139':579,   'La-140':570,
'Ce-140':580,   'Ce-141':581,   'Ce-142':582,   'Ce-143':583,
'Ce-144':584,   'Pr-141':591,   'Pr-143':593,   'Nd-142':602,
'Nd-143':603,   'Nd-144':604,   'Nd-145':605,   'Nd-146':606,
'Nd-147':607,   'Nd-148':608,   'Nd-150':600,   'Pm-147':617,
'Pm-148':618,   'Pm-148m':610,  'Pm-149':619,   'Pm-151':611,
'Sm-144':625,   'Sm-147':627,   'Sm-148':628,   'Sm-149':629,
'Sm-150':620,   'Sm-151':621,   'Sm-152':622,   'Sm-153':623,
'Sm-154':624,   'Eu-nat.':63,   'Eu-151':631,   'Eu-152':632,
'Eu-153':633,   'Eu-154':634,   'Eu-155':635,   'Eu-156':636,
'Eu-157':637,   'Gd-152':642,   'Gd-154':644,   'Gd-155':645,
'Gd-156':646,   'Gd-157':647,   'Gd-158':648,   'Gd-160':640,
'Tb-159':659,   'Tb-160':650,   'Dy-160':660,   'Dy-161':661,
'Ho-165':675,   'Hf-nat.':72,   'Ta-181':731,   'W-nat.':74,
'Au-197':797,   'Pb-nat.':82,   'Bi-209':839,   'Th-227':907,
'Th-228':908,   'Th-229':909,   'Th-230':900,   'Th-232':902,
'Th-233':903,   'Th-234':904,   'Pa-231':911,   'Pa-232':912,
'Pa-233':913,   'U-232':922,    'U-233':923,    'U-234':924,
'U-235':925,    'U-236':926,    'U-237':927,    'U-238':928,
'Np-236':936,   'Np-237':937,   'Np-238':938,   'Np-239':939,
'Pu-236':946,   'Pu-237':947,   'Pu-238':948,   'Pu-239':949,
'Pu-240':940,   'Pu-241':941,   'Pu-242':942,   'Am-241':951,
'Am-242':952,   'Am-242m':950,  'Am-243':953,   'Am-244':954,
'Am-244m':955,  'Cm-241':961,   'Cm-242':962,   'Cm-243':963,
'Cm-244':964,   'Cm-245':965,   'Cm-246':966,   'Cm-247':967,
'Cm-248':968,   'Cm-249':969,   'Cm-250':960,   'Pu-241FP':814,
'Pu-239FP':894, 'U-235FP':854,  'U-238FP':884,  'Pu-241FP':714,
'Pu-239FP':794, 'U-235FP':754,  'U-238FP':784,  'Pu-241FP':713,
'Pu-239FP':793, 'U-235FP':753,  'U-238FP':783}

jfs70g = [ 1.00000000e+07,  7.78800800e+06,  6.06530700e+06,  4.72366600e+06,
           3.67879400e+06,  2.86504800e+06,  2.23130200e+06,  1.73773900e+06,
           1.35335300e+06,  1.05399200e+06,  8.20850000e+05,  6.39278625e+05,
           4.97870688e+05,  3.87742094e+05,  3.01973812e+05,  2.35177500e+05,
           1.83156406e+05,  1.42642297e+05,  1.11090000e+05,  8.65169531e+04,
           6.73794688e+04,  5.24751797e+04,  4.08677109e+04,  3.18278105e+04,
           2.47875195e+04,  1.93045391e+04,  1.50343896e+04,  1.17087998e+04,
           9.11882031e+03,  7.10174414e+03,  5.53084424e+03,  4.30742480e+03,
           3.35462598e+03,  2.61258594e+03,  2.03468396e+03,  1.58461304e+03,
           1.23409802e+03,  9.61116516e+02,  7.48518311e+02,  5.82946594e+02,
           4.53999298e+02,  3.53575012e+02,  2.75364502e+02,  2.14454102e+02,
           1.67016998e+02,  1.30072998e+02,  1.01300903e+02,  7.88932495e+01,
           6.14421196e+01,  4.78511696e+01,  3.72665291e+01,  2.90231991e+01,
           2.26032906e+01,  1.76034603e+01,  1.37095900e+01,  1.06770401e+01,
           8.31528664e+00,  6.47595215e+00,  5.04347706e+00,  3.92786407e+00,
           3.05902290e+00,  2.38237000e+00,  1.85539103e+00,  1.44498003e+00,
           1.12535203e+00,  8.76424789e-01,  6.82560325e-01,  5.31578481e-01,
           4.13993806e-01,  3.22418690e-01,  9.99999975e-06]

jfs18g = [ 1.00000000e+07,  6.06530700e+06,  3.67879400e+06,  2.23130200e+06,
           1.35335300e+06,  8.20850000e+05,  3.87742094e+05,  1.83156406e+05,
           8.65169531e+04,  4.08677109e+04,  1.93045391e+04,  9.11882031e+03,
           4.30742480e+03,  2.03468396e+03,  9.61116516e+02,  4.53999298e+02,
           2.14454102e+02,  1.01300903e+02,  9.99999975e-06]

###
### XML Handler
###
class JFS3Handler(ContentHandler):
    def __init__(self, tagname, type):
        self.isFound = 0
        self.tagname = tagname
        self.type = type
        self.name = None
        self.attrs = None
        self.content = ""
```

```
    def startElement(self, name, attrs):
        if name == self.tagname:
            self.name = name
            self.attrs = attrs
            self.isFound = 1

    def characters(self, content):
        if self.isFound:
            ## self.content += content
            self.content = ''.join((self.content, content))

    def endElement(self, name):
        if name == self.tagname:
            self.isFound = 0

    def endDocument(self):
        self.content = map(eval(self.type), self.content.replace('¥n', ' ').split())

class JFS3Reader:
    def __init__(self, filename):
        self.f = open(filename, 'r')

    def getData(self, type, tagname):
        ch = JFS3Handler(tagname, type)
        saxparser = make_parser()
        saxparser.setContentHandler(ch)
        saxparser.parse(self.f)
        self.f.seek(0)
        return ch.content

###
### Classes
###
class GroupConstant(Mixin.SerializableAndVisible):
    def __init__(self, energy_group=1, energy_boundaries=[1E+6, 1E-5]):
        self.energy_group      = energy_group
        self.energy_boundaries = array(energy_boundaries)
        assert self.energy_boundaries.shape == (self.energy_group+1,)

class GroupMicroscopicCrossSection1D(GroupConstant):
    def __init__(self, energy_group=1, energy_boundaries=[1E+6, 1E-5]):
        GroupConstant.__init__(self, energy_group, energy_boundaries)
        self.fission_spectrum = zeros(self.energy_group, Float)
        self.fission          = zeros(self.energy_group, Float)
        self.nuvalue          = zeros(self.energy_group, Float)
        self.capture          = zeros(self.energy_group, Float)
        self.elastic          = zeros(self.energy_group, Float)
        self.inelastic        = zeros(self.energy_group, Float)
        self.n2n              = zeros(self.energy_group, Float)
        self.muvalue          = zeros(self.energy_group, Float)

    def visualize(self, logx=1, logy=1):
        x  = self.energy_boundaries
        y1 = self._prepGraphData(self.fission)
        y2 = self._prepGraphData(self.capture)
        y3 = self._prepGraphData(self.elastic)
        y4 = self._prepGraphData(self.inelastic)
        y5 = self._prepGraphData(self.n2n)
        gplt.plot(x, y1, "title 'fission'   with linespoints",
                  x, y2, "title 'capture'   with linespoints",
                  x, y3, "title 'elastic'   with linespoints",
                  x, y4, "title 'inelastic' with linespoints",
                  x, y5, "title 'n2n'       with linespoints")
        gplt.xtitle('Neutron Energy (eV)')
        gplt.ytitle('Cross Section (barn)')
        if logx: gplt.logx()
        if logy: gplt.logy()

    def _prepGraphData(self, numarray):
        ret = array(numarray.tolist() + [numarray[-1]])
        return ret

class GroupMicroscopicCrossSection2D(GroupConstant):
    def __init__(self, energy_group=1, energy_boundaries=[1E+6, 1E-5]):
        GroupConstant.__init__(self, energy_group, energy_boundaries)
        self.elastic   = zeros((self.energy_group, self.energy_group), Float)
        self.inelastic = zeros((self.energy_group, self.energy_group), Float)
        self.n2n       = zeros((self.energy_group, self.energy_group), Float)

    def visualize(self):
        y = self.inelastic
        gplt.surf(y)
        y = self.elastic
        gplt.surf(y)
        y = self.n2n
```

```
        gplt.surf(y)

class FTable(GroupConstant):
    def __init__(self, energy_group=1, energy_boundaries=[1E+6, 1E-5]):
        GroupConstant.__init__(self, energy_group, energy_boundaries)
        self.sig0s        = ones(1, Float)
        self.temperatures = ones(1, Float)
        self.rvalues      = ones(1, Float)
        shape_ftable = (self.sig0s.shape[0],
                        self.temperatures.shape[0],
                        self.rvalues.shape[0],
                        self.energy_group)
        self.fission         = ones(shape_ftable)
        self.capture         = ones(shape_ftable)
        self.elastic         = ones(shape_ftable)
        self.total           = ones(shape_ftable)
        self.elastic_removal = ones(shape_ftable)
        self.inelastic       = ones(shape_ftable)

    def _getffactor_with_rvalueindex(self, group, sig0, temp, rvalue=0):
        x0 = log10(self.sig0s)
        y0 = self.total[group-1, rvalue, 0, :]
        tck0 = interpolate.splrep(x0,y0,k=3,s=0)
        x1 = log10(self.sig0s)
        y1 = self.total[group-1, rvalue, 1, :]
        tck1 = interpolate.splrep(x1,y1,k=3,s=0)
        x2 = log10(self.sig0s)
        y2 = self.total[group-1, rvalue, 2, :]
        tck2 = interpolate.splrep(x2,y2,k=3,s=0)
        x3 = log10(self.sig0s)
        y3 = self.total[group-1, rvalue, 3, :]
        tck3 = interpolate.splrep(x3,y3,k=3,s=0)

        debug = 0
        if debug:
            xnew = arange(log10(self.sig0s[0]), log10(self.sig0s[-1]), 0.1)
            ynew = interpolate.splev(xnew, tck1)
            gplt.plot(x1,y1, xnew, ynew)

        out0 = interpolate.splev(log10(sig0), tck0)
        out1 = interpolate.splev(log10(sig0), tck1)
        out2 = interpolate.splev(log10(sig0), tck2)
        out3 = interpolate.splev(log10(sig0), tck3)
        x = log10(self.temperatures)
        y = [out0, out1, out2, out3]
        tck = interpolate.splrep(x,y,k=3,s=0)
        if temp < self.temperatures[0]:
            ret = (tck[1][0]
                   + tck[1][1]*(log10(temp)-x[0])
                   + tck[1][2]*(log10(temp)-x[0])**2)  ## From SLAROM's manual
        elif temp > self.temperatures[-1]:
            ret = (tck[1][0]
                   + tck[1][1]*(log10(temp)-x[-1])
                   + tck[1][2]*(log10(temp)-x[-1])**2)  ## From SLAROM's manual
        else:
            ret = interpolate.splev(log10(temp),tck)

        debug = 0
        if debug:
            xnew = arange(log10(self.temperatures[0]), log10(self.temperatures[-1]+1000),
0.1)
            ynew = interpolate.splev(xnew, tck)
            gplt.plot(x,y, xnew, ynew)

        return ret

    def getffactor(self, group, sig0, temp, rvalue=None):
        if rvalue == None:
            ret = self._getffactor_with_rvalueindex(group, sig0, temp, rvalue=0)
        else:
            out0 = self._getffactor_with_rvalueindex(group, sig0, temp, 0)
            out1 = self._getffactor_with_rvalueindex(group, sig0, temp, 1)
            out2 = self._getffactor_with_rvalueindex(group, sig0, temp, 2)
            out3 = self._getffactor_with_rvalueindex(group, sig0, temp, 3)
            x = self.rvalues
            y = [out0, out1, out2, out3]
            tck = interpolate.splrep(x,y,k=1,s=0)
            ret = interpolate.splev(rvalue,tck)

            debug = 0
            if debug:
                xnew = arange(x[0], x[-1], 10.0)
                ynew = interpolate.splev(xnew, tck)
                gplt.plot(x,y, xnew,ynew)
```

```
        return ret

    def visualize(self, energy_group):
        x1 = self.sig0s
        x2 = self.temperatures
        eg = energy_group
        titlename = "f-table"
        ## group, rvalue, temperature, sig0
        for rvalue in range(self.total.shape[1]):
            y1 = self.total[eg-1,rvalue,:,:]
            gplt.surf(x1,x2,y1)
            gplt.logx()
            gplt.logy()
            gplt.title(titlename)
            gplt.xtitle('sig0')
            gplt.ytitle('Temperature')
            gplt.ztitle('f-fator')

class GroupConstantSet(Mixin.SerializableAndVisible):
    def __init__(self, nuclide=Nuclide(),
                 sig1d=GroupMicroscopicCrossSection1D(),
                 sig2d=GroupMicroscopicCrossSection2D(),
                 ftable=FTable):
        self.nuclide = nuclide
        self.sig1d   = sig1d
        self.sig2d   = sig2d
        self.ftable  = ftable

##
## Functions
##
def getGroupStructure(filename):
    jfs = JFS3Reader(filename)
    energy_group      = jfs.getData('int', 'no_of_energy_group')[-1]
    energy_boundaries = array(jfs.getData('float', 'energy_boundary'))
    return energy_group, energy_boundaries

def readxml_GroupMicroscopicCrossSection1D(filename):
    jfs = JFS3Reader(filename)
    energy_group, energy_boundaries = getGroupStructure(filename)
    ret = GroupMicroscopicCrossSection1D(energy_group, energy_boundaries)
    ret.fission_spectrum = array(jfs.getData('float', 'fission_spectrum'))
    ret.fission          = array(jfs.getData('float', 'fission'))
    ret.nuvalue          = array(jfs.getData('float', 'nuvalue'))
    ret.capture          = array(jfs.getData('float', 'capture'))
    ret.elastic          = array(jfs.getData('float', 'elastic'))
    ret.inelastic        = array(jfs.getData('float', 'inelastic'))
    ret.n2n              = array(jfs.getData('float', 'n2n'))
    ret.muvalue          = array(jfs.getData('float', 'muvalue'))
    return ret

def readxml_GroupMicroscopicCrossSection2D(filename):
    jfs = JFS3Reader(filename)
    energy_group, energy_boundaries = getGroupStructure(filename)
    ret = GroupMicroscopicCrossSection2D(energy_group, energy_boundaries)
    isHydrogen = 0
    ncode1 = jfs.getData('int', 'nuclide_code')[-1]
    if ncode1 == 1: isHydrogen = 1

    if isHydrogen:
        shape_elastic   = (jfs.getData('int', 'max_LA_el')[-1],
                           energy_group)
    else:
        shape_elastic   = (jfs.getData('int', 'max_LA_el')[-1],
                           jfs.getData('int', 'max_LD_el')[-1])

    shape_inelastic = (jfs.getData('int', 'max_LA_in')[-1],
                       jfs.getData('int', 'max_LD_in')[-1])
    shape_n2n       = (jfs.getData('int', 'max_LA_n2n')[-1],
                       jfs.getData('int', 'max_LD_n2n')[-1])

    elastic   = reshape(array(jfs.getData('float', 'xsec_elastic')),   shape_elastic)
    inelastic = reshape(array(jfs.getData('float', 'xsec_inelastic')), shape_inelastic)
    n2n       = reshape(array(jfs.getData('float', 'xsec_n2n')),       shape_n2n)

    for j in xrange(shape_elastic[1]):
        for i in xrange(shape_elastic[0]):
            ret.elastic[i,j] = elastic[i,j]
    for j in xrange(shape_inelastic[1]):
        for i in xrange(shape_inelastic[0]):
            ret.inelastic[i,j] = inelastic[i,j]
    for j in xrange(shape_n2n[1]):
        for i in xrange(shape_n2n[0]):
            ret.n2n[i,j] = n2n[i,j]

    return ret
```

```
def readxml_FTable(filename):
    jfs = JFS3Reader(filename)
    energy_group, energy_boundaries = getGroupStructure(filename)
    ret = FTable(energy_group, energy_boundaries)

    max_sig0        = jfs.getData('int',   'no_of_sig0values')[-1]
    max_temperature = jfs.getData('int',   'max_of_temperatures')[-1]
    max_rvalue      = jfs.getData('int',   'max_of_rvalues')[-1]
    max_reaction    = jfs.getData('int',   'no_of_reaction_ftable')[-1]
    ntemp           = jfs.getData('int',   'highest_energy_group')
    nr              = jfs.getData('int',   'num_of_rvalues')
    code_sig0       = jfs.getData('int',   'code_sig0')

    ret.sig0s         = array(jfs.getData('float', 'ftable_specification'))
    ret.temperatures  = array(jfs.getData('float', 'temperature'))
    ret.rvalues       = reshape(array(jfs.getData('float', 'rvalue')),
                                (max_rvalue, max_sig0))
    ret.rvalues       = ret.rvalues[:,0]

    if max(ntemp) == energy_group + 1 and min(ntemp) == energy_group + 1:
        max_temperature = 1

    has_fission         = 0
    has_capture         = 0
    has_elastic         = 0
    has_total           = 0
    has_elastic_removal = 0
    has_inelastic       = 0

    if 1 in code_sig0: has_fission         = 1
    if 2 in code_sig0: has_capture         = 1
    if 3 in code_sig0: has_elastic         = 1
    if 4 in code_sig0: has_total           = 1
    if 5 in code_sig0: has_elastic_removal = 1
    if 6 in code_sig0: has_inelastic       = 1

    max_reaction = (has_fission
                    +has_capture
                    +has_elastic
                    +has_total
                    +has_elastic_removal
                    +has_inelastic)

    ftable_shape = (energy_group,
                    max_reaction,
                    max(nr),
                    max_temperature,
                    max_sig0)

    ## print ftable_shape
    ## print array(jfs.getData('float', 'code_temperature')).shape

    ftable = reshape(array(jfs.getData('float', 'code_temperature')), ftable_shape)

    ftable_shape = (energy_group,
                    max(nr),
                    max_temperature,
                    max_sig0)

    reac_index = 0

    if has_fission:
        ret.fission = ftable[:,reac_index,:,:,:]
        reac_index += 1
    else:
        ret.fission = ones(ftable_shape, Float)

    if has_capture:
        ret.capture = ftable[:,reac_index,:,:]
        reac_index += 1
    else:
        ret.capture = ones(ftable_shape, Float)

    if has_elastic:
        ret.elastic = ftable[:,reac_index,:,:,:]
        reac_index += 1
    else:
        ret.capture = ones(ftable_shape, Float)

    if has_total:
        ret.total   = ftable[:,reac_index,:,:,:]
        reac_index += 1
    else:
        ret.total   = ones(ftable_shape, Float)
```

```
    if has_elastic_removal:
        ret.elastic_removal = ftable[:,reac_index,:,:,:]
        reac_index += 1
    else:
        ret.elastic_removal = ones(ftable_shape, Float)

    if has_inelastic:
        ret.inelastic = ftable[:,reac_index,:,:,:]
        reac_index += 1
    else:
        ret.inelastic = ones(ftable_shape, Float)

    return ret

##
## alias
##
readxml_GMCS1D = readxml_GroupMicroscopicCrossSection1D
readxml_GMCS2D = readxml_GroupMicroscopicCrossSection2D
readxml_FT_    = readxml_FTable

###
### Utilities
###
def idOfJFS(nuclide):
    assert isinstance(nuclide, Nuclide)
    return jfsids[nuclide.nameOfNuclide()]

def makePickle(nuclname, xmlfilename, pklfilename):
    print "Reading XML file of JFS-3", nuclname, "to convert into Pickle file ..."
    sig1d  = readxml_GMCS1D(xmlfilename)
    sig2d  = readxml_GMCS2D(xmlfilename)
    ftable = readxml_FTable(xmlfilename)
    gcset = GroupConstantSet(Nuclide(nuclname), sig1d, sig2d, ftable)
    gcset.dump(pklfilename)

def makeAllPickles(xmldir="./XML/", pkldir="./PKL/"):
    nucls = jfsids.keys()
    nucls.sort()
    for nucl in nucls:
        xmlfilename = xmldir + nucl + ".xml"
        pklfilename = pkldir + nucl + ".pkl"
        try:
            makePickle(nucl, xmlfilename, pklfilename)
        except SAXParseException:
            print "Warning: Cannot make Pickle file of " + nucl + "."
        except ValueError:
            print "Error: Cannot make Pickle file of " + nucl + "."
            print "       If Processing Th-232 of JFS3J32R, this is a known bug."
            print "       Pass to next."

def makeXML(nuclname, jfsfilename, xmlfilename):
    """Read JFS-3 binary file to convert into XML file.
    """
    JFS2XML = "./jfs2xml.exe"
    print "Reading JFS-3 binary file", nuclname, " to convert into XML file ..."
    os.popen("ln -sf " + jfsfilename + " fort.1")
    os.popen("echo " + str(idOfJFS(Nuclide(nuclname)))
             + " | " + JFS2XML
             + " > " + xmlfilename)
    os.popen("rm fort.1")

def makeAllXMLs(jfsfilename=None, xmldir="./XML/"):
    """Read all the date for nuclides of JFS-3 binary file to convert into XML files
    """
    nucls = jfsids.keys()
    nucls.sort()
    for nucl in nucls:
        xmlfilename = xmldir + nucl + ".xml"
        makeXML(nucl, jfsfilename, xmlfilename)

if __name__ == '__main__':
    jfsname = "JFS3J33.ORG"
    ## jfsname = "JFS3J32R.Y0202.N295"

    makeAllXMLs("~/Work_CVS/JFS3LIB/"+jfsname,
                "../../../NeutronicsData/XML_"+jfsname+"/")
    makeAllPickles("../../../NeutronicsData/XML_"+jfsname+"/",
                   "../../../NeutronicsData/PKL_"+jfsname+"/")
```

図A-10 jfs2xml.f

```
C
C    JFS-3 LIBRARY PRINT OUT UTILITY
C
cyoko      PARAMETER ( ILNMAX = 300 )
      PARAMETER ( ILNMAX = 383 )
      DIMENSION NCODEL(ILNMAX),ENBND(71),CHI(70,5),NCODX(5),
     >          AW(ILNMAX),LD(3,ILNMAX),LA(3,ILNMAX),
     >          MSF(6,ILNMAX),TAB(8,ILNMAX)
      DIMENSION FT(4,ILNMAX),RPARAM(8,4,ILNMAX),
     >          NTEMP(6,ILNMAX),NR(6,ILNMAX)
      DIMENSION SIG1D(ILNMAX,8)
C     DIMENSION STRE(6,ILNMAX)
      DIMENSION STRE(70,ILNMAX)
      DIMENSION STRI(70,ILNMAX)
      DIMENSION STR2(70,ILNMAX)
      DIMENSION FTABB(8,4,4)
C
      CHARACTER*40 DSN
      LOGICAL*4 TR(2)
C
      CHARACTER CMXR1D(8)*3,CMREAC(6)*2,CLDLA(3)*6
      DATA CMXR1D / '   F','   N','   C',' IN',' EL','   U',' ER','N2N' /
      DATA CMREAC / ' F',' C','EL',' T',' ER','IN' /
      DATA CLDLA  / ' ELAS ',' INEL ','(N,2N)' /
C     DSN   = 40H
      DSN   = '
      TR(1) = .FALSE.
      TR(2) = .FALSE.
      INQUIRE(1,NAME=DSN,NAMED=TR(1),OPENED=TR(2) )
C========
    1 CONTINUE
      READ(5,*,END=9999) NUMO
C========
      REWIND 1
C
C   #1 RECORD
C
      READ(1) LNMAX,IMAX,MXCHI,MXR1D,MTXR23,MXDNSE,MXDNSI,MXDNS2,
     >        MXDWNE,MXDWNI,MXDWN2,MXREAC,MXSIGO,MXTEMP,MXR,ISWH
      IF( LNMAX .GT. ILNMAX ) WRITE(6,603) LNMAX
      IF( LNMAX .GT. ILNMAX ) GO TO 9999
  603 FORMAT(1H1,' LNMAX >> ILNMAX ' /
     >       5X,'PLEASE RESET PARAMETER:ILNMAX <---',I3 )
c     WRITE( 6,6000) DSN,
c    >       LNMAX,IMAX,MXCHI,MXR1D,MTXR23,MXDNSE,MXDNSI,MXDNS2,
c    >       MXDWNE,MXDWNI,MXDWN2,MXREAC,MXSIGO,MXTEMP,MXR,ISWH
ccc
ccc Transforming to XML
ccc
      write(6,'(33a)') '<?xml version="1.0" encoding="iso-8859-1"?>'
      write(6,*) '<jfs3>'
c      write(6,*) '<record number=1>'
      write(6,*) '<record1>'
      write(6,*) '  <no_of_nuclide varname="lnmax"> '
      write(6,*)      lnmax
      write(6,*) '  </no_of_nuclide>'
      write(6,*) '  <no_of_energy_group varname="imax">'
      write(6,*)      imax
      write(6,*) '  </no_of_energy_group>'
      write(6,*) '  <no_of_fission_spectrum varname="mxchi">'
      write(6,*)      mxchi
      write(6,*) '  </no_of_fission_spectrum>'
      write(6,*) '  <no_of_reaction_1d_data varname="mxr1d">'
      write(6,*)      mxr1d
      write(6,*) '  </no_of_reaction_1d_data>'
      write(6,*) '  <matrix_specification varname="mtxr23">'
      write(6,*)      mtxr23
      write(6,*) '  </matrix_specification>'
      write(6,*) '  <max_LD_el varname="mxdnse">'
      write(6,*)      mxdnse
      write(6,*) '  </max_LD_el>'
      write(6,*) '  <max_LD_in varname="mxdnsi">'
      write(6,*)      mxdnsi
      write(6,*) '  </max_LD_in>'
      write(6,*) '  <max_LD_n2n varname="mxdns2">'
      write(6,*)      mxdns2
      write(6,*) '  </max_LD_n2n>'
      write(6,*) '  <max_LA_el varname="mxdwne">'
      write(6,*)      mxdwne
      write(6,*) '  </max_LA_el>'
      write(6,*) '  <max_LA_in varname="mxdwni">'
      write(6,*)      mxdwni
      write(6,*) '  </max_LA_in>'
      write(6,*) '  <max_LA_n2n varname="mxdwn2">'
```

```
      write(6,*)      mxdwn2
      write(6,*) '  </max_LA_n2n>'
      write(6,*) '  <no_of_reaction_ftable varname="mxreac">'
      write(6,*)      mxreac
      write(6,*) '  </no_of_reaction_ftable>'
      write(6,*) '  <no_of_sig0values varname="mxsigo">'
      write(6,*)      mxsigo
      write(6,*) '  </no_of_sig0values>'
      write(6,*) '  <max_of_temperatures varname="mxtemp">'
      write(6,*)      mxtemp
      write(6,*) '  </max_of_temperatures>'
      write(6,*) '  <max_of_rvalues varname="mxr">'
      write(6,*)      mxr
      write(6,*) '  </max_of_rvalues>'
      write(6,*) '  <position_of_hydrogen varname="iswh">'
      write(6,*)      iswh
      write(6,*) '  </position_of_hydrogen>'
c      write(6,*) '</record>'
      write(6,*) '</record1>'
C
C   #2 RECORD
C
      READ(1) (NCODEL(I),I=1,LNMAX),    (ENBND(I),I=1,IMAX+1),
     >       ((CHI(I,J),I=1,IMAX),J=1,MXCHI),
     >        (NCODX(J),J=1,MXCHI),     (AW(I),I=1,LNMAX),
     >       ((LD(I,J),I=1,3),J=1,LNMAX),
     >       ((LA(I,J),I=1,3),J=1,LNMAX),
     >       ((MSF(I,J),I=1,MXREAC),J=1,LNMAX),
     >       ((TAB(I,J),I=1,MXSIGO),J=1,LNMAX)
C ========
      CALL AA(LNMAX,NCODEL,NUMO,NUM)
        IF( NUM  .EQ. 0 )  WRITE(6,601) NUMO
        IF( NUM  .EQ. 0 )  GO TO 9000
      CALL AA(MXCHI,NCODX,NUMO,NUM1)
C       IF( NUM1 .EQ. 0 )  WRITE(6,602) NUMO
  601 FORMAT( 1H1,4X, '*********************************************'
     >       /   5X,'**  CODE NO.',I4,' NOTHING  IN THIS LIBRARY  **'
     >       /   5X, '*********************************************' )
  602 FORMAT( 5X, 'CODE NO.',I4,' HAVE NOT FISSION SPECTRUM DATA'  )
C ========
c     WRITE( 6,6010) NCODEL(NUM)
c     WRITE( 6,6020) (ENBND(I),I=1,IMAX+1)
C     WRITE(16,6020) (ENBND(I),I=1,IMAX+1)
c     WRITE( 6,6030)
c     IF( NUM1 .NE. 0 )
c    >WRITE( 6,6040) NCODX(NUM1),(CHI(I,NUM1),I=1,IMAX)
c     WRITE( 6,6050) (CLDLA(JJ),JJ=1,3),(CLDLA(JJ),JJ=1,3)
c     WRITE( 6,6060)
c    >          NCODEL(NUM),AW(NUM),(LD(J,NUM),J=1,3),(LA(J,NUM),J=1,3)
c     WRITE( 6,6070) (CMREAC(JJ),JJ=1,6),(MSF(J,NUM),J=1,MXREAC)
c     WRITE( 6,6080) (JJ,JJ=1,8),(TAB(J,NUM),J=1,MXSIGO)
ccc
ccc Transforming to XML
ccc
c      write(6,*) '<record number=2>'
      write(6,*) '<record2>'
      write(6,*) '  <nuclide_code varname="ncodel">'
      write(6,*)      ncodel(num)
      write(6,*) '  </nuclide_code>'
      write(6,*) '  <energy_boundary varname="enbnd">'
      write(6,*)      (enbnd(i),i=1,imax+1)
      write(6,*) '  </energy_boundary>'
      if( num1 .ne. 0 ) then
        write(6,*) '  <code_given_fission_spectrum varname="ncodx">'
        write(6,*)      ncodx(num1)
        write(6,*) '  </code_given_fission_spectrum>'
        write(6,*) '  <fission_spectrum varname="chi">'
        write(6,*)      (chi(i,num1),i=1,imax)
        write(6,*) '  </fission_spectrum>'
      endif
      write(6,*) '  <atomic_weight varname="aw">'
      write(6,*)      aw(num)
      write(6,*) '  </atomic_weight>'
      write(6,*) '  <max_sink_group varname="ld">'
      write(6,*)      (ld(j,num),j=1,3)
      write(6,*) '  </max_sink_group>'
      write(6,*) '  <lowest_energy_group varname="la">'
      write(6,*)      (la(j,num),j=1,3)
      write(6,*) '  </lowest_energy_group>'
      write(6,*) '  <ftable_specification varname="tab">'
      write(6,*)      (tab(j,num),j=1,mxsigo)
      write(6,*) '  </ftable_specification>'
      write(6,*) '</record2>'
C
C   #3 RECORD
C
```

```
      READ(1)  ((FT(I,J),I=1,MXTEMP),J=1,LNMAX),
     >         (((RPARAM(I,J,K),I=1,MXSIGO),J=1,MXR),K=1,LNMAX),
     >         ((NTEMP(I,J),I=1,MXREAC),J=1,LNMAX),
     >         ((NR(I,J),I=1,MXREAC),J=1,LNMAX)
c     WRITE( 6,61)  (JJ,JJ=1,MXTEMP),(FT(I,NUM),I=1,MXTEMP)
c     WRITE( 6,62)  (JJ,JJ=1,MXSIGO),
c    >              (J,(RPARAM(I,J,NUM),I=1,MXSIGO),J=1,MXR)
c     WRITE( 6,63)  (CMREAC(JJ),JJ=1,6),(NTEMP(I,NUM),I=1,MXREAC)
c     WRITE( 6,64)  (CMREAC(JJ),JJ=1,6),(NR(I,NUM),I=1,MXREAC)
ccc
ccc Transforming to XML
ccc
c      write(6,*) '<record number=3>'
      write(6,*) '<record3>'
      write(6,*) '  <temperature varname="ft">'
      write(6,*)       (ft(i,num),i=1,mxtemp)
      write(6,*) '  </temperature>'
      write(6,*) '  <rvalue varname="rparam">'
      write(6,*)       ((rparam(i,j,num),i=1,mxsigo),j=1,mxr)
      write(6,*) '  </rvalue>'
      write(6,*) '  <highest_energy_group varname="ntemp">'
      write(6,*)       (ntemp(i,num),i=1,mxreac)
      write(6,*) '  </highest_energy_group>'
      write(6,*) '  <num_of_rvalues varname="nr">'
      write(6,*)       (nr(i,num),i=1,mxreac)
      write(6,*) '  </num_of_rvalues>'
      write(6,*) '</record3>'
ccc
ccc
ccc
C
      DO 820 I=1,IMAX
c       WRITE( 6,165) I

        write(6,*) '<energy_group number="',i,'">'
C
C   #4 RECORD
C
      READ(1)      ((SIG1D(M,J),M=1,LNMAX),J=1,MXR1D)
c     WRITE( 6,65)  (CMXR1D(JJ),JJ=1,8),(SIG1D(NUM,J),J=1,MXR1D)
ccc
ccc Transforming to XML
ccc
c      write(6,*) '<record number=4>'
      write(6,*) '<record4>'
      write(6,*) '  <xsec_1d varname="sig1d">'
      write(6,*)       (sig1d(num,j),j=1,mxr1d)
      write(6,*) '    <fission>'
      write(6,*)       sig1d(num,1)
      write(6,*) '    </fission>'
      write(6,*) '    <nuvalue>'
      write(6,*)       sig1d(num,2)
      write(6,*) '    </nuvalue>'
      write(6,*) '    <capture>'
      write(6,*)       sig1d(num,3)
      write(6,*) '    </capture>'
      write(6,*) '    <inelastic>'
      write(6,*)       sig1d(num,4)
      write(6,*) '    </inelastic>'
      write(6,*) '    <elastic>'
      write(6,*)       sig1d(num,5)
      write(6,*) '    </elastic>'
      write(6,*) '    <muvalue>'
      write(6,*)       sig1d(num,6)
      write(6,*) '    </muvalue>'
      write(6,*) '    <elastic_removal>'
      write(6,*)       sig1d(num,7)
      write(6,*) '    </elastic_removal>'
      write(6,*) '    <n2n>'
      write(6,*)       sig1d(num,8)
      write(6,*) '    </n2n>'
      write(6,*) '  </xsec_1d>'
      write(6,*) '</record4>'
C
C   #5 RECORD
C
ccc
ccc Transforming to XML
ccc
c      write(6,*) '<record number=5>'
      write(6,*) '<record5>'
ccc
ccc
ccc
      IF(I.LE.MXDWNE) THEN
        READ(1)            ((STRE(J,M  ),J=1,MXDNSE),M=1,LNMAX)
```

```
c       IF(NUM0.NE.1) WRITE( 6,66) MXDNSE, (STRE(J,NUM),J=1,MXDNSE)
ccc
ccc Transforming to XML
ccc
        if(num0.ne.1) then
          write(6,*) '<xsec_elastic varname="stre">'
          write(6,*) (stre(j,num),j=1,mxdnse)
          write(6,*) '</xsec_elastic>'
        endif
ccc
ccc
ccc
      ENDIF
      IF(ISWH.NE.0) THEN
        READ(1)        (STRE(1,J),J=1,IMAX)
c       IF(NUM0.EQ.1 ) WRITE( 6,166) (STRE(1,J),J=1,IMAX)
ccc
ccc Transforming to XML
ccc
        if(num0.eq.1) then
          write(6,*) '<xsec_elastic varname="stre">'
          write(6,*) (stre(1,j),j=1,imax)
          write(6,*) '</xsec_elastic>'
        endif
ccc
ccc
ccc
      ENDIF
      write(6,*) '</record5>'
C
C   #6 RECORD
C
      IF(I.LE.MXDWNI) THEN
        READ(1)             ((STRI(J,M   ),J=1,MXDNSI),M=1,LNMAX)
c       WRITE( 6,67) MXDNSI, (STRI(J,NUM),J=1,MXDNSI)
ccc
ccc Transforming to XML
ccc
        write(6,*) '<record6>'
          write(6,*) '<xsec_inelastic varname="stri">'
          write(6,*) (stri(j,num),j=1,mxdnsi)
          write(6,*) '</xsec_inelastic>'
        write(6,*) '</record6>'
ccc
ccc
ccc
        ENDIF
C
C   #7 RECORD
C
      IF(I.LE.MXDWN2) THEN
        READ(1)             ((STR2(J,M   ),J=1,MXDNS2),M=1,LNMAX)
c       WRITE( 6,68) MXDNS2, (STR2(J,NUM),J=1,MXDNS2)
ccc
ccc Transforming to XML
ccc
c        write(6,*) '<record number=7>'
        write(6,*) '<record7>'
          write(6,*) '<xsec_n2n varname="str2">'
          write(6,*) (str2(j,num),j=1,mxdns2)
          write(6,*) '</xsec_n2n>'
        write(6,*) '</record7>'
ccc
ccc
ccc
      ENDIF
C
c     WRITE( 6,169) (JJ,JJ=1,MXSIGO)

c      write(6,*) '<record number=8>'
      write(6,*) '<record8>'

      DO 810 M=1,LNMAX
      DO 800 MT=1,MXREAC
      MR=NR(MT,M)
      KT=MSF(MT,M)
      IF(MR.EQ.0.OR.KT.EQ.0) GOTO 800
C
C   #8 RECORD
C
      IF( (MT.NE.6) .OR.   (MT.EQ.6.AND.LA(2,M).GE.1) ) THEN
        READ(1) (((FTABB(J,K,N),J=1,MXSIGO),K=1,KT),N=1,MR)
        IF( M .EQ. NUM ) THEN
c           WRITE( 6,269) MT
          DO 80 N=1,MR
c           WRITE( 6,69) N, (K, (FTABB(J,K,N),J=1,MXSIGO),K=1,KT)
```

```
ccc
ccc Transforming to XML
ccc
              write(6,*) '  <code_sig0 varname="mt">'
              write(6,*)       mt
              write(6,*) '  </code_sig0>'
              write(6,*) '  <code_rvalue varname="mr">'
              write(6,*)       n
              write(6,*) '  </code_rvalue>'
              write(6,*) '    <ftable varname="ftabb">'
              do k=1, kt
                write(6,*)
     &          '    <code_temperature varname="kt" number="',k,'">'
                write(6,*)         (ftabb(j,k,n),j=1,mxsigo)
                write(6,*) '    </code_temperature>'
              enddo
              write(6,*) '    </ftable>'
ccc
ccc
   80     CONTINUE
        ENDIF
      ENDIF
  800 CONTINUE
  810 CONTINUE

      write(6,*) '</record8>'
      write(6,*) '</energy_group>'

  820 CONTINUE
      write(6,*) '</jfs3>'
C
 9000 CONTINUE
      GO TO 1
 9999 STOP
 6000 FORMAT(1H1/' *** JFS-3 LIBRARY ***  ( DATASET : ',A40,')'//
     > ' #1 '/
     > ' NO. OF NUCLIDE :',I5/
     > ' NO. OF ENERGY GROUP :',I5/
     > ' NO. OF FISSION SPECTRUM SET :',I5/
     > ' NO. OF REACTION IN 1-D DATA :',I5/
     > ' INELASTIC MATRIX DATA SPECIFICATION :',I5/
     > ' MAX LD :',3I5/
     > ' MAX LA :',3I5/
     > ' NO. OF REACTION IN F-TABLE :',I5/
     > ' NO. OF SIGO-VALUES :',I5/
     > ' MAXIMUM NO. OF TEMPERATURES :',I5/
     > ' MAXIMUM NO. OF R-VALUES :',I5/
     > ' POSITION OF HYDROGEN :',I5)
 6010 FORMAT(/' #2 NUCLIDE CODE NUMBER'/(8I10))
 6020 FORMAT(/' ENERGY BOUNDARY (EV)'/(1P8E10.3))
 6030 FORMAT(/' FISSION SPECTRUM')
 6040 FORMAT(/' NUCLIDE CODE NUMBER (',I4,' )'/
     >       (1P8E10.3))
 6050 FORMAT(/ 25X,' MAXIMUM SINK GROUP NUMBER '
     >         ,3X,'  LOWEST  ENERGY  GROUP    ' /
     > ' ATOM          WEIGHT      ', 4X,3A6,12X,3A6 )
 6060 FORMAT(I5,F15.4,' (AMU)   '    ,    3I6,12X,3I6 )
 6070 FORMAT(/' F-TABLE SPECIFICATION ' /  8X,6(3X,A2) /
     >       5X,'MSF',6I5 )
 6080 FORMAT(/' SIGMA0-VALUES ' / 3X,'MXSIGO =',I6,7I10 /
     >       5X,'TAB',8F10.1)
   61 FORMAT(/' #3 TEMPERATURE ' / 3X,'MXTEMP =',3X,4(I5,5X) /
     >       5X,'FT ',3X,1P4E10.3 )
   62 FORMAT(/' R-VALUE ' / ' RPARAM',3X,'MXSIGO =',2X,8(I8,2X)/
     > (10X,'MXR =',I2,3X,1P8E10.3)          /
     >        (15X,I2,3X,1P8E10.3)           )
   63 FORMAT(/' HIGHEST ENERGY GROUP THAT TEMPERATURE-DEPENDENCE EXISTS,
     >OTHERWISE  NTEMP=IMAX+1 ' / 10X,6(3X,A2) /
     >       5X,'NTEMP',( 6I5 ) )
   64 FORMAT(/' NUMBER OF R-VALUES '/ 10X,6(3X,A2) /
     >       5X,'NR',3X,( 6I5 ) )
  165 FORMAT(//' * ', I2,' ENERGY GROUP --------------' )
   65 FORMAT(/' #4 1-D DATA ' / 10X,8(7X,A3) /
     >   5X,'SIG1D',2X,1P8E10.3,( 12X,1P8E10.3 ) )
   66 FORMAT(/' #5 ELASTIC MATRIX DATA '/ 3X,'MXDNSE =',I3 /
     >   5X,'STRE',3X,1P10E10.3,/( 12X,1P10E10.3 )  )
  166 FORMAT(/' #5 ELASTIC MATRIX H-DATA  70 ENERGY GROUP '
     >       /( 1P10E10.3 ))
   67 FORMAT(/' #6 INELASTIC MATRIX DATA '/ 3X,'MXDNSI =',I3 /
     >   5X,'STRI',3X,1P10E10.3,/( 12X,1P10E10.3 )  )
   68 FORMAT(/' #7 (N,2N) MATRIX DATA '/ 3X,'MXDNS2 =',I3 /
     >   5X,'STR2',3X,1P10E10.3,/( 12X,1P10E10.3 )  )
  169 FORMAT(/' #8 F-TABLE ' / '    FTAB' ,  6X, 8HMXSIGO =, 8(I5,5X) )
  269 FORMAT( 5X,'MT =',I2 )
   69 FORMAT( 3X,'MR=',I2,2X,'KT=',I2,3X,1P8E10.3 /
     >      (13X,                  I2,3X,1P8E10.3 ) )
```

```
      END
CC
CC
      SUBROUTINE AA(I,NDIM,NIN,NOUT)
      DIMENSION  NDIM(I)
C
      DO 10 J=1,I
        IF( NIN .EQ. NDIM(J) )  THEN
          NOUT = J
          GO TO  99
        ENDIF
   10 CONTINUE
      NOUT = 0
C
   99 RETURN
      END
```


図 A-11 SAGEPutils.py

```
#!/usr/local/bin/python

import re
import cPickle
from Numeric import *
from Scientific.IO.FortranFormat import FortranFormat, FortranLine

import Neutronics.Mixin as Mixin
import Neutronics.Nuclide as Nuclide

###
### Database
###
reacids = {"fission_spectrum": 7,
           "fission"         : 2,
           "nuvalue"         : 3,
           "capture"         : 1,
           "elastic"         : 5,
           "inelastic"       : 6,
           "n2n"             : 9,
           "muvalue"         : 8}

###
### Classes
###
class Sensitivity(Mixin.SerializableAndVisible):
    def __init__(self, name1='', name2='', nuclid=0, reacid=0, ng=18, longname=""):
        self.name1 = name1
        self.name2 = name2
        self.nuclid = nuclid
        self.reacid = reacid
        self.ng = ng
        self.sens = [0.0 for i in range(ng)]
        self.longname = longname

    def __cmp__(self, other):
        if type(self) == type(other):
            return max(self.sens).__cmp__(max(other.sens))
        else:
            raise TypeError
```

```
class SensitivitySet(Mixin.SerializableAndVisible):
    def __init__(self, name1="", name2="", nuclid=0,
                 fission_spectrum = None,    # Not supported yet
                 fission   = Sensitivity(reacid=reacids["fission"]),
                 nuvalue   = Sensitivity(reacid=reacids["nuvalue"]),
                 capture   = Sensitivity(reacid=reacids["capture"]),
                 elastic   = Sensitivity(reacid=reacids["elastic"]),
                 inelastic = Sensitivity(reacid=reacids["inelastic"]),
                 n2n       = Sensitivity(reacid=reacids["n2n"]),
                 muvalue   = Sensitivity(reacid=reacids["muvalue"])):

        self.name1 = name1
        self.name2 = name2
        self.nuclid = nuclid
        self.fission_spectrum = fission_spectrum
        self.fission          = fission
        self.nuvalue          = nuvalue
        self.capture          = capture
        self.elastic          = elastic
        self.inelastic        = inelastic
        self.n2n              = n2n
        self.muvalue          = muvalue

class SensitivityList(Mixin.SerializableAndVisible):
    def __init__(self):
        self.sens_list = []

    def readAll(self, filename, ng=18):
        f = open (filename, "r")
        while 1:
            line = f.readline()
            if line == "":
                break
            n1, n2 = FortranLine(line, FortranFormat("(A10,A30)"))[0:2]
            n1 = n1.strip()
            n2 = n2.strip()
            line = f.readline()
            nid, rid = FortranLine(line, FortranFormat("(2I6)"))[0:2]
            sens = []
            for i in range(int((ng+1)/6)):
                line = f.readline()
                sens.extend(FortranLine(line, FortranFormat("(6E12.0)"))[0:6])
            sens = sens[0:ng]
            ret = Sensitivity()
            ret.name1 = n1
            ret.name2 = n2
            ret.nuclid = nid
            ret.reacid = rid
            ret.sens = sens
            self.sens_list.append(ret)
        f.close()

    def find(self, name1, name2, nuclid, reacid):
        for sens in self.sens_list:
            if (sens.name1 == name1 and sens.name2 == name2 and
                sens.nuclid == nuclid and sens.reacid == reacid):
                break
        return sens

    def makeSensitivitySet(self, name1, name2, nuclid):
        senssset = SensitivitySet()
        senssset.name1     = name1
        senssset.name2     = name2
        senssset.nuclid    = nuclid
        senssset.fission   = self.find(name1, name2, nuclid, reacids["fission"])
        senssset.nuvalue   = self.find(name1, name2, nuclid, reacids["nuvalue"])
        senssset.capture   = self.find(name1, name2, nuclid, reacids["capture"])
        senssset.elastic   = self.find(name1, name2, nuclid, reacids["elastic"])
        senssset.inelastic = self.find(name1, name2, nuclid, reacids["inelastic"])
        senssset.n2n       = self.find(name1, name2, nuclid, reacids["n2n"])
        senssset.muvalue   = self.find(name1, name2, nuclid, reacids["muvalue"])
        return senssset

class SensitivitySetList(Mixin.SerializableAndVisible):
    def __init__(self):
        self.senssset_list = []

    def makeFromSensitivityList(self, sens_list):
        assert isinstance(sens_list, SensitivityList)

        setWithoutDuplication = {}
        for sens in sens_list.sens_list:
            setWithoutDuplication[(sens.name1, sens.name2, sens.nuclid)] = None

        for key in setWithoutDuplication.keys():
            print key
```

```
            self.sensset_list.append(sens_list.makeSensitivitySet(key[0], key[1],
key[2]))

    def find(self, name1, name2, nuclid):
        for sensset in self.sensset_list:
            if (sensset.name1 == name1 and sensset.name2 == name2 and
                sensset.nuclid == nuclid):
                break
        return sensset

##     def dumpSensitivitySetList(self, filename, binary=1):
##         f = open(filename, "w")
##         cPickle.dump(self.sensset_list, f, binary)

def makePickle(sagepfilename, pklfilename):
    senslist = SensitivityList()
    senslist.readAll(sagepfilename)
    senssetlist = SensitivitySetList()
    senssetlist.makeFromSensitivityList(senslist)
    ## senssetlist.dumpSensitivitySetList(pklfilename)
    senssetlist.dump(pklfilename)

if __name__ == "__main__":
    filenames = ["JOYOMK1.SNS",
                 "JOYOMK1BURN.SNS",
                 "ZPPR09.SNS"]

    for filename in filenames:
        print filename
        makePickle("../../../NeutronicsData/SNS_SAGEP/" + filename ,
                   "../../../NeutronicsData/PKL_SNS/"   + filename + ".pkl")
```

図 A-12 LISTutils.py

```
#!/usr/local/bin/python

from Numeric import *

def expandList(oldList, numList):
    assert len(oldList) == len(numList)
    ret = []
    for index in range(len(oldList)):
        ret.extend([oldList[index] for index2 in range(numList[index])])
    return ret

def addUnique(baseList, otherList):
    for item in otherList:
        if item not in baseList:
            baseList.append(item)
    baseList.sort()
    return baseList

def devideLength(baseList, numList):
    pass

def getWidth(baseList, index):
    return baseList[index] - baseList[index-1]
```