原研における燃焼プラズマ研究のための
輸送コード（TOPICS）のモジュール化

原研那珂研　清 水　勝 宏

統合コード研究会
九州大学応用力学研究所
平成１６年3月１８日ー１９日

内　　容

- TASKとTOPICSとの関係
- プロジェクトとしての開発／維持管理の難しさ
- ライブラリーの考え
- 物理コードの導入を如何に簡単にするか
- コード間の結合をデータファイルを中心に
- プラグインライブラリーに向けてのモジュール化

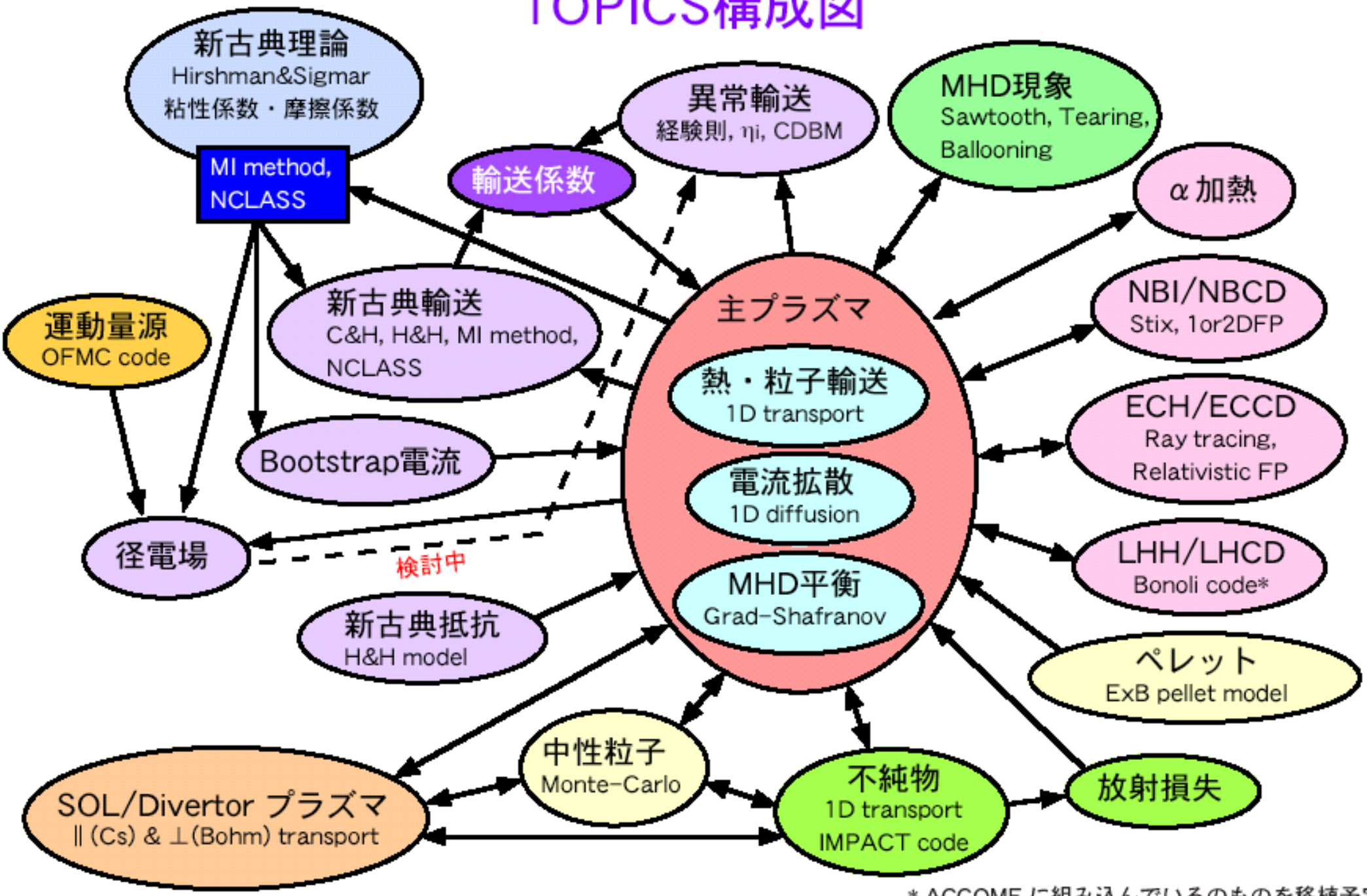
輸送コード(TOPICS）とは

TOkamak Prediction and Interretation Code System

- 2D MHD 平衡と1D 輸送コード
- 定常、非定常の実験解析（輸送係数等の評価）
- 輸送シミュレーション（輸送係数を基にしたシミュレーション）
- 電磁気計測とMSEデータを基にした平衡

- **関連コード**
  - **OFMC (Orbit Following Monte-Carlo) コード**
    - NBIによる高速イオンの挙動評価 (リップル損失、バナナ軌道損失、Charge exchange 損失）
    - 高速イオンの圧力、モーメンタムの評価
  - **高周波加熱・電流駆動コード**
    - ECH／ECCDの評価
  - **MHD安定性コード**
    - バルーニングモード、テアリングモードNTM、キンクモード

TOPICS構成図



＊ ACCOME に組み込んでいるものを移植予定

ITER建設に向けて輸送コードはどう変化

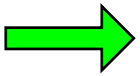
- 解析コードから予測シミューレションが中心に
- 核燃焼プラズマを扱う
- 精度の高い予測計算

  ーJT-60Uをはじめとして，実験データとの比較によりモデルの検証

  ー他研究所の開発した物理モデルの導入

TASKコードとTOPICSコードの関係

お互いに物理モデルをやり取りし，モデル検証を行うと共に，
物理モデル開発に二重投資をしないようにする。



物理モデルの導入を如何に簡単に行えるようにするか

プロジェクトとして，開発／維持管理の難しさ

特徴

- 組み込むべき物理モデルが多い
- 複数でコード開発／維持管理
- 他研究所で開発された物理モデルの組み込み

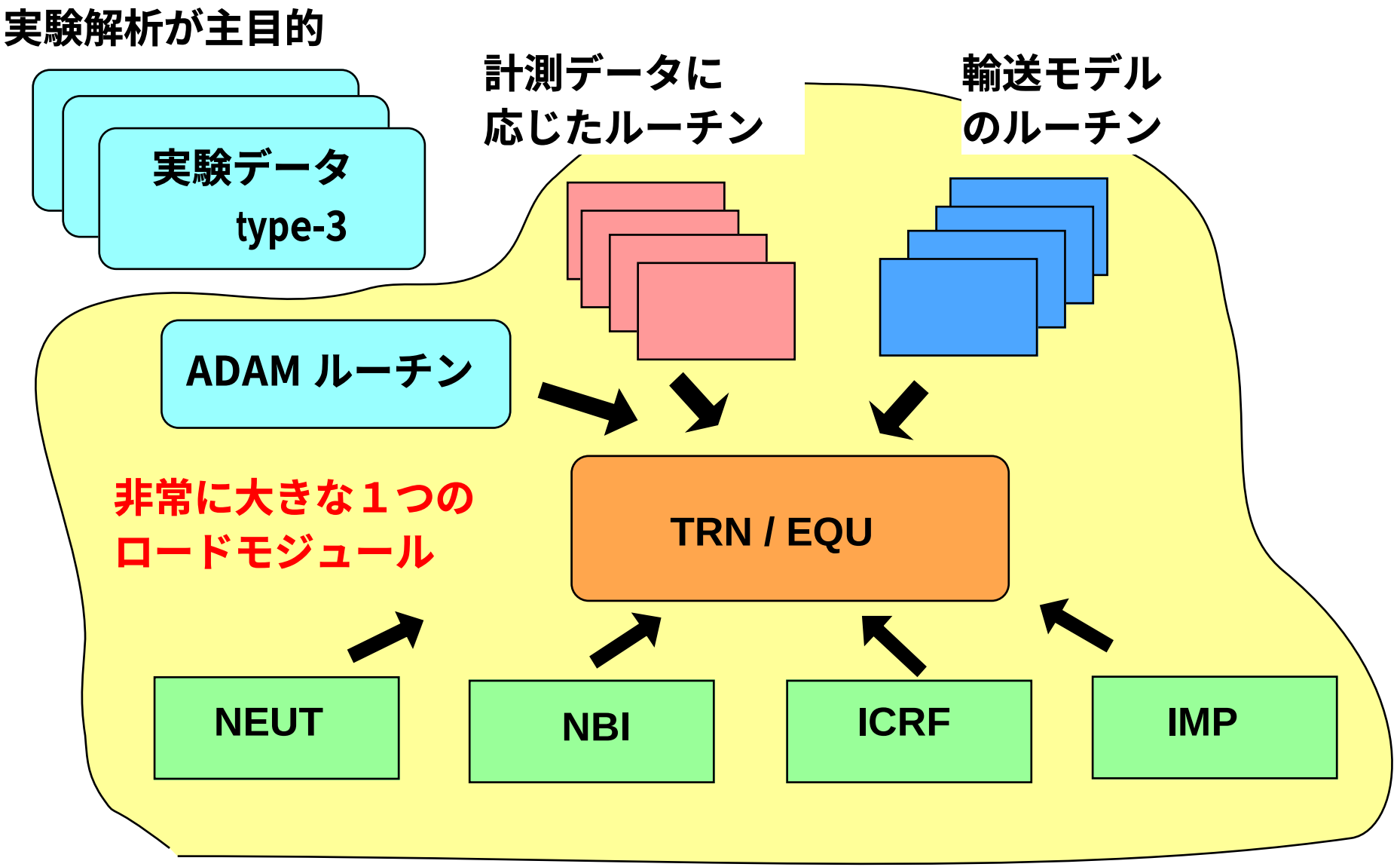
状況

- 物理コード間での干渉のため，導入に当たって修正が必要
- ソフト開発要員による維持管理
- ユーザからの絶え間ない要求

結果

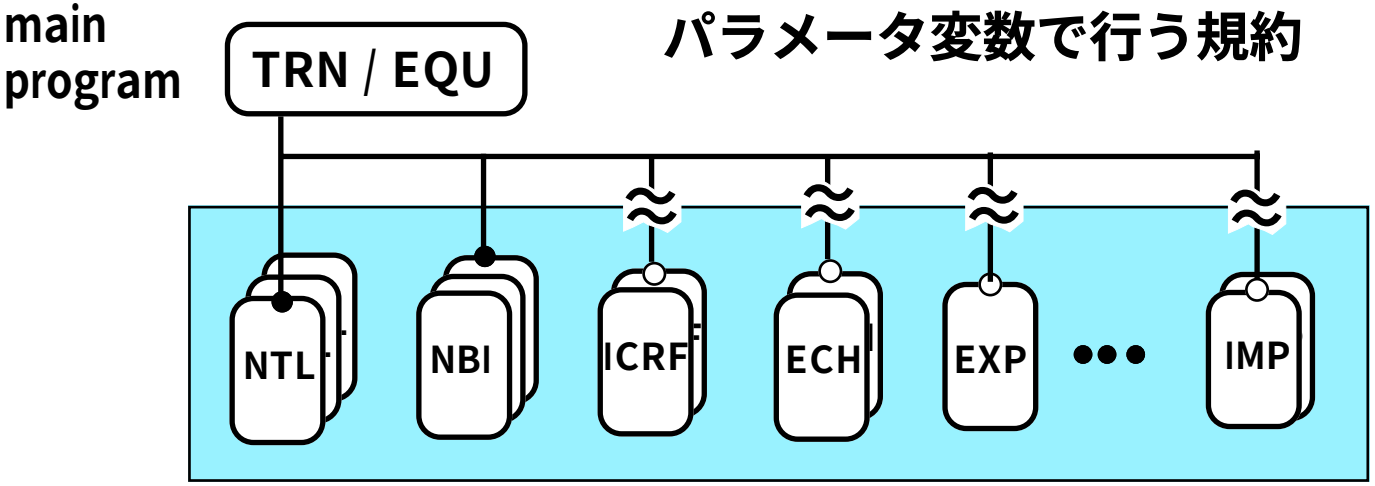
- コードが常に修正を受け，汚く，複雑に
  （オプション管理，多数のバージョン）
- コア部分の管理者一人に大きな負担

現在のTOPICSコード



ライブラリーの考え方

メインプログラムで，物理モデルのコールは
パラメータ変数で行う規約



ユーザが選択する物理モデルに応じて，ツリー構造を解析しながら，
必要なファイルを選択，修正しながら，コンパクトなロードモジュー
ルを作成する。

SPOT : source program organizing tool

（注）計算機がUNIXに変わったときにこの機能を落とす。
その後，個人のツールとして，UNIX簡易版を作成

問題点の解決方法

・コードが常に修正を受け，汚く，複雑に

（オプション管理，多数のバージョン）

・コア部分の管理者一人に大きな負担

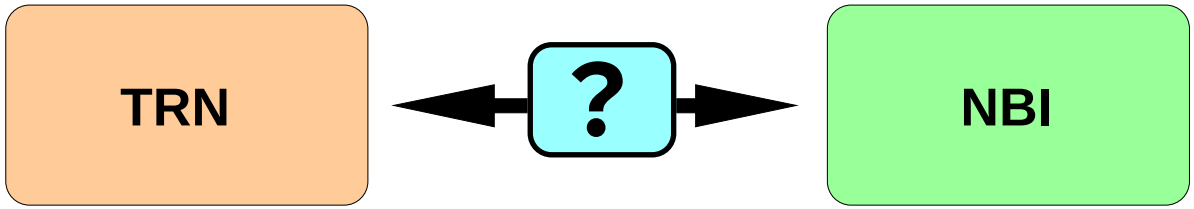
できるだけ，コア部分の基本構造は変えない

できるだけ，物理モデルの組み込みに当たっての修正は行わない

できるだけ，基本物理量から計算される二次物理量はポスト処理で計算

物理モデル間の干渉を少なくし，独立して開発が行えるようにする。

物理モデルの組み込み



- 引数を基本にしたルーチン

  引数が多いので，call文作成が大変。常に行われる修正に対応不可

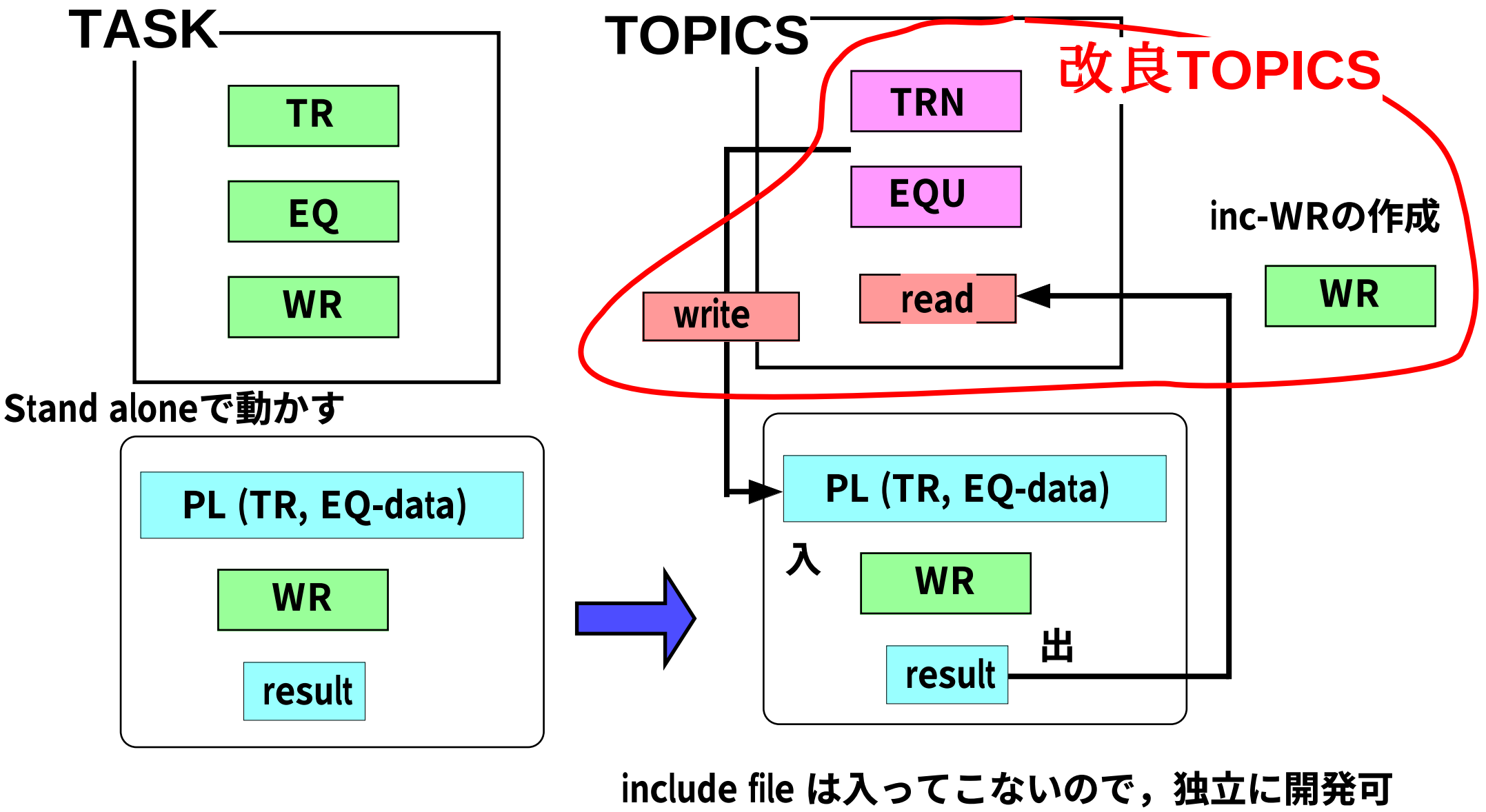
- インターフェイスルーチン

  インクルード（コモン変数を定義した）が必要なので，変数に同一のものが現れたときコードの書き換えが必要
  計算にどの変数が必要かが不明なため，物理モデルの理解が必要

★ データファイル　（そこからインターフェイスへ）

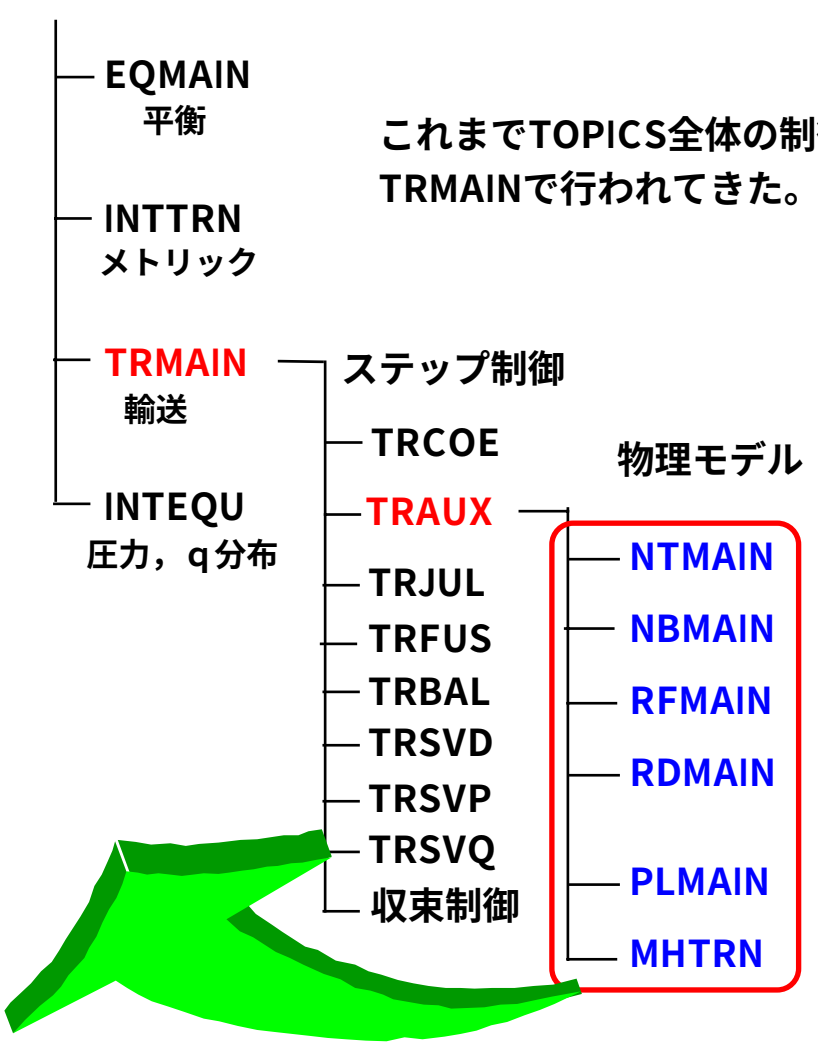
書き込み時と読み込み時の変数名変更ができ，干渉は避けられる
計算に必要な変数が明確

物理モデルの導入

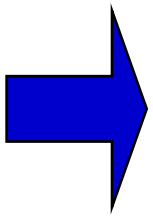


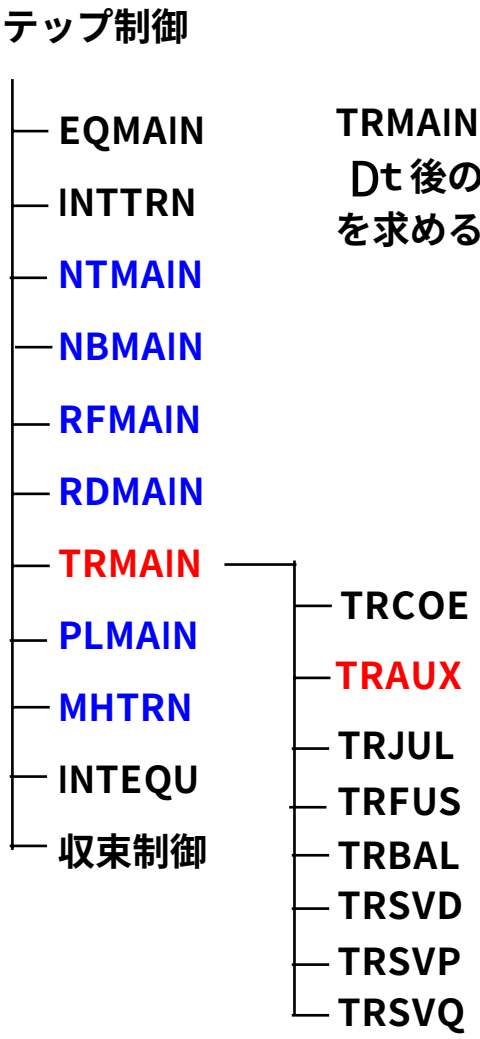
include file は入ってこないので，独立に開発可

TOPICSのツリー構造の変更



これまでTOPICS全体の制御は
TRMAINで行われてきた。





TRMAINは輸送方程式を解いて，
Dt 後のプラズマパラメータ
を求める物理モデルの一つ。

TOPICSのステップ制御

各モジュールのdriveルーチンは
istepで制御する

=0 の時：各モジュールの入力データの読み込み

=1 の時：各モジュールの初期分布の作成

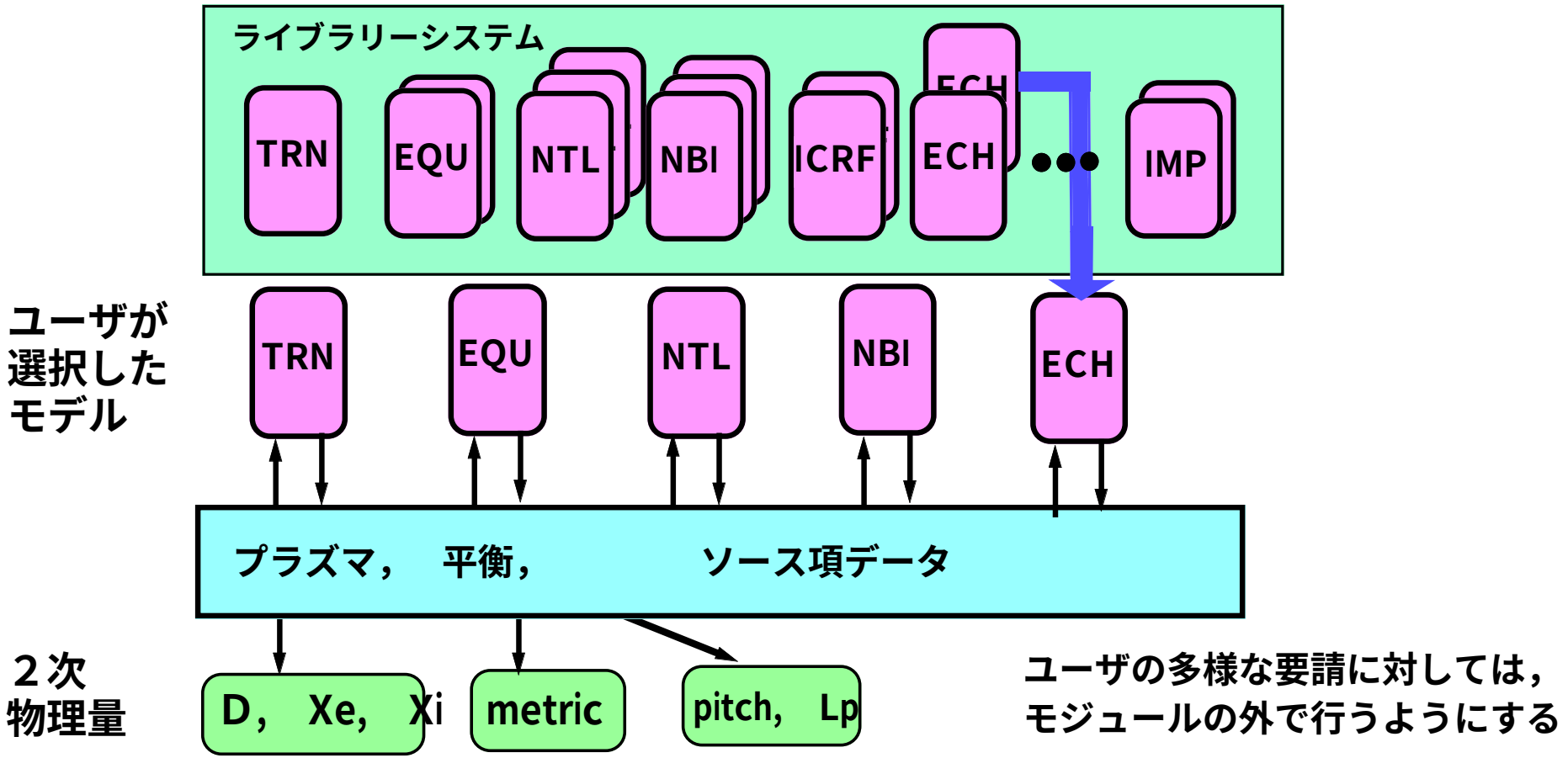
=2 の時：dtime後の分布計算

=3 の時：ポスト処理

ユーザのモデル選択に応じて，主プログラムが作成できる。

→ プラグインライブラリーへ

プラグインライブラリーに向けて



実験解析を目的とした場合，データの流れは，一方向
なので，データを制御する事は比較的簡単

予測計算を目的としたものに対して，このような変更は可能か？

モデル記述ファイル

ユーザがモデル記述ファイルを作成（標準的なものは準備）
　各モジュールのdriveルーチンのあるファイルを指定

```
TRN      : /analysis/topics/src/trn/trmain.f
TRNANO   : /j3859/ANMODEL/trano.f
TRNNEO   : /analyis/topics/src/trn/trneo.f
NTL      : /j3859/sonic/src/neut2d/aamain.f
           /j3859/sonic/src/mesh
NBIbth   : /analysis/ofmc/src/depositon.f
NBIslow  : /analysis/topics/src/nbi/stix.f
ICRF     :  no use
```

モデル記述ファイルより，主プログラムが作成される。

注：以前は，主プログラムの中に多くのオプションパラメータ

ロードモジュール作成

mspot　(mkmfに相当)

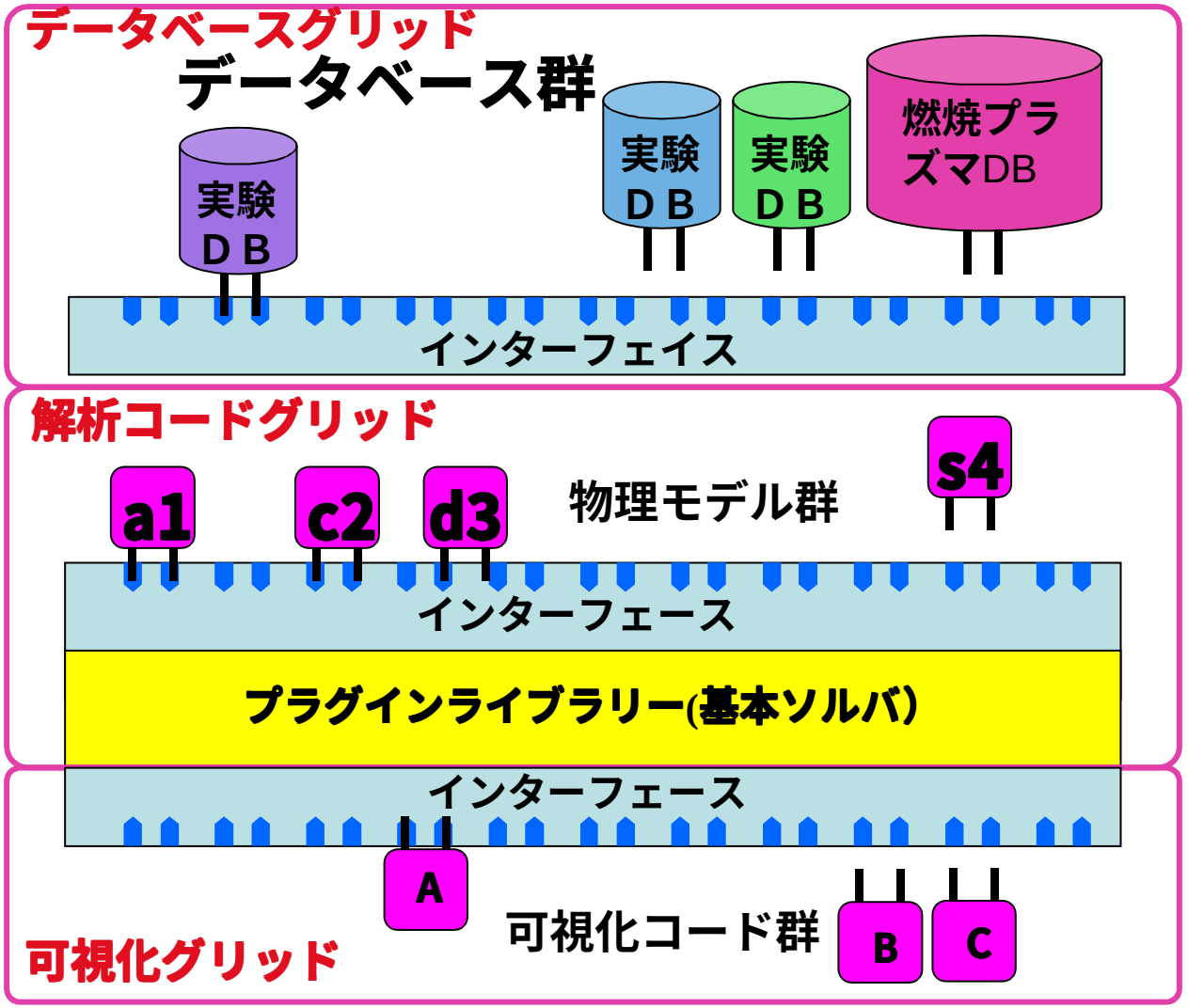
モデル記述ファイルを元に，driveルーチンのあるディレクトリを調べて，sub, fun, callのある文をファイルに出力。このファイルのデータを基に，ツリー構造を調べながら，使用するファイルをリストアップして，spot データ （makefileに相当）を作成

functionの組込を行うには，変数名解析を行わねばならず，この場合解析にはかなりの時間数分掛かる。functionについては，コンパイルして，足りないモジュール名として，function名が出るので，もう一度mspotを行って，追加する。

spot　(makeに相当)

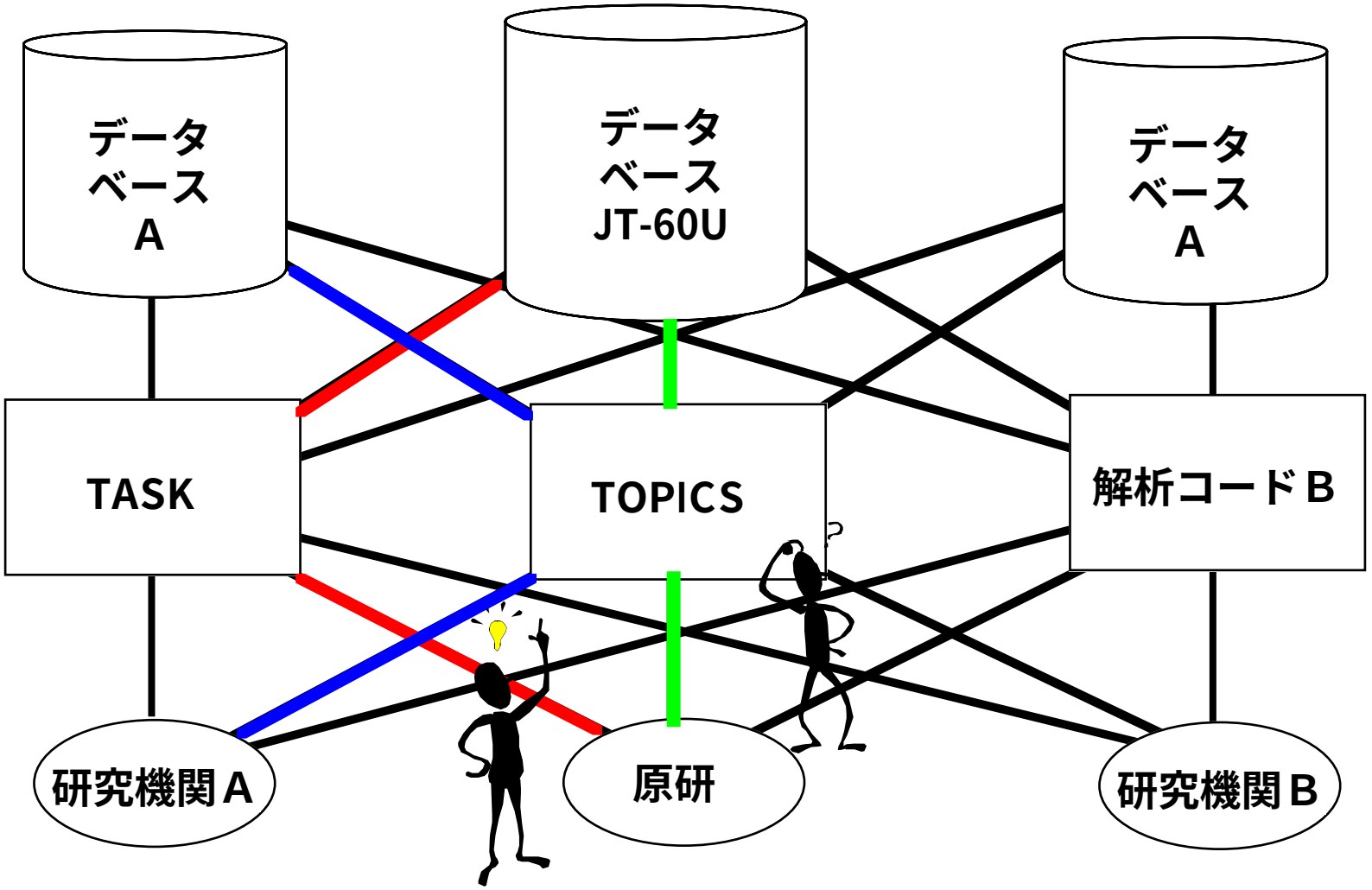
SPOTデータ（ディレクトリとファイル名の羅列したデータ）
の基にロードモジュールを作成

プラグインライブラリー



核燃焼プラズマ解析グリッド概念図

核融合研究グリッド


データベースA
データベースJT-60U
データベースA
TASK
TOPICS
解析コードB
研究機関A
原研
研究機関B

結　　論

- TASKとTOPICSとの協力関係　==>　物理モデルの比較／検証／導入
- コードの導入：stand aloneで動く入出力データを開発サイドで準備
- コード間の結合をデータとする事で計算に必要な変数が 明確になり，
  その後インターフェイス化する事は簡単
- TOPICSコードのプラグインライブラリーに向けてのモジュール化
  - ツリー構造の変更，各モジュールの処理の統一化(istep)，
  - ユーザ指定のモデル記述より，メインプログラムが作成
  - 各モジュールの入出力変数を明確化
  - インクルードファイル整理のための変数の解析ツールの作成
- データ掃き出しのフォーマット？　CDF（DEGAS2の原子データ）