2024年秋の大会

炉物理部会セッション

レガシーシステムの現状と課題
Current status and issues of legacy systems

レガシーシステムのブラックボックス化の現状と課題

Current status and issues of black-boxed legacy systems

*多田 健一[1]
[1]JAEA

計算コード開発と利用の観点からブラックボックス化の現状と課題についてまとめた。なお、ブラックボックス化というと既に開発者がいないコードを指すことが多いが、ここでは、国内において、コードの詳細を理解している人が少ないコードをブラックボックス化したコードとみなす。

1. ブラックボックス化したコードと同等の機能を持つコードの新規開発

ブラックボックス化したコードの新規開発の例としては、核データ処理コードがある。核データ処理は、評価済み核データと放射線輸送計算コードを繋ぐ重要な役割を担っているものの、世界的にも核データ処理に精通した研究者・技術者は少ない。開発されているコードも限られており、我が国では長年に渡って LANL の NJOY や IAEA の PREPRO が使われてきた。しかし、これらのコードに携わる開発者は限られている。PREPRO は元 LLNL の Red Cullen 氏が退職後も長年に渡ってメンテナンスをしているが、後継者は育っていない。NJOY の後継者はいるものの、ほぼ一人でメンテナンスしている状況であり、十分な開発体制が構築できているとは言い難い。

核データ処理コードについては、筆者が中心となって国産の核データ処理コードを新規に開発することになり、FRENDY Version 2 で多群断面積処理までが可能となった。FRENDY の開発は、従来の核データ処理の問題点が明らかになるなど、研究要素もあったお陰で順調に進んでいる。しかし、FRENDY の開発を開始するにあたっては、多くの苦労があった。また、開発が継続できたのは、ひとえに炉物理分野や核データ分野の関係者の手厚いサポートと応援のお陰である。核データ処理コードのように、ブラックボックス化したコードを復活させるには、多くの労力とユーザーの継続した応援が必要となる。

ブラックボックス化してしまったコードの新規開発で必要な要件をまとめると、次の通りになる。

(1) 上層部を納得させられるだけのコード開発の重要性の説明
(2) 産業界や大学など、ユーザー側の応援
(3) 研究要素や論文作成の見通し
(4) オープンソース化
(5) コード開発ができる若手の育成

なお、(3)については JAEA に特有の問題かもしれない。JAEA で若手にコード開発を担当させるためには、若手の博士号取得や昇進を考えて、論文を書ける見通しを示す必要がある。

上記の項目はどれも要素であるが、個人的には(4)と(5)が重要と考えている。オープンソース化により、開発者の増加が期待できるし、仮に開発が停止しても誰かが引き継いでくれる可能性がある。核計算コードのように輸出管理上の問題があるコードはともかく、基礎基盤技術についてはオープンソース化が世界的な主流となっており、オープンソース化されていないコードは使われなくなっていくと思われる。

また、実際の開発は将来的なメンテナンスを考慮しても若手が行うことが望ましく、コード開発ができる若手をいかに育成するか、またいかにこのコードの開発が面白いと思ってもらえるか、に成否がかかっていると言っても過言ではない。若手にコード開発に取り組んでもらう上での課題は、コードのノウハウ部分の

* Kenichi Tada[1]
[1]JAEA

技術伝承と、Fortran で書かれ、GOTO 文が多用されているスパゲッティコードを読み解くことの二点である。これらの内、後者のコードの読み解きについては、OpenAI 社の ChatGPT を始めとした生成 AI の登場により、かなりの部分が自動化できていると考えている。そのため、重視すべきは前者のノウハウ部分の技術伝承になる。ノウハウ部分の技術伝承については、文章化を行うことが最も重要である。例えば FRENDY では、核データ処理に関する計算手法の詳細を JAEA レポートにまとめている。幸いなことに、JAEA 内では核データ処理コードだけでなく、長年に渡って後継者不在の状況が続いていた燃焼計算コード(SWAT-X)や炉心解析コードの開発がスタートしつつある。これらのコードを公開するだけでなく、コード開発を通じて得たノウハウを JAEA レポートの形で残すことで、脱ブラックボックス化に貢献していきたい。

2. ブラックボックス化したコードの利用

NJOY や PREPRO のような核データ処理コードや、SRAC のような多群輸送計算コードでは、一子相伝の秘伝のタレのような入力設定値が存在しており、設定根拠も曖昧なまま使われ続けるということがよくあった。このようなよく分からない入力設定値があると、ユーザーが意図した入力になっておらず、計算結果がおかしくなる可能性がある。計算結果が明らかにおかしい場合には入力が間違っていることに気付けるが、実効増倍率が 1〜0.1%ずれるといった計算結果が正しいかどうか判断しづらい微妙な差異の場合、入力の間違いに気づけずに、誤った入力のまま解析を続けてしまう可能性がある。

このように、既存の計算コードを利用する際にも、注意すべき点が多数存在する。しかし、ユーザー全てがコード開発者と同等の知識を持つことは現実的に不可能である。そのため、計算コードをユーザーが意図通りに実行させるためには、

(1) 計算コードを修正して入力のチェック機能を拡充する
(2) 入力を自動生成するインターフェースを整備する

の二通りの対策が考えられる。(1)は FRENDY など、現在開発しているコードに対しては非常に有効な方法である。しかし、ブラックボックス化した計算コードに対しては、ソースコードに新たなバグを仕込む危険性があり、困難である。そのため、 (2)のインターフェースの整備が現実的な対応である。

JAEA では、様々な計算コードを自動的に実行するための汎用炉心解析システム MARBLE や、マルチフィジックスプラットフォーム JAMPAN を整備している。これらのシステムには、核計算コードの入力を自動生成する機能が実装しており、これらの機能を活用するのも一つの手かもしれない。

また、生成 AI で入力を自動生成させるために、教師データとなるようにサンプル入力データを増やすというのも今後は重要になってくるかもしれない。

3. 結論

ブラックボックス化したコードに対する対策として、コードのノウハウ部分の文章化と、入力を自動生成するインターフェースの充実化を挙げた。これらの対策を行うことで、仮にコードに詳しい人がいなくなったとしても、技術力を維持することが期待できる。

しかし、近年の生成 AI の発展はすさまじく、今後数年で今までの常識が通用しなくなってしまう可能性がある。そのため、上述した対策を行いつつ、生成 AI について理解を深め、ブラックボックス化したコードへの対策に生成 AI を活用することを本格的に考えていく必要がある。