第3章 コードの修正から詳細仕様書の生成まで

現代版文芸的プログラミングによるプログラム改善の実例7選

村井 和夫

ここでは実際にCopilot Chatを使って生成されたソースコードとコメントを使って文芸的プログラミングを実践しつつ，Copilotの実力を探ってみます．

1：ソフトウェア・リファクタリング（既存ソフトウェアの改善）

● Copilotは問題のあるコードをどう修正する？

Copilot Chatを使った代表的な機能が，既存ソフトウェアのリファクタリング（修正改善）機能です．

ソフトウェア教育で基本的なソフトウェアの書き方を示すときに使っている非常に多くの問題を抱えているソースコードを使って，Copilotがどのように修正してくるかを見てみます．

▶例題…UARTの受信割り込みプログラム

リスト1(a)は典型的な受信割り込みプログラムです．中堅クラスの教育でこれを見せて10分ぐらい議論させてみるのですが，このプログラムの問題点を的確に指摘して，誰でも分かるような形に書き換えてくださいと言っても，その場でできる人はほとんどいません．

まず，VSCodeでこのプログラムをいて関数部分を選択して，Copilot Chatに問題点を指摘しろと言うと，いろいろ指摘してきます．本質でないところは無視して，条件に局所性がなく最後までかかるなど，全体に可読性が悪いので修正してくださいというように具体的に指示をすると，**リスト1**(b)のように修正してきました．

▶Chatの指示で完成度を上げられる

ここでは，長くなるので，Chatの細かいやりとりは省略しますが，最初はコメントが全くないプログラムを返してきました．コメントもブロック単位で入れて分かりやすくしてくださいと指示すると，最初の不適切なコメント形式でなく，ブロックごとに章立てができた極めて分かりやすいコメントになっています．また変数名や関数名も分かりやすくしてくださいと指示をしています．

このようにCopilotは分かりやすい形に修正してきました．もちろん関数名や変数名を最低限分かるように変更しているのも，Copilotが修正しやすくなっている理由でもあります．

● Copilotの修正内容

ここで重要なのは，次の4点です．

1. 非常に読みにくい不適切な構造のソースコードの修正方法を提案できる
2. 関数名・変数名を分かりやすい名称に変更させられる
3. 後からでもブロックごとに適切なコメントを入れたりドキュメントを作成したりできる
4. 文法的な誤りや論理的な誤りなどがあるソースコードも問題点の指摘修正ができる

このように非常に品質の低いプログラムでもCopilot Chatを使うと，高品質のプログラムに書き換えることができ，適切なコメントやドキュメントも作成してくれます．

CopilotのようなLLMを使ったAIでは，One ShotやFew Shot（1つや少ない情報）の事前の学習で精度が上がります．さらにCopilotでは画面で開いている他のプログラムも参照対象になっていますので作業環境によって，同じ指示や質問をしてもかなり回答が変わってきます．そのため，いわゆるプロンプト・エンジニアリングや状況に応じた柔軟なChatのやり取りが必要です．

● Doxygen形式のコメントへの変更

文芸的プログラミングを実践するために，コメントからドキュメントの生成も行います．そこでまずはコメントを生成してみました．なお今のところ，CopilotにDoxygen形式のコメントにしてくださいと指示しても，あまり良い結果は得られませんでした．しかし，濃い網掛けの部分のように簡単な変更を加えてDoxygen形式のコメントに変更することにより，**図1**のようにソースコードそのものから詳細仕様書が完成します．