初心者用統合開発環境「双葉」の構築

山崎 秀輔 (Shusuke Yamazaki)

大阪大学工学部電子情報エネルギー工学科情報システム工学科目

yamazaki.shusuke@ise.eng.osaka-u.ac.jp

迅速なソフトウェア開発を実現するためのツールとして，統合開発環境という形態が登場して久しい．数々の統合開発環境が開発され，開発効率の向上を実現してきた．統合開発環境は学習用ソフトウェアとしての特徴も多く備えているが，そのような視点から配慮を行っているものはほとんど存在しない．本研究では統合開発環境を学習に用いることの有用性を示し，既存のものにはない機能を実装した「双葉」を提案，評価する．

1 はじめに

数学や物理の講義においてはやグラフが多用される。数式は概念を表記する道具としては優れているが、未知の概念を理解する道具としては必ずしもそうではないからである。同じ理由から、アルゴリズムの講義においても図やグラフが多用される。コードを最初に見ても理解が難しいからだ。

しかし、アルゴリズムを実装したプログラムを実際に実行すると、図示すべき値の数が多すぎたりして、詳細で正確な図を逐一描いていくのはほとんど不可能に近い。その結果、アルゴリズムの解説は概略図を用いた概念の説明に終わり、初学者はすでに理解したはずのアルゴリズムが実装されているソースを理解するのに七転八倒することになる。

よくできたデバッガにはプログラムがどの様に動作していくかを自動的に図示してくれるものもあるから、これをうまく用いればアルゴリズムとソースの理解を同時に行うことが可能になるのだが、そのようなデバッガはユーザーに大量の機能及び情報を提供するために初学者に扱いにくく、現実にこのような手法がとられることは稀である。

そこで著者は「初学者に理解しやすいこと」を最優先事項とし、デバッガ本来の機能は必要最小限に抑え、プログラムがどの様に動作していくかを視覚化する機能を充実させ、更に教材から統合開発環境を操作する仕組みを導入し、教材と開発環境を統合した教育用統合開発環境「双葉」を提案したい。

2 統合開発環境とは

統合開発環境とは**エディタ，コンパイラ，デバッガなどプログラミングに必要なツールを一貫したユーザーインターフェイスで扱うことができるソフトウェア開発環境**を指す．

ユーザーインターフェイスを統一することで，ユーザーはソフトウェアをどのように操作すればどのような結果がもたらされるのかについてある程度の予測を立てることが出来るため，訓練のために要する時間を大幅に短縮することが可能になる．

また，ツール単体ではアクセスできない情報にもアクセスが可能なため，次のようなより高度な機能を実現しているものが多い．

プログラミング言語を理解するエディタ プログラミング言語の構文解析を行って，種々の機能を実現したエディタ．この種のエディタは文法的な誤りを指摘してその原因や解決法を示したドキュメントへのリンクを示したり，プログラマーが入力したコードの最初の数文字からプログラマー

の意図を予測し，次に入力されうるコードを選択肢の形で提示したりする．また，コードの自動整形やリファクタリングを支援する機能を有するものもある．

データ構造を理解するデバッガ プログラム内に現れるデータの構造を分析して相応しい表示を行うデバッガ．プログラムのデータを単純な要素に分解して表示するだけでなく，データ構造の特徴を鑑みた視覚に訴えかけるような表現を適切に用いることで，デバッグの効率を向上させる．

このような機能を全て備えたソフトウェアとして，例えば Visual Studio 2003[1]，Eclipse[2] などがある．

2.1 統合開発環境の長所

今日においてもプログラミング学習は gcc などのコンソールアプリケーションを用いて行われることが多い．これは研究が主にそのような環境で行われてきた為で教育を受ける側，行う側の双方にメリットがある．

しかし，プログラミング学習において本質的でない箇所でつまずく初学者は多い．例えば，プログラミング環境を自力で構築できない場合，相応の教育機関に属していない限りプログラミング学習を始めることすらできない．

概してプログラミングは始めるまでに時間がかかるし，始めてからも本質的でない様々な知識が求められることが多い．教育するべき内容の本質とはいえない作業に困難がある状況では，教育効果は望むべくもない．

教育効果を更に向上させるためには，明らかに環境整備を自動化して学習者を本質的な部分のみに集中させる必要があるが，統合開発環境はこの点についてはっきりとした長所がある．

統合開発環境の導入は容易である．ほとんどの場合 10 回程度のクリックで導入が完了する．ライブラリ群の相性をも考慮する必要がある gcc の導入とは，比べるべくもない．また，ユーザーインターフェイスが統一されているため使いこなすまでに要する時間が大幅に短縮される．

以上の議論から統合開発環境を用いたプログラミング学習は今後益々広がっていくであろうと考えられるが，一方で，学習用のソフトウェアとして統合開発環境を見てみると改善すべき点も多いことがわかる．

2.2 学習用ソフトウェアとしての統合開発環境

統合開発環境はプロフェッショナルのプログラマが利用する開発用ソフトウェアとして登場して以来，ユーザー層として一貫してプロフェッショナルのプログラマを想定して進歩してきたソフトウェアである．

それは，小規模なソフトウェアを可能な限り迅速に開発するためのコードテンプレート自動生成機能の提供，大規模なソフトウェア開発を可能にするバージョン管理ツールとの統合がかなり早い段階で行われている一方で，教材や課題といったトレーニング用のツールが今日まで提供されていないことからも見てとれる．

今後は統合開発環境に組み込める様々なトレーニングツールが開発されて行くであろうから，統合開発環境にどのようなトレーニングツールをどのように組み込んでいけばより教育効果が上がるのかは，興味深い事象である．

3 学習用統合開発環境 双葉

著者は，統合開発環境を学習用ソフトウェアとして用いる試みとして双葉を開発した．双葉は.NET Framework[3] を組み込んだ Windows 2000/XP 上で動作するソフトウェアで，統合開発環境にトレーニングツールとしてデータ構造の可視化やアルゴリズムの教材などを組み込んでいる．双葉は無償公開されており Vector[4] で入手することができる．

大規模なソフトウェアで実験的な試みを行うのは難しいため，短時間で開発できる小規模プロジェクトとし，統合開発環境としての機能は必要最小限なものに限定した．

3.1 動作プラットフォーム

小規模プロジェクトであるため，プラットフォーム選択においては容易に開発が可能であることを優先し，パフォーマンスについては著しく品質を損なうものでない限り考慮に入れなかった．

開発において技術的な困難を伴うであろうと予想されたのはデータ構造の可視化を実現するため際に必要な，動作しているソフトウェアの情報を取得するデバッガとしての機能の実装であった．

Windows，Unix，MacOS Xなど様々なプラットフォームを検討したが，以下のような特徴を持つ.NET Framework を選択した．

OS 非依存 OS 非依存であればユーザーが学習した内容をより多くの場で役立てることができる．.NET 対応アプリケーションは，.NET Framework がインストールされている環境であれば，OS に依存せず動作することができる．実際，Shared Source CLI[5] として FreeBSD 上の実装が提供されており，Linux，Solaris，MacOS X においても Mono[6] がある．

無償コンパイラ 実用レベルコンパイラの作成は非常に難しく，小規模のプロジェクトには馴染まない．.NET Framework にはコード生成に関するライブラリが存在するため必ず何らかの言語のコンパイラがセットで提供される．

適当な規模のサンプルプログラム 利用できるサンプルプログラムを基に開発を行えば開発効率は飛躍的に向上する．.NET Framework 対応ソフトを開発するための SDK には，.NET Framework に対応するデバッガ cordbg がソースコード入りで組み込まれている．ソースコードのサイズは 800KB 程度で，17MB を越える gdb と比べると非常に小さく，小規模プロジェクトが利用するサンプルコードとして都合のよいサイズである．また，既に.NET Framework の次期バージョンに対応したサンプルプログラムとして mdbg[7] が Microsoft から提供されている．

豊富なドキュメント ドキュメントが貧弱なプラットフォームでは開発に必要な情報を取得するまでの時間が長くなりがちである．小規模プロジェクトの場合，これは特に重要であった．

3.2 可視化コンポーネント

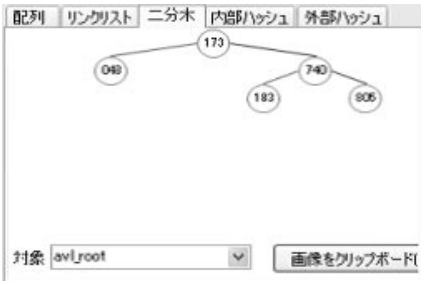

図 1: 二分木の可視化コンポーネント

データ構造の可視化コンポーネントはユーザーが開発したソフトウェアの中から可視化が可能なデータ構造を自動的に見つけだして可視化を行うため，システム定義済みのデータ構造の場合，ユーザーは特別な設定などを行う必要がない．

現在サポートしているのはソフトウェアを開発する上で不可欠な配列，リンクリスト，二分探索木，オープンハッシュ，クローズドハッシュである．図 1 に二分探索木の可視化コンポーネントを示す．

3.3 インターフェイス

初学者にとって使いやすいことを最優先事項としたユーザーインターフェイスを開発した．検索，行ジャンプ，コピー，貼り付けなどといった編集機能は OS に標準添付されるようなエディタのものを踏襲し，デバッグやステップインといったデバッガとしての基本的な操作は既存の統合開発環境のユーザーインターフェイスも取

り入れた．教材はブラウザ上で表示されるため，ユーザーインターフェイスは一般的な WWW ブラウザと同じものにした．

3.4 初等的なアルゴリズムの教科書

双葉には変数の宣言や制御構造などのプログラミング言語の文法から，AVL 木など比較的高度なデータ構造までの範囲を扱ったサンプルプログラムを含む教材が組み込まれている．

教材は統合開発環境と完全に一体化しており，教材に存在するリンクをクリックするだけで双葉が直接サンプルプログラムを開くことができる．このようなインターフェイスにより，初学者はより簡単にサンプルに触れることができる．

3.5 強調表示に対応したエディタ

C#の文法を考慮し，キーワードやコメントなどを強調表示するテキストエディタを開発した．強調表示する範囲は正規表現とキーワード指定によって行われる．

この手法は C#の文法においてはある程度正しく動作するが，プログラミング言語に因らず厳密に文法に従った強調表示を行うためには構文解析を行う必要がある．

3.6 対応言語

.NET Framework 対応のコンパイラを入手しさえすれば，どのような言語にでも対応が可能であるが，プロジェクトの目的を鑑みて対応言語は Windows2000/XP 向けの.NET Framework にコンパイラが標準で付属し，かつ仕様が ISO/JIS 標準となっている C#に限定した．これにより教材やエディタの開発が容易になった．

4 今後の課題

エディタの強調表示を完全に文法に沿ったものにしたり，コード自動補完機能を実現したりするためには構文解析が不可欠である．現在 Coco/R[8] を用いて実装を行っている．

また，積極的に広報活動を行うことで企業や各種教育機関での研修や講義，演習などに利用されることを目指す．

謝辞

双葉は情報処理推進機構 (IPA) が主催する平成 16 年度未踏ソフトウェア創造事業 未踏ユース部門で採択された開発プロジェクトである．開発にあたっては早稲田大学理工学部コンピュータ・ネットワーク工学科の筧捷彦先生，東京大学大学院情報理工学系研究科創造情報学専攻の竹内郁雄先生をはじめとして多くの方の支援を頂いた．この場を借りて厚くお礼を申し上げたい．

参考文献

[1] Visual Studio 2003 ウェブサイト. http://www.microsoft.com/japan/msdn/vstudio/.

[2] Eclipse Foundation ウェブサイト. http://www.eclipse.org/

[3] .NET Framework ウェブサイト. http://www.microsoft.com/japan/msdn/netframework/

[4] Vector における双葉 ウェブサイト. http://hp.vector.co.jp/authors/VA040884/

[5] Shared Source Cli ウェブサイト. http://www.microsoft.com/japan/sharedsource/openstandards/cspecs.mspx

[6] Mono Project ウェブサイト. http://www.mono-project.com/

[7] Mike Stall's .NET Debugging Blog. http://blogs.msdn.com/jmstall/

[8] The Compiler Generator Coco/R. http://www.ssw.uni-linz.ac.at/Research/Projects/Coco/