NII SPECIAL

NEW TREND OF SOFTWARE ENGINEERING

特集

「ソフトウェア工学の新しい潮流」

革新的な「双方向」ソフトウェアの開発

長年、ソフトウェア開発において、大きな課題となってきたのが生産性の向上である。従来は、ソフトウェア開発の工程で何か変更が生じた場合、上流まで遡って修正しなければならず大変な労力を要する。実際には、そのような遡及が行われないまま進行してしまうため、設計と実装が乖離する場合が多くみられた。そこで、胡教授らは、各所の修正を全工程に同期させる画期的な手法を開発、ソフトウェア開発を大きく変えようとしている。この「双方向モデル変換の言語的基盤技術に関する研究」とはどのようなものか、話を聞いた。

胡 振江
Zhenjiang Hu
アーキテクチャ科学研究系教授

日高 宗一郎
Soichiro Hidaka
アーキテクチャ科学研究系助教

ソフトウェア開発を単方向から双方向へ

胡　NIIでは、「グランドチャレンジプロジェクト」といって、次世代の情報学が立ち向かうべきさまざまな難問に取り組んでいます。私たちの「双方向モデル変換の言語的基盤技術に関する研究」は、その中のプロジェクトの１つです。これは、私をはじめNIIの研究者が中心となり、東大、電通大、芝浦工大、北京大学、上海交通大学などさまざまな機関の研究者が参画する、国際的なプロジェクトです。

この研究のキーワードは「進化」。ソフトウェアをただつくって終わりというのではなく、刻々と変化する社会に対応できるようなソフトウェアをつくり、変化に伴って進化させようというのが、この研究の狙いです。そのために必要な技術の核となるのが「双方向」です。すなわち、ソフトウェア開発において、フィードバック機能をもつ技術の開発を目指しています。

たとえば、何かレポートを作成して上司に提出すると、もっとタイトルを大きくとか、言い回しを変えてとか、赤字が入ってきますよね。それを修正して、提出し直すわけですが、それと同じことをプログラムの世界でもできないか、と考えました。これまで、ソフトウェア開発は上流から下流まで単方向で進められてきましたが、ここでは「双方向」ということで、下流の工程で修正を加えると、上流の設計のところまで、自動的に修正されるように、ループさせることでプログラムを進化させたいと考えています。

ところで、従来のプログラミングでは、指令の流れは一方向で、上流に遡ろうとすると、復路用にプログラムをもう１つ書く必要がありました。しかし、往路と整合性をもたせながら２つのプログラムを書いたり、人手で遡って変更したりすることは非常に困難です。そこで私たちは、進化するソフトウェアを開発するための基盤技術を構築し、さまざまな場面に応用したいと考えているのです。

――どのような考え方によってつくられたのでしょうか?

日高　ソフトウェアのプログラミングには言語が必要です。ところが従来のJavaなどの言語は、ほとんどが単方向のものでした。そこで、双方向に計算できるような言語を新たにつくろうと考えました。そもそもプログラミング言語というのは、計算を表すものです。そして、小さい計算をいくつも組み合わせて、大きな計算に組み立て、コンピュータに命令を与えます。そこで私たちは、この小さな計算の部品それぞれに双方向性をもたせることができれば、それを組み合わせて双方向のやり取りができる枠組みをつくれるのではないかと考えたのです。そのためには、いかにきれいに小さな部品に切り分けて、わかりやすく記述するか、というところがポイントになってきます。

加藤　実はソフトウェアの双方向変換については、1980年代から、重要性が指摘されてきました。しかし、それをどのように開発すればいいのか、具体

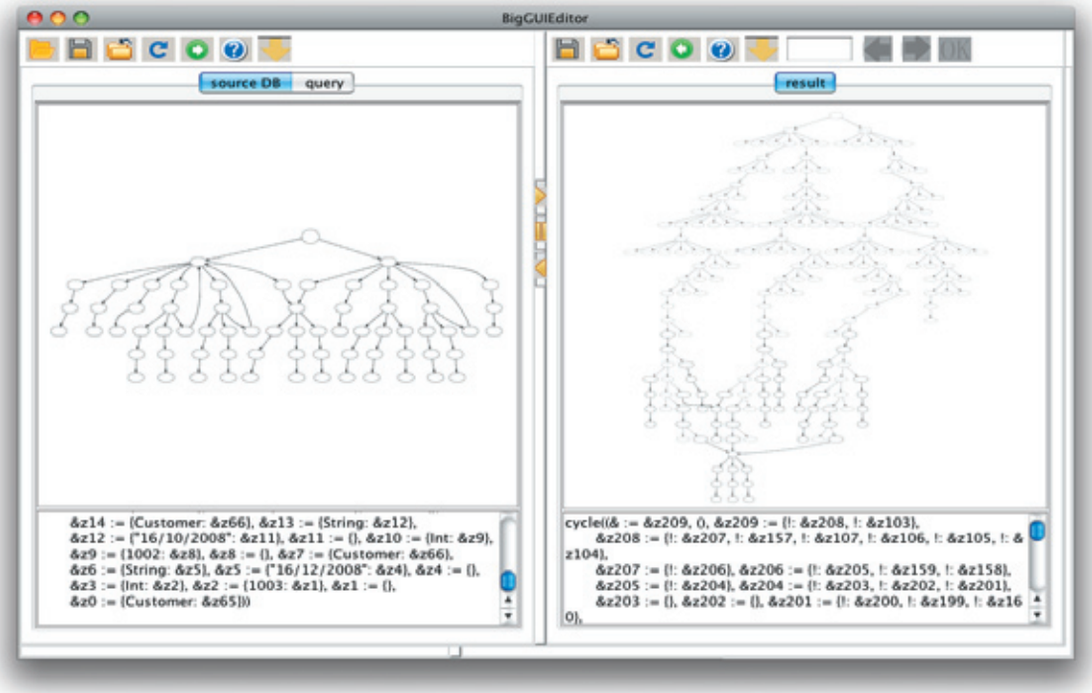

図1　複雑なグラフの変換例のパソコン画面。左のグラフを変換したものが、右のグラフとして表示される。

的な方法論を見出せないまま試行錯誤してきた歴史があります。80年代の成果としては、大きなデータベースに変更を加える際、変更したい部分だけを切り出して修正することができる“View updating”という手法の開発を挙げることができます。そして、2000年以降、双方向の研究が本格的になり、たとえば米国ペンシルバニア大学では、Web上のスケジュール帳に変更を加えると、パソコンやPDA、携帯電話など、違うデバイスでアクセスしても同一の情報が得られるという、同期のための言語を開発しています。

胡　私は2年前に東大からNIIに移ったのですが、私が所属していた武市研究グループでも、ドキュメント(プログラム開発の際につくる仕様書など)のための双方向言語の開発に取り組んでいました。これは、ホームページ(HP)のメンテナンスなどに利用できる仕組みです。HPに変更を加えたい場合、いちいち専門業者に依頼するのではなく、誰でもHP上で簡単に修正を加えることができれば便利ですよね。これを実現するのも双方向の変換システムです。

ちなみに、従来のプログラム言語はほとんどが、数列やデータベース、あるいは「木構造」を対象としてきました。木構造とは、ループをもたない階層構造のことです。そして、これまで開発されてきた双方向変換システムも、木構造上での計算を対象とするものがほとんどでした。ところが実際の社会をみると、上司と部下の関係にしても、電車の路線にしても、双方向にやりとりするネットワーク構造になっています。そこで、私たちは「グラフ」を対象にすることにしたのです。ここで言うグラフというのは、統計などで使う折れ線グラフやレーダーチャートといったものではなく、点(ノード)とそれを結ぶ線(エッジ)で構成されるものです。木構造と違って、ノードを共有していたり、ループを形成していたりするのがグラフの特徴で、グラフを対象とする双方向変換機構に取り組んだのは、私たちが初めてで、この研究分野の先駆けといっていいでしょう。

双方向性を実現する変換言語の開発

――具体的にはどのような手法で駆動させるのですか?

胡　ソフトウェアを開発する際、段階を踏んで実行可能な命令を表現していくのですが、その段階ごとの中間成果物(ソフトウェアの仕様/設計図)を、「モデル」を使って表現します。モデルというのは、ある対象を理解したり、操作したいときに、その対象の本質を定められた方法で表現するものをさします。そのモデル化には、モデリング言語UML(Unified Modeling Language:統一モデル言語)などを使います。この言語から、グラフィカルな記述で抽象化したシステムのモデルを生成します(図1)。つまりここで言うモデルとは、グラフを意味しているのです。

さらに、モデルから次のモデルへ、また次のモデ

加藤 弘之
Hiroyuki Kato
コンテンツ科学研究系助教

NII SPECIAL

New Trend of Software Engineering

ルへと変換していき、最終的にはソフトウェアを実行できるコード(グラフの形)を生成することができます。そして今回、私たちが開発したのが、このモデルの連なりに双方向性をもたせるため、逆方向の変換を可能にする双方向変換言語「UnQL+」です。

加藤　この言語は、既存の問い合わせ言語(コンピュータのデータに対して問い合わせをするための言語)を拡張したものです。つまり、グラフの情報を取り出すための言語を、グラフを別のグラフに変換するための言語へ拡張したわけです。ちなみに、その文法はSQL(問い合わせ言語の１つ)の構文に非常に近いものになっています。

この双方向変換言語を用いれば、実装の段階で変更が生じても、設計や顧客の要求分析まで遡って自動的に一貫性をもたせて修正することができ、ソフトウェアを進化させることができるようになります。モデルを使っているので、ソフトウェアの設計は実際のアーキテクチャと分離されているため、実装技術やアーキテクチャが変化しても、モデルを再利用することができるというわけです。

日高　では、実際にデモをお見せしましょう(図２)。今、画面の左右にモデルが出ていますが、このように、右のモデルで修正したい個所をクリックして、修正を加えると、対応する左のモデルの変更個所が赤く表示されます。どこをどう変更したかが、一目でわかるかと思います。実際のプログラムでは書かれるモデルは膨大ですので、このように一瞬で変更個所が同期して示され、目で見てわかるというのはとても重要なことなんですよ。ちなみに、下に見えている文字列が、上の図を双方向変換言語で表現したものです。今、一瞬で変換できましたが、実は最初に開発したものに比べると、そのスピードは100倍以上速くなっています。

――具体的にどんな場面に当てはめて考えることができるのでしょうか?

加藤　たとえば、ソフトウェアを開発する際に、最終的にテストをしますね。そこで不具合が見つかった場合、修正するわけですが、その修正を設計書まで遡って修正しておけば、その設計(モデル)をほかのプログラムに再利用できるようになります。たとえば自動車の設計にしろ半導体の設計にしろ、テストを経て修正した部分の仕様について、設計まで遡って情報を共有することはとても重要ですが、これまでは人手でやるしかありませんでした。特に、大規模なソフトウェア開発の場合は、設計と実装を手がける人が違っていることが多いので、情報を共有する上でも、後で設計を再利用する際にも、こうした自動的な双方向の同期の仕組みが必要になってくるのです。

胡　一言で双方向と言ってもいろいろな段階があり、たとえば、何か不具合が起こったときに、その原因がどこにあるかを上流まで遡って原因の個所を示す、すなわちトレーサビリティ機能もその１つですし、あるいは、どこかを変更しようと思った際に、どう変更したらいいのかヒントを与えたり、自動的に一貫性をもたせて同期させて変更をするといった機能もそうです。私たちの仕組みを使えば、さまざまなレベルで双方向性を実現することが可能です。

日高　プログラムを書くときに、誰か天才的なプログラマーがいきなり書き始めるということは稀で、通常はプログラミングにかかわる全ての人が理解できる共通の方法で、順を追って設計から実装まで進められていくわけです。また、世の中には似たようなプログラムが無数にあるので、双方向変換によりシステムの一貫性を保証することができれば、ソフトウェア開発のプロセスを再利用することができ、高信頼化、生産性向上に貢献できるようになると思います。

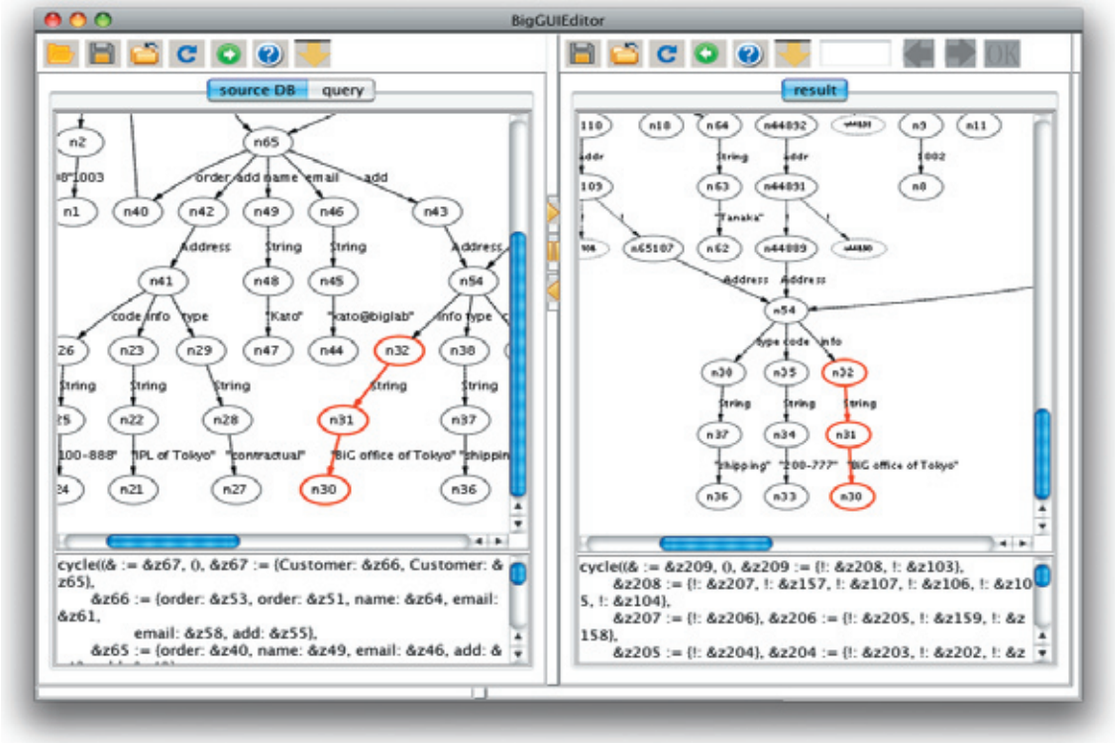

図2　ノードの対応関係を赤で表示したもの。右の画面で修正すると（赤い部分）、左の画面の対応個所が赤く表示されるため、一目で修正個所が特定できる。

注目を集める双方向変換システム

胡　私たちのこの研究は、ソフトウェア工学では世界で最も権威のある学会〔ICSE2009（New Ideas and Emerging Results Track）、FSE2009〕で発表されたり、北京大学からNIIにインターンとして来ている学生の論文がRuntime Model Workshopでベストペーパーに選ばれたり、注目を集めています。2008年には、NII内のソフトウェア工学に関する研究センター「先端ソフトウェア工学・国際研究センター（GRACE）」主催の、「双方向変換に関するGRACE国際集会」が湘南国際村センターで開催され、世界各国から、双方向変換で活躍している研究者が30名ほど集まり議論をする場を提供しました。このように、新しいコミュニティの形成において、私たちがリーダーシップをとっているということも、非常に重要なことです。

さらに、私たちの研究をより多くの研究者の方に知ってもらおうと、プロジェクトのホームページ（http://www/biglab.org）上に、これまでの論文やシステム、ソースコードを全て公開しているだけでなく、先ほどお見せしたように、実際にこの双方向変換システムを動かせるデモを用意しており、誰でも簡単にテストすることができます。最近では、研究機関や企業からの接触もあり、自動車の部品メーカーとディスカッションが始まったり、センサーネットワークの研究への応用が始まったりしています。

――今後の展望

日高　国際会議へ投稿したり、発表したりするたびに、さまざまな研究者から意見や批判、助言などをいただき、改良を加えることができたのは、大変よかったと思います。今後はさらに、このシステムの振る舞いの良さを詰めて、どういう性質をもっているのか、見極めていけたらと思っています。

加藤　私は、この仕組みをより効率よく、より速く動かすための改良をしていくと同時に、何でもできるシステムというのではなく、こういうことに向いているよ、と弱点も含めて示していきたいですね。そうすることで、本当に使いやすいものができるのではないかと思います。

胡　ソフトウェアのメンテナンスというのは、永遠のテーマであり、もっともコストがかかる部分なんですね。さらに一度つくったソフトウェアに新しい機能を付け加えるというのは、たいへん困難です。従来、私たちの研究チームは、プログラミング言語の開発や最適化、プログラムのつくり方についての理論を研究してきました。この研究は言語自体を一から開発し、ソフトウェア工学の新しい分野を切り拓くという壮大な構想であり、とてもチャレンジングな研究です。だからこそ面白い。メンバーは徹夜続きで大変ですが、今後、応用事例を増やしていって、世の中にこのシステムの有用性を示していけたらとても嬉しいですね。

（取材・構成　田井中麻都佳）