ソフトウェア工学分野の技術動向と展望

教授　秋口 忠三

１．はじめに

1960 年代に実施された大規模ソフトウェア開発プロジェクトで納期の遅れや品質問題が深刻化し「ソフトウェア危機」が共通認識となった。これに対処するためにソフトウェア工学という分野が生まれた。それから 40 年以上経過しているが「危機」の状況は続いている。半導体の集積化技術はコンピュータの性能（処理速度・扱えるデータ量・通信速度）を飛躍的に向上させ価格を劇的に低減させた。その結果ソフトウェアに対する需要はコンピュータの応用領域の広がりに比例して増大を続けているからである。

ソフトウェア工学では、ソフトウェア開発の結果として得られる各種の成果物であるプロダクトとそれを作成するプロセスの両面を扱う。成果物はその全てが仕様の記述である。要求仕様から各種の設計ドキュメント・プログラムコードに至るまですべての成果物は技術者の知的な営みによって生産される。ハードウェアの制約以外に物理的な制約を一切受けることがないために極めて複雑で大規模な構造物として進化を続ける宿命にある。人間の理解を優に超えるものになるリスクを常に抱えており、現にソフトウェアの欠陥を原因とする重大な事故が相次いでいる。このようにソフトウェア工学ではソフトウェア開発に関わる問題（ソフトウェア問題）に関する総合的な取り組みが必要とされ、扱うべき分野は極めて広範である。本稿では高品質なソフトウェア開発のための技術としてプログラミングを含むソフトウェアの仕様化技術を中心に現状の課題・技術動向・将来展望について所見を述べる。

２．ソフトウェア仕様化技術の技術動向

プロダクトの複雑さをいかにコントロールするかという課題は、ソフトウェア工学誕生の初期から中心的なテーマであった。1970 年代に提案された構造化プログラミングはプログラム構造の複雑さを抑えるための哲学としてプログラミング方法論・プログラミング言語設計の基本原理となり、構造化設計・構造化分析に発展した。構造化チャートや状態遷移図、デシジョンテーブルなどの図表表現を含む多くの仕様記述言語が提案されてきた。データとそれに対する処理を１つのモジュールにパッケージ化する抽象データ型の概念もこのような技術動向の文脈でとらえることができる。

1980 年代にはオブジェクト指向プログラミングが提案され、Smalltalk をはじめとして C++, Objective-C, Eifel 等のオブジェクト指向プログラミング言語が開発された。オブジェクト指向技術は、抽象データ型を実装するパッケージ化、型の属性と操作の継承、オブジェクトの多態性（インタフェースと実装の分離）の３つのメカニズムによってソフトウェアを構造的に仕様化する技術である。これらは 1970 年代に誕生した構造化技術を発展させたものと捉えることができる。

1990 年代には、オブジェクト指向技術はソフトウェア開発の現場で広く普及するようになった。プログラミング言語からソフトウェアの分析・設計フェーズの仕様化技術としても発展し多くのオブジェクト指向分析・設計の方法論が考案された。UML(統一モデリング言語)はその集大成といえる。1995 年に登場した Java 言語と UML はオブジェクト指向技術を一般に普及･定着させるうえで大きな役割を演じ、今日のソフトウェア開発の中で中心的な役割を演じている。オブジェクト指向技術はソフトウェアの再利用にも大きな革新をもたらした。オブジェクト指向技術と同時期に誕生した GUI は GUI コンポーネントとうい再利用可能なクラスライブラリの開発を促し、その開発経験の蓄積はデザインパターンの概念を生み出した。デザインパターンはオブジェクト指向によるソフトウェアの設計の知識・ノウハウを

パターンとして再利用可能な形式にカタログ化する試みである。ソフトウェアの分析やアーキテクチャレベル設計、開発プロセスでも同様の試みが行われている。

1990 年代に誕生した World Wide Web は爆発的な普及をつづけ Web アプリケーションというこれまでにないソフトウェアカテゴリを生み、ソフトウェア需要をさらに加速させることになった。Web エンジニアリングという言葉はこの状況を象徴的に表している。Web アプリケーション開発は従来のソフトウェア開発に加えて、要求の変化への迅速な対応・スケーラビリティ・信頼性とセキュリティ・ウェブコンテンツ開発との連携等の課題を投げかけた。

2000 年代には、オブジェクト指向技術はソフトウェア技術者の持つべき基本スキルとして認知されるようになった。Java, C#, C++, UML 等のプログラミング言語・モデリング言語、デザインパターンのような設計知識・スキルはソフトウェア技術者の必須のリテラシになった。有効性が明確になった技術の普及・定着が進む一方で、Web アプリケーションや組み込み系ソフトウェアの需要の増大とマルチコアプロセッサの登場など、ニーズとシーズの両面でソフトウェア仕様化技術には新しい課題が出ている。

３．ソフトウェア仕様化技術の展望

オブジェクト指向技術によってソフトウェアを構造化する基盤――インタフェースと実装を分離し、重複を排除して、再利用性に優れた独立性の高いコンポーネントを記述すること――が広い範囲で可能になった。しかしながらソフトウェアとして記述すべき要求内容は多様であり、それらをより簡潔に明確に重複なく表現するための仕様化技術は絶え間なく続いている。以下でその展望を概観する。

・アスペクト指向プログラミング

アスペクト指向はオブジェクト指向を補完する目的で考案された仕様記述法であり、関心事の分離を具現化する概念である。オブジェクト指向でいくつかのクラスに分散するセキュリティポリシー・ユーザインタフェース・データの永続化等の横断的な特徴をクラスアーキテクチャとは独立にアスペクトとして記述し、それををアーキテクチャの中に織り込む仕組みを提供している。Java に対して AspectJ などの言語が考案されている。

・より高い生産性の要求の高まり

Web アプリケーションの需要の高まりにより、Ruby や Python のようなより生産性の高いプログラミング言語に注目が集まった。性能よりもプログラムの記述性を重視し同じ機能をより少ない記述で表現し短期間で動作するプログラムを作成できる点に特徴がある。

・関数型プログラミング言語

マルチコアプロセッサの性能を引き出すために並列プログラミングの必要性が高まり関数型プログラミングが注目をあつめている。Java 言語と互換性をもちオブジェクト指向と関数型の両プログラミングパラダイムをサポートする Scala 言語が Java の後継言語として注目されている。

４．むすび

ソフトウェア工学は増大するソフトウェア需要に対して納期遅れや品質問題と格闘し続けなければならない。本稿では高品質のソフトウェア製品を開発するためのソフトウェア仕様化技術を中心に技術動向と展望について概観した。仕様化技術と開発プロセスはソフトウェア危機に対処するための両輪として継続した改善の努力が必要である。開発プロセスについては別の機会に所見を述べたいと思う。