連載　プロマネの現場から
第 57 回　ソフトウェア工学の未来像
・・ケイパー・ジョーンズ『ソフトウェア工学のベストプラクティス』

蒼海憲治（大手 SI 企業・金融系プロジェクトマネージャ）

新年にあたって、「ソフトウェア工学の未来がどういう姿になっているか」を語っているケイパー・ジョーンズさんの新刊から、ケイパーさんの描く未来像を紹介してみたいと思います。

昨年、版を重ねている『ソフトウェア開発の定量化手法』や『ソフトウェア見積りのすべて』を通して、以前からお世話になっているケイパー・ジョーンズさんの新刊『ソフトウェア工学のベストプラクティス ―ソフトウェア工学の真の工学への発展をめざして―』（＊１）の邦訳がでました。原著は、２０１０年に出ていますが、分厚いものだったので、訳出されて身近に手に取れるようになり、感謝しています。

今回の内容は、ソフトウェア構築におけるベストプラクティスが主テーマになっていて、現在ソフトウェア工学が関わっている様々な分野のベストプラクティスが紹介されています。狭義のソフトウェア工学のスコープである、要求定義から設計・開発・テストだけでなく、ＩＴエンジニアの採用基準・評価・育成計画、ＩＴエンジニアやプロジェクトマネージャのモチベーションやモラル、アウトソーシングや各種スペシャリストの活用法、エンジニアリングに関わる効果的な手法・ツール・プラクティスとその選択基準などが幅広く、目配り良く紹介されているのが特徴です。

そして、現在のベストプラクティスを概観した後、第３章の「２０４９年頃のソフトウェア開発と保守についての予見」において、執筆時から４０年後のソフトウェア開発の現場の未来が語られています。

プログラムの自動化など、過去においても、例えば１９８０年代であれば、２０００年頃には実現していると巷間にいわれていたような気がします。しかしながら、本書では、改めて、ソフトウェア構築の自動化や再利用を図るための条件を明らかにし、その条件を解決するために必要な道具立てを示しています。

ケイパーさんのソフトウェア工学に対する見方はこうです。１９６０年から現在にいたるまで、ソフトウェア工学は、基本的に「工芸」的なものあり、「工学」にはなっていなかった。複雑なソフトウェアアプリケーションは、それぞれ個別に設計され、１行ずつ書かれるソースコードから構築されてきた。このような開発は、効率的でも経済的でもなく、品質も安定せず、セキュリティも満足なものではなかった。このような状況

のソフトウェア工学は、芸術的表現から、真の工学になってほしい、といいます。

≪ソフトウェアが真の工学分野になるためには、
コード開発だけでなくもっと多くのことに関心を寄せる必要がある。
アーキテクチャ、要求定義、設計、コード開発、保守、顧客サポート、訓練、
文書化、尺度と測定、プロジェクト管理、セキュリティ、変更管理、ベンチマーク、
およびその他の多くの主題を考えなければならない。≫

取り上げられているソフトウェア工学のスコープはとても広いのですが、今回は、狭義のソフトウェア工学のスコープである要求定義から設計・開発、そして保守について見てみたいと思います。

１．未来の要求分析

未来の要求分析は、まず最初に、インテリジェントエージェントやアバターを使って、Ｗｅｂやデータベースに蓄積された、ソフトウェアや技術文献、マーケット情報などあらゆる関連情報を収集、分析することからはじまるといいます。

≪・・ソフトウェア再利用が成熟し広範囲にわたる再利用成果物のカタログが
利用できるようであってほしい≫
といいます。
再利用されるソフトウェアは、品質やセキュリティが保証されていること。

また、インテリジェントエージェントは、

≪・・計画中の機能と類似の機能をもつすべての既存特許を収集し分析するために
起動されることになるだろう。≫

なぜなら、特許侵害による巨額の出費により、開発中止になることを未然に防ぐことが必要とされているため。

さらに、

≪・・早期の規模予測機能、開発時の要求変更を予測する機能、
顧客の学習曲線のコストを予測する機能も特許の保護が必要である。≫

現時点の見積りツールには、このような３つの機能を持つものはない。

しかしながら、現在の再利用のレベルはまだまだです。

≪・・ＳＯＡもＯＯ（オブジェクト指向）もアルゴリズムやビジネスルールのレベルでは古いアプリケーションの利用を正式には含めていない。・・
さらに、ＯＯパラダイムは惜しいところまで近づいているものの、
両者ともにすべての新機能を再利用可能な開発対象とは見てはいない。

また、ＳＯＡとＯＯ手法の品質管理とセキュリティプラクティスは本当に安全なアプリケーションにとって必要な厳格さを持ち合わせているとはいえない。≫

２．未来の設計

≪設計は、すべての類似アプリケーションのアークテクチャと設計の注意深い分析から始まる。≫

現在と未来の設計の非常に重要な差異の１つは、

≪企業内部、商業組織、あるいは品質が保証された再利用可能な機能ライブラリからの多くの標準的な再利用部品の利用であろう。≫

≪注意すべきは、経済的に成功するためには再利用成果物はゼロ欠陥レベルに近いことの保証が必要なことである。≫

≪類似の古いアプリケーションからの情報を利用するためにはエキスパートシステムによる設計ツールが必要である。

このツールは静的解析、複雑度分析、セキュリティ分析、アーキテクチャと設計の構造分析の機能を含むだけでなく、古いコードからアルゴリズムやビジネスルールを抽出する機能ももつことになるであろう。≫

上記の機能は、とてもハードルが高いと思いますが、

≪既存のアプリケーションの構造、機能、性能およびユーザビリティの分析が
できるエキスパートシステムは、ソフトウェアを工芸から工学に変えるための基本的な要件であろう。≫
といいます。

３．未来のソフトウェア開発

固有アプリケーションのコードを１行ずつ書くスタイルから、品質やセキュリティが保証されたソフトウェアの再利用が前提となる。

そのためには、新規に開発されるにあたっては、再利用を考慮したものになる必要があります。

≪・・後に加えられる新しい機能に配慮しつつ、既存の再利用可能なコンポーネントをすべて集めて、機能するプロトタイプを組み上げることであろう。
このプロトタイプはユーザビリティ、性能、セキュリティ、品質等の基本的な評価に用いることができる。≫

≪アプリケーションの新機能は単一の利用を目指すのではなく、
それ自体が再利用可能なコンポーネントになることを目標に計画される・・≫

そのためには、品質とセキュリティの欠陥を発見できるインテリジェントエージェントとエキスパートシステムが必須となります。

４．未来の保守と機能拡張

≪ソフトウェアアプリケーションの平均寿命は１０年から３０年以上に及ぶために
真の工学分野としては開発のプロセスのみに注目するだけでは不十分であり、
保守（欠陥修復）と機能拡張（新機能の追加）を含める必要がある。≫

ソフトウェアのアプリケーションは、運用開始後の規模拡大率は年８％ともいわれており、７年で倍になるペースになる。また、規模拡大にともない、複雑度も増していきます。そのため、ソフトウェアの老朽化、衰退速度を遅らせるために、運用後５年くらいでリノベーションをする必要があります。

≪２０４９年頃における保守あるいは欠陥修復では、
バグ報告とその原因箇所の究明を統合する強力なワークベンチ、
自動化された静的／動的解析ツール、
テストライブラリとテストケースへのリンク、
テストカバレッジ分析、
および複雑度分析ツールが利用できるはずである。

自動テストケースジェネレータ、および
コードリストラクチャリングツールや
言語翻訳のような、より特化したツールも現れているであろう。≫

最後に、ちょっと毛色の変わった話題ですが、未来の訴訟を取り上げてみます。
意外にも、ソフトウェアのソースコードや設計が密接に関わってきます。これまで分からなかったことが、分かるようになることで、問題が顕在化する、と指摘しています。

５．未来の訴訟

≪新しい種類の訴訟が間もなく出現するであろう。
データが盗まれ、何千もの顧客や患者が、個人情報の盗用やその他の損害を
引き起こした企業に対して訴訟を起こすことが予想される。
データを盗んだ真の犯人を逮捕することは難しく、
他国に住んでいることもあるため（時には他国政府もある）、
不適切なセキュリティ防護のためにデータを盗まれた企業が責任を問われる訴訟になる。≫

≪ソースコードを分析し、静的解析、複雑度の分析、
不正にコピーされたコードの検出、
ＦＰ尺度による規模の定量化、
テストカバレッジの検討、
欠陥多発モジュールの発見、
セキュリティ欠陥や性能ボトルネックの探索、

その他の重要な分析を行うことができる強力な分析エンジンは、
訴訟問題および老朽化アプリケーションの保守に有効な支援ツールであろう。≫

・・もちろん、こんな統合ツールはまだありませんが、
再利用の効率化のためには、４０年後といわず、数年以内にほしいと思います。

いまから４０年後のソフトウェア開発現場に必要な道具立てがいかなるものか、現時点で分からないものもあります。しかしながら、そのような場合でも、あるべき未来像から逆算することで、現在の仕事の効率化や、効果を求めるべき領域が明らかになってくると思います。

（＊１）ケイパー・ジョーンズ『ソフトウェア工学のベストプラクティス ―ソフトウェア工学の真の工学への発展をめざして―』共立出版　２０１２年刊　富野 壽 (監修), 岩尾 俊二 (監修), 水野 哲博 (翻訳), 吉田 善亮 (翻訳)