システム開発“ 思想 ”の転換をめぐって

野村総合研究所　石田裕三

システム開発はいま転換点に立っていると言えるかもしれない。短期的な生産性や高い処理性能を求めて同じようなシステムを繰り返し開発するやり方は、そのコストを負担する顧客にとってもすでに限界にきているからである。本稿では、中長期的にシステムの開発効率を高めるための基本的な考え方について、おもにソフトウェアアーキテクチャーを中心に考察する。

受託開発から企画提案へ

システム開発にはある種の無駄が発生するものである。たとえば、同じようなシステムを莫大な労力をかけて開発する、同一システム内に同じようなロジックが散在する、次々と移り変わる開発言語に振り回され、ハードの老朽化やOS（基本ソフト）のサポート切れに対処するために莫大な移行作業が必要になる、といった具合である。このような“ 力作業 ”の連続では、生産性向上の工夫を図ったとしても、顧客がそのコストを負担できなくなった時に受託開発型のビジネススキームは崩壊する。

しかし、ERP（統合基幹業務システム）やその他のパッケージソフトだけでは対応できず、受託開発業務が必要な分野は依然として大きい。したがって、開発側としてこの抜本的問題に取り組むには、受託開発から企画提案型の開発へと転換を図る必要がある（図１参照）。急速なソフトウェア技術の革新、ハードウェアの低価格化、そしてオープンソースを含めたノウハウの世界的な流通は、この新しい“ 思想 ”を後押ししてくれるはずである。問題は、このニーズに対してよりスピーディーにかつ効率よく対応し、またグローバルなシステムの利用を視野に入れ、国ごとに異なる制度や慣習に柔軟に対応するシステムとするためにどのような“ 思想 ”が必要かということなのである。

品質属性のトレードオフ分析

新しい思想に基づくシステム開発への転換にとって大きな障害となるのは、過去の実績を過度に重視する姿勢である。システム開発では無数の意思決定プロセスを経るが、そのなかで選択肢の間に存在するトレードオフを認識し、必要ならばあえてリスクもとれる組織の体質を作ることが重要である。ここではソフトウェア開発の肝となるアーキテクチャーに焦点を絞って考えてみたい。

一般にソフトウェアアーキテクチャーはさまざまな視点（品質属性）からの分析・検討を経て決定される。しかしその際に重要視するポイントは、システムに関わる関係者それぞれの立場によって大きく異なるため、システムのライフサイクル全体を考えた最適なアーキテクチャーを選択することは簡単ではない。そして、概して短期的な品質属性、たとえばユーザー機能を実装するための“ 生産性 ”

図１　ソフトウェア商品のマッピング

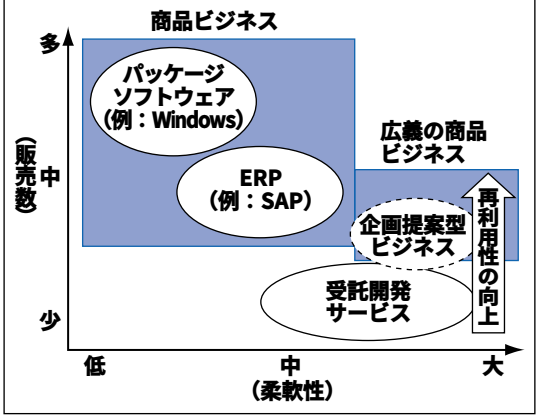
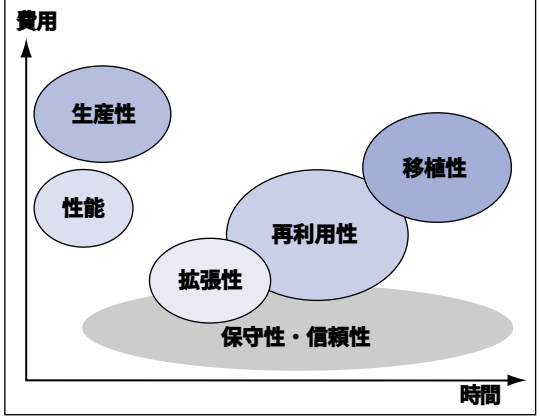


図２　品質属性に関わるコストと時間の関係



や、処理能力を確保するための“性能”が優先されやすい。

しかし短期的な品質属性の最適化だけでなく、機能の追加・変更を効率よく行える“拡張性”や、中長期的に生産効率を上げるための“再利用性”、プラットフォーム変更時の影響を極小化するための“移植性”などにも同様の配慮が必要である（図２参照）。開発コスト削減効果が現れやすい短期品質属性の偏重は、中長期的な品質属性を悪化させ、その対価として将来のコストを増加させるからである。

実プロジェクトへの適用

NRI（野村総合研究所）は、流通業におけるシステム再構築のプロジェクトにあたり、米国カーネギーメロン大学ソフトウェア工学研究所が提唱しているソフトウェアの品質属性のトレードオフ分析手法、ATAM（Architecture Tradeoff Analysis Method）を用いてアーキテクチャーを決定した。

その際に検討されたおもな品質属性は、短期的な視点からは処理速度（オンライン・バッチ共通開発言語で大量バッチ処理に耐えられること）、生産性（スキルのある開発要員を確保すること）、信頼性（未経験の技術を問題を起こさずに利用すること）の３つであった。一方、長期的には、再利用性（長期的な開発効率化と品質向上が実現できること）、移植性（多様なシステムインフラに対応できること）、保守性（コーディングより設計に時間をかけることで将来コストを削減できること）という品質属性が問題になる。

これらの品質属性間にはたしかにトレードオフが存在する。たとえば、長期的な品質属性の可能性に期待することが短期的にはリスクをとることにつながる、といったことが起こる。今回のプロジェクトではこうしたことを認識した上で、長期的生産コストの削減、新規顧客獲得の可能性拡大、新マーケット展

開を睨んだ基盤作りなどを重要視し、あえてリスクをとる決断をした。

柔軟性を重視したアーキテクチャーの採用

今回開発した商品の特徴は、対象となる業務ノウハウを整理・分析・統合し、顧客ごとに異なるデータ構造、ユーザーインタフェースの相違といった要件変化を柔軟に吸収できるようにした点である。

そこでまず留意したのはソフトウェアの構造（アーキテクチャー）である。柔軟性の高い構造を実現するために、徹底した階層化・部品化による修正範囲の極小化を目指した。そのために次のような技術を適用した。

①データ構造をオブジェクトとして操作可能とするOR-Mapping（リレーショナルデータベースをオブジェクトとして操作する）技術

②ユーザーインタフェースのカスタマイズの容易性、リッチクライアント対応を考慮したTemplate Engine（オブジェクトをテンプレートから参照する）の採用

③コアモジュールについては局所的なビジネスロジックの変更を可能とする「簡易コンポーネントフレームワーク」の開発

第2に留意した点は、顧客ごとに異なるシステムインフラを考慮し、プラットフォーム非依存性を可能な限り追求したことである。プラットフォームに依存しないことでソフトウェアの導入はスムーズになり、同時にシステムのライフサイクルが長くなる。顧客は導入にあたってプラットフォームを自由に選択でき、プラットフォームが古くなってもアプリケーションを修正せずにスムーズに最新機種に移行できるからである。また多くの顧客に導入が進めば、一顧客あたりの基盤技術に対する費用を抑えられる。開発側にとっては、アプリケーションの動作確認を顧客ごとの環境で用意する必要がなくなり、コストを抑えられる。プラットフォーム非依存性に加えてアプリケーションサーバーなどの基盤ソフトにおいても、ベンダー非依存とし、また今回開発した商品の中立性にも配慮した。

第3に留意した点は、統一言語での開発による長期的生産コストの削減である。従来はオンラインとバッチで共用できる処理を異なる言語で別々に開発し、テスト・保守してきたが、今回のプロジェクトではオンライン処理、バッチ処理、簡易運用システムまですべてJava（プログラム言語のひとつ）で実装している。プラットフォームに依存しない統一言語での開発により、オンラインとバッチアプリケーション間で処理の共通化が可能となり、大幅な生産コスト削減が可能となった。さらに、開発言語の統一は、人材の教育・確保・流通という観点からも大きなメリットが期待できる。

理論（理想）と実践（現実）の調和

今回のプロジェクトは、コンピュータサイ

エンス（ソフトウェア工学）の世界で以前から謳われてきた理論を実プロジェクトに適用する大きな挑戦でもあった。しかしソフトウェア工学の技術だけでは解決できない点が多々あったことも事実である。たとえば業務部品の適切な大きさを決めるには、要件変更を予測できる豊富な業務知識と、再利用性や性能などを考慮できる技術力の両方が要求される。また、実装段階に至ってもさまざまな局面で品質属性間のトレードオフを判断することが求められた。この間、開発者間の意識合わせを行うために、設計およびコーディングのインスペクションを試みたが、再利用性と生産性とのトレードオフに対して、開発期間を遵守するために生産性を優先せざるを得なかった部分もあった。

再利用性については次のようなことにも注意する必要がある。すなわち、ユーザーインタフェースとデータ構造が比較的近い関係にあるような、ロジックの少ないアプリケーションでは、交換可能な部品が間に入り込むメリットが少なく、無理に既存機能を再利用しようとするとかえってコストが嵩むことがある。このような場合は、開発方法論と設計・コーディングパターン、各種ユーティリティ群を整備・確立することで対応することが望ましい。無理に再利用性を追求するより、短期的生産性を優先するほうがトータルの生産コストを削減できる場合もあるということである。

このように、理論をそのまま実践することが必ずしも正解であるとは言えないが、いずれにせよ再利用性を高め長期的な生産コストを削減するという努力は、今後ますます重要になってくるであろう。なぜなら、顧客が扱うデータ量が近年ますます増えると同時に、アプリケーションへの要求機能が複雑化し、それに加えてリアルタイム化への要求も強くなってきているからである。これら開発・テストが困難なアプリケーションを再利用性への考慮なしに作っていては、開発工数の削減は一向に進まず、また各種品質属性の改善は見込めない。

品質マネジメントに欠かせない長期的視点

ISO9001（品質管理に関する国際標準モデル）やCMM（ソフトウェア開発の品質管理基準）などに代表される品質マネジメントは1990年代後半から大きな注目を集め、いまや企業の事業戦略の一部となり始めている。しかし、品質マネジメントの本質を理解しないと、リスクを極力排除する縮小均衡のソフトウェア開発になりかねない。高度な品質マネジメントを目指す企業にとって、リスク＆リターンの分析に基づいたアーキテクチャーの決定が重要であり、適切なリスクをとれる組織の体質が企業の活性化につながる。これからのシステム開発には、それに加えて本稿で述べたような長期的・戦略的視点が必要となるであろう。 ■