［第4回］

トレジャリー・マネージメント・ソリューションの導入

柳 洋二郎
サンガードジャパン
トレジャリー・ソリューションディレクター

これまで、数回にわたり、トレジャリー・マネージメント業務について述べてきたが、今回はシステムを導入するにあたっての進め方や注意点を弊社のプロジェクトでの体験も交えて述べてみたい。弊社の場合、パッケージをベースとしたシステム導入になるため、以下の内容は所謂スクラッチからの開発には適用できない部分も多々あり、可能な限りその辺りにも言及したい。

プロジェクトの骨組み

システム導入プロジェクトは、一般的に全体のプロセスを局面化して行う。各局面の呼び方はさまざまであるが、要件定義、基本設計、詳細設計、開発、テスト（単体テスト、統合テスト、システム・テスト）、移行という段階が最も一般的である。弊社のプロジェクトにおいてもこの観点は同じであるが、要件定義・基本設計局面をスコーピングと呼んでいる。全体の工数を一〇としたとき、スクラッチ開発では一般的に、要件定義・基本設計が三、詳細設計・開発で四、テストで三の分割をするが、弊社のプロジェクトでは、スコーピング四、詳細設計・開発（設定）が二、テストが四というのが標準的な分割である。このワークロードの分割割合がスクラッチ開発に近づくことは、パッケージにそれなりのカスタマイズ、アドオン開発を行ったことになり、パッケージ使用のメリットが小さくなる。

プロジェクト開始前（事前検討）に検討すべき事項

トレジャリーのプロジェクトは海外拠点を巻き込んで行われるケースが多いため、プロジェクトを立ち上げる前に各拠点と検討すべき項目が多々ある。各拠点の取引銀行、口座数や金融取引の種類、現状の業務の流れの調査に始まり、現地の規制や税金の仕組み等も当然考慮する必要がある。実際にプロジェクトが始まると、各拠点の要望を全て取り込むことは困難なため、事前に集めた情報と戦略的な観点を考慮して標準業務フローを決め、展開することになる。

スコーピング局面での進め方、考慮点

スコーピング局面では、事前検討で想定した業務フローをベースに実現性の確認と詳細の詰めを行うことになる。スクラッチ開発の場合、開発の難易度が非常に高い等の制約がない限り、要件を実現することが可能であるが、パッケージ導入の場合、カスタマイズ、アドオンなしに要件が実現可能なことの検証が最も重要になる。当然、パッケージで実現できない要件が出てきた場合、コストとの見合いでアドオン開発の可否を検討する必要が出てくる。一般的にある一つの業務要件を満たすための実現方法は複数あり、パッケージでは最も一般的な実現方法（ベスト・プラクティス）がサポートされているケースが多い。そのため、当初考えていた方法と異なる方法で実現を余儀なくされることは多々あるが、むしろパッケージが提供する手法を採用することで自社の業務をより効率化できるケースが多いと思われる。スコーピング局面で最も難しいのは、要件の実現方法を見ることができない中で要件を決めなければならないことである。この点、パッケージの場合、短期間でプロトタイプを構築（一～二週間）し、実現方法を検証することができるため、「こんなはずではなかった」というスクラッチ開発でよく見られる問題は発生しない。ただし、これはプロジェクトにエンドユーザーが直接参加して前面に出てきている場合である。

最後に、弊社のプロジェクトでのスコーピング局面の進め方を簡単に紹介する。弊社

では最初に標準化されたヒアリングの質問集を用いて大まかなユーザー要件の確認を行う。この結果をもとに、パッケージで必要な領域を特定し、その部分に関係する機能の研修を行い、パッケージへの理解を深めると共に要件をパッケージで実現するための方法についても詰めていく。この過程の中で、必要に応じコンサルタントが実際にパッケージを見せるため、ユーザーの理解も深まり、精度の高いＦＩＴ&ＧＡＰ分析がなされ、必要なアドオン項目が定義されることになる。

詳細設計・開発（設定）局面での進め方、考慮点

開発局面では、スコーピング局面で定義した要件に基づき実際にパッケージの設定を行う。通常、弊社にてプロトタイプを作成し、ユーザーが確認を行うというサイクルを複数回実施することで進め、この過程の中で最終的に設定が決まることになる。一方、必要となるアドオン機能に関しては、スクラッチ開発と同様の進め方をする。スコーピング局面でもれた要件は、このプロトタイピング中に拾うことになる。弊社の経験では、この段階で発生する要件の多くは帳票開発でカバーすることができるが、スコーピング時に十分に要件を詰めていないと、大きなアドオンの必要性が発生しない保証はない。

また、パッケージの設定は、パッケージ機能に関わる設定（承認の回数、リコンサイルの設定、資金繰り表の定義等）だけではなく、全体の組織構造の確定やカウンター・パーティーの構造、銀行や口座情報等プロトタイプを行う上で最低限必要となるマスター・データの登録を含むことにも注意を要する。

テスト、移行局面での進め方、考慮点

パッケージの設定が終了すると本番で使用するマスター・データ（銀行情報、口座情報、カウンター・パーティー情報等）を登録する。移行データの数が多い場合は、情報をインポートすることになる。以上の環境を準備した上で、実際に日々使用している取引の入力票を使いＵＡＴ（ユーザー受入テスト）を行う。ＵＡＴはユーザーが主役であり、弊社のコンサルタントは側にいて、問題が発生したり困った時に支援する役割になる。テスト内容としては、日々の業務のパターンを網羅するテストが必要である。システム部やベンダーが代わりにテストを行うケースもあるが、それを以ってユーザーの負荷が少なくなることはない。逆に、業務を十分理解していない人がテストをすることで、余分な工数をかけることになるスクラッチ開発の場合、ユーザーがシステムを触る前に十分な機能テストを行うことは常識であるが、パッケージ導入の場合、パッケージ自体の品質確認（これは本来パッケージ提供ベンダーが責任を持って行う作業）は既に終えているため、アドオン開発部分以外は、ユーザーが直接テストを行うことになる。もちろん、業務機能以外の機能（運用、障害対策等）もあるので、その部分の確認はベンダーが事前に十分に行う必要がある。

今日の経済環境は厳しく、今後もフリー・キャッシュ・フローの確保は企業にとり重要な課題となる。さまざまな戦略を実行する上で、キャッシュの「見える化」は必須の要件であり、そのためのツールであるトレジャリー・システムの導入は常に考慮すべき選択肢である。