講演2

JBCC株式会社　SIイノベーション事業部 SIイノベーション本部 本部長　川上 明治氏

自動生成ツールGeneXusの活用で実現するアジャイル開発の真価とは

かつては「大規模プロジェクトには向かない」「小規模開発向けの手法」と言われていたアジャイル開発だが、最近では国内においても大規模プロジェクトへの適用が増えつつある。かねてより規模を問わず様々な案件でアジャイル開発に取り組んできたJBCCの川上氏が、アジャイル開発のメリットや具体的な事例とともに、同社のアジャイル開発を支えるアプリケーション自動生成ツール「GeneXus」を紹介した。

早期確認による品質向上や段階的な開発が可能に

JBCCがアジャイル開発に取り組む最大の理由は「品質の向上」だという。従来のウォーターフォール型開発は、要件定義、外部設計、内部設計、プログラミング、テストと段階を経て開発していくため、ユーザが“実物”を初めて確認できるのが要件定義から1年以上経過後というケースも珍しくない。その段階で設計見直しが必要となった場合には本番稼働が大幅に遅延したり、修正・確認に十分な時間を確保できずに本番稼働後の大きなトラブル発生を招いたりすることにもなりかねない。

これに対して、アジャイル開発は開発対象を細かい単位（ある程度の機能のかたまり）に分割し、そのかたまりごとに設計・実装・確認といったサイクルをごく短期間で回していく。ユーザは早い段階から“動く実物”を何度も確認して判断できる。この開発初期からの繰り返しの確認により、大幅な品質向上が期待できるというわけだ（図1）。もちろん、ウォーターフォール型開発でしばしば問題になる、最終段階での「話していた（イメージしていた）ものと違う」といった事態も防ぐことができる。

なお、JBCCでは、大きく次の3つをアジャイル開発のメリットとして捉えている。

- 予算に合わせた段階的な開発が行える
- 設計の考慮漏れや認識違いが起こらない
- 進捗状況が明確になり、問題に対し先手を打てる

より早い収益化を実現する段階的構築で経営層のニーズに応える

アジャイル開発が多くの経営層のニーズに応える上で有効であることを示す事例として、国内最大手の予備校のシステム開発についても紹介した。このプロジェクトではまず、3つに分かれているシステムの、連携する機能をそれぞれ並行してアジャイル開発し、1年ごとに完成させて順次サービスインする「段階的構築案」を提案。投資効果がより早く得られることから経営層に評価され、受け入れられたが、現場の担当者はまったく逆の反応だった。スピードよりも「安全・安心」であることを重視し、時間がかかっても確実な全体再構築を希望したそうだ。最終的には、社内調整を経

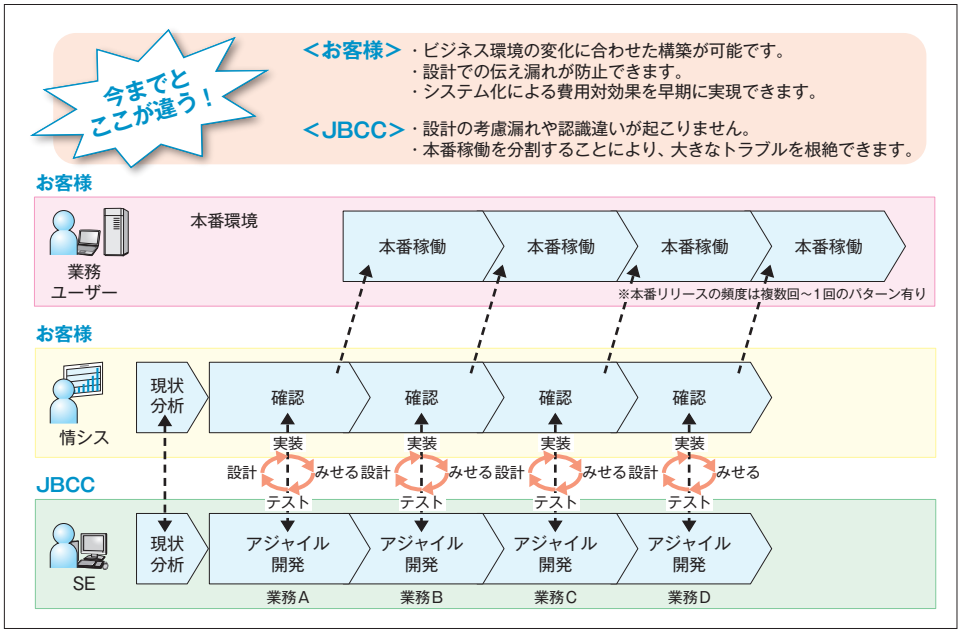

図1：アジャイル開発とは

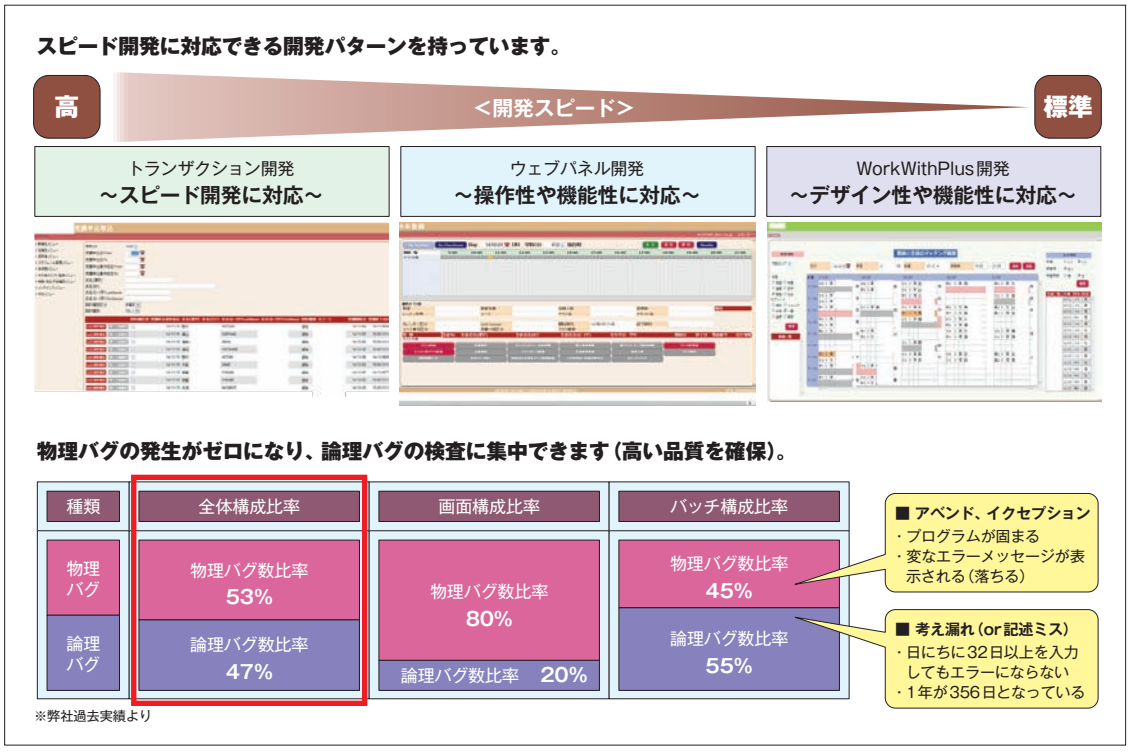

図2：
なぜスピード開発が可能なのか

て経営層の推す段階的構築案が採用された。川上氏は、収益向上が見込めるのであれば「経営層は間違いなくこちらを選択する」ことを強調した。

川上氏はほかにも、アジャイル開発のスピードの優位性を示すエピソードを紹介した。JBCCで通常のSI開発を手がけるチームがウォーターフォール型開発を前提とした見積りとスケジュールを顧客に提案したところ、大幅なスケジュールの短縮を打診され、社内でアジャイル開発チームに相談。アジャイル開発ならスケジュールを約6か月短縮できることを提案した結果、受注決定し、実際にアジャイル開発チーム担当のプロジェクトとしてスタートしているという。

アジャイル開発の要となるツール「GeneXus」

こうしたJBCCのアジャイル開発を支えているツールが「GeneXus」だ。GeneXusは、設計情報からプログラム（アプリケーション）を自動生成するツール。たとえば画面作成では手入力が20～50%、バッチ作成の手入力も70%程度までに軽減され、あとはGeneXusが自動生成する。「自動生成ツール」は世の中に多数存在するが、GeneXusの場合は人間が手を加える必要のない最終的な完成品を生成できるのが大きな特徴だ。手組み開発のようにデータベースとのやりとりを意識する必要がなく、SQLのコーディングもすべてGeneXusに任せられる。

スピード開発への対応としては、シンプルにスピードを追求した開発はもちろん、標準のスピードで手組みと同様にデザイン性や機能性も重視した開発、その中間でスピードと機能性を一定のレベルで両立する開発と、3種類の開発パターンを有している。

最大のメリットとも言えるのが、自動生成で人間がアドオンする余地がない結果、「物理バグが発生しない」ことだ。物理バグとは、アベンドやエクセプション（例外エラー）のことで、GeneXusでプログラムを作成する限り、基本的には一切発生しない。起こり得るバグは、「日にちに32日以上を入力してもエラーにならない」「1年が356日になっている」といった「論理バグ」のみとなる。過去のプロジェクトを対象としたJBCCの障害分析結果によれば、手組みでプログラムを作った場合の物理バグ含有率は50%強。それがGeneXusを使うことでゼロになれば、テストは論理バグの検査のみに集中できる（図2）。結果的に論理バグも減少し、より高い品質の確保につながるだろう。

川上氏は最後に、保守性の高さについても言及。GeneXusの機能で変更範囲の把握や影響分析が簡単にできるため、アプリケーション維持（追加・変更）に手間がかからないこと、ツールの活用方法を習得すればユーザ自身での開発・保守が可能になること、それにより常にシステムの“ありたい姿”を維持し続けられることなどを訴求して、講演を締めくくった。