保険の
2030年
アジェンダ
シリーズ 21

保険業界で急がれる脱メインフレーム ～リホストを起点とした段階移行を～

メインフレームの維持コストの上昇やサポート終了を背景に、保険業界で脱メインフレームの動きが加速している。メインフレームから脱却するためにはリホストを起点に技術を最新化する段階的な移行が有効である。

脱メインフレームを優先とした移行手法の選択

2022年2月に発表された富士通製メインフレームのサポート終了通告[1)]は、ユーザに期限付きで新システム環境への移行を求めるものであった。日立製メインフレームはサポート終了を発表していないものの、すでに製造は中止しており、今後のサポートの見通しは不透明である。また、多くのユーザを抱えるIBM製メインフレームについては維持コストの上昇やベンダーロックインというリスクを抱える。保険業界にとって、メインフレームからの速やかな脱却は喫緊の課題となっている。

これまでの大規模で複雑化する保険会社のシステム再構築は、計画段階から長期化するだけでなく、品質低下による中断や延伸を発生させ、何度もコスト追加を伴う再計画を余儀なくされてきた。過去の失敗を踏まえると、従来の移行手法であるリビルド（すべてのシステムを一から作り直すこと）は、初期投資額が大きくなるだけでなく、システム開発の長期化というリスクもあり、メーカーのサポート終了に間に合わなくなる可能性が極めて高い。

したがって、メインフレームからの速やかな脱却を優先するためにも、既存システムの機能を活用した移行手法を選択する必要がある。代表的な方法としてリホストとリライトがあり、どちらも大手銀行等の金融機関において多数の事例があるため、保険会社でどちらを選択すべきか悩むケースが多い。

リホストは、COBOL等のレガシー言語は変えず、メインフレーム上で動いていたソフトウェアをLinuxサーバー、Windowsサーバー、クラウド等のオープンなプラットフォーム上で動かす手法である。一方、リライトはソフトウェアのレガシー言語をJAVA等の新しい開発言語に機械的に変換するという違いがある。

リライトの課題

リライトのメリットは、システムのオープン化を達成することで最新技術の活用が容易になり、さらに技術者不足のリスクを低減させることができるという点にある。

しかし、リライトを急ぐと、その技術的な複雑性によりシステム開発の長期化と品質低下を招くリスクがある。これはそもそも設計思想が異なる言語をツールで機械的に変換することに限界があるためだ。

例えば、COBOL（手続き型）からJAVA（オブジェクト指向型）への移行は、元々の設計思想を反映した「手続き型のJAVA」が生成されるため、プログラムの可読性が低下し、それに応じた性能向上も必要となる。加えて、設計書がない開発現場では可読性が低下した新システムのコードを直接解読することとなり、効率的な保守が難しくなる。

保険会社のメインフレーム上には、長期にわたり利用されてきた生命保険の契約管理システムのように、過去の商品や計算ロジックが多く実装されたシステムがある。これらは今後の改修予定が少なく、必ずしもリライトによる大規模な言語変換を必要とするわけではない。

脱メインフレームはリホストで

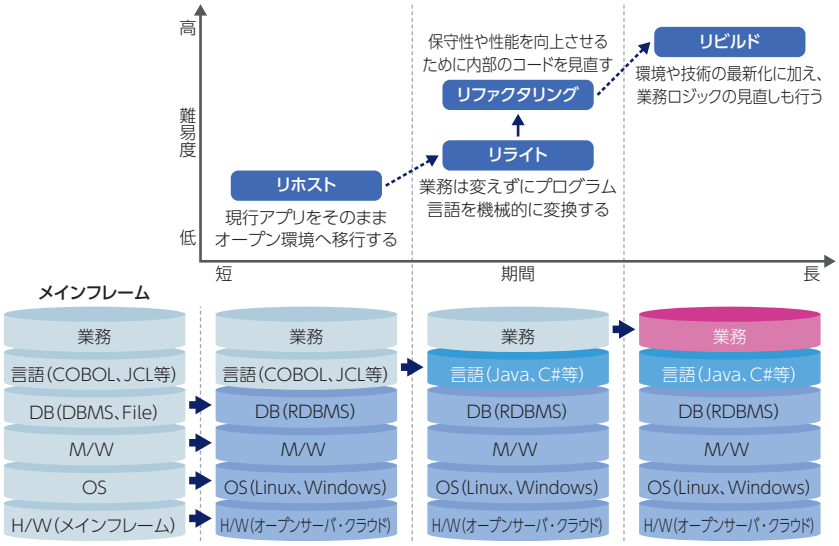
この点、リホストは移行期間を短縮できるだけでなく、安定したプログラムの再利用と現行システムの技術

野村総合研究所　金融デジタルビジネスリサーチ部

NOTE

1) 富士通は、2030年にメインフレームの製造・販売から撤退し、2035年には保守サポートを終了する。
2) 技術的負債（Technology Dept＝テクノロジーデプト）とは、ソフトウェア開発上の問題に対し、根本的な解決策ではなく、安易で不完全な「その場しのぎ」の解決策を選択したために、組織が後で支払うことを余儀なくされる潜在的コストを指す。

図表　リホストを起点とした段階移行



（出所）野村総合研究所

者を維持することができる。今後の改修予定が少ない多くの機能に対して有効な手法であり、移行前後の品質や保守性の劣化を抑止することができる。万が一、移行期間中に新たな要件が発生した場合にも、現行システムを改修することで効率的に対応することが可能である。

ただし、リホストにも地道な作業が発生する点は踏まえておく必要がある。ハードウェアに依存する言語が存在する場合は、オープン環境への移行時に人手による変換が伴い、リビルドが生じることがある。また、メーカー独自のミドルウェアがある場合、現行システムの有識者と協力しながら技術的な調査を進める必要がある。

段階移行により必要となる技術の最新化

リホストの課題はレガシー言語が残ることである。将来的にも技術者を確保することが難しくなるだろう。そのため、リホストはメインフレーム脱却を目的としたモダナイゼーションの第一段階と位置付け、リホスト完了後の新環境で技術の最新化に取り組むべきである。

リホストが完了したら更新頻度の高い機能のみを新環境上でリライトする。段階を踏むことで無駄な作業を抑止するだけでなく、性能等の新環境に起因した課題にも効率的に対処することができる。そして、技術的負債[2)]が蓄積しないよう、継続的なリファクタリングを行うことも重要である。このような段階的な移行を通じて、保守性が高く、低コストで運用できるオープンシステムを徐々に構築していくことが可能となる。

昨今、技術の最新化においては生成AIへの期待が高まっている。例えば、現在のリライトの課題に対応し、COBOLからオブジェクト指向型のJAVAを生成する技術や、生成したプログラムから設計書を自動作成する技術の研究も進んでいる。現在はまだ実用段階には達していないが、リホストが完了する頃にはその技術が十分に普及している可能性がある。最新技術の動向にも注目しながら、段階的な移行計画を策定する必要がある。

Writer's Profile
長江　裕介 Yusuke Nagae
保険インテグレーションデザイン部
エキスパート
専門は保険IT
focus@nri.co.jp