

# Low-Code Platformで変わる ソフトウェア開発の高速化

近年、LCP (Low-Code Platform : ローコードプラットフォーム) と呼ばれるソフトウェア開発手法が注目を集めています。本稿では、LCPが持つ機能の解説や、これまでのソフトウェア開発の歴史におけるLCPの立ち位置、従来の手法と比較してLCPが優れている点、LCPの注意事項、LCPの事例やLCPが実現する未来について解説します。

しまくら ゆうと  
島倉 優人

NTTデータ

## LCPとは

LCPとは、設計・製造・テスト・運用といったソフトウェアのライフサイクル全体をサポートするソフトウェア開発プラットフォームのことです。ソフトウェア開発と一言で表しても、その中で実施することは「開発環境を用意する」「本番運用環境を用意する」「設計書を基にソースコードを書く」「ソースコードを実行可能なソフトウェアに変換する」「ソフトウェアを開発・本番運用環境で実行する」「ソフトウェアを監視・分析する」、など多岐にわたります。LCPは図1に示すように大きく5種類の機能を提供しており、各工程を自動化して高い生産性を実現できます。

### ① 各種環境PaaS<sup>\*1</sup>提供

ソフトウェア実行のためには、典型的な例として次のハードウェアを用意する必要があります。

\*1 PaaS (Platform as a Service) : クラウド上のプラットフォームを利用できるサービスのこと。

- ・ソフトウェアを実行するためのサーバ
- ・データを蓄積するためのデータベース
- ・インターネットに公開するためのネットワーク

また、開発環境、テスト環境、本番環境など複数環境を用意することが一般的です。LCPでは利用開始と同時にソフトウェアを開発・実行できる環境が提供されます。さらに、モバイルなどのデジタル技術を容易に利用できる製品もあります。

### ② 視覚的なモデリング

LCPでは、GUI操作により、視覚

的に画面・データ・ロジックなどをモデリング(設計)します。ドラッグ&ドロップによるGUI操作を中心としており、特定プログラミング言語の知識に依存せずにソフトウェアをモデリング可能です。自動化技術によりコーディング量を極限まで減らすことで高生産性を実現しています。

### ③ マーケットプレイス

LCPではソフトウェアのテンプレートや、画面・ロジックのテンプレート、さらには、細かな部品などの共有・再利用を行うためのリポジトリが提供されることが多いです。これによりソフトウェアを1からつくる必要がなく、

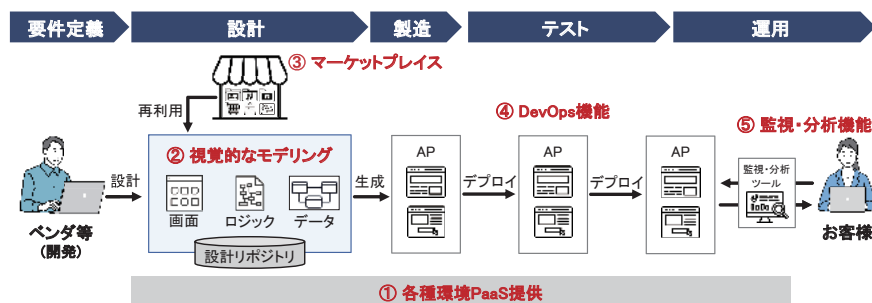


図1 LCPの機能と構成要素

「車輪の再発明」を防いで生産性向上を実現します。

④ DevOps<sup>\*2</sup>機能

モデリング情報からすぐに実行可能なソフトウェアを生成し、LCP内のソフトウェア実行環境に配置・実行する機能が提供されます。この機能により作成したソフトウェアをすぐにリリースできるようになり、リリースまでのリードタイムを短縮できます。

⑤ 監視・分析機能

①から④の機能でソフトウェアを実行したら、そのソフトウェアが期待どおりに動作しているかをモニタリングする必要があります。LCPではこの機能も提供され、利用状況をリアルタイムで確認できます。ソフトウェアへのアクセス状況、エラーの発生状況などに加えて、パフォーマンスやユーザエクスペリエンスを分析できる機能を持った製品もあります。

以上の説明のように、LCPはソフトウェアライフサイクル全体に対して機能を提供しており、これによりソフトウェア開発の高速化を実現できます。ただ、LCPが登場する前からソフトウェア開発高速化の取り組みはされてきました。以降では、従来の手法とLCPが異なる点について説明します。

ソフトウェア開発の歴史からみる LCP

プログラミングの歴史は1940年以前までさかのぼり、コンピュータが直接解釈できる機械言語が登場したことが始まりです。機械言語は二進数で表現されており、人間が理解し使いこなすことはとても難しいものです。その後、アセンブリ言語、C/Javaのようなプログラミング言語が登場しました。なかでもC/Javaのような高級言語は機

械言語と比較して自然言語に近いものになっており、機械言語の複雑さが隠蔽され、人間が理解しやすくなりました。人間が理解しやすくなったことで、生産性や保守性が向上しました。その後、さらに視覚的なモデリングによるビジュアル開発ができる機能を備えたLCPが登場し、より人間が直感的に理解しやすくなりました。図2に示すように、LCPはビジュアル言語であるともいえます。これにより、これまでの開発言語と比べてより高い生産性や保守性の向上を実現できます。

図3に示すように、プログラミング言語の進化と並行して、手でプログラミングすることから脱却する技術も登場しました。進化の流れは大きく2種類あり、「開発自動化」と「パッケージ（PKG）製品活用」です。

開発自動化は、CASEツール<sup>\*3</sup>による開発、MDA<sup>\*4</sup>によるモデル記述

- \*2 DevOps：DevelopmentとOperationsを組み合わせた造語。開発者と運用者が連携してソフトウェアが提供する価値をエンドユーザへ迅速に届ける手法のこと。
- \*3 CASE（Computer Aided Software Engineering）ツール：1980年代から1990年代にかけて多く利用された、ソフトウェア設計の可視化や設計データからソースコードを自動生成するソフトウェア開発支援ツールのこと。
- \*4 MDA（Model Driven Architecture）：業務のフローや機能などを抽象化したモデルを最初に作成して開発を進める手法のこと。このモデルからソースコードを生成する。

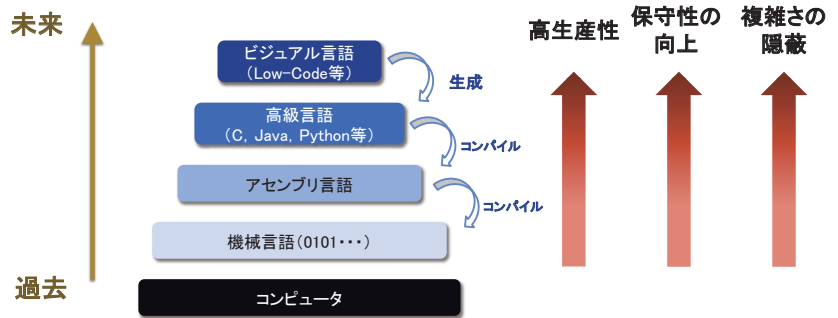


図2 プログラミング言語の進化

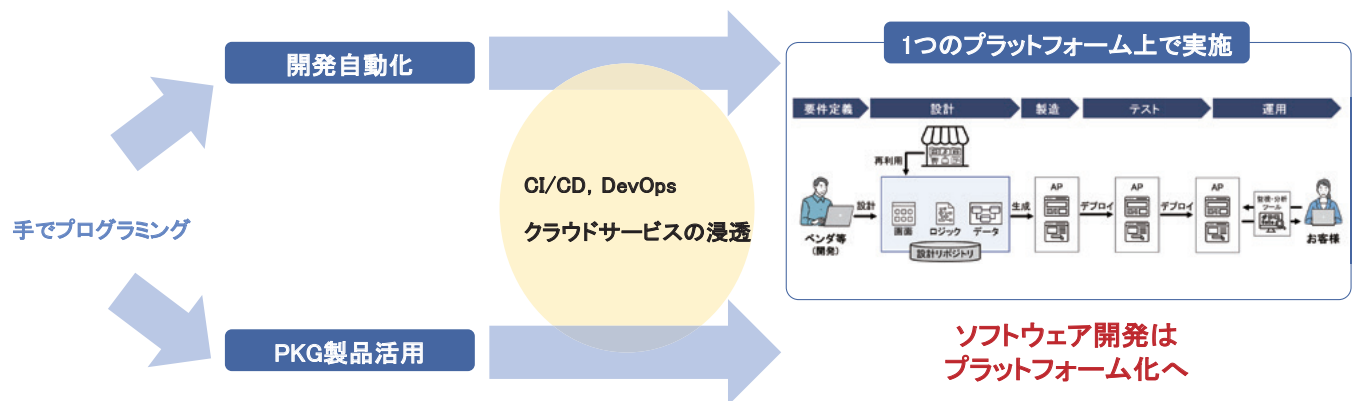


図3 ソフトウェア開発手法の生産性向上・高速化の変遷

などが挙げられます。PKG製品活用は、特定業務の効率化のためのPKG製品を活用するというものです。いずれもソフトウェア開発の「自動化」と「標準化・再利用」であるといえ、生産性向上・高速化は進化と発展を遂げてきました。

そして近年、CI/CD<sup>\*5</sup>、DevOps、クラウドサービスなどさまざまな技術が登場し、自動化の対象領域が拡大しました。いろいろな自動化技術を採用することで生産性向上を期待できますが、それぞれの技術を導入するためのコストがかかってしまいます。そこで、ソフトウェア開発をプラットフォーム化し、1つのプラットフォーム上で複数の自動化技術を利用しながら開発できるかたちに進化しました。LCPはこれに該当します。

プラットフォーム化したことによるメリットとして、図4に示すように、これまでのソフトウェア開発自動化手法よりも多くの工程に適用できるようになったことが挙げられます。ソフトウェア開発自動化手法はこれまで「ソースコード生成特化型」と「統合環境型」の2種類がありましたが、それぞれ課題がありました。ソースコード生成特化型は設計情報からソースコードを生成することに特化したものであり、自動生成されたソースコードからソフトウェアを実行する手法は別

途検討する必要がありました。また、統合環境型はソフトウェアの設計を開始する段階で利用でき、実行可能なソフトウェアを生成してくれる製品もありますが、ソフトウェアの実行環境は別に用意する必要があるなど適用工程は限定的でした。

LCPはプラットフォーム型として提供されることで、システムライフサイクル全般に対して自動化を適用できるようになり、これまでの手法よりも生産性の向上や開発の高速化を実現できるようになりました。

### LCPが実現する未来

近年はVUCA<sup>\*6</sup>時代であり、何が起るかの予測が難しい世の中においては変化に素早く対応する必要があります。そのため、ソフトウェア開発には高いアジリティが求められます。LCPは、ソフトウェア開発ライフサイクルの一連の流れの自動化を提供しますので、アジリティを高めることができます。結果、世の中の変化に追従できる超高速開発の実現に寄与します。

また、近年はDX（デジタルトランスフォーメーション）の推進を実現するために、ソフトウェア開発のニーズが爆発的に増加していて、IT人財のニーズも高まっています。このIT人財は高いIT技術力だけではなく、問題や課題を正確に把握し解決策をソフ

トウェア設計まで落とし込む力、ステークホルダーを巻き込んでソフトウェア開発を推進するプロジェクトマネジメント力など幅広い力が求められます。従来の開発では複数の技術を組み合わせる必要がありましたが、LCPでは採用したLCP製品のことだけを学べば最低限のソフトウェアを開発できます。そのため複数の技術や、それらを組み合わせる方法を学ぶことと比較すると、習熟難易度が下がります。近年のIT人財のニーズの高まりに伴い高度なスキルを持ったIT人財の確保は難しくなっていますが、習熟が比較的容易であるLCPを活用することで、既存人財のリスキングも実現できます。

習熟が容易であることから、従来ソフトウェア開発を外部に委託していた企業において、ソフトウェア開発を自社で行う、いわゆるソフトウェア開発内製化にも向いています。内製開発ではステークホルダー間のコミュニケーションや意思決定を自社内で完結できるため、開発のスピードアップに伴う

\*5 CI/CD (Continuous Integration / Continuous Delivery) : ソフトウェアのビルド(生成)やテスト、デプロイ(配置)などを自動化し、これらを継続的に実施する手法のこと。

\*6 VUCA : Volatility, Uncertainty, Complexity, Ambiguityの4単語の頭文字を並べたもの。不安定、不確実、複雑、あいまいで予想どおりにいかない社会情勢のこと。

進化															
分類	ソースコード生成特化型					統合環境型					プラットフォーム型 (LCP)				
適用工程	外部設計	内部設計	製造	テスト	維持運用	外部設計	内部設計	製造	テスト	維持運用	外部設計	内部設計	製造	テスト	維持運用
特徴	・画面、ロジック、データなどセグメントごとに機能が独立					・AP層より上位に特化し、システムを統一的に構築 ・抽象化モデルにより、GUIで操作が可能					・DevOps機能、基盤、運用等、システムライフサイクル全般をサポート				

図4 ソフトウェア開発自動化手法の進化

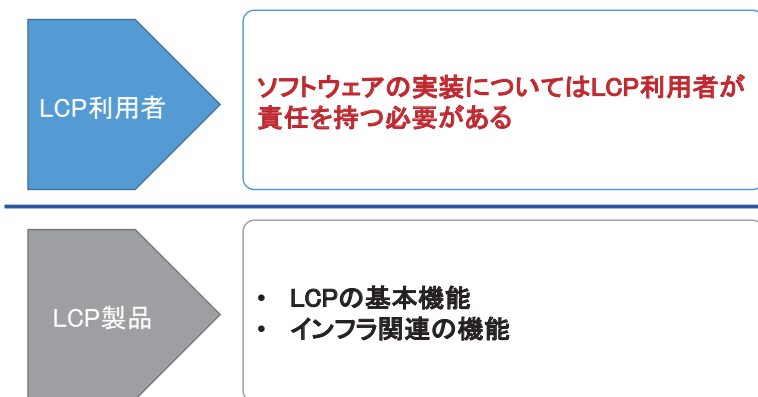


図5 LCPの責任共有モデルのイメージ

アジリティの向上が期待できます。

ここで、ソフトウェア開発内製化の具体的な事例を紹介します。NTT東日本において、内製開発を推進する経営的背景が増していました。そこで、NTTデータサポートの下、社員をLCP人財ヘリスキリングする取り組みを行いました。現在では、その人財が主体となって内製開発をできるようになっています。経営層による緊急情報把握が3日程かかっているという経営課題がありましたが、危機管理の仕組みの刷新と合わせて内製開発による迅速化、コスト最適化を達成しました。

## LCPの注意事項

前述の説明だけですとLCPは夢のようなツールに見えてきますが、使い方を誤るとさまざまな問題が発生する可能性があります。

① 求められる品質が高い場合は、品質保証ストーリーも高度になるLCPを用いることで、ソフトウェア開発のための技術のハードルが下がり、「どうつくるか」、ではなく、「何をつくるか」に注力できるため、ITベンダのようなプロの技術者だけではなく、ビジネス部門の担当者がソフトウェア開発に携わる、いわゆる市民開発者も利用すると考えられます。例えば、市民開発者が社内のミッションク

リティカルなソフトウェアの開発を実装することを考えます。ミッションクリティカルである場合は求められるソフトウェア品質のレベルも高くなり、高度な品質保証ストーリーが必要になります。これはソフトウェアプロセスの有識者が必要になり、市民開発者だけでは困難になります。

② シャドーITの増加やITガバナンスの低下

LCPでは簡単にソフトウェアをつくれることから、企業のIT部門が把握していないものが存在してしまう、いわゆるシャドーITの発生リスクが高まります。その問題を解決するためには、IT部門による統制が必要です。

③ LCP製品のFit&Gap

近年LCPの市場は急速に成長しており、200を超える多種多様なLCP製品・サービスが登場しています。各製品にはそれぞれ特徴があるため、目的や用途に応じて適切な製品を選定する必要があります。

④ LCP製品との責任共有モデルの理解

クラウドサービスでは責任共有モデルを理解したうえで利用することが重要ですが、LCPについても同様です。

図5に示すように、LCPでは、ソフトウェアの実装に関する問題は利用者責任となることが多いです。例えば、

本来はユーザ認証による権限設定をする必要があったが忘れてしまいセキュリティインシデントが発生した場合、ソフトウェアの実装方法が起因の問題となりますのでLCP利用者側の責任となります。利用する製品が担保する仕様について理解をし、それ以外のところは利用者自身で問題が発生しないかを確認する必要があります。

## 今後の展望

LCPの利用には利点がある反面、注意しなければならない点もあることをご理解いただけたかと思いますが、昨今の時代背景を踏まえると導入を推進すべきソリューションであると考えます。NTTデータでは、LCPの導入障壁を払拭する手法を確立し、先見の事業変革をお客さまとともに実現します。

### 参考文献

- (1) <https://www.nttdata.com/jp/ja/data-insight/2022/1018/>
- (2) <https://www.nttdata.com/jp/ja/data-insight/2019/0304/>
- (3) <https://www.nttdata.com/jp/ja/data-insight/2021/0818/>



島倉 優人

「これまで経験がないけど、リスキリングでソフトウェア開発に携わりたい」「今よりも高い生産性でソフトウェア開発をやりたい」と考えられている方、ぜひLCPを検討してください。NTTデータでは、さまざまなLCP製品を取り扱っています。連絡をお待ちしています。

### ◆問い合わせ先

NTTデータ  
技術革新統括本部 システム技術本部 ADM技術部  
E-mail adm\_contact@kits.nttdata.co.jp