LCP における開発プロセスの勘所について

Vital points of the development process in LCP

フューチャーアーキテクト株式会社　ビジネスコンサルティンググループ
Future Architect, Inc Business consulting Group
○長坂 昭彦
○Akihiko Nagasaka

Abstract: In recent years, many companies have adopted LCP (Low Code Platform) as a technology to automate and streamline white-collar work. In this trend, the demand for LCP development has been increasing year by year, however, some issues in LCP development have emerged.
For example
・LCP development has not established an estimation method and it may be underestimated.
・LCP development does not have established process definitions and quality standards, and system quality tends to vary.

To solve these problems, the Japan Function Point Users Group (JFPUG) started a study group (lcncSig) in FY2022 to study low-code/no-code development by its member volunteers. Based on actual data from of JFPUG member organizations and the knowledge of experts, we have studied estimation using the function point method, quality evaluation, standardization of development processes, etc., and have worked to solve the problems.

Among them, this presentation focuses on the standardization of the development process and presents tips documenting the LCP process definitions and findings from the field.

I hope this paper will help the spread and development of LCP/NCP in Japan.

1. はじめに

近年多くの企業がホワイトカラーの作業を自動化・効率化する技術として LCP（Low Code Platform）を採用している。そのような潮流の中、LCP 開発の需要は年々増加傾向だが一方で LCP 開発における課題も顕在化してきた。例えば
・LCP 開発は工数見積り手法が確立されておらず、見積精度にバラツキが出やすい
・LCP 開発は工程定義や品質基準が確立されておらず、システム品質にバラツキが出やすい
が挙げられる。
これらの課題を解決するため、日本ファンクションポイントユーザ会（以降、「JFPUG」）では 2022 年度からローコード/ノーコード開発における会員有志の研究会（以降、「lcncSig」）を開始した。JFPUG 会員組織の実績データや有識者の知見をもとに、ファンクションポイント法による見積り、品質評価、開発プロセスの標準化等の検討を行い、課題解決に取組んできた。その中でも今回は開発プロセスの標準化に焦点を当て、策定した LCP 工程定義や現場の知見を文書化した Tips を発表する。本発表が LCP/NCP の普及・発展の一助となれば幸いである。

フューチャーアーキテクト株式会社　ビジネスコンサルティンググループ
Future Architect, Inc Business Consulting Group
〒141-0032
東京都品川区大崎 1-2-2 アートビレッジ大崎セントラルタワー
Tel: 050-5305-8428
e-mail: a.nagasaka.5b@future.co.jp
1-2-2, Osaki Shinagawa-ku, Tokyo, 141-0032 Japan

【キーワード：】JFPUG、ファンクションポイント、lcncSig、LCP/NCP、FP 簡易推定法、開発生産性、開発プロセス、Tips、実績調査表

2. 開発における課題と背景

2.1 見積精度にバラツキあり

機能の複雑度や品質要件によって LCP 開発工数は数人日から数人月と大きく変動する。1 機能の開発に数か月を要する事もあるが、『1 機能 X 人日』など、一律見積を行うと過小見積りとなりスケジュール遅延を招く。要因として以下が挙げられる。

(1) 見積プロセスが未整備
通常のシステム開発とは異なり、LCP 開発は市民開発の名のもと EUC (End User Computing) の一環として実施される場合がある。その場合、要件確定後に工数・期間の確定見積り（概算見積りからの再見積り）を実施するプロセスとなっていない場合がある。

(2) 開発初期段階での FP 算出が困難
開発初期段階は要件自体が曖昧だが、予算や体制確保のためには一定精度で概算を見積る必要がある。従来の FP 算出方法ではファンクション抽出に基本設計レベルの情報が必要となり、開発初期段階での FP 算出が困難な場合がある。

(3) 信頼できる工数見積りの基準値欠如
開発初期段階で FP が算出できたとしても、工数=FP÷生産性のため、工数を算出するには生産性の基準値が必要だが、信頼できる基準値が存在しない。
一部の企業は自社の LCP 開発実績を蓄積しているが、各社のデータ収集基準や、開発プロセス、作成物等前提が異なる、または不明瞭なため、仮に他社情報を取得しても自社で利用しづらい。

2.2 システム品質にバラツキあり

LCP 開発では標準コンポーネントを利用し実装するため、通常のスクラッチ開発と比べて必要となるテストの量や質を過小に評価し品質劣化しやすい。結果、同等の品質要件のシステムであってもプロジェクトにより品質にバラツキが生じる。要因として以下が挙げられる。

(1) LCP に即した開発知見が不十分
通常の開発と異なり、LCP に即した開発上の留意点があるが、LCP 開発の経験が発注者/委託者双方に不足していると、やらなくても良いタスクを実施していたり、逆に怠ってはいけないタスクを怠っている場合がある。

(2) スクラッチ開発との差異が不明瞭
開発プロセスにウォーターフォールを採用した場合、通常のスクラッチ開発と LCP 開発に大工程レベルでは大きな差異はないが、各工程で実施するタスクや作成物、その作成方法は異なるため、差異が不明瞭なまま開発を進めると生産性や品質の低下を招く場合がある。

(3) LCP に即したテスト完了基準が不明瞭
コーディングしない事により、単体テスト工程及び結合テスト工程の目的や完了基準が曖昧となり、結果、不具合検知が遅れリリース遅延や本番障害に繋がる場合がある。

(4) 信頼できる品質基準値の欠如
従来の sloc ベースの品質基準（テスト密度や不具合密度）は利用できず、信頼できる基準値が存在しない。

3. 解決策

前述の課題と要因、及び検討した改善施策を列挙する。
今回は多くのプロジェクトにおいてその活用が見込める『LCP に即した開発知見の文書化』と『LCP に即した工程定義の明確化』について後述する。

図 1　LCP 開発における課題・要因・解決策

3.1　LCP に即した工程定義の明確化

全ての工程が LCP 開発固有ではなく、設計と製造・単体工程に特徴があるため、LCP 開発の実績を集計する事で、該当工程におけるスクラッチ開発との違いが明確になれば、LCP 開発で実施すべきこと、しなくて良いことが明確になるのでは、という仮説をもとに、開発実績を収集する調査表（以降、『LCP 実績調査表』）による調査や検討を実施した。

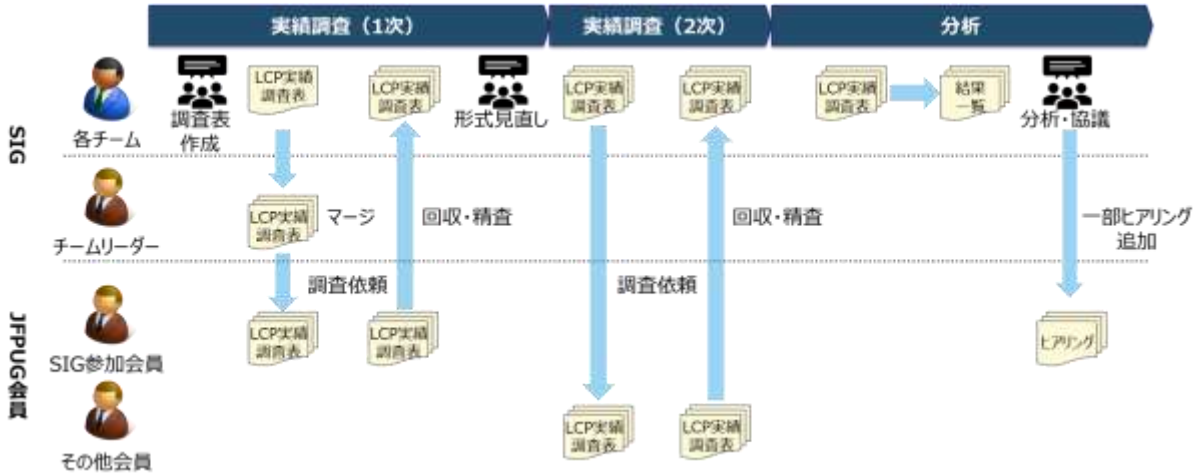

図 2　仮説検証フロー

LCP 実績調査表にて JFPUG 会員組織の LCP 開発プロジェクトの実績を 16 件集計し調査した。LCP 開発に一般的に必要と考えられるタスクと作成物を工程毎に仮説で定義したうえで、実態との乖離がないか検証した。また、LCP 製品機能の利用有無も合わせて調査した。なお、プロセス定義は原則開発 5 工程（基本設計～総合テスト（ベンダ確認））とした。プロセスを定義するにあたっては各社の方言に依存し認識齟齬が発生しないよう、工程やタスク定義はソフトウエア開発データ白書の「工程の呼称と SLCP マッピング」[1]を原則利用のうえ、作成物とその概要は今回新たに定義した。加えて、プロセス定義を補完する取組みとして単体・結合テストの明瞭化も実施し

た。

単体テストと結合テストの境界明瞭化

- ローコード開発においてはソフトウェアユニットの開発は原則不要となるため、単体テストと結合テストを次の通り位置づけて明瞭化

【単体テスト】画面機能の単体テスト及び結合の一部シナリオのプレテスト

【結合テスト】複数機能間におけるデータの登録／更新／削除／照会を伴うシナリオテスト

図 3　プロセス定義を補完する取組み

調査の結果、基本設計～製造・単体テスト工程に LCP 機能の活用が集中していることが判明した。

工程	LCP利用率
要件定義	10%
基本設計	13%
詳細設計	46%
製造・UT	25%
結合テスト	0%
総合テスト	0%
ユーザテスト	0%

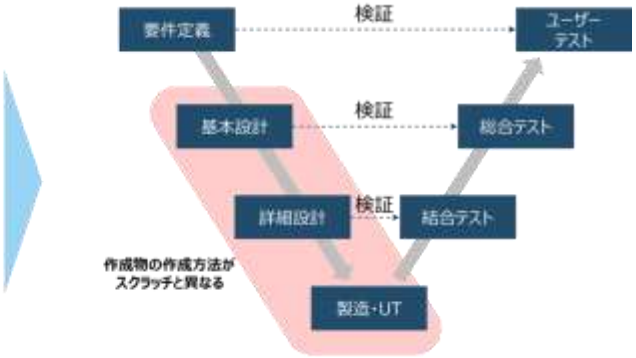

図 4　ウォーターフォールにおける LCP 開発の特徴

基本設計～製造・単体工程の作成物の有無は通常のスクラッチ開発と大差はないが、作成方法は大きく異なる。LCP 開発では基本設計や詳細設計の結果から実行プログラムが自動生成されるため、LCP 機能を使って設計する事が生産性の観点では重要であることをプロセス観点で確認した。

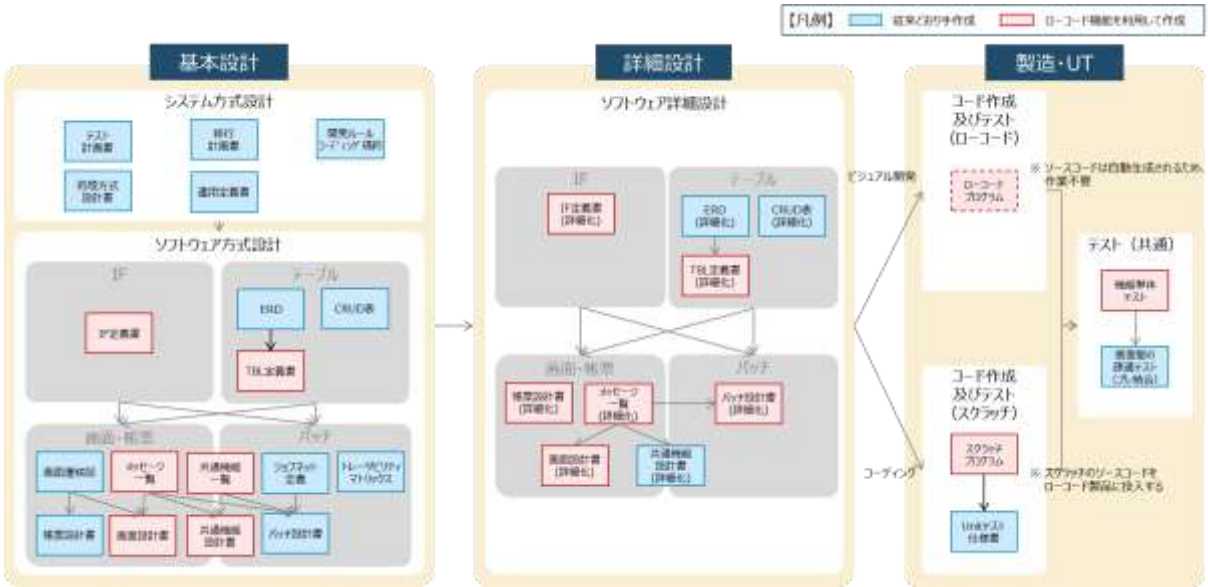

図 5　ローコード開発作成物フロー（基本設計～製造・単体）

表 1　LCP 開発プロセス一覧（一部抜粋）

大工程	プロセス	作成物	作成物概要
基本設計	システム方式設計	処理方式設計書	アプリ/モジュール構成や 2 段階認証方式、バージョン管理、リリース管理の方式等
		テスト計画書	結合テスト、システムテストのスケジュール、環境、体制、実施内容等
		移行計画書	業務移行、システム移行、データ移行のスケジュール、環境、体制、実施内容等
		開発ルール・コーディング規約	LCP 上での開発ルールやローコード/スクラッチのコーディング規約等
		運用定義書	パスワード誤り時の問合せ対応やシステムメンテナンス時の運用仕様等を定めたもの
	ソフトウェア方式設計	画面設計書	レイアウト、項目定義、表示順、バリデーション、更新対象データ種等記載されたもの
		帳票設計書	レイアウト、項目定義、表示順、バリデーション、更新対象データ種等記載されたもの
		バッチ設計書	レイアウト、項目定義、表示順、バリデーション、更新対象データ種等記載されたもの
		ジョブネット定義	バッチ処理の実行サイクルや実行順、待ち受け条件等定義したもの
		ERD	テーブルの関連と多重度、主要なデータ項目を論理レベルで表現したもの
		CRUD 表	各機能とデータの直行表、データのライフサイクル（CRUD）がわかるもの
		TBL 定義書	各テーブルの項目別の属性を論理レベルで表現したもの
		IF 定義書	各 IF の項目別の属性を論理レベルで表現したもの
		共通機能一覧	共通的に利用される画面やログ出力等の機能が一覧化されたもの
		共通機能設計書	一覧に記載された各機能の用途や機能が記載されたもの
		メッセージ一覧	各画面で表示される各種メッセージを一覧化したもの
		画面遷移図	画面間の繋がりが網羅的に記載され画面遷移が把握できるもの
		トレーサビリティマトリックス	デグレ防止を目的とした LCP の部品と画面の直行表

3.2 LCP に即した開発知見の文書化

LCP 開発プロセスを検討する中で LCP 開発経験者から過去に苦労した点を失敗事例も含め拝聴し、多くの教訓を得ることが出来た。その中でも開発 5 工程を対象として教訓を取捨選択し、得られた教訓に「実施する効果」と「実施しない影響」を加え『ローコード開発 Tips 集』として文書化した。なお、特定の LCP 製品に該当するものは除外し、どの LCP 製品を利用する場合も考慮しなければいけない Tips とした。各自のプロジェクト事情に応じて「実施する効果」と「実施しない影響」を鑑みて、取捨選択のうえ活用頂きたい。

ローコード開発 Tips 集

(1) 作成するドキュメントは意識的に減らす

＜実施する効果＞

LCP では設計情報がビジュアル表示されるため、「LCP 上で確認できる設計情報は、LCP 上で確認する」を基本姿勢とする。それにより、設計書作成の一部を LCP 機能で代替することができ、開発スピードを向上することができる。

＜実施しない影響＞

従来の開発と同じように設計書を作成した場合、LCP が単なる「ソースコード自動生成ツール」になってしまい、設計情報の 2 重メンテにより想定以上のコストがかかる。

(2) LCP 製品の仕様制約を把握のうえ、制約に抵触しないよう設計する

＜実施する効果＞

LCP 特性をふまえた設計となり、非機能要件の実現性が高まり、ユーザの期待値コントロールにも繋がる。

＜実施しない影響＞

非機能要件を保証できない、または改修時の修正コストが増加する。

(3) マーケットプレイスはサポートレベルを確認し利用方針を定める

＜実施する効果＞

マーケットプレイスを有効活用することで、品質を維持して生産性を高めることができる。

＜実施しない影響＞

品質劣後のコンポーネントを利用した場合、後で作り直しが発生する可能性がある。サポートレベルが不明瞭なコンポーネントを利用した場合、突然サポート切れになる可能性がある。

(4) 各コンポーネントの利用・変更有無は業務機能と紐づけ、影響範囲を明確にしておく

＜実施する効果＞

トレーサビリティマトリックス（LCP コンポーネントと業務機能の直行表）を作成するなどして、どのコンポーネントをどの業務機能で利用しているか、網羅的に把握する事により仕様変更時の影響調査やテスト範囲の見極めが容易となる。（保守効率化にも有効）

＜実施しない影響＞

仕様変更や不具合改修の際にデグレードが発生する。

(5) 開発ルールや規約を定義し、それらに準じた実装テンプレを用意する

＜実施する効果＞

LCP 開発は動くモノを素早く実装できる半面、類似した振る舞いをもつ機能でも、開発担当者ごとに異なる方法で実装する可能性がある。このような個体差を防ぐことで、その後の保守性が向上する。

＜実施しない影響＞

類似仕様を改修する際に実装方法がバラバラとなりコスト増となる。また、保守性に対する開発担当者の意識低下を招き、他と実装方法が異なる“一点物”の機能が増加し保守コストが増加する。

(6) 画面、帳票、IF 等のインターフェースは型桁最大最小等の属性レベルまで設計する

＜実施する効果＞

機能要件に従って詳細レベルまで仕様を明確にする事でレビューの実効性が増す。
＜実施しない影響＞
仕様を明確にしないまま開発してしまい、テスト工程にて不具合として顕在化する。

(7) 処理構造やロジックを第三者がレビューする
＜実施する効果＞
実装品質や保守性を保つことができる。また、LCP 開発はデザインパターンがまだ確立されていない中で、良い実装・避けるべき実装をチームで素早く共有することができる。
＜実施しない影響＞
実装を見ないとケース網羅が確認できないテストが漏れることにより不具合が潜在する。また、不具合検出後は LCP 標準機能で実現できない場合は拡張コードを作成することになるため、コスト増に繋がる。

なお、開発 5 工程から外れるため Tips 集には記載していないが、要件定義では Fit&Gap を実施し、業務をシンプルにして Gap 抑制のうえ、基本設計では LCP 製品の仕様制約に抵触しないようアプリケーションを設計すること（Tips(2)）が、プロジェクトの成功率を高めることに繋がる。重要なため併せて述べておく。

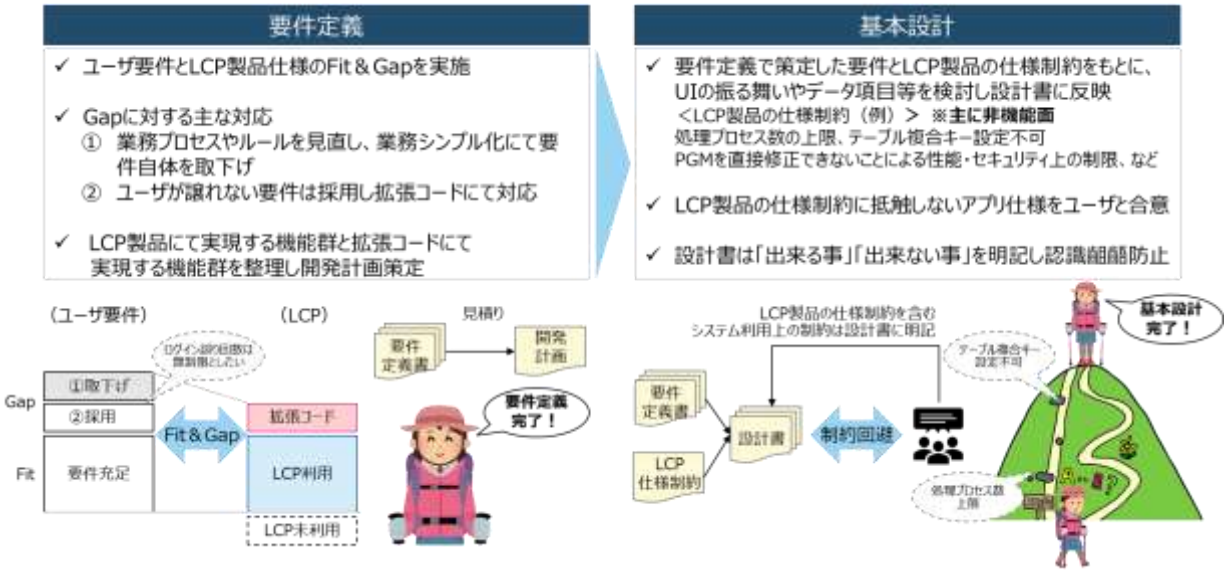

図 6「Fit&Gap」と「LCP 製品の仕様制約回避」の実施イメージ

4. 活動の振返り

前述の活動を振返り、自己評価と得られた気付きを述べる。

4.1 LCP に即した工程定義の明確化

LCP 開発の在るべき開発プロセスを明確にする事ができ一定の成果を出す事が出来た。また、LCP 開発は導入しやすい反面、データモデリング、構造設計、アルゴリズム設計等は必要（≠設計レス）である。一見敷居の低い LCP 開発だが、設計やテストのスキルは必要という心構えが得られた。

4.2 LCP に即した開発知見の文書化

教訓を形式知として文書化した事により、これからローコードに取り組む層が直面する課題に対して示唆を提供する事ができた。また、LCP 製品から何らかの仕様制約を受けるため、制約を事前に把握しておく事が重要であり、仕様制約に抵触しないためには、基本設計を担当する設計担当者も LCP の製品仕様を正しく理解しておくことが重要と再認識できた。

5. 今後に向けての取組み

2023 年度も lcncSig は活動を継続しており、本年度は LCP に加えて NCP も対象に広げつつ、見積・品質の基準値や、LCP/NCP 製品選定の観点、アジャイル適用時の考慮点などを整備していく予定である。

参考文献

[1] 独立行政法人情報処理推進機構（IPA）技術本部　ソフトウエア高信頼化センター（SEC）、「SEC BOOKS ソフトウエア開発データ白書 2018-2019」P333
A.1　工程の呼称と SLCP マッピング
https://www.ipa.go.jp/publish/wp-sd/ps6vr70000011kwd-att/000069381.pdf