ソフトウェア工学 73−13
（1990. 7. 20）

プログラムチェックリスト（ＰＣＬ）の改良効果

大野　治　　降旗由香理

ファコム・ハイタック（株）　ハイタック本部

ＥＡＧＬＥ／Ｐには、標準パターンに応じたＰＣＬがあり、それを標準ＰＣＬと呼んでいる。本稿では、その改良効果によるプログラムの品質と生産性の向上について述べる。

ＰＣＬの改良は、単体および結合デバッグ工程時におけるバグの分析結果に基づき、行った。さらに、プロジェクトにおいて試行し、「バグの摘出し易さ」、「ＰＣＬの作成効率」の観点から評価した。これより、結合デバッグ工程時におけるバグの約60％を占めていた単体デバッグの残存バグが減少し、さらに、ＰＣＬの作成効率が向上したという結果が得られた。

Improvement of Program Check List and its Evaluation

Osamu Ohno　　Yukari Furuhata

FACOM-HITAC LIMITED　HITAC DIVISION

Aoba NO.1 Bldg. 3-1, Kudanminami 2-chome, Chiyoda-ku, Tokyo 102, Japan

This Paper describes the effect of applying EAGLE/P standard PCL(Program Check List). In unit debugging we are using EAGLE/P standard PCL which we prepared corresponding to standard program pattern.

Now we have just arranged these standard PCL in order to improve software reliability and productivity.

Before arranging we reserched on program bugs which were found during unit debugging and combination debugging and found out the fact that sixty percent of bugs which were found in combination bugs could have been found in unit debugging.

The arranged standard PCL made it possible to decrease the number of bugs remaining after unit debugging and to make out PCL efficiently.

１．はじめに

結合テスト(以後「ＣＤ」と呼ぶ)工程時において摘出されるバグの約60％は単体テスト(以後「ＵＤ」と呼ぶ)工程時における残存バグである。これはプログラムの機能確認を行うプログラムチェックリスト(以後「ＰＣＬ」と呼ぶ)が不充分であるために、ＵＤ工程時においてバグを充分に摘出できないことによる。

現在、ＥＡＧＬＥ／Ｐ＊には標準パターンに応じて標準ＰＣＬが設定されている。これを改善することによって、プログラムの品質および生産性の向上を図ることを目的として、バグの分析結果に基づき標準ＰＣＬの再作成を行なった。さらに、プロジェクトにおいて試行し、これを評価した。本稿では、その概要について報告する。

２．バグの分析

ＰＣＬの品質は、ＵＤ工程時において、バグを確実に摘出できるかどうかで問われる。

バグを確実に摘出するためには、バグを分析し、ＰＣＬの充実という観点から充分な解析を行うことが必要である。それは、バグを分析することによって標準ＰＣＬのチェック項目の有効性が評価でき、さらに、不足しているチェック項目の抽出ができるからである。

このため、標準ＰＣＬの改善に先立ち、バグの分析を行うことにした。まず、ＵＤ工程時のバグを分析し、続いてＣＤ工程時のバグを分析して、ＵＤ工程時のバグとの関連性を調べた。

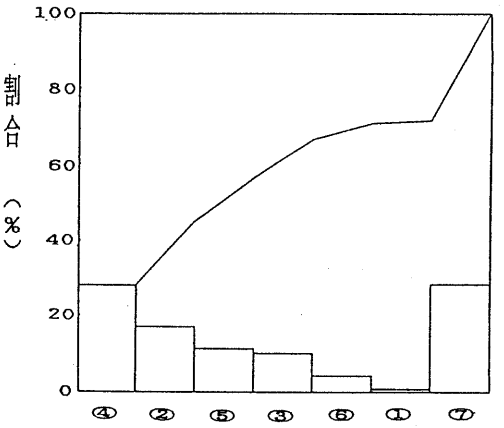
２．１　ＵＤ工程時におけるバグの分析

バグは、発生の重点となる原因を把握するために、発生要因による分類を行った。分類に当っては、プログラムの構造に着目し、入力ファイルのＲＥＡＤ、レコードのＷＲＩＴＥ、その他の各編集処理の３つに分けた。そして、さらにその内容を分類した。また、構造からはとらえにくい、制御、作業領域部分を考慮し、追加した。こうして、表１に示すような７つの要因に分類した。

表１　バグの本質的要因の分類表

項番	分類の観点	バグの発生要因
1	入力ファイルのREAD	入力処理
2	編集処理	ループ・判定処理
3		サブルーチン使用の処理
4	レコードのWRITE	出力処理
5	作業領域	エリア
6	制御関係	フラグ・キー等
7	その他	その他

尚、「その他」には、どの要因にもあてはまらないものの他に、バグの内容を把握できなかったものも含んでいる。要因別のバグ密度をまとめ、パレート分析を行った結果を図１に示す。



発生要因は表１の項番に対応

図１　バグ発生件数のパレート分析

分析結果から分かることは、次の３点である。

第一には、プログラムの構造に関らず、「出力処理」に関するバグが最も多いこと

＊ EAGLE/P : Effective Approach to achieving hiGh LEvel software productivity/Prototype

である。特に、帳票出力のプログラムにおいて「改頁時における集計行の出力」によるバグが極めて多いことは、注目に値する。

第二には、分析したプログラムの全ての構造を通じて、上位４つのバグ発生要因が同じであることである。この４つの要因とは、「出力処理」、「ループ、判定処理」、「エリア」、「サブルーチン使用の処理」である。そして、この４つの要因によるバグが全体の67％を占めている。

第三に、「サブルーチン使用の処理」については、サブルーチン使用率を考慮し、その使用がバグの発生にどの程度影響を与えているかを、明確にする必要がある。今回の分析では、サブルーチン使用の処理によるバグが全体に占める割合は、10％という結果が出ているが、その使用率は、57％であった。したがって、サブルーチン使用のプログラムのみを対象とした場合には、この割合はさらに大きくなると言える。このため、サブルーチンの使用がバグの発生に与える影響は軽視できないと考える。

以上より、バグの発生に影響を及ぼす要因は、主に以下の４つの項目に絞ることができる。

① 出力処理
② ループ・判定処理
③ サブルーチン使用の処理
④ エリアに関する部分

２．２　ＣＤ工程時におけるバグの分析

１節では、ＵＤ工程時におけるバグを分析し、標準ＰＣＬの評価を行った。しかし、過去のバグ分析では、「ＣＤ工程時に摘出されるバグの約60％は、ＰＣＬが不充分であることによるＵＤ工程時の残存バグである」という結果が得られている。そこで、本節では、ＵＤ残存バグの確認を行い、ＵＤ工程時のバグとの傾向を検証することを目的として、１節で分析したプログラムのＣＤ工程時におけるバグの分析を行う。

分析は、ＵＤ工程時のバグ分析と同様に、発生要因による分類を行う。分類にあたり、バグを、本来ＵＤで見つけうるバグ（以後「ＵＤ残存バグ」と呼ぶ）とＣＤで初めて見つけうるバグ（以後「ＣＤバグ」と呼ぶ）に分けた。そして、各々について原因を分類し、その本質的要因を考えた。ＵＤ残存バグについては、ＵＤ工程時のバグと同じ方法で分類した。そして、ＣＤバグについては、プログラム間のインターフェースと仕様書に着目して、分類した。こうして、表２に示すような９つの要因に分類した。

表２　バグの本質的要因の分類表

項番	バグ区分	分類の観点	バグの発生要因
1	ＵＤ残存バグ	入力ファイルのREAD	入力処理
2		編集処理	ループ・判定処理
3			サブルーチン使用の処理
4		レコードのWRITE	出力処理
5		作業領域	エリア
6		制御関係	フラグ・キー等
7	ＣＤバグ	インターフェース	インターフェース誤り
8		仕様書	仕様書の不良
9	その他	その他	その他

尚、「その他」には、どの要因にもあてはまらないものの他に、バグの内容を把握できなかったものも含んでおり、バグ区分が明確にできなかった。

要因別にバグ密度をまとめ、パレート分析を行った結果を図２に示す。

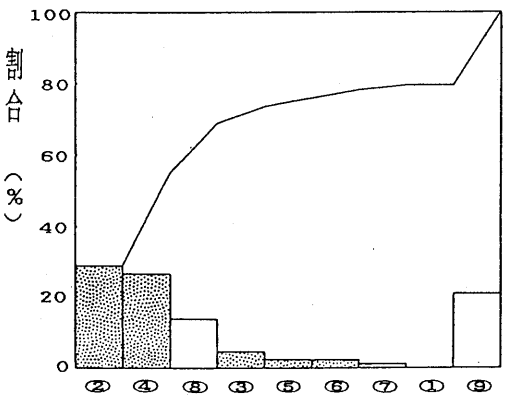


図２　バグ発生要因のパレート分析

分析結果から以下の考察を行った。

まず、第一には、ＵＤ残存バグが極めて多く、全体の64％を占めていることである。これは、過去のバグ分析結果（約60％）と一致するものであり、ＣＤ工程時に摘出されるバグにおいてＵＤ残存バグがいかに多いかを物語っている。

第二には、ＵＤ残存バグの中でも、特に「ループ・判定処理」、「出力処理」によるバグが全体の56％を占めていることである。これは注目に値する。なぜなら、ＵＤ工程時におけるバグにおいても、この２つの要因によるバグが全体の45％を占めているという結果が得られているからである。すなわち、ＵＤ工程時のバグとＵＤ残存バグでは、その発生要因に同じ傾向が見られると言える。したがって、ＵＤおよびＣＤ工程時のバグでは、その発生要因に同じ傾向があると考えられる。

以上より、ＣＤ工程時のバグを分析し、以下の結論を得た。

① ＵＤ残存バグがＣＤ工程時に摘出されたバグに占める割合は大きい。
② ＵＤおよびＣＤ工程時のバグには、その発生要因に同じ傾向がある。
③ バグの発生要因として大きな割合を占めるものは、「ループ・判定処理」、「出力処理」である。

３．標準ＰＣＬの再作成

バグの分析結果から、ＣＤ工程時においてＵＤ残存バグが多いことが明らかになった。そこで、ＵＤ残存バグを減少させることを目的として、ＰＣＬの改善を試み、再作成した。

３．１　構成

再作成した標準ＰＣＬには次の２種類がある。

（１）パターン別ＰＣＬ
ＥＡＧＬＥの標準パターンに応じたパターン固有のＰＣＬであり、ＰＣＬの骨組みとなる。

（２）処理別ＰＣＬ
部品として位置付けられるＰＣＬであり、プログラム仕様に応じ、必要な処理に対して使用する。
今回は、①ループ・判定処理用および②サブルーチン使用処理用の２種類を作成した。

３．２　改善点

再作成にあたり改善した点は次の６点である。

① 処理別チェックリストの作成
② セクション別チェックリストの作成
③ 確認内容の明確化
④ 二者択一方式採用
⑤ 入力レコードの少ないチェック項目から順番に配置
⑥ 不要なチェック項目を削除した。

上述の６点について補足説明をする。バグの摘出が多い要因について、①では、「ループ・判定処理」と「サブルーチン使用処理」の、②では「出力処理」のチェックを充実し、ＵＤ工程時においてバグを確実に摘出することを目的とした。

さらに、②ではセクション毎にチェック項目を設定したチェックリストによって、処理のタイミングの表現を試みた。

③では、編集処理のチェックを確実に行うこと、確認する機能を理解して入力データを作成することによって、作成の誤り、もれの減少を図った。

④は、集計レコードの出力位置に「固定時」と「浮動時」があるように一意に処理を決められない場合の対策である。これは、考えられる複数の処理を用意し、ユーザが選択し、使用できることによって、ユーザの追加記入部分の減少を図るものである。

⑤では、入力レコードの少ないデータからテストを行うことにより、バグ原因の究

明を容易にし、テストの効率化を図った。

⑥では、バグを摘出できないと推定できるチェック項目を不要と考え、削除した。削除したチェック項目の例をあげると、帳票のヘッダに関する項目などである。これについては、定義した帳票レイアウトを本番イメージで確認できるテスト印刷機能が、ＥＡＧＬＥ／Ｐに追加されたためと考える。尚、チェック項目の有効性は、バグの発生要因とＰＣＬのチェック項目を照合することによって判断した。

４．再作成した標準ＰＣＬの試行

再作成した標準ＰＣＬをプロジェクトにおいて試行した。そして、その効果を調べるために、従来の標準ＰＣＬを用いたプロジェクト（今後「プロジェクトＡ」と呼ぶ）と再作成した標準ＰＣＬを用いたプロジェクト（今後「プロジェクトＢ」と呼ぶ）のプログラム開発の結果を比較することにした。詳細を表３に示す。

表３　試行の環境

項番	項　目	従来のＰＣＬ プロジェクトＡ	再作成ＰＣＬ プロジェクトＢ
1	開発期間　(日)	１５	１４
2	開発人員　(人)	４４	４３
3	開発規模　(ks)	２７．１	３１．８
4	仕様書の作成	同一グループ	
5	作成者のレベル	同一グループ	

表より、この２つのプロジェクトの環境は、ＰＣＬの比較を行うには条件が整っている。したがって、これより得られた結果はデータとして信頼性が高いと考えている。

５．再作成した標準ＰＣＬの評価

本章では、再作成した標準ＰＣＬの評価を行う。

評価は、プロジェクトＡとプロジェクトＢのプログラム開発におけるバグ分析結果、作成工数などを比較することによって行う。尚、評価の観点は「バグを確実に摘出すること」、「効率よくＰＣＬを作成すること」の２点とする。

５．１　バグの分析による検討

本節では「バグを確実に摘出すること」という観点よりＵＤ残存バグの変化に着目した評価を行う。

ＵＤおよびＣＤ工程時におけるバグの分析結果を表４に示す。尚、数値はプロジェクトＡを100とした相対数値である。

表４　バグの分析結果

項番	項　目		従来のＰＣＬ プロジェクトＡ	再作成ＰＣＬ プロジェクトＢ
1	ＣＤ	ＣＣＬ密度	１００	９２
		バグ密度	１００	７６
2	ＵＤ	ＰＣＬ密度	１００	１４３
		バグ密度	１００	１３０
3	合計バグ密度		１００	１０６

バグの分析結果よりＵＤとＣＤ工程時の合計バグ密度は変わらないが、プロジェクトＢでは、ＣＤ工程時で摘出するバグが減少することがわかった。

そこで、この原因を調べるために、ＣＤ工程時のバグを発生要因により、「ＵＤ残存バグ」「ＣＤバグ」「その他」のバグの３つに分類し、その構成を調べた。結果は図３となる。尚、数値はプロジェクトＡにおけるＣＤ工程時のバグ密度を100とした相対数値である。

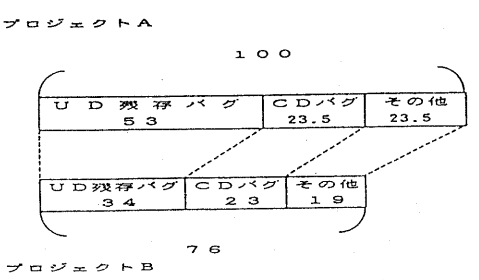


図３　ＣＤ工程時のバグ密度

この結果から、プロジェクトBでは、プロジェクトAと較べると、「CDバグ」および「その他」のバグには、ほとんど変化がないが、UD残存バグは大きく減少していることがわかる。このため、プロジェクトBにおいて、CD工程時のバグ密度が減少するのは、UD残存バグの減少によると言える。したがって、再作成した標準PCLには、UD残存バグの減少という効果があることが明らかになった。

5．2　作成工数による検討

本節では「効率よくPCLを作成する」という観点より作成工数に着目した評価を行う。

そこで、プログラムおよびPCL作成時間を調べ、プロジェクトAとBを比較した。その結果を表5にまとめる。尚、作成時間とPCL密度の数値は、プロジェクトAを100とした総対数値である。

表5　作成工数の比較

項番	項目		従来のPCL プロジェクトA	再作成PCL プロジェクトB
1	規模（ks）	全体	10.5	9.0
2		プログラム当たり	0.50	0.45
3	プログラム数　（本）		21	20
4	PCL密度　（件/ks）		100	143
5	作成時間	プログラム	100	94
6		PCL	100	67

表より、プロジェクトBでは、プロジェクトAに較べると、

①PCL密度の増加

②プログラム作成時間の増加

③PCL作成時間の減少

が、明らかである。これは、再作成した標準PCLの使用時は、従来のそれの使用時に較べて、より少ない時間で、より充実したPCLを作成したことを表すものであり、PCL作成効率の向上を意味する。

そこで、この原因を明らかにする1手段として標準PCLの使用状況を調べた。方法としては、従来の標準PCLと再作成した標準PCLにおいて、標準PCLへ追加チェックした項目数と標準PCLから削除したチェック項目数の比較を行うことにした。結果を表6にまとめる。

尚、数値はプロジェクトAにおけるPCL（全チェック項目）密度を100とした相対数値である。

表6　標準PCLの使用状況の比較

項番	項目		従来のPCL プロジェクトA	再作成PCL プロジェクトB
1	規模　（ks）		10.5	9.0
2	プログラム数　（本）		21	20
3	プログラム作成者　（人）		16	15
4	PCL	全チェック項目	100	143
5		追加したチェック項目	22	8
6		削除したチェック項目	7	15
7	密度	標準PCLで提供するチェック項目	85	151

結果より考察を行う。

第一に、追加チェック項目数が著しく減少したことである。現在、標準PCLは不足しているチェック項目をユーザが追加記入することによって使用されている。このため、チェック項目の設定もれ、誤りなどの問題が生じることは避けられない。ところが、追加チェック項目数が減少するということは、このような問題を防止することになる。

尚、追加チェック項目数の減少は、

①チェック項目の充実化

⇒ 処理，標準パターンに応じたチェックリスト作成

②ユーザ追加記入部分の減少

⇒ ユーザによる二者択一方式を採用

によると考えている。

第二に、削除したチェック項目数が増加したことである。これは、標準ＰＣＬのチェック項目数が増加し、使用しないチェック項目数が増加したためと考えている。しかし、チェック項目を削除することは、追加するよりも容易であり、誤りの生じることも少ないとみており、特に、ＰＣＬのマイナス要因とは考えられない。

したがって、再作成した標準ＰＣＬの使用では、ＰＣＬの作成時間の減少により、ＰＣＬの作成効率の向上が確認できた。そして、それは、ユーザによる追加記入が少なくなったことが原因と思われる。

６．おわりに

本稿では、標準ＰＣＬの再作成を行い、プロジェクトにおいて試行した。その結果、再作成した標準ＰＣＬでは「ＵＤ残存バグの減少」と「標準ＰＣＬ作成の効率向上」が認められ、評価を得た。しかし、標準パターンに応じたＰＣＬに追加、削除を行い、ＰＣＬを作成する現行の方法では、作成効率の向上に限界がある。さらにプログラム固有の処理に対してチェック項目をサポートしていないため、チェック項目の設定についても誤りやもれという問題が生じることは避けられない。そこで、今後の課題として、個々のプログラム仕様に応じたＰＣＬの自動生成を考えている。

参考文献

（１）石井康雄「ソフトウェアの製造」
日科技連出版社（１９８６）

（２）津田，葉木　他：「システム開発支援ＥＡＧＬＥの開発」，情報処理学会第３０回全国大会，４Ｓ－１，
ＰＰ．６１９－６２０，１９８５