ソフトウェア工学　28－4
（１９８３．２．９）

ソフトウェアの再利用における阻害要因の分析

西村　高志，　廣道　博史
（　ＩＰＡ　技術センター　）

１．　はじめに

計算機化される業務種類の増加と，個々の業務処理内容の複雑化により，企業がソフトウェアの開発と保守に投資する費用の総量は，急激に増加している。企業にとって重要なソフトウェア問題は，この投資額の総量を，企業の負担できる範囲内に抑えることにある。

この課題を解決するためには，

①　要求分析，設計，作成，検査，保守の各作業フェーズについての技術開発による，個々のソフトウェア・システムの開発・保守費の軽減化。

②　ソフトウェアの再利用の促進化による，すでに，ひとつのシステム開発に投資した費用の効率的運用。

この２つの連結が，最も効果的な方法と考える。

①については，ソフトウェア工学分野のこれまでの技術開発により進展している。

ここでは，②の障害となっている，ソフトウェアの再利用を阻害する要因について，分析した結果を報告する。

２．　対象分野，対象段階

ここで扱う対象分野－－企業業種①，プログラミング言語②，応用業務③－－と，ソフトウェアのライフサイクル上での対象段階④について説明する。

①と②については，「コンピューター白書　１９８１」［１］の「産業別電子計算機実働状況」と「使用言語」のデータより，次のものとする。

①　対象業種は，計算機利用企業。

②　対象プログラミング言語は，COBOL。

③と④については，計算機利用企業を対象とした「ソフトウェア調査」で収集した，各企業が重要視しているソフトウェア・システム４６４件の，「適用業務分布」と「再開発間隔分布」のデータ［２］より，次のものとする。

③　対象業務は，定常的事務処理業務一般－－経理，購買など。

④　ライフサイクル上での対象段階は，「初期開発」および「再開発」。

「再開発間隔分布」のデータ［２］によると，再開発の間隔は，中央値２年半，１～４年に集中した分布をなしている。この点から，保守費用に占める再開発の費用が，大きいと判断した。

3．　ソフトウェアを再利用する場面の分析

ソフトウェアの再利用を，提供者と再利用者間での，ソフトウェアのやりとり，とみなせる。この節では，ソフトウェアの再利用場面を構成する，次の3点について分析する。

－　再利用者の関心事。

－　機能分割による設計方法。

－　プログラミング言語COBOLが提供している機能。

（1）再利用者の関心事

再利用の際に必要な項目を明らかにするために，再利用者の関心事を調べる。

プログラムの再利用は次の3段階からなる。

第1段階：　必要な機能を満たすプログラムを検索する。

第2段階：　選択したプログラムを使えるように変更する。

第3段階：　分担しているソフトウェアに組込む。

各段階での再利用者の関心事を整理する。

①　検索については，

－　どこを探したらいいのか。――蓄積されているプログラムの所在――

－　必要な機能を満たすプログラムが見つかるのか。――検索方法――

－　使えるか，あるいは組み込めるかは，どのようにして知るのか。その調査に要する費用や時間がどれほどか。これらを知るのに，プログラムの中身を見るのでは，たまらない。

－　選択されたプログラムの信頼性は保障されているのか。

②　変更については，

－　そのまま変更なしで，組み込めないのか。

－　変更するとしたら，どこをどう直せばよいか。その調査に要する費用，時間はどのくらいか。

－　直すのに必要な費用，時間はどのくらいか。

－　変更後の検査はどのようにすればよいのか。

③　組込みについては，

－　組込みに要する費用，時間はどのくらいか。

－　組込んだプログラムの検査はどうするか。

－　組込んだ後の保守はだいじょうぶか。

再利用者は，分担しているシステムでの機能を満たし，信頼性の保障されたプログラムを要求する。ソフトウェアはハードウエアと比べて，複雑度が高く，柔軟である。そして，利用者の要求する機能が特殊化すればするほど，選択できるプログラムは少なくなり，必要な変更の量も多くなる。開発しているシステムの設計に，一般的なワク組みがないとしたら，再利用者の要求を満足するプログラムの発見は困難である。

（２）機能分割による設計方法

ソフトウェアの一般的な設計方法である機能分割を，再利用の観点から考察する。

機能分割は，次の２段階からなる。

第１段階：　ソフトウェア全体としての機能から始まって，プリミティブな機能にいたるまで，機能を階層的に分割する。

第２段階：　第１段階でつくられた機能の階層構造に従って，プログラムを割り当てる。

こうして出来上がったプログラムの構造は，次のようになる。

－　各プログラムは機能階層上での，下位プログラムの機能を合わせて，一つの機能を実現する。

－　各プログラムは，機能階層上での，上位プログラムの機能の一部を分担し，兄弟に当たる他のプログラムを考慮して，機能が決められる。

ソフトウェア開発者の最大関心事は，要求される機能を満たすシステムを実現することに置かれている。仮に，要求される機能を，個人的基準で分割するとすれば，多数の分割の組み合わせが生じうる。その結果，分割後のプログラムの機能は，そのシステムに固有となり，再利用は困難となる。

１９６０年代のモジュラ・プログラミングの時代が到来した時，人々は，何百もの再利用可能なブロック・プログラムを抽象化し，それをプログラム・ライブラリーに追加することによって，最終的に他人の作業の上に自分のプログラムを構築できるようになるだろう，と期待した。しかし，現実には，蓄積はされたが，プログラムが固有すぎていたために，利用はされなかった。［３］

（３）プログラム言語COBOLが提供している機能

提供者，利用者間でやりとりされるソフトウェアの，主要な表現手段である，プログラミング言語について分析する。

COBOLが，プログラム単位を利用するために提供している機能は，次の３つである。

① ｐｅｒｆｏｒｍ文
一般形式：　ＰＥＲＦＯＲＭ　手続き名－１　ＴＨＲＯＵＧＨ　手続き名－２
同じプログラム中の手続きを呼び出す。一つのプログラムを手続きに分割して，扱いやすいようにまとめる機能である。呼び出される手続きを同じプログラム中に限定しているために，再利用には，利用しずらい。

② ｃｏｐｙ文
一般形式：　ＣＯＰＹ　原文名　ＩＮ　登録集名　ＲＥＰＬＡＣＥ　－－－　ＴＯ　－－－
登録集中のプログラム・テキストを，作成中のプログラムに埋め込む。埋め込む際に変数名等の置換ができる。開発されるソフトウェア内での，共通的な宣言や手続きを，ソフトウェア内の，プログラムにまたがって共有する機能である。

③ ｃａｌｌ文
一般形式：　ＣＡＬＬ　プログラム名　ＵＳＩＮＧ　パラメーター列
プログラムを，パラメーター列の実引数により呼び出す。ソフトウェア内での，プログラムを結合する機能である。

COBOLの提供する機能のうち，再利用に使えそうな，copy文とcall文の２つについて，再利用に使う際の欠点と，その欠点より引き起こされる，再利用上の問題点をまとめる。（表－１）

表－１　copy文とcall文の再利用から見た問題点

	（再利用に使う際の欠点）	（再利用上の問題点）
copy文	埋め込むプログラム・テキスト（原文）の変更許容性が少ない。	名前等を除いて，要求に，ほぼ完全に一致するプログラムでないと，再利用できない。
call文	呼び出すプログラム名と，インターフェイスとなるパラメーターの属性が，プログラム中で明示的に宣言される。この結果，プログラム間の結びつきが強くなる。	あるプログラムを再利用しようとすると，そのプログラム中でcopy文，call文で参照されている，他のプログラムが引きずられてくる。その結果，不必要なものが含まれ，そのために，機能的にも不適切となる。

COBOLの提供する機能では，プログラム間の結合が緊密すぎるために，再利用には不適当である。

以上の分析から，次の３点が，プログラム再利用の重要項目と考える。

① プログラムを分割する際の，共通の基準。

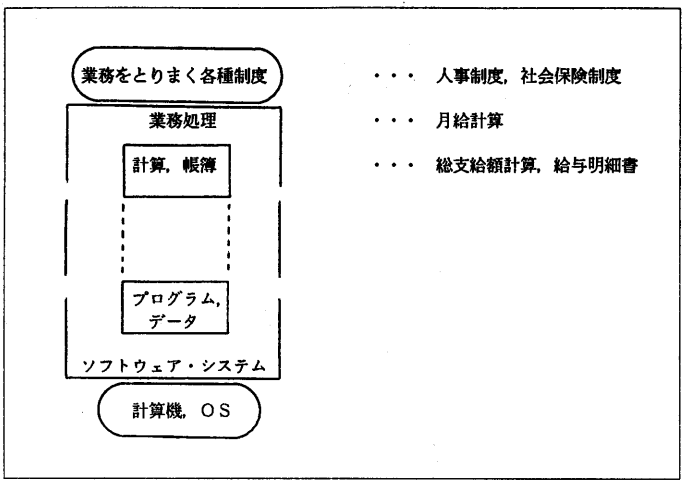
② 自由度の大きい，プログラム間の結合手段。

③ 必要なプログラムを選択できる，検索手段。

４．　事務処理業務ソフトウェアの特徴

前節で述べた，再利用の重要項目に関連する，事務処理業務ソフトウェアの特徴を整理する。

事務処理業務ソフトウェアを，業務処理上の実用知識を計算機で扱えるように表現したもの，と見ることができる。［４］　この見方では，「月給計算」という業務は，人事制度，社会保険制度などの，月給計算業務をとりまく各種制度のワク組みのなかで，総支給額計算，給与明細書作成などを行う処理である。そして，この業務をプログラムやデータにより，計算機，オペレーティング・システム上で組み立てたものとして，「月給計算システム」を，とらえる。（図－１）



図－1　ソフトウェアの位置づけ

そして，業務処理もソフトウェアも，ともに業務をとりまく制度からの強い束縛を受けている。上記の模式図上で，上部での変化－－業務をとりまく制度の変更－－につれて，下部であるソフトウェアに要求される機能が変わる。下部についても，データ・ベース化，オンライン化につれて，ソフトウェアでの実現形態が変わる。業務処理上での計算，帳簿などの単位と，実現しているプログラム，データ間の対応のズレが，ソフトウェア問題の主要な原因と，私は考えている。

再利用の観点からみると，ソフトウェア初期開発の際には，計算機化される現在の業務処理単位に一致するプログラムは，利用できる。そして，再開発の際には，ソフトウェアが業務処理の単位に従って分割されていれば，現在のシステムの再利用率は高まる。さらに，定常的事務処理業務については，業務処理の分析も，標準化も進んでいる。以上より，業務処理上の単位を，プログラムを分割する基準に適用できると，私は考えている。

5　プログラム再利用の促進について

再利用についての残りの重要項目－－結合手段，検索手段－－に関連して，UNIX--trade mark of Bell Lab.－－のツール・キットを取り上げる。私は，これをテキスト加工用ソフトウェア分野での再利用の成功例と考える。

UNIXのパイプライン機構は，小さなツール・キットの結合により，まとまった機能を持つソフトウェアの組み立てを可能としている。この結果，標準化されたツール・キット群の蓄積が促進される。［5］

UNIXにおいて，再利用可能なプログラムである，ツール・キットの蓄積を促進しているキーポイントは，次の４点と考えている。

① テキスト・ファイルの加工という，共通のプログラム分割単位。

② 順編成テキスト・ファイルという，共通のプログラム間の入出力ファイル。

③ パイプラインという，プログラム連結の機構。

④ テキスト処理に限定することによる，ツール・キット選択の容易性。

UNIXは，テキスト・ファイルという物理レベルの提供により，再利用を促進している。一般の事務処理業務については，処理の意味を含めたレベルとして，次の項目を提供することで，再利用が促進されると考える。

① 帳表作成という，プログラム分割単位。

② データ・フローによる，プログラム結合。

③ 各業務内に限定することにより，業務の固有な言葉による検索。

6． おわりに

ソフトウェアの再利用について，再利用者，提供者，プログラミング言語の３つの面から，問題点を分析した。この分析から，再利用を阻害する主要な要因を，提供者，利用者間での，共通のプログラム分割単位の欠如と，とらえた。そして，一般の事務処理業務ソフトウェアの分析から，業務処理に従ってプログラムを分割する基準を提供出来れば，ソフトウェアの再利用を進めることができると考えている。

謝辞

技術センターの川合所長には，論文の内容全般について意見をいただいた。再利用者の関心事については，日本電気㈱寺本　雅則氏の指摘を参考としている。

参考文献

[1] コンピューター白書１９８１，日本情報処理開発協会編.

[2] 西村： 計算機利用企業が重要視するソフトウェアにみられる再開発工数分布，情報処理学会第２５回全国大会，１９８２，６E－４.

[3] Bergland, G. D. : Structured Design Methodologies, 15th Annual Design Automation Conf., 1978, pp. 475～493.

[4] 西村： 表現された知識として見たソフトウェア，第２４回プログラミング・シンポジウム報告集，１９８３年1月，pp. 39～47.

[5] Kernighan, B. W., Maskey, J. R. : The Unix Programming Environment, COMPUTER, Vol. 14, No. 4 (April, 1981), pp. 12～24.