

ソフトウェア開発における部品化の実践と評価

福田由紀雄 土肥しのぶ 蓮田広保
株式会社 東芝 システム・ソフトウェア技術研究所

ソフトウェア生産工業化システム IMAP (Integrated software Management and Production support system)の中で、部品化・再利用は重要な役割をしめており IMAPでは品質の保証された部品を組合わせてアプリケーションプログラムを作成するということを工業化の重要なコンセプトの一つであると考えている。我々は、この基本思想に基づいて IMAPのサブシステムであるソフトウェア開発管理システム (DMS/SPCS: integrated Database Management for Software products/Software Production Control System)のソフトウェアを開発した。本システムは新規開発であり開発手順は次のようにした。(1)各アプリケーションの機能から部品となる機能の抽出 (2)各アプリケーション間で部品候補として選ばれた機能の共通化と部品の仕様化 (3)部品を使った各アプリケーションの設計 (4)部品の先行作成と検査 (5)アプリケーションの組合わせ試験

その結果、部品化の対象となった各機能に対して部品利用率が50~90%となり機能面および品質面については良い結果が得られた。しかし部品の利用のしやすさについては、特に経験の浅い技術者の観点からみると十分でない点(例えば、実例の提示、資料の自己完結性など)が指摘され、実践を通じて部品についての種々の指針を得ることができた。

THE PRACTICE AND THE EVALUATION ABOUT SOFTWARE PARTS IN SOFTWARE DEVELOPMENT

Yukio FUKUDA, Shinobu DOHI and Hiroyasu HASUDA

Systems and software Engineering Laboratory, TOSHIBA CORPORATION
70 Yanagi-Cho Saiwai-Ku Kawasaki, Kanagawa, 210 Japan

IMAP aims to industrialize the software development process. (IMAP stands for Integrated software Management and Production support system). It is one of the major concepts in IMAP to make application programs by assembling the software parts that are guaranteed the quality.

We are developing the software development management system named DMS/SPCS that are subsystems of IMAP. (DMS/SPCS stands for integrated Database Management for software products/ Software Production Control System) using this software parts technique.

We took the sequence as follows. (1) to extract similar functions from decomposed functions of application systems (2) to unify functions from similar functions and to specify the software parts (3) to design application programs based on software parts (4) to implement and test software parts before application design (5) to test application programs after the end of parts test.

Expected good results were produced about functions and quality of parts. The rate of using parts is 50~90%. But unexpected results were produced about the ease to use parts especially for beginner of programming. So we got some guides for software parts.

まえがき

我々は、ソフトウェア生産工業化システム I MAP (Integrated software Management and Production support system) を開発してきた。

I MAPでのソフトウェア工業化とは生産過程を工程に分割して各工程内の作業基準を規定し、かつ品質基準を満たした製品を、指定期間内に一定工数で生産できるようにすることである。この工業化を実現するために、

- (1) 管理手法の確立
- (2) 作業手順の明確化と分業化
- (3) 部品化・再利用の促進
- (4) 生産環境の確立
- (5) 技術者育成のための教育体系の確立

を重視している。

特に部品化・再利用については、ハードウェアと同様にソフトウェアにおいても品質の保証された標準部品の存在が工業化の前提となる。

1. I MAPでの部品の考え方

同じ分野の類似ソフトウェアの生産において、既存ソフトウェアの流用あるいは、繰り返し利用されている部分を“切り出し”、再利用することは現在でも行われている。特に同一組織による類似ソフトウェア作成の場合、60%以上再利用可能[1]なシステムもある。

しかし、切り出された部分を他の分野あるいは他の組織のソフトウェア生産に再利用しようとするとき、切り出された部分の理解や修正に手間どり、新規に生産する工数よりも工数が多かったりトラブル時に対応できない例も発生している。

ソフトウェア生産工業化における標準部品とは、既存のソフトウェアからの“切り出し”ではなく、再利用することを前提とした部品を標準部品規定に基づいて製品開発に先だって作成し、品質を保証し、正式な手続を踏んでオーソライズされた部品のことである。この標準部品規定には、部品の仕様記述法、インタフェース仕様記述法、品質基準と検査方法などは当然として標準部分をオーソライズする正式機関の位置づけ、権限、構成メンバの規定も含まれる。[2]

いいかえると、アプリケーションプログラムを設計してから仕様に合った部品をさがすのではなく、標準部品に合わせた設計手法を確立することが重要である。品質の保証された標準部品を使用することは生産性向上だけでなく、ソフトウェアの品質向上にとっても不可欠である。

I MAPでの開発業務の組織関連図を図1に示す。

2. 本試行の目的

一般的に対象システムが新規の場合、最初の開発では部品はほとんど整備されておらず開発が済んだあとで共通機能を部品として整備するのが普通である。

我々は、新規開発のシステムに対して部品の整備と製品の開発とを同時進行させ、製品のアプリケーションプログラムを設計する前に部品の仕様を整備し、部品を使ってアプリケーションを設計・作成し、検査済の部品を使ってアプリケーションを組立・検査することによって、新規システムの開発の時から部品を利用したソフトウェア開発を実践することをねらいとした。

3. 開発体制

開発チームを部品担当と製品担当に明確に分離した。

部品担当の仕事は次の通りである。

- (1) アプリケーション機能から共通機能の抽出
- (2) 部品の仕様化および部品の製造・検査

製品担当の仕事は次の通りである。

- (1) 部品を使ったアプリケーションの設計・製造
- (2) アプリケーションの組立・検査

開発要員の投入に関しては今回のシステム開発では部品の機能・品質がアプリケーションの機能・品質を左右するため経験者を部品担当に優先して割当てた。

* 部品担当……経験者 (3~5年)	3人
新人 (1~2年)	2人
* 製品担当……経験者 (3~5年)	1人
新人 (1~2年)	5人

4. 対象システムの概要

今回の試行対象は I MAP のサブシステムであるソフトウェア開発管理システム[3]とした。このシステムの機能は次の通りである。

(1) データベース管理機能

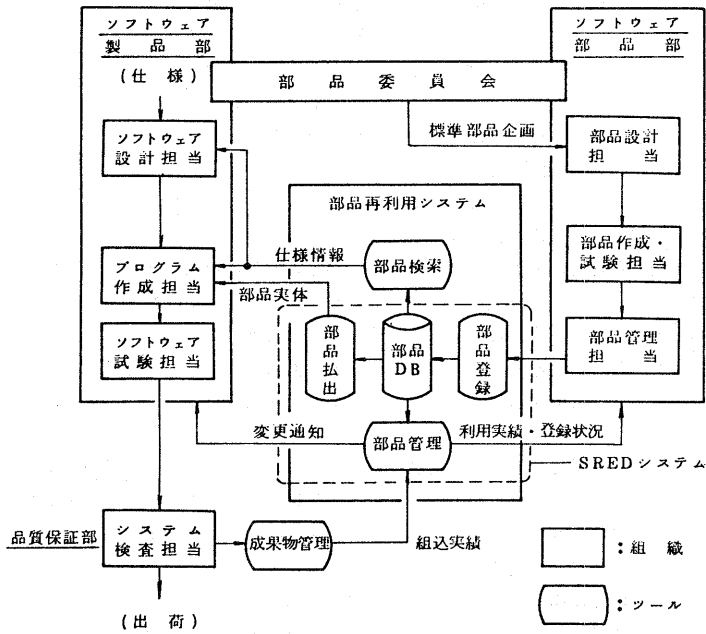
ソフトウェア構成表をベースとした成果物の一貫データベースと開発管理と成果物管理とを密に連携する統合データベースであり、データベースはローカルホスト上に構築されネットワークでワークステーションと接続されている。

(2) ワークステーション機能

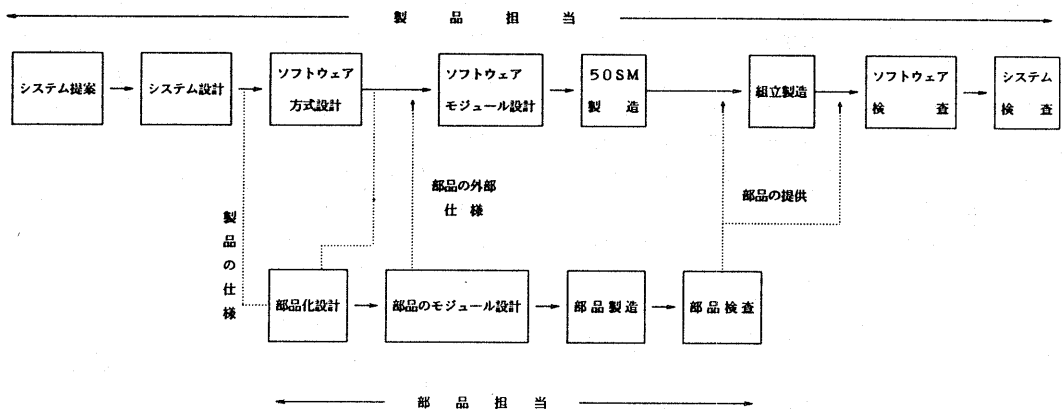
作業管理ツール (作業指示、作業進捗、作業承認など) や成果物管理ツール (構成表の登録・更新、成果物の保管・払出など) をワークステーション上で対話形式で使用する。

このシステムを試行対象として選んだ理由は次の通りである。

ソフトウェア開発業務組織関連図 (図1)



本試行での開発手順 図2



(1) 新規開発のシステムであるため。

(2) 多くのツール群から構成されているので、あるツールを対象にして実施した部品を残りのツールに適用出来るため。

5. 開発手順

開発手順を図2に示す。主な手順は次の通りである。

(1) 主なアプリケーションの機能を分解して部品の候補と思われる機能（共通性の高いもの）を抽出した。抽出された機能はデータベースアクセス機能、画面表示・画面制御機能、データ処理機能などであった。（製品担当・部品担当共同で経験者中心）

(2) 抽出された部品候補の機能を共通化した。そして他のアプリケーション機能を分解して共通化した機能がどの程度有効かを評価し見直した。（製品担当・部品担当共同で経験者中心）

(3) 共通化した機能を部品として仕様化した。（部品担当で経験者中心）

(4) 部品の利用を前提として各アプリケーションプログラムを設計した。（製品担当で経験者・初心者）

(5) 部品の設計（部品担当で経験者中心）

(6) 部品の製造（部品担当で経験者・初心者）

(7) 部品の検査（部品担当で検査仕様は経験者中心）

(8) 製品固有のモジュールの製造（製品担当で経験者・初心者）

(9) 製品の組立・検査（製品担当で部品は検査されたもののみ対象）

（注）(4)～(8)は並行して作業を進めた。

6. 部品化の判断基準と試行結果の部品分類

6.1 部品化の判断基準

部品化に関しては、標準部品規定に基づいて実施したが、特に次の点を重視した。

(1) アプリケーションプログラム（製品としてのソフトウェア）に対して共通性／再利用性が高いこと。

(2) アプリケーションプログラム側からみて、使いやすいこと。

① 利用者が望んでいる直接的な機能だけが部品のインタフェイスに反映されていること。

即ち、利用者が望んでいる機能を実現するために必要な準備・手続等の機能は隠蔽されていること。

（細かいことを知らなくても利用できること）

② 利用者の望む機能階層レベルで利用できること。

即ち、高位レベルの機能として使うか、下位レベルの機能を組合せて使うかを自由に選べること。

③ 部品を使用する時に、引数の設定がスムーズであること。

即ち、ある部品の引数を設定するために、コード変換などの準備に手間どらないこと。

6.2 部品分類

本試行での部品分類は、次の通りである。

(1) 画面表示・制御用	11本
(2) データベースアクセス用	57本
(3) データ処理用	15本
(4) CPU間通信用	38本
(5) その他	10本

合計 131本

部品の内容は対象システムの特徴（データベース、ワークステーションでの対話、ネットワークなど）が反映された結果となっている。

製品担当側から見ると部品群は一種のパッケージ群としてもとらえられる。

7. 試行対象システムの部品利用状況

部品利用状況をアプリケーションプログラム中における部品の割合でとらえる。

[1] 利用率の定義

部品の利用率を 使用部品の総ステップ数／アプリケーションプログラムの総ステップ数 で定義する

部品の利用率のとらえ方として我々は、次の2つの指標で評価した。

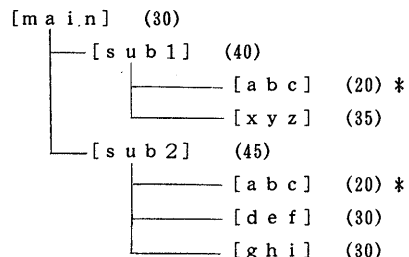
(1) 実質ステップ数ベース

モジュールの重複使用を考慮せずに実際に作成したステップのみでとらえる。

(2) 重複ステップ数ベース

モジュールの使用頻度を考慮したステップ数であり例えば或るモジュールAが2つの異なる上位モジュールから呼ばれていれば、モジュールAのステップ数を2倍した値とする。特にモジュールの使用頻度を考慮した重複ステップ数ベースの考え方は、機能の共通化への努力を重視した指標である。

例えば、以下のようなモジュール構成のプログラムにおいて、実質ステップ数は、230ステップであり、重複ステップ数は、250ステップである。



* 重複モジュール

[] : モジュール () : ステップ数

[2] 利用状況

アプリケーションプログラムにおける部品の利用状況を各アプリケーション毎の総ステップ数と使用した部品の総ステップ数との対比結果より把握した。試行対象の主なツールレベルでの部品の利用状況の一例を図3に示す。

部品利用率は、次のようになる。

実質ステップ数 56～96% (平均 89%)

重複ステップ数 71～99% (平均 97%)

今回の試行においては、部品の利用率は実質ステップ数ベースで平均89%、重複ステップ数ベースで平均97%であり、我々が当初目標とした新規開発の場合で部品利用率60%以上を達成できたことは、機能面の共通化という観点からの部品化が成功したといえる。

8. 部品についてのアンケート調査結果

今回の試行対象者に対して、アンケート調査を行った。アンケート内容は、自分が作成した部品に対する自己評価と、他人が作成した部品に対する評価である。アンケート結果を図4に示す。

(1) 利用のしやすさへの配慮

作成者は、理解性への努力をしたという意識を持っているが、使用者(他人)からみるとまだまだ不十分であるという評価であり意見のくいちがいが大きかった。

(2) 機能の実現性

今回の部品化の主目的である機能面の有効性については、アプリケーション側の想定していた機能を満たしているという評価であった。

(3) 部品の品質

部品の品質については利用者が考えていたレベルの品質ではなかったという評価であった。

これは部品の検査体制が当初十分でなかったために結果として一部の部品については検査の終わっていない段階の部品をアプリケーション側に提供してしまったことが影響している。

9. 利用のしやすさに関する作成者と利用者の意識のずれに関する分析

8項のアンケート調査では、部品の作成者としては、利用のしやすさを配慮したという意識を持っているのに対して、部品の利用者としては理解性についてあまり配慮されていないという意識を持っているという結果が与えられた。

本項では、この点について分析し、考察する。

部品作成者と部品利用者として部品の利用のしやすさに関して意識のずれを生じている背景として、技術レベル・対象システムに関する知識の量の違いがあげられる。

意識のずれを大きく生んだ状態は次のような場合である。

・技術レベルに関して

部品作成者>部品利用者

(高) (低)

・対象システムに関する知識の量

部品作成者>部品利用者

(多) (少)

9. 1 ドキュメントに関して

[1] 作成者の認識

標準部品規定に従って部品の仕様やインタフェース(コーリングシーケンス、引数の意味、データ構造など)や制約事項を記述した。

従ってこの部品のドキュメントを使ってアプリケーションプログラムを作成することについては、大きな問題はないはずである。

尚、一部のドキュメントには記述不足があったがそれは、ごく少数である。

[2] 利用者の認識

部品は全体として標準部品規定に従って記述されていたが、部品の利用のしやすさへの配慮を欠いていた点として次の項目をあげている。

①使い方の例の記述がない点

②参照すべき資料が多い点

③部品の動きが具体的な入出力データの関係を例にして示されていない点

④用語のうち意味のわかりにくいものがある点
そして、以下に示す具体的な要望をあげている。

(1) 使い方の例の記述

a) コーディングレベルで使い方を例示してほしい。

b) データ宣言などが必要な場合には、コーディングレベルで例示してほしい。

c) 関連する部品との間で順序関係(例えば事前に呼んでおくなど)について説明してほしい。

この場合文章で一言のべるだけでなくコーディングレベルで例示してほしい。

d) 或る目的を実現するための部品の使い方の例題(含む複数部品の組み合わせ)を示してほしい。

e) アプリケーションプログラムとしてどのように部品を使えばよいのかを示してほしい。

(2) ドキュメントの自己完結性を高めてほしい。

ドキュメントの自己完結性が低いものは、他の資料を参照しなければならないが、具体的に資料のどこにそれが書いてあるのかさがせないこともあるので必要資料は、添付してほしい。また共通的な資料を参照する場合でも極力添付してほしい。そして参照する場合は、どの部分かを明示してほしい。

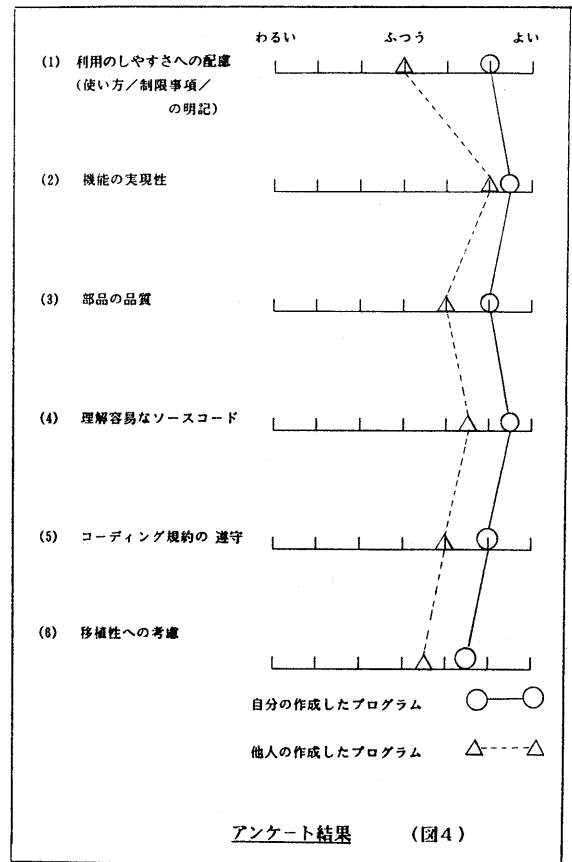
(3) 部品の働きを入力出力データの関係として例示する。

例えば可変長のテキスト形式が入出力データの場合には、例示がほしい。

(4) 用語の説明

使う人にとっては明確でも他人にとって多義性（複数の意味が考えられる）のある用語は、意味を図示や式で記述してほしい。

（例えば相対アドレス、ポインタ、相対ロケーション、相対位置など）



部品利用状況 (図3)

ツール		ツール A	ツール B	ツール C	ツール D	ツール E	合計
部品	実質	77,963	3,529	24,748	4,794	42,988	154,022
	重複	438,708	6,469	155,725	11,971	954,975	1,567,848
	比	5.6倍	1.8倍	6.3倍	2.5倍	22.2倍	10.2倍
非部品	実質	3,370 (4%)	1,544 (44%)	2,898 (12%)	1,576 (33%)	7,195 (17%)	16,583 (11%)
	重複	5,620 (2%)	1,544 (24%)	4,290 (3%)	1,576 (13%)	31,617 (3%)	44,647 (3%)
	比	1.7倍	1.0倍	1.5倍	1.0倍	4.4倍	2.7倍
部品	実質	74,593 (96%)	1,985 (56%)	21,850 (88%)	3,218 (67%)	35,793 (83%)	137,439 (89%)
	重複	433,088 (98%)	4,925 (76%)	151,435 (97%)	10,395 (87%)	923,358 (97%)	1,523,201 (97%)
	比	5.8倍	2.5倍	6.9倍	3.2倍	25.8倍	11.1倍

- 実質ステップ数……モジュールの重複使用を含まないステップ数
- 重複ステップ数……モジュールの重複使用を含めたステップ数
- 比……重複ステップ数/実質ステップ数

[3] 分 析

今回の部品作成者の認識と部品利用者の認識を客観的に比較すると、作成者は、大局的に見ている（即ち特に問題となったものは例外として除外してみている）のに対して、利用者は、局所的に見ている（即ち、一部わかりにくかったものについての悪い印象のみが強調されていて、良い方への印象はほとんど強調されない）という相異がある。

しかし、利用者が指摘した事項は、たとえ局所的とはいえ、利用のしやすさに関して重要な問題提起をしている。

本試行では製品担当と部品担当とに役割を分担して開発を進めたが、チーム編成上、部品担当を重視して経験者を割当てたのに対して製品担当が初心者中心に近くなってしまった。こうした点から部品担当側への期待度・依存度が高くなり、部品担当側がプログラムとしての部品の作成に追われ、利用のしやすさに対する配慮が、プログラムの初心者に対するレベルまでは、いきとどかなかったという実態であったと考えられる。

10. おわりに

本試行によって、新規システムの開発に対しても最初から部品化し、この部品を使ってアプリケーションを作成するという手順が可能であることがわかった。今後、別のシステムに対しても本試行を適用し評価する予定である。

<参考文献>

- [1] 高橋 他
"ソフトウェア生産工業化システム IMAP"
東芝レビュー 1986 Vol.41 No.8
- [2] 金子 他
"ソフトウェアの部品化・再利用支援技術"
東芝レビュー 1986 Vol.41 No.8
- [3] 福田 他
"ソフトウェア管理とソフトウェア生産環境"
東芝レビュー 1986 Vol.41 No.8