ソフトウェア工学 64－14
（１９８９．２．３）

部品指向の設計支援環境 50SM

－ 部品化・再利用支援機能の強化 －

宗近修久　小尾俊之　蓮田広保　松村一夫

（株式会社 東芝　システム・ソフトウェア技術研究所）

ソフトウェア開発における生産性と品質の向上を目的としたソフトウェア設計支援環境５０ＳＭを構築中である．５０ＳＭ(50 Steps/Module)は，部品化・再利用技術を基本とし，そのための部品指向のモジュール設計記述法ＴＦＦ(Techincal description Formula for Fifty steps/module design)を用意している．本稿では，まず従来の５０ＳＭツールの評価から部品化・再利用を促進するための課題をまとめる．次に，ＴＦＦの特徴であるパッケージ型モジュールの考え方に基づいた部品開発支援，部品を利用した製品開発支援，部品管理の機能について述べる．

Software Design Support System 50SM: Development and Reuse of Software Components

Nobuhisa MUNECHIKA, Toshiyuki OBI, Hiroyasu HASUDA and Kazuo MATSUMURA

Systems & Software Engineering Laboratory, TOSHIBA Corporation
70, Yanagi-cho, Saiwai, Kawasaki, 210, Japan

50SM (50 Steps/Module) is a software design support system aimed at improvement of productivity and quality. 50SM is based on reuse oriented and includes design description methods TFF.

We have investigated an early version of 50SM and recognized partially weakness of reuse promotion. This paper describes the functions of 50SM promoting software reuse, including

* development of components called package type,
* reuse of package type and white box components, and
* management of components.

It is effective to support development and reuse of package type components.

1. はじめに

ＣＡＳＥ(Computer Aided Software Engineering)環境が最近注目を集めており，ソフトウェアの生産性の向上と品質の向上に期待が寄せられている．

我々は，ソフトウェア生産の工業化を目的とした一貫生産支援システムＩＭＡＰ(Integrated software MAnagement and Production support system)を研究・開発し普及推進している[1]．工業化を目指すには生産性，品質など，個人の能力に依存しないようにする必要がある．そのための１つのアプローチとして部品化・再利用技術がある．これは，品質を保証した標準部品をあらかじめ用意しておき，それらの部品を組み合わせてソフトウェアを作り上げるアプローチである．

本稿では，パッケージの考え方を導入した部品開発支援とパッケージ型部品を利用した製品開発支援を特徴とする設計支援環境５０ＳＭ(50 Steps/Module)について述べる．第２章では５０ＳＭの概要について述べる．第３章では従来の５０ＳＭのＣＡＳＥツールとしての使用評価とその課題について述べる．第４章では第３章の評価に基づき今回機能強化を行った設計書エディタの部品化・再利用支援機能を中心に述べる．

2. ５０ＳＭの概要

2.1 基本概念

ソフトウェアライフサイクルのモジュール設計（詳細設計）からコーディングまでを支援する方法論，ツールを５０ＳＭと呼ぶ．５０ＳＭはＴＦＦ(Technical description Formula for Fifty steps/module design)という部品指向のモジュール設計記述法をベースとしている[2]．

ＴＦＦの特徴は次の通りである．

(1) ３種類のモジュール

タスクの構成単位として，処理型，データ型，パッケージ型の３種類のモジュールを用意している．パッケージ型モジュールについては後述する．

(2) 50ステップ制限

１つのモジュールを50ステップで作成するという積極的制約は，一覧性の向上，制御構造の単純化，モジュール分割の目安，テスト，品質保証のしやすさなどにつながる．ＴＦＦで設計を行った場合とそれを使わないで設計を行った場合のソースコードの保守性品質の比較実験によると，ＴＦＦ使用の方がばらつきの少ない品質の安定したモジュール作成が行えることが分かった[3]．また，50ステップ程度を目安にすることによりバグが少ないという実験結果も得られている．

(3) モジュール部品を前提とした設計

タスクの上位のモジュール分割の目安がついた時点で必要そうなモジュール部品（以降は単に部品と呼ぶ場合もある）を調達し，設計は可能な限りモジュール部品を使用して設計を行う．部品を使用することで，生産性，品質の向上が期待できる．

(4) フォームシートと階層的な図式記法

記述のもれをなくし設計書の標準化ができるように６種類のフォームシートを用意している．

・シートＡ　モジュールの関連図
・シートＢ　処理型モジュールの外部仕様
・シートＣ　処理型モジュールの内部仕様
・シートＤ　データ型モジュールの内部・外部仕様
・シートＥ　パッケージ型モジュール外部仕様
・シートＺ　自由形式の補足説明

ＴＦＦでは外部仕様と内部仕様を区別して記述できる．外部仕様はモジュールの外から見たふるまい，インタフェースなどを記述でき，ブラックボックス型部品の記述には不可欠である．内部仕様はモジュールの実現方法を記述するために図式言語を用意した．特徴としては，順次，反復，分岐の３つの展開構造によるトップダウン設計ができる，抽象化の考えや構造化設計が徹底できる，モジュール部品の呼び出し（使用）がすぐ分かるように目立たせる，などがある．

2.2 システム構成

モジュール設計データベースを中心に，ＴＦＦ設計書の編集，コード生成などのツール群と，ＴＦＦＥＤで作成したモジュール部品のデータベースを中心に，部品を蓄積，管理，検索するツールから構成されている．システム構成を図1.に示す．

おもなツールは次の通りである．

(1) ＴＦＦＥＤ (TFF EDitor)

モジュール設計記述法ＴＦＦに基づいた設計書を編集するための設計書エディタ．

(2) ＤＯＴ (DOcument Tracer)

シートＣ上でデザインレビューを支援するツール．

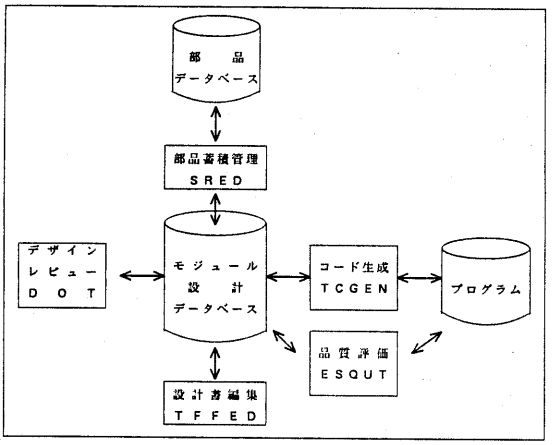


図1. システム構成図

(3) ＴＣＧＥＮ (Tff to Code GENerator)

ＴＦＦ設計書からソースプログラム，ソースプログラムからＴＦＦ設計書を生成するツール．

(4) ＥＳＱＵＴ (Evaluation of Software Quality from User's viewpoinT)

ＴＦＦ設計書とＣソースプログラムのレベルでの保守性品質を定量的に評価するツール．

(5) ＳＲＥＤ (Software module Reuse Environment to assist software Development)

部品の登録，検索，払い出しなどを管理するツール．

2.3 ツールの特徴

ここでは部品化・再利用支援機能に関連する設計書エディタＴＦＦＥＤと部品蓄積管理ツールＳＲＥＤについて説明する．

2.3.1 設計書エディタＴＦＦＥＤ

(1) ＴＦＦ設計書とソースコードの統一管理

コード生成ツールを使用することにより設計書とソースコードが対応管理できる．

(2) 部品開発と製品開発への対応

部品化・再利用を支援する機能が今回強化された．部品開発ではパッケージの考えに基づいた部品作成を，製品開発では部品利用を支援する．

(3) ＴＦＦ設計書の編集

統一したユーザインタフェースによる編集ができる．特にシートＣはモジュール分割，制御パターンの組み込みなどの豊富な編集機能，レイアウト自動配置などで手書きに比べて編集，修正が容易である[4],[7]．また，入力チェック，構文エディットガイダンスなどで，誤った設計情報の入力を防いでいる．

(4) ＴＦＦ設計書のＬＢＰ出力

Ａ４サイズで高速，高品質に出力できる．

(5) 自動化

シートＣを作成するとモジュールの参照関係を表す全体関連図（シートＡ）が自動生成される．

(6) モジュール設計情報の一貫性

モジュール設計データベースにより関連する設計情報の反映を自動的に行い，一貫性を保っている．

(7) データ互換

エンジニアリングワークステーション(当社AS3000)版とパーソナルワークステーション(当社J-3100)版で相互に設計データを利用できる．

(8) プログラムの自動生成

コード生成ツールを使うことによりＴＦＦ設計書からＣ，ＦＯＲＴＲＡＮ，Ａｄａの各種言語のソースコードが生成可能である．なお，Ａｄａのコード生成については試作段階である．また，コード生成，コンパイル，リンクをＵＮＩＸのｍａｋｅと同等の機能で自動実行できる．

2.3.2 部品蓄積管理ツールＳＲＥＤ

(1) 部品データベース

部門-プロジェクト-個人の３階層に分けて管理する．これにより，担当者ごとに検索対象のカストマイズ，検索範囲の限定などが可能である．

５０ＳＭの環境で扱うモジュール部品とは，基本的にはＴＦＦ設計書，ソースコード，オブジェクトコードからなる．そのうち，ＴＦＦ設計書は必須であるが，ソースコード以下はターゲットマシンの環境などにより，なくてもよい．

(2) 部品検索

部品の検索は，キーワード検索，分野や使用頻度に基づいたブラウザ検索がある．

(3) 部品払い出し

部品の個別払い出しのほかに，パッケージ単位での一括払い出しの機能もある．

(4) 部品登録

管理しやすく，かつ検索しやすい構造で部品を部品データベースに登録する．

3. ＣＡＳＥツールとしての５０ＳＭの評価

５０ＳＭツールはいままで各種プロジェクトに適用されている．ここでは，使用者へのインタビュー，アンケート結果から，特に部品化・再利用支援の弱点を明確にし強化することを目的として，以下の観点に限定して整理する．評価の観点は次の通りである．

(1) 再利用支援

(2) 設計，コーディング支援

(3) 大規模プロジェクトでの運用

評価の対象としたプロジェクト10件のうち，代表的な3件（いずれも制御用の組み込みソフトウェア）を表1.にまとめる．

以下，観点別に長所/短所（課題）の概要を述べる．なお，第２章の内容はこの評価に基づいて機能強化した５０ＳＭの概要である．したがって，「2.3.1 (2) 部品開発と製品開発への対応」や「2.3.2 ＳＲＥＤ」については，評価時点では機能は不十分であった．

(1) 再利用支援

[長所]

・再利用率向上

再利用率（(部品数/全モジュール数)*100）が60％くらいのプロジェクトが多く，従来より再利用率が向上したという意見が多かった．これは，５０ＳＭの部品

プロジェクト名	A	B	C
ステップ数	約500K	約60K	約10K
使用言語	C	C	ｱｾﾝﾌﾞﾗ
使用者数	約120人	約20人	7人

表1. プロジェクトの概要

化の思想による効果である．ただし，どんな部品をあらかじめ作るかに強く依存する．

・工数削減

部品利用を前提にすることで重複開発，テスト工数などが減った．ある試算によると，約85％の工数削減になるというデータもある〔5〕．

以上の長所は，ツール支援というよりは５０ＳＭの部品化・再利用の考え方によるものである．

〔短所/課題〕

★部品ツールとの連携不足

ＳＲＥＤの機能が限定されていて，部品の管理など使用者にかなり負担がかかっていた．また，ＳＲＥＤとＴＦＦＥＤとの連携が不十分であり，部品を利用できる状態にするのに時間がかかった．

★パッケージ支援機能の不足

ツールとしてパッケージの支援が十分でなかった．したがって，パッケージはバインダに設計書をファイリングしてマニュアル運用する場合が多かった．

(2) 設計支援

〔長所〕

・デザインレビュー，コードレビューのしやすさ

アセンブラの場合も含めてプログラム構造が単純化された〔3〕．また，フォームシートやビジュアルな図式言語によりデザインレビューの際にチェックしやすい．また，コード生成機能を使用した場合，設計書とソースコードの対応やコメント付けによりコードレビューがしやすい．

・モジュールがコンパクト

50ステップの目安でモジュールがコンパクトになる．高級言語を使用したプロジェクトでは，ほとんど90％が50ステップ以内であり，50ステップ制限が有効に作用している．また，バグの出現も50ステップを越えると急激に増加する傾向がある．

・設計書の編集，修正が非常に容易

エディタを使えば編集，修正が非常に簡単であり，初期入力の場合は約1/2で済む〔4〕,〔7〕．また，関連するモジュール設計情報もツールにより生成，反映されるので，手書きに比べて一貫性の保持が容易である．

〔短所/課題〕

・操作性に起因する思考の中断

ツール操作の習熟度にもよるが特に初期入力の際，本来の設計作業に集中しにくい場合がある．

★関連する部品の検索，参照のしやすさ

ＴＦＦＥＤで設計中に，関連する部品を検索し内容を確認することが容易でない．使用者にとってはツールの切り替えを意識する必要があり，思考の中断が起こる．また，関連する部品の集合であるパッケージを支援することで，部品の参照のしやすさ，利用のしやすさ，管理のしやすさは格段に向上すると考えられる．

★設計書上での部品の扱い

設計書上で部品と使用者が作成したモジュールの区別が明確でない．管理上もデバッグ，テストをする上でも，区別するほうが効率がよい．

(3) 大規模プロジェクトでの運用

〔長所〕

・作業分担のしやすさ

タスク単位でＴＦＦＥＤを運用するので，タスク単位に担当者を割り当てることは容易にできる．他にもモジュール設計とコーディングに対してそれぞれ担当者を割り当てる，タスクの上位のモジュール分割と外部仕様までとそれ以外で分担する，などの運用も可能である．

・設計の標準化

設計書の標準化ができ，プロジェクトメンバのコミュニケーションの共通基盤となる．また，設計作業もツールを使用することで標準化できる．５０ＳＭは大人数のプロジェクトにおいて，設計の記述法がきちんと決められている，それに基づいた教育ができる，ツール支援がある，などの理由で有効である．

〔短所/課題〕

・大量ドキュメント出力

ドキュメントが大量なので，高速出力は欠かせない．また，出力フォーマット，設計書修正時のページ付け，必要なドキュメントのみを出力する部分出力など柔軟に対応する必要がある．

★マスタ情報の管理

作業を分担して行うので，版管理を含めてマスタ情報を一元管理しておく必要があるが，運用で補うことが多かった．

これまで述べた評価のうち，★印の付いた項目は部品化・再利用を支援するのに必要な機能である．次章では，これらの支援機能を中心に説明する．

なお，今回の評価は前述の通り観点を限定し，かつ定性的な評価が中心であったが，今後は定量的な評価を行い別途報告したい．

4. 部品化・再利用支援機能の強化

まず，我々の部品化・再利用の前提となる考え方を述べてから，ツールの支援機能に説明する．

4.1 部品管理と部品利用の環境

4.1.1 組織体制

ＩＭＡＰではモジュール部品を前提とした設計を行うことにしている．そのため，体制も部品を開発する部品部と，部品を使用して製品を開発する製品部とに分かれている．部品部またはグループを専任組織として独立させることにより，部品の品質向上，標準化の促進，責任の明確化を図っている．

4.1.2 開発環境

ホストマシン(G8050)，ＥＷＳ(AS3000)，ＰＷＳ(J-3100)をネットワークで接続する．部品部の部品蓄積用

のデータベースはホストにある．ワークステーションは部品開発と製品開発に使用する．ワークステーション側では製品開発で使用する部品の払い出しを受ける．

4.1.3 部品とモジュールの管理

ＴＦＦＥＤで管理する実体をモジュール，ＳＲＥＤで管理する実体を部品と呼び区別する．

部品指向の設計支援環境として重要なことは，モジュール，部品の作成ばかりでなく，作成した部品を利用しやすい形で蓄積し使用者に提供することである．

５０ＳＭでのモジュールと部品の管理運用を述べる．

〔モジュールの管理〕

・階層構造のディレクトリを単位として，複数のモジュールを管理する．(図2.参照)

・１つのディレクトリで１つの製品（タスク）またはパッケージを管理する．したがって，１つのディレクトリには複数の製品，パッケージは存在しない．

・ＴＦＦＥＤで開発の終了したモジュール部品は，部品としての認定を受けた後，ＳＲＥＤを使い登録する．

・複数のディレクトリをまたがってシートの編集はしない．つまり，１つのタスクのシートしか編集しない．別のディレクトリにあるシートを編集する場合は，払い出しを行う必要がある．ただし，編集しないで参照だけ（パッケージ型部品などの場合）なら別のディレクトリにあっても可能である．

〔部品の管理〕

・ブラックボックス型部品として扱う場合は，部品蓄積用のデータベース（部品データベース）へのリンク情報をＴＦＦＥＤが管理し，実体の管理はＳＲＥＤが行う．

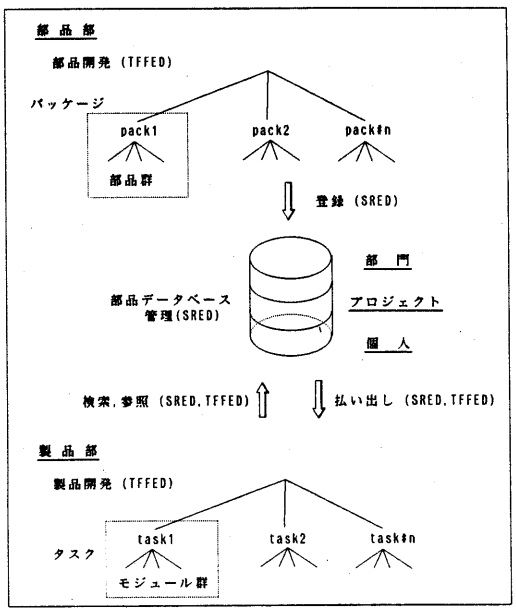


図2. モジュールと部品の管理

・部品データベースを部門-プロジェクト-個人の３階層に分けて管理する．

・階層間での払い出し，登録はＳＲＥＤで行う．

・部品データベースからＴＦＦＥＤ管理下への払い出しは，ＴＦＦＥＤとＳＲＥＤのそれぞれで可能である．

このように，５０ＳＭではモジュールと部品を分けて管理し，使用者が必要な部品を容易に検索し利用できる環境になっている．

4.1.4 必要な機能

部品開発と製品開発では必要とする機能や運用は異なる．ここでは，部品部と製品部で必要な機能を表2.にあげる．

部品と製品の開発（設計）は，ＴＦＦＥＤを使用する．

部品の管理は，基本的にはＳＲＥＤで行うが，簡単な部品検索などはＴＦＦＥＤでも行える．

なお，成果物の管理用ツールは別にある．

部　品　部	製　品　部
部品開発（設計）	製品開発（設計）
部品管理（登録， 払い出し）	部品管理（検索， 参照， 払い出し
	製品（成果物）管理

表2. 部門別の機能一覧

4.2 部品開発の支援機能

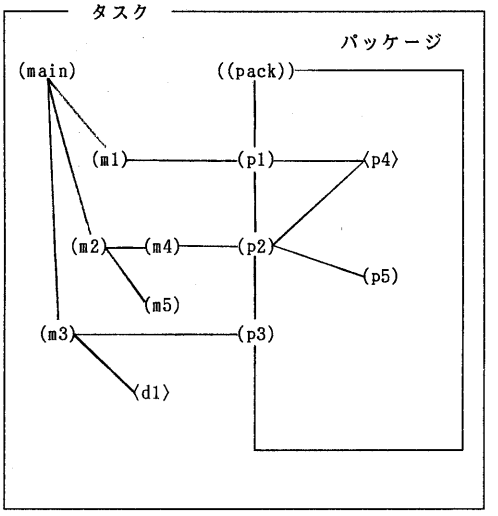
５０ＳＭにはパッケージの概念があり，手書きシートの運用では使われていたが，ツール上では十分支援されていなかった．ここでは，まずパッケージの概念を述べてから，今回強化したパッケージ作成支援機能を説明する．

4.2.1 パッケージについて

ＴＦＦの特徴の１つにパッケージ型モジュールがある．これは，論理的に互いに関連したデータ型モジュールと処理型モジュールをひとくくりにまとめたものである．Ａｄａのパッケージ[6]に近い考え方である．これを使えば抽象データ型に基づいた設計が行え，変更に強く再利用しやすい設計ができる．

モジュール部品をパッケージ型モジュールの形式（パッケージ型部品と呼ぶ）で実現，提供することは，部品開発，製品開発の両方にとって重要である．１つ１つバラバラよりもパッケージというまとまった単位で実現，提供する方が管理，保守しやすい．特に，ソフトウェアの規模が大きくなるにしたがい，パッケージは有効である．

図3.にタスクとパッケージの関係を示す．



( ) 処理型モジュール
〈 〉 データ型モジュール
(( )) パッケージ型モジュール
——— 参照関係

図3. タスクとパッケージの概念図

4.2.2 パッケージモードによる作成

(1) パッケージモードで起動

部品（パッケージ）開発と製品（タスク）開発ではツールの機能，動作が一部違うので起動時にパッケージモードを指定する．

(2) シートEの編集

パッケージ型モジュールの外部仕様であるシートEの記述項目には，パッケージ全体の表題，概要，使用形式（呼び出し方法），外部インタフェースモジュール，使用上の制限，パッケージ内部モジュールがある．この中で外部インタフェースモジュールとパッケージ内部モジュールはパッケージに所属する処理型モジュールとデータ型モジュールのモジュール名（と引数名）とその表題を並べる．ここで，外部インタフェースモジュールとは直接パッケージの使用者から呼び出せるモジュール（図3.のp1,p2,p3）であり，パッケージ内部モジュールとは外部から使えないモジュール（図3.のp4,p5）のことである．

シートEを書きはじめる時点で外部インタフェースモジュール，パッケージ内部モジュールが全部決定しているのは，例えば次のように限られている．

・以前（分野や対象が）同じようなパッケージを作成した経験がある．

・インタフェースのみあらかじめ決定していて，それ以降各モジュールの設計，コーディングは別の担当者が行う．

実際，机上（手書きシート）での作業を調べてみると，各モジュールの外部仕様を書き終えたあとシートEの外部インタフェースモジュール，パッケージ内部モジュールに写すという運用が多い．この場合，同じ設計情報を2度書かなくてはならないため，書き間違えや書き落としなど，設計情報としての一貫性を損なう可能性がある．したがって，ツール支援を行う場合はこれらを考慮して次のようにして一貫性を保っている．

・シートEの外部インタフェースモジュール，パッケージ内部モジュールを記述した場合，自動的に対応するモジュールの外部仕様にモジュール名（と引数と）表題を設定する．なお，モジュールの外部仕様が未作成の場合はシート自体を自動的に作成し設定する．

・パッケージに所属するモジュールの外部仕様を作成した場合，シートEの外部インタフェースモジュール，パッケージ内部モジュールに自動反映する．

つまり，パッケージの外部仕様と処理型／データ型モジュールはどの順に作成しても設計情報の一貫性は保たれる．

(3) モジュール作成

各モジュールの外部仕様の所属の欄には自動的にパッケージ名が記入される．また，外部仕様の表題は自動的にシートEの外部モジュール，パッケージ内部モジュールの表題に反映される．

シートCを作成すると自動的にパッケージの全体関連図が自動生成される．

(4) 全体関連図（シートA）の自動生成

パッケージの全体関連図はシートB，C，D，Eを作成すると自動生成される．

タスクの場合と違い，パッケージの場合はタスクのメインモジュールに相当するモジュールがないので全体関連図の意味あいが少し異なり，その1レベル目にはパッケージの外部インタフェースモジュールを並べる．これは，シートEの外部インタフェースモジュールから自動生成する．

また，2レベル以降の（外部インタフェースモジュール，パッケージ内部モジュールの）参照関係はシートCより自動生成する．

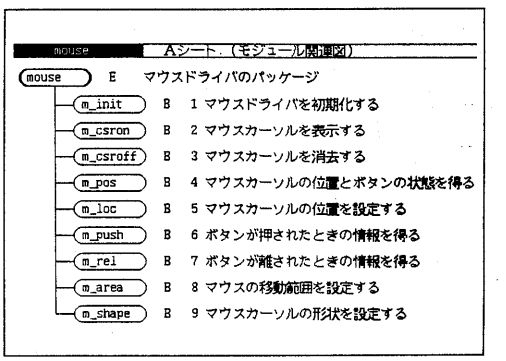


図4. パッケージの全体関連図

Ｅシート上でも外部インタフェースモジュールとパッケージ内部モジュールは区別しているが，パッケージの全体関連図の方がより視覚的で分かりやすい．

パッケージの全体関連図の例を図4.に示す．この例では，すべてのモジュール(m_*)が外部インタフェースモジュールである．

(5) ソースコード，オブジェクト生成

Ｃ，ＦＯＲＴＲＡＮはＴＦＦ設計書からのコード生成機能により，パッケージに所属するモジュールのソースコード生成，さらにコンパイルまで自動実行できる．

パッケージ型部品をオブジェクトファイルの形式で提供すれば，ＴＦＦＥＤのプログラム生成の機能を使い，パッケージを使用した大規模な製品開発にも十分対応できる．

また，１つの製品をＣとアセンブラなど複数の言語で開発することがある．その場合，アセンブラなどソースコードの自動生成を支援していない言語で記述する部分をパッケージ，それ以外を製品として切り分けることで適用できる．

(6) パッケージ型部品のテスト，登録

品質を保証するためのテストを行う．そのためには，部品は実行単位ではないので，実行可能なテストプログラムを作る必要がある．テストプログラムは１つの製品（タスク）としてＴＦＦＥＤを使用して作成する．パッケージ型部品を使用した製品開発は次で説明する．

以上，パッケージ型部品の開発支援について述べた．ツール支援による手書き運用での煩わしさの解消，設計情報の一貫性維持などにより，使用者の負担が軽減され部品化しやすい環境が構築できる．

4.2.3 ＳＲＥＤによる部品登録

パッケージ型部品のテストが終了し，その品質が保証されたら部品としての認定を受け，部品データベースへの登録をその管理者がＳＲＥＤを使用して行う．

4.3 設計書エディタからの部品利用機能

部品を使用して製品開発を行う場合のツール支援に説明する．

4.3.1 部品検索機能

部品検索機能の使い方は次の２通りが考えられる．

１つは，部品を利用した製品開発を行うにあたり，あらかじめ使えそうな部品を調達する場合である．この場合は，ＳＲＥＤを使い，キーワード検索，分野や使用頻度に基づいたブラウザ検索などができる．

もう１つは，設計書編集中に部品を確認する場合である．この場合は，ＴＦＦＥＤを使い，キーワード検索などで内容の確認が行える．

なお，ＥＷＳ上ではマルチタスク環境であるので，設計書の編集と部品検索の画面が別々のウィンドウで同時に表示できる．

4.3.2 部品の取り込み

ＴＦＦＥＤのモジュール管理法により，パッケージ型部品のようなブラックボックス型部品とホワイトボックス型部品では再利用する場合の扱いが違う．

まず，ホワイトボックス型部品の再利用支援機能について説明する．ホワイトボックス型部品の場合は部品データベースから製品のディレクトリに実体をコピーし，製品の管理下に置く必要がある．そのため，ツールでは２通りの支援をしている．

１つは，ユーティリティの「部品の再利用」で実体をコピーする方法である．コピー時に部品名を変更することもできる．この場合，未参照モジュール（実体はあるがどのモジュールからも呼ばれていないモジュール）として全体関連図に登録される．実体をコピーしたあとは，通常のモジュールとして編集すればよい．なお，そのモジュールのもとになった部品はシートＢの「部品名」に記述することで履歴が分かる．

もう１つは，シートＣの編集中に「部品の再利用」を使う方法である．この機能を選択し部品名を入力すれば，指定位置に自動的にモジュール呼び出しの記号が付く．そして実体のコピーも行われる．

このようにホワイトボックス型部品の再利用時には，実体のコピーと設計情報の更新を自動的に行い，全体の一貫性を保っている．また，あらかじめ再利用する部品を一括してコピーしたい場合はユーティリティを使用し，１つずつ必要なときにはシートＣ編集の中から使用するというように，運用の形態にあわせて使い分けることができる．したがって，使用者の負担は非常に少なくなり再利用が促進される．

次に，ブラックボックス型部品の再利用支援機能について説明する．ブラックボックス型部品はパッケージ型部品として提供すると仮定する．

製品で使用するパッケージ型部品は製品とは別に管理されていて，製品側からは参照だけできる．なお，参照できるのは外部仕様だけである．内部仕様は情報隠蔽の立場から製品開発者には非公開になっている．

パッケージ型部品がホワイトボックス型部品の場合と違うのは，実体のコピーを伴わないで呼び出し関係だけをリンクするということである．プログラム生成機能などを実行する場合にこの情報を使用する．

パッケージ型部品の使用は通常のモジュールの場合と同様，全体関連図に自動反映される．図5.にその例を示す．

4.3.3 設計図面上での部品の扱い

シートＣ編集でパッケージ型部品を使用する場合は，通常のモジュール呼び出しの記号を横に長くした記号を使う．記号の中には所属するパッケージ名と部品名を'/'で区切って書く．パッケージ名はそのパッケージに所属するモジュールを参照するときに必要な情報なので，そのパッケージの使用をあらかじめ宣言しているかを入力時にチェックし，誤ったパッケージ名の指定を防いでいる．

全体関連図上でもパッケージ型部品は，通常のモジュールとは区別して書く．

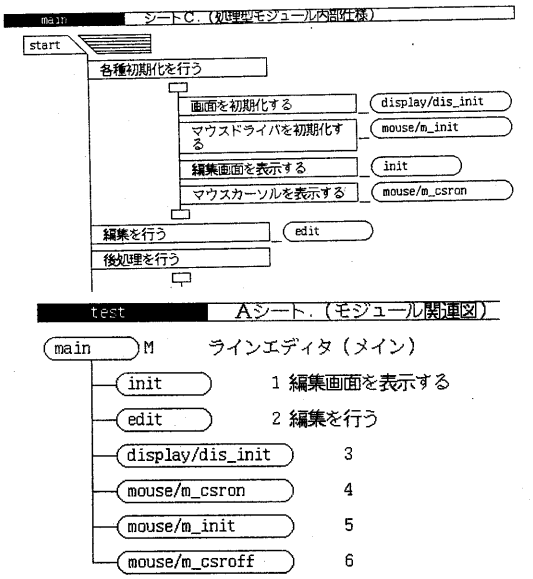


図5. パッケージを使用したシートC，A

パッケージを使用したＴＦＦ設計書の例を図5.に示す．この例では，display,mouseの２つのパッケージのモジュールを使用している．なお，init,editは使用者が作成するモジュールである．

このように，製品でのパッケージ型部品の使用を通常のモジュール呼び出しの記号と区別して書くことにより，部品との切り分けが明確になり，プログラムを理解する上でも，保守する上でも有効である．部品は品質が保証されているので，テストやデザインレビューを行う場合など，部品とのインタフェースにのみ着目すればよく，テストやレビューの効率が上がる．

4.3.4 設計書出力での部品の扱い

ＴＦＦ設計書のＬＢＰ出力は，製品で作成したモジュールはもちろんのこと，使用しているパッケージ型部品（ただし，外部仕様に限る）など，デザインレビューや保守に必要な設計ドキュメントがまとめて出力できる．

5. まとめ

部品指向の設計支援環境５０ＳＭの部品化・支援機能を中心に述べた．

部品化・再利用を行う上での問題点のいくつかは，５０ＳＭにより解決される．期待される効果をまとめる．

(1) 理解性

部品の機能や使い方は，外部仕様書を見ればすぐに分かる．

部品を使用した製品開発では，部品部から提供された（パッケージ型）部品と使用者が作成したモジュールを明確に区別し理解しやすくなっている．変更による影響範囲の把握も容易である．

パッケージの支援も，部品の管理や再利用をする上で非常に有効である．

(2) 信頼性

部品は専任組織の部品部がきちんと品質を保証し，責任が所在が明確である．

(3) 利用性

部品を利用するには，検索，払い出しを行う必要がある．部品の調達はＳＲＥＤの機能により容易に行える．本来の設計作業ではない部品の管理などもツール支援し，使用者の負担を軽減する．

今後とも，実際のプロジェクトに５０ＳＭを適用し，部品化・再利用支援機能を中心として定量的な評価を行いたい．

参考文献

[1]大筆他：ＩＭＡＰシステム(1)～(10),情報処理学会第31回全国大会4F-1～4F-10,1985

[2]松村他：ソフトウェア設計記述技法,東芝レビュー，VOL.41 NO.8,1986

[3]古谷他：設計記述法ＴＦＦの評価,情報処理学会第32回全国大会5J-4,1986

[4]宗近他：５０ＳＭにおけるソフトウェアＣＡＤシステムの構築,情報処理学会第36回全国大会5M-8,1988

[5]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,東芝レビュー,VOL.41 NO.8,1986

[6]石畑清・疋田輝雄著：Ａｄａプログラミング,岩波書店,1986

[7]古川他：モジュール設計・コーディング一貫支援システム５０ＳＭの評価,情報処理学会第34回全国大会5M-8,1987