

グループ開発における構造化分析作業の考察

1U-4

内藤 創 黒木 宏明 斎 直人 国立 勉
NTTソフトウェア研究所

1. はじめに

ソフトウェアの大規模化にともない、システム分析/設計作業が複雑になってきている。このため、システム分析/設計作業でも複数人で分担開発するグループ開発作業が行われている。さらに、ソフトウェアの品質向上、生産性向上のためにCASE (Computer Aided Software Engineering) ツールが導入されつつある。CASEツールでもグループ開発作業を支援するため、排他制御機能等を組み込んでいるが、十分整理された作業モデルに基づいて構築されている訳ではない。

本稿では、構造化分析/設計手法をベースにした開発作業を分類し、各作業モデルの適用および各モデルの設計情報の一元管理を実現するために必要な機能を整理する。

2. 設計情報の構成と問題点

構造化分析における設計情報は、

- ①構造化分析/設計手法に基づいて、機能やデータの階層的な構造や論理的な対応関係を記述する各種ダイアグラム
- ②ダイアグラムから抽出された設計情報を格納しているデータベース

から構成されている。構成を図1に示す。

主なダイアグラムを以下に示す。

- (1) データフロー図
- (2) データ構造図
- (3) ER図
- (4) モジュール構造図
- (5) 状態遷移図
- (6) 状態遷移表

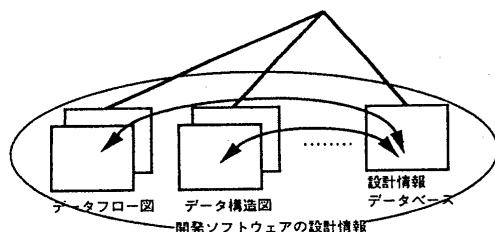


図1 設計情報の構成

構造化分析手法をトップダウンに分割するアプローチのソフトウェア開発に適用すれば、システム分割により複数人で同時に並行して作業を行える。しかし、構造化分析手法では設計情報を名標により識別しているため、論理的に等しいデータには同一名標を付与する必要がある。このため、複数人で並行開発作業を進めると以下の問題がある。

- ①異なる名標を同一意味で使用する誤りがある。
- ②同一名標を異なる意味で使用する誤りがある。
- ③システム全体で設計情報の一貫性を保持するため、他部分を含めてチェックする必要がある。
- ④設計情報を共有した場合、情報の同時アクセスが生じる。

これらの問題を解決する必要がある。

具体的な作業モデルを検討することにより、上記条件を支援する機能を抽出する必要がある。

3. 作業モデルの検討

分析設計作業を設計情報の管理方法の観点から整理する。

3.1 設計情報の管理

複数担当者で設計情報を分割し設計作業を行う場合、作成する設計情報は以下の2点に大別できる。

(1) 共通設計情報

担当者が個別に設計するサブシステム共通の設計情報であり、各サブシステムから参照される設計情報である。このため、共通設計情報の変更は全サブシステムに影響を与える。

構造化分析手法では、上位のデータフロー図、データ構造図、テーブル名に相当する。

(2) 個別設計情報

担当者が個別に設計するサブシステム内でのみ有効な設計情報であり、サブシステム内でしか参照されない設計情報である。

構造化分析手法では、下位のデータフロー図、プロセス仕様書、データ項目、属性に対応する。

3.2 作業モデル

共通、個別設計情報を複数の担当者で管理するモデルを以下に示す。

(1) 単一モデル

開発者を一人とする開発モデルである。

複数人が作業する場合でも仮想的には一人で実施する作業モデルである。このため、共通、個別設計情報の管理は各人で実施することになる。

Study of Structured Analysis works among Software Development Teams

So NAITO, Hiroaki KUROKI, Naoto ITSUKI,

Tutomu KUNITACHI

NTT Software Laboratories

(2) 共有モデル (図2)

共通設計情報は各担当者で共有し、個別設計情報は各担当者に割り当て、詳細化を進めるモデルである。しかし、設計情報データベースは各担当者共通である。共通設計情報に対して複数アクセスを許さない排他制御を実施することにより設計情報の同時アクセスを防ぐモデルである。このモデルには、以下の問題点がある。

- 設計情報のチェックは、他のサブシステムも合わせて行われるため、設計作業効率が低下する。

実際、大規模システムでは設計情報が膨大でチェックに時間がかかるから設計作業中には設計作業のチェックがおこなわれず、最後に一括してチェックしようとする。この結果、修正作業量が膨大となり最終的に一貫性のとれた設計情報データベースの構築ができないのが現実である。

(3) 分割統合モデル (図3)

共通設計情報およびサブシステムに分割された設計情報をもとに各担当者が詳細化する。担当者毎に設計情報データベースを持ち最後に詳細化された設計情報の統合を行うことでシステム全体の設計情報を構築するモデルである。

このモデルでは、以下の問題点がある。

① 担当者による共通設計情報の修正

各担当者が共通設計情報を勝手に修正できるため、サブシステム間でインタフェースがとれなくなる恐れがある。

② 担当者間での名標衝突

各担当者は他のサブシステムの設計情報を意識することなく詳細化できるため、名標衝突が生じる恐れがある。このため、設計情報の統合時には矛盾なく統合する必要がある。

このモデルでは、大規模ソフトウェア開発でも各担当者に関して設計情報のチェックを行えるため、設計情報の一貫性を保持しやすい。また、チェックする範囲が限定されたため作業効率が良い。

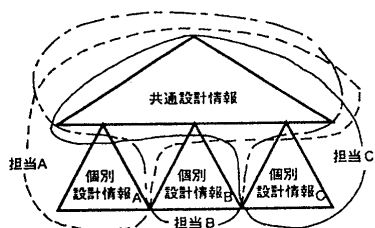


図2 共有モデルの設計情報

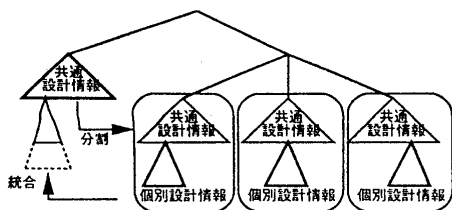


図3 分割統合モデルの設計情報

4. 各作業モデルの問題点と実現性

各作業モデルを以下の観点で比較する。

- (1) 設計情報の共有
- (2) 設計情報の隠蔽
- (3) 設計情報の排他制御
- (4) 設計情報の整合性

共通設計情報と個別設計情報についての比較を表1に示す。

大規模ソフトウェア開発では、設計情報の管理の容易さおよび設計情報のチェックのしやすさから分割統合モデルが適している。

5. 分割統合モデルの支援機能検討

分割統合モデルを実現するために必要な機能を以下に抽出する。

(1) 設計情報の分割

共通設計情報と詳細化に必要な情報をセットにして分配する機能。

(2) 設計情報の統合

① 設計情報を統合するとき、各個別情報から統合設計情報を作成する機能。

② 統合時の情報をチェックする機能。

以下の項目をチェックする。

- 共通情報が変更されていない
- サブシステム内で矛盾がない
- サブシステム間で設計情報が衝突していない

(3) 個別設計情報設計時の支援

各担当者に設計情報を分配してから、共通設計情報の変更作業に伴う個別設計情報の修正を支援する機能。

- 影響波及解析
共通設計情報が変更されたときの影響波及範囲を解析する機能。
- 編集履歴管理
設計情報の変更履歴を管理する機能。
- 設計情報のチェック
名標の衝突による修正の影響を少なくするため、各個別設計情報間での名標の衝突を起こさないようチェックする機能。

6. まとめ

今後は、技術的課題が多い分割統合モデルについて今回提案した支援機能を詳細化するとともに、グループ開発作業のモデルを明確にする。

表 1 作業モデルの比較

		共有	隠蔽	排他制御	整合性
単一モデル	共通設計情報	○	—	×	△
	個別設計情報	—	×	×	△
共有モデル	共通設計情報	○	—	要対処	○
	個別設計情報	—	要対処	要対処	○
分割統合モデル	共通設計情報	—	—	—	要対処
	個別設計情報	—	—	—	要対処