

構造化分析/設計手法の一評価

高橋健司 岡 敦子 山本修一郎 磯田定宏

NTTソフトウェア研究所

構造化分析/設計(SA/SD)手法の効果を評価するために、ドキュメントの理解容易性評価実験と要求分析/設計作業実験を実施した。比較対象として、テキスト形式ドキュメント手法を採用した。第1の実験では、要求仕様の側面別(データ構造、機能、データフロー)の質問に答える形式で、理解作業、誤り検出作業、影響解析作業を評価した。SA手法の場合には、データ構造とデータフローの正答率がより高い、関連情報がドキュメントの異なるページにある場合に理解作業の誤りが多いなどの結果が得られた。第2の実験では、SA手法を用いた場合、テキスト形式ドキュメントに比べ、同等以上の品質、生産性で、より詳細なドキュメントを作成できることを明らかにした。

A Study on the Effects of the Structured Analysis and Structured Design Methodology

Kenji Takahashi Atsuko Oka Shuichiro Yamamoto Sadahiro Isoda

Software Laboratories

Nippon Telegraph and Telephone corporation

Two experiments are carried out to evaluate effectiveness of the Structured Analysis and Structured Design (SA/SD) methodology in manipulating requirement specifications, and analyzing requirement and designing program modules. In the first experiment, subjects are questioned to perform the following three tasks: understanding, error-detection, and impact analysis. The experiment shows that subjects using the SA documents make higher scores concerning the data structure and data flow features and they are prone to fail to understand specifications when relevant constituents exist on different pages. The second experiment shows that subjects using the SA/SD documents produce more refined documents at higher quality and productivity.

1. はじめに

要求分析、設計段階における誤りが原因のプログラム・バグを除去するには、多大な工数を要する[1]。要求分析、設計における誤りの主要な原因として、要求仕様書の不備や把握しにくさがある。このような誤りの防止や検出に、構造化分析/設計(SA/SD)手法[2]など、要求仕様を把握しやすい図形表現で形式的に記述する手法が有効であることが提唱されている[3]。

中条ら[3]は、SA/SD手法を用いることにより、ソフトウェア開発で発生する誤りを削減できることを報告している。また、Cioch[4]やNosek[5]らは、ユーザが自分の要求が正しく記述されているかどうか確認するための手段として、SA手法で用いるデータフロー図とテキスト形式ドキュメントとを比較評価している。彼らは、SA手法の知識のないユーザにとっては、両ドキュメントによる要求仕様の把握状況に差がないことを明かにした。しかし、CiochやNosekらはSA手法の知識を持つ被験者に対するSA手法の効果には言及していない。

そこで、筆者らはSA手法の知識を持つ被験者を対象とし、SA手法ドキュメントの理解容易性を評価する実験および要求分析/設計作業におけるSA手法の効果を調べる実験を行った。本稿では、この実験手順および結果について報告する。

2. 理解容易性評価実験

2.1 実験内容

2.1.1 実験手順

本実験では、SA手法のドキュメント

を用いるグループとテキスト形式ドキュメントを用いるグループを被験者とした。SA手法の持つ3つの側面(データフロー、機能、データ構造)のいずれかに関する質問を用意した(表1)。被験者は、特定のシステムの要求仕様書に関する質問を与えられ、要求仕様理解作業、要求仕様の誤り検出作業、影響範囲解析作業を実施するように求められた。

質問の対象のシステムとして、4つの小規模(1~2KLOC)のトランザクションシステム(図書会計システム、銀行口座管理システム、医薬品管理システム、受注管理システム)を用いた。

		側面			作業合計
		データ構造	データフロー	機能	
作業	理解	30	30	62	122
	誤り検出	4	13	1	18
	影響範囲解析	5	3	8	16
側面合計		39	46	71	156

表1. 作業別/側面別の質問数

2.1.2 被験者

SA手法グループ

被験者は、3人の情報工学関係の大学院生であった。被験者は、実験の前に、3日間の、SA手法およびSoftDAの講習会を受講している。理解作業では、被験者に印刷されたデータフロー図およびデータ構造図が与えられた。誤り検出作業および影響解析作業では、印刷された図は与えられず、ワークステーション上でのみ操作可能であった。被験者は、SA手法に基づく統合化CASE環境SoftDA[6]を用いてデータフロー図およびデータ構造図を操作した。

テキスト形式ドキュメントグループ

被験者は、情報工学関係の学生3人(4年生1人、大学院生2人)であった。被験者は実験に関する講習を受けなかった。被験者には印刷されたテキスト形式のドキュメントが与えられた。ドキュメントは機能項目から構成されている。機能項目には、機能名、機能の説明、入出力データ名および入出力データの構造を説明する表が記述されている。記述形式が異なるが、記述内容はSA手法グループのものとはほぼ同じである(図1)。

両グループの被験者6人全ては、3年以上のプログラミング経験がある。また、予備的なアンケートおよびテストの結果、プログラム能力にはほぼ差がなかった。

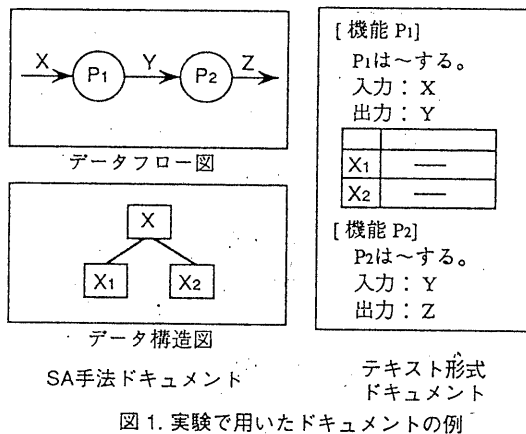


図1. 実験で用いたドキュメントの例

2.2 実験結果とその考察

2.2.1 SA手法に基づく要求仕様書の理解しやすさの分析

理解作業、誤り検出作業、影響範囲解析作業、の3作業を通じて、SA手法グループでは、データ構造とデータフローについての平均正答率が高かった(表2)。

データ構造についての平均正答率に関して、両グループの間には、統計的

に有意な差が認められなかった。しかし、データフローに関する質問に対する正答率では、テキスト形式ドキュメントグループ(50.7%)に比べ、SA手法グループ(74.6%)の方が有意に(有意水準5%)高かった(図2)。これは、図1に示すように、データフロー図では、データがどのプロセスからどのプロセスに流れるかを明示的に記述しているが、テキスト形式ドキュメントでは、データがどこからどこへ流れるかを知るためには、機能項目別に記述された入出力データを辿っていく必要があるからである。

機能に関しては、両グループの間で3つの作業に一貫した傾向は見られなかった。また、被験者による得点の分散が大きいため、作業および側面別の平均正答率にも、有意な差がなかった。

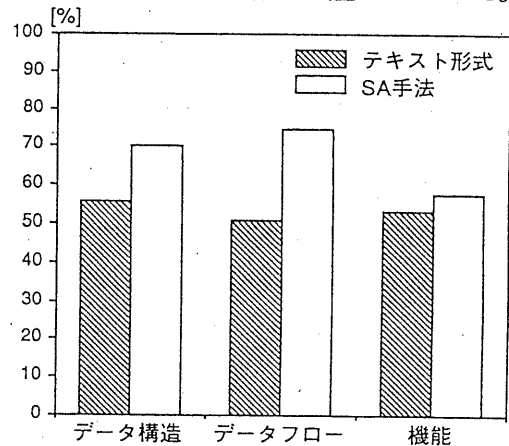


図2. 側面別の正答率

側面	データ構造	データフロー	機能	作業別平均
理解	70.0 / 80.0	60.0 / 86.7	52.7 / 57.5	58.5 / 70.2
誤り検出	8.8 / 25.0	30.8 / 56.4	0.0 / 33.3	24.1 / 48.1
影響範囲解析	6.7 / 46.7	22.2 / 33.3	66.7 / 62.5	43.8 / 52.1
側面別平均	55.6 / 70.1	50.7 / 74.6	53.1 / 57.7	

表2. 作業別/側面別の正答率 [単位: %]
(凡例: テキスト形式 / SA手法)

2.2.2 SA手法における誤りの分析

理解作業に関する質問に対する誤答の原因を分析した(誤り検出作業および影響範囲解析作業については、質問数が少なすぎたので、同様の分析ができなかった)。分析の結果、関連する情報が要求仕様書の複数のページ上に存在している場合に、被験者は誤りを犯しやすいことがわかった(図3)。これは、たとえば"Aに関連するXを答えよ"という問題において、AとXの間にYが介在しているとき、または、"Xが複数個ありドキュメント上の様々な場所に現れたとき"のような場合に起こる(図4)。以降では、このような場合を、"関連情報のページ分散"と呼ぶ。

テキスト形式ドキュメントグループでは、関連情報のページ分散がある場合とない場合の平均正答率の差は有意でなかった。しかし、SA手法グループでは、関連情報のページ分散がある場合の平均正答率(43.6%)は、ない場合(77.4%)に比べ、有意に低かった(有意水準5%)。これは、SA手法の持つ「データフロー図の異なる階層を異なるページに記述し、データフロー図に現れるデータの構造をデータ構造図に記述する」という特徴によると考えられる。関連情報がデータフロー図の別の階層に跨ったり、データフロー図とデータ構造図とに跨ると、その関連情報を発見しにくい。一方、テキスト形式ドキュメントは、図1に示したように、階層的に記述されておらず、関連情報が近傍に記述されているため、このような問題は比較的生じにくい。

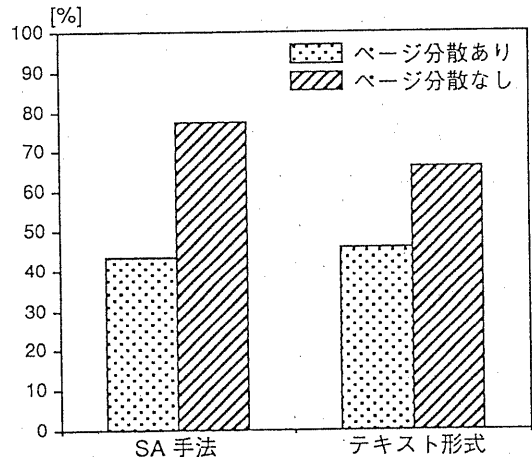


図3. 関連情報のページ分散の有無と正答率

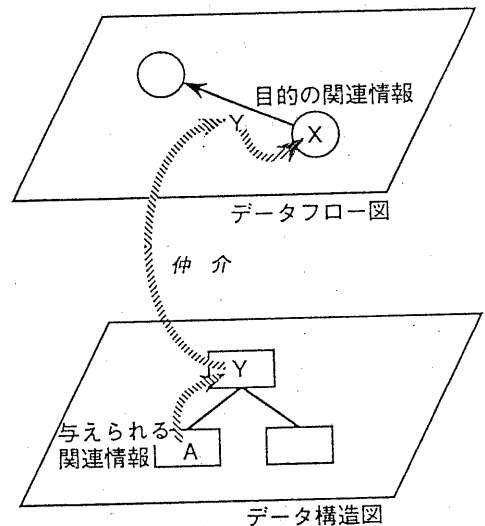


図4. 関連情報のページ分散の例

この実験結果は、SA手法を支援するCASE (Computer Aided Software Engineering) 環境では、関連情報の把握の困難さを低減するために、データフロー図の全体構造や、データフロー図やデータ構造図などの異なるドキュメント間の関係を把握しやすい形式で表示する機能が必要なことを示している。さらに、関連情報の検索機能や影響波及箇所解析/修正機能も必要である。

3. 要求分析/設計作業実験

3.1 実験内容

3.1.1 実験手順

被験者は、理解容易性作業実験と同じ、SA手法グループとテキスト形式グループの2グループである。この2グループが、日本語で記述された同じ要求仕様を与えられ、それをもとに要求分析作業およびモジュール設計作業を次の順序で実施した。

- (1) 要求分析 (要求仕様から要求仕様書を作成)
- (2) 要求仕様書レビュー
- (3) 要求仕様書修正
- (4) モジュール設計 (要求仕様書からモジュール設計書を作成)
- (5) モジュール設計書レビュー
- (6) モジュール設計書修正

実験対象システムは、小規模(6 KLOC)のトランザクションシステム(会議管理システム)であった。各グループのメンバは分担して、グループ全体で一つのシステムを作成した。

3.1.2 被験者

SA手法グループ

SA手法グループは、統合化CASE環境SoftDAを用いて、ドキュメントを作成した。要求分析作業では、SA手法に従いDFDおよびDSDを作成した。モジュール設計作業では、SoftDAのモジュール構造図(SC)自動生成機能を用いてDFDからSCを生成した。さらに、構造化分析手法に従い、そのSCを修正することによりSCを完成させた。要求分析および設計レビュー作業では印刷したドキュメントをメンバ全員で読み合わせた。

テキスト形式グループ

テキスト形式グループは、ワープロを用いて、ドキュメントを作成した。要求分析作業では、理解容易性評価実験で説明したドキュメントと同形式のテキスト形式ドキュメントを作成した。モジュール設計作業では、モジュール関連図を作成した。モジュール関連図は、モジュール間の呼び出し関係を図式化したもので、SCから制御情報およびパラメータを除いたものに相当する。要求分析および設計レビュー作業では、印刷したドキュメントをメンバ全員で読み合わせた。

3.2 結果と考察

3.2.1 要求分析作業結果と考察

SA手法グループは、以下に述べるようにテキスト形式グループに比べて、同等以上の品質、生産性で、より詳細なドキュメントを作成できた(表3)。

	SA手法	テキスト形式
プロセス数	50	17
要求分析工数 [人・時間]	78	72
プロセス当りの 未発見誤修正バグ数	0.5	0.7
提供機能数 (総数 33)	22 $\begin{pmatrix} 11 \\ 11 \end{pmatrix}$	19 $\begin{pmatrix} 11 \\ 8 \end{pmatrix}$

表3. 要求仕様書の比較

テキスト形式ドキュメントの機能項目はDFDのプロセスに対応しているもので、ここでは両者を同等のものとして、ドキュメントの詳細化レベルや量を比較する。

テキスト形式グループは、全ての機能項目を2レベルまでしか詳細化していないが、SA手法グループは、DFDを平

均3.7レベルまで詳細化している。このため、SA手法ドキュメントでは、プロセス数が50、テキスト形式ドキュメントでは、機能項目数が17となり、ドキュメントの記述量の差が大きくなった。一方、両グループの分析工数は、ほぼ同じであった。したがって、SA手法グループは、テキスト形式グループに比べ、同等以上の生産性であると言える。

実験対象のようなトランザクションシステムでは、機能の上位レベルでは、外部データ(DFDにおけるデータストア)を介して機能間のデータ受渡しを行い、下位レベルでは、外部データを介さずに直接機能間でデータを受け渡すことが多い。今回の実験対象システムもそのような構成であった。しかし、テキスト形式ドキュメントでは、下位レベルにおける機能間の直接のデータの受渡しを記述するのが困難であるため、下位レベルは非形式的な文章で記述するか、あるいは全く記述しなかった。この結果、詳細化のレベルが低かったと考えらる。

要求仕様書レビューで未発見のバグおよび発見したが正しく修正できなかったバグは、SA手法では、33個あった。ただし、制御フローをデータフローとして記述したバグなどのDFDの形式上のバグ8個を除く、実質的な未発見/誤修正バグは、25個であった。テキスト形式グループでは、SA手法のような形式的なバグはなく、実質的な未発見/誤修正バグは12個であった。したがって、1プロセス(もしくは機能項目)あたりのバグ数は、SA手法0.5、テキスト形式0.7となり、SA手法ドキュメントは、テ

キスト形式ドキュメントに比べ、同等以上の品質であった。

被験者に与えた要求仕様に記述された会議管理システムとして必須の機能は11個、オプションな機能は24個ある。一方システムがユーザに提供する機能は、SA手法ドキュメントで22個、テキスト形式ドキュメントで19個であり、これらのうちで両者に共通する機能は14個あった。必須機能は、SA手法およびテキスト形式ドキュメントのいずれでも全て記述されており、両者に共通しない機能は、全てオプションな機能であった。したがって、両グループのドキュメントがカバーする機能には、本質的な差はないと言える。

テキスト形式ドキュメントには見られなかったSA手法ドキュメントの形式的な誤りとして、制御フローをデータフローとして記述してしまうバグ、モジュールとして記述する方が適切である内容(データベースへのアクセスなど)をプロセスとして記述してしまうバグ、プロセス仕様として記述する方が適切である内容(処理アルゴリズムなど)をプロセスとして記述してしまうバグがあった。これらのバグは、被験者がDFDの記述に不慣れであったため、生じたと考えらる。このようなバグを検出・修正する機能やプロセスの分割を支援・自動化する機能がCASEにあれば、SA手法の初心者にも有効であると考えらる。

3.2.2 設計作業の結果と考察

SA手法グループは、以下に述べるようにテキスト形式グループに比べて、

同等以上の品質、生産性で、より詳細な情報を含むドキュメントを作成できた(表4)。

	SA手法	テキスト形式
モジュール数	108	69
設計工数 [人・時]	42	33
モジュール当りの 設計工数 [個/人・時]	0.13	0.16
モジュール当りの モジュール関連の 未発見、誤修正バグ	0.074	0.116

表4. モジュール設計書の比較

SA手法グループでは、108個のモジュールを、テキスト形式グループでは、69個のモジュールを設計した。SA手法ドキュメントは、パラメータおよび条件分岐などの制御情報も記述されており、より多くの設計情報を含んでいる。このため、ドキュメント作成により多くの工数がかかっている。しかし、1モジュール当りの作成工数[人・時間]は、SA手法グループで0.13で、テキスト形式グループで0.16であることから、SA手法グループの生産性は、テキスト形式グループに比べ、同等以上であったと言える。

SA手法ドキュメントのモジュールに関する未発見/誤修正バグ数は、1モジュール当たり0.074個であり、テキスト形式ドキュメントの未発見/誤修正バグ数は1モジュール当たり0.116個であった。すなわち、SA手法ドキュメントは、テキスト形式ドキュメントと同等以上の品質であった。

テキスト形式グループでは、要求仕様書からモジュール関連図を作成する際に、要求仕様書に明示的に機能項目

として記述された機能の他に、前節で述べたような非形式的に記述された詳細レベルの機能もモジュールとして手作業で追加した。また、外部データの入出力モジュールを手作業で追加した。一方、SA手法グループでは、DFDからSCを自動的に変換した。このとき、外部データの入出力モジュールも自動的に追加された。被験者は、新しいモジュールを追加することはなかった。

テキスト形式グループでは、非形式的に記述された機能を手作業でモジュール化するため、要求仕様書に記述されている処理がモジュール関連図に記述されていない、要求仕様書の機能間の関係が正しくモジュール関連図に反映されていない、テーブルからの入力モジュールが記述されていないなどのバグが発生した。

SA手法グループでは、SoftDAを用いてDFDからSCへ自動的に変換しているため、テキスト形式グループのように要求仕様書の内容がモジュール仕様書にない、もしくは誤って反映されているというバグは、発生していない。しかし、DFDからSCに変換する際、外部データとの間のデータフローごとに、外部データの入出力モジュールを機械的に生成しているため、冗長なモジュールがあるというバグが発生している。DFDからのSC自動生成機能には、外部データにアクセスするプロセスの処理内容などを考慮した入出力モジュールの追加が必要である。

4. まとめ

構造化分析(SA)手法ドキュメントとテキスト形式ドキュメントを対象に、理解容易性評価実験と、要求分析/設計作業実験を実施した。理解容易性の評価実験では、要求仕様の側面別(データ構造、機能、データフロー)の質問に答える形式で、理解作業、誤り検出作業、影響解析作業を実施した。その結果、SA手法では、データ構造とデータフローの正答率がより高い、関連情報がドキュメントの異なるページにある場合に理解作業の誤りが多いことなどを明かにした。一方、要求分析/設計作業実験の結果、SA手法を用いた場合、テキスト形式ドキュメント手法に比べ、同等以上の品質、生産性で、より詳細なドキュメントを作成できることを明かにした。

trans. on Software Engineering, pp.1372-1375, vol. 14, no.9, 1988.

[6] S.Isoda et al, "Integrated CASE System: SoftDA", NTT review, pp.52-61, vol.2, no.2, 1990.

謝辞

実験の企画、実施にご尽力頂いた三浦和則様に深謝します。実験に協力して頂いた被験者の皆様に感謝します。

【参考文献】

- [1] A.Davis, "Software Requirements - Analysis & Specification - ", Prentice Hall, 1990.
- [2] E.Yourdon, "Modern Structured Analysis", Prentice-Hall, 1989.
- [3] T.Nakajo, "A case history analysis of software error case-effect relationship", IEEE trans. on Software Engineering, pp.830-837, vol. 17, no.8, 1991.
- [4] F.Cioch, "Measuring Software Misinterpretation", J.System Software, pp.85-95. no.14, 1991.
- [5] J.Nosek et al, "User validation of information system requirements: some empirical results", IEEE