

構造化分析 (SA) / 構造化設計 (SD) ツール

棕本 龍雄、山本 倫之、久保 哲也

情報処理振興事業協会 シグマシステム開発本部

1970年代に提案された構造化分析/構造化設計手法はソフトウェアの上流工程を支援する手法として主に米国でその効果が確認され普及してきた。特にリアルタイム支援の手法が考案され、この手法を支援するツールが最近の高機能のワークステーション上に実現されるに至ってその普及は加速度的になっている。またソフトウェア開発工程の全般を支援するCASEの研究、開発が活発に行われており、構造化分析/構造化設計手法が中核をなす手法として採用されている。

一方日本の現状を見て見ると、この手法そのものがソフトウェア開発現場にほとんど普及していない。その原因は色々考えられるが、最大の理由はこの手法を実現した安価で日本語化されたツールが無く、気軽に試験的に使って見る機会が得られないためであると思われる。このような状況を鑑み、情報処理振興事業協会が進めているソフトウェア生産工業化システム構築事業であるΣ(シグマ)プロジェクトにおいて、構造化分析/構造化設計手法の普及を最大の目的として、手軽に使える、簡便なツールとして開発を行った。

本報告では、手法の簡単な紹介、リアルタイム支援機能およびツールの機能について報告する。

Structured Analysis(SA)/Structured Design(SD) tool

Tatsuo MUKUMOTO, Michiyuki YAMAMOTO, Tetsuya KUBO

SIGMA Project, Information-technology Promotion Agency

SF Akihabara Sanwa Toyo BLDG. 3-16-8, Sotokanda, Chiyoda-ku, Tokyo, 101 Japan

The methods of structured analysis(SA) and structured design(SD) were developed in 1970's. SA/SD methods have been available for more than a decade and have much effect on improving the practice of software development in the U.S.A. The addition of real-time analysis and the tools with SA/SD methods on many powerful workstations, have promoted popularization of SA/SD methods.

Computer-aided software engineering(CASE), which supports the entire software life cycle, has recently been studied and developed. SA/SD methods are major technology in many CASE environments.

Even now SA/SD methods are not popular at software development fields in Japan. The matter that we cannot get a low-priced SA/SD tool and we have little opportunity for using the tool, has caused obstruction to popularize the SA/SD methods. It is most important that the tools are available to many software engineers. Then we have developed a SA/SD tool for the purpose of popularizing the SA/SD methods in SIGMA project, Information-technology Promotion Agency(IPA). In this report, we explain the methods of SA/SD, the function of real-time support and the other functions of the tool.

1. 背景

コンピュータシステムの需要はますます高まり、また高度な機能が要求されている。しかし、ハードウェアの進歩が非常に著しいのに比べて、ソフトウェアの進歩としては、依然として生産性や品質の向上に対する要望が指摘されている。ソフトウェアの開発環境はどちらかという、下流工程において進歩してきているといえる。

一方、上流工程としてのソフトウェア開発環境はこれまで各方面で試作、試用が行われているにもかかわらず、いまだ決定的なものは出現していないといっても過言ではない。

このような状況のなかで、「構造化分析 (Structured Analysis) 手法」に基づいて階層構造をもつデータフロー図等による要求定義分析を行い、その結果から「構造化設計 (Structured Design) 手法」に基づいて設計を行う手法が、主に米国で効果が認められている。「構造化分析手法」そのものは、すでに70年代に一旦広く普及したにもかかわらず、80年前半にはあまり使われなくなった。その理由は、リアルタイム制御を必要とするリアルタイムシステムに適用しにくいためであったが、最近の米国で普及しているツールはこの点を改良し、データフロー図と制御フロー図を組合せてリアルタイムシステムに応用している。また、この手法を基に高機能なワークステーション上で支援ツール化されており、さらにこの支援ツールは、上流工程から下流工程へのつながりを含めたCASEツールの中核になってきている。

プロジェクトとしては、上流工程の一つの手法として米国で効果が認められている構造化分析/構造化設計手法を手軽にだれでも使用できる簡便なツールの提供を行うために開発した。

2. 目的

本報告の構造化分析支援ツールは、この構造化分析/構造化設計手法に基づいて実現したツールである。開発対象システムに要求される機能をデータの流に着眼して、ディスプレイ画面上にデータフロー図を描いて、システム分析、設計を支援する。すなわちデータフロー図を段階的に詳細化し、階層化された要求仕様仕様書を作成することを目的としている。

また、これらの仕様書を作成することによって次の効果が得られる。

- 1) 要求分析から設計まで一貫した設計作業を可能とする。
- 2) 仕様書作成時に、仕様書間の論理的矛盾を検出することができる。
- 3) 作成した仕様書の再利用を可能とする。

以上のことから、仕様書の質が向上し、ユーザの要求を正しく理解でき、仕様書の信頼性が向上する。

本ツールでは、まず構造化分析/構造化設計手法の基本的な機能を全て実現した。また、リアルタイムに関連した支援も行っているが、更に米国の動向を調査しながら拡張していく予定である。

3. 構造化分析支援ツールの概要

本ツールはSA手法を支援するSA支援ツールとSD手法を支援するSD支援ツールおよびリアルタイムを支援するツールから構成される。ツール構成図を図1に示す。

3.1. SA手法

構造化分析手法は、Tom DeMarcoなど米国のソフトウェア技術者によって1970年代後半に提唱された要求分析手法の総称である。主としてシステム開発の要求分析から基本設計の工程で使用する。構造化分析支援ではデータの流に着眼しながら段階的に図形を表現し、システム全体を1つのモデルとして対象システムに要求される機能を明確にしていくものである。この手法ではDFD (データフロー図)、DD (データディクショナリ)、MS (ミニスペック) と呼ぶ仕様書に体系化されている。図2にDFD、図3にDD、図4にMSの例を示す。

DFDにおいてはデータを処理するプロセス (円で表現)、データの流れを示すデータフロー (矢線で表現)、データを格納するデータストア (2本の並行線で表現)、そしてデータの源であるデータ源泉 (四角形で表現) によってデータが加工されていく流れを示し、要求内容を表現する。

| SA | | | | | SD | | | | |
|-------------|---|---|---|---|--------|---|---|--|--|
| | | | | | リアルタイム | | | | |
| D | D | M | S | D | S | S | M | | |
| F | D | S | T | F | C | C | D | | |
| D | 編 | 編 | D | D | へ | 編 | S | | |
| 編 | 業 | 業 | 編 | の | の | 業 | 編 | | |
| 業 | | | 業 | 拡 | 変 | | 業 | | |
| | | | | 張 | 換 | | | | |
| 検 証 | | | | | | | | | |
| デ ー タ ベ ー ス | | | | | | | | | |

- SA : 構造化分析 (Structured Analysis)
- SD : 構造化設計 (Structured Design)
- DFD : データフロー図 (Data Flow Diagram)
- DD : データ辞書 (Data Dictionary)
- MS : ミニスペック (Mini Spec)
- SC : 構造図 (Structured Chart)
- STD : 状態遷移図 (State Transition Diagram)
- MDS : モジュールスペック (Module Spec)

図1 ツール構成図

最上位層のDFDを特に概観図と呼び、システムの目的と外部環境およびそれらからの入出力データが明示される。データフローには固有のデータフロー名が与えられ、その内容がDDに登録される。DDにおいては複数のDFDで使用されるデータを、辞書のように集中的に定義する。

MSにおいてはDFD中のプロセスを更に詳細なDFDにしていき、これ以上分解できないブレイクダウンされた最下層のDFD上のプロセス一つに対して、プロセスの処理内容を記述する。

3. 2. SA支援機能

(1) DFD編集機能

ディスプレイ画面上にデータフロー図を作成編集する機能である。マウスやキーボードを使って容易に図形と付随する文字を設定しデータフロー図を作成することができる。作成済みのデータフローは、図形(シンボル)や文字単位で複写、移動、追加、削除等の機能により編集できる。

- ・図形(シンボル)設定機能(図形としてはデータの流れ(直線/円弧)、プロセス(円)、源泉/吸収(矩形)、ファイル(上下線)である)

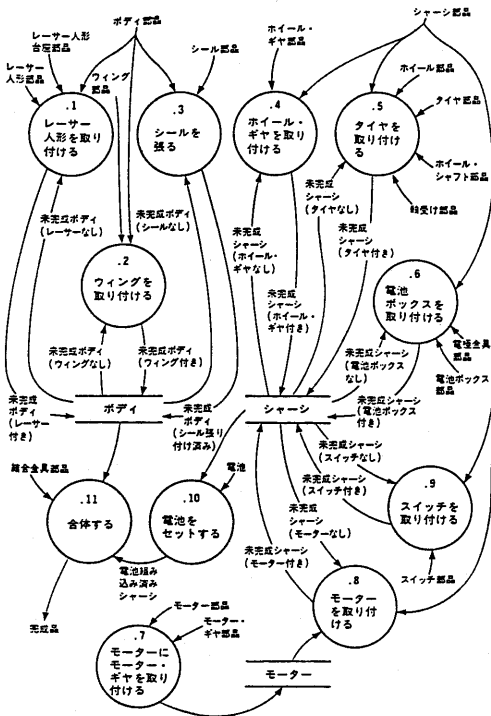


図2 DFDの例

- ・プロセス番号の自動付与機能
- ・文字設定機能
- ・シンボル間のリンク機能
- ・図形編集機能
- ・階層構造DFDの操作機能
- ・DFD自身の検証
- ・プリンタへの出力機能

パーツ・キット = ボディ・キット +
 シャーシ・キット +
 モーター・キット +
 電池ボックス・キット +
 シール部品 +
 車体結合金具部品

電池部品 = 2 {乾電池} 2 /* モーターを駆動する電源 */

ボディ・キット = ボディ部品 +
 ウィング部品 +
 レーザー人形部品 +
 レーザー人形台座部品

シャーシ・キット = シャーシ部品 +
 ホイール・ギヤ部品 +
 タイヤ部品 +
 ホイール・シャフト部品 +
 軸受け部品 +
 ホイール部品 +
 スイッチ部品

ホイール部品 = 2 {前輪用ホイール部品} 2 +
 2 {後輪用ホイール部品} 2

スイッチ部品 = /* シャーシに取り付けるスイッチの金具 */

前輪用ホイール部品 = /* 前輪用の小径のプラスチック製
 ホイール */

後輪用ホイール部品 = /* 後輪用の大径のプラスチック製
 ホイール */

図3 DDの例

MINISPEC レーザー人形を取り付ける [1. 1]
 /* 人形の顔が台座の前方向と一致するように */
 レーザー = レーザー人形 + レーザー人形台座部品
 未完成ボディ = [ボディ部品 | 未完成ボディ] +
 レーザー

ENDSPEC

MINISPEC ウィングを取り付ける [1. 2]
 未完成ボディ = [ボディ部品 | 未完成ボディ] +
 レーザー

ENDSPEC

MINISPEC シールを張る [1. 3]
 FOR EACH SECTION
 シール片 = シール部品 [SECTION]
 未完成ボディ = [ボディ部品 | 未完成ボディ] +
 シール片

ENDFOR

MINISPEC 電池をセットする [1. 10]
 /* プラス極とマイナス極を間違わないようにセットする */
 電池組み込み済みシャーシ = シャーシ + 電池
 WHILE (モーター != 動作)
 スイッチを調整する

ENDWILE

ENDSPEC

MINISPEC 合体する [1. 11]
 完成品 = ボディ + シャーシ

ENDSPEC

図4 MSの例

(2) DD編集機能

DFD編集機能により設定したデータフローからデータ・ディクショナリ(DD)を作成するための、データの名前、構造等を定義するテキストエディタ機能である。

- ・ファイル処理(テキスト編集)機能
- ・未定義、二重定義データの検証機能
- ・データのクロスリファレンス作成機能

(3) MS編集機能

MSは、DFDにおけるプロセスの処理内容をDDに定義してあるデータを使用し、構造化された疑似言語で記述したものである。ミニスペック編集機能はこの仕様書を効率良く編集するための構文エディット機能である。

- ・ファイル処理(テキスト編集)機能
- ・データのクロスリファレンス作成機能
- ・スペック構造の検証機能
- ・予約語処理機能

(4) 相互検証機能

DFD, DD, MSの各仕様書間の整合性を検証する。次のような仕様書間の整合性を検証する。

3.3. SD手法

SA手法の仕様書を中心にDFDを構造図に変換する手法を定義し、分析だけでなく構造化設計まで一連の体系的手法としてまとめたものが構造化設計手法である。

3.4. SD支援機能

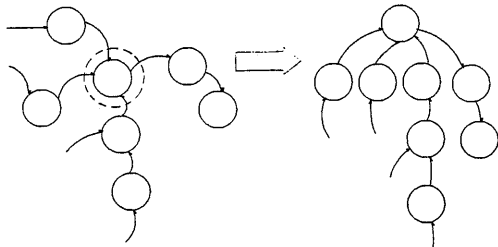


図5 DFDから構造図への変換概要

(1) SA-SD変換

DFDから階層的な構造図に変換する機能である。DFDはデータに、構造図はソフトウェアで言うモジュールに着目して対象となるシステムを表現している。このことから同じ図になることはことはないが原則的には同一形状を持つはずである。DFDから構造図への変換の概略を次に示す(図5を参照)。

- ブレイクダウンされた最下層のDFD中のプロセスのなかで、処理の中心となるプロセスまたはプロセスグループを見つける。
- 見つけたプロセスをつまみ上げると各プロセスは糸でつながったようにぶらさがりようになる。
- このときに出来る形状が構造図の形状になる。

このように入出力の関連から機能階層を作成するが、次のようなことを人間が判断しながらより良い構造図へと改良する。

- 実行効率
入出力の関連から構造図へ変換していることから、機能面の効率を考慮する必要がある。

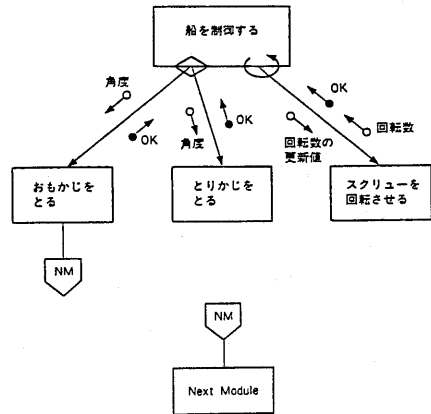


図6 構造図

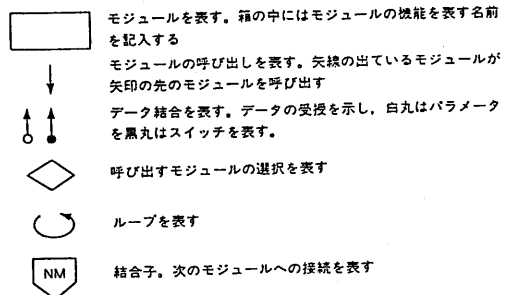


図7 構造図の記述ルール

b) 機能の整理

各モジュール間の独立性、まとまり具合、モジュールの大きさなどを検討する必要がある。

c) 部品化対応

実際のプログラミングでは、部品化されたモジュールを使用することがある。ところが、変換されたものは部品を意識していないため、部品を使用することを考慮する必要がある。

(2) 構造図編集機能

構造図の例を図6に示す。また、構造図の記述ルールを図7に示す。4. 2の(1)で記述したように、変換機能で作成した構造図は完全なものではない。そこで、構造図を修正する必要性がでてくる。本機能は構造図を効率良く編集するための図形編集機能である。

- ・ シンボルの設定, 移動, 削除
- ・ プリントへの出力機能

(3) モジュール仕様書編集機能

図8にモジュール仕様書の例を示す。モジュール仕様書は詳細設計時に作成する一般的なモジュール仕様書である。構造図中の1つ1つのモジュールに対してモジュール仕様書を構造化言語を使用して作成する。このエディタでは構造化言語の文法チェックの機能を提供する。

3. 5. リアルタイム支援

従来手法ではプロセスの実行順序、タイミングについては記述できない。リアルタイム支援ではこれらを支援するために、DFDの中に制御の働きかけを駆動するコントロールフローとその内容を記述するコントロールプロセスを追加している。

MODSPEC キーボード入力

IF (1文字入力 != EOF)
CALL エコーバックする (入力データ : 結果)

ENDIF

ENDSPEC

(記述ルール)

- ・ MODSPECの行が見出し行であり、その後にはモジュール名を記入する。
- ・ MODSPECの次の行からENDSPECの上の行までにモジュールのロジックを記述する。この例では、日本語と制御構造を示す予約語を組み合わせた構造化言語を使っている。CALLは呼び出しを表す予約語であり、条件式、パラメータは()で囲んでいる。ロジックの記述には構造化言語以外の表現方法を用いてもよい。

図8 モジュール仕様書の例

(1) コントロールプロセス/コントロールフロー

DFD上でプロセスに同期して動作するコントロールプロセス(点線の円で表現)や制御信号の流れを示すコントロールフロー(点線の矢印)を作成編集することができる。作成するための機能としてはDFDエディタと同じである。

コントロールプロセスやコントロールフローを使用した、簡単なDFDを図9に示す。

「全体を制御する」というコントロールプロセスは3つのデータプロセス(コントロールプロセスと区別してデータプロセスと呼ぶ)と同期して動作する。また、6つの点線の矢印はコントロールフローで制御の信号を示している。これらのコントロールプロセスやコントロールフローの処理の詳細はコントロールプロセスに対応したミニスペックで記述する。

(2) 状態遷移図(STD)エディタ

STD(ステート・トランジション・ダイアグラム)はタイミングや状況の変化の状態を図示したもので

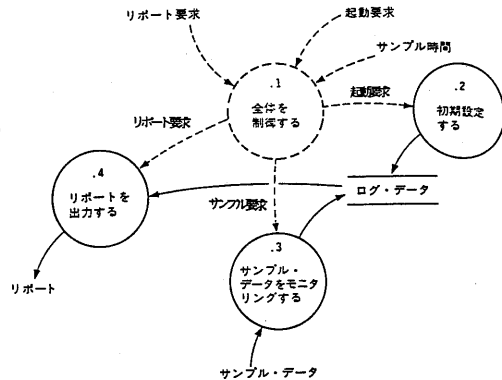


図9 リアルタイム用DFDの例

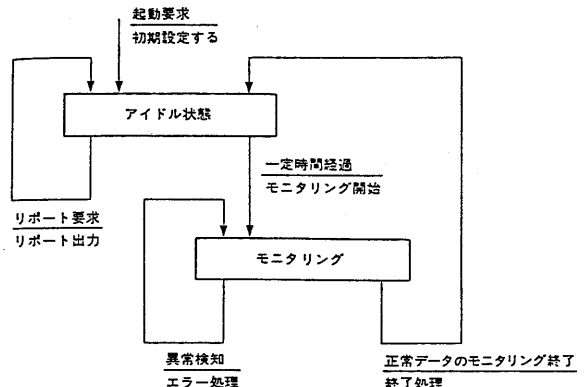


図10 STDの例

ある。図9のコントロールプロセスに対応するミニスベックは図的表現ではないため、これを図的に表現したものがSTDである。図10に図9のDFDに対応する状態遷移図を示す。

図10において四角形がステートと呼ばれるもので、ある状態を示している。各ステートの右下には条件と行動が分離子(直線)に上下に区切られて記述されている。分離子の上には状態が変化するときの条件を記述し、下にそのときにとる行動を記述する。本機能はSTDを効率良く編集するための図形編集機能である。

- ・ステートを表す状態シンボルとステートの移動を表すフロー図、条件、動作等の設定機能
- ・プリンタへの出力機能

(3) 検証機能

リアルタイムにおける検証機能としては、次のものがある。

(a) STDの検証機能

STD自身の記述方法の誤りを検証する。

(b) STDとDFDの相互検証機能

STDとDFDの相互の整合性を検証する。例えば、DFD上のコントロールプロセスやコントロールフローとSTD上の表現に矛盾がないかなどの整合性を検証する。

検証には、DFD上のコントロールプロセスに対応するミニスベックを参照することによって行う。

(4) STD一状態遷移表変換機能

STDと状態遷移表は共に、状態の遷移を表現している。STDは図を使用して表現し、状態遷移表は表形式に表現したものである。用途によってこれらの仕様書を使い分けるために、これらの仕様書の変換機能が必要である。

4. リアルタイム支援のまとめ

現状ではDFDにおけるコントロールフローで制御信号を表現し、コントロールプロセスでデータプロセスの順序制御を行う。これらによりデータプロセスの活性化の流れ、メカニズムを記述できる。STDは条件と行動、それらに伴う状態遷移を図式表現することで制御のタイミングを明確に記述できた。

一方、米国においてはハトレー(Hatley)氏のSA手法によってリアルタイムの支援が行われてい

る。そこでこれらと本ツールの機能とどのような関連があるかを示す。(表1)

ハトレー氏のリアルタイム手法におけるデータ概観図は従来の概観図と同じものであり、制御概観図と区別するためにデータ概観図という名称を使用している。制御概観図は従来の手法にはないものであるが、リアルタイムの機能で記述したコントロールプロセス、コントロールフローを使用した概観図が制御概観図である。制御フロー図も同じくDFDにコントロールプロセス、コントロールフローを使用したDFDが制御フロー図である。

プロセススベックはコントロールプロセスの処理内容を記述したものであり、MSと同じものである。

制御スベックはタイミングや行動について記述したものであり、STDで記述可能である。

タイミングスベックはタイミングに関する関係を記述したものであり、信号の属性や入力と出力信号の関係が記述される。DDで記述可能である。

要求辞書はデータ信号と制御信号の両方に関する関係を記述したものであり、DDで記述可能である。

以上のようにハトレー氏のSA手法にこだわらなければ、リアルタイムシステムの開発には適用可能であると判断できる。現状の日本のSA/SD手法の普及状態を考慮しつつ、今後の機能向上を検討していく必要がある。

表1. ハトレー氏のリアルタイムSA手法との比較

| リアルタイムSA手法 | 本ツールのSA手法 |
|------------|-----------|
| データ概観図 | 概観図 |
| 制御概観図 | 概観図 |
| DFD | DFD |
| 制御フロー図 | DFD |
| プロセススベック | MS |
| 制御スベック | STD |
| タイミングスベック | DD |
| 要求辞書 | DD |

5. 今後の課題

1) 検証機能の強化

SA/S D手法のツールにおいては単なるお絵書きツールではなくなるように検証機能を強化していく必要がある。今後の課題としてはSTD自身の検証やSTDとDFDとの相互検証機能をサポートしていく。

2) リアルタイム支援の強化

制御スペックでは対象システムの性質によりいろいろな表記法が要求される。状態遷移表(STT)、デジションテーブル(DT)をサポートしていく。

6. 最後に

本報告に際し、技術的な協力をいただいたヴェストソフトウェア(株)松崎純一氏に本紙面を借りてお礼を申し上げる。また、本報告にて使用した各種の図は「構造化分析ワークベンチ」(参考文献5.を参照)より引用した。

参考文献:

1. 構造化分析とシステム仕様
Tom DeMarco著 高梨、黒田監訳
日経マグローヒル社
2. 構造化手法によるソフトウェア開発
Edward Yourdon著 黒田、渡部訳
日経マグローヒル社
3. ハトレー氏のリアルタイムSA手法
bit vol. 19、No. 14
Derek J. Hatley/立田訳
共立出版
4. ワード氏のリアルタイムSA手法
bit vol. 20 No. 5
Paul T. Ward/立田訳
共立出版
5. 構造化分析ワークベンチ
松崎 純一著
日経マグローヒル社