

ソフトウェア設計過程における思考軸について

佐藤 隆、島 健一、門田充弘

ATR 通信システム研究所

ソフトウェア設計過程における設計者の思考過程を理解する上で有用な概念として思考軸を提案する。思考軸は、設計者のある観点に沿った思考順番を表わすものである。思考軸の概念を用いて、各種実験データを整理した結果を示し、思考軸導入の有効性を示す。

Thinking Axis in Software Design Process

SATO Takashi, Ken-ichi SHIMA, Michihiro MONDEN

ATR Communication Systems Research Laboratories

Communication Software department

Twin 21 Bldg. MID tower, 2-1-6 Shiromi Higashi-ku, Osaka, 540 Japan

This article proposes the concept of 'thinking axis' which is useful for analyzing and understanding designer's thinking process in software design process. A thinking axis represents thinking order from a designer's certain viewpoint. Applicability of the concept is shown by demonstrating several design process description examples which are arranged introducing the concept of thinking axis.

1 はじめに

ソフトウェアの設計作業を計算機により知的に支援するためには、人間の思考過程を明らかにし、どのような支援が可能であるかを検討する必要がある。ここで、「知的に」とは、人間の思考の流れに沿って、設計作業を支援するという意味である。

ソフトウェア設計過程の分析は、Kant[6]、Adelson[1]、Curtis[2]、Guindon[4]などが行なっている。ここでは、通信ソフトウェア開発過程において思考結果(知識)の表現方法について主に注目し、その整理法について議論する。

本稿では、思考軸の定義、性質についていくつかの仮定を設定した。この思考軸により、設計者の頭の中に形成されていく知識の分析が可能となり、知識がどのような仕組みで形成されるか、また、形成された知識の使用法についての指針が得られると考える。

思考軸を検証するために、ある通信ソフトウェア設計記録に対する分析を思考軸をもとに行なった結果について述べる。この分析により、思考軸に関して、以下のことが明らかになった。

- (1) 思考過程が思考軸に沿って行われる。
- (2) 思考軸の取り替えにより、設計が進む。
- (3) 思考軸の変更理由と移行先。

2 ソフトウェア設計作業における思考軸の役割

ソフトウェアの設計作業モデルを図1のように設定する。このモデルでは、設計者の頭の中にある知識を以下のように分類する。

方法知識 — 設計方法に関する知識

ソフトウェアを設計する作業方法、および、それらの順序に関する知識である。たとえば、SDL図の作成方法や作成時期、プロトコルの検証方法や実施時期、といったものがある。

領域知識 — 設計対象領域に関する知識

設計対象の属している領域の知識である。たとえば、電話機用ソフトウェアの領域知識として「電話をかける手順」、「電話番号の桁数」、「交換プロトコル」などがある。

対象知識 — 設計対象に関する知識

設計したものの具体的なプロトコルや、(オブジェクト指向で作成するならば)構成オブジェクトとメッセージ、などの知識である。機能や性能などの設計内容だけでなく、設計時に立てた計画や実施した手順なども含んでいる。設計作業により変化するのは、

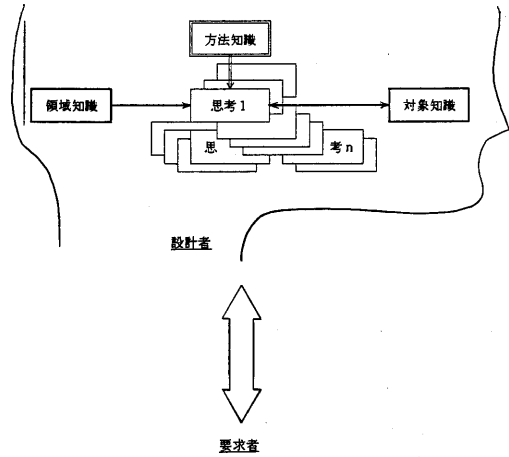


図1: 設計作業モデル

この対象知識である。

ソフトウェアの設計は、ユーザなどの外界からくる要求と、設計者自身のもっている3種の知識とのインタラクションにより行われる。この時、設計者は設計過程の様々な場面で、関連のある知識をある観点(たとえば処理順や、性質の違いなど)に従って順序付けて整理しながら設計していると考えられる。ここでは、その整理された設計対象の項目の並びを「思考軸」と呼ぶ。

たとえば、SDL図作成の場合、作業手順の時間的流れによる思考軸、タスク発生の流れによる思考軸、検索結果の個数による思考軸などがあることがわかり、設計過程の作業を説明できた。また、設計フェーズごとの作業の進め方が、作業内容ごとに思考軸を替えることにより行われていると考えられる。

ソフトウェア設計作業では、与えられた問題の解決に必要な対象知識のうち、未解決な対象知識を埋めるために、一連の思考を行なっている。設計者は、設計作業をランダムに行なっているのではなく、ある特定の道筋に従って行なっている。設計作業を制御している道筋を、次の3つに分類する。

一般的道筋 — 設計方法論によるもの。

個人的道筋 — 個人の思考パターンによるもの。

個別的な道筋 — 対象知識によるもの。

設計作業を効率的、かつ漏れなく行なうための一般的道筋がソフトウェアの設計方法論である。この設計方法論は、作業順序や作業内容について、大まかな枠組みを与えてくれる。しかし、設計を実施する段階では、設計者の個性による個人的道筋、設計している対象知識に対応する個別的な道筋があり、どこをどう考えるかは設計者に任されている。設計方法論と設計の実施との違いは、方法論が対象知識そのものを利用していない(利用できない)ことに対し、実施では設計者ごとの個性の違い、対象知識を取り扱わなければならないという点である。

ここでは、対象知識による個別的な道筋を思考軸により分析している。

3 思考軸の設定

3.1 観察実験の概要

人間がソフトウェアをどのように設計しているかを知るために、通信ソフトウェアの設計過程を観察する実験(以下、観察実験と呼ぶ)を行なった。この観察実験での作成対象ソフトウェアは、「音声によりダイヤルできる電話機」の制御部分である。周辺のハードウェアは、フックスイッチ部(HS)、マイク部(MIC)、スピーカ部(SP)、記憶部(MEM)、符号化部(CG)、復号化部(CD)、である。これらとのインタフェース部分は、すでに作成されているという条件で設計を行なった。設計作業は、要求仕様書に記載されているシステム構成や利用手順などから、状態遷移図(SDL図)などを作成するところまでである。

設計作業は、以下の手順で行なわれた。

- (1) 要求仕様書の理解
上位設計者から渡された要求仕様書を理解した。
- (2) SDL図の作成
要求仕様書に記載された利用手順、シーケンス図からSDL図を作成した。
- (3) 共通領域の定義
タスク間でデータを受け渡すための共通領域の名称や書式を定義した。受け渡すデータ項目については、「SDL図の作成」で、すでに取り出されている。
- (4) SDL図の検証
「SDL図の作成」で作成したSDL図を検証した。被検者は、発信側SDL図と受信側SDL図とを交互にたどりながら検証した。
- (5) タスクの定義
検証済みのSDL図からタスクを取り出し、対応するメソッドを定義した。
- (6) 制御表構造の定義
状態遷移を制御するタスク分析表、タスクの詳細を定義するタスク・マクロ表の構造を定義した。

3.2 思考軸の設定動機

前章で「思考軸」は、設計者の頭の中に整理された設計対象の項目の並びであると定義した。それは、関連のある知識をある観点に従って、順序付けて整理しながら設計していると考えられるからである。その他、設計者の思考を分析するうえで考慮しなければ点がある。

- (1) 多視点 (Multi view point)
同時に複数の視点から対象を検討しているときかどうかである。分析では、複数の思考軸を組み合わせることが明らかな場合以外は、分析者の主観が大きなウェイトを占めている。
- (2) 分析範囲 (Granularity)
分析する知識の大きさをどの位に設定するかである。これについては、発話プロトコルに現われた事象や、それらの関連を詳細に取り出すようにした。

ソフトウェア設計過程における設計者の思考対象の変化を追跡した結果の一部を図2に示す。このグラフは、観察実験のプロトコル・データを分析することにより得た。横軸は経過時間を表わしており、そのときに参照していた利用手順を下に示す。縦軸は思考対象の装置である。この図より、思考対象の変化は、ある程度の規則性があることが判る。その規則とは、設計方法と、電話機の利用手順である。この規則性を形式的に表現するために導入したものが思考軸という概念である。

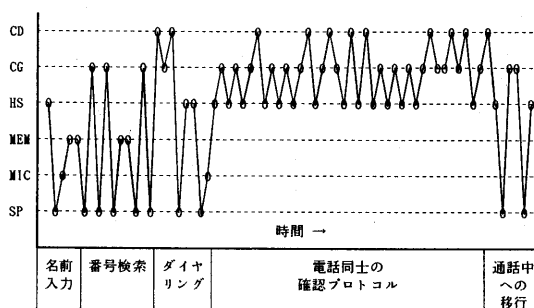


図2: 思考対象の変化

思考軸の存在確認、性質を明らかにするため、ソフトウェア設計過程を実際に分析することにした。その分析にあたり2つの仮定を設定した。

仮定1: 設計過程における思考は、思考軸に沿って行なわれる。

どの思考軸を選択するか、どちらに進むかを決定することは、設計方法論として記述される。

仮定2: 設計者は、局面ごとに思考軸を取り替えながら設計を進めている。

思考の種類により適用できる思考軸が異なっており、その時々思考内容により適切な軸が選択されている。思考軸の要素種類の違いや、順序付けの観点の違いにより、思考軸の適用局面が決定される。

3.3 思考軸の表現法

思考過程のデータ分析では、思考軸を次のように表現する。

- (1) 思考軸は、事物・事象と、それらの間の半順序とにより表現する。
集合で表現した方がよいものでも順序付けて表現する。順序付けの基準として、考えた順や、データ・フロー中での出現順などが考えられる。要素が同じでも、順序付けの基準が異なれば、異なった思考軸と考える。
- (2) 複数本の思考軸により思考過程を表現する。
思考軸により、 n 次元の空間が作成される。この空間の格子に知識を付け加えることにより設計対象の整理が進むと考えている。
- (3) 設計者の思考が、抽象・詳細レベルのように分かれているときには、思考軸もレベル分けする。

4 観察実験により抽出された思考軸

4.1 SDL 図の作成時の思考軸

図3にSDL図を作成しているときの思考軸を示す。この図は、要求仕様書に記述されていた発信時の利用手順から、「状態、イベント、タスクを取り出し」、「イベント、タスクごとの入出力を明確」にし、「入出力データに注目して、タスクを詳細化」という一連の作業において使用された思考軸を示している。

(1) 作業手順による思考軸

SDL図作成全体を制御する思考軸として、方法知識に含まれている作業手順によるものが用いられている。この作業手順の1番目に従って、状態、イベント、タスクを取り出す思考が行なわれた。その結果、「アイドル状態」から「発信タスク」にいたる対象知識が追加された。

この思考が終了すると、作業手順の2番目に従って、イベント、タスクのごとの入出力を詳細化するという思考が開始される。

(2) 状態・イベント・タスク発生順による思考軸 必要情報による思考軸

作業手順による思考軸の下で、イベントとタスクの入出力を明確にするため、直前の思考により追加された「アイドル状態」から「発信タスク」にいたる対象知識による思考軸と、そこで必要な情報による思考軸（方法知識より）を思考軸として、思考が行なわれ、イベントとタスクの入出力に関する知識が対象知識に追加された。

(3) 検索結果による思考軸

入出力の取り出しが終了したことにより、上位の作業手順による思考軸に戻り、3番目の作業に取り掛かった。イベント・タスク発生順に従って、入力と出力との対応を追いかけ、検索結果と電話番号との間にギャップがあることに気付いた。電話番号を検索した結果の個数が「なし」、「1つ」、「複数」の3つケースがあり、設計者はこの思考軸に従って思考し、イベント・タスク発生順の思考軸を変更した。

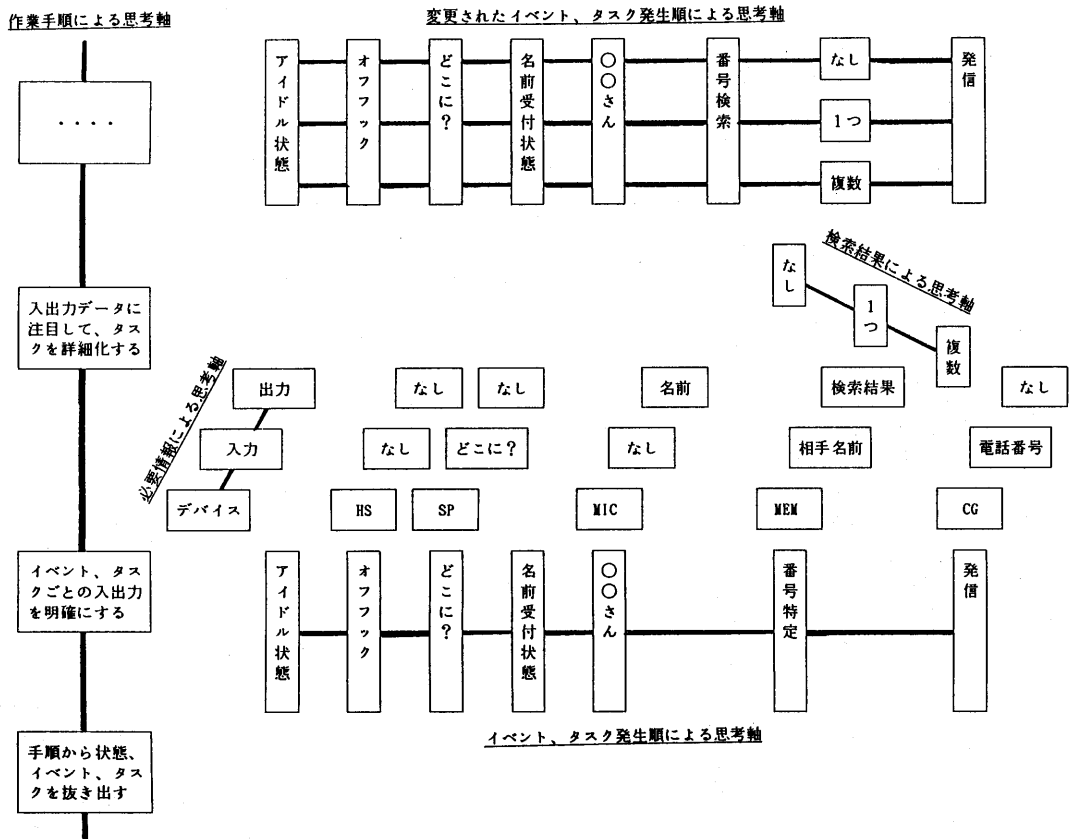


図 3: SDL 図の作成時の思考軸

4.2 SDL 図の検証時の思考軸

SDL 図に誤りがあるかどうかを検証したときの思考軸を図4に示す。この図は、発信側手順による思考軸 (X 軸) と受信側手順による思考軸 (Y 軸) とを突き合わせ、通信が正しく行なわれるかを確認する作業において使用された思考軸とその用いられ方を示している。番号の付いている箱は、設計者の思考を表わしており、その思考対象が発信側手順による思考軸と着信側手順による思考軸に表現される。たとえば、10 番の思考では、発信側手順の着信応答受信と、着信側手順の着信応答送信とを思考対象としている。設計者の思考対象は、次のように変化した。

(1) 1, 2, 3, 4, 5, 6, 7

最初、発信側手順に従って思考し、着信側からのイベントを待つところ (7) まで進んだ。

(2) 7, 6, 8, 9

次に、着信側手順に移り、復号化装置から発信側のアイドル状態から、ダイヤルに対応するイベント、着信を受け取り、符号化装置に対して着信応答を返すところ (9) まで検証した。

(3) 9, 10, 11, 12

着信応答が送られてきたことにより、中断されていた発信側手順に戻り、思考が再開された。

以下、送信側手順による思考軸と、受信側手順による思考軸とを交互に用いて思考が進んだ。

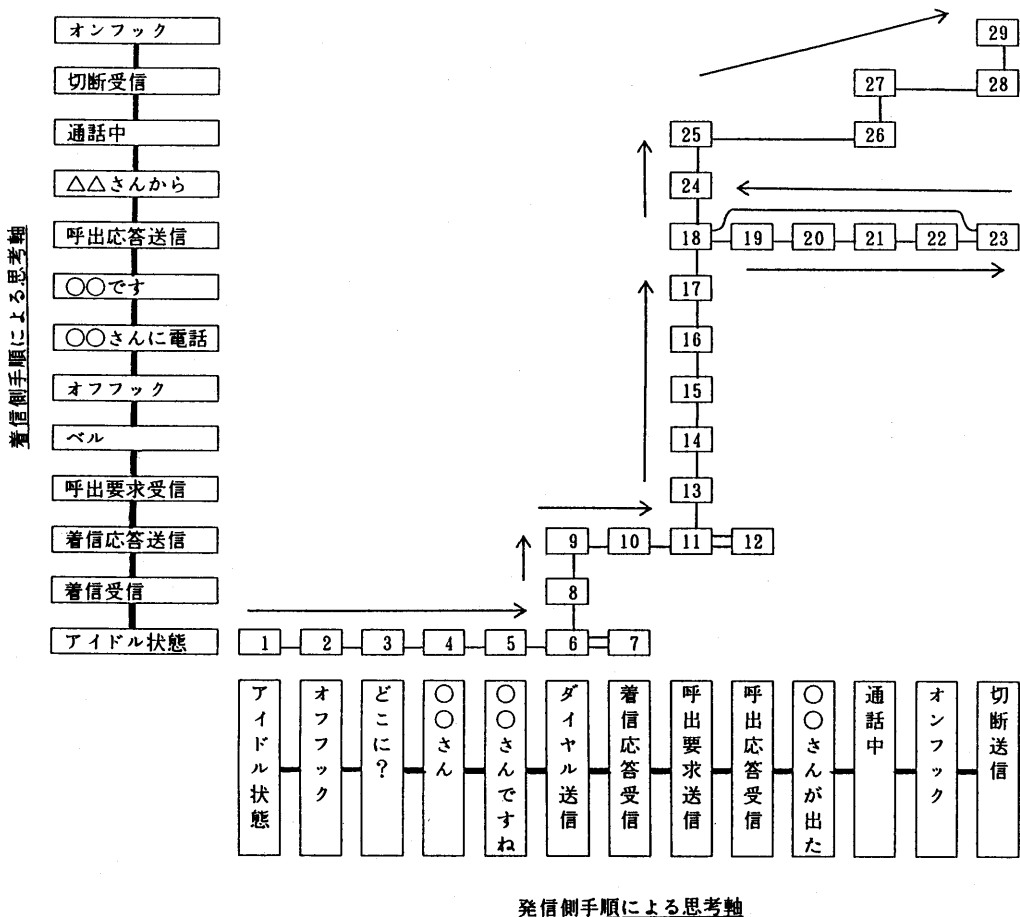


図 4: SDL 図の検証時の思考軸

4.3 タスクの定義時の思考軸

タスクからメソッドを決定する思考に使われた思考軸を図5に示す。この図は、完成したSDL図から、制御部で行なう「タスクを抽出」し、「タスクをデバイス別に分類」し、「タスクに対応するメソッドを決定」という一連の作業において使用された思考軸を示している。

(1) 作業手順による思考軸

タスク定義全体を制御する思考軸として、方法知識に含まれている作業手順によるものが用いられている。この作業手順の1番目に従って、SDL図からタスクを取り出す思考が行なわれ、対象知識が追加される。

この思考が終了したことにより、作業手順による思考軸に従って、2番目のタスクの分類に思考が移った。

(2) タスク発見順による思考軸

共通部分の発見や、整理に都合がよいようにタスクをデバイスごとに分類した。この時には、前に抽出した対象知識による、タスク発見順による思考軸を使用している。

(3) デバイス順、タスク順による思考軸

タスクの分類が終了したことにより、上位の作業手順による思考軸に戻り、3番目の作業に取り掛かった。スピーカ部、符号化部、記憶部の各部のタスク順に従って思考し、タスクに対応するメソッドを決定した。

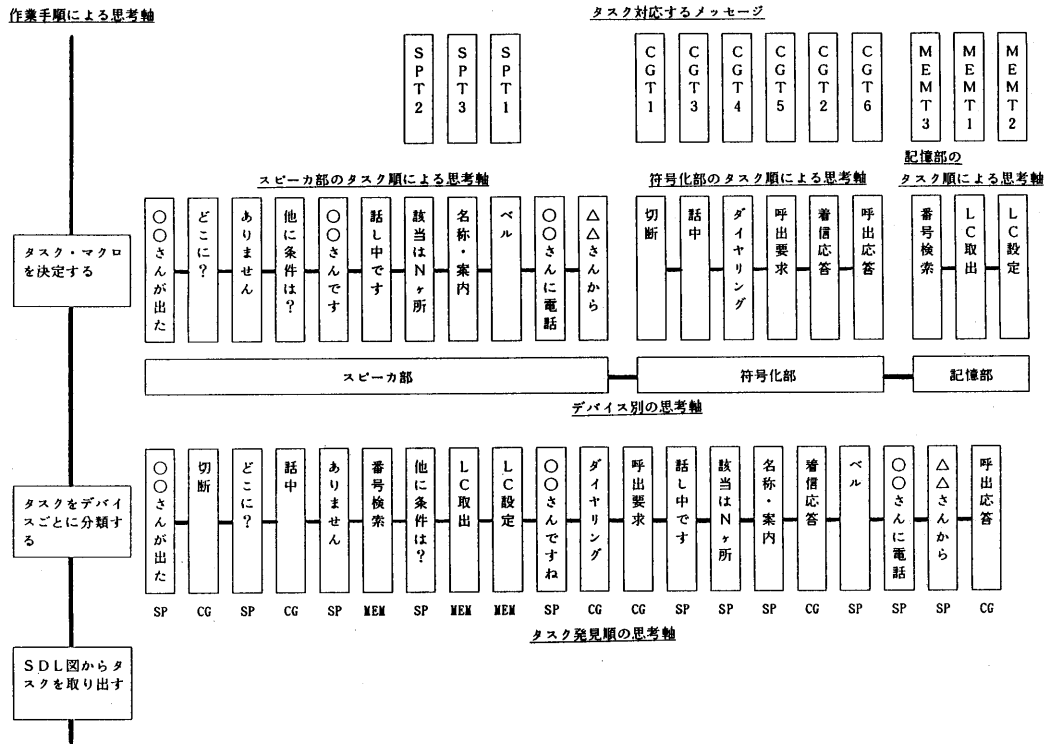


図 5: タスクの定義時の思考軸

4.4 思考軸に関する考察

- (1) 観察実験のプロトコル・データの分析により、思考軸と設計過程に関する仮定1:「設計過程における思考は、思考軸に沿って行なわれる」が確認できた。
- (2) また、仮定2:「設計者は、局面ごとに思考軸を取り替えながら設計を進めている」についても、設計フェーズごとの作業の進み具合が、作業内容ごとに思考軸を替えて行われているということで確認できた。
- (3) 同質の思考軸を複数組み合わせ使用することがある。観察実験での例として、4.2の例があった。この例では、思考軸が組み合わせられて用いられていても対等に取り扱われるのではなく、発信側手順が主で、着信側手順が従となっている。
- (4) 設計過程の中で、思考軸が変更される理由と、その移行先として、以下のものが挙げられる。
 - 思考軸の終端まで思考した。
→ 次の軸に移る。
 - 思考を続けることに意味がなくなった。
観察実験中では、次のような例があった。
 - プロトコルにエラーがあった。
→ 対象知識を変更して、元の思考軸に戻る。
 - 他の案よりも評価結果が悪い。
→ 次の軸に移る。
 - 順序付けの観点と同じ思考軸を複数個組み合わせ使用しているとき、軸の交差点の設計対象の性質に関連あるときには、従から主に移りやすい。
→ 中断されていたペアの思考軸を再開する。観察実験では、確認のために後戻りする例があった。
 - 外部から割り込みが入った。
→ 割り込みを終了後、元の思考軸に戻る。観察実験では、確認のための後戻りや、見間違いの箇所から再開する例があった。
- (5) 次の思考軸は何かは、1つ上位の思考軸によって決定される。たとえば、図5では、スピーカ部のタスク軸に従ってメソッド定義を終了すると、1つ上位の思考軸であるデバイス軸に従って符号化部に移り、その下位の思考軸である符号化部のタスク軸に従って思考が進んだ。
- (6) 思考軸として何を作成するかを決定するのは設計方法論である。方法論は、思考軸とそれに付随するルールによって記述できる可能性がある。

5 おわりに

本論文では、思考軸という概念を提案し、通信ソフトウェア設計時の対象知識の変化を思考軸を用いて分析した。しかし、思考軸は対象知識からだけでなく、領域知識、方法知識からも発生する。思考軸は、このうち、主に対象知識の整理に利用できる見通しを持った。思考軸が方法知識や領域知識に適用できるかどうかは、今後の

課題である。今後は、対象知識とともに、領域知識、方法知識を観察・分析する必要がある。ただし、思考軸を利用することにより、ソフトウェア設計過程での知識を統一的に記述できる感触も得ている。

設計者の思考内容(操作、およびその入力、出力)を思考軸に沿って蓄積し、再現することにより、ソフトウェアの理解性を向上させることができる。また、今回提案した思考軸という概念を、思考軸の作成、選択をコントロールするルールを付け加えたメタ思考軸に発展させることにより、本当の知識支援が可能になるであろう。

参考文献

- [1] Adelson B, Soloway E, "The Role of Domain Experience in Software Design", IEEE Trans. Softw. Eng., Vol. SE-11, No. 11, 1985.
- [2] Curtis B, "Five Paradigms in the Psychology of Programming", MCC Technical Report STP-132-87, 1987.
- [3] Greeno J. G, "A Study of Problem Solving", Lawrence Erlbaum Associates, Inc., 1978. 山口 他 訳, "問題解決の過程", サイエンス社, 昭和60年
- [4] Guindon R, Curtis B, Krasner H, "A Model of Cognitive Processes in Software Design: An analysis of Breakdowns in Early Design Activities by Individuals", MCC Technical Report STP-283-87, 1987.
- [5] Hirakawa Y, Uchida S, Monden M, "Communication Software Design Support Environment Based on the Information Growth Model", Proc. Joint Conf. Computer Network and Switching System, 1988.
- [6] Kant E, Newell A, "Problem Solving Techniques for the Design of Algorithms", Information Processing & Management, Vol. 20, No. 1-2, 1984.
- [7] 甲 他, "言語プロトコル分析手法とそのインタラクティブな分析支援ツール", 第3回ヒューマンインタフェースシンポジウム 3241, 1987.
- [8] Lindsay P. H, Norman D. A, "Human Information Processing An Introduction to Psychology 2nd Edition", Academic Press, Inc., 1977. 中溝 他 訳, "リンゼイ/ノーマン 情報処理心理学入門 III 言語と思考 [第2版]", サイエンス社, 昭和60年.
- [9] Newell A, Simon H. A, "Human Problem Solving", Prentice-Hall, 1972.
- [10] 佐藤 他, "KANTにおけるソフトウェア要求分析・定義に関する一考察", 第35回情報処理学会全国大会論文集 7M-4, 1987.
- [11] 佐藤 他, "設計過程における人間の思考過程の分析", 第36回情報処理学会全国大会論文集 2R-6, 1987.
- [12] 佐藤 他, "図式表現による通信ソフトウェア設計過程の整理方法", 第37回情報処理学会全国大会論文集 5H-2, 1988.

- [13] 島 他, “次世代通信を目指したソフトウェア自動作成システム KANT の構想”, 第 35 回情報処理学会全国大会論文集 7M-3, 1987.
- [14] 内田 他, “KANT におけるヒューマンインタラクション機能の考察”, 第 35 回情報処理学会全国大会論文集 7M-9, 1987.
- [15] 内田 他, “思考履歴を利用した設計支援環境の考察”, 第 37 回情報処理学会全国大会論文集 4L-6, 1988.