社団法人 情報処理学会　研究報告
IPSJ SIG Technical Report

2004－SE－145　(7)
2004／8／20

ソフトウェア工学知識の体系化と その管理法に関する一接近

朱　霊宝†,　　池田　満†,　　落水　浩一郎‡
北陸先端科学技術大学院大学
知識科学研究科†　　情報科学研究科‡

概要

ソフトウェア工学知識の整備にあたっては，ソフトウェア開発にかかる実践知・理論知を一貫性のある体系のもとで整理する必要がある。本研究では，オントロジー工学的手法を適用することにより，ソフトウェア工学知を捉える包括的視点を基本概念構造として明らかにし，その概念構造を通じた知識へのアクセス手法を提供することによって, ソフトウェア工学知識管理の効率と質の向上を目指している．本稿では，その出発点として行った，ソフトウェア工学知の体系 SWEBOK (Software Engineering Body of Knowledge)の分析の結果について報告する．

An Approach To Systemize and Manage the Software Engineering Knowledge

Lingbao ZHU†,　　Mituru IKEDA†,　　Koichiro OCHIMIZU‡
Japanese Advanced Institute of Science and Technology
School of Knowledge Science†　　School of Information Science‡

Abstract

We will report the basic consideration about knowledge structure on Software engineering with applying the technique of ontology engineering to "design / construction" knowledge described in SWEBOK, focusing inter-accessibility of knowledge. We need to compensate the tacit knowledge that are missed during the generalization of heuristic knowledge into generalized software engineering principles. We show the conceptual scheme to represent the knowledge on software engineering and also show the management scheme and access method based on the structure and relationships in the conceptual scheme.

1．はじめに

学問や技術の細分化が進むとともに，大量の知識が急速に生み出されつつある。一方，WWWが拓いた情報の蓄積や交流の新しい形態は，知識や情報の肥大化に拍車をかけ，知識の不特定性，分散性，表現の多様性が顕著になってきている。すなわち，どこにどのような知識どんな形態で記録されているのか，また，ある知識の全体における位置づけはどのようになっているかなど，個々の知識のアイデンティティが従来の体系の枠組みでは捉えにくくなってきている。

このよう状況で，個々の知識の存在を明確にする知識体系を構築し，アクセス性を向上させることへのニーズが高まっており，様々な分野において知識の体系化が試みられている。

ソフトウェア工学の分野でも知識の体系化が試みられている．SEEK(Software Engineering Education Knowledge) は大学教育のための視点,

SWEBOK (Software Engineering Body of Knowledge)は技術者のための視点から, それぞれ知識の体系化を目指している。

このような試みをさらに発展させるために, 我々は, オントロジー工学の手法を用いて, ソフトウェア工学知の蓄積・利用の基礎理論を発展させることが有意義であると考えている。

その第一歩として, 本稿では 2002 年に公表された Trial 版の SWEBOK(software engineering body of knowledge)をもとに, ソフトウェア工学知識の体系化に関する手法を確立するための基礎考察について報告する.

2. 目的と手段

ソフトウェア工学知識の利用にあたってはいろいろな問題があるが, そのひとつに知識構成の問題がある。知識構成問題とは, 実践知から理論へ抽象する知識創造プロセスで, 基本構成要素または適応条件などの, 実践知として本質性の高い要素が捨象されるために生じる問題である。ソフトウェア開発の実践知から方法論などの理論知を一般化し抽象するソフトウェア工学においては特に問題となる。このような問題をオントロジー工学の手法で解決する上での方針を以下に議論する。

2.1 ソフトウェア工学知識の利用に関する問題

ソフトウェアプロダクトやソフトウェアプロセスは複雑かつ不可視であるため, 学習者がその内容を理解し適用する際に困難が伴いがちである. また, 異なる文脈から抽象化して得られた知識群から, 直面している実問題に適切な知識を選択し, 実情に即してカスタマイズして適用することも困難な問題である。

ソフトウェア工学の多くの知識は, 組織または個人の開発経験を抽象した理論・技法・考え方といったベストプラクティスの集合体であるために, 例えば, 経験を通じた実践知を方法論に抽象する過程において, 実際の場面に適用するのに役に立つ情報が含まれていない[4]ことがしばしばある。また, 人による抽象度のレベルが不均一性であり, 知識の表現が多種多様で, かつ, 断片的に散在する傾向が高い[2]。

2.2 オントロジー工学のアプローチ

知識工学分野ではオントロジー工学を「知識の構成概念に関する理論」という意味で用いている。知識の共有と再利用という知識工学の重要課題を解決するために, 知識に関する共通理解の基礎となる概念体系を明確に記述しようという思想のもとに生まれた知識工学技術の一つである。

知識の相互伝達の質と効率は, 知識の受け手と送り手の間での知識構成概念の共通性に強く依存する. 共通性が高ければ知識は効率よく適切に伝達できるが, 知識の構成概念に対するお互いの認識にズレがあれば, 相互伝達の質と効率は低下する.

オントロジー工学は, この問題に対して, 知識の構成概念を体系化し, それに基づいて,

(1) オントロジーに基づいて知識を表現する共通語彙を設定する

(2) 個々の知識の位置づけをオントロジーの構成概念を用いて明確にする.

(3) 個々の知識の内容をオントロジーの構成概念を用いて明確にする

ための手法の確立を目指している. (1)〜(3)は実現する順序（(1)は(2)の実現に必要, (1),(2) は(3)の実現に必要）であり, その順に相対的に実現が困難な課題になっている. (1)の目標を, シソーラスと対比して特長づけると, シソーラスが語と語の間の関係で語の意味を明確にしようという試みであるのに対して, オントロジーベースのアプローチは, まず基本となるオントロジーを構築し, その構成概念と語の対応関係によって語の意味を明確にしようという試みであると言える. 知識に関する様々な表現に共通語彙を用いることで, 知識の相互伝達の質と効率を向上させようという試みである. (2)は知識の位置づけ（主題, 使われる文脈等）を概念体系との対応で明確にしようという試みである. 知を実体として扱い, それぞれの知識について, どのような時に, どのような内容について, 有用かということを, オントロジーの構成概念を用いて明確に表現する. オントロジーは関係者が共有可能な概念体系として, 知識へのアクセスのインデックスを提供する. (3)は知識を, オントロジー中で定義された概念を構成概念

として, 推論エンジンが操作可能な形式（ルール・フレーム等）で知識を表現しようという試みである.

このうち本研究では，ソフトウェア工学知識について(1),(2)の手法の確立を目指している． (2)の考え方は，全ての知識に対して，それを位置づける概念構造を設定し，その概念構造をインデックスとして知識への柔軟で多角的なアクセス手法を提供するものである．例えば，実践知（現場での経験的知識）を記録したドキュメント，現場の熟達者が持っている暗黙知，ソフトウェア工学理論，といった多様な知識が，「どのような文脈で，どのように有用であるか」という観点から，一貫性の高い概念構造を通じて位置づけられることになる.

2.3 本稿の対象とアプローチ

我々はこれまでに，SWEBOK 中の３章「ソフトウェア設計」と４章「ソフトウェア構築」の２つのアクティビティに関して，それらのアクティビティに関する知識へのアクセスインデックスとして， 図 1 に示す概念構造を抽出した．図１に示す概念体系において，「(開発）主体がアクティビティを遂行する時，作業上の制限条件（原則・様式）を特定し，それに基づいてツール・技法を選択して利用し，入力から出力を生み出す」という活動を表現している。以下，本稿では，この概念構造を基礎にして，SWEBOK で提示されている知識記述および関連参考文献を関連づける。

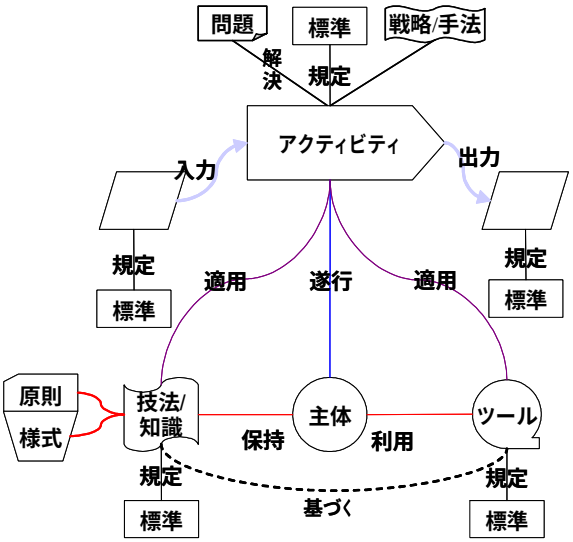

図 1　ソフトウェア工学知識の概念構造

3. ソフトウェア構築･設計に関するアクティビティテンプレート

SWEBOK ではウォータフォールライフサイクルのフェーズに沿って設計、構築の順で説明されているが，ここでは説明の便宜上，概念構造が相対的に単純な構築アクティビティから説明する.

3.1 ソフトウェア構築に対する記述

SWEBOK の第４章「ソフトウェア構築」に記載された知識に対応する概念を抽出し，その関係を記述することにより，以下の結果を得た。

図 2 は図 1 を第４章における記述に適合させた概念構造である．また，図 2 中の各概念に SWEBOK 中の対応する記述（定義，説明など）を付与している.

この基本概念体系をソフトウェア構築に関する知識を整理するための土台とする。図２は図１に従って，SWEBOK における「ソフトウェア構築の定義」, いくつかの記述を概念体系に対応づけた例である。

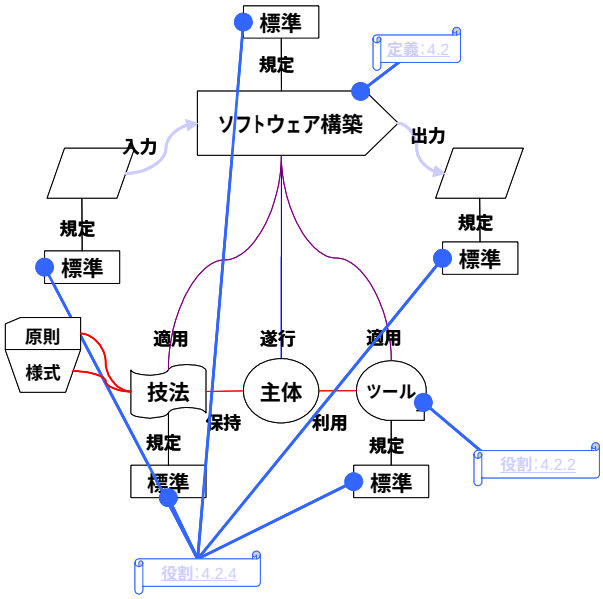

図 2　ソフトウェア構築アクティビティの概念構造

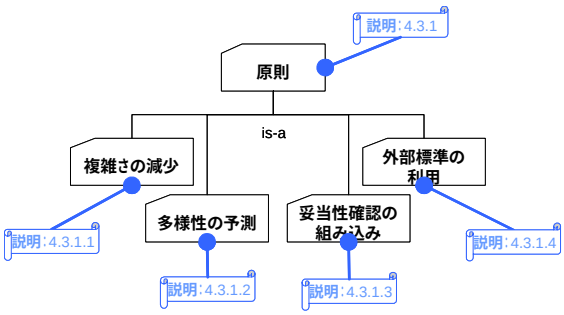

図 3 ソフトウェア構築の原則

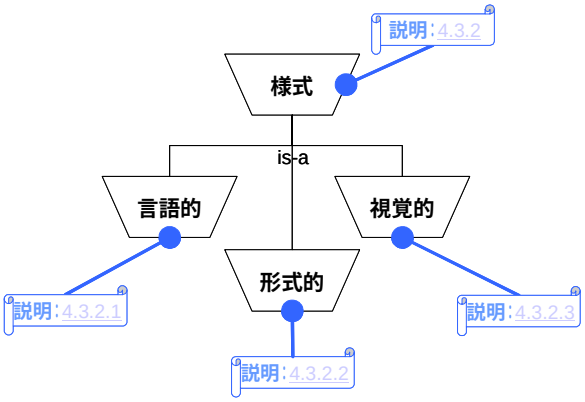

図 4 ソフトウェア構築の様式

ソフトウェア構築アクティビティに強く影響を及ぼす要素は, 図 3 の原則, 図 4 の様式である。また, 原則・様式に対応して, どのような具体的な技法があるのかを図 5, 図 6 に示している。

図 5 では, 原則を「複雑さの減少」とすることで, 技法が「複雑さの除去」「複雑さの自動化」「複雑さの局所化」という下位概念に詳細化される例を示す。

図 6 はソフトウェアを構築する際に, 複雑さを減少させる原則のもとでの言語的様式として有効な技法として,「ソフトウェアテンプレート」,「コンポーネントライブラリとフレームワーク」,…といったものがあることを示している。論文「BENOO Chap2,3」「KE99 Chap2,3」はそれらの技法を記述した文献を表している。このように SWEBOK では, ソフトウェア構築アクティビティに有用な知識は原則・様式概念を基準とし概念構造に沿って整理されている。

ところで, SWEBOK の記述には以下のような問題点が存在している。

以上の概念構造から技法の選択はツール機能に基づいて行い, ツールは技法に適合できるかどうか問題がある。このような緊密の関係で知識体系ではツールに関する知識も整備される必要がある.

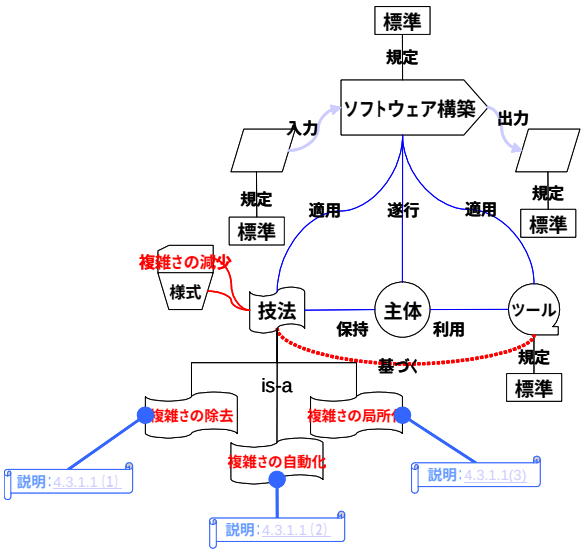

図 5 原則を「複雑さの現象」とした場合

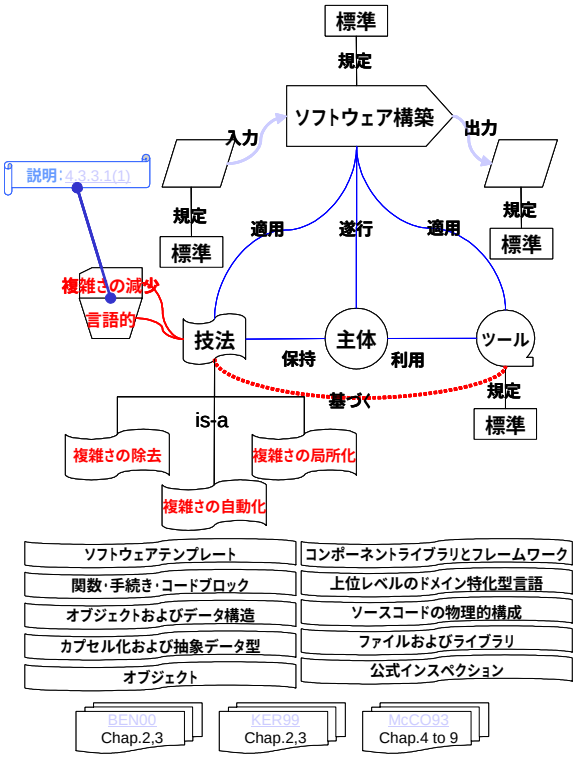

図 6 原則を「複雑さの現象」, 様式を「言語的」とした場合に適用可能な技法

実際にソフトウェアを開発する際には，技法とツールがきわめて重要な要素であると多くの実践者は考えるものと思われるが，SWEBOK ではツールの役割だけを記述しており，「取り扱いの深さ」の理由でツールそのものに対する知識の記述を省略している。また，既に示した選択基準の「原則・様式」にしても，それぞれどのようなツールに関わっているか，さらに技法とツールとの関係も記述されていない。

「複雑さの減少」といった原則はソフトウェアエンジニアリングプロセスのいたるところで適用されうるが，他のところで使われている技術とどのようなつながりがあるか，というような情報が記述されていない。また，実際に図６においては「技法」の下位概念として「複雑さの除去」「複雑さの自動化」「複雑さの局所化」などのものがあるとともに「ソフトウェアテンプレート」，「コンポーネントライブラリとフレームワーク」，…といった知識として確立したものもあるが，この二種類の下位概念間の対応付けは明確にしていないために，技法とそれを記述する一般化する知識との対応関係も明確化されていない。すなわち，技法やツールを体系化するうえで十分な分解能をもっていない。

また，一つの技法を理解するのに，その知識を十分に記述している参考資料への参照が整理されていない。

3.2 ソフトウェア設計についての記述

SWEBOK の「ソフトウェア設計」章の知識に対して概念を抽出し，その関係を明確化することにより以下のような結果を得た。

(1) 図 7 は図 1 の基本概念構造を，第 3 章「ソフトウェア設計」の内容に適合させたものである．この概念構造には，問題状況下でのトレードオフスタディとしてソフトウェア設計のための知識を整理する意図が込められている。

(2) ソフトウェアシステムを設計する際には，数多くの問題と向き合わねばならない。その主要問題を図 8 のように表すことができる。

(3) 図 8 で示された共通問題・基礎問題を解決するために，アプローチの基盤をなす概念や思想にとって重要な指針である原則を図 9 に示した推奨技法（原則）で表すことができる。

(4) 図 8 で示された機能横断的な問題に対してどのような設計技法があるか，SWEBOK では言及されていない。

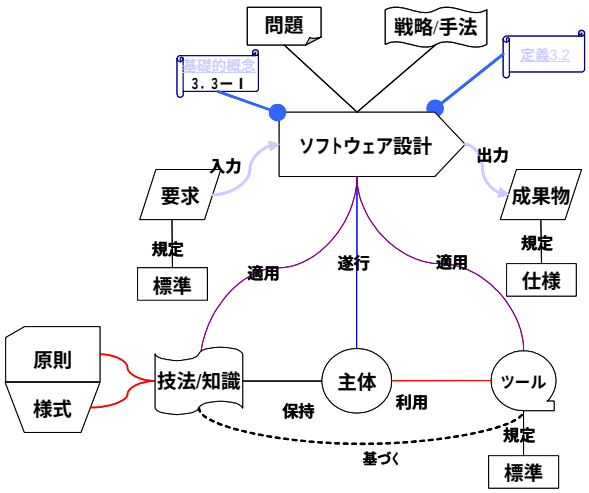

図 7　ソフトウェア設計アクティビティの概念構造

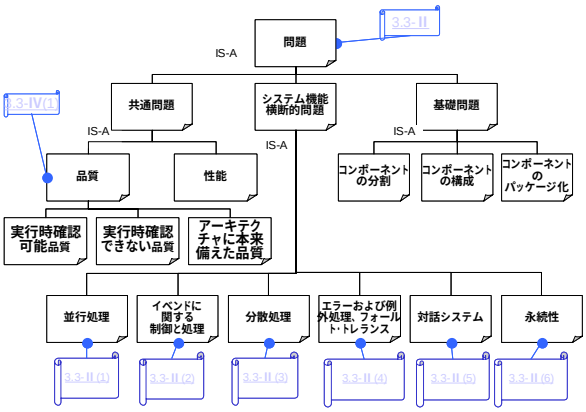

図 8　ソフトウェア設計における問題

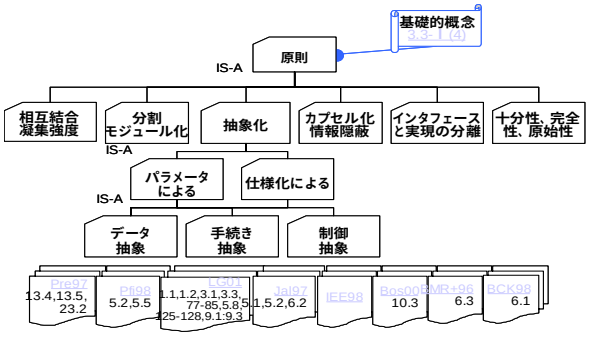

図 9　ソフトウェア設計における原則

(5) 図 8 で示されたソフトウェア構造とアーキテクチャを含む基礎問題を解決するために，有用な設計知識は図 10 のように表すことができる。

(6) 図 8 で示されたソフトウェアの「品質」という共通問題を解決するために，ソフトウェア設計というアクティビティにおいては，種々の品質特性を考慮する必要がある。設計成果物の品質特性を検証し保証するために，図 9 で示した原則に基づき，図 11 で示された品質保証技法を設計というアクティビティに適応させた品質分析と評価が必要である

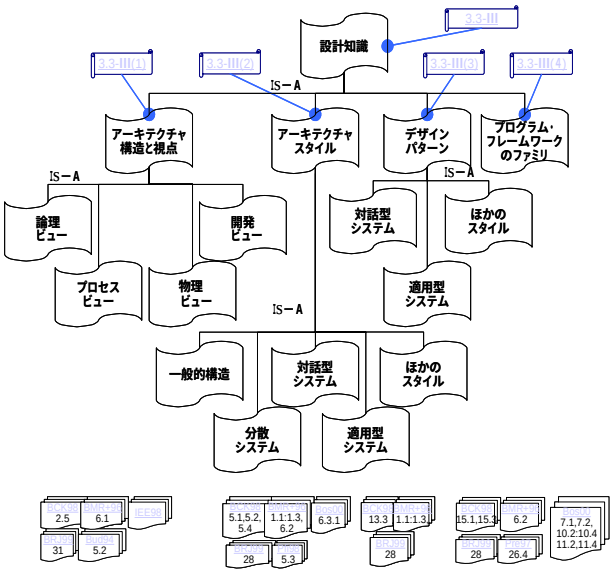

図 10 ソフトウェア設計知識

(7) 図 8 で示された概念構造の中の「様式」による特徴づける技法は図 12 と図 13 で示したように，ソフトウェア設計成果物を表現するために数多くの表記法が存在している。異なる視点によって，表記法も違ってくる。

(8) 図 8 で示された基礎問題と共通問題を解決するために，プロセスをガイドするものとして全体戦略の選択が必要である。図 14 は概念構造の中の「全体戦略」を詳細化したものであるが。SWEBOK では手法を全体戦略と対照したものとして記述されているが，概念としては種類分けの戦略として使われている。図 15 で表す。

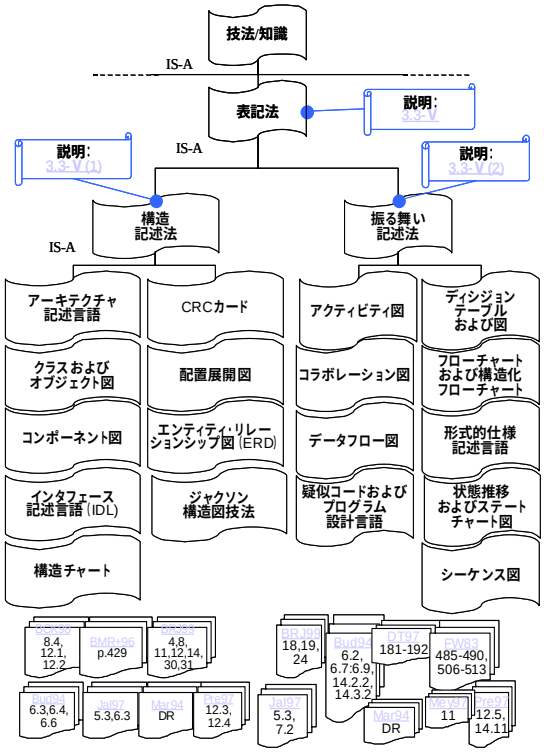

図 12 ソフトウェア設計ための表記法

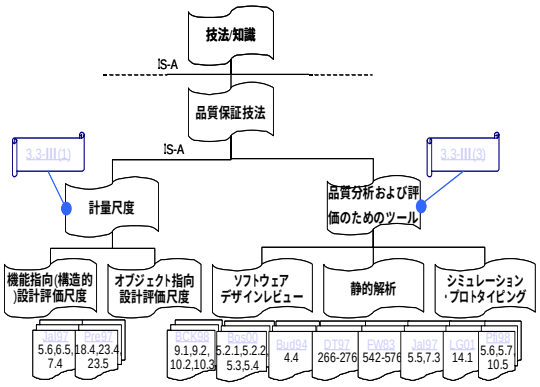

図 11 ソフトウェア品質保証ための技法

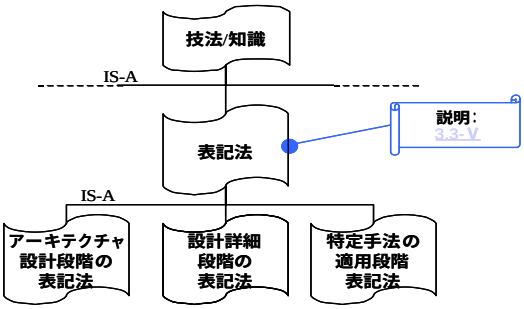

図 13 ソフトウェア設計段階における技法

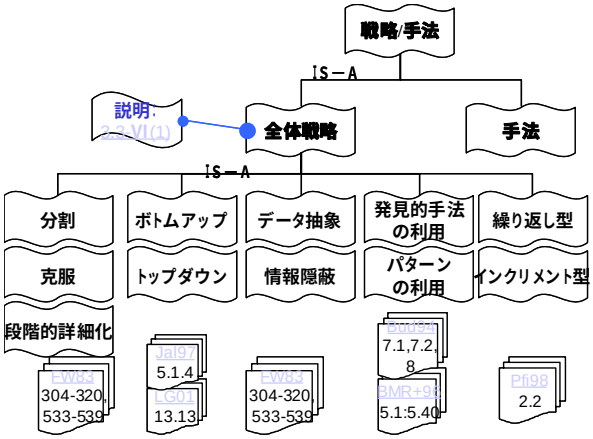

図 14　設計のための全体戦略

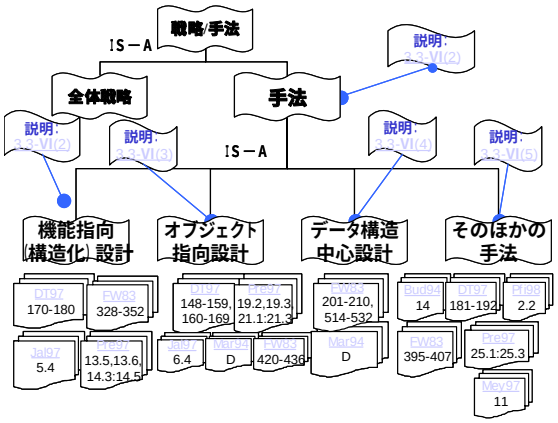

図 15　ソフトウェア設計のための手法

以上で(1)から(8)までのように，SWEBOK の「ソフトウェア設計」章を基に共通理解のベースとしての概念構造を抽出し，それに沿って知識を有機的に組織化することができる。しかし，この手続きが適用される過程に以下の問題があると考えられる。

ア．設計アクティビティに適用できる技法が，開発対象の重要度と難易度，コスト，価値[4]といった背景状況によって制限されている、それについては記述されていないので、概念構造に沿ってそれが整理できない。
SWEBOK では主要問題およびそれらを解決するための技法を提示したが問題とそれに関係する技法との対応関係は明確ではない。また，共通問題と基礎問題についても詳細に記述されていない。

イ．実際に，ソフトウェアアーキテクチャの構造と視点としてのビュー，アーキテクチャパターンとしてのメタモデルおよびデザインパターンといった図 10 で表した設計知識に沿ってソフトウェアの開発が行われ，それに対応する技法が重視される。しかし, SWEBOK ではこれらに関する知識も参考文献も記述されていない。

ウ．SWEBOK では「技法」と「ツール」または「戦略」と「手法」という概念の使い分けを行っていないために，図 14，図 15 でそれらの概念を抽出されても，それに相関する知識を詳細化することが難しく，利用者にとって混乱のもとになりかねない。このような用語の不適切さは SWEBOK に多く存在している。

エ．　推奨技法（原則）に基づいて，技法を適用することになるが，一つの推奨技法（原則）に沿って具体的などのようの技法が適応できるかという具体的な知識へアクセスのための経路が明確化されていない。

オ．表記法について，SWEBOK では図 12 と図 13 とそれぞれの視点から分けたが，図 13 で示された概念にたいする関連知識が記述されていないために，それに対する詳細化が行えない。

3.1,3.2 での問題分析から知識体系としての SWEBOK は以下のような課題が残されていると考えられる。

A)　トピックを記述する概念の使い分けは一貫性が足りないために，概念構造の抽出，概念の詳細化は特に難しい.

B)　トピックが独立に記述されたため，概念間の相関関係の特定が難しくなり，概念を構造化するのが容易ではない。

C)　トピックに関する参考文献の粒度の設定に合理性と一貫性が欠けている。

D)　ソフトウェア設計やソフトウェア構築といった開発アクティビティはソフトウェアシステムの規模によってさらに多くの子アクティビティに分けられている。実際に，ISO/IEC12207 および ISO/IEC12588 ではライフサイクルプロセス，アクティビティ，タスクなどの共通フレームで多くの作業アクティビティを定義されている。それに適用できる技法とツールが多く使われているが，

整理されていない，知識として特定し，整理が必要である。

さらにソフトウェア開発の実践から考えると以下のような問題もある。

i. 「トピックの採用基準」にこだわるために，広い範囲の知識は限られたトピックで提示され，部分的な知識の提示に留めている。また，トピック間のつながりが記述されていない。同時に，多くの場合に適用できる重要な知識が一つ場合に限定され，知識体系としての整合性が足りない。

ii. 知識の合意性と一般性にこだわるために，専門的な知識に踏み込まないことは実践者にとって役に立つとはいえない。

iii. ソフトウェア開発環境の制限によって適用できる方法論も違ってくる。トピックとして提示された知識はどのような場合に有効であるかといった付加知識が記述する範囲に入っていない。

iv. 実践開発する際には，コミュニケーションによる知識がソフトウェア開発チームに必要である。ところがｉによって知識を共有する基盤がないために，コミュニケーションを円滑に進めることができない。

4. まとめと今後の課題

本稿では，オントロジーの手法を用いてSWEBOK の「ソフトウェア構築」と「ソフトウェア設計」で提示された知識から概念を抽出し，それぞれの概念構造で知識を体系化することができた。このようにオントロジー手法で実際のアクティビティを概念構造でモデリングし，それに沿って知識を体系化と管理する手法して確立できると例示した。また，SWEBOK で知識分野体系化する場合に存在する問題点を明らかにした。

同時に，人間の主観を多いに含んだソフトウェア工学知識を共有することの壁の高さを実感しこのような手法を適応する過程に概念間の関係を適切に設定できない箇所も多く存在し，さらなる概念体系の改良が必要である。

さらに，ソフトウェア工学の知識は方法論や思想や戦略などを中核としているので，知識体系化する際に，それを概念体系で明確に記述することが大切である。また，技法やツールの選択基準に関する知識の記述もきわめて重要であると認識しており，今後の課題と考えている．

また，ソフトウェア開発、特にソフトウェア設計・構築アクティビティは、入力として与えられる対象世界における価値(functinal requirement)と制約(non-functinal requirement)を、納期、コスト、標準、品質、開発構成員のスキルレベル等を考慮しつつ、適切なバランスで充足させる活動(trade-off study)である。曖昧で抽象的な対象に対して，このように高度な知的活動が求められる点が，ソフトウェア開発の特色と言えるであろうし，ソフトウェア開発を熟達するに困難さの理由の一つであろう．今回の分析を通じて，そのような知的活動の基礎となるソフトウェア工学知の体系について， SWEBOKが明確にしきれていない点が多々あることがわかった．そのような点を着実に明確にしていくことが，ソフトウェア工学知の相互伝達の基礎を作るうえで重要であると考えている．

5. 謝辞

日頃討論いただく，北陸先端大ソフトウェア工学-オントロジー研究グループの皆さんに感謝の意を表します。

参考文献

[1] IEEE Computer Society Software Engineering Coordinating Committee (SWECC): Guide to the Software Engineering Body of Knowledge Stone Man Trial Version 1.00, http://www.swebok.org, 2001.（松本吉弘監訳：ソフトウェアエンジニアリング基礎知識体系－SWEBOK－，オーム社，2003.）

[2] 池田　満，落水　浩一郎，林　雄介，長谷川　忍：“ソフトウェア科学知マネジメントへのオントロジー工学的アプローチ”,ソフトウェアシンポジウム 2004, 2004.06

[3] 溝口理一郎：“オントロジー研究の基礎と応用”，人工知能学科論文誌, Vol. 14, pp.977-988, 1999.

[4] 落水浩一郎：“ソフトウェア開発方法論とソフトウェアパターン”，情報処理学会シンポジウムシリーズ IPSJ Symposium Series Vol. 2004, No.4, pp 65-66, 2004.