「オントロジー」など「汎用語」の情報処理分野への転用史

塚田 良央
(株)ジェーエフピー
tsukada_yoshio@jfp.co.jp

佐々木 千春
(株)ジェーエフピー
sasaki1000@jfp.co.jp

漆原 憲博
(株)ジェーエフピー
japanfp@jfp.co.jp

栗田 太郎
ソニー(株)
taro.kurita@sony.com

要旨

新たな語の登場は，既存の学問や産業領域の転換を想起させてきた．そう考え，情報処理分野で広く用いられている「システム」など,いわば「汎用語」の変遷をたどってみた．本論文で扱う語は，「アーキテクチャ」，「オントロジー」，「システム」，「モデル」とした．変遷とは他の分野からの語の転用（導入）や変化などを指す．これらが，われわれの開発現場にどう影響するのか，効果はどうか，を探ってみた．

「システム」や「アーキテクチャ」は導入当初はいかであれ，いまは不可欠で誤解の生まない，あるいは生みようのない語で，特に「システム」などはどの分野でも，また日常生活のどこででも使用されている語になった．しかし，語の普及また定着の一方で，「システム」などは時代の要請から安全などが問われ，語もまた例えば「system-theoretic（システム理論的）」などと意味を発展させた新たな方法論が進展している．

「モデル」は説明されることはあっても，定義はされない語ではないかと思われる．なんでも「モデル」になる，と時に陰口も聞くが，定義されない，したがって他を定義する語としての役割があるようである．陰口もここに起因するなら「モデル」に罪はなく，むしろ，業界には孝行息子というべきである．

時代の課題（要請）があり，新たな語が投入される．それぞれの語を概観しつつ，最後に「オントロジー」に少し紙幅を費やすこととした．Web 上，また要求仕様分野など，「概念」の「存在」のための方法論であるが，哲学から借りた用語である．

1. はじめに

「オントロジー」などという異分野の，いささか耳慣れない語が，組込みシステムの開発現場にも現れたりすると，現場やその取り巻く環境に新たな変化が起きたのか，などと思う．そして，異分野の耳慣れない語が新たな産業に浸透し，産業を発展させて来たことに気づかされる．

新たな産業といえば，われわれの依拠するコンピューター産業も，社会にその職業が認知されていまだ半世紀前後のものであろうと思われる．「アーキテクチャ」は「オントロジー」と対比するには，あまりに耳慣れた語であるが，「アーキテクチャ」も元々は異分野である建築業界の用語であったろう．「システム」もまた，当業界ではいまやあまりに当たり前であるが出自があったに違いない．そして，ときにバズワードともいわれかねない「モデル」もしかりである．

もっとも，出自といっても，これらの語がすべて海外生まれであることも，日本人には一つの「出自」にはなる．しかし，これらの語はどれも日本生まれではないから，日本のいわば「やまとことば」が持っているとする意味から辿ることはできない．言語間ハンディとでも言うべきものを負っているわけであるが，ここではそのことはいったん置き，本論文で異分野の語という場合，産業や学問などの分野の違いということとする，

以上挙げた語，その中でも「システム」はコンピューターの開発業界では広く使われているという意味で「汎用語」と命名できるかもしれない．あるいは，単に広く使われているだけではなく，仕事上の意味伝達をする上で，基本的な役割をするので「基本語」というべきかもしれない．あるいは，意味の領域区分のようなものがあり，その領域の基盤，あるいは土台のようなものを成しているという意味で，「基盤語」などという命名も可能かもしれない．

本論文では，用語の出自や，用語が当業界に登場してきた経緯などを変遷としてたどる．そして，そのことが現場の，これらの語が持つ課題へのヒントになればと期待するものである．

2. 用語の変遷

2.1. 用語「アーキテクチャ」の変遷

「アーキテクチャ」という語は本来建築分野の用語で「構造物」や「構造」を指すといわれる.

コンピューターに対してこの「アーキテクチャ」の語を使用したのは, 1960 年, Johnson が始まりである[1]. また複合語「システム・アーキテクチャ」は, あるシステムを例に, そのシステムを構成する基本的な要素と, それぞれの機能, およびそれらの機能が協調して動作する方法を意味していた. したがって, システム・アーキテクチャは, 論理設計や回路設計よりも抽象的な階層を指していた. この複合語は, 1964 年の IBM 社のコンピューターSystem/360 で最初に使われたといわれる[39].

ソフトウェアに対して「アーキテクト」の語を使用したのは, 1969 年の会合が始まりである[2]. Naua は会合で, 「ソフトウェアデザイナーの仕事はアーキテクト(建築家)に似ている, これは大規模で異種混交(heterogeneous) な構造を対象とするからである」, と発言した.

続く 1969 年の会合での Sharp の発言では「ソフトウェア・アーキテクチャ」の語を用い, その意義を強調した[3]. Sharp は, ソフトウェアの細部が理論面でも実践面でも素晴らしいものになったとしても, 全体を統合することがうまくいかなければソフトウェア開発は失敗すると論じた. ソフトウェア・アーキテクチャは, この場合, ソフトウェアの全体を統合するという課題を指して用いられた.

その後, アーキテクチャの語は, もっぱらシステム・アーキテクチャ, すなわちコンピューターの物理的な構造に対して使われた[4]. 狭い意味では, コンピューターのインストラクションセットを意味することも多かった[4].

1991 年には Royce 親子がソフトウェア・アーキテクチャの語を用いた[5]. その後, 1990 年代にかけて, ソフトウェア・アーキテクチャの研究が興隆を見た[4].

日本でも, 1970 年ないし 71 年には, 「アーキテクチャ」の語が使われた[6],[7],[8],[9]. これはコンピューターのハードウェアの基本的な構造ないし機能を意味する. またこの語「アーキテクチャ」は System360 や後継の System370 への追随を狙っていた日本の研究者や技術者の間では使用されていたであろう. また 1970 年前半に System370 を導入した日本企業のエンジニアの間でも同様である.

2.2. 用語「オントロジー」の変遷

「オントロジー」は日本語では「存在論」と訳され, 究極の存在に関する哲学的議論とされてきた. 存在の究極を質料と形相とした形而上学を打ち立てた古代ギリシャのアリストテレスに端を発するといわれるが, 「オントロジー」という用語が哲学の領域に正式に登場したのは18 世紀といわれる.

しかし, 存在に関する議論が物理科学の常識をあまりに超えたことから, 経験科学や数理論理学の立場から存在を論ずる議論が起こった. 哲学者カルナップやクワインなどの, 構文と意味規則を定義しつつ, 理論の系をはみ出すことなく存在を議論する方法や, 在るとは何か(what there is)と, 存在記号“ヨ”を問いながら平易に「在ること」を論じる方法は, 20 世紀半ばに大きな流派を築いた一群の議論である.

このような論理的また実証的な立場の哲学の方法が, こんにち Web 意味論など「オントロジー」の方法にも見て取ることができる.

2.2.1. 情報処理分野

Powers は自然言語の機械学習についての論文で「オントロジー」の語を用いている[12]. オントロジーについての言及があるのは記号接地 (symbol grounding) についての節である. Powers はオントロジーという語を明確に定義していないが, 語彙と環境(生物あるいはロボットにとっての) の総体, すなわち存在, という意味合いのようである.

Gruber がオントロジーという用語を使用した当初は, 知識ベースがドメインごとに孤立して存在しており, 知識の共有と再利用が容易でないことが問題意識としてあった[13]. 共通の知識ベースの開発が Gruber の関心の対象である. Gruber は, 「共通オントロジー」を提唱するが, これは, 複数の知識ベースを接続し, 知識の共有と再利用を可能にすることを意図したものである.

その後, Gruber は語「オントロジー」を「概念化の明示的な仕様書(an explicit specification of a conceptualization))と定義し, この語は哲学から借りたという. オントロジーは「存在(Existence)に関する体系的な言説」であり, 「存在するもの(what “exists”)」はまさに表現可能なものであり, それゆえ, 知識ベースの開発に「オントロジー」を用いる, とした

[14].

また，上の「概念化」を Gruber は，Genesereth らを引用し，ある関心領域に存在すると推定されるオブジェクト、概念、その他のエンティティを保有する関係と定義している[15]．そして，Gruber によれば，概念化とは、何らかの目的のために表現したい、抽象的で簡略化された世界の見方である，という[14].

Feilmayr と Wöß は，オントロジーの定義を「複雑さの増大によって要求される高度な意味論的表現能力に特徴づけられる，共有された概念化の形式的で明示的な仕様書」とした[16]．上の Gruber らの定義をさらに詳しくしたものである．

2.2.2. 日本での展開

日本では，1989 年から 1991 年にかけて，定性推論と自然言語処理に関連して「オントロジー」の語が見られるようになった[17],[18],[19]．この時期の「オントロジー」の用法は論者によって異なる．

1993 年には中島らによるソフトウェア工学の論文で「オントロジー」が言及されたが，その応用の可能性については否定的な見解となっている[20]．中島らによる論文では，オントロジーを「システムを説明するために必要な基本的な考え方とそれに基づく具体的な概念の集合」と定義している[20].

他方，同じ 1993 年の人工知能分野での論文では，ティヘリノ A．ジュリらは「オントロジーとは，哲学では存在論という意味であるが，人工知能では物事を表現する際の基本となる最小の単位（プリミティブ）の集合という意味を持っている」としている[21].

このように，1990 年代初頭においては，「オントロジー」という語の定義や用法は論者によって異なり，オントロジーという概念が導入途上であることがうかがえる．ただし，オントロジーに「基本」とか「要素」などプリミティブな意味を持たせているのは共通である．

2.3. 用語「システム」の変遷

1940 年代から 50 年代にかけて，Wiener や Ashby などによって，サイバネティックス論，さらにシステム論という学問分野が盛んになった．"system"という語も，それに応じてよく使われるようになった[22],[23].

一方，ソフトウェアの分野ではどうか．"system software"という語は，例えば 1972 年の文献では，"user software"と対比させて使われている[24]．ベンダーの責務で提供するソフトウェア（system）と，ユーザーの責務で作成するソフトウェア(user)との間に境界を設けている．用途別の形容を加えて，語「システム」を使っている．

その後は，同じ事柄を指すとしても"system software"と"user software"よりも，"operating system"と"application program"または"application software"が多く使われるようになる[25].

1973 年当時日本のコンピューター技術をリードしていた富士通のコンピューターの解説書でも，“システム”と“ソフトウェア”の語はあるが，両語が組み合わされた“システムソフトウェア”の語はない．“オペレーティングシステム”は存在する．また“ソフトウェア”の下位に“プログラム”が位置している．“ハードウェア”は“ソフトウェア”と同じ位置にある．また“アプリケーションプログラム”は“ソフトウェア”，“ハードウェア”と同位置にある[33].

以上のことは，新たな語の導入の歴史が浅ければ浅いほど，指示対象が不安定であることを物語る．と同時に，指示すべき新たな事柄も生まれていなかった，ということも示すであろう．

2.4. 用語「モデル」の変遷

「モデル」という語はさまざまな分野で使われており，「モデル」を含む語句も多くある．ソフトウェアの分野に絞っても，例えば，以下の語句がある．これらの語句は，何らかの方法論を示す「用法」とでも呼べばよいかもしれないが，語の変遷という視点から，単に語句と呼ぶこととする．

ソフトウェア分野で"model"の語句といえば，UML すなわち"Unified Modeling Language"がある．UML には源流となった 3 つのアイディア，Booch，OMT，OOSE がある[31]．このうち"model"という語を使用しているのは OMT，すなわち"Object-modeling technique"のみである[32].

"software process model"という語句もある．この語句の代表例が"waterfall model"であるので，これらの語句の登場の時期を調べると，1986 年には"software process model"という語句が使われている[28]．1983 年あるいは 1988 年には"waterfall model"という語も見える[29],[30]．ただし，"waterfall chart"という語も使われており，こちらの方が使用頻度が高い．"model"よりも"chart"の方に語の利便があったということであろうか．あるいは，"model"が"chart"よりも語としてのなじみがなかったということでもあったろうか．ただし，その後"waterfall chart"は"waterfall model"に置き換

わって行く.

3. 用語の概観

以上の4つの用語を取り上げた，その順は特に意図があったわけではない．要求仕様の開発の現場で，よく「アーキ」が使われているので，そこから論を始めたに過ぎない．「オントロジー」は「アーキ」の連想である．アーキテクチャが落ち着かないなら（いろいろ図を書き変えざるを得ないなら），「オントロジー」なら単にいろいろ書いてみるのではなく，2，3の基本的な文から推論などで「アーキ」を仕上げる手法かと推測したからに過ぎない．その後，調べは「システム」，「モデル」と進んだ．

本章では，以上の調査を経たので，用語を関係づけながら概観することとする．

3.1. システムとアーキテクチャ

「システム」も「アーキテクチャ」も当業界では不可欠な用語である．ただ「システム」の方がより広く使われているであろう．開発現場では業務のプロセス名を，システム開発，システム検証などと「システム」を用いて表し，またなにかを調査する場合には，調査の対象を「○○システム」などと，今度は対象○○に「システム」を付け加える．

また，「アーキテクチャ」は先に見たように，「システム・アーキテクチャ」などと使われ，「システム」と「アーキテクチャ」はよく連結して用いられる．そして，「アーキテクチャ」を簡略に「アーキ」と呼んだりする．

このように「システム」も「アーキテクチャ」も語が「熟して」とでもいうべきか，間断なく，また誤解もなく使われているように思われる．

しかし，ふと「アーキ」と簡略化して言う場合，個別の回路部品を指しているのであろうか，それとも構造を指しているのであろうか，などと疑問が湧く．出自はもともと後者であると思われるが，「このアーキは大丈夫？」などという場合，特定の個体（ここでは部品）を指していることも多い．

もちろん，語は語られる分脈で意味を持つ，と言われれば，この議論は大方片が付くのであるが，多少込み入った議論がある．

先に「変遷」の節で見たように，再確認的にいうとアークテクチャは，要素も指すが，構造を指す．構造を指すのは，アークテクチャが建築分野を出自にするからと納得できる．また「アーキテクチャ」がかつて「設計思想」と訳されたことも，「設計」という構造的側面に視点が置かれたせいと思われる．

しかし，「アーキ」と短く言う所為か，構造ではなく，要素，あるいは個体を指す，ちょっとした意味の変化が見られるような気がする．

他方「システム」はどうか．システムの定義の一つを見ると，システムは単なる要素の集まりではなく，要素どうしの秩序や関係を持つものとある[34]．要素が個別に，いわばバラバラあるのではなく，秩序を持ってきちんとあるものを，特に「システム」と呼ぶということである．

すると何か「システム」も，中に秩序は持っているが，一つとしてふるまうときには単独の個体に見える．すると，「アーキ」も「システム」も同じく個体という意味を指す．したがって，2つの語が連結しても，同じく個体を意味するのだから，同語反復ならぬ，同義反復にならないか，と少しうがった問いが湧いてくる．

しかし，実際には，「アーキテクチャ」もまた「アーキ」も，「システム」と連結されることで，固有の意味を取り戻すようである．つまり「システム・アーキテクチャ」は語が連結されることで，かえって両語の意味を際立たせ，ここでは「アーキテクチャ」は構造という本来の意味を取り戻すといえるのではなかろうか．

語が意味を，おそらくは文脈に応じて臨機応変に変化させ，しかし実際には固有の意味を「いざとなれば」取り返し，したがって誤解を生まないということは，いよいよ語が成熟していることを物語っているのかもしれない．このことが，特定の語が業界で長く生き残っている秘訣ではあるまいか．

ところが，このような機に応じた語の微妙な意味の変化は，すでにみたオントロジーと関係し，ややこしい問題を生む．先に Powers は記号接地をオントロジーの重要課題としている．記号接地とは，記号と記号の指示対象（存在）の関係をいう．記号の指示内容と指示対象が一致しているとき真という同定をわれわれはしている．

ディープラーニング（DL）技術では，この関係が怪しくなることがある．DL が例えば「シマ」と「ウマ」を判断できても，それだからといって「シマウマ」を自動的に合成し判断することはできない．DL には人間のできる認識の合成ができないというオントロジー（存在）に対する課題が残る．

この「シマウマ」の例は，上の「システム・アーキテクチャ」の話に似る．「アーキ」という「アーキテクチャ」の簡略表現が，単に語を簡略にしただけのはずが，オ

リジナルの長い語とは意味が変わってしまう．あるいは変わりやすくなる．

しかし，語が慣用的に使われてきた従来の語の結合に戻ると，従来からの本来の意味を取り戻す．このことは，語という記号と，記号の指示する事柄（意味）が一定ではなく，事情に応じて変化することを指す．「シマウマ」の例と異なり，議論の対象が目に見えない，「概念」の問題であるから，内面的で客観性に欠ける．これも議論をややこしくする．

この議論は語の成熟から発したものであるが，当然ながら，この議論は，文脈問題や複合語の問題で大方片づこう．

しかし他方，語には，生き延びてきた語ほど，語を認識する側（人間）とは独立に，語のサバイバル，概念の進化のような側面もあるのではないかという論点もありうるかもしれない．この論点は，オントロジーを形而上学と解した場合の概念の実在というような議論に近い．また数学の概念を，それを「存在」というフレームで語るときの議論とも似る．数学の世界はどういう様相で存在するか，という議論である．もちろん，「アーキ」などの語は経験的な場で使用されるものなので，同じに議論はできない．

本節の議論は，「システム」や「アーキテクチャ」のものであった．が，議論が DL などに派生したのも，これらの語が成熟し安定したうえで，場に応じたいろいろな意味を持つからと思われる．

3.1.1. System of Systems

挿話的にいえば，このごろ"System of Systems"を耳にする．語句は，複数の Systems を統括する System，という意味であろう．新たな語の導入は，導入を促す環境の変化にあるとすれば，すぐに思いつくのは，近来のシステムの「大規模・複雑化」である．この「大規模・複雑化」という表現は，システムに関する，サイズと内部構造の変化を表している．

"System of Systems"に戻れば，この"System of Systems"という語句は，この語句の先頭の"System"が，"of"のうしろに来る複数の"Systems"を対象物として統括的に制御すること，そして語"System"の階層は複数の"Systems"よりも上位の階層にあることを示している．

"System of Systems"や「大規模・複雑化」が少し誇張気味に表現している背後には，1つは安全性の課題がある．そして，その解法の1つとして STAMP が話題になる．STAMP は安全性の要求分析技法で，"Systems-Theoretic Accident Model and Processes"の略である．ここでは"Systems"に"Theoretic"を付記した方法論として提唱されている．

本論文は語に関する議論が中心なので，ここでの問いは"Systems-Theoretic"とは何か，である．この語は「複数システムの理論（化）」とでも解釈できるが，先に見たように，「システム」の意味は内部の（要素間に）秩序のあることを含んでいる．すると，「複数システムの理論（化）」とは，秩序をもった複数のシステムどうしを更に理論的（Theoretic）に秩序付けること，と理解される．秩序の秩序というような，概念の階層の違いに注意が要りそうである．

また，この概念の階層と関連するかもしれないが，STAMP の方法を適用するうえで，対象とするシステムが「全体は部分の単なる集合にはあらず」というホーリズムがアプローチの仕方と考えられている[40]．概念のいわば過不足の問題ともいえる．

STAMP はシステムに対して，事前にリスクとなる部分を見つけようとする．しかし，事故が発生して初めて，システムに事故の原因が潜んでいたことが明らかになる．すると，潜在的な部分が全体を超えていたということになる．全体，すなわちシステムの，その内部にある部分の秩序が十二分に把握しきれていなかった，ということでもある．

語の議論にもどる．「システム」などは，すでにみたように長くこの業界の語彙群の中核に位置している．他方，「システム」は語としてではなく，今度は安全という観点から，議論の対象となっている．語が単なる言葉としてではなく，具体的な現実の課題（この場合は「安全」）を持ち続けることが，この語の「生命」を生き生きとさせ，語のサバイバルを生き抜いている要因ともいえよう．

3.2. モデル

「システム」や「アーキテクチャ」の語が成熟し，取り立てて対象の要素と要素間の関係を明示的に区別しなくても誤解なく利用されているからであろうか，と，そんな解釈でもしたくなるほど，「モデル」が当業界に広く使用されるようになってきた．UML の普及によるのか，あるモデリングツールの称揚によるのか，「モデルベース」という表現もよく耳にする．

このモデルという語は，対象 What を表すのか，また方法 How を表すのかはっきりしない．モデルベースと称するツールが設計を支援するとすれば，それは How である．How で出来上がったものがモデルと

すれば，What である．

また数理の世界とソフトウェア工学の間では「モデル」の意味がむしろ逆とする指摘もある[35]．論理学や数学では，例えば公理から演繹された式は，「模範的な例」であるゆえ，「モデル」と呼ぶそうである．他方，ソフトウェア工学では事例を集めたもの，また集めてやや共通化し，模範となるものをモデルと呼ぶと指摘する．

前者が演繹的で，後者は一種帰納的であると対比できる．しかし，両者とも，模範的で模式的で，いずれも語「モデル」に含む意味を汲み取っているといえる．

さらに同書には，「モデル」という語が過去の長きに渡って使用されてきた息の長い語であるということが紹介されている．「これは「モデル」が他の用語と結合して使われてきたからであろう」としている．引用が長くなるが、同書には以下のようにある。

「，，，当然のことながら Software という語の出現頻度が他を引き離して多いが，それを除けば，つぎのような順序であった．---1．System，2．Design，3．Specification，4．Model，5．Analysis---

Model という語は第 4 位にランクされただけでなく，1975~94 年の 20 年，はやりすたりなく使われているという特徴をもつ．たとえば第 2 位にランクされた Design は，この 20 年のうちのとくに前半に出現頻度が高く後半は減少するが，Model にはそのような変動がほとんどみられない．これは「モデル」が他の用語と結合して使われてきたからであろう．結合する相手は，たとえば，life cycle， design，system，analysis，process ， product ， quality ， domain ， object ， computation，data，data flow， entity-relationship，function，application，state transition， . . . とおびただしくある．これらの結合相手にははやりすたりがあったが，モデルのほうは結合する相手を変えながら，一貫して人気を保ってきたのである．」[36].

「モデル」が結合しやすい語であるという特徴は，逆に定義される語ではなく，他を定義する基本的な(primitive)語であるということから来るのかもしれない．実際，意味の分析的議論を厳密に行う書に，「モデル」が未定義のままに使用されている[37]．同書に「分析のモデルの構成」という節があるが，同節にはモデルの定義や説明はない．モデルを構成要素たる分析的事項が 11 個記されているが，これら 11 個がモデルを指す．同書の著者には「モデル」が定義不要なプリミティブな語と認識されていたのであろう．

プリミティブな語は定義もされない（できない）語であるが，しかし説明は可能である．実際，同書[37]のモデルとされた分析的事項には説明がある．

「モデル」がプリミティブな語であるとすれば，「モデル」は他の語の説明や定義に使用され，当情報処理分野において様々な概念を作ってきたということもうなずける．

そして反面，他の語との親和性の高さは意味を曖昧にするというマイナス面もあろう．「モデル」はときに「バズワード」と非難される場合もある．しかし，説明すれば足りる．

3.3. オントロジー

新たな語の導入という観点からいえば，語"Ontology"が,組込みシステムの分野に登場したのは,システムの大規模・複雑化が課題となり始めたころと思われる．2005 年に Kaiya らの論文がある[26]．Kaiya らの論文では，組込みシステムを事例として取り上げていることから，「オントロジー」の組込みシステムへの適用と理解される．

Kaiya らは，要求仕様書に対して，オントロジーを適用することによって，要求仕様書から矛盾や不備を減らし，文書の品質を高めることを提案する．Kaiya らは，どの要求仕様書の文も，意味の原子的構成要素（atomic constituents of meaning）に基づいて解釈されるべきで，オントロジーはそのような知識を得るために使われる，とする．その結果，文書はどの関係者にも同じく解釈される原子的概念を持つものとされる．

Kaiya らのオントロジーは，要求仕様を満たす条件や観点からなるフレームワーク（枠組）ではないかと思われる．Kaiya らは次のように考える．

Kaiya らの原子的構成要素とは，「概念（concepts）」と「関係（relations）」である．要求分析は要求項目（items）に対して行うが，その際，概念と関係を（いわばフレームのように（筆者注））あてはめる．概念には，機能（function）や対象（object），環境（environment），制約（constraint），品質（quality）があり，環境の中には「actor」と「platform」がある．関係には，一般化（generalize），集合（aggregate），同義（synonym），異義（antonym），関係づけ（associate），矛盾（contradict），因果（cause），適用（apply），要求（require），支援（support）がある（上記は簡便さのために日本語では体言表現とした）．これらの原子的構

成要素の各々の視点から要求項目を検査する．すると，項目内容の不備や不足，項目どうしの関係が矛盾しているなどが見つかるという．

例えば，任意の項目が他の項目と関係して存在するものである場合，要求仕様書の中に片方の項目しかない場合には，関係して存在するという条件を満たさない．このような場合には，たとえば要求（require）という関係を働かせることにより，関連項目の不足をシステムは要求（require）することになる．

このような条件や観点の枠組，あるいは仕組みがオントロジーである．条件や観点が基本的なものゆえ，「存在」という基本的なことを表わし，かつこの中には推論規則も含まれるゆえ，これも基本的なことを表すといえる．よって，このような仕組みを，基本的な存在物，あるいは基本的な存在の方法（ありかた）であるとみなし，「オントロジー」と称するのであろう．

このようなオントロジカルな方法は何度も繰り返され，要求仕様書は正確になり，また詳細になる．これを精義化とでも呼ぼう．

ところで，Kaiya らは，このような精義化は，自然言語処理（NLP）を適用しなくても可能であるとしている．他方，先に取り上げた Web 意味論では，Gruber は先の文献[14]で，文「どの作家も読者に誤解されている」を，「その人が作家であるならば，どの人も，何人かの作家を誤解している読者を持つ」という風に述語命題に分解することを行っている．

以下が，文献[14]で，述語論理を適用し，自然言語の文を述語論理の構文で再構成する例である．

For example, the following is a KIF sentence that says, “All writers are misunderstood by some reader.”

```
(forall ?W
  (=> (writer ?W)
      (exists (?R ?D)
          (and (reader ?R)
               (document ?D)
               (writes ?W ?D)
               (reads ?R ?D)
            (not (understands ?R ?D))))
```

なお，上記 KIF の文は，文献[14]によれば，「Genesereth & Fikes, 1992」の考案した構文である．どの自然言語の文もこのような量を含んだ文に自動的に分解できるので，文どうしの導出が構文規則に基づき可能となる，といえる．

Gruber らは，自らの構文論（構文と推論規則），意味論（文の真偽の規則）を Web の領域に適用している．これらの構文論，意味論が Gruber らの「オントロジー」に他ならない．そして，これらの道具立てをもって，存在するものは表現可能なものとうたい，自分たちの世界（仕様）を Web 上に表現，すなわち存在たらしめている．

Kaiya らは，自然言語処理（NLP）を使用しないとしている．NLP は不使用としても，自然言語で書かれた要求仕様書を対象にしたい．要求仕様書の品質は上げる必要がある．自然言語の解析を行う要求仕様記述ツールも出ているが[41]，Kaiya らの要求仕様のための条件や観点を含むオントロジカル・フレームは要求項目の不足を補うなど，有効と思われる．今後が期待される．

語の用法に戻る．Feilmayr らは語「オントロジー」を用いているシステムや文献を詳しく調べ，「オントロジー」がこの2，30年，大いに称揚されてきたが，必ずしも成功していないといっている[16]．語の誤用や，またバズワードとしての「オントロジー」の使用もあるとも指摘している．ただし，これは Web やビジネスアプリを対象にしての調査である．

他方，オントロジーの適用領域を広げるべきとする，Pileggi らの最近（2018 年）の，いささか宣伝めく論文がある[27]．方法論の明示はないが，開発のライフサイクル，すなわち要求の開発から廃棄までの全プロセスに応じたオントロジーの方法の開発，適用，普及を促すものである．背景には，通信とモバイル技術の急速な発展に対して多くのソフトウェアを提供しなければならないが，その質が追い付いていないという危機感がある．「オントロジー」を広く検討する価値があろうかと思う．

4. 結び

「オントロジー」への，Pileggi らの呼びかけに対して思い起こすのは，「システム」である．「システム」は万能の概念として，コンピューター分野に適用される以前に，産業や組織，また個別科学などの多くの分野で期待されたという．何ごともシステム的にとらえよ，ということであったろうが，果たしてそれで十分であったかは推して知るべしである．1960 年前後のことのようだが，「システム」はいまや当たり前のことばとなって日常生活の中で用いられている．

「モデル」も「アーキテクチャ」も，どの分野でも使用

されている. しかし, 分野が変われば, 意味も異なる. 定義の難しい「モデル」などは, 分野ごとに指す意味が異なるともいえる.

しかし,「オントロジー」が「システム」のようにどの分野でもごく普通に使われるという光景はやや想像しがたい. Web 上のさまざまのサービスが, Gruber のいう存在するものは表現可能であり, したがってそのような光景が「オントロジー」, すなわち「存在のすがた」といわれてもなかなかピンと来ない.

しかし, 100 年後, あるいはもっと前かもしれないが, Web 上で, 自走車や, 自飛車(造語)などの存在物たちが, 自然界の営みのように, あるいは自然以上に十分制御されて, かつ自律的に動いていたとしたならば. そしてその光景に名をつけよ, とでもいわれたなら,「オントロジカル・ランドスケープ」, すなわち「存在論的景観」などと命名してしまうかもしれない.

空想めく話で恐縮だが, その言い訳に, 明治維新で招聘された, いわゆる雇われ外国人学者の話を引用させていただく. 30 年近く日本で医学を教えたベルツ氏は, 日本人の学問に取り組む姿勢に不満だったらしい. すぐ何に役に立つかと問いたがる態度に, 日本の学問は原理原則を大事にしないと映ったという[38].

こんにちでいえば成果を急ぐあまり基礎的な研究をしない, ということにでもなろうか. 構文論, 意味論, そもそも言語などが, 原理原則論になるのかは異議もあろう. そして中には, Gruber らの言うように, 存在するものは表現できるが, その逆, すなわち表現できるものは存在するのか, つまり表現により概念化(conceptualization)されたもの, すなわちそれは「概念(concept)」となると思うが, その概念は存在するのか, 存在するとすればどこにどう存在するのか, という問いも湧きそうである.

が, このような原理原則というべきか, さらにいっそう原理原則に戻るような議論が存在する一方で, 足下の, 例えば要求仕様書の問題に話が戻ると, 文も論証もよいが, 図などでもっと文書をわかりやすく, などという要望にも目を配らざるをえない.

するとまた, わかりやすさとは? また論理の厳密さとは? あるいは, どちらがソフトウェアの品質や生産効率に奏功するのか? しかも, システムなりソフトウェアなりのライフサイクルの中で, などと注文が出る. 議論が長く, ときに未整理のまま続きそうである. しかし, 前提となることがらを丹念にそろえ, 洗い出し, それこそ原理原則に立ち返って議論してみることが大事そうである.

いささか分を超えてしまったかもしれない. しかし, 当議論が多少面白い話題としていただけたら幸いである. 各位のご議論の美味しい肴たらん!

一品, 語の組合せパズルをつけさせていただいた.

表 語の組合せパズル:行→列と読む, △;可能?

	システム	アーキテクチャ	モデル	オントロジー
システム	○of	○	○	○
アーキテクチャ	△	△of	○	△of
モデル	○	△of	△of	△of
オントロジー	○	○	○	△of

参考文献

[1] Johnson, Lyle (1960). "A Description of Stretch" (PDF). p. 1. Retrieved 7 October 2017.

[2] P. Naur; B. Randell, eds. (1969). "Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7–11 Oct. 1968" (PDF). Brussels: NATO, Scientific Affairs Division. Retrieved 2012-11-16.

[3] Software Engineering Techniques: Report of a Conference Sponsored by the NATO Science Committee, B. Randell and J.N. Buxton, eds., Scientific Affairs Division, NATO, 1970, p. 12.

[4] P. Kruchten; H. Obbink; J. Stafford (2006). "The past, present and future of software architecture". IEEE Software. 23 (2): 22. doi:10.1109/MS.2006.59. S2CID 2082927.

[5] W.E. Royce and W. Royce, "Software Architecture: In-tegrating Process and Technology," TRW Quest, vol.14, no. 1, 1991, pp. 2–15.

[6] 石井 治, 最近のメモリー, 計測と制御, 1970, 9

巻, 8 号, p. 580-589, 公開日 2009/11/26, Online ISSN 1883-8170, Print ISSN 0453-4662, https://doi.org/10.11499/sicejl1962.9.580, https://www.jstage.jst.go.jp/article/sicejl1962/9/8/9_8_580/_article/-char/ja

[7] 特公昭 51-001381

[8] 情報処理 Vol. 12 No. 9 Sept. 1971 pp. ニュース pp.594

[9] 高橋延匡, 牛島和夫, 新田謙治郎, 淵一博, 土居範久, 山地克郎, 亀田尋夫 & 真子ユリ子. (1971). タイムシェアリング・システムの問題点をめぐって. 情報処理, 12(6).

[10] Rudolf Carnap, " Empiricism, Semantics, and Ontology" (in "The Logical syntax of Language A Study in Semantics amd Modal Logic"), The University of Chicago Press, 1947, pp.205-221.

[11] W. V. Quine, "On what there is" (in "From a logical point of view"), Harvard University Press, 1953, pp.1-19.

[12] Powers, David (1991). Preface: Goals, Issues and Directions in Machine Learning of Natural Language and Ontology. AAAI Spring Symposium on Machine Learning of Natural Language and Ontology. DFKI.

[13] Gruber, T. R. (1991). The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases. In J. A. Allen, R. Fikes, & E. Sandewall (Eds.), Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference, Cambridge, MA, pages 601-602, Morgan Kaufmann.

[14] 14>15 Gruber, T. R. (1993). "A translation approach to portable ontology specifications". Knowledge acquisition, 5(2), 199-220.

[15] Genesereth, M. R., & Nilsson, N. J. (1987). Logical Foundations of Artificial Intelligence. San Mateo, CA: Morgan Kaufmann Publishers.

[16] Feilmayr, Christina; Wöß, Wolfram (2016). "An analysis of ontologies and their success factors for application to business". Data & Knowledge Engineering. 101: 1–23. doi:10.1016/j.datak.2015.11.003.

[17] 元田浩, & 吉田健一. (1989). 定性推論と深い推論 (< 特集>「定性推論」). 人工知能, 4(5), 538-546.

[18] 元田浩, & 吉田健一. (1991). 定性推論の応用: 定性推論の知識獲得への応用. 情報処理, 32(2).

[19] 牧野武則. (1991). 語彙の概念と知識について. 情報処理学会研究報告自然言語処理 (NL), 1991(37 (1991-NL-083)), 105-112.

[20] 中島震, 小泉昌紀, & 松本正雄. (1993). 問題領域指向リエンジニアリング-GUI ソフトウエア. 情報処理学会研究報告情報システムと社会環境 (IS), 1993(4 (1992-IS-042)), 67-76.

[21] ティヘリノ A. ジュリ, 池田 満, 北橋 忠宏, 溝口 理一郎, タスクオントロジーと知識再利用に基づくエキスパートシステム構築方法論 : タスク解析インタビューシステム MULTIS の基本思想, 人工知能, 1993, 8 巻, 4 号, p. 476-487, 公開日 2020/09/29, Online ISSN 2435-8614, Print ISSN 2188-2266, https://doi.org/10.11517/jjsai.8.4_476, https://www.jstage.jst.go.jp/article/jjsai/8/4/8_476/_article/-char/ja

[22] Norbert Wiener (1948), Cybernetics, The technology Press, John Wiley & Sons, Inc. New York. Hermann et Cie, Paris 同書訳,『サイバネティックス』(池上止戈夫ら訳), 岩波書店 (1962)

[23] W. Ross Ashby (1956). An Introduction to Cybernetics, Chapman & Hall. 同書訳,『サイバネティクス入門』(篠崎武ら訳), 宇野書店(1967)

[24] Kenneth A. Fox, Marc P. Pasture, and Peter S. Showman (1972). A Human Interface for Automatic Measurement Systems, Hewlett-Packard Journal April 1972 Volume 23 Number 8

[25] Shane Dickey (1972). Distributed Computer Systems, Hewlett-Packard Journal November 1972 Volume 26 Number 3

[26] H. Kaiya and M. Saeki, "Ontology based requirements analysis: lightweight semantic processing approach," IEEE,Fifth International Conference on Quality Software (QSIC'05), 2005, pp. 223-230, doi: 10.1109/QSIC.2005.46.

[27] Pileggi, Salvatore F.; Lopez-Lorca, Antonio A; and Beydoun, Ghassan, "Ontology in Software Engineering" (2018). ACIS 2018 Proceedings. 92. https://aisel.aisnet.org/acis2018/92

[28] Basili, Victor R., and H. Dieter Rombach. Tailoring the software process to project goals and environments. 1986.
[29] Boehm, Barry W. "Seven basic principles of software engineering." Journal of Systems and Software 3.1 (1983): 3-24.
[30] Zelkowitz, Marvin V. "Resource utilization during software development." Journal of Systems and Software 8.4 (1988): 331-336.
[31] "OMG Unified Modeling Language (OMG UML), Superstructure. Version 2.4.1". Object Management Group. Retrieved 9 April 2014.
[32] Rumbaugh, James, et al. Object-oriented modeling and design. Vol. 199. No. 1. Englewood Cliffs, NJ: Prentice-hall, 1991.
[33] 富士通, FACOM 230 28 解説, 1973
[34] 松田正一,『システム理論序説』, オーム社, 1971, p.8
[35] 玉井哲雄,『ソフトウェア工学の基礎』, 岩波書店, 2004, p. 47
[36] 玉井哲雄,『ソフトウェア工学の基礎』, 岩波書店, 2004, p.48-49
[37] 永井茂男,『分析哲学－言語分析の論理的基礎－』, 弘文堂, 1969, p.206
[38] E.ベルツ,『ベルツの日記』(上), 岩波文庫, 2020, pp.238-240
[39] https://ja.wikipedia.org/wiki/System/360
[40] 兼本茂, 安全性向上員会, 組込みシステム技術協会, 2022 April,『組込みシステムの安全解析と障害原因診断の統合アプローチ』, IPA, 2015
[41] N. Urushibara and C. Sasaki, "Integration of Two Kinds of Syntax for Requirements Description and Its Future Development," 2018 1st International Workshop on Easy Approach to Requirements Syntax (EARS), 2018, pp. 3-8, doi: 10.1109/EARS.2018.00007.