2015.8.16

知平

知働化のゆくえ

大槻 繁
知働化研究会運営リーダ

本節は、2014 年の元日に執筆した同名の作品群『知平』[Otsuki2014a]を再編集・加筆したものです。毎年元日には、その時の、現状認識、心づもり、今後の活動の方向性について、赴くままに書いています。書くということは、考えを進めるということでもあります。

タイトル**「知平」は、「知」の「地平線」といった意味合いの造語**です。どこまで走っても行き着かない世界が広がっていることを象徴しています。この作品群全体の骨組みは、哲学から始めて、科学、工学、そして、経済・ビジネスへという流れに従って書き進めています。最近私が提唱している**「ソフトウェア参謀（ブレーン）」**[Otsuki2014b]を最後に位置づけてあります。こういったコンサルティングソリューションに至る思考過程を表現しており、この過程そのものが「知働化」の実践といってもよいと思っています。

１．哲学のすすめ

学生の頃から、学問とは何かとか、真理とはとか、形而上学的な思考にはまる癖があり、心理学や哲学書を読みあさった時期がありました。心理学については、フロイトやユングのみならず、その後の近代実験心理学についても読みましたが、胡散臭さがぬぐいきれず、今一つ納得感が得られませんでした。哲学についても、プラトン、アリストテレスに始まりデリダ、ローティといったポストモダンなものまで一通り眺めましたが、こちらもぴんとくるものは見つかりませんでした。

大学の時にコンパイラの研究をしていく中で、プログラミング言語そのものに興味を持ち、徐々に言語設計や仕様記述手法などに興味がでてきて、その等を基礎づける意味論や形式手法、言語哲学を深掘りしていきました。その折に、見出したのがヴィトゲンシュタイン（Ludwig Josef Johann Wittgenstein）の哲学です。その昔、ヴィトゲンシュタインの著作、解説書などを、たくさん読んだ記憶があります。特に、**『論理哲学論考』**[Wittgenstein1921]と呼ばれる小冊子を見たときの感動は今でも忘れられません。言語の記述、命題論理の基礎付けを、これ以上ないというほど厳密に、かつ、余計なことをそぎ落とした記述で、英語版（原文はドイツ語）で 70 頁程度のものです。内容は、以下の７つの主要な文から成っています。

1. **世界は、成り立っていることの総体である。**
   The world is all that is the case.
2. **成り立っていること、すなわち事実、とは、事態の成立である。**
   What is the case – a fact – is the existence of states of affairs.
3. **事実の論理的像が思念である。**
   A logical picture of facts is a thought.
4. **思念は有意味な命題である。**
   A thought is a proposition with a sense.
5. **命題は要素命題の真理関数である。**
   A proposition is a truth-function of elementary propositions.

6.　**真理関数の一般式は、[p,ξ,N(ξ)]である。これが命題の一般形式である。**

The general form of a truth-function is [p,ξ,N(ξ)].

This is the general form of proposition.

7.　**人は、語り得ぬものについては、沈黙しなければならない。**

What we cannot speak about we must pass over in silence.

p は要素命題の集合、ξは全命題集合、N（ξ）は全命題の否定連言

当時としては、不毛な形而上学的議論に意味が無いことを示したとても過激なものです。最後の文の「**語り得ないものに対して沈黙すること**」は重要なメッセージですが、これと同時に、「**語り得るものは明瞭に語らなくてはならない**」ということも示しています。世界を言語によって切り取り、言語による記述の構造と、思念（像）の構造とが同型であることも言っています。

ジャクソン（Michael Jackson）が提唱している JSD（Jackson System Development）でも、「**言語という眼鏡をかけて、実世界を見る**」という表現がでてきますが、同じことを示していると思います。**ソフトウェアエンジニアリングは、最終的にはコードに変換され、あるいはコードに帰着させていくプロセスを対象とした、語り得る「記述」に関する学問**なのです。

ヴィトゲンシュタインは、**『論理哲学論考』の考え方を、後に捨て去り、『哲学探究』[Wittgenstein2003]という一連の書き物の中で、「言語ゲーム」という概念に到達**しています。言語を生活形式の中での多様な活動と見なし、ダイナミックに変化し、コミュニケーションによって共通化・共有化していく概念の合意形成のプロセスを捉えています。**言語の意味は、日常的な「用法」によって定義される**のです。「ゲーム」という呼称は、いささか違和感を覚えるかもしれませんが、**コミュニケーションを行う際の社会的な規則（ルール）が語り得ない形で存在する**ことを暗に示しています。

私は、ヴィトゲンシュタインの**『論理哲学論考』から『哲学探究』への移行が、学問、そして、世界観の大きなパラダイムシフト**だと考えています。ヴィトゲンシュタインがこの後期哲学に至ったのは 1930 年台前半だと思いますが、多くの哲学者、研究者がこの考え方、あるいは、それに近い考え方に至っています。

特に、掲げておきたいのがデザイン論の分野でのクリッペンドルフの提唱です。『意味論的転回』[Krippendorff2006]では、ヴィトゲンシュタインの言語ゲームを引用しつつ、人工物（アーティファクト）のデザインに、利用者の視点を位置づけ、それを制御することが重要だと説いています。当然、ソフトウェアも人工物の一つですから、クリッペンドルフの考え方を適用することができます。

もう一つの流れが、パース（Charles Sanders Peirce）、ジェームス（William James）、

デューイ（John Dewey）が提唱した**米国プラグマティズムの流れ**です。プラグマティズムとヴィトゲンシュタインの哲学との関係を、言語や数学に関わる哲学の研究で著名なパトナム（Hilary Whitehall Putnam）が『プラグマティズム』[Putnam1995]で解説しています。ヴィトゲンシュタインの**「言葉が用法によって定義される」**ということは、プラグマティズムの格率（格言）として**「対象の概念を明晰にとらえようとするなら、その対象の効果を考察せよ。すると、この効果についての概念は、その対象の概念と一致する。」**という形にまとめられます。

さらに、これを日本の産業界を元気付けるための基軸にしようという提唱が、『プラグマティズムの作法』[Fujii2012]で述べられています。専門化して縦割りになっている領域の壁を打破するために、それぞれの専門領域が外からどのように見えるか、どのように他人の役に立つかどうかを真摯に見直そうということを提唱しています。このことを端的にまとめると以下のようになります。壁に貼っておきたい教えです。

- **何事に取り組むにしても、その目的を見失わないようにする。**
- **その目的が、お天道様に対して恥ずかしくないものかどうかを問い続ける。**

ソフトウェアの分野は、まだパラダイムシフトが十分とは言えませんが、少しずつ変わっていくでしょう。この流れを加速していくためには、「ソフトウェア哲学」を中心とした研究領域を立ち上げて探求していくのがよいと思っています。ソフトウェア哲学の中心課題は、以下のようなものになるでしょう。

- **ソフトウェアとは何か、世界観を問う**
- **ソフトウェアに関する潮流をつかむ**
- **ソフトウェアに関する研究・技術開発の接近の方法を確立する**

ソフトウェアに関する世界観は、ソフトウェアの利用面、実世界との関係に中心が移ってきているので、多様な世界の捉え方やその中でのソフトウェアの位置づけについて検討していかなくてはなりません。また、「実行」という概念も、計算理論で言う実行のみならず、人間の行動、遺伝子操作、粒子や量子の時間的推移なども視野に入れていく必要があります。

ソフトウェアエンジニアリングに関する技術は、さまざまな言語処理系、モデル検証系、フレームワーク、開発・保守プロセス、プロジェクトマネジメント・ファシリテーション手法などが提唱されてきています。技術の成熟度や実践状況、今後の潮流の方向性などについて検討していく必要があります。また、分野を横断した学際あるいは業際的な活動が多くなってきています。おそらく今後、学会やコミュニティ間の

交流はもっと多様化していくと予想しています。

ソフトウェアに関する研究パラダイムは、従来は純粋な基礎理論、応用技術、実践適用といったリニアなスタイルが多く採られていましたが、大手企業の基礎研究所の閉鎖や、生産技術部の縮小などに見られるように、技術開発に関する社会的な役割分担も変化しています。さらに、**理論と実践の並行化、ソフトウェアに関わる専門家の種類は、従来のアナリスト、プログラマ、テスタなどの分担から、もっと他の新しい職種が生まれていくのに従って、研究アプローチも変わっていく**と考えています。

２．仏教からのヒント

仏教に興味を持ったのは、最近のことです。2008 年の正月に、濱勝巳さんが「ソフトウェアは、空（くう）である」と言い始めたのが契機になっています。私は仏教徒ではありませんが、その後、啓蒙書を何冊か読み[Hashizume2013] [Go2011]、共感するところもあり、腑に落ちる事項も多いため、少しずつ勉強し、自らの研究の基礎付けとしても活用し始めています。

私が仏教の考え方に価値を置いている理由は、以下の３つに集約されます。

- **西洋のものを、日本人が腑に落ちる形にするヒントが得られる**
- **暗黙知、あるいは、語り得ないことを扱う方法を提供している**
- **考え方を広めていくための社会的な仕組みのヒントが得られる**

最近では、仏教は葬儀の時にしかお坊さんにお目にかかりませんが、もともとは２千年以上昔に、お釈迦様（BC463～BC383）が始めた宗教です。**キリスト教が「救い」の宗教だとすれば、仏教は「覚り」の宗教**と言えます。キリスト教文化では、神が中心であり、神との関係を起点とする「契約」が人の行動にさまざまな制約を課しています[Hashizume2011]。現代のビジネス活動もその枠組みは組織や個人との間の取引契約を結ぶことが基底となっています。この意味で、グローバル化すること、西洋文化を受け入れるということは、知らぬ間にキリスト教の考え方に影響を受けていると言えます。

一方、日本は、八百万の神、自然とともに生きる文化で、人の関係性も和を中心とするもので契約の概念はありません。宗教観は柔軟で、こだわりもなく、仏教が大陸経由の儒教思想と融合したものとして伝来してきても、素直に受け入れています。

日本人が異国文化を柔軟に取り入れるころができる理由は、国民性に依るところが大きいようです。イングルハート（Ronald F. Inglehart）の世界価値観調査によれば、世俗・合理的価値 × 伝統的価値、生存価値 × 自己表現価値　の２つの軸で世界の国々のアンケート調査結果を俯瞰すると、**日本人は、世俗・合理的価値に極端に重き**

を置き、自己表現価値は先進国の中では相対的に低いところにマッピングされています[Tachibana2012]。従って、あまり宗教や思想に固執することなく、役に立つものは貪欲に、柔軟に取り入れるところが日本人の国民性と言えます。表面的な形と実質的な行動とを使い分ける器用さも備えています。

さて、肝心の仏教についてですが、考え方として参考になるのは、**お釈迦様の原点としての原始仏教**です。「小乗」という差別的な呼び方をされることもありますが、思想としては、「大乗」より純粋です。自らが覚ることを重視する。まさに覚りの思想です。**一切皆苦、諸行無常、諸法無我という三法印に集約**されています。

仏教では、自らの生き方を見つめ、心の奥底を内観していきます。「無我」を目指しますが、我の正体は五蘊（ごうん）である肉体、感覚、表象、意志、意識ですし、感覚器官の五根（ごこん）として眼根、耳根、鼻根、舌根、身根があります。覚りに至る道は、中道と呼ばれ、八正道としての正見、正思、正語、正業、正命、正精進、正念、正定が示されています。基本的な態度の要点は、以下の通りです。

- **世界を因果律によって理解する**
- **超越者の存在を認めない**
- **努力（修行）の領域を、肉体ではなく精神とする**

この人間中心の考え方は、科学的な態度と整合していますし、科学にとってもキリスト教の呪縛に比べるとずっと親和性が高いと言えます。**仏教学者の佐々木閑教授は、この流れを「人間化」と呼んでいます**[Sasaki2006]。

大乗仏教は、お釈迦様の入滅後 100 年くらい経って、教義解釈の部派が乱立して分化していったものです。大乗は、利他、仏を目指す未完成の存在としての菩薩があり、そして、すべての民衆を済渡することを説いています。日本の鎌倉時代に伝来したのは、この大乗仏教になります。多くの宗派に派生したこと、そして政治とも結びついていきました。この展開については、宗教のみならず思想を展開していく際に参考になる点も多いと考えています。

よくあるエセ宗教の楽して極楽に行けるとか、クリック一つでプログラムが生成されるといった言葉は大衆の心をつかむようです。霊魂の不滅とか、輪廻転生なども、大衆化する際の魅力の一つになります。何ごとかを組織的に展開していく際には、何らかの意味での**「救い」や「ご利益（りやく）」によって求心力を保っていくことが戦略としては有効**だと考えられます。

３．知の基盤としての数学

数学の面白さを知ったのは中学生の頃だったと思います。初等幾何学に夢中になった覚えがあります。ちょっとした補助線を引くことによって、証明がエレガントにできたり、定規とコンパスを使って美しい図形を描いたりして、世界が開けたような気がしたものです。つい最近まで存命していたコクセター（Harold Scott MacDonald Coxeter）[Coxeter1969]は、晩年まで初等幾何学の研究や啓蒙活動を積極的にやっていました。大学に入り教養としての解析学や線形代数は、さほど感激はなかったのですが、情報科学科での離散数学や位相幾何を学んだあたりから、理論の美しさ、対称性、表現力の豊かさに魅力を感じていたように思えます。

数学は、人間の知を表現したものです。公理や前提条件を設定して、その下で成立する命題を定理として表すので、普遍的な知であることは間違いありません。数学で表せない知があることは確かですが、数学的な定式化を図る努力は常に行うことは意味のある活動だと思っています。数学は裏切らない永遠の知です。

コンピュータ、ソフトウェアの動作原理は論理や計算の理論による数学的な下支えがあるから成立しています。その意味で、ソフトウェアエンジニアリング、コンピュータサイエンスは、もっと数学的なアプローチに重点があってもよいように感じています。

ソフトウェアの素晴らしいところは、人間が自由に概念を操作し、全く新しい世界を構築できるところにあります。人間の抱くイメージ、イデアを扱うという点では数学と同じです。

- **新しいソフトウェア、新しい人工物（アーティファクト）を生み出す創造的な活動は、発見なのでしょうか？**
- **それとも、発明なのでしょうか？**

このような問いは形而上学に属する哲学的な事項だと思いますが、これを検討すると、いろいろなことが見えてきます。

数学の世界で、**イデアの世界が存在し、真理、定理が実在として在り、人間はそれ等を発見していると見なす立場をプラトン主義**と言います。数学的真理は人類が出現する前から在り、人類が滅亡しても在り続けるという考え方です。いわば神が創った完全な世界の真理を、不完全な人間が覚っていくというものです。実際に、多くの数学者はこういった感覚を持っているようです。

プラトン主義者によれば、自然数、整数、点、線も実在するものです。しかし、人類が進化する過程で、数を数えることや、視覚的な空間を認識することが、生き残りの条件として有効であったからこういった概念を身につけることが必要だったと見

なすこともできます。もし、犬が進化したとしたら、嗅覚に関わるような、位相空間の概念が実在として認識されていたかもしれません。つまり、数学というのは、人間の生活の中から生まれた知ではないでしょうか？　人類が居なければ、我々が蓄積しているような意味での数学は無いのです。

一方で、数学には、虚数、非ユークリッド幾何学といった人間の日常とは関係ないような発明もあります。虚数やそれに伴う複素関数論が無ければ、量子力学は無いですし、非ユークリッド幾何学やリーマン幾何が無ければ、一般相対性理論は語ることができません。

こうして見ると、発明か発見かという問いは以下のように集約することができます。

- **数学の公理には、人間の日常感覚の延長線上の発見的な公理系と、人間が発明した公理系とがあることになります。**
- **問題は、その公理の集まりを材料として、その先にある世界を説明し、記述するための定理を選択して使う発明的な構成力が必要になることです。**

我々が、言語を使う時に、語彙や形式といった材料を使い、それ等を構成してひとつの物語やまとまった作品を作ると見なすことができます。おそらく数学は、言語と同様の一つの表現方法なのです。これは、建築、音楽、絵画、ファッションなども同じように考えることができます。

この材料と構成物との関係は、どちらが先ということではなく、相補的な関係にあります。例えば、物理学と数学との関係は、相互に絡み合っています。ニュートン力学、マックスウェルの電磁気学、アインシュタインの相対性理論、ハイゼンベルクの量子論、ウィッテンの弦理論といった発展の過程の中で、数学としての解析、幾何、位相、多様体、代数なども同時進行的に発展してきています。おそらく物理世界を見る目や観点を、数学は提供していますし、物理世界を説明するために数学は新しい概念を開発し、定式化していきます。**数学的活動は「言語ゲーム」**なのです。

４．新ソフトウェア宣言その後

ソフトウェアとは何か？　という問いは、永遠のテーマです。これからの時代のソフトウェアとはどういうものになっていくかという点については、2010 年の夏に**『新ソフトウェア宣言』**[Ses2010]としてまとめ、以下の 7 つの観点を得ました。

1. **ソフトウェアは、数学的理論探求の上に成り立つ**
2. **ソフトウェアは、部分に還元することが不可能な全体である**

3. ソフトウェアは、実行可能な知識である
4. ソフトウェアは、学びの副産物に過ぎない
5. ソフトウェアは、制約条件下で創造される美しい人工物である
6. ソフトウェアは、富を生む経済活動の資源である
7. ソフトウェアは、言語ゲームである

その後、表現や順序を練り直し、最終的に以下のようにしました[Hama2014]。例えば、「制約条件下で創造される」は、「美しい人工物」に含意されていますし、「部分に還元することが不可能」は、端的に「分解不能」ということです。「経済活動」は、「富を生む活動」です。「1.～7.」の項番は、「月～日」にし、さらに、それぞれの曜日に相当する陰陽道や社会通念上の意味を配慮し、順序を設定しました。美しい「月」、複雑な振る舞いの「火」、成長のための「水」、知識の「木（樹）」、富を生む「金」、下支えする「土」、そして、遊ぶ（ゲーム）「日」といった具合で、記憶にも残りやすいでしょう。

月：ソフトウェアは、美しい人工物である
火：ソフトウェアは、分解不能な全体である
水：ソフトウェアは、学びの副産物に過ぎない
木：ソフトウェアは、実行可能な知識である
金：ソフトウェアは、富を生む資源である
土：ソフトウェアは、数学理論探求の上に成り立つ
日：ソフトウェアは、言語ゲームである

これ等の説明は、『ソフトウェア宣言概説』[Otsuki2010]に詳細を記していますが、2010 年当時以降の認識の進展について、以下に述べておこうと思います。

月：ソフトウェアは、美しい人工物である

この宣言文中で「美しい」という言葉を使っているところがポイントです。ブルックス Jr の『デザインのためのデザイン』[Brooks2010]によれば、テクニカルデザインにおける美は、以下のように集約されています。

- エレガント：少しの要素で多くを達成すること
- 無駄があること：冗長性
- 明瞭性：構造概念が一目瞭然であること
- メタファ：馴染みのある比喩による理解

- **一貫性：直交性（独立なものの組み合わせになっている）、**
  本来性（重要で無いものは入れない）、
  汎用性（本質的なものを制限しない）

これはなかなか含蓄のある項目です。私も直感的には、シンプルであること、対称性があること、直交していることなどが美の根底にはあり、さらに、茶の湯の千利休にあるような**「わざと欠けを作って不完全な美を演出する」ようなことも大事なこと**だと思っています。これは、キリスト教の三位一体で言う「父」と「子」と、もう一つ「精霊」があることによって、過剰性が担保され、世界に広がりがでるということに通じているとも言えます[Nakazawa2006]。物理学の理論も同様で、信ずるに足る美しさが要求されます。ヒッグス機構の理論では、弱い力の対称性が破れる場合があることを主張していて、対称性の理論のエレガントさを保ちつつ、現実世界の予言も行っている魅力ある美しいもので、多くの物理学者が信じている理論です[Randall2012]。

美に代表される「真善美」は哲学的な問題です。このような哲学的問題を、神様や宗教抜きで人間化して論じるのは、難しいことです。おそらく現象学的なアプローチが唯一の方法かもしれません。**真というのは「ほんとう・うそ」という科学的な事項、善というのは「よい・わるい」という社会的な事項、美というのは「きれい・きたない」というエロスの問題です**[Takeda1993]。

火：ソフトウェアは、分解不能な全体である

ソフトウェアは、実世界の中に置かれるマシンですから、実世界の振る舞いに左右されます。実世界は自然界であっても、社会であっても複雑系の様相を呈しています。複雑系に関しては、多くの研究者が取り組んでいますが、総じて説明困難な複雑な現象を研究しているものの、まだまだ発展途上の領域で、これといった成果が得られているようには思えません。

複雑系とは、**「数多くのコンポーネントから構成されながらも、単純な運用規則を持つのみで中央制御機構を持たない大規模なネットワークから、集合体としての複雑な振る舞い、複雑な情報処理や、学習、進化による適応が生じるシステム」**と定義されます[Michell2009]。また、内部・外部の制御装置やリーダーの存在なくして組織化された振る舞いを生むことを「自己組織化」と言い、単純な規則によって予測の難しい複雑な振る舞いが巨視的に生じることを「創発」と言います。**複雑系は、この自己組織化と創発が生ずることが特徴**だと言えます。

ソフトウェアの世界も複雑系の局面は多く出現すると考えられ、この世界での予測

や意思決定を行う方法を模索していく必要があります[Johnson2009]。**ソフトウェアシステムの改訂や、新しい要求への対応、フィードバックの方法、関与者やチームの行動規範などの局所的な変化が、システム、ひいては、その周辺の社会・経済に対して大局的な影響を及ぼす**と考えられます。一方で、複雑系とコンピューティング（計算）との関係も新しいものが出現してきて、遺伝的アルゴリズムや自己増殖型オートマトン、ゲーム論を取り入れたシミュレーションなどを使った予測や制御の方法が整備されてくるかもしれません。いずれにせよ古典的なソフトウェアエンジニアリングの知見だけでは、立ち行かなくなるのは明らかです。

水：ソフトウェアは、学びの副産物に過ぎない

我々は日々、仕事をし、勉強し、遊び、生活をしながら、いろいろな事柄の理解を深め、学習し、覚っていきます。このことにより人は安心します[Yasutomi2011]。**何か目標を設定して、それを実現するという考え方では、人は幸福にはなれません。**成長し続けること、学び続けることが大切です。社会というのは、常にそれぞれの人々が学び続けることが常態なのです。その人々の活動の中でソフトウェアは、開発され、維持され、廃棄されるものです。

では、学習し続ける社会的な仕組みをどのように作っていけばよいのかという問いに対する一つの解が、コミュニケーションの方法を提供することです。**「コミュニケーションとは、お互いに学習するプロセスである」**ということを、安冨歩は『経済学の船出』[Yasutomi2010]の中で述べています。このアイディアに基づいて、知働化研究会では、**「知のフリマ」**[Otsuki2012b]というコミュニティ活動の方法を提案し、展開しています。これはソフトウェアのみならず、いろいろな異文化の人々が互いにコミュニケーションを通じて刺激し合い、成長していく場を提供しています。

社会的・組織敵な活動の仕組みで、もう一つの局面が、知の蓄積をどのように図っていくかという問題があります。**「知のフリマ」でもコミュニケーションの直接の成果は、各自の成長ということになりますが、これを何らかの形で蓄積していくことも検討していく必要があります。**私は、これ等を総称して**「知のプラットフォームの構築」**という問題設定をしています[Psec2013b]。

知のフリマは一つの実験的な活動ですが、こういった社会的な仕組みとソフトウェアに関する活動とを連係させて、新しいソフトウェアづくりに取り組んで行くにはどうしたらよいかというのが、現時点の問題認識です。

木：ソフトウェアは、実行可能な知識である

元来、『新ソフトウェア宣言』の検討は、知働化研究会の活動の一環として企画したものです。この**知働化研究会のコンセプトが「実行可能な知識」です**[Yamada2004, 2009]。この考え方に基づき、従来のアジャイルプロセスの意義を再検討し、開発プロセスを**ソフトウェアが置かれる世界からのフィードバックを明確に位置づけたものが『ΛＶプロセス』**です[Otsuki2012a]。

現段階で実行可能知識であるソフトウェアについての研究課題は以下の項目があります。

- **ソフトウェアが創造的人工物（artifact）であり、これをデザインする方法を確立する。そのために、「創造性」に関する理論と実践方法を探求しなくてはならない。**
- **ソフトウェアの利用面、実世界側でのプロセスを明確にする。特に、実世界側の要素やプロセスの「抽象化」に焦点をあてて探求しなくてはならない。**
- **「実行」という概念そのものを、明確にする。従来のソフトウェアでの実行概念に加え、計算、処理などの社会的な意味、人や組織を含んだ意味での実行概念を探求しなくてはならない。**

金：ソフトウェアは、富を生む資源である

富を生むということは、端的に言うと、ビジネスに組み込まれて価値を生み出すということになるのですが、重要なビジネスモデルが、**「プラットフォーム戦略」**だと考えています[Hirano2010][Negoro2013]。大まかに言うと、**プラットフォーム戦略とは、生産者/消費者、提供者/利用者の取引や交流の場を設定し、その場代を収益の柱とするものです。『経済学の船出』の中でも「関所資本主義」と評されているもの**でもあります。E マーケットプレイス、銀行、自動車会社、大学などもプラットフォームビジネスと見なすことができます。Amazon、Apple、Google が典型例です。

ソフトウェアとプラットフォーム戦略との関係は、今後検討していく必要があると思っています。検討すべき事項は、以下の通りです。

- **プラットフォームビジネスを構成する基本要素の洗い出し**
- **ビジネスモデルとシステムアーキテクチャの決定**
- **開発/利用のフィードバックの方法、特に、指標の開発**

これ等の検討がまとまってきたら、アジャイル経営/開発の手法として提示することが可能になるでしょう。

土：ソフトウェアは、数学理論探求の上に成り立つ

ソフトウェアづくりの本質が人間の主要な能力の一つである概念操作であると考えています。いろいろな概念を数学的に定式化していく方法を確立すること、逆に、数学的な概念がどのような人間の認知メカニズムに依っているのかなどを究明していかなくてはなりません。レイコフ（George Lakoff）の『数学の認知科学』[Lakoff2001]では、数学的な概念である点、線、空間、集合、無限、複素数などが、**人間の身体的な経験に基づいてどのように獲得されてきたか**を説明しています。また、**人間の新しい抽象的な概念獲得の思考パターンとしてブレンド（混合）とメタファ（隠喩）が本質的**であることを主張しています。このあたりは、ソフトウェアの世界での新しいビジネスモデルや仕組みをデザインしていく創造的活動の接近法を示唆していると考えています。

数学には大別して幾何学と代数学とがあります。素朴な言い方をすれば、幾何学は右脳の直感的な空間認識や視覚に関わっており、代数学は時間や演算・計算に関係しています[Atiyah2010]。両者相補的な関係にあります。幾何学はユークリッドに始まり、ニュートン（Isaac Newton）力学、位相幾何学のポワンカレ（Jules-Henri Poincaré）、シンプレクティック幾何学の牽引者としてのアーノルド（Vladimir Igorevich Arnol'd）へといった系譜で捉えることができます。一方、代数学は、ライプニッツ（Gottfried Wilhelm Leibniz）、ヒルベルト（David Hilbert）、ブルバキ（Nicolas Bourbaki という架空の名前を冠した集団）の系譜です。代数は一つの公式を作ることを目的としていて、計算によって答えが得られるようなマシンを作ることになります。ソフトウェアは、幾何学と代数学という対峙から言うと、代数の世界の手段として捉えられると思います。両者は相補的ですが、**アイディアや新しい概念創造の源流は幾何学寄りのところに在ると確信**しています。

日：ソフトウェアは、言語ゲームである

ヴィトゲンシュタインの提唱した**「言語ゲーム」の考え方は、産業界を活性化するための思想としての「プラグマティズム」にも通じる**ものがあります[Putnam1995]。『プラグマティズムの作法』[Fujii2012]では、専門化して縦割りになっている領域の壁を打破するために、それぞれの専門領域が外からどのように見えるか、どのように他人の役に立つかどうかを真摯に見直そうということを提唱しています。このコンセプトに従って、コミュニティ活動として P-sec（実践的ソフトウェア教育コンソーシアム）のオープンフォーラムを開催しました[Psec2013a]。コミュニティ活動を含め

て人間社会の仕組みをデザインする際に多様性を抱擁することはとても大切なことです。

ソフトウェアが､利用面を含む文脈(コンテクスト)が中心となってきていること、時間経過とともに適応・変化していくことを考慮するには、**ソフトウェアを取り巻く文法、規範、制度設計をしていかなくてはなりません**。ハート（Herbert Lionel Adolphus Hart）は、言語ゲームを法律の領域に適用して、法の制定や改訂のプロセスを二層のルールのモデルで提案しています[Hashizume2009]。社会制度設計では、たとえ法律の世界であっても語り得ない世界があり、しかも、その世界をある程度制御する方法が必要になってきます。

５．これからの予測技術

主としてソフトウェア開発（システム構築）に関する見積り予測は、プロジェクトデータの蓄積に基づいて、予測対象のプロダクトの規模・品質などの特性に応じて、どのような工数、期間がかかるかを算出するものです。**基礎になる理論は、ベイズ統計と非線形の多変量解析**です[Kikuchi2008]。

データを採取して、そこから何らかの予測をする理論らしきものがあるとすれば、統計学になるわけですが、多変量解析は複数の変量間の関係から方程式を得るものです[Matsubara2007]。この式が得られてしまえば、パラメタを入力したら、所望の予測値を得ることができるということになります。ところが､データが欠損していたり、観測できるデータが限られていたりと実践上の課題と向き合うことになります。

統計学は、複数の現象間の因果関係を直接は扱わないのですが、この関係を推定する原理を与えている統計学の中での唯一の理論がベイズ統計です[McGrayne2011]。

ソフトウェアの世界で、いくら定量化され、方程式が得られたとしても、予測技術の本質的な問題は、もう少々別のところにあります。

1. **予測そのものが対象世界に影響を及ぼすこと**
2. **主観的な世界を扱うこと**
3. **戦略の影響を受けること**
4. **複雑系の振る舞いを扱うこと**

１番目の予測が対象世界に影響を及ぼすことについては、自然現象ではなく、社会現象である限り避けがたいことです。特に、**プロジェクト予測の場合には、予測が目標値として機能する**ことが多いため、このことを織り込む必要がでてきます。

２番目の主観的な世界を扱うことは、近年の行動経済学に代表されるように、人の

判断が合理的ではなく、**感情や心理的要因を扱う**ようになってきたことと関連しています[Tomono2006]。

３番目の戦略の影響については、競争相手や協調関係について扱うゲーム理論の適用領域になります[Okada2011]。予測の種類には、プレーヤのうちの誰が勝つか、例えば、次の選挙の勝利者、都知事が誰になるかといったことも含まれます。その場合には、影響を及ぼすプレーヤを特定すること、各プレーヤの行動規範を推定すること、結果に及ぼす要因の優先度を分析すること、各プレーヤ間の影響度を推定することなどを、**ゲーム理論を適用する前**に正確に把握する必要があります[Mesquita2009]。

４番目の複雑系の問題は、単純な定式化ができる世界と、本質的に複雑な世界とを峻別する方法を確立するところから検討していく必要があります。**複雑系については、あまり理論が整備されていない**ので、予測不能であることが分かった場合の実践的な対応策も決めておく必要があるかもしれません。当面は、複雑系の事例で、経済、情報ネットワーク、自然現象、生物の分野での成功例をソフトウェアの世界に適用できるかを検討していくレベルから始めることになると思われます。

よくよく考えてみると、**予測は、ソフトウェア、もっと言うと、実行可能知識です**。予測がどのように使われるか、予測するということがどのような影響を及ぼすかということを、あらかじめデザインしなくてはなりません。新ソフトウェア宣言で言うところの、数学、複雑系、知識論、進化、デザイン、経営・経済、言語ゲームなどの要素全てを含んでいます。

当面、予測技術が有効な領域は、ビジネス戦略です。「ビジネスアナリシス」の知識体系が整備され、用語や概念が定義されてきています[Ross2011]。ビジネスの目標、戦術、ポリシー、プロセス、ルールといったビジネスの構成要素の中で、的確な評価、判断を行っていくためには、予**測技術とそれに基づく意思決定手法**が必要になります。

ビジネス領域では、市場、ユーザ、競合他社、技術進歩といった不確実な要素、社会的な現象を扱わなくてはならず、多くの事項は未知です。この未知の事柄に対し、何らかの仮説、因果関係、定式化を可能な限り行い、意思決定に活かしていかなくてはなりません。

何らかの相関関係がある現象から、因果関係を導きだし、それに基づいた物語をビジョンや戦略として語るといったことを進めていくためには、実世界の現象の観測方法、データの採取法、分析法、統計解析、仮説検証といった実践的な（予測）プロセスを構築していく必要があります。

広義のアジャイルプロセスでは、ビジネスプロセスと開発（ソフトウェア）プロセスとの連動が大切ですが、ビジネスプロセスをどのように作り、運用していくかを、その予測手法とともに設定していかなくてはなりません。この点で、ビジネスアナリシス、とりわけ、ビジネスルール[Ross2013]とその指標化は重要な役割を果たしてい

くことでしょう。**一般のビジネスアナリシスの手法に対し、言語ゲームや本質的なアジャイルプロセスの考え方を取り入れたものにしていくとよい**と考えています。

６．ソフトウェア参謀

[Otsuki2014b]

ソフトウェアは、我々の社会にとってなくてはならないものですし、ソフトウェア産業に携わっている人々は、日本国内だけでも何十万人という人々がいます。ここのところ耳にするのは、製品欠陥の露見やリコール、開発プロジェクトの頓挫、不条理なマネジメント、サービスの停止などが散見され、関わっている人々も決して恵まれた境遇とは言えないようです。こういった事態に陥っているのは、ソフトウェアの本質が見えていないことや、本当の問題の在処を誤解しているところに要因があるように思えてなりません。

ソフトウェアと何か？

「ソフトウェアとは何か？」と問われて、明快に答えられる人は少ないでしょう。どんな言葉や概念でも同じですが、人や状況によってとらえ方は異なります。

開発者側の観点からすると、**「ソフトウェアとは、コンピュータ上で実行されるプログラムコードと、それに関わる文書である。」**といったところがもっともらしい定義です。操作マニュアル、仕様書、設計書といったものは無論のこと、開発途中の段取りや、品質保証のための検査書類などもソフトウェアの一部と考えてもよいでしょう。

利用者側からのソフトウェアとは、何か機器に組み込まれているとか、パソコンやタブレットの画面を通じて、「何か動くもの」といったとらえ方しかできません。つまり、「ソフトウェアが入っているよ」と言われれば、そのことを信じるしかありません。素朴な言い方をすれば、**「何かよくわからないけれど、賢そうな反応、動作、計算などをやってくれるもの」**と認識されるのが、利用者にとってのソフトウェアなのです。

抽象機械

ソフトウェアは、**人間が意思を持って制作する人工物**です。建築ではビルや建物、芸術では彫刻やオブジェ、工芸では茶碗や皿など、世の中は人工物で満ちています。では、ソフトウェアが一般の人工物と違うところは何でしょう？

一つの特性は、「動く」ことです。これは本質的な特性で、開発者側からすれば、プログラムの「実行」であり、利用者側からだと「動作」という言葉に現れています。

コンピュータの動作原理は、アラン・チューリングやフォン・ノイマンといった天才が基礎付けをした厳格な理論の上に成り立っています。コンピュータの構成要素である演算装置、メモリ、入出力装置といったものは、**抽象的に計算、記憶、通信といった概念に昇華**させることができます。

従って、**「ソフトウェアとは、抽象的なマシン（機械）である」**というのが、本質的な特性を的確に表していると思います。マシンは、動きます。自動車、ロボット、家電品などは、マシンです。さらに、「抽象的」というところがポイントです。物理的な制約を受けることなく、人間が自由に想像して作り上げることができるのです。四角い地球でも、西から登る太陽でも、どんな妄想も在りです。

プログラミングは、汎用コンピュータに対する操作指令の列を作る作業です。実は、汎用コンピュータは、数学的な計算の理論によって限界が示されているので、何でも作れるわけではありません。簡単に言うと、**有限の時間で、手順的に解けるものは作れますが、人間が本来持っている創造的な活動は、抽象機械として構築するのは難しい**のです。東大の入試問題は解けるけど、問題を作ることは人間にしかできません。
スキル

では、抽象機械としてのソフトウェアに関わる人に要求される能力やスキルは何でしょう？　今までのソフトウェア産業でベンダと言われる開発企業がやってきたことは、ユーザ企業が IT やソフトウェアの仕様を決定し、それの実装を請け負うということです。これは言うなれば、ベンダ側は、どのようなソフトウェアを作るべきかという問題設定が完了した段階以降、**汎用コンピュータとプログラミングの知識に基づいて、問題を解く役割**を担っていたことになります。

コンピュータが得意とすることは、帳票処理、会計管理、在庫管理などの、決まり切っていて手順化可能な業務です。世の中には同じような業務が溢れているので、これをソフトウェアによって置き換えていく活動は、それなりに意味があることでした。同じようなソフトウェア開発を繰り返していくうちに、プログラミングの部品化や再利用も進み、**コンポーネント、フレームワーク、パッケージといった組み合わせで、半自動的に開発ができる**ようになってきています。

従来のソフトウェア開発企業が限界にきているのは、産業構造上の問題です。ユーザ/ベンダという図式でソフトウェア開発をとらえる限り明日はありません。上流が大切だとか、要件定義のスキルを向上させるとか、はたまた、大規模なプロジェクト管理能力を養成したところで、旧来のビジネスの延長線上での話に過ぎません。開発技術は進化していきます。開発プロセスの中で、手順化・機械化できるものは、どんどん自動化していくことになります。つまり、開発のための単純なツール操作によって置き換わっていくことになるでしょう。**生産性向上は開発業務の効率化に過ぎず、手順的な開発作業は、より安価な人件費で済む海外新興国へ移転してしまう**ことにな

ります。

原理的に言うと、**一度解かれた問題は、二度解く必要はありません**。同じソフトウェアを二度開発する必要もありません。我々がやるべきことは、**新たな領域を開拓し、新たな問題設定をすることです**。

プログラミング言語の歴史をたどってみれば分かりますが、コンピュータへの指令の仕方は、機械語あるいはアセンブラ言語と呼ばれる低レベルのものから始まり、より人間が扱いやすいCやJavaといった高級プログラミング言語へ発展してきています。これは抽象機械の抽象度も上がってきていることを意味しています。抽象機械を集めて、その上に、より高度の抽象機械を構築していくことができます。そう、ソフトウェアが解決すべき問題は、どんどん抽象度が上がり、より高度になっていく宿命にあるのです。そこで必要なものは、新しい世界を描き、新たな問題設定をする能力です。

新しい専門家像

ソフトウェアに関する主要な仕事は、想像力を駆使して、抽象機械による新たな世界を創造することです。数学的な定式化能力、論理力、抽象化能力のみならず、世界を創り出すデザイン力も必要になります。これ等を一言で表すなら**「概念操作の専門家」**ということになるでしょう。おそらく抽象機械の構成要素である、計算、記憶、通信といった諸概念が、実世界でどのような価値を生むのかを描いていくことになります。

これからは知識主導の社会です。勝ち残っていく企業は、競争力の源泉となる知識を、手順化できるものはソフトウェアに組み込んで自動化し、人間がやらなくてはならない創造的な活動に専念していくことになるでしょう。もっと言うと、**ソフトウェアを含めたビジネス領域全体、組織や仕組みをデザインする**ことが要請されています。従って経営に直接結びつく活動になりますから、私はこういった職種を**「ソフトウェア参謀」**と呼ぼうと考えています。

おわりに

ものを書くというのは、考えをまとめるために、とても効果的です。日頃のSNSのつぶやきや、会話の中の断片的な表現では、「知」の蓄積は十分ではありません。完結した書き物に仕上げることによって、その作品を前提とし、次の段階に進むことができます。本節に掲げた作品は、元日に書き下ろし、半年以上の間、推敲あるいは熟成させてきたものです。

気づかれた方もいらっしゃるかと思いますが、本節には図が一つもありません。あ

えて、日頃良く描いている箱や矢印で構成された見栄えのする図を排したのです。以前、大手企業研究所に勤めていた時に、内容に自信が無い時には、形式を重んじ、見栄えのする図を描けと指導されたことを思い出します。

１．哲学のすすめ
２．仏教からのヒント
３．知の基盤としての数学
４．新ソフトウェア宣言その後
５．これからの予測技術
６．ソフトウェア参謀

本節は、随想集としては６編からなる短いものです。中程の『新ソフトウェア宣言その後』が中心になっています。この宣言は骨格がしっかりとしていて、完備で美しい宣言だと思っています。**何かしら新しいことを考えついても、７つの宣言文のどこかに落とし込まれます**。思想的な基盤が安定しているので、今後は、この方向に沿った理論と実践を繰り返していくことによって、新たな「知平」が開けていくと確信しています。

参考文献

[Atiyah2010] Michael F. Atiyah, What Is Mathematics?: Selected Essays of Michael Atiyah, 2010.11.25
（マイケル・Ｆ・アティヤ, 『数学とは何か？』, 朝倉書店, 2010.11.25）

[Brooks2010] Frederic P. Brooks, Jr., The Design of Design: Essays from a Computer Scientist, Addison-Wesley, 2010.4.1
（フレデリック・Ｐ・ブルックス Jr, 『デザインのためのデザイン』, ピアソン, 2010.12.25）

[Coxeter1969] Harold Scott MacDonald Coxeter, Introduction to Geometry, 1969.1.15
（H.S.M. コクセター,『幾何学入門』(上/下), 筑摩書房(学芸文庫), 2009.9.9）

[Fujii2012] 藤井聡, 『プラグマティズムの作法 : 閉塞感を打ち破る思考の習慣』, 技術評論社, 2012.4.18

[Go2011] 呉智英, 『つぎはぎ仏教入門』, 筑摩書房, 2011.7.25

[Hama2014] 濱勝巳, 『Agile 開発コンセプトとその動向』, 科学技術振興機構 第 8 回システム科学検討会, 2014.6.23

[Hashizume2009] 橋爪大三郎, 『はじめての言語ゲーム』, 講談社現代新書, 2009.7.20

[Hashizume2011] 橋爪大三郎, 大澤真幸, 『不思議なキリスト教』, 講談社現代新書, 2011.5.20

[Hashizume2013] 橋爪大三郎, 大澤真幸, 『ゆかいな仏教』, サンガ新書, 2013.11.1

[Hirano2010] 平野敦士カール, 『プラットフォーム戦略』, 東洋経済新聞社, 2010.8.12

[Johnson2007] Neil Johnson, Simply Complexity: A Clear Guide to Complexity Theory, Oneworld Publications, 2007

（ニール・ジョンソン, 『複雑で単純な世界：不確実なできごとを複雑系で予測する』, インターシフト, 2011.12.15）

[Kikuchi2008] 菊地奈穂美, 飯泉純子, 亀田康雄, 細川宣啓, 渡辺千恵子, 大槻繁, 『見積り法 COCOMOⅡ概説』, SEC Journal Volume12, 2008.1.15

[Krippendorff2006] Klaus Krippendorff, The Semantic Turn: A New Foundation for Design, Taylor & Francis Group,LLC, 2006

（クラウス・クリッペンドルフ, 『意味論的転回：デザインの新しい基礎理論』, 星雲社, 2009.4.1）

[Lakoff2001] George Lakoff and Rafael E. Nunez, Where Mathematics Comes From, Basic Books, 2001.8.7

（レイコフ and ヌーニェス, 『数学の認知科学』, 丸善出版, 2012.12.10）

[Matsubara2007] 松原望, 『入門統計解析：医学・自然科学編』, 東京図書, 2007.10.25

[McGrayne2011] Sharon Bertsch McGrayne, The Theory That Would Not Die: How Bayes' Rule Cracked the Enigma Code , Hunted Down Russian Submarines, and Emerged Triumphant from Two Centuries of Controversy, Yale University Press, 2011

（シャロン・バーチュ・マグレイン, 『異端の統計学ベイズ』, 草思社, 2013.10.29）

[Mesquita2009] Bruce Bueno de Mesquita, The Predictioneer's Game, Random House Publishing Group, 2009

（ブルース・ブエノ・デ・メスキータ, 『ゲーム理論で不幸な未来が変わる』, 徳間書店, 2010.5.31）

[Michell2009] Melanie Mitchell, Complexity: A Guided Tour, Oxford Press, 2009

（メラニー・ミッチェル, 『ガイドツアー：複雑系の世界』, 紀伊國屋書店, 2011.12.7）

[Nakazawa2006] 中沢新一, 『三位一体モデル TRINITY』, 東京糸井重里事務所, 2006.11.1

[Negoro2013] 根来龍之監修, 『プラットフォームビジネス最前線』, 翔泳社, 2013.12.12

[Okada2011] 岡田章, 『ゲーム理論』〔新版〕, 有斐閣, 2011.11.25

[Otsuki2010] 大槻繁, 『新ソフトウェア宣言概説』,知働化研究会, 2010.7.15

[Otsuki2012a] 大槻繁, 『実行可能知識のデザインプロセス：創造的ソフトウェア開発プロセスΛVモデル』, 日本デザイン学会 デザイン学研究特集号 デザイン思考 Volume19-4, No.76, 2012.12

[Otsuki2012b] 大槻繁, 野口隆史, 『知のフリマのすすめ方』, 知働化研究会, 2012.12.21

[Otsuki2014a] 大槻繁, 『知平』, 一（いち）セカンダリサイト http://www.ichicorp.biz/secondary/20140101.html, , 2014.1.1

[Otsuki2014b] 大槻繁, 『ソフトウェア参謀という仕事』, 一（いち）セカンダリサイト http://www.ichicorp.biz/secondary/20140213.html, , 2014.2.13

[Psec2013a] P-sec, 『ソフトウェア社会の手法基盤と人材基盤』,実践的ソフトウェア教育コンソーシアム WG1 活動報告書(2011年11月～2013年3月), 2013.3.31

[Psec2013b] P-sec WG2（大槻まとめ）, 『仙石社長：知のプラットフォーム講演会解説』, WG2（ソフトウェアの方法論に関する実践知と社会基盤）オプショナルツアー（三技協様訪問）, 2013.10.27

[Putnam1995] Hilary Putnam, Pragmatism, 1995

（ヒラリー・パトナム, 『プラグマティズム：限りなき探求』, 晃洋書房, 2013.2.20）

[Randall2012] Lisa Randall, Knocking on Heaven’s Door: How Physics and Scientific Thinking Illuminate the Universe and the Modern World, Ecco; Reprint 版, 2012.10.2

（リサ・ランドール, 『宇宙の扉をノックする』, NHK 出版, 2013.11.30）

[Ross2011] Ronald G. Ross and Gladys S.W.Lam, Building Business Solutions Analysis with Business Rules, Business Rules Solutions, LLC, 2011

（ロナルド・G・ロス, グラディス・S・W・ラム, 『IT エンジニアのためのビジネスアナリシス』, 日経 BP 社, 2012.11.26）

[Ross2013] Ronald G. Ross, Business Rule Concepts: Getting to the Point of Knowledge (Fourth Edition), Business Rules Solutions, LLC, 2013

（ロナルド・G・ロス, 『アジャイル経営のためのビジネスルールマネジメント入門』）, 日経 BP 社, 2013.7.29

[Sasaki2006] 佐々木閑, 『犀の角たち』, 大蔵出版, 2006.7.20

[Ses2010] 大槻繁,濱勝巳ほか, 『新ソフトウェア宣言』,ソフトウェア技術者協会, ソフトウェア・シンポジウム 2010, ソフトウェアエンジニアリングの呪縛 WG, Post Proceedings WG1, 2010.6

[Tachibana2012] 橘玲, 『かっこにっぽんじん（日本人)』, 幻冬舎, 2012.5.10

[Takeda1993] 竹田青嗣, 『はじめての現象学』, 海鳥社, 1993.4.27

[Tomono2006] 友野典男, 『行動経済学：経済は「感情」で動いている』, 光文社新書, 2006.5.20

[Wittgenstein1921] Ludwig Wittgenstein, Tractatus Logico-Philosophicus, Routledge & Kegan Paul, 1961(First Edition 1921)

[Wittgenstein2003] Ludwig Wittgenstein, Philosophische Untersucheungen, Suhrkamp Verlag Frankfurt am Main, 2003

（丘沢静也訳, 『哲学探究』, 岩波書店, 2013.8.29）

[Yamada2004] 山田正樹, 『実行可能知識と様相研究所作品集』

（・実行可能な知識とソフトウェア(1):システムに適した知識の表現方法を探る,@IT 記事, 2004.3.17

・実行可能な知識とソフトウェア(2):知識とソフトウェアのギャップ、それをどう埋めるのか?, @IT 記事, 2004.4.6

・知識創造とソフトウェア開発(Software People,Vol.2,2003.4.15)

・Software as Executable Knowledge(2006.7.20)），

http://www.metabolics.co.jp, 2004-2006

[Yamada2009] 山田正樹, 大槻繁, 『人働説から知働説へ』, 知働化研究会, 2009.7.15

[Yasutomi2010] 安冨歩, 『経済学の船出：創発の海へ』, NTT 出版, 2010.12.2

[Yasutomi2011] 安冨歩, 『生きる技法』, 青灯社, 2011.12.25

■