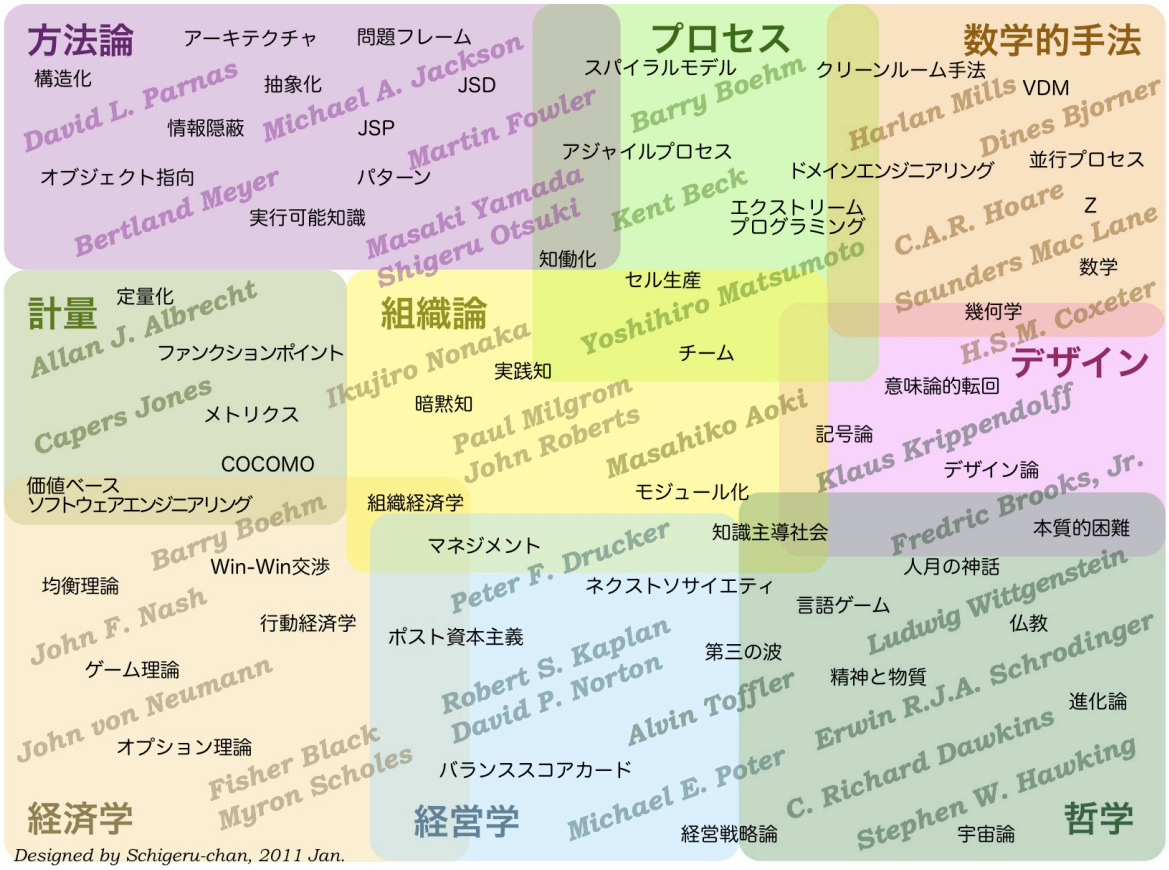
小論

2011.6.20

研究マップ

研究アプローチの全体像



株式会社一（いち）

はじめに

ソフトウェアエンジニアリングというのは総合的な知識体系です。背景となる研究アプローチや基礎付ける学問もさまざまです。常に複数の学問領域の間の境界で新しいことが起こっています。我々は、コンサルティングソリューションを検討するに当たり、見積り評価やプロジェクト診断などの経験を通じて、図に示したような広範な知識が必要であると考えるようになりました。こういった捉え方は、人それぞれで、恣意的なものになると思いますが、自らの研究や技術開発のアプローチを示す一つの有効な手段と考えています。図のデザイン上の美しさを狙って、バランスを保ったキーワード、人名を掲げています。時々、この図を見直していますが、この図を描き始めて５年目になりますが、大局的には変更はありません。

全体の構成は、今、検討を進めている「一の新戦略」の世界観です。昔から読み込んで影響を受けた書籍、お会いして尊敬している方々、そして、きちんと勉強しなくてはならないと考えているものなどを掲載しています。本質的でインパクトがあり、思想として深いものが多いと思います。図の領域がオーバラップしている表示も、微妙に調整しています。レイアウト上の唯一の心残りは、二次元の配置ではしかたがないのですが、Barry Boehm が２カ所になってしまったことです。

全部で９つの学問領域（Academic Domain）から成り立っています。

【方法論】

古来、構造化プログラミングや抽象機械の提唱で著名な Dijkstra（ダイクストラ）、複合設計法の Myers（マイヤーズ）あたりも掲げたかったのですが、Parnas（パルナス）の情報隠蔽に集約しました。1970 年代初頭に階層化や構造化の考え方も整備され、以降、良構造であることやモジュール化、カプセル化といったソフトウ

ェアエンジニアリングの本質的な考え方が整備されていきます。

オブジェクト指向の考えもこの延長線で捉えることができるでしょう。オブジェクト指向については、SmallTalk をはじめ、C++や Java といったプログラミング言語がたくさん提唱されてきていますが、言語の美しさの観点から Eiffel（エッフェル、あるいは、アイフル）で有名な Meyer（メイヤー）をあげています。Meyer は、Design by Contract（契約による設計)、さらに、Hoare（ホア）ロジック（事前・事後条件論理）の導入を図り、ソフトウェアエンジニアリングへのインパクトも大きいと考えています。

ソフトウェアの世界で「抽象化」というと、データ抽象、関数抽象、そして、状態（振舞い）抽象があります。今回の研究マップではレイアウトの都合上落としてしまいましたが、この状態について、すばらしい仕事をしたのが Harel（ハレル）です。CACM という有名な雑誌に 1988 年に “On Visual Formalisms” という論文を発表しました。これが今でいう UML(Unified Modeling Language)の状態記述の元祖です。Brooks, Jr（ブルックスジュニア）のいう本質的困難の中で、状態に関する複雑性(complexity)に対処する原理を提唱しています。

Fowler（ファウラー）は、リファクタリングやデザインパターンで有名です。実践的な観点で本質や理論をおさえてうまくまとめる力が傑出しています。アジャイルプロセスの実践者としても実績があり、アーキテクチャ、実装面の造形も深いくとても参考になります。

この領域の最後にして、最も注目に値する人物は、何といっても Jackson（ジャクソン）です。JSP, JSD, 問題フレームなどのすばらしい提唱をしています。概ね、Jackson の思想は世の中一般の人々が理解するのに１０年くらいはかかるので、Jackson の書籍をしっかり読み込んでおくというのは、それだけで価値があると思います。蛇足ですが、Michael Jackson のご子息は Daniel Jackson（MIT 教授）ですが、 “Software Abstraction” という書籍を出版しています。とてもセンスのよい形式手法（Alloy という言語）だと言われています。

方法論とプロセス、そして、組織論との境界領域といってもよい位置になりますが、「実行可能知識」と「知働化」を位置づけてみました。提唱コンセプトリーダの山田正樹氏に並び、僭越ながら大槻の名前も配しています。

【プロセス】

プロセスのところで掲げている事項は、いわゆる「パラダイム」の話です。従来のウォータフォール型の開発プロセスの原理的な問題が指摘されたのが 1980 年代初頭のライフサイクル論争です。当時、IEEE SIGSE（ソフトウェアエンジニアリンググループ）の雑誌には毎号、いろいろな人が論陣をはっていました。そういう中で 1988 年に Boehm（ベーム）は、スパイラルモデルという計画からリスクまでを織り込んだ実践的なパラダイムを提唱しています。

最近のアジャイルプロセスの盛り上がりのきっかけは何といっても Beck（ベック）の XP（エクストリームプログラミング）でしょう。人の数ほどアジャイルプロセスの種類はありますが、カリスマ性といい Beck の右に出る人はいません。2001 年に、テクノロジックアート社の長瀬さんとともに、Beck を日本に招聘させてもらいましたし、現在のアジャイルプロセス協議会を立ち上げるモチベーションの多くは、この手法が時流をえた新しい取組みだと信じることができたからです。

今回の研究マップでは削除しましたが、プロダクトラインという考え方も、もともとは、日本のソフトウェアファクトリーとか、ドメインエンジニアリングとかで呼ばれていたものに呼応した動きです。ICSE2001 がトロントで開催された折に、Northlop（ノースロップ）の華麗なる講演を聴いたのが印象的です。日本ではキーワードとしてはもう消費されてしまい、製造業系の水面下では検討はされているようです。これに関連して、松本吉弘教授が提唱された「（ソフトウェア）セル生産」は、注目すべきものと考えています。これについては、アジャイルプロセス協議会のワーキンググループ活動を行いました。

【数学的手法】

ソフトウェアエンジニアリングの領域で重要だと思っているのが、形式手法を活用した厳密な(rigorous)アプローチです。以前の研究マップではこの学問領域を「形式手法」と呼んでいましたが、『一の新戦略』に書いたように「数学」を中心に据え、「数学的手法」とすることにしました。VDM（ウィーン開発法）を提唱した Bjorner（ビヨナー）は、優れたアーキテクトの道具として数学的な考えを活用するという意味でのドメインエンジニアリングという提案をしています。

クリーンルーム手法は、1990 年代初頭に IBM のプロモーションで普及した手法です。この手法を現場に展開するための教育やコンサルティングの仕事をした経緯もあり、かつて『ソフトウェアクリーンルーム手法』という著作を出版しました。クリーンルーム手法の基本的なアイディアは Mills（ミルズ）が 1979 年に提唱しています。構造化プログラミング＋関数等価性の検証、ソフトウェアが良構造であるということ、証明可能であることなどがポイントになります。おそらく、状態やプロセス世界でもの証明可能な良構造があると予想され、もしそれが発見できるならばブレークスルーをもたらすと考えています。

概して、形式手法や論理の研究は欧州が活発です。その一つの中心が Hoare（ホア）です。方法論のところでも記載したホアロジックや、ソフトウェアエンジニアリングを支えるサイエンス（欧州では Informatics：インフォマティクスといいます）で多彩な仕事をしています。CSP(Communicating Sequential Process)という並行プロセスのモデルも有名です。

また、数学そのものの考え方も直接ソフトウェアの世界にインパクトをもたらすことがきると考えています。Mac Lane（マクレーン）は、数学全体の概念の関係

やその機能と形式とを系統的にまとめています。特に、「幾何学」は我々が今後数学的手法の中で中心に据えていきたいと考えているものです。

【哲学】

ソフトウェアに関する「哲学」は、あまり正面から論じられることは少ないのですが、とても重要な領域と考えています。「方法論」の方に分類した「知働化」や「実行可能知識」は、その根底には哲学的な思想があります。

この分野での偉人と言えば、Brooks, jr.（ブルックスジュニア）が挙げられます。“Mythical Man-Month（人月の神話）”で有名です。ソフトウェアエンジニアリングの分野全体の問題設定という意味では、 これについては、他の書き物でも多く書いているので省略しますが、Brooks の言う本質的困難が最も重要で、まさに本質的だと思っています。

もともとコンパイラ（言語処理系）や言語については興味があります。言語学という分野では Chomsky（チョムスキー）が有名です。私が興味を覚えているのは言語そのものがどのように（社会的コードとして）規定されていくのかとか、そもそもコミュニケーションの手段としての言語とは何かといった問題意識があります。Searle（サール）は、言語行為論を提唱しましたし、もっとさかのぼると Wittgenstein（ヴィトゲンシュタイン）の言語ゲームにまでいかざるを得ません。

ヴィトゲンシュタインは、20 世紀初頭に活躍した哲学者です。『論理哲学論考』（命題論理基礎付け）と、『哲学的探求』（言語ゲーム論）という大きく２つの仕事

をしています。いわゆる奇才です。実はジェットエンジンとか住宅の設計なども手がけていますが、すごく合理的で美しいものです。哲学書というのは日本語で勉強するのは大変なのですが、黒崎宏氏というヴィトゲンシュタイン研究に生涯を捧げたような優れた方の翻訳があるので助かっています。

【経営学】

経営（マネジメント）という概念そのものを創出したのが Drucker（ドラッカー）です。多くの著作があるのですべては見ていませんが、「ネクストソサイエティ」は晩年の著作としてとても優れていると思います。知識主導社会の予見も適確でしたし、一（いち）の経営も 、人的ネットワーク、知識の重要性、コミュニティ型の社会への移行等、 Drucker のコンセプトに従って経営しています。

Kaplan（カプラン）と Norton（ノートン）の BSC（バランススコアカード）は経営戦略立案の手法としてとても優れています。財務以外の観点についても関与者の合意形成に役に立つもので、普遍的な方法論提示をしているといえます。Porter（ポーター）は「競争の戦略」という古典的名著で有名です。ファイブフォース分析やバリューチェーンといった経営学の基礎づけを行った人です。

【経済学】

小室直樹の「経済学をめぐる巨匠たち」という書籍をよんで、誰をおさえればよいかを探りました。また、一橋大の岡田先生にゲーム理論の道案内もしていただきました。技術評論社の特集記事『ソフトウェア経済学のすすめ』をまとめるためにオプション理論の概要も調査しました。ソフトウェアエンジニアリングの分野の巨人 Boehm（ベーム）は、今世紀に入り、『価値ベースソフトウェアエンジニアリング』を記し、新しい展開を示唆しています。

概ね、入門的解説書からとっついてみているのですが、かえって混乱することが多いので、いろいろ試行錯誤しています。経済学は人間のどろどろとした社会･経済的活動を対象としつつ、道具立ては数学なので、数学がきちっと書いてあるものは比較的理解しやすい傾向があるようです。

【組織論】

組織論といっても明確に確立された分野があるというわけでもないようです。私が注目しているのは、「モジュール化」です。情報隠蔽、カプセル化の産業や組織版といってよいでしょう。Boldwin&Clark（ボルドウィン、クラーク）の「デザイン・ルール」という書籍の元になっている概念を青木昌彦氏が提唱しています。私の感覚としては、ソフトウェアエンジニアリングで培われてきた知見を社会や経済の分野に援用する位置づけになります。この手の例では、コンカレントエンジニアリングというのが、並行プロセスを原初的概念としているところにも見受けられます。

また、「組織経済学」は、時々、交渉とか配分とかの定式化の際に参照するバイブルのようなもので、百科事典のようによくまとまった書籍です。最近は、Milgrom（ミルグロム）や Roberts（ロバーツ）の書籍翻訳で時々名前を目にする谷口和弘氏の書籍も興味深くウォッチしています。

組織論の中で重要だと思うのがチームの話です。このあたりは、理論としては真空地帯のようです。松本吉弘教授のソフトウェアセル生産方式の話は、一つのチーム論として捉える方がよいかもしれません。

【デザイン】

「デザイン」は、研究マップ作成の経緯から言うと今回初登場です。『一の新戦略』で述べているように、ソフトウェアエンジニアリングを創造的な人工物作成活動に応用しようという意図があるために設定しました。Krippendolff（くりっぺンドルフ）の「意味論的転回」は、デザイン論として我々に響くものがあります。基底に哲学として Wittgenstein の「言語ゲーム」があるからです。また、Brooks,Jr. の最近の著作 “The Design of Design”では、組織的に人工物を作っていくことについて扱っています。

ソフトウェアに関する研究の最近の傾向は、そのソフトウェアが置かれるコンテクストや、人間の認知、解釈の世界を探求するところにあります。その意味では、古くからある「記号論」の世界も押さえておくべきと考えています。

【計量】

最後に採り上げる分野は、「計量」です。我々はソフトウェアやそのプロジェクトに関わる「予測」を主務としています。そのための前提となる事項が「計量」です。ファンクションポイント、COCOMO と、実務で常用している領域です。Boehm の価値ベースソフトウェアエンジニアリングも結局は価値を測るというところに行着きます。

■