Vol.2020-SE-206 No.11
2020/11/27

データ駆動ソフトウェア工学への考察

青山 幹雄 [1]

概要：ビッグデータの収集，分析や機械学習などの技術の発展と共に，科学･工学においてデータを活用したデータ駆動科学･工学の萌芽が見られる．ソフトウェア工学においても，従来の人によるモデルに対して，デジタルツインなどのデータを基礎として実世界を理解し，ソフトウェアを開発する学問体系の可能性が生まれている．本稿では，これまでのソフトウェア工学をモデル駆動ソフトウェア工学としてモデル化し，それに対してデータ駆動の考え方に立つデータ駆動ソフトウェア工学のモデルを議論する．

キーワード：ソフトウェア工学，データ駆動ソフトウェア工学，データ駆動科学・工学，機械学習，Software 2.0

Toward Data-Driven Software Engineering

MIKIO AOYAMA[1]

Abstract: This article discusses a model of DDSE (Data-Driven Software Engineering, which is intended to develop data-driven software, a class of software utilizing big data. The concept of data-driven software generalizes machine learning software and software including machine learning components. This article reveals some aspects of data-driven software and key research issues in DDSE.

Keywords: Software Engineering, Data-Driven Software Engineering, Data-Driven Science and Engineering, Machine Learning, Software 2.0

1. 研究の背景と課題

1.1 DX がもたらす産業と社会の変容

DX(Digital Transformation)はデジタル技術による事業変革である．DX は産業界のみならず社会全体の重要な課題となっている[4][5].

2018 年に著者が座長を務めた「デジタルトランスフォーメーションに向けた研究会が発行した「DX レポート」はその副題「2025 年の崖」とあいまって産業界で DX が認識される機会となった．2020 年の新型コロナ禍は社会全体がデジタル化の遅れを認識する契機となった．

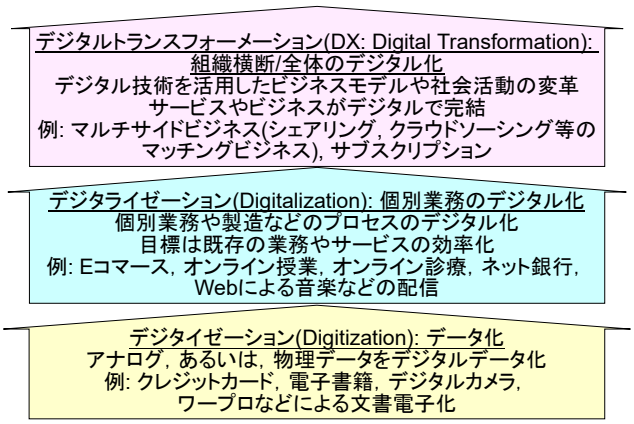

図 1 DX の 3 段階

Fig. 1 Three Stages to Digital Transformation

DX 推進のプロセスは図 1 に示す 3 段階が知られている．この図が示唆するように DX のためには，まず，デジタイゼーション(Digitization)が基礎となる．

社会全体をデジタイゼーションする概念の提案の一つは 1991 年に Gelernter による Mirror Worlds にある[16]．同書の副題“The Day Software Puts the Universe in a Shoebox”はソフトウェアにより世界全体が靴箱程度のスペースに変換されることを示唆している．

この概念は，その後，Google による文書のデジタイゼーションから始まり，Google Maps, Google Earth など物理世界のデジタイゼーションとして実現されてきた．

2007 年の iPhone の導入や近年の自動車の高度運転支援(ADAS)，自動運転などの開発と導入に伴い，これらに搭載されているカメラ，LiDAR, GPS などを用いて個人や自動車などの行動のデジタイゼーションが進んできた．社会活動のデジタイゼーションと言える．この中で，自動運転などの新たな応用を目的として 3 次元高精細度ライブ地図のように物理世界のデータ化精度向上に加え，交通や気象などの社会活動を表す動的データの重ね合わせも進んできた[6][21].

1 南山大学
Nanzan University

一方，製造業においては 1993 年に Grieves が提案したデジタルツイン(DT: Digital Twin)の概念[19]が NASA での研究などを経て発展している[47]．デジタルツインとは個々の機器の動作などをデジタイゼーションすることを意味する．

さらに，SNS の普及により人と人との心理的関係や人の嗜好などの心理に関わるデジタイゼーションが進んでいる．

企業経営においてもデータを活用するデータ駆動経営の概念とその実践が広がりつつある．このようなデジタイゼーションから DX への進展が意味することは社会活動がデータによる進められるデータ駆動社会(Data-Drive Society)へ移行しつつあることである[40].

1.2 DX とソフトウェア工学の課題

DX により事業や社会の変革の主眼はソフトウェアのユーザや社会を起点とした社会課題の解決による価値創出にある．そのため，社会課題の発見に焦点を当てている．

従来のソフトウェア工学，あるいは，要求工学においては課題が概ね既知であり，その解決として要求を定義し，ソフトウェアとして実現してきた．それに対して，DX のアプローチは個人や機器などから得られるデータからこれまで発見できなかった課題の発見が必要となる．ここで対象とする個人行動や機器の動作から得られるビッグデータは人が扱える規模，速度，多様性を超えている．Ratzesberger らは e コマースのユーザによるクリックデータのストリームに対して，経験的にタグ数が 100 を超えると人手では把握できないと指摘している[42].

このことと近年の機械学習技術の急速な発展，普及とがあいまって機械学習を組込んだソフトウェアの開発が活発となっている．このようなソフトウェアは機械学習ソフトウェア(MLS: Machine Learning Software)とも呼ばれる．機械学習の統計的な振舞いは訓練データ，入力データに依存し，従来のソフトウェアのアルゴリズムに基づく決定的論的な振舞とは本質的に異なる性質を持つ．そのため，MLS の開発はソフトウェア工学の新たな課題といえる[3][25][36][44].

本稿では，このような機械学習を組込んだソフトウェアを含むソフトウェアをその性質からデータ駆動ソフトウェア(Data-Driven Software)と呼ぶことにする．ここで，データ駆動を用いるのは次の 2 つの理由がある．

(1) MLS を含むデータにより振舞いが決定されるソフトウェアのクラスとしての扱いを可能とする．
(2) 他の科学・工学，あるいは経営などの分野との比較などを行えるようにする．

1.3 研究課題

上記の背景から，本稿の研究課題は以下の 3 点である．

(1) データ駆動ソフトウェアとは何か?
(2) データ駆動ソフトウェア工学とは何か?
(3) データ駆動ソフトウェア工学の課題は何か?

2. アプローチ

本稿では，研究課題に対してソフトウェア工学の課題の観点と科学・技術研究におけるデータ駆動の観点から，図 1 に示すような論点整理を通して研究課題を議論するアプローチをとる．

まず，ソフトウェアとソフトウェア工学を取り巻く主要な変化を取り上げ，その論点を明らかにすることによりデータ駆動ソフトウェア工学の概念の創出を試みる．なお，以降，データとはビッグデータの意味で用いるが，強調の意味でビッグデータも用いている．

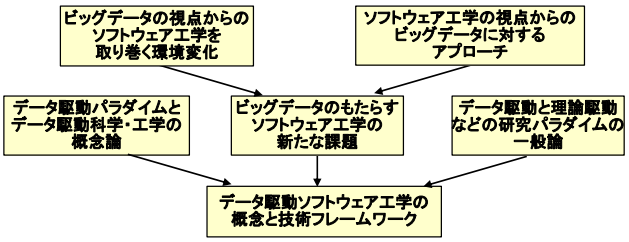

図 2 アプローチ
Fig. 2 Approach

3. データ視点からのソフトウェア工学を取り巻く環境変化のもたらす課題

スマートフォンや IoT，さらには，センサ技術の発展に伴い，非構造データ，あるいは，半構造データの大規模な集積が進んだ．このようなデータから価値を抽出したり，洞察を得るための分析技術が必要となっている．この点からソフトウェア工学への課題を明らかにする．

3.1 Software 2.0

Karpathy は従来のソフトウェアを Software 1.0 とし，それに対して機械学習によるソフトウェアを Software 2.0 として概念づけ，その開発のあり方を示した[24]. これにより，機械学習を用いたソフトウェアの概念とそれを開発するためのソフトウェア工学の課題が提起された．これらの議論から，表 1 に Software 1.0 と Software 2.0 の対比を示す．

表 1 Software 1.0 と Software 2.0 の比較
Table 1 Comparison of Software 1.0 and Software 2.0

	Software 1.0	Software 2.0
解決対象	人によりモデル化された特定のシステム	データを生成する任意のシステム
開発	人手によるプログラミング	訓練データのアノテーションにより NN を訓練し，最適なパラメータ集合として学習モデル(Learned Model)を得る
成果物	ソフトウェア	学習モデル(パラメータ集合)
構造特性	不均一(制御構造など)	均一(多層 NN，マトリクスの重みと積計算)
スケーラビリティ	設計依存	設計によらずスケーラブル(NN のスケール)
説明可能性	分析可能	不可能，あるいは，困難
障害発生	システミック障害(アルゴリズムに基づく)	予測困難(データに依存)

Software 2.0 の概念は機械学習によるソフトウェアと従来の Software 1.0 との違いを明らかにし，際立たせた．それに対し，Meijer はソフトウェア工学の研究の視点から両者の共通性に着目する必要性を指摘している[37]．この議論では，ニューラルネットワークは浮動小数点データの多次元配列を入出力とする一つの関数であると指摘している．さらに，様々なニューラルネットワークを提供するフレームワークそのものが Software 1.0 としてプログラミングされていることも指摘している．

Software 2.0 の概念は概括的でもあり機械学習のみから構成されるソフトウェアと解釈されるが，その構成は明らかではない．また，Software 2.0 に対するソフトウェア工学のあり方への直接的な展開は明確ではない．しかし，今後のソフトウェア工学のあり方を議論する起点として認識しておく必要がある．

3.2 機械学習ソフトウェアがもたらす課題

近年の機械学習技術の発展とあいまって，機械学習を組込んだソフトウェアシステム(以降，MLS と略記)の開発が急速に広まっている．

MLS の出力はその訓練データや入力データに依存し，同一入力データに対して出力データは同一となるとは限らない．また，入力データの変化に対して学習モデルが対応できなくなるコンエプトドリフトなど従来のアルゴリズムに基づくソフトウェアとは本質的に異なる性質を持つ．このような MLS の開発技術はソフトウェア工学のあらゆる技術に関わり，大きな課題となっている[25]．

本稿では MLS をデータ駆動ソフトウェアの主要なサブクラスとして位置づけている．

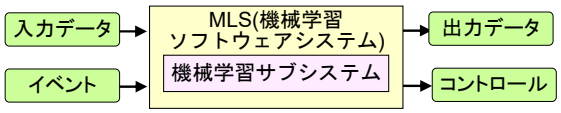

図 3 機械学習ソフトウェアの構成モデル

Fig. 3 An Architectural Model of Machine Learning Software

4. ソフトウェア工学の視点からのデータに内在する課題

ビッグデータとして集積されるデータは非構造，半構造データであり．個々のデータはいわばスモールデータである．かつ，それ自体ではその意味づけは困難である．これは局所データ(Local Data)であるとも言える[32]．従って，局所データの集積からのみでは価値ある情報を抽出できるとは限らない．

実際には，実世界の個人や工場の個々の機器などの特定の対象からデータを収集する．しかし，ここで得られる情報は個人や個々の機器など，インスタンスのある側面に関するデータとなる．このようなデータがデジタルツインとなる．従って，個人や個々の機器などの局所データで，かつ，抽象化されていないデータとなる．デジタルツインは個人や機器の行動特性を発見し，予測するための有力なアプローチとして期待されている．しかし，その対象の数の増大に対して効率的に価値を抽出する技術が求められる．機械学習はその有力なアプローチといえる．このため，例えば，データの意味づけ(アノテーション)を行うことが一般に行われているが，そのコストと期待される価値の関係は明らかとは言えない．

5. データ駆動・科学のアプローチ

5.1 データ駆動パラダイムとその対比的概念

データ駆動に基づく研究パラダイムをデータ駆動パラダイムと呼ぶ．この呼称の起源は 2007 年に Jim Gray が提案した第 4 のパタダイム(The Fourth Paradigm)にある[22]．彼は研究パラダイムとして，エンピリカル(Empirical)，理論的(Theoretical)，計算的(Computational)に続きデータ中心(Data-Intensive)を提案した．本稿ではデータ中心をその意味からデータ駆動と呼ぶこととする．

ここでデータ駆動とは理論と実験，シミュレーションを統合し，機器やシミュレーションから得られたビッグデータからデータ駆動発見(Data-Driven Discovery)と呼ばれる新たな発見のアプローチに基づく研究パラダイムとしている．

近年，ビッグデータの集積が進んだことにより科学・工学の研究アプローチとしてデータ駆動科学・工学が提唱されている．Brunton と Kurz はデータから低次元のパターンの発見とそれからシステムの簡略化のための変換(Transform)による制御システムのデータ駆動最適化(Data-Driven Applied Optimization)について議論している[10]．Kutz は，このようなアプローチの理由として厳密な分析よりコンピュータによるデータ分析に意味があるとの考えを示している[28]．

データ駆動科学・工学の概念は科学・工学のすべての分野に影響を及ぼしつつある[30][35]．

6. ソフトウェア工学におけるデータ駆動アプローチの萌芽

ソフトウェア工学におけるデータ駆動の概念は要求工学分野で萌芽している．しかし，ソフトウェア工学コミュニティ全体としては，依然として，議論は極めて少ないといえる．

6.1 データ駆動要求工学

要求工学におけるデータ駆動の概念は Maalej らの提案が嚆矢といえる[33]．この論文では Web 上でのユーザの利用ログや機器のセンサからのデータを要求獲得に活用するアプローチを示した．著者らもステークホルダの発話データからステークホルダ分析を行う方法を提案してきた[15]．

Maalej らはデータ駆動要求工学のアップデート版として，特にアプリケーションユーザからのフィードバックに焦点

を当てて議論をしている[34].

その後，データからの要求獲得の方法として機械学習の適用が広がりつつある[27].

このような研究の背景には隣接分野と言えるマーケティングにおいて，データ駆動マーケティング(Data-Driven Marketing)が 2010 年頃から活発に議論されていることがある[23]．さらに，e-コマース，SNS の普及に伴い，アプリケーションユーザのレビューや購買履歴，推薦などのユーザ要求につながる多くのデータの集積と活用の広がりがある．

6.2 連続的ソフトウェア工学におけるデータ駆動意思決定

連続的ソフトウェア工学(Continuous Software Engineering)はアジャイル開発の概念は組込みソフトウェア開発で実現することを課題として始まったヨーロッパを中心とする研究コミュニティである．この中で，2019 年に開催されたワークショップ DDrSE(Data-Driven Decision, Experimentation and Evolution)2019 において，顧客経験などの連続的実験(Continuous Experimentation)から得られるデータに基づき迅速な意思決定，フィールドバックを行うアプローチが提示された[18]．この報告では機械学習は主眼とはなっていない．しかし，顧客経験データの活用に光を当てた点でデータ駆動ソフトウェア工学への萌芽と考えられる．

6.3 データ駆動プロセス発見とプロセスマイニング

データを活用するソフトウェア工学の技術としてプロセスマイニングが挙げることができる[1]．データ駆動プロセス発見(Data-Driven Process Discovery)とも呼ぶ．

プロセスマイニングは業務活動や機器のオペレーションから得られるイベントログ，すなわちデジタルツインからプロセスの構造や挙動を生成する技術である．リバースエンジニアリングの一種と言える．Aalst らによるプロセスマイニングツールの発展は商用ツールの提供と共に産業界での応用が広がっている．

図 3 にプロセスマイニングの一般的なプロセスを示す[2].

ここで，ステージ 1 では人手による活動が必要である．さらに，生成されたプロセスは個々の機器のイベントログから変換された制御フローであるように必ずしも抽象化されているとは限らない．

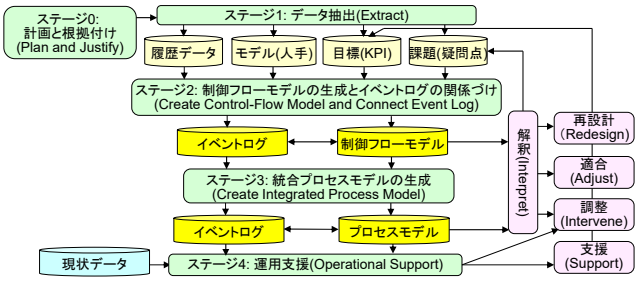

図 4 プロセスマイニングのプロセス

Fig. 4 A Generic Process of Process Mining

7. データ駆動ソフトウェアの構成モデル

Software 2.0，MLS などの議論に基づきデータ駆動ソフトウェアを抽象化した構成モデルを従来のモデル駆動ソフトウェアの構成モデルを含めて図 5 に示す.

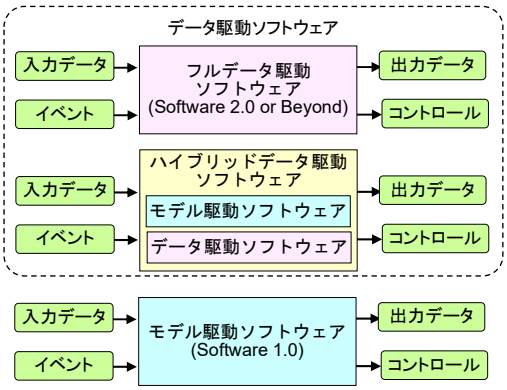

図 5 モデル駆動ソフトウェアから
データ駆動ソフトウェアへ

Fig. 5 From Model-Driven Software to Data-Driven Software

図ではモデル駆動ソフトウェアに加え，データ駆動ソフトウェアを次の 2 つに分けている.

(1) モデル駆動ソフトウェア
開発者が定義したモデルに基づきアルゴリズムを実行するソフトウェア

(2) ハイブリッドデータ駆動ソフトウェア
図 3 に示した機械学習サブシステムなどのデータ駆動ソフトウェアとモデル駆動ソフトウェアが連携して実行するソフトウェア.

(3) フルデータ駆動ソフトウェア
データ駆動ソフトウェアがデータに基づく自律的に実行するソフトウェア.

さらに，ハイブリッドデータ駆動ソフトウェアとフルデータ駆動ソフトウェアをまとめてデータ駆動ソフトウェアと呼ぶこととする.

8. モデル駆動ソフトウェア工学とデータ駆動ソフトウェア工学のフレームワーク

モデル駆動ソフトウェアとデータ駆動ソフトウェアの分類に基づき，モデル駆動ソフトウェア工学とデータ駆動ソフトウェア工学のフレームワークを議論する.

8.1 モデル駆動ソフトウェア工学のフレームワーク

本稿では，従来のソフトウェア工学をモデル駆動ソフトウェア工学(MDSE: Model-Driven Software Engineering)と呼ぶ．類似の概念としてモデル駆動開発(MDD: Model-Driven Development)があるが本質的に同一と考えられる[9]．従って，MDD の基礎であるモデル駆動アーキテクチャ(MDA: Model-Driven Architecture)もモデル駆動ソフトウェア工学に包摂されると考えている．このフレームワークを図 6 に

示す．

Pressman らによるソフトウェア工学の教科書[41]に示されているように，ソフトウェア開発プロセスにおいて最初に行うべき活動として対象システム(SUS: System Under Study)のモデル化がある．一般に，モデルは次の 3 つの性質を満たすことが求められる[31]．

(1) SUS との対応が可能(SUS の写像となっている)
(2) 簡略化(人が扱えるようにデータ量の削減)
(3) 工学的利用可能

Seidewitz が指摘するように，このモデルは対象システムの仕様となることから，以降のソフトウェア開発はこのモデルに基づく[45]．このような仕様としてのモデルはソフトウェア以外の伝統的工学においても共通である．

モデル化では SUS の観測を通して得られる観測可能なデータや SUS に関するドキュメントなどを人手により取捨選択し，モデルが構築される．そのため，モデル化に用いられるデータ量は人が扱えるように削減される．一方，そのようにして構築されたモデルの妥当性を確認することは困難である．

ここで，SUS を取り巻くコンテキストはユースケース図などで定義されるように，限定され，固定される．これは人が考える範囲を限定し，扱うデータ量を削減することを意味する．

さらに，モデルの複雑度を軽減するために，Gero の提案したように 3 つの異なる視点 FBS (Function, Behavior, Structure) に分けてモデル化される[17]．この 3 つの視点からのモデルの表現は UML でサポートされている．このようなモデル化はモデル駆動ソフトウェア工学の基礎となってきた．

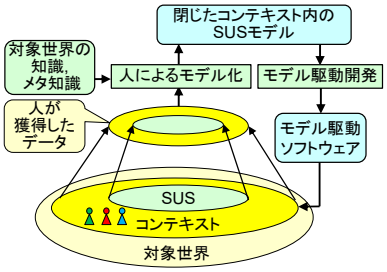

図 6 モデル駆動ソフトウェア工学
Fig. 6 Model-Driven Software Engineering

8.2 モデル駆動ソフトウェア工学の課題

モバイルアプリケーションなどの個人利用，かつ，コンテキストが変化するコンテキストアウェアソフトウェア(以下，コンテキストアウェアと呼ぶ)ではコンテキストが固定という前提は成立しなくなっている．個人毎に価値観とその価値基準が異なり，かつ，コンテキストの変化に応じて価値が変化するからである．

コンテキストアウェアでは，Chang が指摘しているようにユーザの希望や意図などは人による観測可能な情報のみでは獲得できないという課題もある[12]．

より根源的には，モデル駆動ソフトウェア工学でコンテキストが閉じているという前提は閉世界仮説(Closed-World Assumption)に基づいている．これに対して，コンテキストアウェアでは閉世界仮説が成立しないと考えることが妥当である．このような前提に立ち，Moore らはコンテキストアウェアの開発に開世界仮説の導入を提案している[38]．

閉世界仮説はもともとデータベース設計で提示されたが，そこにおいても課題が指摘されている[7]．

実際にコンテキストアウェアではスマートフォンなどからユーザの行動データを収集し，コンテキストに応じたサービスの提供が行われている．例えば，スマートフォンで実行するナビゲーションアプリケーションでも機械学習が応用されている[46]．

このような指摘はモデル駆動ソフトウェア工学において人が獲得したデータは不完全であり，モデル化における知識も不完全であり，従ってモデルも不完全であることを意味する[8]．一方，対象世界は常に変化し，不確定性(Uncertainty)を含む[29]．ここに，現代社会におけるモデル駆動ソフトウェア工学の限界があり，データ駆動ソフトウェア工学の必要性を示唆している．

8.3 データ駆動ソフトウェア工学のフレームワーク

図 5 に示したハイブリッドデータ駆動ソフトウェアに対して，これまでの議論からデータ駆動ソフトウェア工学のフレームワークを図 7 に示す．

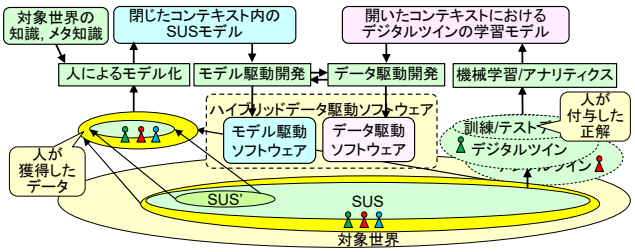

図 7 ハイブリッドデータ駆動ソフトウェアのための
データ駆動ソフトウェア工学
Fig. 7 Data-Driven Software Engineering for
Hybrid Data-Driven Software

ハイブリッドデータ駆動ソフトウェアであることからモデル　駆動ソフトウェアとデータ駆動ソフトウェア開発とが連携する必要がある．以下，データ駆動ソフトウェア開発に焦点を当て，あわせて，モデル駆動ソフトウェア開発との連携がもたらす問題を議論する．

データ駆動ソフトウェアはビッグデータを直接入力とするという前提に立つ．かつ，本稿では，モデル駆動ソフトウェアを機械学習，かつ，教師あり学習を応用するソフトウェアを想定する．

データ駆動ソフトウェアでは，モデル駆動ソフトウェアにおける人によるモデル化が訓練データによる学習に代替

される．図 8 に機械学習ソフトウェアの開発プロセスを示す．ここで，機械学習とは学習データ，テストデータに基づきフィーチャ(Feature)，あるいは表現(Representation)を選択し，それに対して評価関数の下で最適なパラメータ集合を探索する最適化問題に帰着される[14]．

このプロセスを実行するために，データ集合の抽出，学習モデルの選択，訓練データとテストデータの作成などは人手で行われる．特にデータ集合の選択は学習に大きく影響し，学習モデルの品質に直接影響する．

学習済みモデルを機械学習ソフトウェアとして実装し，実行中に実世界のデータは変化する．それに対するコンセプトドリフト，すなわち，学習モデルの不適合が起こりえる[13]．これに対応するため，学習モデルの再学習，あるいは，追加学習も必要となる．

このような人手による訓練データやテストデータの生成は対象世界の中で，データを通して機械学習のコンテキストに制約を加えることに相当する．しかし，その制約は明示的ではない点でモデル駆動ソフトウェア工学におけるコンテキスト定義とは異なるという問題を提起する．

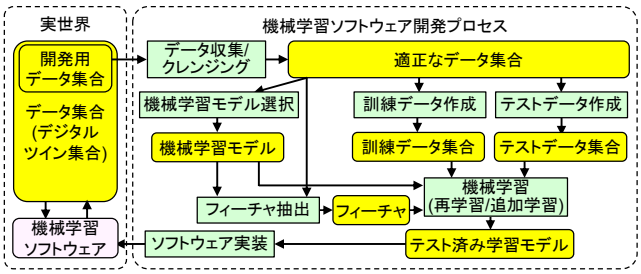

図 8 機械学習ソフトウェアの開発プロセス
Fig. 8 development Process of Machine Learning Software

8.4 機械学習ソフトウェアを対象とするデータ駆動ソフトウェア工学の課題

このような機械学習ソフトウェアの開発プロセスにおける人手による活動は実装された機械学習ソフトウェアに対して次のような影響をもたらすと考えられる．

(1) 開発プロセスに内在する課題

機械学習ソフトウェア開発プロセスにおけるデータ集合の抽出，訓練データとテストデータの抽出の方法は確立しているとは言えない．そのため，開発者の個人差などが入り込む余地がある．

また，テストデータは訓練データと独立で，かつ，学習データによる機械学習のネットワークを網羅し，かつその量は訓練データを含むデータ集合全体の10%以上にすべきとの主張がある[14][20]．テストデータ選択の基準などがあるが，学習データによる学習を網羅するテストデータを適切に選択する方法が確立されているとは言えない．

一方，データに対するフィーチャはソフトウェア工学におけるデータの属性(Attribute)に対応する．機械学習では，入力データのフィーチャ学習(表現学習とも呼ばれる)により低次元のベクトルとして生成される．フィーチャの特定などの一連の技術はフィーチャ工学(Feature Engineering)[14]として研究が行われているが，フィーチャ選択などは依然として問題である．

(2) 機械学習に起因する課題

機械学習そのものに起因する課題は表 1 に例示したように，学習過程がブラックボックスであることに起因している．そのため，機械学習を組込んだMLS の品質保証が本質的に困難となっている[39]．

また，機械学習が統計に基づく非決定的な振舞いを行うことも，その開発の根本的な課題である．

(3) モデル駆動とデータ駆動が混在することに起因する課題

MLS はモデル駆動ソフトウェアと機械学習などのデータ駆動ソフトウェアが混在する．モデル駆動ソフトウェアではシステム全体の制御やデータ駆動ソフトウェアが対応しない機能などを提供する．しかし，モデル駆動ソフトウェアとデータ駆動ソフトウェアの対象世界の中でのコンテキストは一致するとは限らない．さらに，モデル駆動ソフトウェア工学で得られるモデルとデータ駆動ソフトウェア工学で得られる学習モデルの抽象度は異なると考えられる．

また，2 つのソフトウェアの振舞いはアルゴリズムによる決定的と統計に基づく非決定的と，異なることからその連携にはアーキテクチャのミスマッチが起こりえる．

9. 問題空間と解空間の分類に基づくデータ駆動ソフトウェア工学の議論

モデル駆動ソフトウェア工学とデータ駆動ソフトウェア工学の位置づけを問題空間の開放性と解空間のアプローチから分類し，例とともに図 9 に示す．

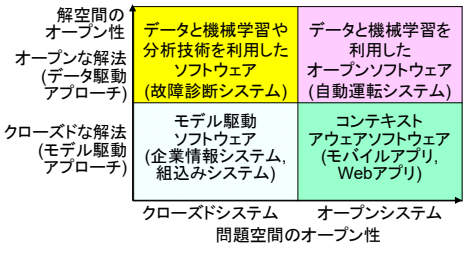

図 9 問題空間の開放性と
モデル駆動/データ駆動アプローチ
Fig. 9 Openness of Problem Space and
Model-Driven/Data-Driven Approaches

これまでのソフトウェア工学では，コンテキストを定義することによりクローズドシステムを対象としてきた．それによって，人手によるモデル化が可能となり，そのモデルをアルゴリズムに変換し，ソフトウェアとして実装してきた．本稿では，このようなソフトウェアをモデル駆動ソフトウェアと呼び，その開発技術をモデル駆動ソフトウェア工学として議論してきた．

モデル駆動ソフトウェアは企業情報処理システムや組込みシステムなど，コンテキストの変化が極めて限定される場合や変化が緩やかな場合は有効に働いた．しかし，モバイルアプリケーションなどのコンテキストアウェアにおいてはその前提が成立しなくなった．従って，モデルの不確定を考慮に入れる必要がでてきている[11].

コンテキストアウェアに対してモデル駆動ソフトウェア工学で対応する一つのアプローチとして，複数のコンテキストを設定することにより近似解を提供することがある．しかし，コンテキストアウェアにおける位置や時刻に加え，個人毎の嗜好などへの対応はモデル駆動ソフトウェア工学のアプローチでは限界があるといえる．ここに，データ駆動ソフトウェアの可能性があるといえる．

コンテキストに制約のある例として故障診断システムが想定できる．特定の機器が生成するデータはその稼働状況などにより変動範囲が制約されるからである．

一方，自動運転における認知に機械学習が適用されている．このような問題では自動車だけでなくその周囲に他の自動車，人，モノ，あるいは天候など多様なコンテキストの要素がリアルタイムに変化する．このため，コンテキストを制約することは困難である．従って，高速道路などコンテキストを制限できるユースケース(ODD: Operational Design Domain と呼ばれる)から適用を始める方法がとられている．

これらの議論をまとめるとモデル駆動ソフトウェア，データ駆動ソフトウェアのいずれにおいても従来は必ずしも焦点が当てられてこなかったコンテキストの重要性を指摘することができる．

10. データ駆動ソフトウェア工学の課題と展望

データ駆動ソフトウェアとデータ駆動ソフトウェア工学には多くの未解決な技術課題がある．

また，本稿では触れなかったがデータ駆動ソフトウェアのプラットフォームとしてデータプラットフォームの構築も大きな課題である．これまでのソフトウェア工学では開発作業を支援する機能の提供が主眼であった．データ駆動ソフトウェアでは図 10 に示すように，データの収集から分析，利用へ至るデータのライフサイクルに沿った支援が必要となる[26]．このようなデータプラットフォームと併せて，企業や社会の DX を推進し，その運用基盤となるデジタルプラットフォームの構築も求められている[43].

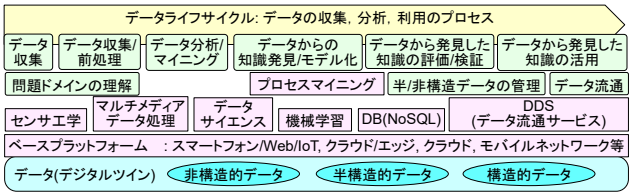

図 10 データプラットフォーム
Fig. 10 Data Platform

11. まとめ

DX によりユーザ起点の課題解決を通した新たな価値創出が期待されている．そのため，顧客の行動データや機器のイベントログなどからデータを活用した課題発見技術が求められている．機械学習はその有力なアプローチとして多くの研究と適用が進んでいる．しかし，機械学習を組込んだ MLS には多くの課題があり，その開発方法も確立しているとは言えない．

本稿では MLS を含むデータに基づくソフトウェアをデータ駆動ソフトウェアとして定義し，その開発技術をデータ駆動ソフトウェア工学として，他の工学分野とも照らして議論した．

データ駆動ソフトウェア工学は本稿でモデル駆動ソフトウェア工学としてモデル化した従来のソフトウェア工学の拡張として新たな可能性をもたらすと期待できる．しかし，その基礎から応用に至るあらゆる面で多くの課題がある．ここに多くの研究の可能性があるといえる．

ソフトウェア工学の研究者，実務者の皆様がこのような課題に取り組まれることを期待する．

謝辞

本稿をまとめるにあたって，丸山宏氏による講演，ならびに同氏との議論から刺激と洞察を得たことに感謝する．

参考文献

[1] W. M. P. van der Aalst, Process Mining, 2nd ed., Springer 2016 [青山 幹雄 (監修), プロセスマイニング，インプレス, 2019].
[2] W. M. P. van der Aslst, et al., Process Mining Manifesto, Proc. of BPM 2011 Workshop, Part 1, LNBIP Vol. 99, Springer, Aug. 2011, pp. 169-194.
[3] 青山 幹雄，ソフトウェア工学基礎から機械学習ソフトウェア工学基礎への考察，ソフトウェア工学の基礎 XXVI，日本ソフトウェア科学会/近代科学社, Nov. 2019, pp. 139-144.
[4] 青山 幹雄, DX(デジタルトランスフォーメーション)とは何か?: DX の現状と展望，情報処理技術の課題と機会，情報処理，Vol. 61, No. 11, Nov. 2020, pp. 1-7 (オンライン版).
[5] 青山 幹雄, DX が提起する人材，教育，雇用のデジタル化，情報処理，Vol. 61, No. 11, Nov. 2020, pp. 1-5 (オンライン版).
[6] 東 将大，地球をまるっとデジタル化，日経エレクトロニクス，Vol. 1221, Nov. 2020, pp. 24-45.
[7] A. Badia and D. Lemire, A Call to Arms: Revisiting Database Design, SIGMOD Record, Vol. 40, No. 3, ACM, Sep. 2011, pp. 61-69.
[8] G. E. P. Box, Science and Statistics, J. of the American Statistical

Association, Vol. 71, No. 356, Dec. 1976, pp. 791-799.
[9] A. W. Brown, et al., Models, Modeling, and Model-Driven Architecture, S. Beydeda, et al. (Eds.), Model-Driven Software Development, Springer, 2005, pp. 1-16.
[10] S. L. Brunton, and J. N. Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press, 2019.
[11] L. Burgueño, et al., Belief Uncertainty in Software Models, Proc. of MiSE 2019, ACM/IEEE, May 2019, pp. 19-26.
[12] C. K. Chang, Situation Analytics: A Foundation for a New Software Engineering Paradigm, IEEE Computer, Vol. 49, No. 1, Jan. 2016, pp. 24-33.
[13] M. Chui, et al., Notes from the AI Frontier: Insights from Hundreds of Use Cases, Discussion Paper, McKinsey & Company, Apr. 2018, pp. 1-32.
[14] P. Duboue, The Art of Feature Engineering, Cambridge University Press, 2020.
[15] 藤本 玲子, 青山 幹雄, データ駆動要求工学の提案とステークホルダ分析への適用評価, 第 191 回ソフトウェア工学研究会, Vol. 2016-SE-191, No. 15, 情報処理学会, Mar. 2016, pp. 1-8.
[16] D. Gelernter, Mirror Worlds: or the Day Software Puts the Universe in a Shoebox…,How It Will Happen and What It Mean, Oxford University Press, 1991.
[17] J. S. Gero, Design Prototypes: A Knowledge Representation Schema for Design, AI Magazine, Vol. 11, No. 4, AAAI, Dec. 1990, pp. 26-36.
[18] I. Gerostathopoulos, et al., Continuous Data-Driven Software Engineering - Towards a Research Agenda, SIGSOFT Software Engineering Notes, Vol. 44, No. 3, ACM, Jul. 2019, pp. 60-64.
[19] M. Grieves, et al., Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems, F.-J. Kahlen, et al. (Eds), Transdisciplinary Perspectives on Complex Systems, Springer, 2017, pp. 85-113.
[20] M. T. Hagan, et al., Neural Network Design, 2nd ed., Martin Hagan, 2014.
[21] HERE, HERE Maps, https://www.here.com/platform/mapping/map-data.
[22] T. Hey, et al. (Eds), The Fourth Paradigm: Data-Intensive Science Discovery, Microsoft Research, 2009.
[23] M. Jeffery, Data-Driven Marketing, John Wiley & Sons, 2010 [佐藤 純, ほか (訳), データ・ドリブン・マーケティング, ダイヤモンド社, 2017].
[24] A. Karpathy, Software 2.0, Nov. 2017, https://medium.com/@karpathy/software-2-0-a64152b37c35.
[25] K. Kersting, et al., SE4ML - Software Engineering for AI-ML-Based Systems, Dagstuhl Reports, Vol. 10, No. 2, 2020, pp. 76-87, https://drops.dagstuhl.de/opus/volltexte/2020/13060.
[26] M. Kim, Software Engineering for Data Analytics, IEEE Software, Vol. 37, No. 4, Jul.-Aug. 2020, pp. 36-42.
[27] 久保井 恵里香, ほか, 深層学習(RNN/LSTM)を用いた 2 段階発話意図分析方法の提案とソフトウェア開発会議への適用評価, SES2018 論文集, 情報処理学会, Sep. 2018, pp. 64-73.
[28] J. M. Kutz, Data-Driven Modeking and Scientific Computation, Oxford University Press, 2013.
[29] P. A. Laplante and C. J. Neill, Uncertainty: A Meta-Property of Software, Proc. of SEW 2005, IEEE/NASA, Apr. 2005, pp. 228-233.
[30] C. Ley, et al., Data-Centric Engineering in Modern Science from the Perspective of a Statistician, an Engineer, and a Software Developer, Data-Centric Engineering, Vol. 1, Article No. e2, Jun. 2020, pp. 1-10.
[31] B. A. Lieberman, The Art of Software Modeling, CRC Press, 2019.
[32] Y. A. Loukissas, All Data are Local: Thinking Critically in a Data-Driven Society, MIT Press, 2019.
[33] W. Maalej, et al., Toward Data-Driven Requirements Engineering, IEEE Software, Vol. 33, No. 1, Jan.-Feb. 2016, pp. 48-56.
[34] W. Maalej, et al., Data-Driven Requirements Engineering: An Update, Proc. of ICSE 2019-SEIP, IEEE Computer Society, May 2019, pp. 289-290.
[35] W. Maass, Data-Driven Meets Theory-Driven Research in the Era of Big Data: Opportunities and Challenges for Information Systems Research, J. of AIS, Vol. 19, No. 12, Dec. 2018, pp. 1253-1273.
[36] 丸山 宏, 城戸 隆, 機械学習工学へのいざない, 人工知能, Vol. 33, No. 9, Mar. 2018, pp. 124-131.
[37] E. Meijer, Behind Every Great Deep Learning Framework is an Even Greater Programming Languages Concept, Proc. ESEC/FSE 2018, ACM, Oct. 2018, p. 1.
[38] P. Moore and H. V. Pham, On Context and the Open World Assumption, Proc. of ICAES-2015 (ICAES-2015 Workshop), IEEE Computer Society, Mar. 2015, pp. 387-392.
[39] K. Nakamichi, et al., Requirements-Driven Method to Determine Quality Characteristics and Measurements for Machine Learning Software and Its Evaluation, Proc. RE 2020, IEEE Computer Society, Sep. 2020, pp. 280-290.
[40] A. Pentland, The Data-Driven Society, Scientific American, Vol. 309, No. 4, Oct. 2013, pp. 78-83.
[41] R. S. Pressman, and B. R. Maxim, Software Engineering, 8th Ed., McGraw Hill, 2015.
[42] O. Ratzesberger, et al., The Sentient Enterprise, Wiley, 2017 [杉田 真 (訳), データ駆動型企業, 日経 BP 社, 2019].
[43] M. de Reuver1, et al., The Digital Platform: A Research Agenda, J. of Information Technology, Vol. 33, Jun. 2018, pp. 124-135.
[44] D. Sculley et al., Machine Learning: The High Interest Credit Card of Technical Debt, Proc. of SE4ML 2014 (NIPS Workshop), Dec. 2014, pp. 1-9.
[45] E. Seidewitz, What Models Mean, IEEE Software, Vol. 20, No. 5, Sep.-Oct. 2003, pp. 26-32.
[46] 田中 優之, 青山 幹雄, 機械学習ソフトウェアシステムの環境変化適応の課題とアプローチ: スマートフォンのナビゲーションアプリケーションを例として, 第 2 回機械学習工学研究会(MLSE 夏合宿 2019)論文集, 日本ソフトウェア科学会, Jul. 2019, pp.49-54.
[47] F. Tao, et al., Digital Twin in Industry: State-of-the-Art, IEEE Trans. on Industrial Informatics, Vol. 15, No. 4, Apr. 2019, pp. 2405-2415.