制御機器に同居したITアプリケーション環境の提案

西垣 弘二，荒井 航

近年、生産現場では、制御機器の生み出すデータを活用し、情報処理技術を活かした生産性や歩留りの向上、生産立ち上げ期間の短縮といったニーズが増えてきている。

そのニーズは多様化しており、また、情報処理技術は向上し続けることから、現場のニーズに素早く対応し、かつ最新の情報処理技術を用いたアプリケーションを導入できる環境が求められている。

さらに、従来の手段では、一度情報処理層にデータを持ち出すことでIT技術者の手によるデータ解析などが行われており、生産現場でのタイムリーなデータ活用が困難であった。

そこで我々は、コントローラ内部に制御と同居した情報処理のためのアプリケーションプラットフォームを組み込み、さらに生産現場で柔軟にかつ強力なアプリケーションを開発するためのアプリケーションフレームワークを提案する事で、制御によって生み出される質の良い高精度データを生産現場において収集・活用できるようにした。

Proposing an IT Application Platform in Controller

NISHIGAKI Koji and ARAI Wataru

In recent years, there has been an increasing need for production to improve by utilizing information processing technology by utilizing the data generated by control equipment at production sites.

Since the needs are diversifying and the improvement of information processing technology does not stop, there is a need for an environment where applications that can quickly respond to the needs of the field and introduce applications using the latest information processing technology are required.

Furthermore, in the conventional means, data analysis by IT engineers is performed by taking out data to the information processing layer once, and it is difficult to utilize data to demonstrate the strengths of the site at the production site.

Therefore, we incorporated an application platform for information processing that coexisted with control inside the Controller and proposed an application framework for developing flexible and powerful applications at the production site, so that high-speed and high-precision data generated by control can be collected and utilized at the production site.

1. まえがき

従来、ファクトリーオートメーション（以下、FA）では、データベース接続やFA分野におけるデータ交換方法の国際標準規格であるOPC UAなどを用い、制御機器が生成するデータを情報処理層に取り込み、情報処理技術（以下、IT）を活用して生産改善に活かしてきた。しかしながら、近年、装置を制御するコントローラが高速・高精度化し、データ更新周期の短縮による同期したデータへのアクセス可能時間の減少や単位時間当たりのデータ量の増大による通信負荷の増大、およびデータアクセス負荷の増大による制御周期への影響など、制御周期に同期したデータを漏れなく情報処理層に取り込むことは困難になって来ている。

また、この様な高速・定周期でサンプリングされた質の良い高精度データ（以下、質の良い高精度データ）の活用には高度なITスキルが必要であることが生産現場における改善活動でのデータ活用の妨げになっており、生産現場における継続的改善といった現場の強みを十分に活用できていない。

一方、オムロンが開発したAI搭載マシンオートメーションコントローラ（以下、AIコントローラ）では、コントローラに時系列データベース（以下、TSDB）と呼ばれる高速に高精度なデータを収集・蓄積するデータベースを搭載しており、蓄積したデータを解析し、解析結果から生成したAI機械学習モデルによって外れ値を検知することに

Contact : *NISHIGAKI Koji* koji.nishigaki@omron.com

より予知保全などのアプリケーションを実現している[1)]。

しかし、第四次産業革命[2)] を目指す取り組みが加速する中、生産現場でのデータ活用の取り組みは、AI コントローラが実現している AI を使ったアプリケーションにとどまらず、様々なアプリケーションが構想・提案されている。

オムロンも、現場データ活用サービス「i-BELT」を立ち上げ、生産現場におけるデータ活用の取り組みを加速させている[3)]。また、知能化セルライン（Cell Line Control system 以下、CLCS）では、人の判断や作業を支援するために、様々なデータの連携が必要とされている[4)]。さらに、オムロンが進めている様々な共創において、お客様独自のアプリケーション（データ活用手段）を実現するためにコントローラが生み出す質の良い高精度データとデータを扱うための情報処理環境が必要とされている。

本稿では、生産現場でのデータの活用を目的とし、従来制御を主としてきたコントローラに、質の良い高精度データを収集し IT を使って活用するための能力を付与するための設計と、現場で時系列データを活用するアプリケーションを創り出す活動（以下、アプリ創造活動）を支えるアプリケーションフレームワークについて述べる。

アプリケーションフレームワークは、既存のオムロンのコントローラの一つであるデータベース接続 CPU ユニット（以下、DB モデル）を拡張し、プロセス間通信を用いた変数アクセス API と Java 実行環境を利用する事で、コントローラ内で動作する様々な IT アプリケーションを開発・実行するための仕組みを構築、制御への影響を抑えながら、コントローラの制御周期と同期した質の良い高精度データを利用した様々なアプリケーションを実行可能とすることを目指す。

また、データ処理に特化したアプリケーションを作成するためのパイプラインアプリケーションフレームワークを提案し、機能毎に用意された Node と呼ばれるソフトウェア部品を選択し、そのつながりを定義することで、IT スキルの高くない現場作業員が、データの取得・処理・フィードバックを行えるアプリケーションを容易に実現できるようにした。

2. 課題

2.1 情報処理との共存による制御への影響

製造現場で用いられるコントローラは、あらかじめ定められた順序に従い制御を行うシーケンス制御やモータの位置制御を行うモーション制御などを行う制御機器である。オムロンのコントローラの場合、その制御周期は 0.125 ms から数 ms であり、高速・高精度制御をリアルタイムに実行している。

コントローラ内で情報処理を実行する場合、情報処理による影響で制御プロセスの実行時間にばらつきが生じると、制御が正しく行われなくなる。そのため、コントローラ内で情報処理を実行するにあたっては、制御周期を乱さないことが必須の要件となる。

一方、AI に代表される高度なデータ解析には、コントローラの高速な制御周期と同期した高精度な時系列データが必要であるが、従来の手段では収集間隔が数 ms 程度にとどまり、かつ収集間隔のゆらぎが存在したため、要件を満たすことができなかった。

2.2 IT の FA 現場への適用

生産現場では、IT に習熟した要員が常に存在するわけでは無い。そのため、生産現場の日々の改善活動において、IT を使ってコントローラが生み出すデータを活用するためには、IT に不慣れな現場作業員でも容易に IT を活用できるような仕組みを用意する必要がある。

例えば AI コントローラでは、解析用データを収集し、AI 機械学習モデルを作成、AI による外れ値検知を高速に行い、装置の故障を予測することなどが可能であるが、現場に適用する際には収集対象データの選択や AI 機械学習モデルの作成などに高度な IT 知識が必要である（図 1、2 参照）[1)]。

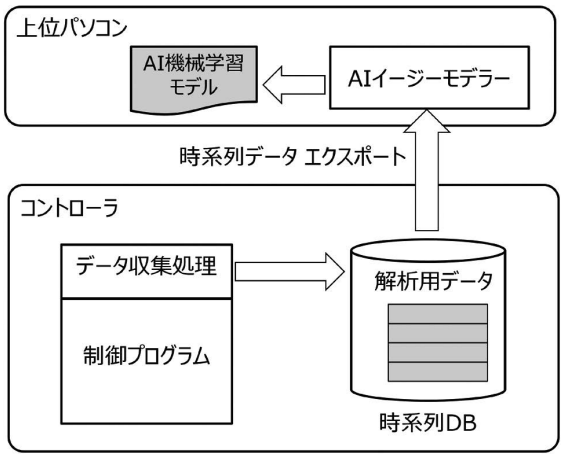


図 1　AI コントローラにおける分析フェーズの処理フロー

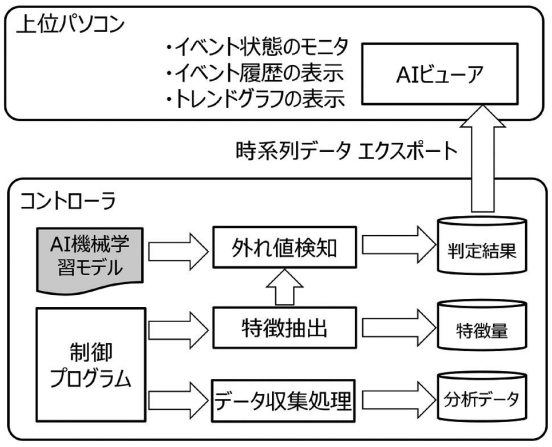


図 2　AI コントローラにおける活用フェーズの処理フロー

一方、FA に不慣れな IT 技術者には、FA に特有の知識を有していなくても IT を FA に適用できる仕組みが必要となる。

3. 技術内容

3.1 アプリケーションプラットフォーム

本稿では、制御を専門とする従来のコントローラに対し、制御周期に影響を与えることなくコントローラ内での情報処理アプリを実現する手段を提案する。

従来のコントローラでは、PLC Engine によるリアルタイム制御を実現し、その中の DB モデルでは、Java VM 上で動作する DB Connection Application を搭載してリレーショナルデータベース（以下、DB）接続機能を実現している（図 3 参照）。

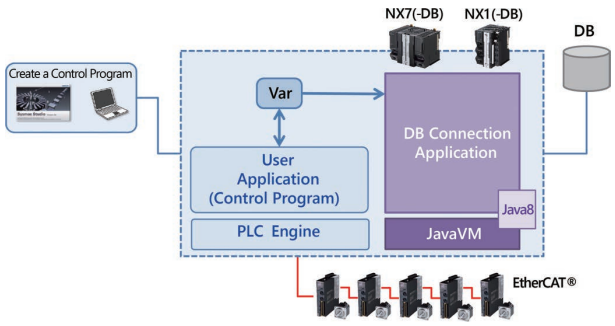


図 3　DB モデルシステム構成

DB モデルでは、プロセス間通信を用いた変数アクセス API と Java 実行環境を活用することで、非同期プロセスを Java に集約し、Java の実行優先度を低くすることで制御に与える影響を抑えながら、コントローラから DB へのデータ入力や、ストアドプロシージャを使用した DB 操作を行うアプリケーションを実現している。

本稿で提案する手段の適応例として、DB モデルを拡張し、制御と同居しながら IT 処理を実行しつつコントローラが実現している高速・高精度な制御に影響を与えることなく、コントローラ上で動作するアプリケーションを実現するための Sysmac Java Application Platform（以下、アプリケーションプラットフォーム）を開発した。アプリケーションプラットフォームは、DB モデルの既存の Java 実行環境を活用し、Java ベースのアプリケーションプラットフォームとして Java 実行環境上に構築した。アプリケーションプラットフォームでは、Pipeline Application Framework（以下、パイプラインアプリケーションフレームワーク。詳細は 3.2 章で説明）を含む、Java で実装された様々なアプリケーションを実行・管理できる（図 4 参照）。

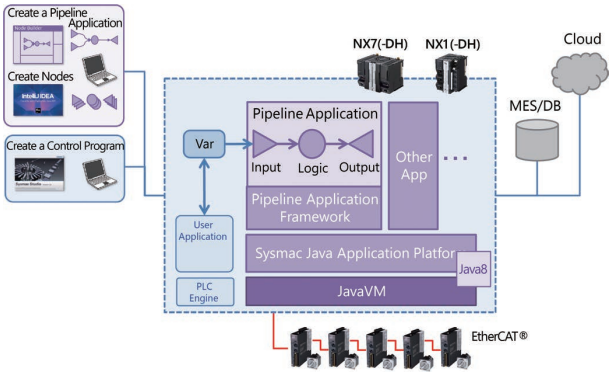


図 4　アプリケーションプラットフォーム

アプリケーションプラットフォームは、オープンソースソフトウェア（以下、OSS）の Vert.x[5] を利用し、コントローラが起動しアプリケーションプラットフォームが起動した後はコントローラの状態やモードに関係なく独立してアプリケーションを実行できる（図 5、6 参照）。

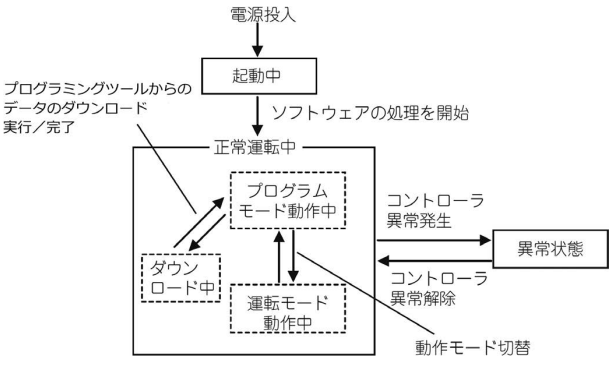


図 5　コントローラの状態・モード[6]

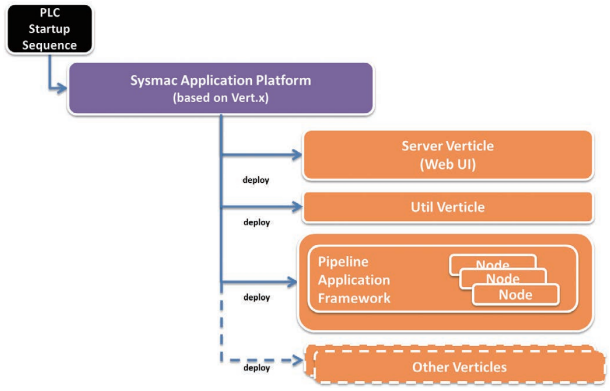


図 6　アプリケーションプラットフォームの起動シーケンス

アプリケーションは、Vert.x 上で動作する Verticle（Vert.x で管理されるアプリケーションの実行単位）として実装する事で、コントローラ内で Java アプリケーションとして動作する。よって、OSS を含めた様々な Java ライブラリを利用したアプリケーションをコントローラ上で実現することができる（Java プログラムからネイティブライブラリに

アクセスするための手段である JNA/JNI を使う事で C 言語などを用いて実装されたライブラリも利用可能)。

アプリケーションプラットフォームは、制御に影響を与えないようするために、制御を実行するプロセスよりも優先順位の低いプロセスとして実行される。アプリケーションはアプリケーションプラットフォーム上で動作するため、制御を実行するプロセスの空き時間で非リアルタイムに実行され、制御プロセスの実行を妨げない。

また、アプリケーションプラットフォームでは、コントローラ内部の制御リソースへのアクセス制限を行うことで、アプリケーションが制御プロセスのリソースアクセスを妨げないようにした。

さらに、既存の変数アクセス API を拡張し、かつ制御プロセスで実行されるプログラムと連携する仕組みを取り入れる事で、アプリケーションからより高速に制御周期と同期したデータを収集できるようにした。変数アクセスの際、コントローラが提供している API では、プロセス間通信のコマンド―レスポンスのプロトコルが用いられており、1 変数アクセス毎のオーバーヘッドが大きい。そこで、収集対象の変数を制御プロセスで実行されるプログラムで一時的に配列に保存し、収集対象を配列にすることで、実質的な 1 変数当たりのオーバーヘッドを小さくした。この時、コントローラ内部での通信であることから、大量の変数データを受信した場合に要する時間は十分小さく、全体の時間を削減できる（図 7、8 参照)。

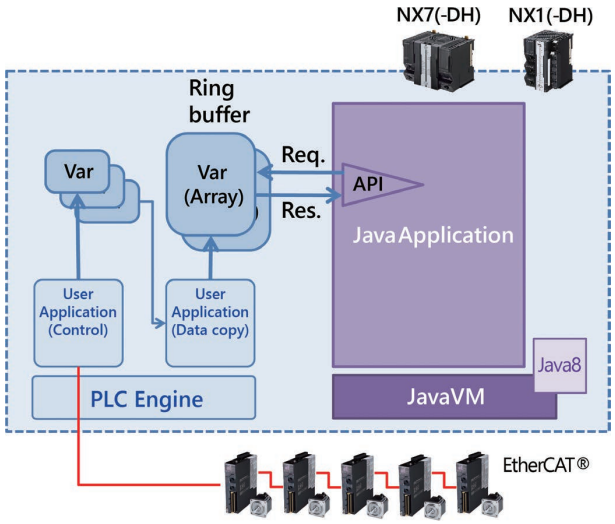


図 7　制御プログラムと連携した変数アクセス

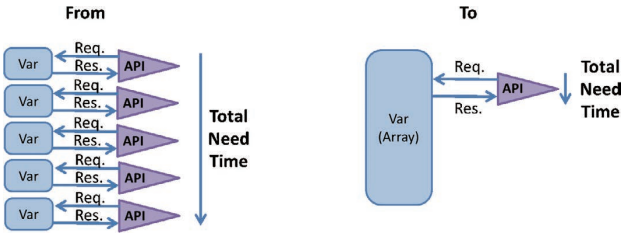


図 8　変数アクセスに必要な処理数と時間

3.2 パイプラインアプリケーションフレームワーク

本稿で提案する手段の適応例として、高度な IT スキルを持たない FA 技術者が、必要な機能の組み合わせとパラメータの設定のみで必要なデータの収集・処理・出力を行うパイプラインアプリケーションフレームワークを開発した。

パイプラインアプリケーションフレームワークでは、データを入力し、入力したデータを加工・処理し、加工・処理した結果を出力できるパイプラインアプリケーションを実現できる。

また、パイプラインアプリケーションフレームワークでは、データ入力（Input：図 9 の右向き△)・データ処理（Logic：図 9 の○)・データ出力（Output：図 9 の左向き△）の 3 種類の機能を、それぞれ Input Node、Logic Node、Output Node といった Node という形でソフトウェア部品化し、GUI ツールを使用して Node の定義と Node 間の関係の定義を行う事で、入力したデータをメッセージ（Message：図 9 の□）として受け渡し、処理した上で出力するアプリケーションを簡単に実現・変更できる（図 9、10 参照)。

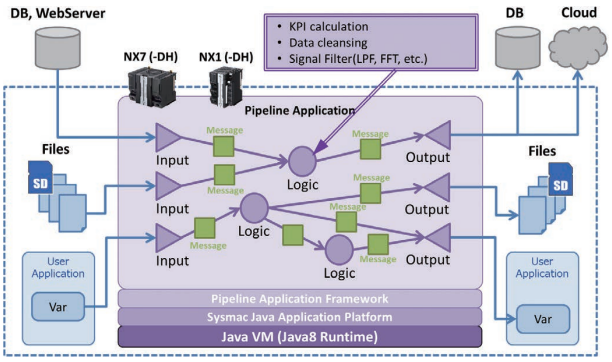


図 9　パイプラインアプリケーションフレームワーク

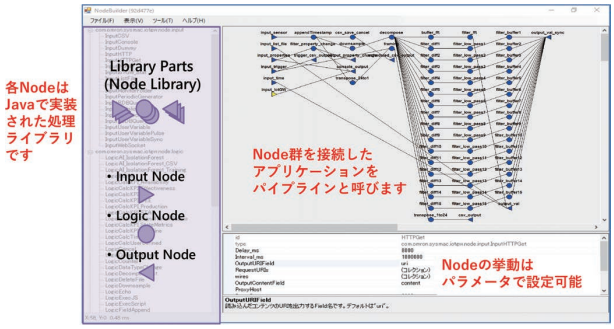


図 10　GUI ツール

さらに、FA 現場技術者が活用したい IT を IT 技術者が Node という形で開発することで、新しい機能をアプリケーションとして容易に FA 現場に導入することができる（IT 技術者と FA 現場技術者の橋渡しを実現）。

アプリケーションプラットフォームとパイプラインアプリを活用することで、様々な用途のアプリケーションを実現することができた。実現したアプリケーションの詳細は 5. 章に記述する。

4. 検証（制御性能への影響と性能）

4.1 制御性能への影響の検証

制御性能への影響を検証するために、オムロンのコントローラである NX102 および NX701 において、本稿の技術を搭載した NX102/NX701 データベース接続 CPU ユニットの時系列データ収集システム搭載商品（以下、TSDC モデル）でデータ収集を実行した場合のタスク実行時間への影響を検証した。

TSDC モデルでデータ収集を実行しなかった場合とデータ収集を実行した場合のベンチマーク用の制御プログラムにおけるタスク実行時間（最大/最小/平均/標準偏差）を測定し、比較した。NX701 では制御周期を 1ms として 4008 サイクル実行、NX102 では制御周期を 2ms として 9855 サイクル実行し測定を行った。測定結果は下表の通りである（表 1、2 参照）。

表 1　NX102 の結果（単位 μs）

	データ収集 非実行時	データ収集 実行時	差
最大	1060.205	1097.815	37.61
最小	698.495	699.665	1.17
平均	789.339	866.413	77.074
標準偏差	53.448	65.081	11.633

表 2　NX701 の結果（単位 μs）

	データ収集 非実行時	データ収集 実行時	差
最大	622.05	702.664	80.614
最小	584.47	586.075	1.605
平均	598.50	602.995	4.495
標準偏差	5.85	8.444	2.594

測定結果から、データ収集実行時はデータ収集非実行時と比較して、タスク実行時間の最大／最小／平均／標準偏差が少し大きくなることが確認できた。

一方、NX シリーズ CPU ユニットの場合、タスク実行時間の最大値の目安の計算式は、コントローラのマニュアルから、

([タスク実行時間の平均値]+([タスク実行時間の平均値]−[タスク実行時間の最小値]))×1.2+25 μs

と定義されている[6)]。

そこで、TSDC モデルでデータ収集を実行した場合のタスク実行時間の最大値の目安を計算し、計算結果が制御周期を超えないことと測定した最大値が計算結果を超えないことを確認した。

計算式にそれぞれのデータ収集実行時の測定結果を代入すると、NX102 では、

(866 μs+(866 μs−699 μs))×1.2+25 μs=1264 μs となり、制御周期の 2000μs を下回り、かつデータ収集実行時の最大値 1097 μs はこれを下回っている。

また、NX701 では、

(602 μs+(602 μs−586 μs))×1.2+25 μs = 766 μs となり、制御周期の 1000μs を下回り、かつデータ収集実行時の最大値 702 μs はこれを下回っている。

このことから、データ収集の影響で制御実行時間とばらつきが増大するものの、制御周期のリアルタイム性を損なうものではないことを確認できた。

4.2 データ収集性能の評価

3. 章で説明した技術を用いた TSDC モデルでは、制御周期に影響を与えることなく制御周期に同期したデータ収集を実現できた。TSDC モデルの詳細は 5.1 章で記述する。既存の商品である DB モデルと TSDC モデルの収集性能を（表 3）に示す。

今回開発した技術では、時系列データの収集を制御プログラムで配列としてバッファリングし、バッファリングされたデータを制御の空き時間を用いてアプリケーションに一括で取り込むことで、質の良い高精度データを制御に影響を与えることなく取得・活用することを実現できた。

表 3　DB モデルとの収集性能の比較

	収集性能	
	TSDC モデル	DB モデル
NX701	4 KB/ms	2 KB/ms
NX102	0.4 KB/ms	0.25KB/ms

TSDC モデルでは、コントローラに NAS を接続し、コントローラで収集した時系列データを CSV ファイルとして NAS に保存する（図 11 参照）。

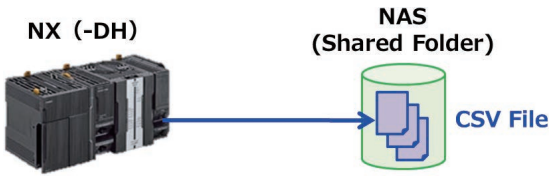


図 11　時系列データ収集システム

5. 効果（アプリ創造活動）

5.1 アプリ創造活動からの商品化サイクル

制御性能に影響を与えることなく、コントローラで時系列データの収集・活用が可能になったことで、本稿で提案する技術を使ったアプリケーションが、様々な製造現場のデータ活用の共創に用いられた。

その一つとなる自動車業界顧客における共創では、本稿で提案した技術を使用する事で、高速制御と同時に高速制御で生じる質の良い高精度データを、制御周期に影響を与えることなく収集し、コントローラから直接 NAS に保存、顧客アプリで解析し、その結果を制御にフィードバックするシステムを実現できた。その結果、一つのコントローラでのコストダウンと立ち上げ効率・保守性向上を実現した。

本共創の結果、アプリケーションの有用性が確認できたことから、用途限定商品としてデータベース接続 CPU ユニット　時系列データ収集システム搭載商品（TSDC モデル）の商品化を行った。

この様に、TSDC モデルの商品化では、共創・アプリ創造活動を通じて既に顧客現場でその有用性が証明されていることから、通常の商品開発における工程の一部を省略することができたため、比較的短期間でリリースを行い、複数顧客へ展開することができた。

さらに、継続して行われている共創では、TSDC モデルを拡張した新たなアプリケーションを開発し、新しい顧客への展開を実現するというアプリ創造活動からの商品化サイクルを実現できた。

5.2 現場データ活用サービス（i-BELT）

オムロンでは、オムロンのユニークなデータ活用サービス「i-BELT」を提供している。「i-BELT」は、生産現場にあるデータを活用し、その収集から見える化・分析・制御をお客様の課題に寄り添い、共に解決する共創サービスである（図 12 参照）。

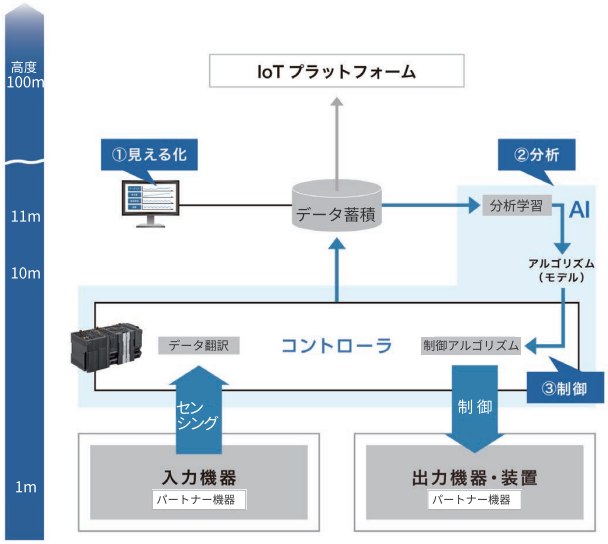


図 12　i-BELT

本稿で提案した技術を用いることで、収集された制御データは、一つのコントローラ内の利用にとどまらず、他のコントローラの制御データや情報データと結びつけて、より広範囲なデータ解析が可能になる。

5.3　データを活用した歩留り改善（社内共創）

社内共創の例では、本稿の技術を用いることで、歩留り改善のためのデータ解析に必要な、上位システムのデータなどの様々な関連するデータの紐づけ（トレース・結合・演算）を容易に実現することができた。その結果、識別コードを付与した生産条件、検査結果情報に加えて、製品一つ一つに紐づいた ILO（Input-Logic-Output）データを記録しトレースを行う事で、データ分析に基づく因果特定と結果のコントロールが可能となり、検査工程での不良排除ではなく、生産工程における前工程での不良排除が実現できた。また、部品の寸法のばらつきに応じて良品となる組み合わせを、再帰的に求めることができた。

本稿で提案した技術を用いることで、前工程での不良品排除を達成するために必要な、工程の増減に柔軟に対応し、かつ複数工程間トレース可能なデータを一元化するシステムを実現できた。コントローラに情報処理機能があることで、部材情報、検査結果、上位システム、工程データ、PC データからなるデータ構造（図 13 参照）を一元管理できるようになった。

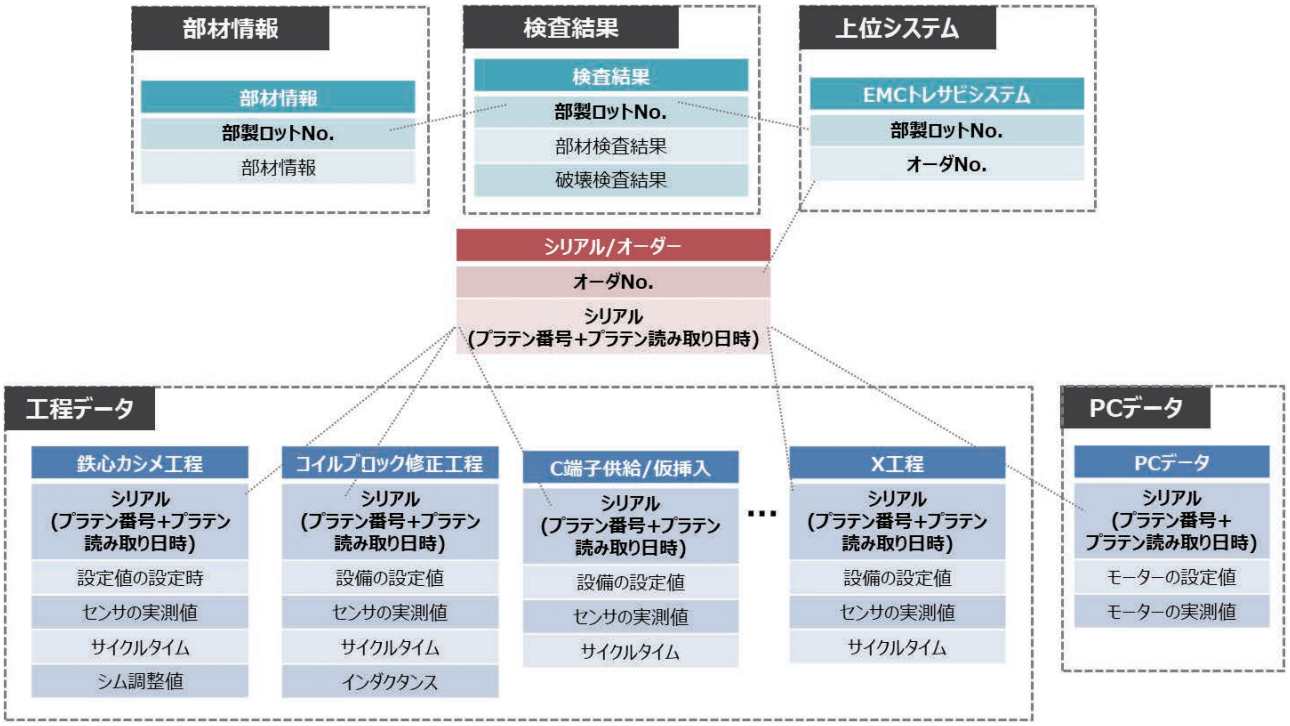


図 13　システム全体でのデータ構造図

5.4 Cell Line Control System（CLCS）

市場ニーズの多様化や需要変動の拡大など、難易度を増していくモノづくりに対し、熟練作業者は不足している。また、収益構造の見直しに伴う生産移管も加速し、作業者の早期育成や工程管理の徹底といった、4M 変動管理が複雑化している。この様な変種変量生産に対応するには、作業者に多くの判断が求められ、人手だけに頼ったままでは、作業習熟度のばらつきにより、品質担保に限界がでる。リスク管理においても、不良品を流出させてしまった場合の影響範囲の特定や、良品を証明できる情報がすぐに出せるよう、対策が必要となる。

オムロンが提供する CLCS では、工程管理トレーサビリティの導入と、デジタル作業指示による判断レス化で製品・作業の品質管理と、非熟練者でもミスのない作業を実現する。

本稿で提案する技術を用いることで、製品トレース情報と工程・作業トレース情報の一元化による工程品質管理を容易に実現することができた。また、工程や作業者の状態などの製造情報を製品に紐づけて一元管理し、カン・コツに頼らない品質ばらつき要因分析による工程改善を実現できた（図 14 参照）。

5.5 クラウド環境への拡張と業界標準プロトコルへの拡張

本稿の技術を用いることで、NX コントローラに容易にクラウド接続性を付与することができた。

Azure では、Azure 接続に必要なライブラリを公開しているが、本稿の技術を用い、公開ライブラリを取り込んで Node を開発することで、短期間で Azure 接続アプリを実現し、Azure 認証[7] を取得することができた（カタログには非掲載）。

同様に、AWS IoT[8] や MindSphere[9] といったクラウドへの接続機能や Hermes Standard[10] の様な FA で用いられる標準通信プロトコルを容易にコントローラ上に搭載することができた。

6. むすび

近年、生産現場では、制御機器の生み出すデータを活用し、情報処理技術を活かした生産性や歩留りの向上、生産立ち上げ期間の短縮といったニーズが増えてきており、現場のニーズに素早く対応し、かつ最新の情報処理技術を用いたアプリケーションを導入できる環境が求められていた。

さらに、従来の手段では、高度な IT スキルが必要であり、生産現場でタイムリーにデータ活用できる手段が求められていた。

本稿での取り組みにより、コントローラの本来の機能である制御周期に影響を与えることなく、コントローラ自身が生成する質の良い高精度データを活用した生産現場の改善に寄与できる環境をコントローラの内部に実現することができた。

また、この様な環境を活用したアプリ創造活動を通じて、コントローラ自身に情報処理機能を持たせるというア

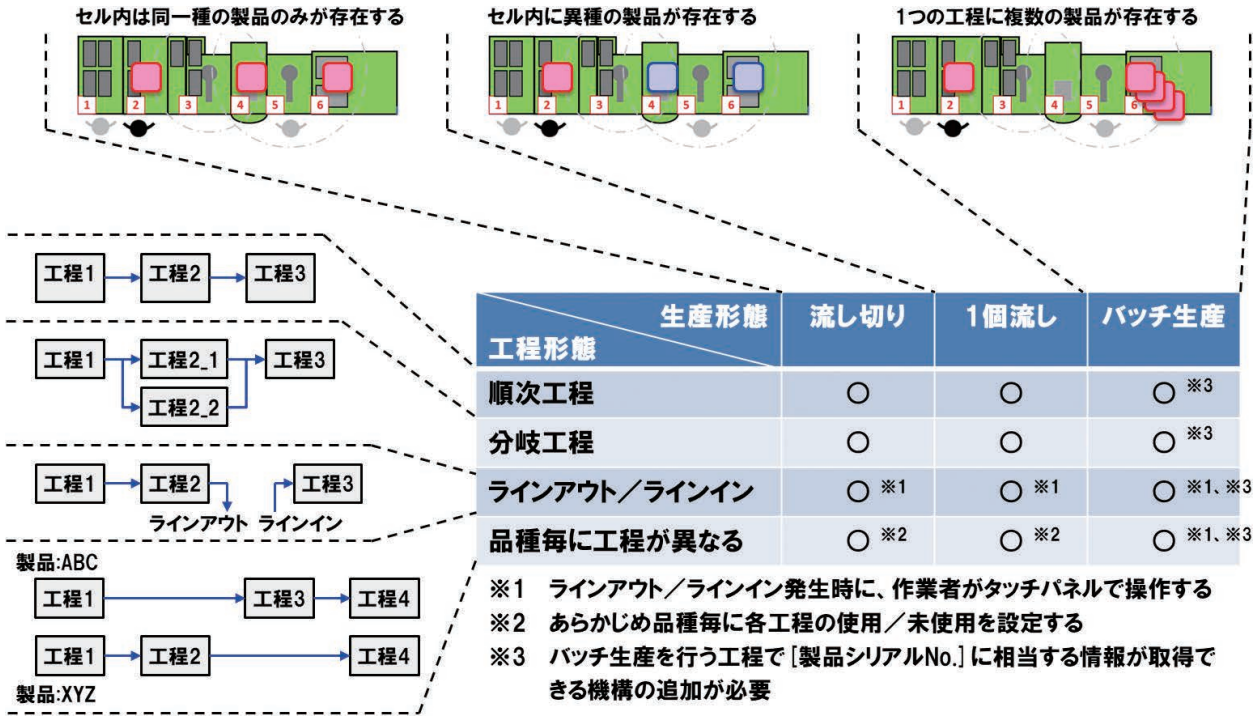


図 14　CLCS における一元管理

イデアの実現性と有効性を実証できた。

今後も、アプリケーションプラットフォームとしての機能強化や、Node による機能拡張だけでなくパイプラインアプリケーション以外のアプリケーションを開発するための開発環境を追加する事で、より高度で高性能なアプリケーションを実現できることを目指したい。

また、TSDB の移植を含めた変数アクセスの高速化に取り組みたいと考えている。

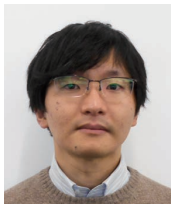
本技術を活用することで、共創を通じたアプリ創造活動、販売先を限定する用途限定商品化と複数顧客への展開、汎用商品化と汎用展開のサイクルの内、アプリ創造活動から用途限定商品化と複数顧客への展開、さらなるアプリ創造活動のサイクルを実現できた。今後は、汎用商品化につながるサイクルを実現したい。

参考文献

1）太田政則，西山佳秀．AI 搭載マシンオートメーションコントローラの開発（2）．OMRON TECHNICS. 2019, vol.51 No.1, p.45-51.
2）総務省，厚生労働省，文部科学省．2018 年版ものづくり白書．経済産業調査会，2018, 325p.
3）オムロン株式会社．“現場データ活用サービス i-BELT”．オムロン制御機器．https://www.fa.omron.co.jp/solution/i-belt/，（参照 2022-01-11）.
4）オムロン株式会社．“トレーサビリティで実現する変種変量生産の品質管理”．オムロン制御機器．https://www.fa.omron.co.jp/solution/proposal/app_006/，（参照 2022-01-11）.
5）Eclipse. Vert.x. https://vertx.io，（参照 2022-01-11）.
6）オムロン株式会社．NJ/NX シリーズ CPU ユニットユーザーズマニュアル ソフトウェア編．2021, 838p.
7）Microsoft Corporation．“Azure Certified Device プログラムの概要”．https://docs.microsoft.com/ja-jp/azure/certification/overview,（参照 2022-01-11）.
8）Amazon.com Inc．“IoT（モノのインターネット）- ユースケース別クラウドソリューション”．https://aws.amazon.com/jp/iot/，（参照 2022-01-11）.
9）Siemens AG. “MindSphere”. https://www.plm.automation.siemens.com/global/ja/products/mindsphere/，（参照 2022-01-11）.
10）The Hermes Standard Initiative. IPC-HERMES-9852，（2019）.

執筆者紹介

西垣 弘二　NISHIGAKI Koji
インダストリアルオートメーションビジネス
カンパニー
技術開発本部 第 1 技術部
専門：情報科学
博士（工学）



荒井 航　ARAI Wataru
インダストリアルオートメーションビジネス
カンパニー
技術開発本部 第 1 技術部
専門：ソフトウェア工学