Supported by 日本財団 THE NIPPON FOUNDATION

2021 年度

配電システムの IoT 化による状態監視及び予防保全アプリケーションの技術開発

成果報告書

2022 年 3 月

一般社団法人　日本舶用工業会

はしがき

本報告書は、BOAT RACE の交付金による日本財団の助成金を受けて、2020 年度に一般社団法人日本舶用工業会が実施した「配電システムの IoT 化による状態監視及び予防保全アプリケーションの技術開発」の成果をとりまとめたものである。

本開発は、2020 年度、2021 年度の２年計画で、ＢＥＭＡＣ株式会社に委託して実施しており、その最初の１年度分の報告書をここにまとめたものである。

ここに、貴重な開発資金を助成いただいた日本財団、並びに関係者の皆様に厚く御礼申し上げる次第である。

2022 年 3 月

(一社)日本舶用工業会

目　　次

第Ⅰ部　2020 年度

第Ⅱ部　2021 年度

1．事業の目的

船舶の安全性を向上させる上で配電系統の状態監視は重要であるが、現在の配電系統の監視では、最低限の警報監視しか実施していない場合が多く、無人運航船の実現を想定した常時監視が行えるシステムの構築は困難である。そこで本事業では、配電系統の発停制御ユニット等の各制御機器でのみ使用されていた情報を収集するなどIoTを活用し、蓄積・解析できる状態監視アプリケーションの開発を行う。

また、配電系統のトラブルは突発的に顕在化するケースがあり、現状の船陸通信の通信間隔、時間でこれを事前に陸上で察知することは難しい。このため、機械学習技術、シミュレーション技術の応用により、陸上で船内の状態を推測する予防保全アプリケーションを開発する。こうした発想は、従来の船舶支援アプリケーションにはない画期的な取組である。

本事業の目的は以上のとおりであるが、個別の事項について補足すると以下のとおりである。

運輸安全委員会事務局の報告によると、平成20年から平成30年の間に発生したインシデントのうち、ブラックアウトによる事故及びインシデントは約5%（49件）を占める。一方、国内保険会社に支払われる海上保険の保険料総額は年間2,500億円であることから、ブラックアウトのリスクに対して保険料のみで年間125億円もの負担が生じていることになる。本事業では、配電系統の情報も含めた状態監視及び予防保全アプリケーションを開発する事により、この内電気的要因で発生した14件の事例、及び、整備不良や点検不足といった人的要因により発生した21件の事例を未然に防ぐ、もしくはブラックアウト発生後の対応遅れを無くす事で、衝突事故や乗揚事故等の重大事故の発生を防ぐ事を目標とする。

この目標を実現するため、機械学習技術を活用する。配電系統による電流・電圧の制御機構の依存関係や挙動を機械学習にてモデル化し、船舶から収集したデータをこのモデルに当てはめることで、当該船舶の電力系統の理論的に整合が取れた状態を表現できる。ただし、実際に船舶から収集するデータは、こうした理論値から若干の誤差が発生する。この誤差が単なるノイズでなく意味があるか又は理論的に整合が取れた状態からどれだけ乖離しているのかを統計学的に評価し、この乖離等の傾向と幅が確率的に稀と認められる場合を異常として検出するものである。ある計測点の異常が認められた場合、配電系統の依存関係のモデル構造を用いてその原因を推論する別の機械学習の手法の応用も検討する。

これらの機械学習アルゴリズムの実装を行うには、一般的にはトラブルや異常が発生した前後の情報を使って、モデル等のパラメータを学習させる場合が一般的である。しかし、船舶の船陸通信の頻度は多い場合でも分単位の通信間隔であるため、極めて短時間の間にトラブルに至る配電系統の場合、分単位で取得されるような通常の電流・電圧のモニタリングデータだけでは、十分な学習用データが得られないと想定している。そこで、従来の配電系統の監視のように電流・電圧の情報だけを収集するトップダウン的なアプローチだけでなく、上述の通り電気系統の機器構成をモデル化し、それぞれの機器及び部品の依存関係から、電

気系統のどの部分の故障・トラブルのリスクが高まっているのか、というトラブル原因のボトムアップ的なモニタリングによる異常検知を試みる。そのため本事業では、現状船員が行っている配電系統の監視において、あまり重視されていなかった各制御機器でのみ使用されていたわずかな手がかりとなる情報（系統電力、周波数、ACB(Air Circuit-Breaker:空気遮断器)開閉状態、各モーターの始動信号等）も活用し、より精緻な電気系統の依存関係のモデル化を行う。

2．事業の目標

2.1　本事業の最終目標

1) 予防保全アプリケーションを設計・開発し、当該アプリケーションで検知しようとする電気関連トラブルに発展しうる事象の予兆のうち、50%以上の予兆を、収集したデータから検知できるようにする。

2.2　2020 年度の目標

1) 船上サーバー標準規格（ISO19847/ISO19848）に準拠した船上、陸上双方のデータ集積設備を用いた効率的なデータ収集・利用システムの開発により、船上データの利用を行う際のシステム構築に要する作業工数を 30%以上削減する。
2) 状態監視アプリケーションの開発により、配電系統を含めた、船内電気関連トラブル発生時のメーカアフターサービス員の補修作業工数を 30%以上削減できるスキームを確立する。

3．2020 年度の実施内容

3.1　配電システムの制御データの調査

3.1.1　配電システムの現状調査

配電システムの IoT 化による状態監視アプリケーションの開発に向け、まず現状把握として BEMAC 製配電システムにて自動化制御を行っている発電機コントローラシステムをターゲットとし現状調査を実施した。

1) 発電機コントローラシステムの概要

BEMAC 製発電機コントローラシステムは図 1 に示すように、各発電機盤に自動化ユニット、発停制御ユニットが備えられている。また、外部システムとのデータ送受信の為に通信ゲートウェイユニットが搭載されている。

自動化ユニットでは、発電機の同期投入、負荷分担、定周波数等の制御、母線電圧、周波数監視などを実施している。発停制御ユニットでは発電機の発停、解列及び台数制御、優先遮断や異常発生時の切り替え制御等を実施している。通信ゲートウェイユニットでは、現状では主に BEMAC 製データロガーシステムへ警報データを送信するために使用されている。

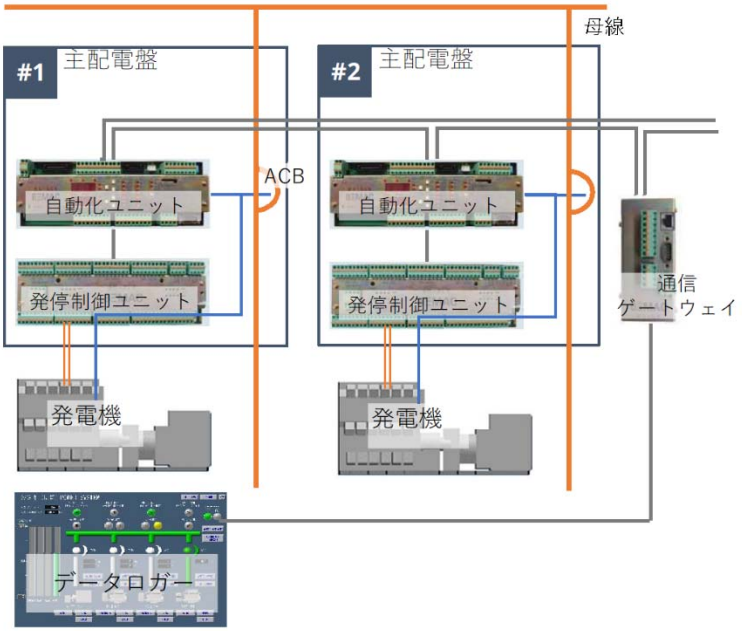


図 1 発電機コントローラシステム

2) 発電機コントローラシステムの出力データ仕様

発電機コントローラシステムでは先に述べたように、出力データは通信ゲートウェイユニットを介して主にデータロガーシステムに出力されている。通信データはあらかじめ発電機コントローラシステムとデータロガー間で標準データが取り決められており、それ以外のデータを取得するためには都度、データ内容の取り合いを決定し、個別に対応する必要がある。標準データとしては、自己診断情報及びアナログ値情報、システム共通状態情報、発電機個別の状態情報等システム規模に応じて多少増減があるが警報監視に必要最低限な 50 点程度である。以下表 1、表 2 に代表的な標準データを記す。

表 1 発電機コントローラシステム標準出力データ例(1)

項目	データ名称
自己診断情報	NO.X 自動化ユニット動作状態
	NO.X 発停制御ユニット動作状態
	システム通信 A ライン異常
	システム通信 B ライン異常
アナログ値情報	NO.X 発電機電圧
	NO.X 発電機周波数
	NO.X 発電機負荷
	母線電圧
	母線周波数

（x=発電機号機）

表 2 発電機コントローラシステム標準出力データ例(2)

項目	データ名称
システム 共通状態	優先遮断発生
	母線電圧異常警報
	母線周波数異常警報
	絶縁低下
各発電機状態	NO.X 過速度危急停止警報
	NO.X 潤滑油低圧危急停止警報
	NO.X 冷却清水高温危急停止警報
	NO.X 起動失敗警報
	NO.X ACB 異常トリップ警報
	NO.X ACB 投入不能警報
	NO.X 過電流警報
	NO.X 速度継電器異常警報
	NO.X 運転信号

（x=発電機号機）

3) 発電機コントローラシステムの内部データ仕様

3-1) 自動化ユニットのデータ仕様

自動化ユニットについては、母線監視、同期投入、負荷分担制御の為に、以下表 3 のようなデータを内部に持っている。

表 3 自動化ユニットの内部データ

項目	データ名称
アナログデータ	母線電圧
	母線周波数
	発電機電圧
	発電機周波数
	発電機電流
	発電機負荷
	負荷力率
自己診断	発電機コントローラシステム通信異常
	発停制御ユニット通信状態
制御信号	ガバナ制御信号（上げ、下げ）
	同期投入可能電圧差、周波数差
カウントデータ	D/G 運転時間及び起動回数
	ACB 閉時間及び投入回数
	ユニット動作時間

表 3 のアナログデータと自己診断情報に関しては、多くの項目で標準でのデータ出力に対応しているが、制御信号、カウントデータなどはデータ出力が考慮されていないため出力が難しい状態となっている。

また、本ユニットには異常発生時に状態イベントを記録できるデータロギング機能が搭載されている。通常設定では ACB 異常トリップ警報または母線異常警報が発生した場合に本機能が動作する。記録されるデータは以下のとおりである。

・発生前 50 秒、発生後 10 秒間の 100ms ごとの以下の計測値データ
　母線電圧、母線周波数、発電機電圧、発電機周波数、発電機電流、発電機負荷
・発生時前後 25 点の自動化ユニット及び発停制御ユニットの入出力情報

本データロギング機能を使用することにより異常発生時の詳細な制御、計測値を参照することが出来る。しかしながら本状態イベントデータは現状では外部への標準出力に対応しておらず、本船上ではユニットに搭載されている 7 セグメント表示器に表示させることによる目視確認しかできない状態となっている。イベントデータの確認を実施するためには、必要な部分だけ間引いて確認するとしても数百点の確認が必要になり大きな労力と時間が掛かっている状態である。

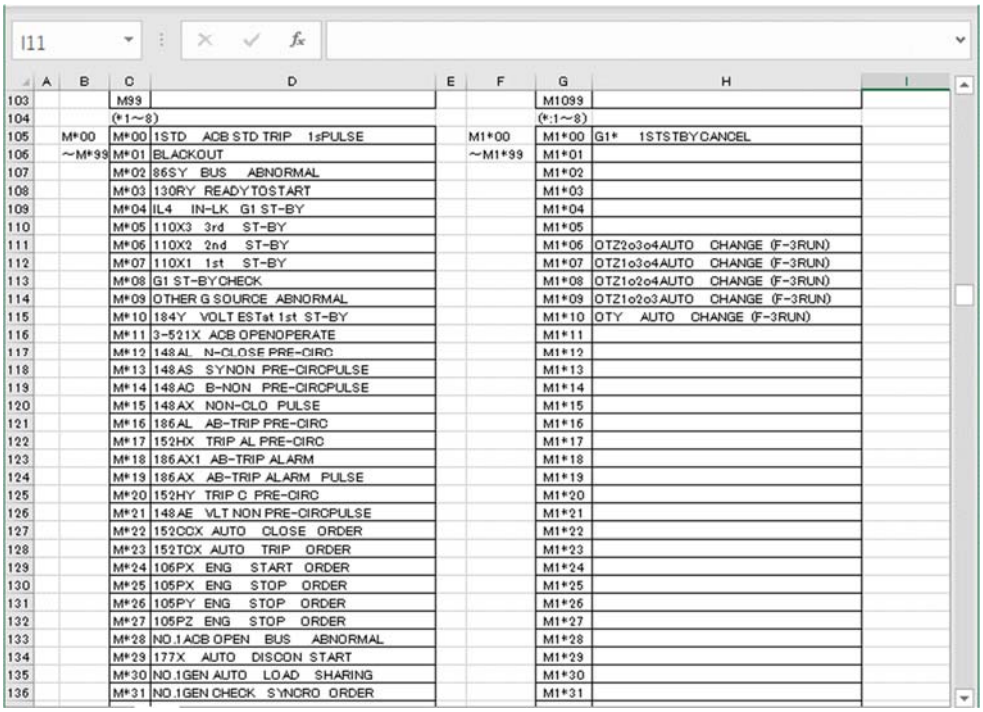
3-2) 発停制御ユニットのデータ仕様

発停制御ユニットは、主機能としてプログラマブルロジックコントローラとして動作する。入力リレー32 点、出力リレー32 点、内部リレー8192 点、内部ワードメモリ 2000 点などを持っている。このリレーの範囲内でラダー言語を用いて自由にソフトウェアを構築可能となっている。上記の 10000 点強のデータ領域中、約 4000 点は他発電機との制御データの共有化に用いられている。また、ソフトウェアは発電機メーカー、制御仕様などにより大まかには共通化されている。標準的なシステムでは上記の 10000 点中、3500 点程度をソフトウェア用制御データとして使用している。但し上記は発電機 8 台という本システムでの最大スペック時を考慮した場合での使用点数である。実際は発電機 1 台当たり 400 点程度の制御データとなっている。そのうち 170 点程度は他発電機と共有されておらず、内部データとしてのみ使用されている。

前項で調査した結果、標準出力データは発電機 1 台当たり 20 点弱であったが、実際は内部に 400 点程度の大量のデータが存在していることが調査の結果判明した。
図 2 に割り付け例の一部を示す。

3-3) 通信ゲートウェイユニットのデータ仕様

通信ゲートウェイユニットは、発電機コントローラシステム間で負荷分担などの制御に必要なデータを共有するネットワークに接続されている。このネットワーク内のデータを解析し、外部出力用フォーマットに変換して送信する機能を有している。本ネットワーク上には制御に必要なデータなど 2400 点弱（300 点×最大発電機 8 台）のデータが約 1 秒周期で共有されている。この 2400 点中必要最低限の 50 点程度が外部へ標準データとして送信されている。



	A	B	C	D	E	F	G	H	I
103			M99				M1099		
104			(*1～8)				(*:1～8)		
105		M*00	M*00	1STD ACB STD TRIP 1sPULSE		M1*00	M1*00	G1* 1STSTBYCANCEL	
106		～M*99	M*01	BLACKOUT		～M1*99	M1*01		
107			M*02	86SY BUS ABNORMAL			M1*02		
108			M*03	130RY READYTOSTART			M1*03		
109			M*04	IL4 IN-LK G1 ST-BY			M1*04		
110			M*05	110X3 3rd ST-BY			M1*05		
111			M*06	110X2 2nd ST-BY			M1*06	OTZ2o3o4AUTO CHANGE (F-3RUN)	
112			M*07	110X1 1st ST-BY			M1*07	OTZ1o3o4AUTO CHANGE (F-3RUN)	
113			M*08	G1 ST-BYCHECK			M1*08	OTZ1o2o4AUTO CHANGE (F-3RUN)	
114			M*09	OTHER G SOURCE ABNORMAL			M1*09	OTZ1o2o3AUTO CHANGE (F-3RUN)	
115			M*10	184Y VOLT ESTst 1st ST-BY			M1*10	OTY AUTO CHANGE (F-3RUN)	
116			M*11	3-521X ACB OPENOPERATE			M1*11		
117			M*12	148AL N-CLOSE PRE-CIRC			M1*12		
118			M*13	148AS SYNON PRE-CIRCPULSE			M1*13		
119			M*14	148AC B-NON PRE-CIRCPULSE			M1*14		
120			M*15	148AX NON-CLO PULSE			M1*15		
121			M*16	186AL AB-TRIP PRE-CIRC			M1*16		
122			M*17	152HX TRIP AL PRE-CIRC			M1*17		
123			M*18	186AX1 AB-TRIP ALARM			M1*18		
124			M*19	186AX AB-TRIP ALARM PULSE			M1*19		
125			M*20	152HY TRIP C PRE-CIRC			M1*20		
126			M*21	148AE VLT NON PRE-CIRCPULSE			M1*21		
127			M*22	152CCX AUTO CLOSE ORDER			M1*22		
128			M*23	152TCX AUTO TRIP ORDER			M1*23		
129			M*24	106PX ENG START ORDER			M1*24		
130			M*25	105PX ENG STOP ORDER			M1*25		
131			M*26	105PY ENG STOP ORDER			M1*26		
132			M*27	105PZ ENG STOP ORDER			M1*27		
133			M*28	NO.1ACB OPEN BUS ABNORMAL			M1*28		
134			M*29	177X AUTO DISCON START			M1*29		
135			M*30	NO.1GEN AUTO LOAD SHARING			M1*30		
136			M*31	NO.1GEN CHECK SYNCRO ORDER			M1*31		

図 2 発停制御ユニット内部リレー割り付け

3-4) 各ユニット間のデータの流れ

各ユニットの内部データの調査を実施した結果から、次にこのデータが各ユニット間でどのようにやり取りされているかを調査した。概略を図 3 に示す。本図は仕様の最大値を記載しており、実仕様での使用点数はこれより少ない。

発停制御ユニットの内部データ 10000 点はおおよそ 4 割の 4000 点が自動化ユニットとやり取りされている。半数はユニット内でのみ使用され他で参照することはできない。

自動化ユニットの内部データと発停制御ユニットから送られてきたデータは他の号機のユニットとの共用の為、発電機毎に自動化ユニットのデータ約 70 点、発停制御ユニットのデータ約 230 点の約 300 点がネットワークで送受信されている。送受信データの一部は上述の 4000 点の一部を使用し、発停制御ユニット間でのデータ共有を実現している。

通信ゲートウェイユニットではネットワーク上の 2400 点弱のデータの内 50 点程度を外部出力している。

上記のように、各ユニットの内部データは必要なものは通信によって他機器との共有が図られているが、製品化当時は機器スペックの都合等で不要と判断され現状では外部出力が不可能なデータが多数存在している。

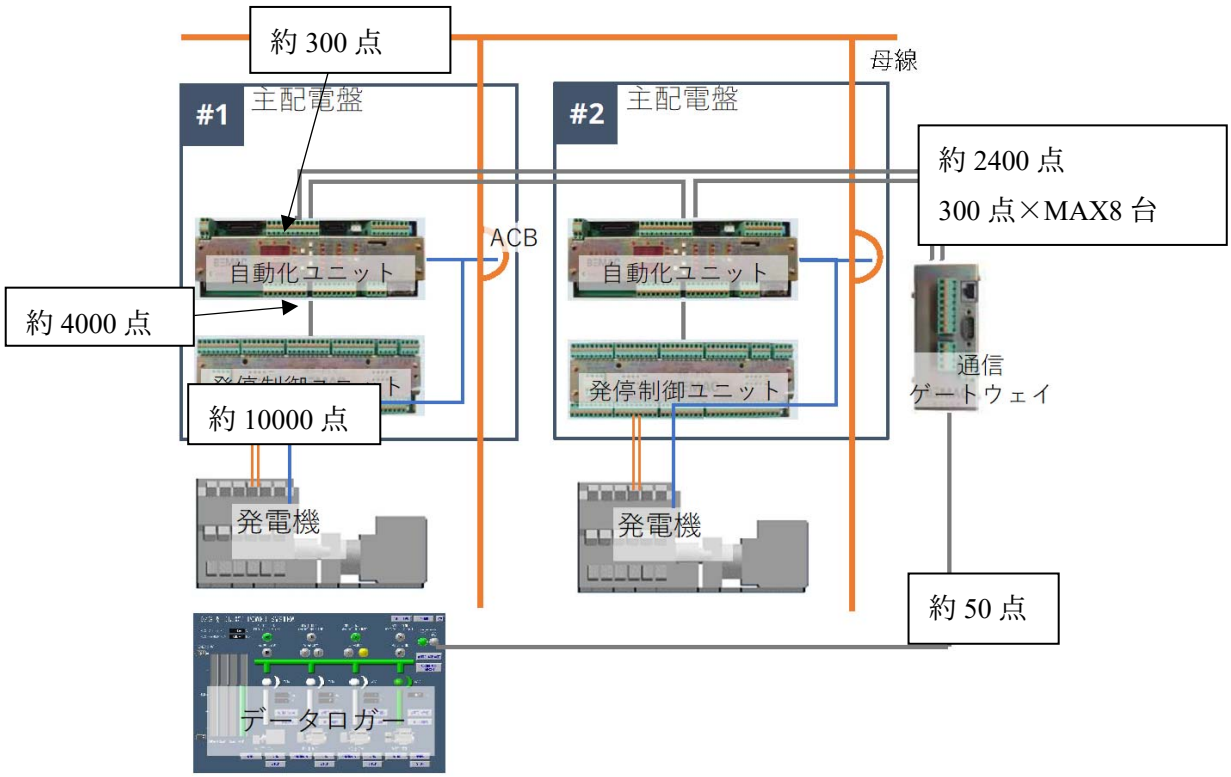


図 3 発電機コントローラシステム間のデータやり取り

3.1.2 配電システムの IoT 化に必要なデータ項目のリストアップ

前項で調査した現状を踏まえ、必要データのリストアップを実施した。今後の機能拡張も見据え現状不要と考えられるデータも含めて可能な限り存在するデータの多くをデータベース化することとした。以下詳細を報告する。

1) 自動化ユニットの必要データのリストアップ

3.1.1 の 3-1）項のデータ調査から、内部データとしては表 3 の約 20 種類、複数存在するものを含めて合計 30 点程度のデータ、およびデータロギング機能による電圧、電流、周波数、電力の各計測値（100ms 分解能 600 点）のデータおよび 25 点の入出力情報データが存在していることが判明した。

この中から必要なデータのピックアップを実施した。母線電圧、周波数、発電機負荷などのアナログデータは IoT 化には必須と考えられる。また、瞬時的な動作を解析するためには各種制御に使用されている制御信号の入手も不可欠である。カウントデータについては代替手段もあり、重要ではないと考えられるが、長年の傾向を判断するためには必要となる。また、異常発生時の状態イベントを記録できるデータロギング機能は状態監視に必要不可欠である。現状データとして出力する手段が乏しいので標準的にデータ出力可能とすることを必須としてシステムを検討する。

結果、前項のデータ調査で得られた結果のほぼすべてのデータが配電システムの IoT 化には必要となるため、現状出力に対応していないデータの出力を実現することが必要であると判断した。

2) 発停制御ユニットの必要データのリストアップ

3.1.1 の 3-2）項のデータ調査から、必要なデータのピックアップを実施した。存在しうるデータはユニットの仕様上 10000 点以上存在するが、標準的なシステムでは発電機 1 台当たり約 400 点、最大 3200 点程度を使用する仕様となっている。この 400 点のデータの中から必要データをリストアップすることを検討したが、

・既存配電システムでは仕様により内部のデータ構成が異なっている。

・今後の配電システムの機能拡張により使用領域の変更の可能性が高い。

・現状不要と判断しても、のちに必要となる可能性がある。

等の理由により、現在の時点ではリストアップは実施せずに、400 点の使用領域だけではなく仕様上の最大点数である 10000 点を出力可能とした。次項で述べるデータの紐づけにより、配電システムに応じた必要データのピックアップ及びスペックアップに対応できるようにする。

3) 通信ゲートウェイユニットの必要データのリストアップ

3.1.1 の 3-3）項のデータ調査から、本ユニットのデータはほぼすべてが上記の自動化ユニット、発停制御ユニットの共有用データであることが判明した。よって本ユニットにしか存在しないデータは皆無であり、本ユニットからのリストアップの必要性はなくなり、データ出力も不要と判断した。

3.1.3 船上サーバー標準規格への必要データの紐づけ

今回のターゲットである発電機コントローラシステムについてはデータロガーとの通信データ内容に変更があった場合、先に述べたように個別対応が必要な状況となっている。このようなやり取りのデータを標準化させ、機器やシステムの連接利便性を高めるために ISO19847/19848 が日本舶用工業会スマートナビゲーションシステム研究会の活動により規格化されている。本開発では、船上サーバーは ISO19847/19848 に対応した BEMAC 製 IoT データサーバーを使用する。

これまでの調査で必要データの抽出は完了し、次にデータを容易に各アプリケーションで使用するために船上サーバー標準規格への紐づけについて検討を実施した。

1) ISO19847（実海域データ共有化のための船内データサーバー）

ISO19847 では、航海系、機関系、その他の系統の実海域データについて、時間軸をそろえた形で共有化するために設けられるデータサーバーの諸要件が定義されている。主な規格内容として、

・船上機器やシステム及びセンサーから自動的に入力されるデータのほか，手動で入力されるデータに対する要件

・データの入力プロトコル要件

・データサーバーの処理機能及び性能要件

・データサーバーからの出力プロトコル要件
・データサーバーから出力されるデータの要件
等が定められている。規格の詳細は本報告書では割愛する。

2) ISO19848（船舶機関及び装置のデータ標準）

船内に搭載されている機器間、システム間などでは通信フォーマットとしてはイーサネットや Modbus 等ある程度標準化されたものが存在するが、その中でやり取りされるデータについては標準化されたフォーマットがなく、相互に通信で接続される場合は個別に対応することが避けられない状態である。

本規格ではデータの標準化のために、名前の付け方の要件を規定している。名前は図 4 に記されている通り、対機械、対人との容易なデータ交換を可能にするために URL（Unique Resource Locator)に準拠した名称になっている。名称の ID 内部構造として Ship ID、 Local ID、Short ID、Local Data Name 等様々な項目があるが詳細は割愛する。

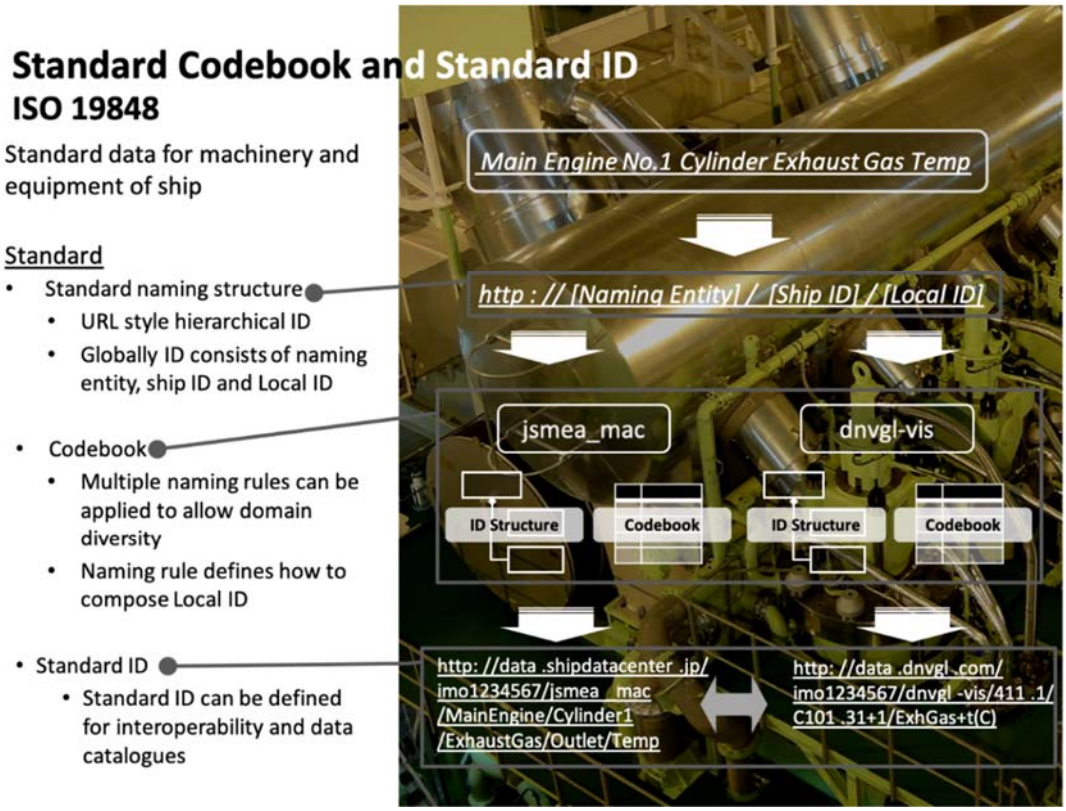


図 4 ISO19848 の名前の付け方

出典：日本舶用工業会 SSAP ホームページ https://www.jsmea.or.jp/ssap/jp/

また、本名称 ID を実現するデータフォーマットとしては Time Series Data と Data Channel List という論理構造が定義されている。この構造を記述する言語として ISO で規定されている一般的な XML 言語などが採用されている。

3) 配電システムのデータ紐づけ

データ項目名については、出力データすべてに ISO19848 に準拠した名称を付与することは未使用領域やシーケンス内部データなど名称付けが難しいものがあり全点を標準化させることは現実的ではなかった。まずは、通常のデータ分析に供せられる部分、今回開発した状態監視アプリケーションに必要な部分について重点的に紐づけ設計を行った。他の計測点名称があるデータについては Local ID 等厳密には準拠しておらず、仮で名称付けを行っている状態となった。今後効率的な紐づけツールの開発を実施し対応データを増やしていく予定である。

まずは、データ名称に対する ISO19848 の Local ID の紐づけを実施した。

表 4 に今回紐づけを実施したデータ名称の一部を記載する。

表 4 配電データの船内データ標準規格への紐づけ

データ名称	ISO19848 Local ID
#1 発電機ターニングスイッチ	jsmea_mac/DieselGeneratorSet/Generator1/TurnBarSwitch///Position/Status
#1 発電機リモートスイッチ	jsmea_mac/DieselGeneratorSet/Generator1/Remote///Position/Status
#1 発電機スピードリレー低速	jsmea_mac/DieselGeneratorSet/Generator1/SpeedRelay///ON/Status
#1 発電機ハンドルスイッチ	jsmea_mac/DieselGeneratorSet/Generator1/HandleSwitch///Position/Status
#1 発電機 ACB 開	jsmea_mac/DieselGeneratorSet/Generator1/ACB///OpenOutput/Status
#1 発電機始動弁	jsmea_mac/DieselGeneratorSet/Generator1/StartValve///ON/Status
母線電圧	jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusVoltage//Voltage//Inst
母線周波数	jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusFrequency//Frequency//Inst
#1 発電機電圧	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Voltage//Inst
#1 発電機周波数	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Frequency//Inst
#1 発電機電流	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Current//Inst
#1 発電機電力	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Power//Inst
#1 発電機 ACB 閉	jsmea_mac/DieselGeneratorSet/Generator1/ACB///Close/Status
#1 発電機ガバナ上げ	jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Raise/Status
#1 発電機ガバナ下げ	jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Low/Status
#1 発電機同期投入電圧差	jsmea_mac/DieselGeneratorSet/Generator1/Electric//VoltageDifference//Status
#1 発電機同期投入周波数差	jsmea_mac/DieselGeneratorSet/Generator1/Electric//FrequencyDifference//Status
#1 発電機 ACB 閉制御出力	jsmea_mac/DieselGeneratorSet/Generator1/ACB///CloseOutput/Status

次にこの名称で実際にデータサーバーに登録するために Data Channel List の作成を行った。以下図 5、6 に実際の XML 構造の一部を記す。

```
            <LocalID>jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusVoltage//Voltage//Inst</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>Decimal</Type>
            </Format>
          - <Unit>
              <UnitSymbol>V</UnitSymbol>
            </Unit>
            <Name>NO.1 BUS VOLTAGE</Name>
          </Property>
      </DataChannel>
    - <DataChannel>
        - <DataChannelID>
            <LocalID>jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusFrequency//Frequency//Inst</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>Decimal</Type>
            </Format>
          - <Unit>
              <UnitSymbol>Hz</UnitSymbol>
            </Unit>
            <Name>NO.1 BUS FREQUENCY</Name>
          </Property>
      </DataChannel>
    - <DataChannel>
        - <DataChannelID>
            <LocalID>jsmea_mac/DieselGeneratorSet/Generator1/Electric//Voltage//Inst</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>Decimal</Type>
            </Format>
          - <Unit>
              <UnitSymbol>V</UnitSymbol>
            </Unit>
            <Name>NO.1 GENERATOR VOLTAGE</Name>
          </Property>
      </DataChannel>
    - <DataChannel>
```


図 5 Data Channel List 構造(1)


```
    - <DataChannel>
        - <DataChannelID>
            <LocalID>jsmea_mac/DieselGeneratorSet/Generator1/ACB///Close/Status</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>String</Type>
            - <Restriction>
                <Enumeration>On</Enumeration>
                <Enumeration>Off</Enumeration>
              </Restriction>
            </Format>
            <Name>NO.1 GENERATOR ACB CLOSE</Name>
          </Property>
      </DataChannel>
    - <DataChannel>
        - <DataChannelID>
            <LocalID>jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Raise/Status</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>String</Type>
            - <Restriction>
                <Enumeration>Raise</Enumeration>
                <Enumeration>0</Enumeration>
              </Restriction>
            </Format>
            <Name>NO.1 GENERATOR GOVERNOR RAISE</Name>
          </Property>
      </DataChannel>
    - <DataChannel>
        - <DataChannelID>
            <LocalID>jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Low/Status</LocalID>
          </DataChannelID>
        - <Property>
          - <DataChannelType>
              <Type>Inst</Type>
              <UpdateCycle>1</UpdateCycle>
            </DataChannelType>
          - <Format>
              <Type>String</Type>
            - <Restriction>
                <Enumeration>Low</Enumeration>
                <Enumeration>0</Enumeration>
              </Restriction>
            </Format>
```


図 6 Data Channel List 構造(2)

3.2　データ出力ソフトウェアの設計

3.2.1　データ出力システム構成

3.1 項において既存の発電機コントローラシステムの各ユニットのデータ構成及びネットワーク構成が明らかとなった。初期検討の段階では通信ゲートウェイユニットの改良によりデータ出力を実施する予定であったが、

・現仕様では最大でも約 2400 点しか出力できず決定仕様を満足できない

・自動化ユニットから通信ゲートウェイへの通信方式の改良により出力点数を満足しても出力周期が長くなってしまい、データを確実に取得できる保証がなくなる

・データロギング機能のデータ出力の為にはシステムの基本設計の変更が必要

等の懸念点が発生したため、新たな方式の検討を実施した。

検討の結果、発電機コントローラシステムと IoT サーバー間に新たに収集ユニットを設置することとした。本項ではソフトウェア以外のシステム構成についても述べる。

大きく分けて、

・収集ユニットでの発電機コントローラシステムからのデータ収集機能

・収集ユニットから IoT データサーバーへのデータ出力機能

・IoT サーバー内でのデータベース化・解析・抽出機能

の 3 の機能を組み合わせることにより出力点数や周期の仕様を満足したシステムを構築することが可能となった。システムの概要構成を図 7 に記す。次項にて詳細を報告する。

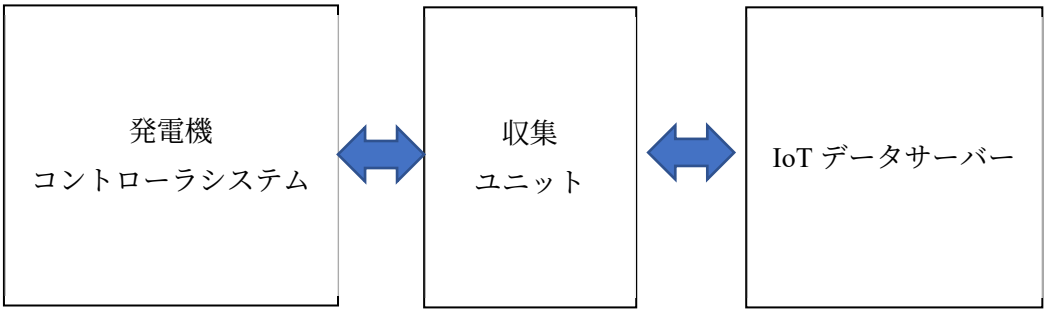


図 7　データ出力システム構成

3.2.2　収集ユニットでの発電機コントローラシステムからのデータ収集機能

1) 発電機コントローラシステムからのデータ出力インターフェイス

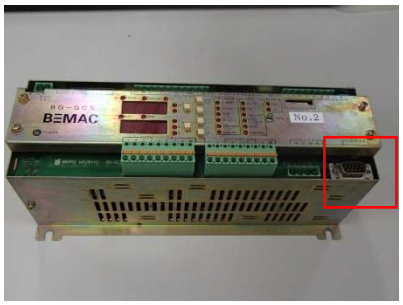
発電機コントローラシステムからのデータ出力は、各ユニットに未使用の通信ポートが存在した為、このポートを用いてデータ出力を実施することとした。以下にポート仕様を記すとともに図 8 に各ユニットの使用ポートを図示する。

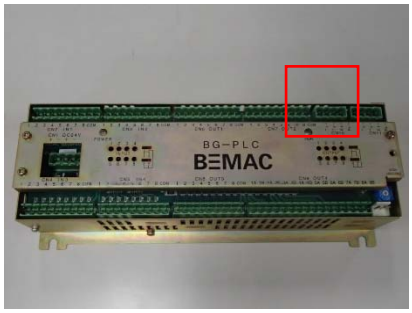
・通信インターフェイス　RS-422 または 2 線式 RS-485

・通信速度　　最大 115.2kbps （変更可能）

・通信コネクタ　　D-sub9 ピンコネクタ（自動化ユニット）

4 極 2 ピースコネクタ（発停制御ユニット）





自動化ユニット　　発停制御ユニット

図 8　ユニット通信ポート

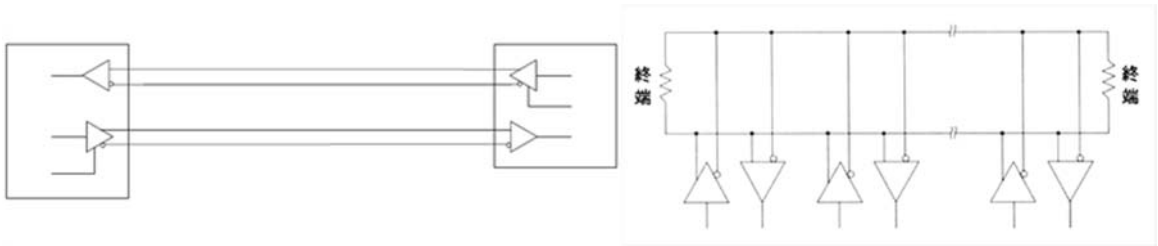
次に通信方式の検討を実施した。使用するポートが対応している RS-422 と 2 線式 RS-485 で比較を実施し、RS-422 方式を選定した。

以下表 5 に通信方式の比較表を示す。

表 5　通信方式比較

項目	RS-422	2 線式 RS-485
ライン構成	差動	差動
送信（ドライバ数）	1	最大 32
受信（レシーバ数）	最大 10	最大 32
通信方式	全二重通信	半二重通信
最大ケーブル長	1200m	1200m
通信接続構成	ポイント･ツゥ・ポイント 1:1	マルチドロップ N:M

大きな違いとして、図 9 に示すように RS-422 は基本的には１対１の通信しか対応していないが、RS-485 ではマルチドロップの通信接続が可能となっており、一系統の通信線で最大３２台までの相互通信が可能となる。



RS-422　　RS-485

図 9　通信接続構成

省配線に適したRS-485をまず検討したが、各発電機のユニットに対して収集ユニットから順にデータを取得した場合、通信速度とデータ数の関係から仕様として決定した1秒以内のサンプリング周期でのデータ収集ができない恐れがあることが判明した。よって、1：1通信であるRS-422を採用することに決定した。

2) 発電機コントローラシステムからのデータ出力プロトコル

接続インターフェイスの次にデータ出力を実施するプロトコルを選定した。プロトコルとは、複数の主体が滞りなく信号やデータ、情報を相互に伝送できるよう、あらかじめ決められた約束事や手順の集合のことである。

プロトコルには産業界で実績がありデファクトスタンダードともいえるModbus通信を採用した。Modbusの仕様は広く公開されており利用についても無料である。また実装が比較的容易な特徴がある。Modbusプロトコル通信方式のシリアル伝送モードにはASCII（American Standard Code for Information Interchange）モードとRTU（Remote Terminal Unit）モード の2種類がある。ASCIIモードは、1バイト（8ビット）データを2文字のASCIIコードに変換して伝送する。一方、RTUモードは、1バイト（8ビット）データをそのまま伝送する。データ出力には、ASCIIモードより伝送効率が良いRTUモードを採用した。

Modbusプロトコルはマスタ/スレーブ方式となっている。本システムでは収集ユニットをマスタ、発電機コントローラシステムをスレーブとし、収集ユニットからのリクエストで発電機コントローラシステムからデータを出力する方式とした。

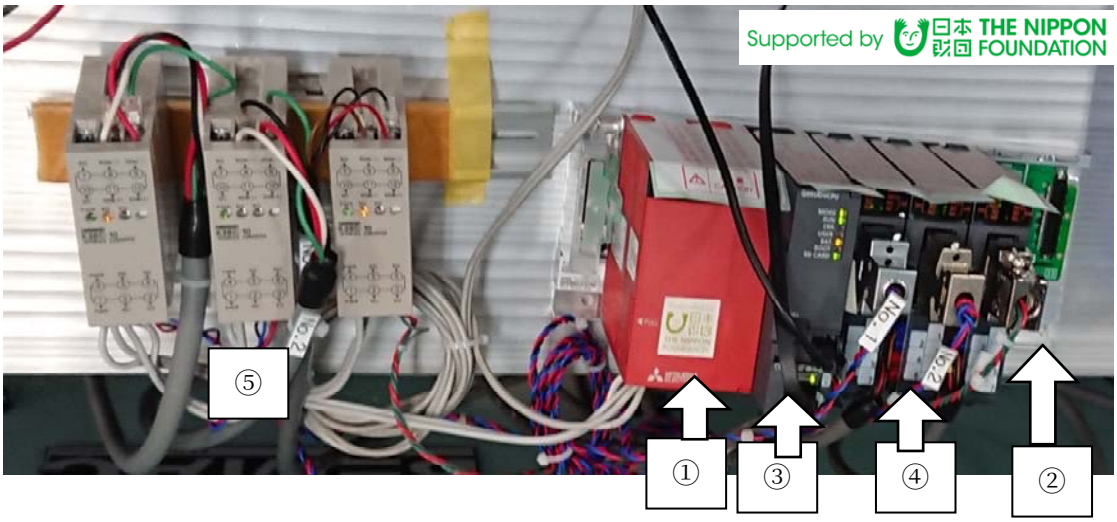
3) 収集ユニットのデータ収集構成

前項の発電機コントローラシステムからのデータ出力に応じた収集ユニットの構成を検討して以下のように決定した。

・通信ポート数　　RS-422最大8ポート
・通信プロトコル　　Modbus/RTU

各発電機に対して自動化ユニットとの通信で1ポート、発停制御ユニットとの通信で1ポートの計2ポート必要である。システムでの発電機台数上限を4台とし8ポートとした。また、発電機コントローラシステムとのシームレスな通信を実施するためにModbusプロトコルへの対応は必須である。

本要求を満足するために収集ユニットハードウェアの新規開発設計も検討したが、PLCを用いたユニット構成とすることとした。PLCとしては三菱電機製のシーケンサMELSEC　Qシリーズを採用した。本シリーズにはModbus通信専用の通信ユニットもシリーズ化されており、自動交信機能などが搭載されている。大きなソフトウェア開発が不要でModbus通信の実装を行うことが出来るため今回検討したシステムとの親和性が非常に高い。以下に使用する主要ハードウェア一覧及び、システム構成図を以下、図10、11及び表6に記す。



	名称	型名	メーカー	備考
①	電源ユニット	Q63P	三菱電機	入力 DC24V 出力 DC5V
②	基本ベース	Q35B		5 スロット
③	CPU ユニット	Q06UDVCPU		SD カード併用
④	MODBUS ユニット	QJ71MB91		発電機台数分使用
⑤	通信変換機	K3SC-10	オムロン	RS232⇔RS422 変換

※　MODBUS ユニットのインターフェイスの仕様上通信変換機が必要となった。

図 10　収集ユニット主要ハードウェア一覧

表 6 Modbus 通信ユニット仕様

伝送仕様	インターフェイス	RS-232C 1 チャンネル RS-422/485 1 チャンネル
	伝送速度	300,600,1200,2400,4800,9600,14400 19200,28800,38400,57600,115200 上記より選択
マスタ機能	自動交信機能	1 チャンネルあたり 32 台 交信可 送信 7 ファンクション
	専用命令交信	1 チャンネル当たり 1 命令 送信 28 ファンクション
スレーブ機能	自動応答機能	受信 17 ファンクション
	同時受付可能要求 電文数	1 チャンネルあたり 1 要求
	局番	1～247

出典：三菱電機ホームページ MODBUS®インタフェースユニット QJ71MB91 より抜粋

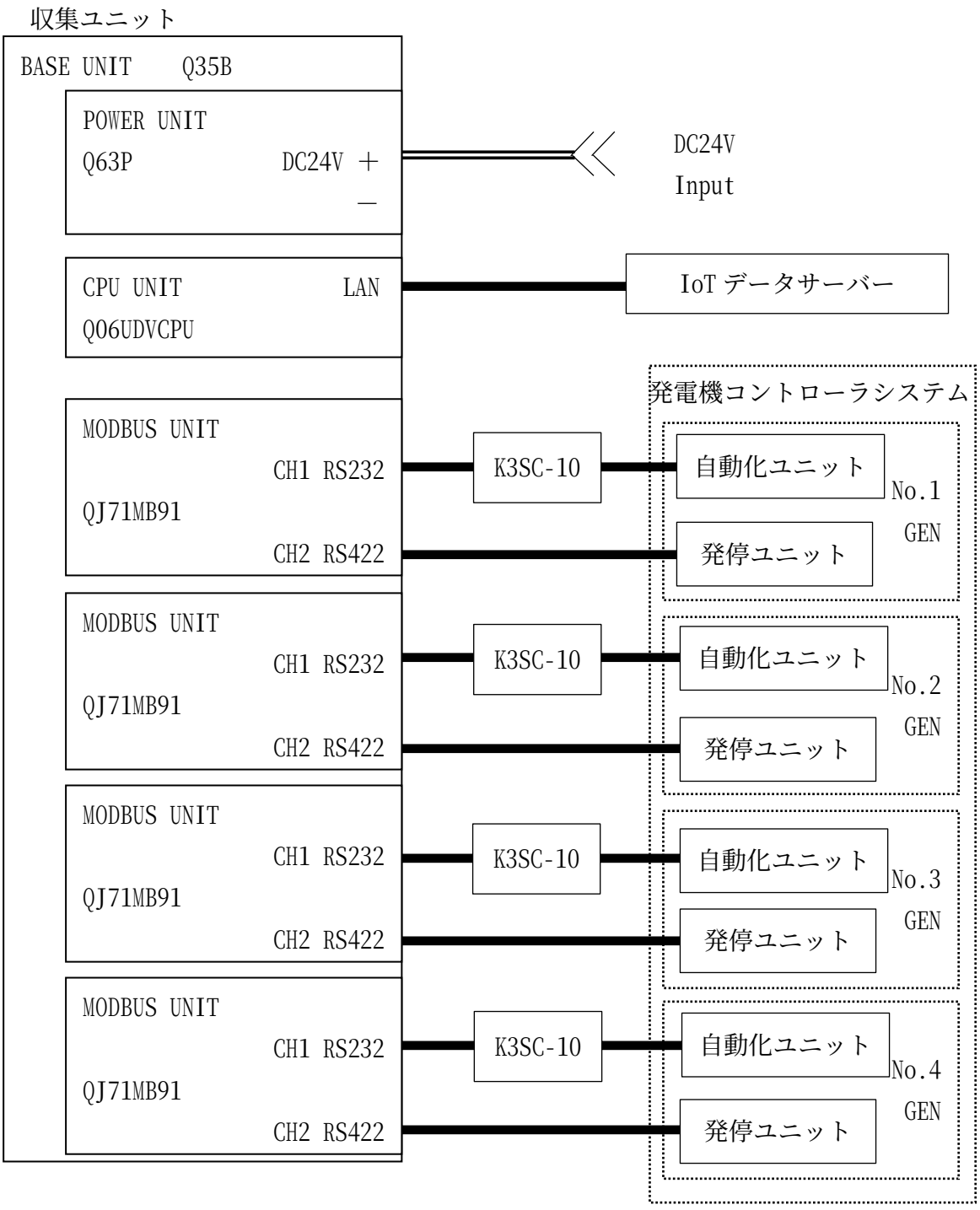


図 11　配電データ収集システム構成

3.2.3　収集ユニットからIoTデータサーバーへのデータ出力機能

収集ユニットからIoTデータサーバーは船内LAN経由での出力となる場合がほとんどであり、必然的にETHERNET接続となる。接続は収集ユニットのCPUユニットのLANインターフェイスを使用する。ISO19847/19848準拠のデータをHTTP通信インターフェイスで出力することも検討したが、PLC内部でソフトウェア処理するには限界があり、プロトコルとしてUDP通信での独自プロトコル接続を採用し、ISO19847/19848準拠のデータへの対応はIoTデータサーバーのリソースを使用して実施することとした。UDP通信は通信負荷が軽く高速なためリアルタイム性に優れている。データ送信の信頼性は低いが、1秒定周期で収集データを出力する方式としては最適となっている。（表7参照）

データロギング機能のデータは異常発生時のみのデータであり、周期的なデータではないため、別途収集ユニット内でCSVファイルとして保存しておきFTPプロトコルでIoTデータサーバー側のソフトウェアが任意のタイミングで取得するようにした。

表7　TCPとUDPプロトコルの違い

プロトコル	TCP	UDP
通信方式	コネクション型	コネクションレス型
信頼性	高い	低い
転送速度	低速	高速
上位プロトコル	HTTP、Telnet、FTP、POP等	DNS、NTP、DHCP、SNMP等
主な用途	Web、メール、ファイル転送	音声通話、動画ストリーミング、少量のデータ転送

3.2.4　IoTデータサーバー内でのデータベース化・解析・抽出機能

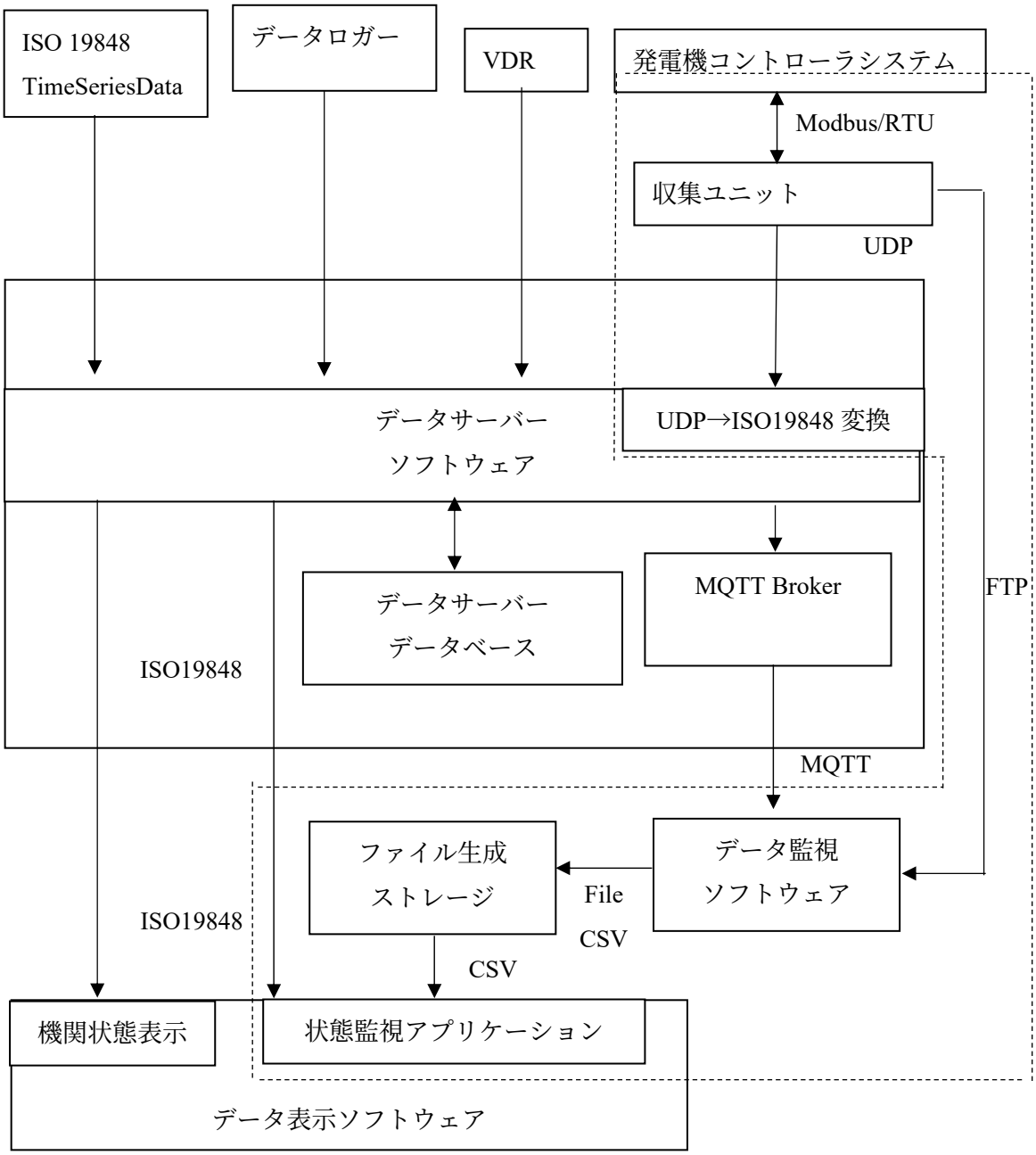
収集ユニットから送信するデータは UDP 通信の独自プロトコルとなっており、ISO19847/19848 のデータサーバー規格では容易に使用することが出来ない状態である。集められたデータをUDPの独自プロトコルからISO19848の規格で入力可能なように変換する必要がある。本変換はISO19848に基づくLocal ID等を付与しているData Channel listのXMLファイルとUDPのプロトコル情報を紐づけて変換させるようにした。変換後はサーバー標準規格に準拠したデータとなっているため、既存の機関監視用アプリケーション等で使用されているIoTデータサーバーのソフトウェアや命令コマンドによって動作させることが出来るようになる。

また、データベースから、状態監視アプリケーションで使用するデータを抽出する機能の設計を行った。本機能はサーバー標準規格で提供されている MQTT プロトコルを用いて実現する。

最後にデータロギング機能のデータは、UDP データ内にデータロギング機能の CSV が生成されたフラグを設け上記 MQTT によるデータ監視により CSV ファイル生成を検知し、FTP サーバー機能を用いて入手する仕組みとした。

抽出したデータとデータロギング用 CSV は状態監視アプリケーションに適したフォーマットに変換した後ファイル生成し、状態監視アプリケーションに送られる仕組みとなっている。

以上 3.2 項の設計結果の概念を図 12 に示す。



※ 鎖線内が本システムでの設計範囲である

図 12 IoT データサーバー概念図

3.3　データ出力ソフトウェアの開発

3.3.1　収集ユニットと発電機コントローラシステム間のデータ収集機能開発

1）発電機コントローラシステムのデータ出力ソフトウェアの開発

前項で決定した仕様の通りデータ出力を行うためのソフトウェア開発を実施した。各ユニットの未使用ポートに Modbus/RTU のスレーブ機能の実装を行う。

1-1）自動化ユニットの改修

自動化ユニットでは特殊な対応として Modbus/RTU 通信に対応した実績があった。今回はこのソフトウェアを元に収集ユニットとのやり取りに適するように改修を実施した。

通信設定は他機能モジュールへの影響を考慮し、上述対応で実績のある設定をそのまま採用することにした。以下表 8 に概要を示す。

表 8 自動化ユニット通信設定

項目	設定
通信速度設定	9600 bps
データビット	8 bit
パリティビット有無	あり
奇数/偶数パリティ	偶数
ストップビット	1

Modbus/RTU 通信でアクセスが必要なデータは、3.1.2-1）項で検討した通り、アナログデータ、カウンタ状態、イベントトリガー状態等である。

Modbus の通信ファンクション（読み込み、書き込み機能）の代表的なものを以下表 9 に示す。

表 9 Modbus ファンクションコード

コード	機能説明
0x01	コイル読み込み（ビット）
0x02	入力ステータス読み込み（ビット）
0x03	保持レジスタ読み込み（ワード）
0x04	入力レジスタ読み込み（ワード）
0x05	単一コイル書き込み（ビット）
0x06	単一保持レジスタ書き込み（ワード）
0x0F	複数コイル書き込み
0x10	複数レジスタ書き込み

上記ファンクションコードの機能は Modbus 規格が PLC 用に作られたため、PLC 機能に沿った名称となっている。

本ユニットは PLC ではないため厳密には上述の区分けが出来ていないが、出力が必要なデータのほとんどがビットデータではなくワードデータの為、送信するためにファンクションコード 0x03 の保持レジスタ読み込みコマンドに反応して送信動作を行うように実装を行った。

また、Modbus にはファンクションコードのほかにデータのアドレスとデータ長の指定が必要である。アドレスは下表 10 のように割付を実施した。本アドレスと内部データを紐づけておりデータ要求があれば紐づいたデータを送信するように実装を行った。

表 10 自動化ユニット Modbus マッピング（抜粋）

データ	コード	アドレス	内容
アナログデータ	0x03	782	母線電圧
		784	母線周波数
		786	発電機電圧
		788	発電機周波数
		…	…
		797	力率
カウンタ情報		1900～1904	エンジン起動回数、時間
		1905～1909	ACB 投入回数、時間
		…	
自己診断情報		2340	各通信ライン状態
		…	
データロギング イベント履歴		4000～4004	イベント履歴 1（時間、状態）
		4005～4009	イベント履歴 2（時間、状態）
		…	
		4245～4249	イベント履歴 50（時間、状態）
データロギング アナログ履歴		6000～6299	母線電圧 0.2 秒 x300 点
		6300～6599	母線周波数 0.2 秒 x300 点
		6600～6899	発電機電圧 0.2 秒 x300 点
		6900～7199	発電機周波数 0.2 秒 x300 点
		7200～7499	発電機電流 0.2 秒 x300 点
		7500～7799	発電機負荷 0.2 秒 x300 点

1-2）発停制御ユニットの改修

発停制御ユニットでは、自動化ユニットと異なり Modbus/RTU 通信に対応した実績がない。自動化ユニットのソフトウェアモジュールを参考に収集ユニットとのやり取りに適するように実装を行った。

通信設定は今回使用する未使用ポートを Modbus 以外の用途で使用した実績のある設定をそのまま採用することにした。以下表 11 に概要を示す。

表 11 発停制御ユニット通信設定

項目	設定
通信速度設定	19200 bps
データビット	8 bit
パリティビット有無	あり
奇数/偶数パリティ	奇数
ストップビット	1

Modbus/RTU 通信でアクセスが必要なデータは、3.1.2-2）項で検討した通り、存在するデータほぼすべてである。具体的には内部リレーエリアが 8192 点、入力リレーエリアが 1024 点（実使用は内 32 点）、出力リレーエリアが 1024 点（実使用は内 32 点）、内部ワードメモリエリアが 2000 点存在している。

Modbus の仕様上 1 つのファンクションに割り付けられるアドレス点数は 10000 点までとなっている。よって、すべてのエリアを同じファンクションに割り付けることはできないため、機能により的確にファンクションを使い分けるように実装を行った。表 9 に示した、Modbus の通信ファンクションの、0x01 コイル読み込み、0x02 入力ステータス読み込み、0x03 保持レジスタ読み込みのコマンドの実装を行った。

また動作確認用として 0x0F 複数コイル書き込み、0x10 複数レジスタ書き込み機能の実装も行った。

表 12 にアドレスマッピングを示す。

表 12 発停制御ユニット Modbus マッピング

データ	コード	アドレス	内容
内部リレー	0x01	0～8191	内部リレー0～8191
入力リレー	0x02	0～1023	入力リレー0～1023
出力リレー	0x02	1024～2047	出力リレー0～1023
内部メモリ	0x03	0～1999	内部メモリ 0～1999

なお実装は前ページ表 12 の通り行ったが、入力リレー、出力リレーは内部リレーにコピーされており、内部メモリはほとんど使用していないので収集ユニットでは内部リレーのデータ収集のみを実施している。

上記実装を行った後、妥当性を確認するため単体での動作検証を行った。

検証環境： 発停制御ユニット 開発ツール Yellow IDE Version 6.46
Modbus エミュレータ QmodMaster0.4.8

図 13 に検証システムを示す。

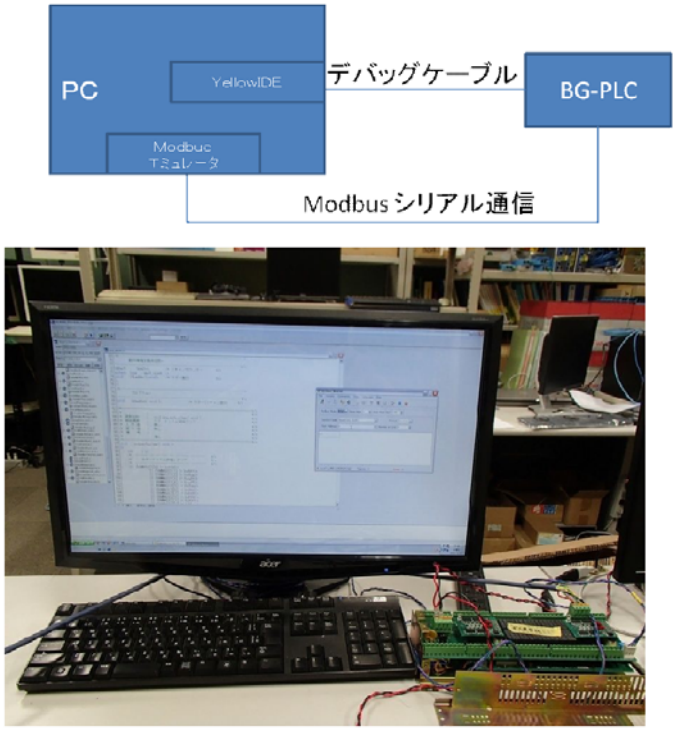


図 13 発停制御ユニット Modbus 実装単体検証システム

本検証では以下の動作を確認し良好であった。

・0x01 コイル読み込み機能

開発ツールで内部リレーを操作し、Modbus エミュレータで 0～8191 までのデータが想定通り読み込み可能なことを確認。

・0x02 入力ステータス読み込み機能

1）開発ツールで入力リレーを操作し、Modbus エミュレータで 0～1023 までのデータが想定通り読み込み可能なことを確認。

2）開発ツールで出力リレーを操作し、Modbus エミュレータで 1024～2047 までのデータが想定通り読み込み可能なことを確認。

・0x03 保持レジスタ読み込み機能

開発ツールで内部メモリを操作し、Modbus エミュレータで 0～1999 までのデータが想定通り読み込み可能なことを確認。

次に実際に収集ユニットで使用する三菱電機製 PLC と実際に接続し各状態を操作し動作確認を実施した。

検証環境：　ソフトウェア改修済みの発停制御ユニット
三菱電機 PLC Q06UDVCPU（CPU） QJ71MB91 及び開発ツール GX-Works2

図 14 に検証システムを示す。

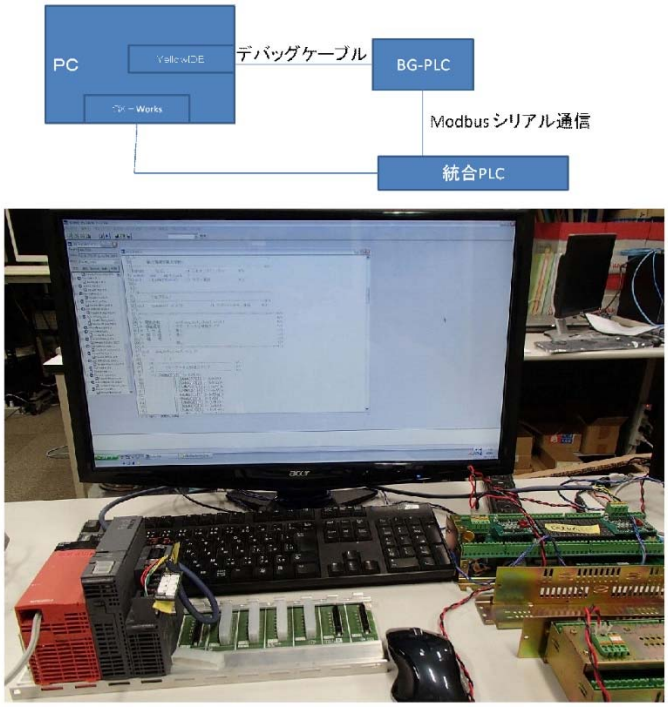


図 14 発停制御ユニット Modbus 通信接続検証システム

本検証では以下の動作を確認し良好であった。

・GX-Works2 を用いて 0x0F 複数コイル書き込み機能を設定して発停制御ユニットへの書き込みを実施。その後 0x01 コイル読み込み機能を設定して同一アドレスを読み込みデータの整合性を確認。

・発停制御ユニットの入力ポートに実際に入力を入れたのち、GX-Works2 を用いて 0x02 レジスタ読み込みアドレス 0～32 を設定して発停制御ユニットの入力値が読めていることを確認。

・発停制御ユニットの出力ポートを実際に出力させたのち、GX-Works2 を用いて 0x02 レジスタ読み込みアドレス 1024～1056 を設定して発停制御ユニットの出力値が読めていることを確認。

・GX-Works2 を用いて 0x10 複数レジスタ書き込み機能を設定して発停制御ユニットへの書き込みを実施。その後 0x03 レジスタ読み込み機能を設定して同一アドレスを読み込みデータの整合性を確認。

最後に発停制御ユニットを発電機コントローラシステムに接続し実際の制御ソフトウェアを動作させ動作確認を実施した。なお本動作検証は次項 2）で述べるデータ収集ソフトウェアの基本機能の試験を兼ねている。

検証環境：　発電機コントローラシステム一式
　　発停制御ユニットモニタリングソフトウェア
　　三菱電機 PLC Q06UDVCPU（CPU） QJ71MB91 及び開発ツール GX-Works2

図 15 に検証システムを示す。

図 15 発停制御ユニットの発電機コントローラシステム接続検証システム

本検証では以下の動作を確認し良好であった。

・0x01 コイル読み込み機能
　　発停制御ユニットモニタソフトで内部リレー0～8191 を操作し、収集ユニットの GX-Works モニタで読み取れることを確認。

・0x03 保持レジスタ読み込み機能
　　発停制御ユニットモニタソフトで内部メモリ 0～1999 を操作し、収集ユニットの GX-Works モニタで読み取れることを確認。

・0x0F 複数コイル書き込み機能

収集ユニットの GX-Works モニタで書き込みを実施。発停制御ユニットモニタソフトで内部リレーが変化することを確認。

・0x10 複数レジスタ書き込み機能

収集ユニットの GX-Works モニタで書き込みを実施。発停制御ユニットモニタソフトで内部メモリが変化することを確認。

以上を確認し、発電機コントローラシステムのデータ出力ソフトウェアの開発を完了した。

2）収集ユニットのデータ収集ソフトウェアの開発

発電機コントローラシステムのデータ出力ソフトウェアに引き続き、収集ユニット側で発電機コントローラシステムからデータを取得するソフトウェアについても開発した。

発電機コントローラシステムからの周期データ取得は、Modbus ユニット QJ71MB91 の自動交信機能設定を用いてデータ受信を行う。本機能はラダーソフトウェアを作成せずに、PLC の開発ツールにおいて設定を行うと自動で通信処理を実施する機能である。PLC メーカーが保証した機能であり、信頼性が高く、開発期間も短縮できるため今回採用した。データロギング部分の取得は頻度が定周期ではないため必要な時にのみ通信処理が起動するようにラダー言語で通信ソフトウェアを開発した。本データ取得も PLC 側で Modbus 専用命令を持っておりラダーでパラメータ入力設定を行い通信開始することによって容易に通信処理を構築できる。

通信設定で得られたデータは収集ユニット内の汎用データレジスタに保存するようにした。

まず、データ交信における通信プロトコル設定の内容を表 13 に示す。本設定は発電機コントローラシステム側の設定に合わせている。

表 13 Modbus ユニット通信設定

項目	CH 1（自動化ユニット）	CH 2(発停制御ユニット)
データビット	8 bit	8 bit
パリティビット有無	あり	あり
奇数/偶数パリティ	偶数	奇数
ストップビット	1	1
フレームモード	RTU モード	RTU モード
RUN 中書き込み	許可	許可
通信速度設定	9600 bps	19200 bps
CH1,2　局番設定	自動交信設定の設定を優先	

以下図 16 に自動交信設定の設定画面の一部を示す。

項目	CH1	CH2
自動交信パラメータ	QJ71MB91をマスタとして, 自動交信機能を使用する場合に設定します。	
自動交信パラメータ1	自動交信に関するパラメータを設定します。	
設定有無	1:有効	1:有効
対象局番	1	1
次要求間隔タイマ値	0	0
応答監視タイマ値/ブロードキャストディレイ値	200	100
対象MODBUSデバイス種別指定	0500h:保持レジスタ読出し	0100h:コイル読出し
読出し設定	スレーブからの読出しデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	1000 h	2000 h
対象MODBUSデバイス先頭番号	782	0
アクセス点数	17	2000
書込み設定	スレーブへの書込みデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	0000 h	0000 h
対象MODBUSデバイス先頭番号	0	0
アクセス点数	0	0
自動交信パラメータ2	自動交信に関するパラメータを設定します。	
設定有無	1:有効	1:有効
対象局番	1	1
次要求間隔タイマ値	100	0
応答監視タイマ値/ブロードキャストディレイ値	99	100
対象MODBUSデバイス種別指定	0500h:保持レジスタ読出し	0100h:コイル読出し
読出し設定	スレーブからの読出しデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	1032 h	207D h
対象MODBUSデバイス先頭番号	1900	2000
アクセス点数	33	2000
書込み設定	スレーブへの書込みデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	0000 h	4000 h
対象MODBUSデバイス先頭番号	0	0
アクセス点数	0	10
自動交信パラメータ3	自動交信に関するパラメータを設定します。	
設定有無	0:無効	1:有効
対象局番	1	1
次要求間隔タイマ値	0	0
応答監視タイマ値/ブロードキャストディレイ値	0	100
対象MODBUSデバイス種別指定	0000h:指定なし	0100h:コイル読出し
読出し設定	スレーブからの読出しデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	1033 h	20FA h
対象MODBUSデバイス先頭番号	1900	4000
アクセス点数	1	2000
書込み設定	スレーブへの書込みデータに関するパラメータを設定します。	
先頭バッファメモリアドレス	0000 h	0000 h
対象MODBUSデバイス先頭番号	0	0
アクセス点数	0	0

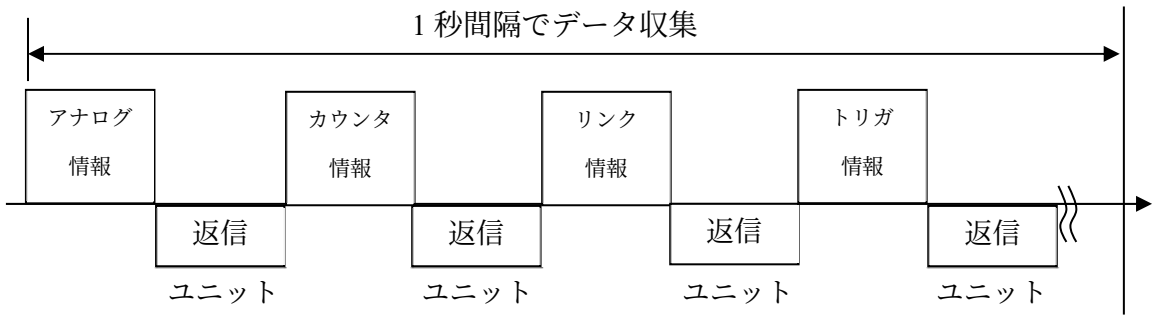
図 16 自動交信設定

1 秒周期でデータ収集を行うように要求間隔タイマ、応答監視タイマなどを設定することにより定周期での通信を可能とした。

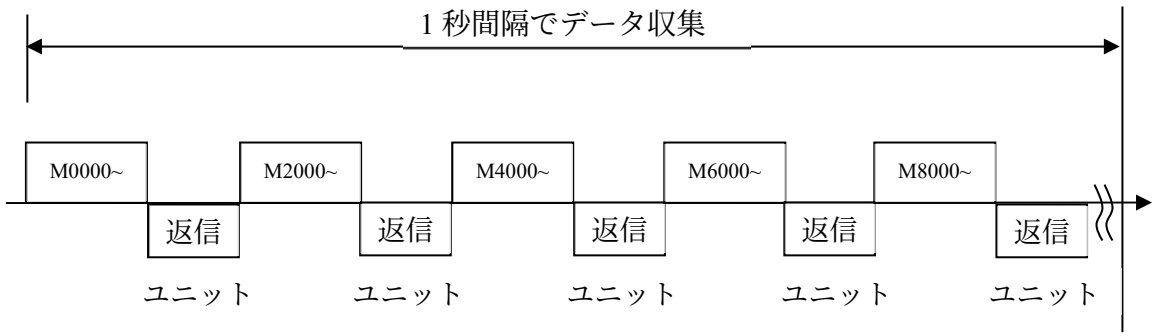
CH1 の自動化ユニットのデータはアナログデータ、その他データの 2 回に分けて通信処理を行うようにした。CH2 の発停制御ユニットは必要な点数が 8000 点を超えている。自動交信機能の制限として一度にアクセスできるビット点数が 2000 点までに限られているため、内部リレーの取得の為、2000 点を 4 回と 192 点を 1 回の計 5 回通信処理を実施するようにした。

この機能で実現される通信タイミングの概念を以下図 17 に示す。

CH1 ⇔ 自動化ユニット　通信状態



CH2 ⇔ 発停制御ユニット 通信状態



※ 自動化ユニット通信状態のトリガ情報は必要な場合のみ専用命令で追加される

図 17　発電機コントローラシステムとのデータ交信タイミング

次に、データロギング部分の通信による取得について説明する。

データロギング機能のデータ取得は異常発生時のみの不定期となる。異常発生時にデータロギング機能が動作すると、自動化ユニットのアナログ情報内にフラグデータを設けており、当該フラグが動作する。本フラグをラダー言語内で監視し、動作した場合自動化ユニットへトリガ情報取得コマンドを発行するようにソフトウェアを作成した。

ラダーソフトウェアは、

・フラグ周期監視のメインソフトウェア Trigger Check

・トリガデータ取得用ファンクションブロック ReadTriggerData

・実データ取得処理のファンクションブロック Modbus1Shot

という階層構造で作成した。図 18 に概要を示す。

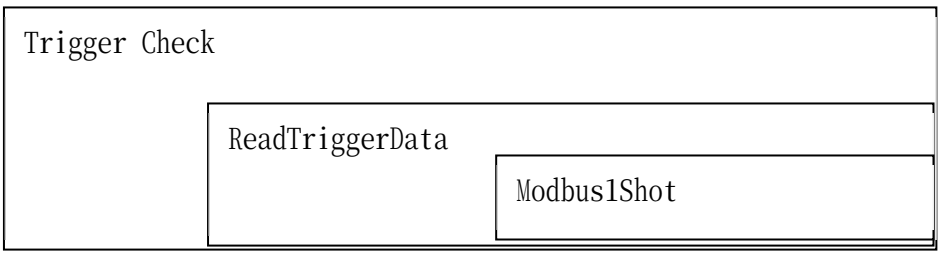


図 18 データロギング収集ソフトウェア構造

まず Trigger Check でデータロギング機能が動作フラグを監視している。動作フラグが反応すると、ReadTriggerData が動作する。本ソフトウェアで表 10 の 4000～4249、6000～7799 のデータを取得する。この時、Modbus ユニットの制約で一回にワードデータは 125 点までしか取得できないため上記データを 25 点 x2 回、100 点 x18 回の計 20 回に分けて取得する。定周期通信のタイミングに影響を及ぼさないように、1 定周期で 1 回のみ取得する。よって一連の取得で図 17 のトリガ情報取得部分が 20 回挿入される。ReadTriggerData 内にはどこまでデータを取得したかのカウンタ機能も持たせている。トリガデータの取得についてリアルタイム性は求められておらず上記方法でも 30 秒程度で取得でき問題はない。

Modbus1Shot は ReadTriggerData 内のカウンタ機能から通信アドレスを判定し設定し送受信処理を実施する。

以下図 19 に作成したラダーの抜粋を示す。

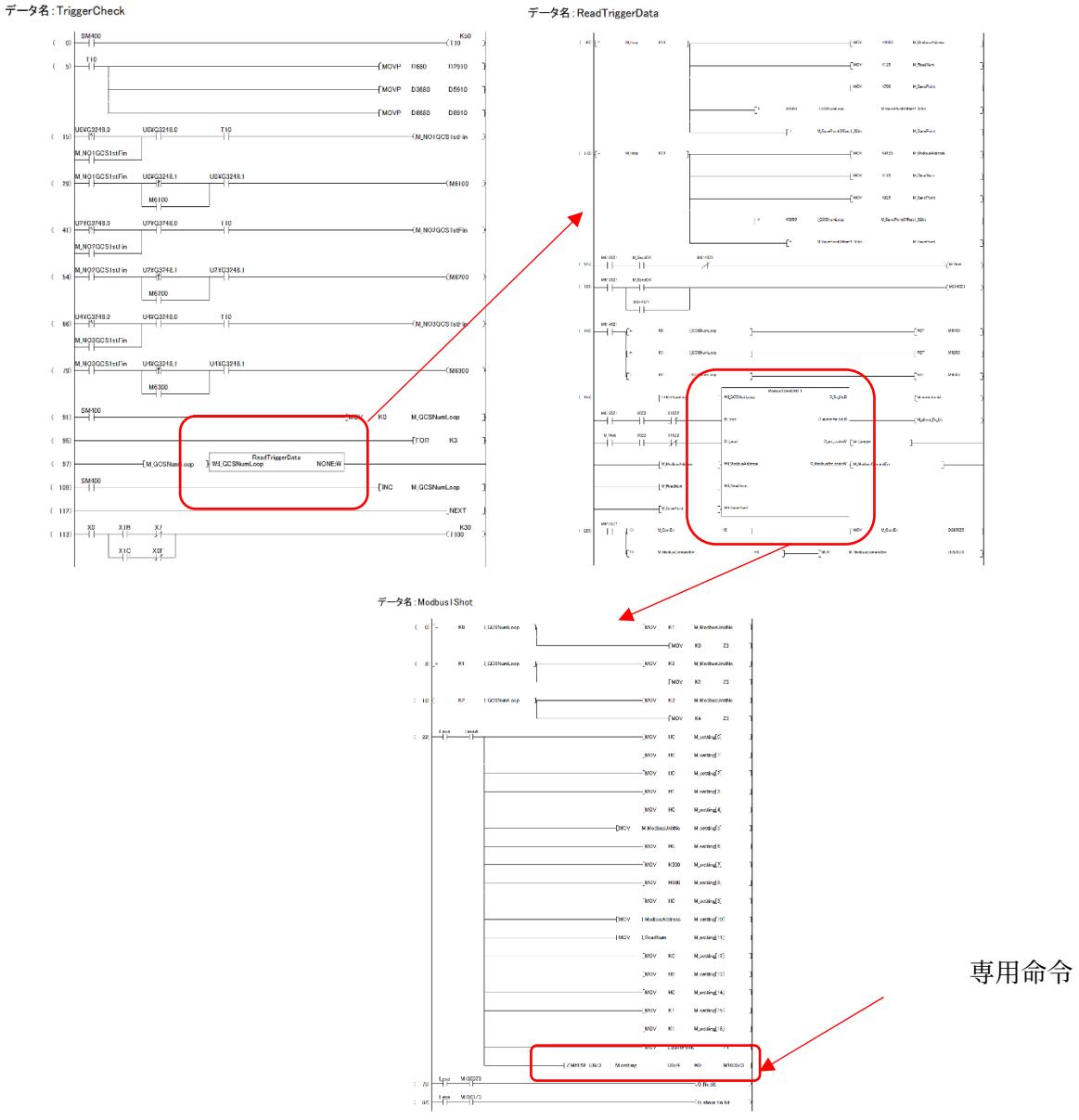


図 19 データログ取得プログラム

開発したソフトウェアが正常動作するかデバッグシステムを用いて動作検証を実施した。

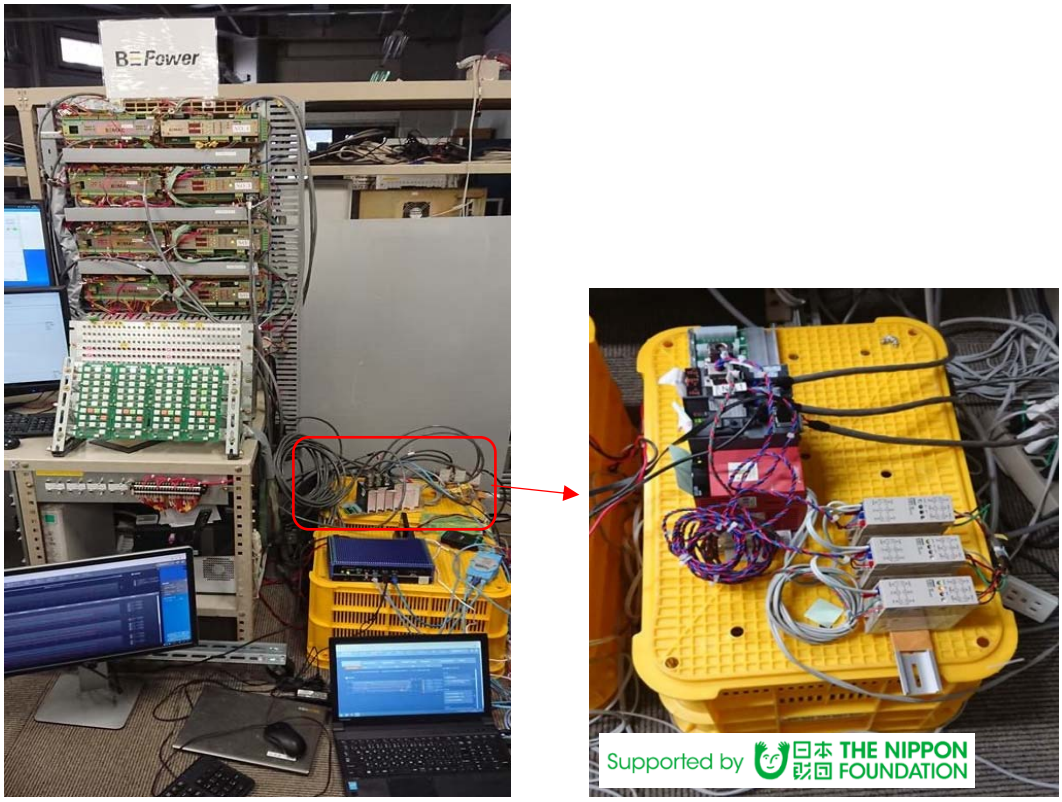


図 20 収集ユニットと発電機コントローラデバックシステムとの接続

図 20 の通り実際の収集ユニットを発電機コントローラデバックシステムとの接続を実施し、動作検証を実施した。当初、収集データの自動交信設定やデータロギング収集ソフトウェアに不具合があったが、改善を行い、遅延なく安定したデータ収集が可能となった。本検証は以下の項目について実施した。

・発電機コントローラシステム及び収集ユニットを実際の動作状態とし、収集ユニットのデバイスモニタ機能を用いて取得する周期データが指定のアドレスに収集できていてモニタ可能かどうかを確認した。

・発電機コントローラシステムで実際に異常状態を発生させ、データロギング機能が作動すること、また、トリガデータが収集ユニットに正常に収集されているかデバイスモニタ機能を用いて収集ユニット内の指定アドレスを目視確認した。

・周期取得のタイミング計測を実施し所要の 1 秒周期に対して、0.5 秒～0.8 秒でデータ取得できており、仕様を満足できていることを確認した。

・一か月以上の長期連続動作検証を実施し、従来機能が劣化したり喪失したり影響を及ぼさないかどうかを確認した。

上記までの結果をもって、収集ユニットのデータ収集ソフトウェアの開発を完了した。

3.3.2　収集ユニットから IoT データサーバーへのデータ出力機能開発

3.3.1 項の開発によって収集したデータは、IoT データサーバーへ送信される。

3.2.3 項の設計結果より、収集ユニットと IoT データサーバー間は船内 LAN で接続され ETHERNET でデータ出力される。定周期の送信は UDP 通信での接続で、1 秒定周期で収集データを出力する。異常発生時のログデータは CSV ファイルとして保存して FTP 機能で IoT データサーバーが任意のタイミングで取得するようにしている。

以下に詳細を記載する。

1) 定周期データの出力（UDP 送信）

前項の開発により、収集ユニットの Modbus 通信ユニットでデータを受信することが可能となった。収集したデータは収集ユニットの CPU 内部の汎用データレジスタエリアに保存されている。

まず、収集したデータは 1 秒周期で UDP 送信に適したようにバイナリデータを ASCII データへ変換する。1 秒周期は PLC の機能である定周期起動モード設定を使用した。本機能はソフトウェアでタイマなどの設定やカウントをしなくても周期的にソフトウェアを動作させることが出来る。変換したデータは IoT サーバーでデータ解析が容易となるようにワードデータ毎のコンマ区切りデータとした。この結果、自動化ユニット及び発停制御ユニットから取得したデータは、発電機 1 台当たり 4168 バイトの単一データに成型される。本処理はシステムによって発電機台数が変化するため、使用レジスタ領域を変更し発電機台数分同一処理をループ実施するようにした。（最大 4 台まで対応）

成形されたデータは、IoT データサーバーに向けてデータ送信を行う。データ送信には UDP 用にも専用命令コマンドが設けられておりそれを使用することにより容易に出力が可能である。UDP プロトコルでは表 7 にある通り、高速ではあるが信頼性が低いという特徴がある。送信処理にエラーが発生した場合は何らかの形で対策をする必要がある。今回はエラー発生時のデータは失われてしまうが、通信エラーを検知した場合、次の送信時にエラー情報を送信し IoT サーバー側に通知する方式を実装した。

UDP によるデータ送信仕様及び送信タイミングを以下、表 14、図 21 に記載する。また、UDP データ送信動作フローを図 22、ラダーソフトウェアの抜粋を図 24 に示す。

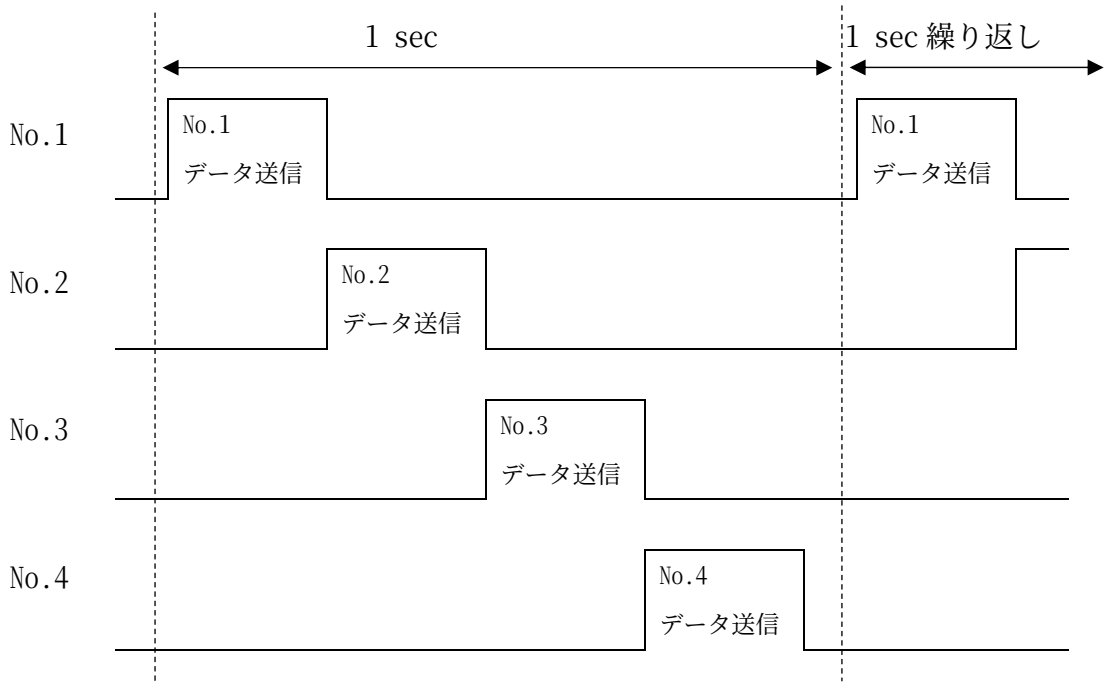
ソフトウェア実装後、単体で動作検証を実施した。

検証環境： 汎用 PC ネットワークプロトコルアナライザ（WIRE SHARK）
収集ユニット 三菱電機 PLC Q06UDVCPU（CPU）QJ71MB91

収集ユニットから汎用 PC に UDP データを定期送信した。結果を図 23 に示す。1 台分のデータがおおよそ 17 ミリ秒で送信されており、最大 4 台分のデータは 70 ミリ秒、100 ミリ秒以内には送信が完了し約 900 ミリ秒は休止状態である。十分余裕を持った定期送信ができていることを確認した。

表 14 UDP データ送信設定

項目	仕様	備考
送信データ	各発電機　4168 バイト	※ASCII 変換済
送信間隔	1 秒(最大 4 台分)	
LAN	100/10Mbps	自局 IP：192.168.2.100
通信方式	UDP ソケット通信	独自プロトコル
自局ポート番号	4000	
交信相手アドレス	192.168.2.51	IoT Data Server
交信相手ポート番号	40001	



※上記は概略図であり、実際はデータ未送信の休止時間が多い

図 21 データ送信仕様及び送信タイミング

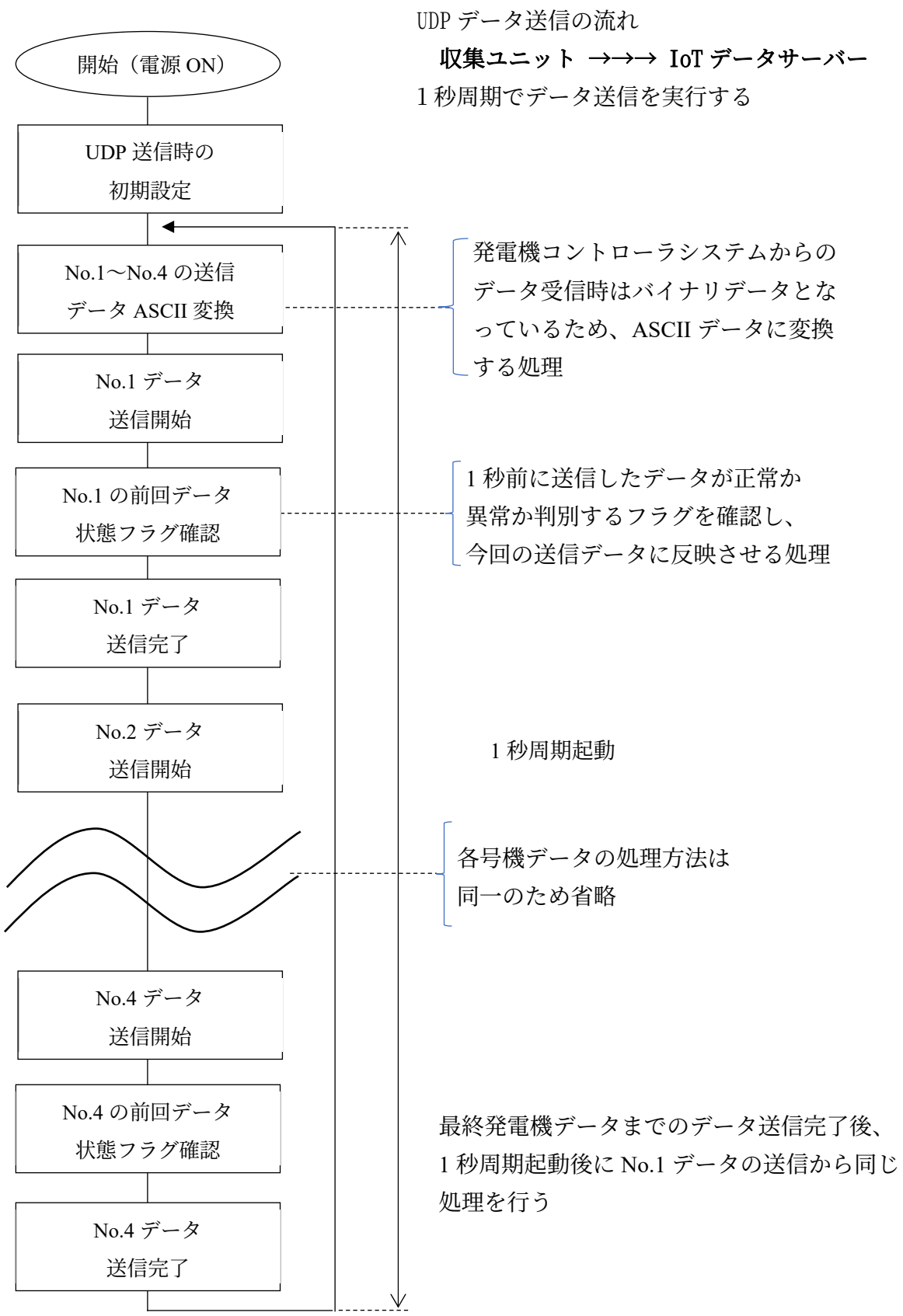


図 22 データ送信動作フロー

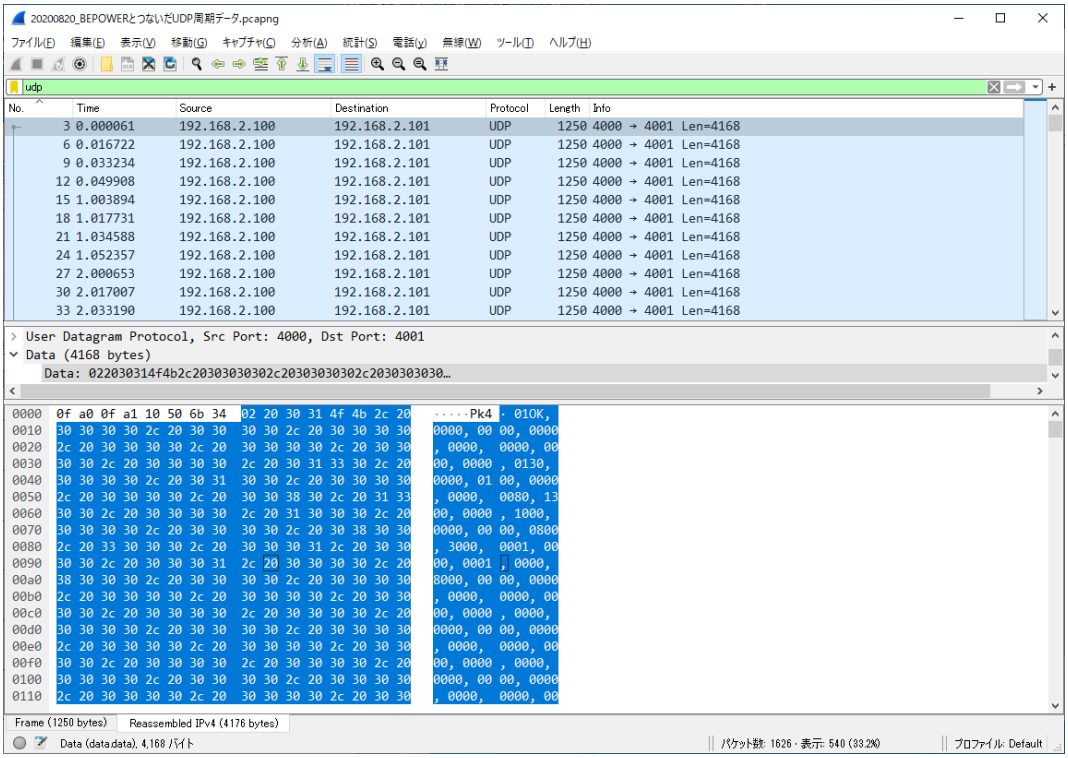


図 23 ネットワークプロトコルアナライザによる検証

データ名：UDP

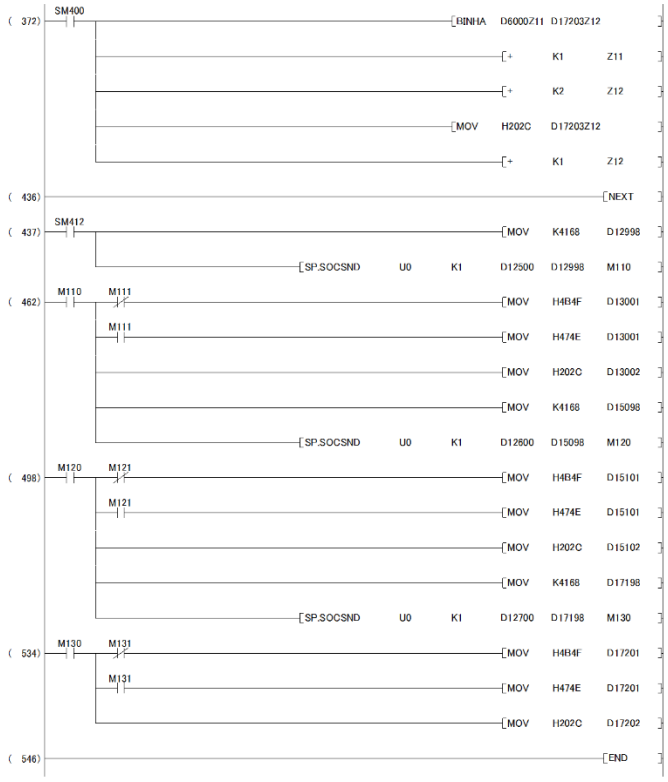


図 24 UDP 送信ラダープログラム

2) イベントデータの出力（CSV ファイル）

船内でのブラックアウト発生、母線電圧異常などの異常発生した場合に収集したデータも前項と同じく収集ユニットの汎用データレジスタに保存されている。前項で説明したようにデータロギング機能のデータ収集は 30 秒程度で完了するため、監視フラグ検知後 60 秒後に、データ取得完了フラグが動作するように実装を行った。データ取得完了フラグが動作した時に該当する発電機号機の汎用データレジスタにあるデータを CSV ファイルとして成型するように実装を行った。なお実装は収集ユニット PLC の CSV ファイル作成支援コマンドを用いた。汎用データレジスタのエリアを指定すればデータのバイナリ→ASCII 変換、CSV ファイルの作成を一括で実施することが出来る。CSV ファイルは収集ユニット内の SD カードに保存される。

生成される CSV ファイルは、表 15 のように各発電機でファイル名を分割して保存する。

なお、本 CSV ファイルはあくまでも IoT データサーバーへのデータ受渡用であり長期保存は考慮していないため、01～09 まで 9 個ファイルが生成された場合、10 個目は 01 を上書きするようにした。よって IoT データサーバーは 9 回保存が行われるまでにデータを取得する必要がある。

表 15 保存 CSV ファイル名

No.1 発電機	No.2 発電機	No.3 発電機	No.4 発電機
NO1LOG01.csv	NO2LOG01.csv	NO3LOG01.csv	NO4LOG01.csv
NO1LOG02.csv	NO2LOG02.csv	NO3LOG02.csv	NO4LOG02.csv
NO1LOG03.csv	NO2LOG03.csv	NO3LOG03.csv	NO4LOG03.csv
NO1LOG04.csv	NO2LOG04.csv	NO3LOG04.csv	NO4LOG04.csv
NO1LOG05.csv	NO2LOG05.csv	NO3LOG05.csv	NO4LOG05.csv
NO1LOG06.csv	NO2LOG06.csv	NO3LOG06.csv	NO4LOG06.csv
NO1LOG07.csv	NO2LOG07.csv	NO3LOG07.csv	NO4LOG07.csv
NO1LOG08.csv	NO2LOG08.csv	NO3LOG08.csv	NO4LOG08.csv
NO1LOG09.csv	NO2LOG09.csv	NO3LOG09.csv	NO4LOG09.csv

上述のように、速やかなデータの受け渡しが必要なため、前項で説明した UDP 送信データ内に CSV 作成完了フラグを設けている。本フラグを IoT データサーバーで監視し速やかなデータの受け渡しを実現している。

PLC の CSV ファイル作成支援コマンドは同時に 1 処理しか実行できない。しかしながら本処理は 2 台に発電機ユニットで異常が同時に発生するなど多重で動作させなければならない場合があるため、優先度をつけ、欠落なく順に CSV ファイルを生成するように実装を行った。

以下図 25 に本 CSV 生成処理のフローチャートを記載する。

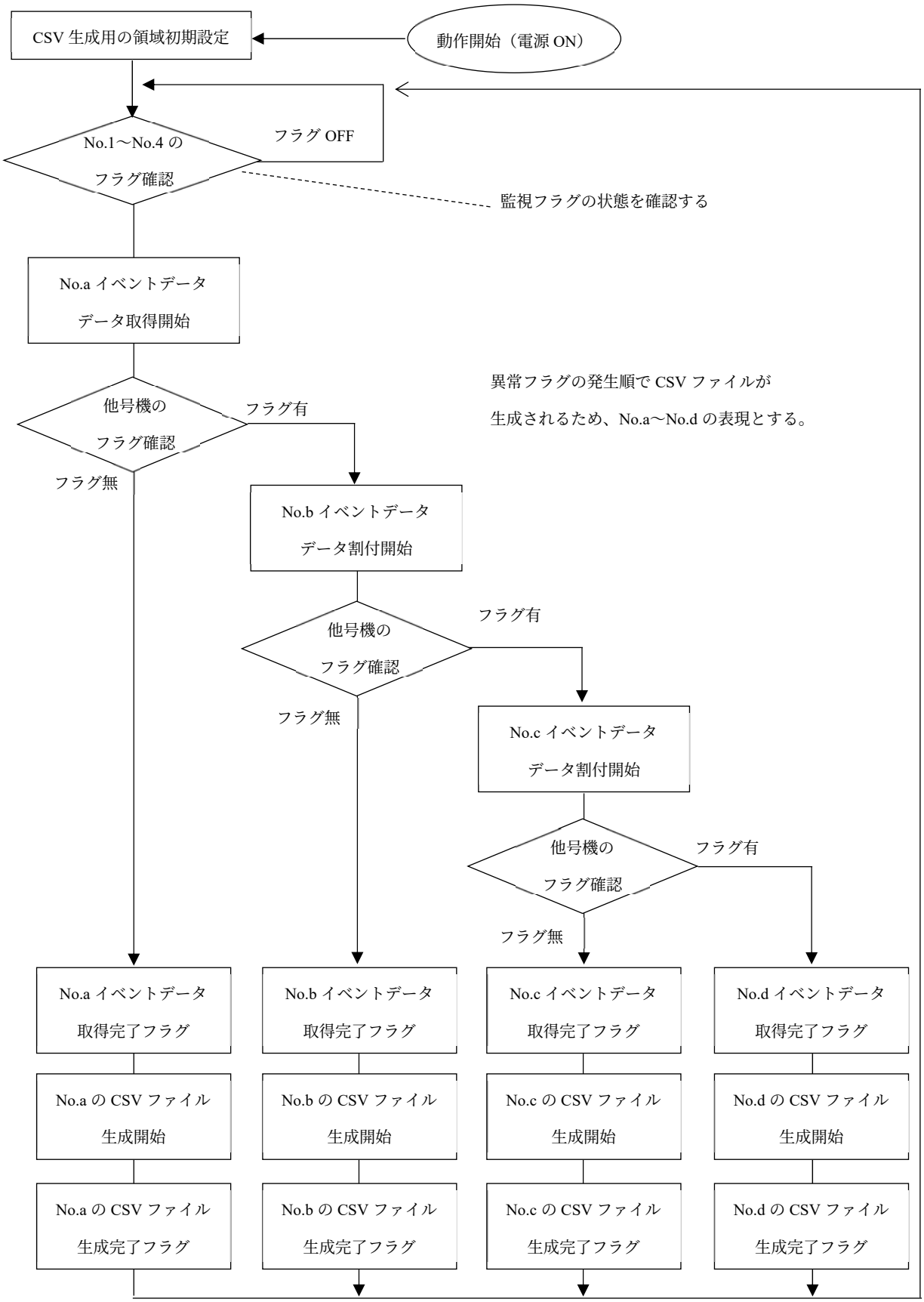


図 25　CSV ファイル生成動作フロー

作成された CSV ファイルは収集ユニット PLC の FTP サーバー機能（下図 26）によって外部へ公開されるように設定を行った。本機能により、IoT データサーバー側でサーバー名とパスワードを用いて FTP コマンドで CSV ファイルを取得可能となった。

図 26 FTP 設定

ソフトウェア実装後、動作検証を実施した。

検証環境：発電機コントローラシステム

収集ユニット 三菱電機 PLC Q06UDVCPU（CPU）QJ71MB91

収集ユニットと発電機コントローラシステムを接続し、異常動作を発生させ、収集ユニットのモニタソフトで各フラグのデータが想定通り動作しているか確認した。その後ユニットから SD カードを抜き取り CSV ファイルが生成されているか確認した。また、汎用 PC から FTP コマンドでアクセスして CSV ファイルを取得できることを確認した。

3.3.3　IoT データサーバー内でのデータベース化・解析・抽出機能開発

3.2.4 項で設計した仕様に基づき、Data Channel list の XML ファイルと UDP のプロトコル情報を紐づけて変換させるプログラムを構築した。変換後はサーバー標準規格に準拠したデータとして扱え、既存の機関監視用アプリケーション等で使用されているソフトウェアや命令コマンドによって動作させることが出来るようになる。実際に、プログラム構築後、既存の機関監視用アプリケーションなどからデータに正常にアクセスできることを確認した。

次に、データベースから状態監視アプリケーションで使用するデータを抽出する機能の設計に基づき、データ監視ソフトウェアの実装を行った。まず後述の 3.5 項で設計した状態監視アプリケーションの画面を作成する上でトリガになる条件データを MQTT プロトコルで周期的に監視する。条件データに変更があれば、ISO19848 Time Series Data を HTTP で必要な間隔のデータをデータベースから抽出・取得する。必要なデータは条件データ更新タイミングの前 50 秒、後ろ 10 秒である。

データロギング機能のイベントデータは、前項で実装を行ったデータロギング機能の CSV が生成されたフラグについても条件データの監視に含めることにより、MQTT によるデータ監視により収集ユニットでの CSV ファイル生成を検知し、FTP サーバー機能を用いて入手する仕組みとした。

抽出したデータとデータロギング用 CSV は状態監視アプリケーションに適したフォーマットに変換される。行に UTC 時間、列にデータ項目が入力された CSV ファイルを生成する。本ファイルは IoT データサーバー内のファイルストレージに保存され、状態監視アプリケーションに送られる仕組みとなっている。以下図 27 にファイルストレージ構成を示す。

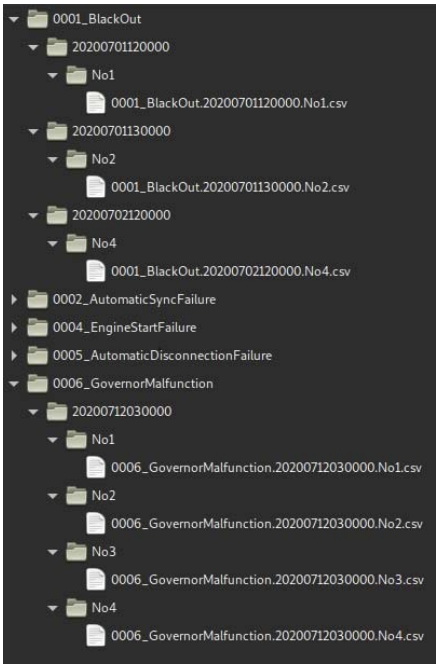
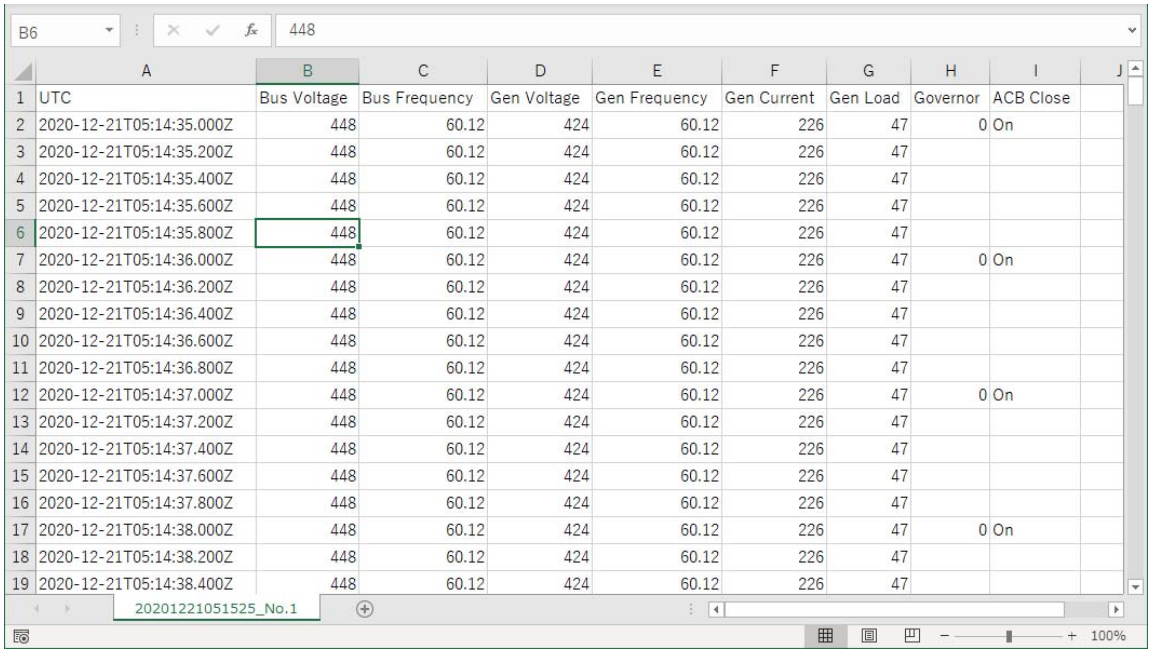


図 27 ファイルストレージ構成

図 28 に生成された CSV ファイルのサンプルを記載する。本 CSV ファイルのデータと実動作させた発電機コントローラシステムの状態が合致することを確認でき、状態監視アプリケーション用のデータを仕様通り生成することができた。

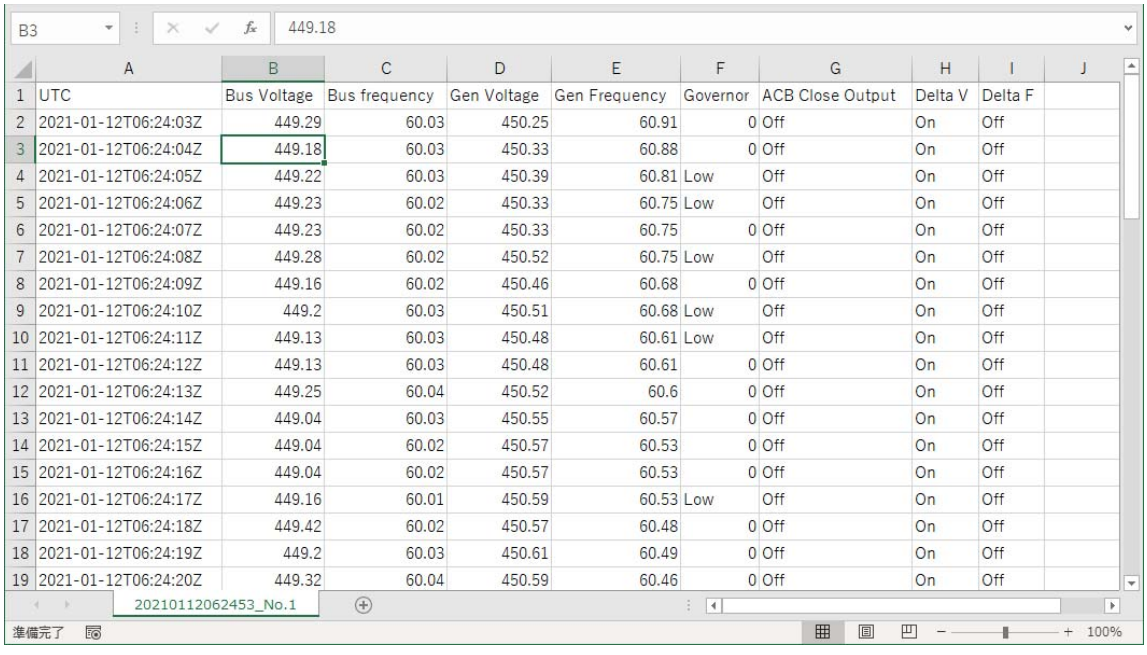


B6 | 448

	A	B	C	D	E	F	G	H	I	J
1	UTC	Bus Voltage	Bus Frequency	Gen Voltage	Gen Frequency	Gen Current	Gen Load	Governor	ACB Close	
2	2020-12-21T05:14:35.000Z	448	60.12	424	60.12	226	47	0	On	
3	2020-12-21T05:14:35.200Z	448	60.12	424	60.12	226	47			
4	2020-12-21T05:14:35.400Z	448	60.12	424	60.12	226	47			
5	2020-12-21T05:14:35.600Z	448	60.12	424	60.12	226	47			
6	2020-12-21T05:14:35.800Z	448	60.12	424	60.12	226	47			
7	2020-12-21T05:14:36.000Z	448	60.12	424	60.12	226	47	0	On	
8	2020-12-21T05:14:36.200Z	448	60.12	424	60.12	226	47			
9	2020-12-21T05:14:36.400Z	448	60.12	424	60.12	226	47			
10	2020-12-21T05:14:36.600Z	448	60.12	424	60.12	226	47			
11	2020-12-21T05:14:36.800Z	448	60.12	424	60.12	226	47			
12	2020-12-21T05:14:37.000Z	448	60.12	424	60.12	226	47	0	On	
13	2020-12-21T05:14:37.200Z	448	60.12	424	60.12	226	47			
14	2020-12-21T05:14:37.400Z	448	60.12	424	60.12	226	47			
15	2020-12-21T05:14:37.600Z	448	60.12	424	60.12	226	47			
16	2020-12-21T05:14:37.800Z	448	60.12	424	60.12	226	47			
17	2020-12-21T05:14:38.000Z	448	60.12	424	60.12	226	47	0	On	
18	2020-12-21T05:14:38.200Z	448	60.12	424	60.12	226	47			
19	2020-12-21T05:14:38.400Z	448	60.12	424	60.12	226	47			

20201221051525_No.1 | 100%

ブラックアウト診断用データ例



B3 | 449.18

	A	B	C	D	E	F	G	H	I	J
1	UTC	Bus Voltage	Bus frequency	Gen Voltage	Gen Frequency	Governor	ACB Close Output	Delta V	Delta F	
2	2021-01-12T06:24:03Z	449.29	60.03	450.25	60.91	0	Off	On	Off	
3	2021-01-12T06:24:04Z	449.18	60.03	450.33	60.88	0	Off	On	Off	
4	2021-01-12T06:24:05Z	449.22	60.03	450.39	60.81	Low	Off	On	Off	
5	2021-01-12T06:24:06Z	449.23	60.02	450.33	60.75	Low	Off	On	Off	
6	2021-01-12T06:24:07Z	449.23	60.02	450.33	60.75	0	Off	On	Off	
7	2021-01-12T06:24:08Z	449.28	60.02	450.52	60.75	Low	Off	On	Off	
8	2021-01-12T06:24:09Z	449.16	60.02	450.46	60.68	0	Off	On	Off	
9	2021-01-12T06:24:10Z	449.2	60.03	450.51	60.68	Low	Off	On	Off	
10	2021-01-12T06:24:11Z	449.13	60.03	450.48	60.61	Low	Off	On	Off	
11	2021-01-12T06:24:12Z	449.13	60.03	450.48	60.61	0	Off	On	Off	
12	2021-01-12T06:24:13Z	449.25	60.04	450.52	60.6	0	Off	On	Off	
13	2021-01-12T06:24:14Z	449.04	60.03	450.55	60.57	0	Off	On	Off	
14	2021-01-12T06:24:15Z	449.04	60.02	450.57	60.53	0	Off	On	Off	
15	2021-01-12T06:24:16Z	449.04	60.02	450.57	60.53	0	Off	On	Off	
16	2021-01-12T06:24:17Z	449.16	60.01	450.59	60.53	Low	Off	On	Off	
17	2021-01-12T06:24:18Z	449.42	60.02	450.57	60.48	0	Off	On	Off	
18	2021-01-12T06:24:19Z	449.2	60.03	450.61	60.49	0	Off	On	Off	
19	2021-01-12T06:24:20Z	449.32	60.04	450.59	60.46	0	Off	On	Off	

20210112062453_No.1 | 準備完了 | 100%

自動同期投入不良診断用データ例

図 28 状態監視アプリケーション用 CSV データ

3.4　データ収集（実船搭載）

3.4.1　実船搭載前の総合試験

実船搭載に先立ち、3.3 項及び後述の 3.5 項で開発したソフトウェアが本船構成で正常動作するか、BEMAC 社内のシミュレーション盤を用いて総合試験を実施した。

試験日：2021 年 1 月 12 日及び 13 日

試験場所：BEMAC 株式会社 大西工場

試験方法：シミュレーション盤を本船ソフトウェアと同一状態とし、試験装置にて実環境を模した試験を実施

総合試験において以下項目の試験を実施し良好であることを確認した。

1)　システム動作中のデータ取得

試験中、データが途切れなく収集でき状態監視アプリケーションで表示できていることを確認した。具体的には状態監視アプリケーションの標準画面で連続してデータが表示できていることを確認した。

2)　状態監視アプリケーションの各画面の動作確認

状態監視アプリケーションの各画面データが、正常に表示されることを確認した。具体的には試験装置にて異常を発生させ、それに反応しデータを取得し各画面データが正常に表示されることを確認した。

図 29 にシミュレーション盤を使用した試験状況を示す。

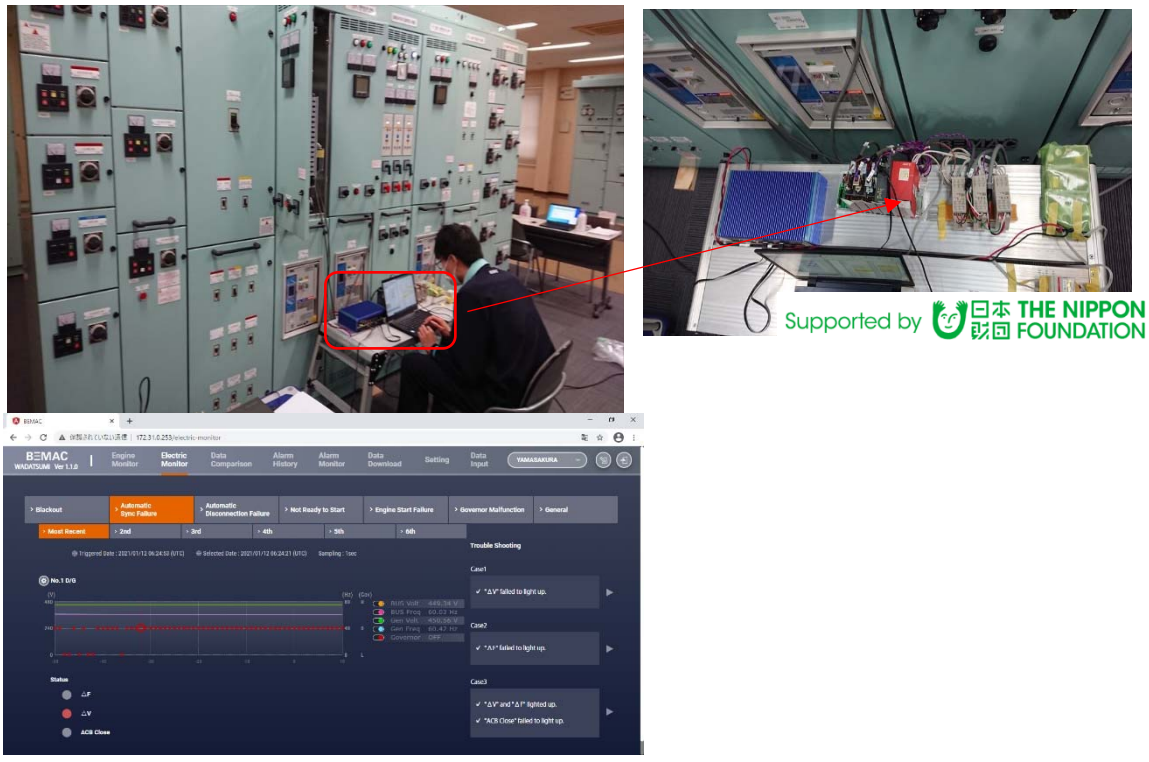


図 29 シミュレーション盤を使用した総合試験

3.4.2　実船搭載前の準備

実船搭載を行う前に搭載船選定、事前調査及び準備を行った。搭載は以下仕様の船舶に対して実施する。

船種：石炭運搬船　（NK（MO） 沿海 非国際 （内航船））
発電機仕様：AC450V 3φ 60Hz 750kVA 963A　3台

選定の理由として、

・内航船であるため通信インフラに 4G などの携帯回線を使用することが可能で、船内の衛星設備等を用いることなくデータ取得検証が可能
・数日周期で航海・停泊・入出港・荷役等が行われ多彩なデータを取得可能
・搭載後に実地検証が必要な場合、容易

等の理由により選定した。
実船調査の結果に基づいて、必要な通信線などの部材の長さを決定し事前準備を実施するとともに各部材の設置場所を決定した。

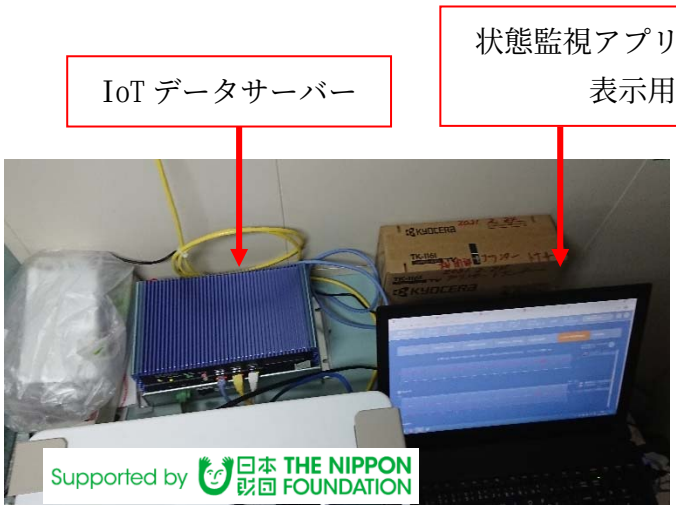
なお、搭載船選定後すぐに船主様及び管理会社様に対して説明を実施したが、船内データの取り扱い、既存システムの改造による影響範囲の確認に時間が割かれた。またコロナウイルスによる緊急事態宣言の影響などにより出張、対面での打ち合わせに制限が発生した結果、搭載時期が 12 月予定より 2 月下旬のドック入り時に遅延した。
その結果、本技術開発事業についても 2 か月の実施期間延長を余儀なくされた。実施期間延長を申込ご了承いただいたことに感謝申し上げる。

3.4.3　実船搭載作業

事前準備を実施後、実船搭載を実施した。

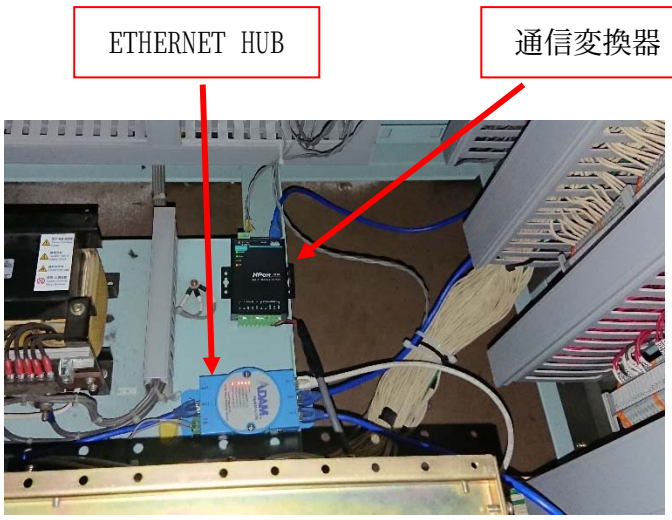
搭載日：2021 年 2 月 24 日
搭載場所：瀬戸内地区造船所修繕ドック

以下に図 30～33 に搭載の状況を示す。なお搭載に際し、次年度の予兆検知に生かすために、今回開発した配電 IoT システム、状態監視アプリケーションのほかに、機関状態監視アプリケーションおよびその周辺機器も同時に搭載を実施した。



監視盤のログプリンタ後方へ設置

図 30 IoT データサーバー及び表示用 PC 設置



ETHERNET HUB は、
・収集ユニット
・データロガー（通信変換）
・データロガー（直接）
が IoT データサーバーに接続されるため設置
通信変換器は機関監視用途

図 31 監視盤内への通信変換機設置

主配電盤内の自動化ユニット及び発停制御ユニットの各ユニットに信号ケーブルを接続し、盤コーナー部分背面に搭載した収集ユニットへ接続する。

なお、自動化ユニット及び、発停制御ユニットのソフトウェアを今回開発したデータ出力ソフトウェア対応バージョンに更新した。

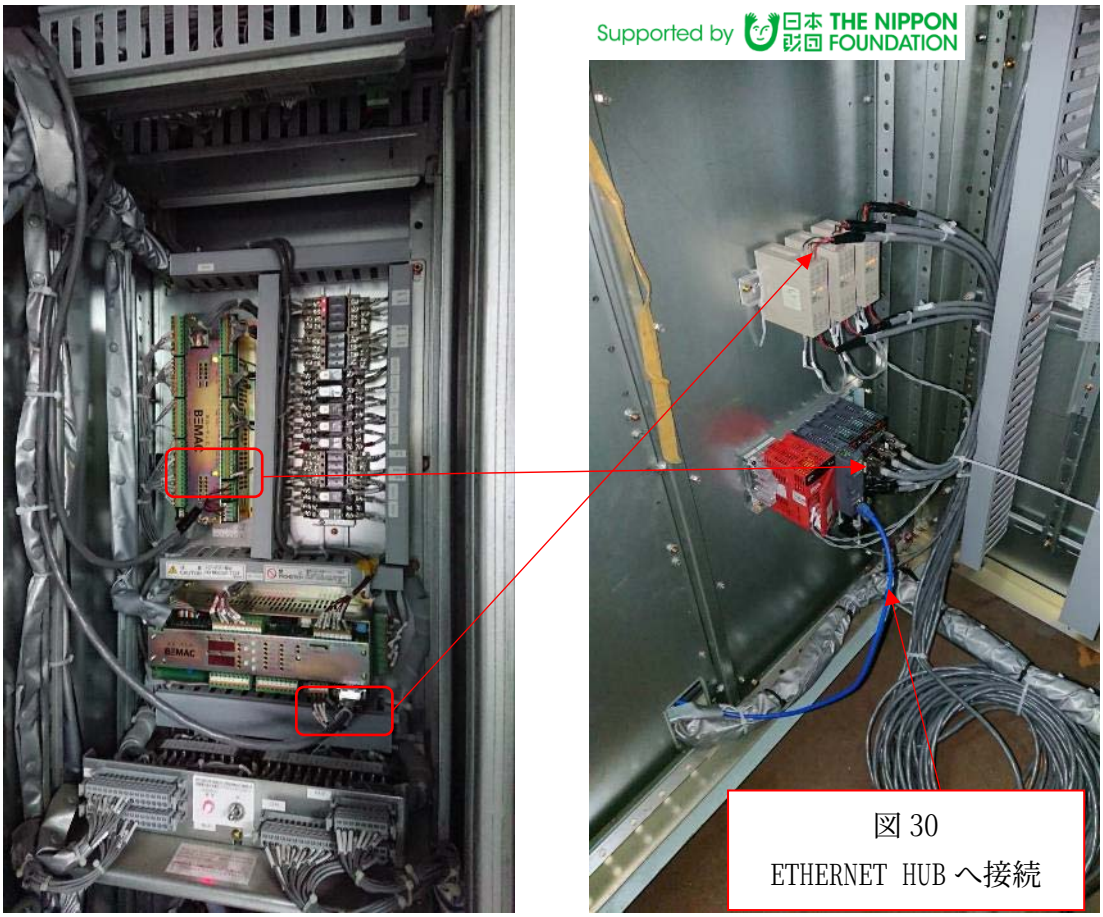


図 32 配電盤内への収集ユニット設置

実船搭載検証では、陸上から頻繁にデータを入手する予定であるので本船設備を使用せず、下図 33 の通り 4G 携帯回線を用いたデータ通信端末を使用して船陸通信を行うようにした。ルータは本船上アッパーデッキの窓際に仮置設置した。

図 33 データ通信端末

3.4.4　実船搭載動作確認

実船搭載後、2021 年 3 月 3 日に一連の動作確認を管理会社、船長、機関長の協力により実施した。ブラックアウト、その他を疑似的ではあるが実設備にて発生させて頂いた。3.4.1 項で確認した模擬システムと同様に動作良好を確認した。動作結果は後述の 3.5 項の図 34～42 によって示される。

実船搭載後、図 33 のデータ通信端末については電波の圏外になる事もなく、約 1 か月間、連続でデータ取得できていることを確認した。

本確認によって、発電機コントローラシステムからのデータ出力機能、収集ユニットでのデータ収集、IoT データサーバーへの出力、IoT データサーバーでのデータベース化、解析、状態監視アプリケーションの動作などすべての項目が仕様通り動作することが確認できた。

報告が前後するが 3.5 項では状態監視アプリケーションについて記載する。

3.5　状態監視アプリケーションの設計・開発

前項までで配電システムの IoT 化によるデータ出力、蓄積が可能となった。本項では具体的なデータ使用方法として状態監視アプリケーションの設計・開発を行う。なお本アプリケーションは協力者である FutureRays 株式会社と共に設計・開発を実施した。

3.5.1　状態監視アプリケーションの項目選定

状態監視アプリケーションの設計を行うにあたって、まず、主に配電盤のアフターサービスを行っているサービス員にヒアリングを実施した。ヒアリングの結果、配電盤で起きる不具合で一番重大なものはブラックアウトである。ブラックアウトの状態監視は必須である。他にも配電盤の自動化に関わる部分の制御不良、発電機エンジンの始動不良等今までに経験した不具合事項を列挙していただいた。その結果、状態監視アプリケーションに対しては、以下機能を盛り込むこととした。

1．ブラックアウト解析機能

電圧、周波数、ガバナ信号、ACB 動作信号を監視、表示し何が原因でブラックアウトが発生したのか原因把握を行う。

推定原因：AVR、励磁装置、ガバナ、燃料、発電機関異常、ACB 等

2．同期投入不良解析機能

先発、後発機の電圧、周波数、ACB 動作信号等を監視し自動化ユニットでの同期投入不能の場合の原因把握を行う。

推定原因：電圧、周波数制御不良、ACB 不良 等

3．自動解列不能解析機能

自動解列時、発電機負荷、周波数、ガバナ、ACB 動作信号等を監視し解列不能の場合の原因把握を行う。

推定原因：ガバナ、ACB 不良 等

4．始動準備完了解析機能

自動化システムにおいて各条件を監視し発電機が自動モードにならない要因を解析する。

推定原因：制御スイッチ、リミットスイッチ等

5．発電機エンジン始動失敗解析機能

自動化システムにおいて始動信号等を監視し発電機が始動しない要因を解析する。

推定原因：発電機関、スタート信号、スピードリレー等

6．ガバナ動作異常解析機能

ガバナが動作不良を起こしたときに要因を解析する。

推定原因：ガバナモーター、発電機関等

上記 6 項目の不具合解析機能に加え、通常状態での電圧、電流、周波数、負荷のトレンドを表示する標準画面の 7 画面の構成とすることにした。

3.5.2　状態監視アプリケーションの設計

3.5.1 項で決定した表示項目に対し、表示方法の設計を行った。事象発生時の個別解析に必要な項目（電圧、電流、電力、制御信号等）を選定し、発生時前 50 秒、後ろ 10 秒間の 1 秒間隔のデータをグラフで表示させる機能、（ブラックアウトのみ 0.2 秒間隔データ）、主要な制御信号をランプ表示させる機能、容易に推定原因解析を可能とするフローチャート機能などを搭載することとした。履歴機能としては、不具合は頻繁には発生せず解析には過去のデータは不要なこともあり、収集ユニットに保存している 9 回分、IoT サーバーに保存している多数のデータすべてではなく状態監視アプリケーションでは過去 6 回分を表示するようにした。

なお、上記アプリケーションに必要なデータは収集ユニットから IoT データサーバーに送られ蓄積したデータより選定され、CSV ファイルの形式で保存されるように 3.3.3 項迄の部分で開発を完了している。

詳細の設計画面は後の 3.5.3 項に記載する。

なお、フローチャート機能について現状は目視による判断とするが、令和 3 年度に実施の予防保全アプリケーションの開発が完了した段階で自動判断に置き換える予定である。

3.5.3　状態監視アプリケーションの開発

前項で設計した仕様に準じて状態監視アプリケーションの開発を実施した。アプリケーションは IoT データサーバーに保存されている CSV ファイルを解析し対応する機能、表示を実施している。また、各解析画面のフローチャート機能にはトラブルシューティングマニュアルへのリンクを搭載し、サービス員だけでなく、船員、監督等でも初期対応が可能なようにした。

以下に実船搭載時に取得した画面の詳細を記載する。すべての項目が時間のずれもなく正常に取得されており動作に問題なく開発を完了した。

1) ブラックアウト解析機能

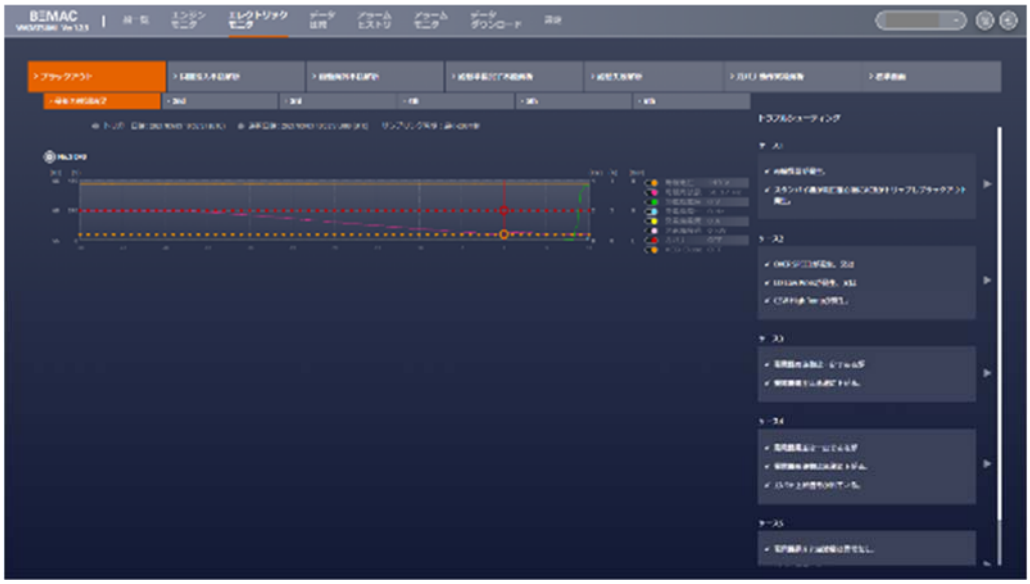


図 34 ブラックアウト解析

図 34 にブラックアウト解析機能画面を記す。この画面ではブラックアウト発生時の、該当する発電機の状態が表示される。

発電機コントローラシステムから取得したデータロギング機能のイベントデータを元に構成している。ブラックアウト画面ではグラフ上で、母線電圧・周波数、発電機電圧・周波数・電流・電力、ガバナ状態、ACB の状態が表示される。グラフ内の任意の場所をクリックすることで、凡例に選択した時の値を表示させる仕様となっている。

また、グラフのズーム機能も搭載されており、より細かい値の変化を確認する事ができる。

ブラックアウト画面のグラフのみサンプリング周期は 200ms となっており、詳細なデータを確認することができる。

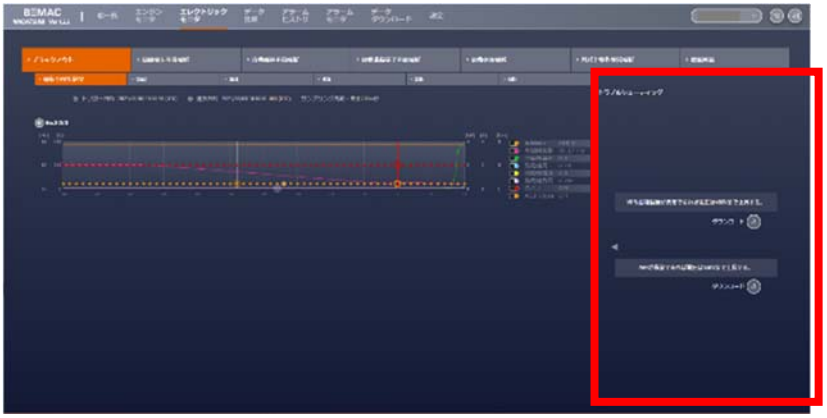


図 35 フローチャート選択

図 35 に示すように、画面右のトラブルシューティングフローチャートのケースを選択すると、図 36 のようにトラブルが起こった原因が表示される。ここでダウンロードボタンをクリックすることで、トラブル発生時の対処法のマニュアルが表示される。

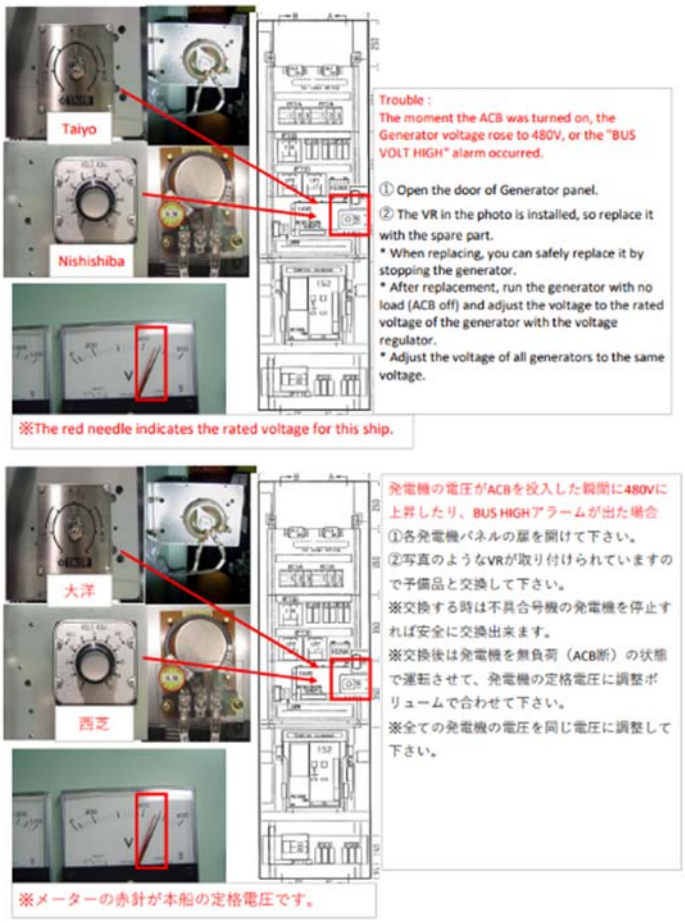


図 36 トラブルシューティングマニュアル
ダウンロードボタンをクリックした際に表示されるマニュアルの一例である。

2) 同期投入不良解析機能

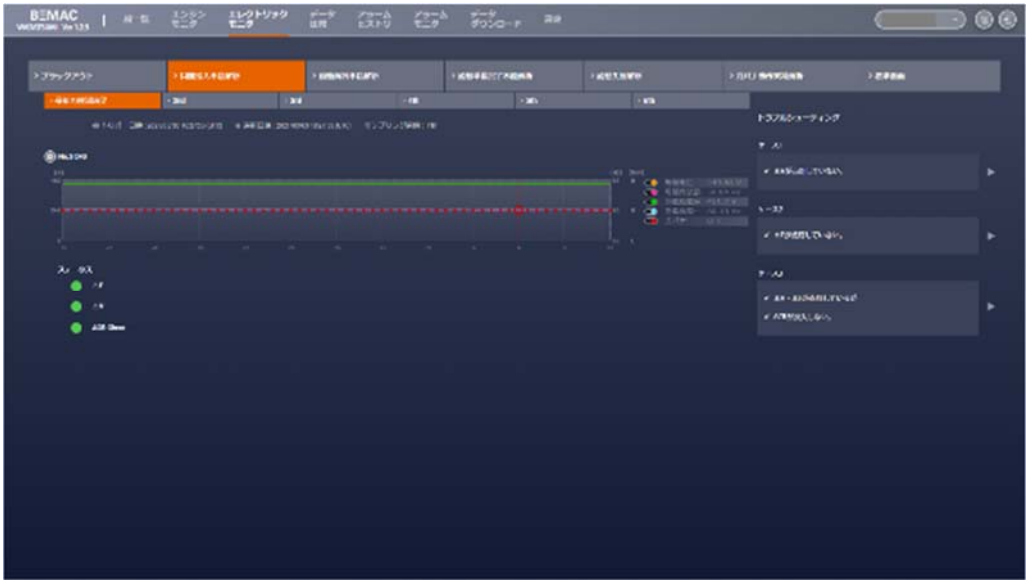


図 37 同期投入不良解析

図 37 に同期投入不良解析画面を記す。この画面では同期投入不良発生時の、該当する発電機の状態が表示される。基本機能はブラックアウト画面と同じであるがグラフのサンプリング周期は 1s である。

同期投入不良解析画面のグラフは母線電圧・周波数、発電機電圧・周波数、ガバナ状態が表示され、また△F・△V・ACB Close はランプ表示となっている。ランプ表示は、選択したグラフデータと連動してランプが点灯する仕様となっており、トラブルシューティングを参照する際にはグラフとこのランプ表示を見ることで、対処法を確認することができる。

3) 自動解列不良解析機能

図 38 自動解列不良解析

図 38 に自動解列不良解析機能画面を記す。この画面では自動解列不良発生時の状態が表示される。自動解列不良画面のグラフは発電機周波数・電力、ガバナ状態、ACB の状態が表示される。また、発電機 4 台のデータが表示され、解列不良発生時の各発電機の状態を同時に確認してトラブルシューティングを参照する仕様となっている。

4) 始動準備完了解析機能

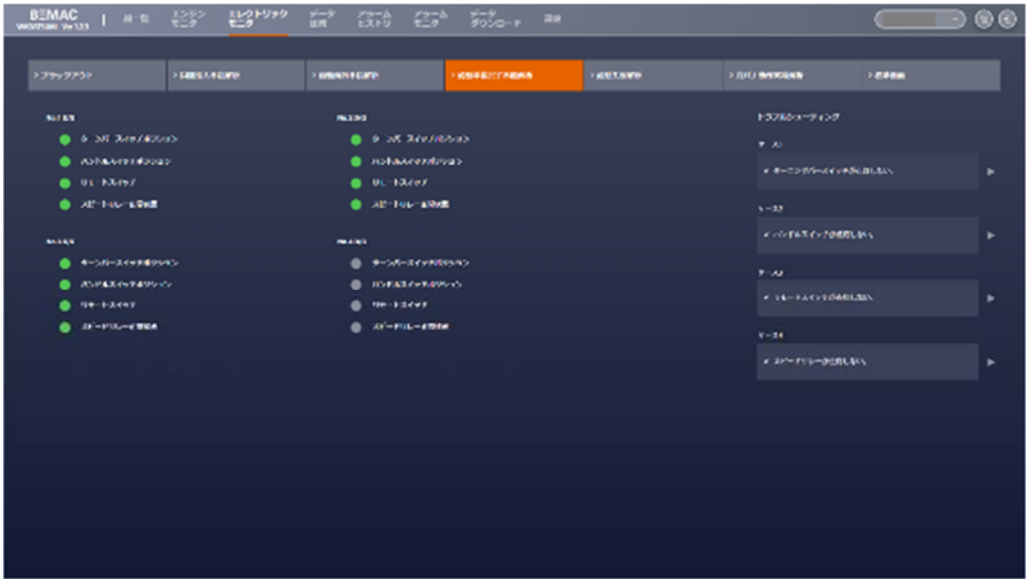


図 39 始動準備完了解析

図 39 に始動準備完了解析機能画面を記す。この画面は始動準備完了不能時に解析する画面である。この画面は他の画面とは構成が異なっており、グラフではなくランプ表示の画面でデータは 1 分周期で現在の状態を表す仕様とした。具体的な計測点は、ターニングバースイッチポジション、ハンドルスイッチポジション、リモートスイッチ、スピードリレー正常状態の 4 種類である。 始動準備完了できない際、この 4 種類の状態と右側のトラブルシューティングを見て解決方法を確認することができる仕組みとなっている。

5) 始動失敗解析機能

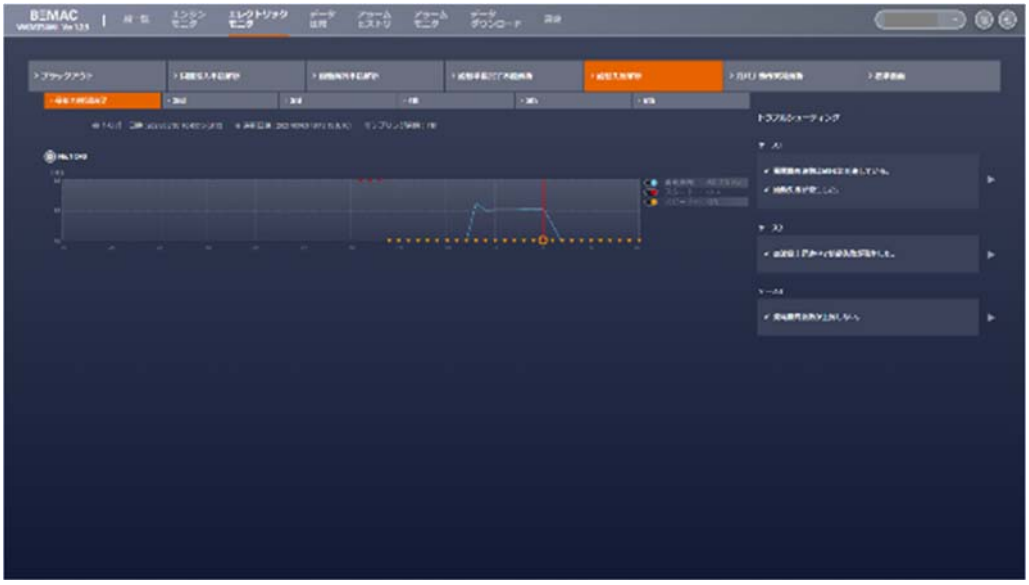


図 40 始動失敗解析

図 40 に始動失敗解析機能画面を記す。この画面では発電機始動失敗時の、該当する発電機の状態が表示される。グラフには発電機周波数とスタートバルブ状態、スピードリレー信号が表示される。

6) ガバナ動作不良解析

図 41 ガバナ動作不良

図 41 にガバナ動作不良解析機能画面を記す。この画面ではガバナ動作異常時の状態が表示される。この画面は過去 1 回分のみ、発電機 4 台のデータが表示される仕様となっている。

7) 標準画面

図 42 標準画面

図 42 に標準画面を記す。この画面では、各発電機の電圧・周波数・電流・電力の 1 分周期のデータが表示される。不具合解析などを行うアフターサービス員などの意見も踏まえ、現在の時刻から約 36 時間前までの発電機のデータが表示される仕様とした。もし、それ以前のデータを参照したい場合は旧来機能としてデータダウンロード機能が実装されておりそちらで参照可能となっている。カスタムでは、任意のチャンネルを軸に設定し、グラフの表示をすることができる。

3.6　データ収集・状態監視による効率化の検証

前項までの結果を踏まえ、データ収集、状態監視による効率化の検証を行った。

船上データを収集・活用するためのシステム構築を行うためには、従来は配電システムとデータロガーとの仕様すり合わせ、データロガーでの設定等が 8 時間程度必要であった。今回のシステムでは、データサーバーに配電データを可能な限り多く、直接出力しデータ構造として ISO 船上データサーバー規格に準拠するようにしたため上記の手間はゼロとなった。配電システム内で標準化されていないデータ項目があった場合に ISO 準拠の ID を別途割り付ける作業が 4 時間程度発生するが工数削減することができた。

アフターサービス部門のトラブル対応については、実際に配電システムに異常が発生し本船船員とやり取りを行った以下の実例について聞き取りを実施した。

1．発電機逆電力により No.3ACB 異常トリップした不具合

2．発電機電圧異常低下により NO.3ACB 異常トリップした不具合

不具合発生時、まず船員からの聞き取りを実施し原因究明の為、発電機コントローラ内のデータを収集するように要求した。1.の不具合では、必要部分の発電機電圧、電流、周波数、電力のイベントデータ 800 点との入出力情報データ 50 点の計 850 点を船員により

確認頂き表計算ソフトウェアに入力して頂いた。2.の不具合では母線電圧、周波数、発電機電圧、周波数の自動化ユニットの表示データを 10 分弱の動画データとして送付いただき、約 600 点をアフターサービス員が目視にて表計算ソフトウェアに入力した。

このデータ収集には目視による確認に 2 時間、書き写しや入力作業で 2 時間、他のデータ確認や連絡時間なども含めると 8 時間はかかると予想でき、当日の内にデータ解析が可能となることは不可能であった。入手したデータはアフターサービス員やユニット開発部門で整形しグラフ化するなどし、図 43 のようにデータ解析・異常判断を実施した。この作業も 4 時間はかかっており、原因推測し本船へ連絡するのに 2 日かかっていた。

状態監視アプリケーションの開発により上記項目は不要となり上記実例にある発電機逆電力、発電機電圧異常などはブラックアウト解析画面で表示されるようになった。船陸間通信の頻度を考慮しても、10 分以内に不具合データをグラフとして見ることができるようになった。結果、データ入手、解析にかかっていた 12 時間が 10 分となり、他の本船への連絡も 4 時間程度の半日もあれば十分となり工数削減できた。

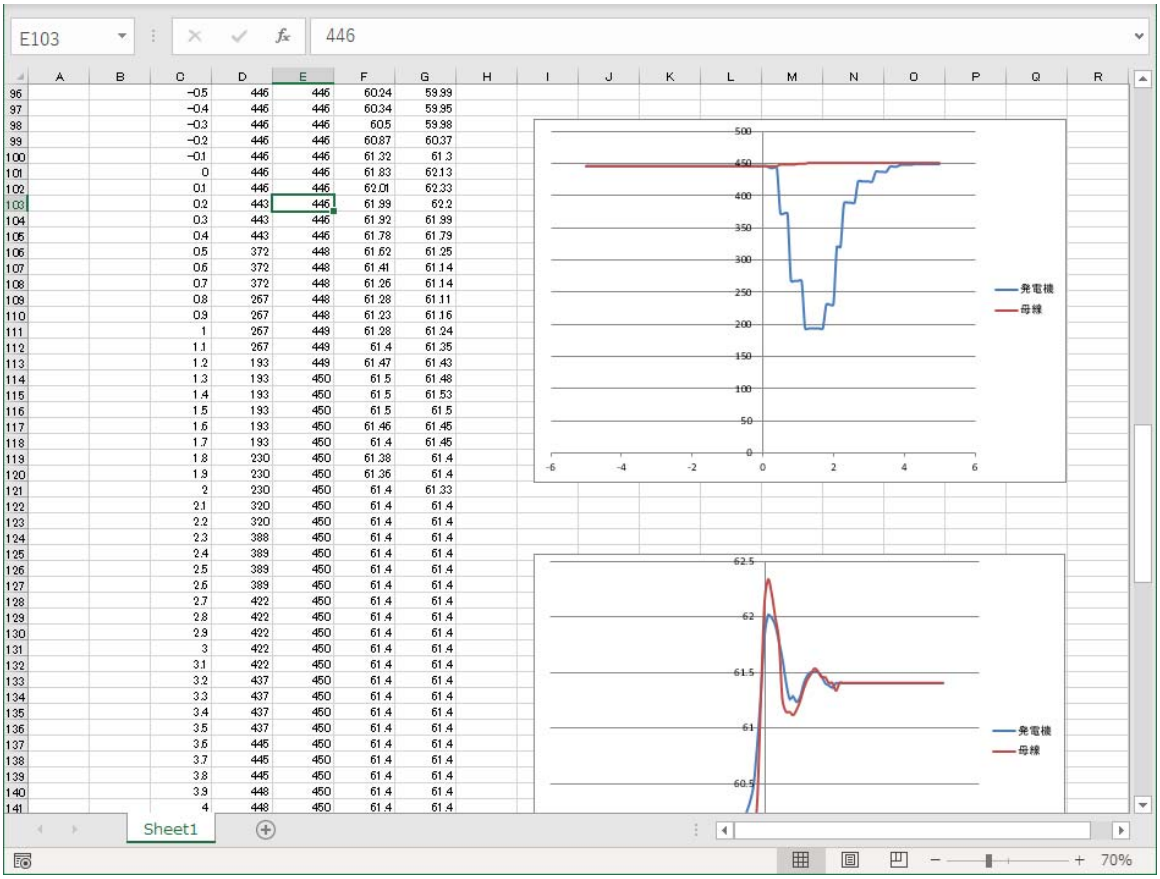


図 43 データ手入力による解析

４．目標の達成状況

4.1　2020 年度の目標の１）の達成状況

船上サーバー標準規格（IS019847/IS019848）に準拠した船上、陸上双方のデータ集積設備としてデータ出力ソフトウェアを開発した。

従来、システム構築を行うにあたって配電システムとデータロガーとの仕様すり合わせ、データロガーでの出力設定等に 8 時間程度の工数が必要であった。本ソフトウェアの開発により、特殊な標準化していないデータ項目があった場合は割り付ける作業が 4 時間程度発生する可能性が残るが、システム構築に係る工数を約 50%削減することが可能となり、目標にしていた 30%削減を大きく超える結果となった。

4.2　2020 年度の目標の２）の達成状況

状態監視アプリケーションの開発により、データ出力ソフトウェア及び状態監視アプリケーションを介して 10 分以内に不具合データを見ることが出来るようになり半日程度で原因を解析することが可能となった。トラブル発生時、アフターサービス部門では船員からの聞き取り、データ収集、解析を実施し原因推測をするのに 2 日弱かかっていた。結果、原因推測まで 2 日かかっていたのが半日で十分となりアフターサービス員の補修作業工数を約 75%削減できるスキームを確立することができた。目標にしていた 30%削減を二倍以上超える結果となった。

5．2021 年度の実施内容

配電システムの IoT 化による状態監視及び予防保全アプリケーションの技術開発における 2021 年度の実施内容を報告する。

5.1　予防保全アプリケーションの設計

船内で収集されたデータを基にどのようなアルゴリズムで船内状況を推定するかを検討し、機器の予防保全の支援を行うアプリケーションを設計した。

5.1.1　トラブルシューティング自動化機能

2020 年度に開発した電気系統における異常のトラブルシューティングを可能とする状態監視アプリケーションにおいて、目視判断となっている部分を自動判断に置き換えるべく設計を行った。設計にあたり、実施者がアフターサービス部門へヒアリングを行った。ヒアリングの結果、フローチャートによるトラブルシューティングの条件が複数に分岐していて、目視確認の手間が大きい以下の 4 機能を対象とすることとした。各機能について発生事象と確認すべきデータおよび検知する異常との関係をフローチャートの形で整理を行った。

1）ブラックアウト解析機能

ブラックアウト解析機能に関するフローチャートは図 44 に示す通り。

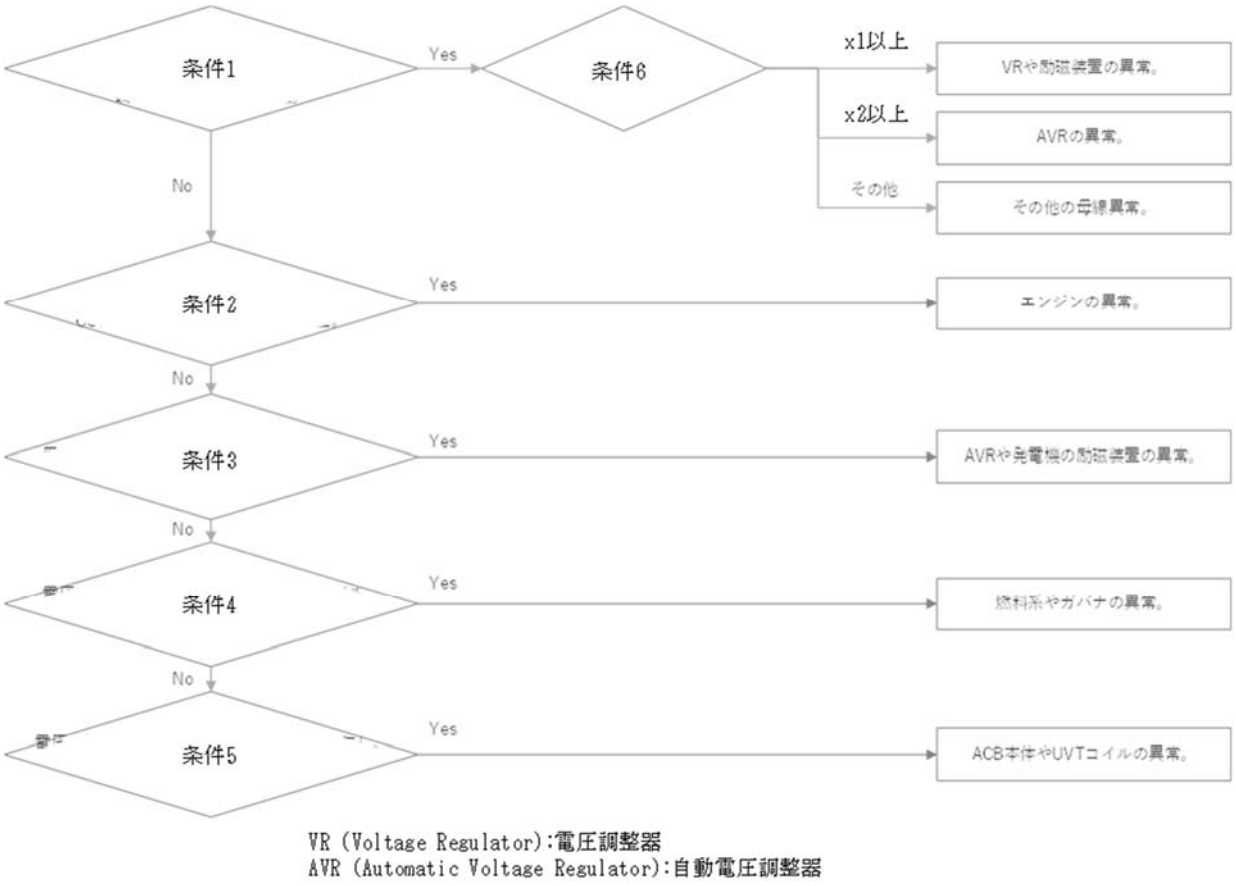


図 44　ブラックアウト解析機能に関するフローチャート

図 44 に示したフローチャートを基にトラブルシューティング自動化機能を設計し、意図的に発生させたデータで異常検知を行い、想定通りに対応する異常を検知することができた（図 45）。

図 45　ブラックアウト解析機能のトラブルシューティング自動化機能に関する評価結果
（赤丸は異常を検知した箇所を示す）

2）同期投入不良解析機能

同期投入不良解析機能に関するフローチャートは図 46 に示す通り。

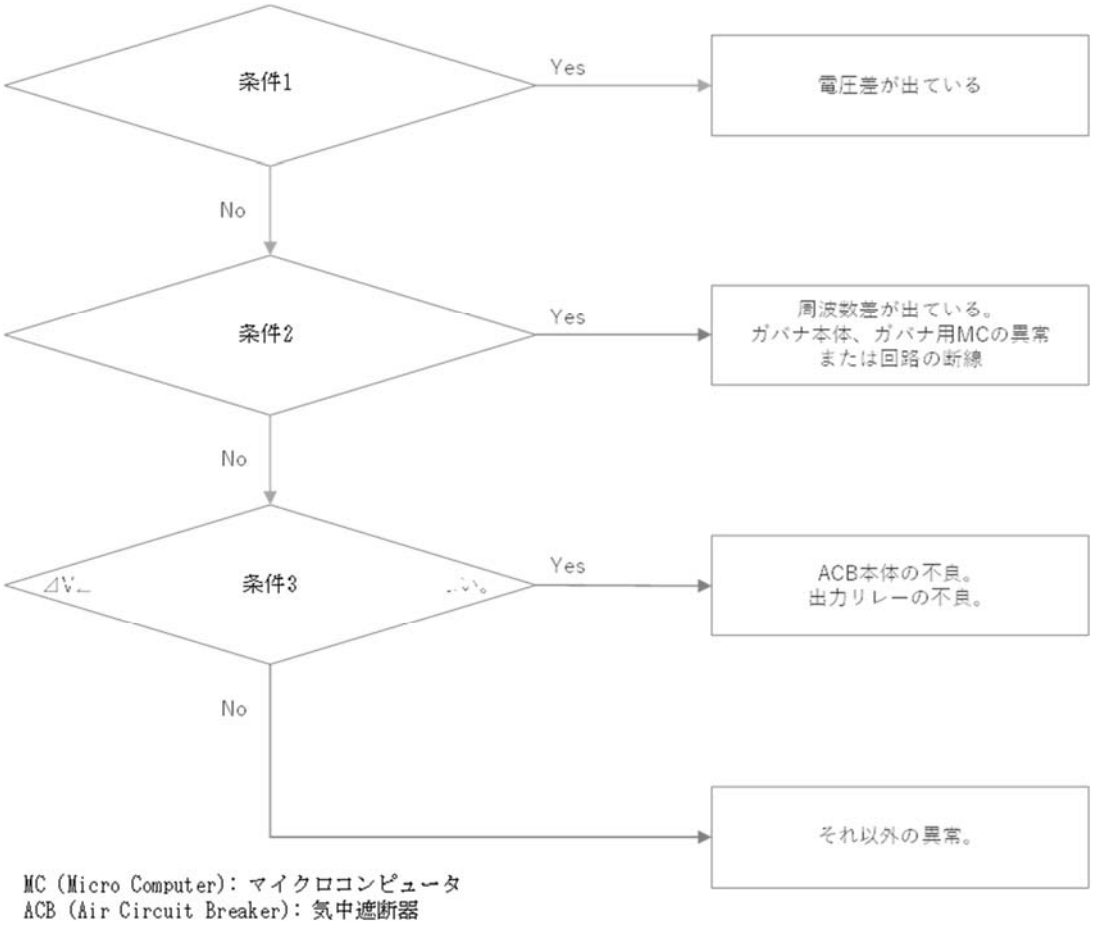


図 46　同期投入不良解析機能に関するフローチャート

図 46 に示したフローチャートを基にトラブルシューティング自動化機能を設計し、意図的に発生させたデータで異常検知を行い、想定通りに対応する異常を検知することができた（図 47）。

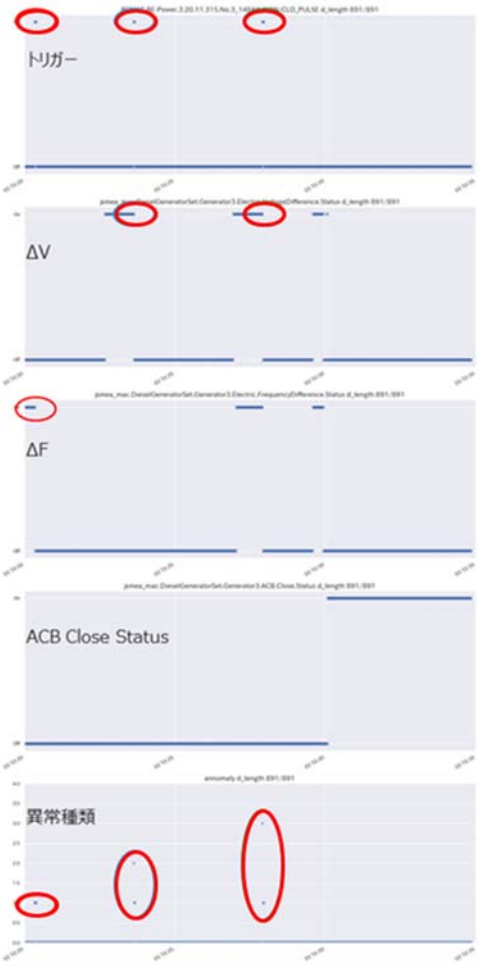


図 47　同期投入不良解析機能のトラブルシューティング自動化機能に関する評価結果
（赤丸は異常を検知した箇所を示す）

3）自動解列不良解析機能

自動解列不良解析機能に関するフローチャートは図 48 に示す通り。

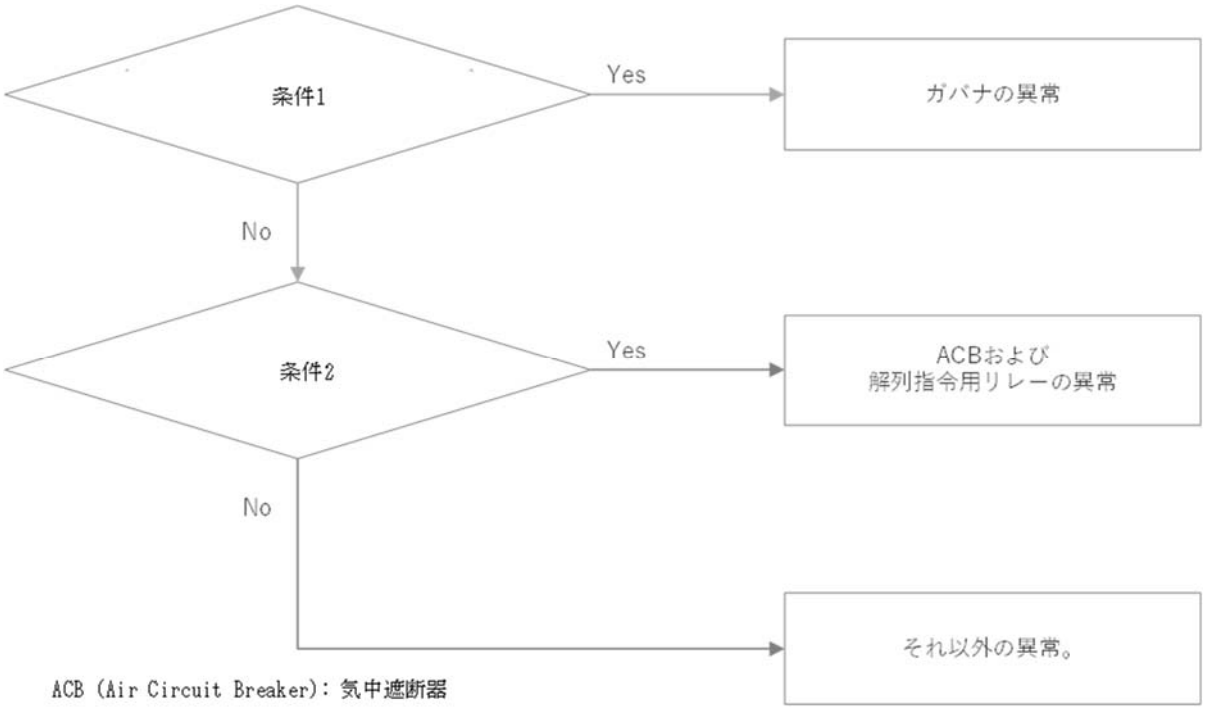


図 48　自動解列不良解析機能に関するフローチャート

図 48 に示したフローチャートを基にトラブルシューティング自動化機能を設計し、意図的に発生させたデータで異常検知を行い、想定通りに対応する異常を検知することができた（図 49）。

図 49　自動解列不良解析機能のトラブルシューティング自動化機能に関する評価結果
（赤丸は異常を検知した箇所を示す）

4）始動失敗解析機能

始動失敗解析機能に関するフローチャートは図 50 に示す通り。

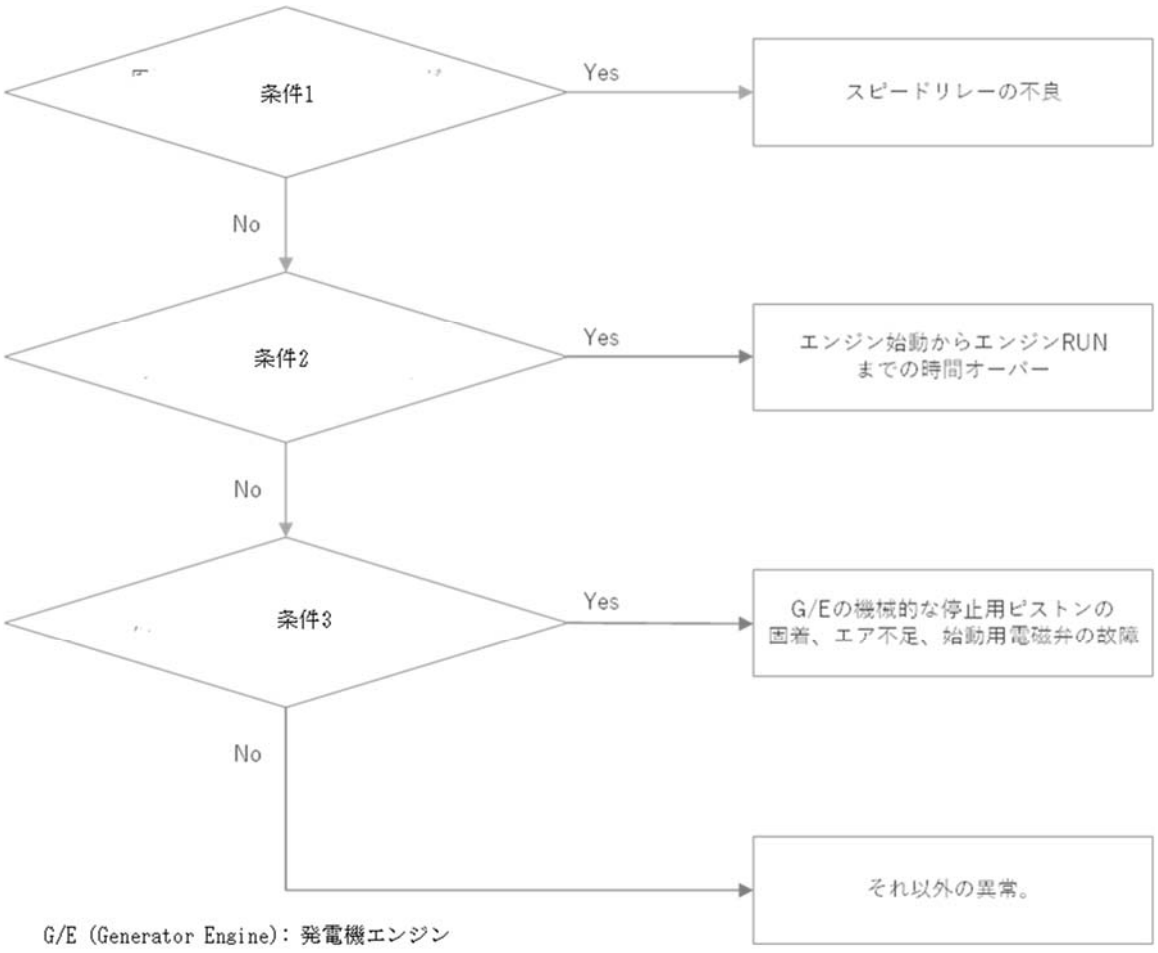


図 50　始動失敗解析機能に関するフローチャート

図 50 に示したフローチャートを基にトラブルシューティング自動化機能を設計し、意図的に発生させたデータで異常検知を行い、想定通りに対応する異常を検知することができた（図 51）。

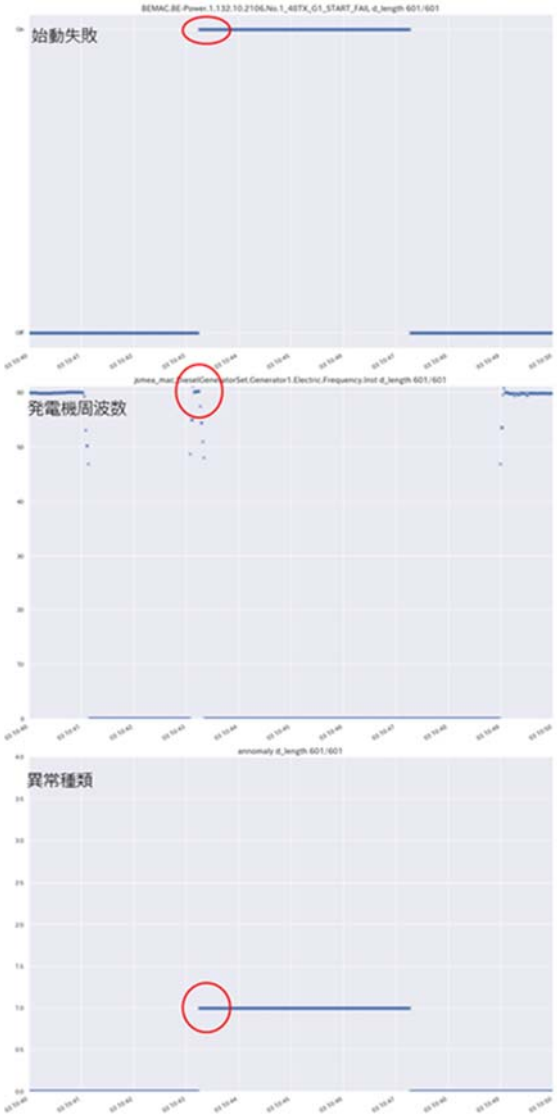


図 51　始動失敗解析機能のトラブルシューティング自動化機能に関する評価結果
（赤丸は異常を検知した箇所を示す）

5.1.2　異常検知トリガ自動化機能

異常が含まれているデータが蓄積されている場合、異常前後の事象に基づいて予兆検知を実現することが一般的である。しかしながら、船舶では定期的なメンテナンスが実施されているため、異常が含まれているデータが不足している。

2020 年度に開発した状態監視アプリケーションでは、手動設定されたトリガで電気系統における異常を検知している。しかしながら、人がこれまでに経験していない異常に関するトリガを手動設定することは困難である。そこで、手動設定となっているトリガを自動設定に置き換えるべく設計を行った。

船舶の電気系統における状態を機械学習でモデル化するアプローチを採用する。これにより、人が異常と認識していた事象だけでなく、人が異常と認識していない事象も把握することが可能となる。本機能で検知した異常を含むデータを蓄積し、そのデータの分析で得られた知見も蓄積することで、予兆検知の実現を将来的に目指す。

以下に本機能の詳細について記載する。

1）方法

複数の機械学習の手法を比較してみた結果、NGBoost (Natural Gradient Boosting for probabilistic perdition)による区間推定を採用した。理由としては、他の手法と比べて誤検知が少なく、検知精度とのバランスが優れていること、SHAP (SHapley Additive exPlanations)という機械学習のモデルによる予測を解釈する手法を併用することで、変数間の関係の調査が他の手法と比較して容易であることなどが挙げられる（図 52）。

分類	手法	特徴	検証結果
機械学習	MT法	変数間の相関を考慮した距離(マハラノビス距離)を用いて異常度を定義し、異常度が大きい場合に異常とみなす手法	
	相関のあるデータ同士の差分	ある変数Aと相関の強い変数を選んで各近傍との差分の平均、標準偏差を使って、異常を検知する方法	異常の閾値を手動で設定する必要があるため、不採用
	機械学習による予測値との差分	実測値と予測値のマハラノビス距離を計算し、異常度を定義	
	自然勾配ブースティングを用いた区間推定（NGBoost）：未来の予測	正規分布や対数正規分布を仮定して平均と分散を予測するモデルをつくり、実測値が予測値区間から外れているかで異常を判断	時系列分析では、LSTMの方が精度が良いため不採用
	深層学習による区間推定（LSTM）：未来の予測	過去の傾向を学習し、未来を予測することが得意であるLSTMを使用し、実測値が予測値区間から外れているかで異常を判断	一度採用したが、1期先の予測をする際に変動が大きく、方針を変更
	自然勾配ブースティングを用いた区間推定（NGBoost）：現在の予測	現在時点の複数の説明変数から、現在時点の目的変数を推定するモデルを構築し、実測値が予測値区間から外れているかで異常を判断	誤検知と精度のバランスが最もよく採用 SHAPを併用することで結果を解釈することが可能

図 52　手法の比較結果

現時点の複数の説明変数から、現時点の目的変数を推定するモデルを NGBoost で構築し、実測値が予測値の区間から外れている場合に異常と判定する手順を異常検知トリガ自動化機能として開発した（図 53）。

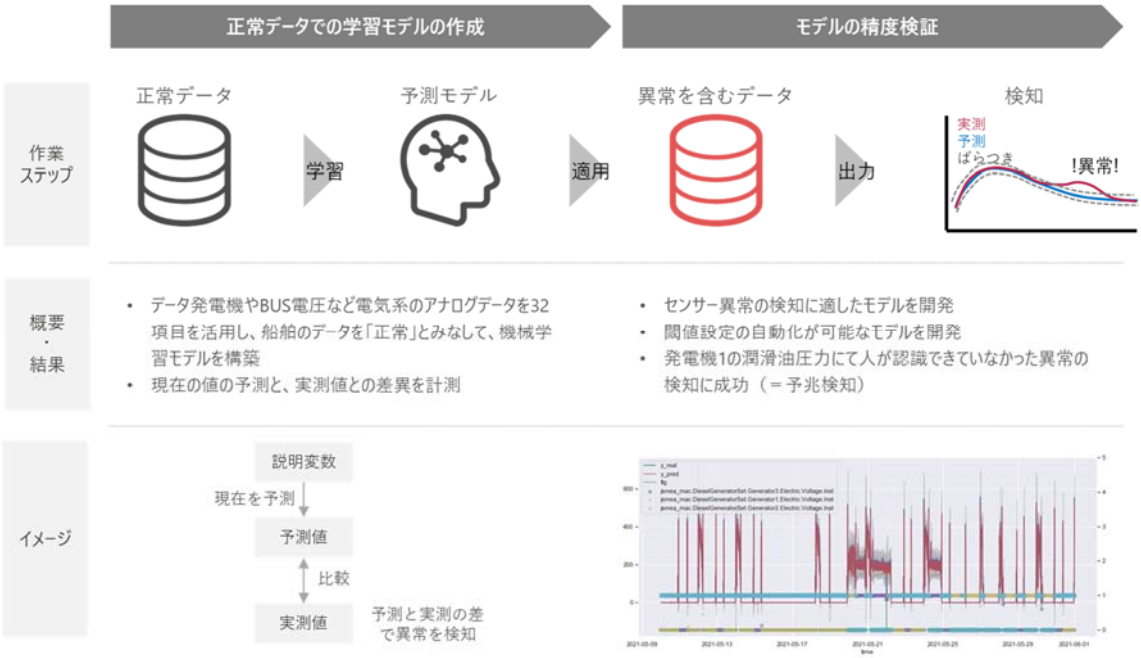


図 53　異常検知トリガ自動化機能の概念図

2）考慮点

全てのデータを利用すれば、機械学習で予測性能の高いモデルを構築することが理論的に可能であるが、そのためには膨大な計算リソースが必要となる。例えば、一般的な計算リソースの PC だと、学習時間が膨大になったり、メモリ不足が起こり得る。そのため、データにおける変数、計測期間、計測間隔などを調整しながら、一般的な計算リソースでも構築可能で、予測性能の高いモデルを構築していくことが必要となる。以下に機械学習でモデルを構築する上での考慮点を記載する。

- 変数

実施者のエンジニアリング部門とのヒアリングを基に発電機の電流、電圧、冷却水温度や On/Off のステータスなど船舶の電気系統を状態監視する上で重要である 41 の変数を選択した（表 16）。

表 16　変数一覧

#	ローカルID	共通チャンネル名	種類
1	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Current//Inst	NO.1 GENERATOR CURRENT	説明変数兼目的変数
2	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Frequency//Inst	NO.1 GENERATOR FREQUENCY	説明変数兼目的変数
3	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Power//Inst	NO.1 GENERATOR ELECTRIC POWER	説明変数兼目的変数
4	jsmea_mac/DieselGeneratorSet/Generator1/Electric//Voltage//Inst	NO.1 GENERATOR VOLTAGE	説明変数兼目的変数
5	jsmea_mac/DieselGeneratorSet/Generator2/Electric//Current//Inst	NO.2 GENERATOR CURRENT	説明変数兼目的変数
6	jsmea_mac/DieselGeneratorSet/Generator2/Electric//Frequency//Inst	NO.2 GENERATOR FREQUENCY	説明変数兼目的変数
7	jsmea_mac/DieselGeneratorSet/Generator2/Electric//Power//Inst	NO.2 GENERATOR ELECTRIC POWER	説明変数兼目的変数
8	jsmea_mac/DieselGeneratorSet/Generator2/Electric//Voltage//Inst	NO.2 GENERATOR VOLTAGE	説明変数兼目的変数
9	jsmea_mac/DieselGeneratorSet/Generator3/Electric//Current//Inst	NO.3 GENERATOR CURRENT	説明変数兼目的変数
10	jsmea_mac/DieselGeneratorSet/Generator3/Electric//Frequency//Inst	NO.3 GENERATOR FREQUENCY	説明変数兼目的変数
11	jsmea_mac/DieselGeneratorSet/Generator3/Electric//Power//Inst	NO.3 GENERATOR ELECTRIC POWER	説明変数兼目的変数
12	jsmea_mac/DieselGeneratorSet/Generator3/Electric//Voltage//Inst	NO.3 GENERATOR VOLTAGE	説明変数兼目的変数
13	jsmea_mac/DieselGeneratorSet1/AirCooler/CoolingFreshWater/Inlet/Press/common/Inst	NO.1 G/E COOLING FRESH WATER IN P	説明変数兼目的変数
14	jsmea_mac/DieselGeneratorSet1/CylinderCommon/CoolingFreshWater/Outlet/Temp//Inst	NO.1 G/E COOLING FRESH WATER COMMON OUT T	説明変数兼目的変数
15	jsmea_mac/DieselGeneratorSet1/Engine/LubOil/Inlet/Press//Inst	NO.1 G/E LO IN P	説明変数兼目的変数
16	jsmea_mac/DieselGeneratorSet1/Engine/LubOil/Inlet/Temp//Inst	NO.1 G/E LO IN T	説明変数兼目的変数
17	jsmea_mac/DieselGeneratorSet1/StartAirLine/Air/Inlet/Press//Inst	NO.1 G/E STARTING AIR IN P	説明変数兼目的変数
18	jsmea_mac/DieselGeneratorSet1/TurboChargerUpper/ExhaustGas/Inlet/Temp//Inst	NO.1 G/E EXH GAS T/C IN (Upper) T	説明変数兼目的変数
19	jsmea_mac/DieselGeneratorSet2/AirCooler/CoolingFreshWater/Inlet/Press/common/Inst	NO.2 G/E COOLING FRESH WATER IN P	説明変数兼目的変数
20	jsmea_mac/DieselGeneratorSet2/CylinderCommon/CoolingFreshWater/Outlet/Temp//Inst	NO.2 G/E COOLING FRESH WATER COMMON OUT T	説明変数兼目的変数
21	jsmea_mac/DieselGeneratorSet2/Engine/LubOil/Inlet/Press//Inst	NO.2 G/E LO IN P	説明変数兼目的変数
22	jsmea_mac/DieselGeneratorSet2/Engine/LubOil/Inlet/Temp//Inst	NO.2 G/E LO IN T	説明変数兼目的変数
23	jsmea_mac/DieselGeneratorSet2/StartAirLine/Air/Inlet/Press//Inst	NO.2 G/E STARTING AIR IN P	説明変数兼目的変数
24	jsmea_mac/DieselGeneratorSet2/TurboChargerUpper/ExhaustGas/Inlet/Temp//Inst	NO.2 G/E EXH GAS T/C IN (Upper) T	説明変数兼目的変数
25	jsmea_mac/DieselGeneratorSet3/AirCooler/CoolingFreshWater/Inlet/Press/common/Inst	NO.3 G/E COOLING FRESH WATER IN P	説明変数兼目的変数
26	jsmea_mac/DieselGeneratorSet3/CylinderCommon/CoolingFreshWater/Outlet/Temp//Inst	NO.3 G/E COOLING FRESH WATER COMMON OUT T	説明変数兼目的変数
27	jsmea_mac/DieselGeneratorSet3/Engine/LubOil/Inlet/Press//Inst	NO.3 G/E LO IN P	説明変数兼目的変数
28	jsmea_mac/DieselGeneratorSet3/Engine/LubOil/Inlet/Temp//Inst	NO.3 G/E LO IN T	説明変数兼目的変数
29	jsmea_mac/DieselGeneratorSet3/StartAirLine/Air/Inlet/Press//Inst	NO.3 G/E STARTING AIR IN P	説明変数兼目的変数
30	jsmea_mac/DieselGeneratorSet3/TurboChargerUpper/ExhaustGas/Inlet/Temp//Inst	NO.3 G/E EXH GAS T/C IN (Upper) T	説明変数兼目的変数
31	jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusFrequency//Frequency//Inst	BUS FREQUENCY	説明変数兼目的変数
32	jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusVoltage//Voltage//Inst	BUS VOLTAGE	説明変数兼目的変数
33	jsmea_mac/DieselGeneratorSet/Generator1/ACB///Close/Status	NO.1 GENERATOR ACB CLOSE	説明変数
34	jsmea_mac/DieselGeneratorSet/Generator2/ACB///Close/Status	NO.2 GENERATOR ACB CLOSE	説明変数
35	jsmea_mac/DieselGeneratorSet/Generator3/ACB///Close/Status	NO.3 GENERATOR ACB CLOSE	説明変数
36	jsmea_mac/DieselGeneratorSet/Generator1/StartValve///ON/Status	NO.1 GENERATOR START VALVE	説明変数
37	jsmea_mac/DieselGeneratorSet/Generator2/StartValve///ON/Status	NO.2 GENERATOR START VALVE	説明変数
38	jsmea_mac/DieselGeneratorSet/Generator3/StartValve///ON/Status	NO.3 GENERATOR START VALVE	説明変数
39	jsmea_mac/DieselGeneratorSet/Generator1/SpeedRelay///ON/Status	NO.1 GENERATOR SPEED RELAY LOW SIGNAL	説明変数
40	jsmea_mac/DieselGeneratorSet/Generator2/SpeedRelay///ON/Status	NO.2 GENERATOR SPEED RELAY LOW SIGNAL	説明変数
41	jsmea_mac/DieselGeneratorSet/Generator3/SpeedRelay///ON/Status	NO.2 GENERATOR SPEED RELAY LOW SIGNAL	説明変数

- 計測期間と間隔

機械学習で予測性能の良いモデルを構築するためには、データにパターンが網羅的に含まれている必要がある。対象船には 3 台の発電機が搭載されており、発電機の On/Off が切り替わるタイミングで計測値が変化する特徴があった（図 54）。そこで、3 台の発電機の On/Off のパターンを網羅的に含む期間である 2021 年 3 月 20 日から 2021 年 4 月 13 日のデータでモデルを構築した。また、機械学習はデータの量が多くなると学習時間が多くなる傾向にある。対象船のデータは 1 秒間隔であるため、10 秒間隔にサンプリングすることで、学習時間を約 10 時間に抑えながら、予測性能の高いモデルを構築することができた。

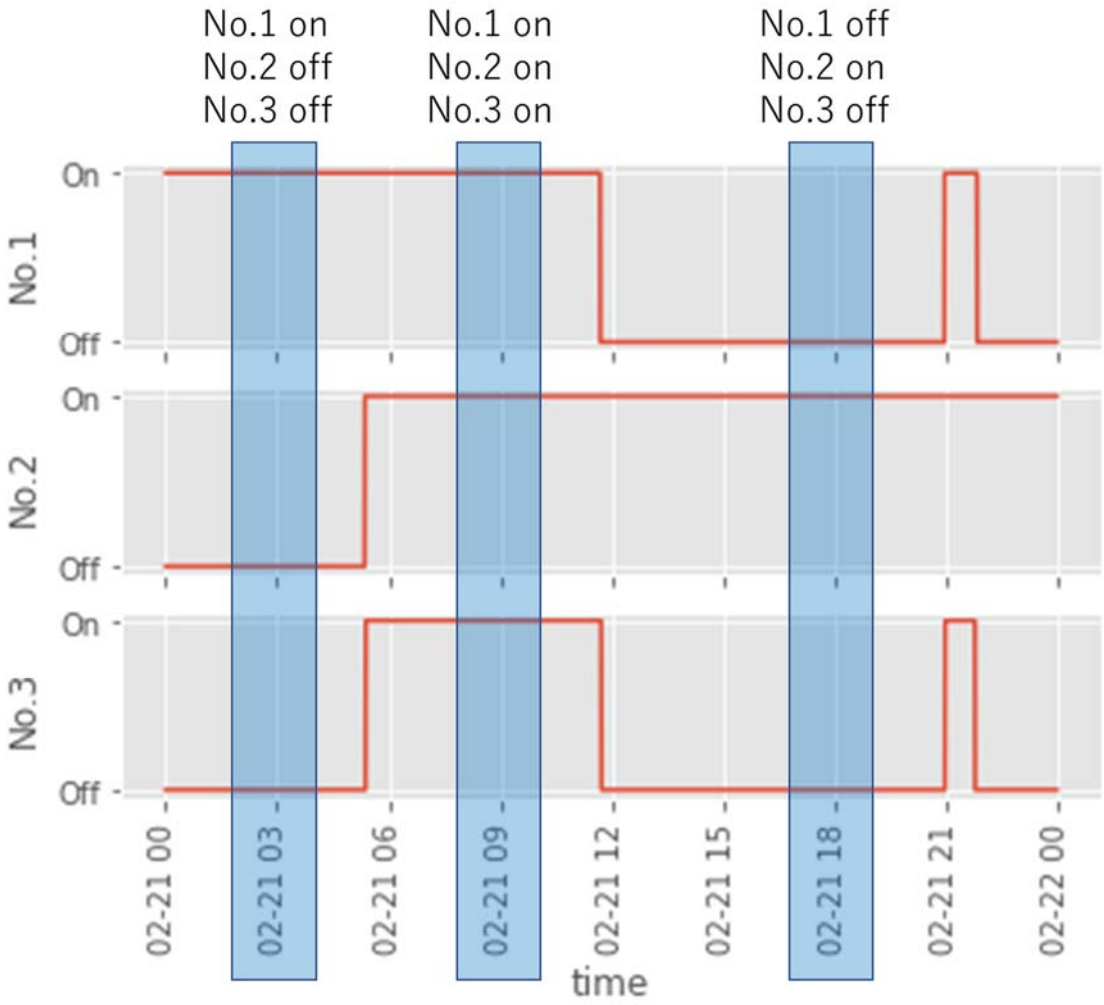


図 54　3 台の発電機の On/Off の関係の一例

2020 年度に開発した状態監視アプリケーションにおいてフローチャートを基にトラブルシューティングを自動化する「トラブルシューティング自動化機能」を検討・設計した。各トラブルに対応する発生事象と確認すべきデータ及び検知する異常との関係をフローチャートの形で整理を行った。更に船内で収集されたデータに機械学習の手法を利用して得た理論値と実測値の差異から異常検知のトリガ設定を自動化する「異常検知トリガ設定自動化機能」を検討・設計した。複数の機械学習手法を検討した結果、NGboost による区間推定を採用した。これにより、人が異常と認識していた事象だけでなく、人が異常と認識していない事象も把握することが可能になった。これらの機能を 2020 年度開発した状態監視アプリケーションに追加することで、機器の予防保全の支援を行うアプリケーションを設計した。

5.2　予防保全アプリケーションの開発

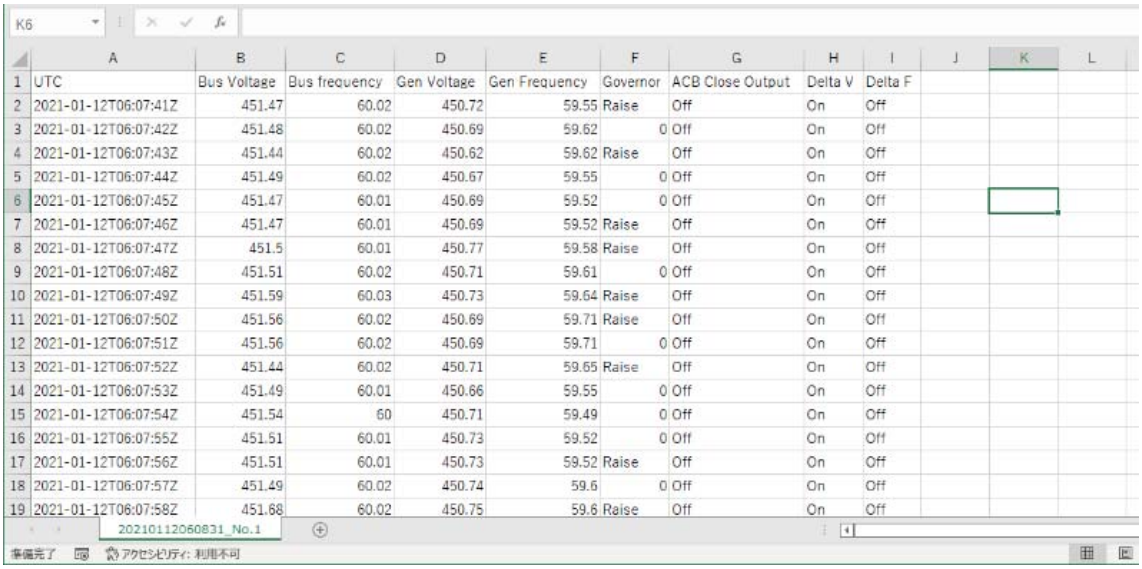
5.1 にて開発したトラブルシューティング自動化機能と異常検知トリガ設定自動化機能を状態監視アプリケーションで使用できるようにするため IoT データサーバーへの組み込みを実施した。以下に組み込み機能別に詳細を記載する。

5.2.1　IoT データサーバーでの必要データの抽出機能の設計・開発

組み込み動作の為には、まず開発したトラブルシューティング自動化機能と異常検知トリガ設定自動化機能に必要なデータを抽出する必要がある。データの抽出は新たに機能を開発するのではなく、2020 年度に開発したデータを抽出するデータ監視ソフトウェアを改良することで実施した。データ監視ソフトウェアでは MQTT プロトコルによりトリガ条件を周期的に監視しており、条件データに変更があれば ISO19848 Time Series Data を HTTP で必要な間隔だけデータベースから抽出・取得している。今回は必要な間隔のデータは既存の状態監視アプリケーションと同一期間で十分と判断し、条件データ更新タイミングの前 50 秒、後ろ 10 秒と変更は実施していない。

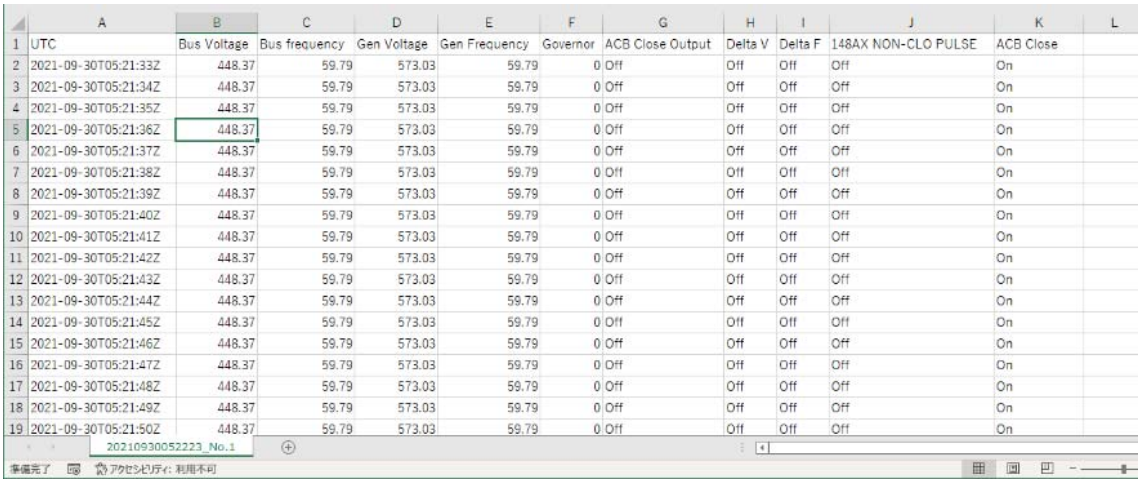
抽出したデータは従来から実装されている状態監視アプリケーション用 CSV ファイルの列を追加する形で生成される。生成された CSV ファイルを開発したトラブルシューティング自動化機能と異常検知トリガ設定自動化機能で参照することにより自動解析機能が IoT サーバーで動作可能となった。

図 55 および図 56 に従来の CSV ファイルと改良した CSV ファイルのサンプルを記載する。例として自動同期投入不良診断では NON-CLO PULSE および ACB Close 信号がトラブルシューティング自動解析機能に追加で必要なため、J,K 列に追加した。



	A	B	C	D	E	F	G	H	I
1	UTC	Bus Voltage	Bus frequency	Gen Voltage	Gen Frequency	Governor	ACB Close Output	Delta V	Delta F
2	2021-01-12T06:07:41Z	451.47	60.02	450.72	59.55	Raise	Off	On	Off
3	2021-01-12T06:07:42Z	451.48	60.02	450.69	59.62	0	Off	On	Off
4	2021-01-12T06:07:43Z	451.44	60.02	450.62	59.62	Raise	Off	On	Off
5	2021-01-12T06:07:44Z	451.49	60.02	450.67	59.55	0	Off	On	Off
6	2021-01-12T06:07:45Z	451.47	60.01	450.69	59.52	0	Off	On	Off
7	2021-01-12T06:07:46Z	451.47	60.01	450.69	59.52	Raise	Off	On	Off
8	2021-01-12T06:07:47Z	451.5	60.01	450.77	59.58	Raise	Off	On	Off
9	2021-01-12T06:07:48Z	451.51	60.02	450.71	59.61	0	Off	On	Off
10	2021-01-12T06:07:49Z	451.59	60.03	450.73	59.64	Raise	Off	On	Off
11	2021-01-12T06:07:50Z	451.56	60.02	450.69	59.71	Raise	Off	On	Off
12	2021-01-12T06:07:51Z	451.56	60.02	450.69	59.71	0	Off	On	Off
13	2021-01-12T06:07:52Z	451.44	60.02	450.71	59.65	Raise	Off	On	Off
14	2021-01-12T06:07:53Z	451.49	60.01	450.66	59.55	0	Off	On	Off
15	2021-01-12T06:07:54Z	451.54	60	450.71	59.49	0	Off	On	Off
16	2021-01-12T06:07:55Z	451.51	60.01	450.73	59.52	0	Off	On	Off
17	2021-01-12T06:07:56Z	451.51	60.01	450.73	59.52	Raise	Off	On	Off
18	2021-01-12T06:07:57Z	451.49	60.02	450.74	59.6	0	Off	On	Off
19	2021-01-12T06:07:58Z	451.68	60.02	450.75	59.6	Raise	Off	On	Off

図 55　自動同期投入不良診断 CSV データ（従来）



	A	B	C	D	E	F	G	H	I	J	K	L
1	UTC	Bus Voltage	Bus frequency	Gen Voltage	Gen Frequency	Governor	ACB Close Output	Delta V	Delta F	148AX NON-CLO PULSE	ACB Close	
2	2021-09-30T05:21:33Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
3	2021-09-30T05:21:34Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
4	2021-09-30T05:21:35Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
5	2021-09-30T05:21:36Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
6	2021-09-30T05:21:37Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
7	2021-09-30T05:21:38Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
8	2021-09-30T05:21:39Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
9	2021-09-30T05:21:40Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
10	2021-09-30T05:21:41Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
11	2021-09-30T05:21:42Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
12	2021-09-30T05:21:43Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
13	2021-09-30T05:21:44Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
14	2021-09-30T05:21:45Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
15	2021-09-30T05:21:46Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
16	2021-09-30T05:21:47Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
17	2021-09-30T05:21:48Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
18	2021-09-30T05:21:49Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	
19	2021-09-30T05:21:50Z	448.37	59.79	573.03	59.79	0	Off	Off	Off	Off	On	

20210930052223_No.1

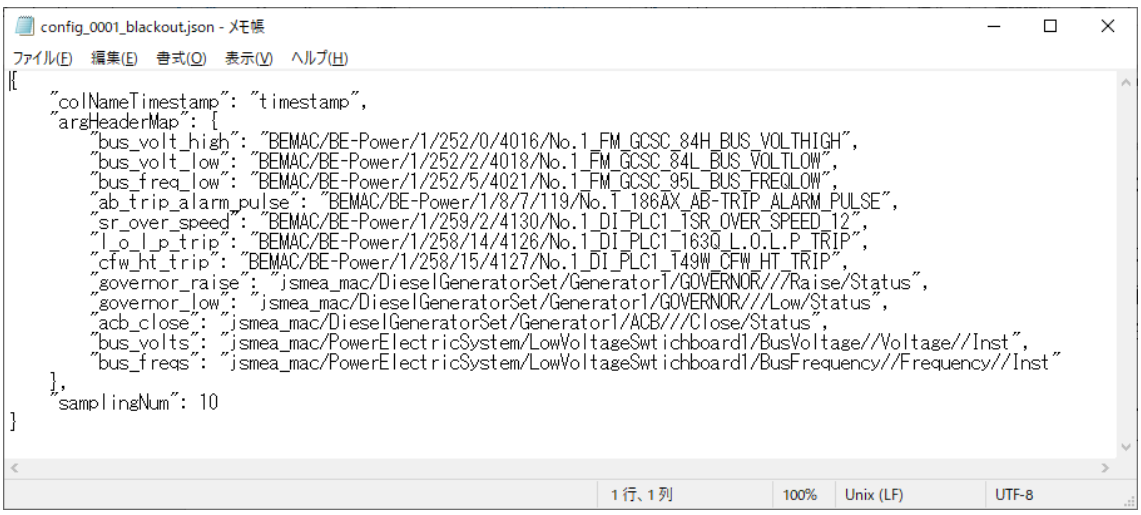
準備完了　アクセシビリティ: 利用不可

図 56　自動同期投入不良診断 CSV データ（改良）

5.2.2　IoT データサーバーへの自動化機能の組み込み

5.2.1 で生成した CSV ファイルでトラブルシューティング自動化機能と異常検知トリガ設定自動化機能が動作するように設計、開発を実施した。

CSV ファイルで動作するようにするためには、ソフトウェア内で必要なデータが何処に保存されているかを適切に設定する必要がある。ソフトウェア内に直接指定した場合、今回実施したように CSV ファイルの項目追加や名称変更などが発生すると、都度ソフトウェアコードの修正が必要になる可能性がある。その手間を省くため項目設定ファイルを設け、それを参照する形で柔軟に参照データの紐づけを変更できるように設計を実施した。項目設定は軽量なテキストベースのデータ交換用フォーマットである JSON 形式を使用している。図 57 に項目設定ファイルの例を示す。



config_0001_blackout.json - メモ帳

ファイル(F)　編集(E)　書式(O)　表示(V)　ヘルプ(H)

```
{
    "colNameTimestamp": "timestamp",
    "argHeaderMap": {
        "bus_volt_high": "BEMAC/BE-Power/1/252/0/4016/No.1_FM_GCSC_84H_BUS_VOLTHIGH",
        "bus_volt_low": "BEMAC/BE-Power/1/252/2/4018/No.1_FM_GCSC_84L_BUS_VOLTLOW",
        "bus_freq_low": "BEMAC/BE-Power/1/252/5/4021/No.1_FM_GCSC_95L_BUS_FREQLOW",
        "ab_trip_alarm_pulse": "BEMAC/BE-Power/1/8/7/119/No.1_186AX_AB-TRIP_ALARM_PULSE",
        "sr_over_speed": "BEMAC/BE-Power/1/259/2/4130/No.1_DI_PLC1_1SR_OVER_SPEED_12",
        "l_o_l_p_trip": "BEMAC/BE-Power/1/258/14/4126/No.1_DI_PLC1_163Q_L.O.L.P_TRIP",
        "cfw_ht_trip": "BEMAC/BE-Power/1/258/15/4127/No.1_DI_PLC1_149W_CFW_HT_TRIP",
        "governor_raise": "jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Raise/Status",
        "governor_low": "jsmea_mac/DieselGeneratorSet/Generator1/GOVERNOR///Low/Status",
        "acb_close": "jsmea_mac/DieselGeneratorSet/Generator1/ACB///Close/Status",
        "bus_volts": "jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusVoltage//Voltage//Inst",
        "bus_freqs": "jsmea_mac/PowerElectricSystem/LowVoltageSwtichboard1/BusFrequency//Frequency//Inst"
    },
    "samplingNum": 10
}
```

1 行、1 列　100%　Unix (LF)　UTF-8

図 57　項目設定ファイル例

上記項目設定ファイルと共に、トラブルシューティング自動化機能と異常検知トリガ設定自動化機能をカスタマイズして IoT データサーバー内に実装した。

5.2.3　状態監視アプリケーションへの自動解析結果の受け渡し

5.2.2 で設計・開発されたソフトウェアにより、自動解析結果を導き出すことができるようになった。次に自動解析結果を状態監視アプリケーションへ受け渡す必要があるためその機能の設計を実施した。

受け渡しには、アプリケーション間のデータ交換用として適しているため前項と同じ JSON ファイルを用いることにした。JSON ファイルは状態監視アプリケーションから参照や対比がしやすいように IoT データサーバー内の CSV ファイルと同じ場所に保存されるようにした。図 58 に受け渡しファイルのサンプルを示す。ファイル内はトラブルシューティング自動化の結果番号が記載されているのみのシンプルなものとなっている。

図 58　自動解析結果受け渡しファイル

5.2.4　状態監視アプリケーションの自動化対応設計

5.2.3 までで状態監視アプリケーションのトラブルシューティング自動化機能と異常検知トリガ設定自動化機能を追加するための解析の実施、結果の出力及び受け渡しの設計、開発を完了した。本項では解析結果を状態監視アプリケーションに表示させる方法を設計し、開発を実施した。なお本アプリケーションは 2020 年度に引き続き協力者である FutureRays 株式会社と共に設計・開発を実施した。

2020 年度に開発した状態監視アプリケーションは図 59 の通り、表示グラフや状態ランプをもとに、右側のトラブルシューティングより推定不具合要因を手動で見つけ出す方式を取っていた。

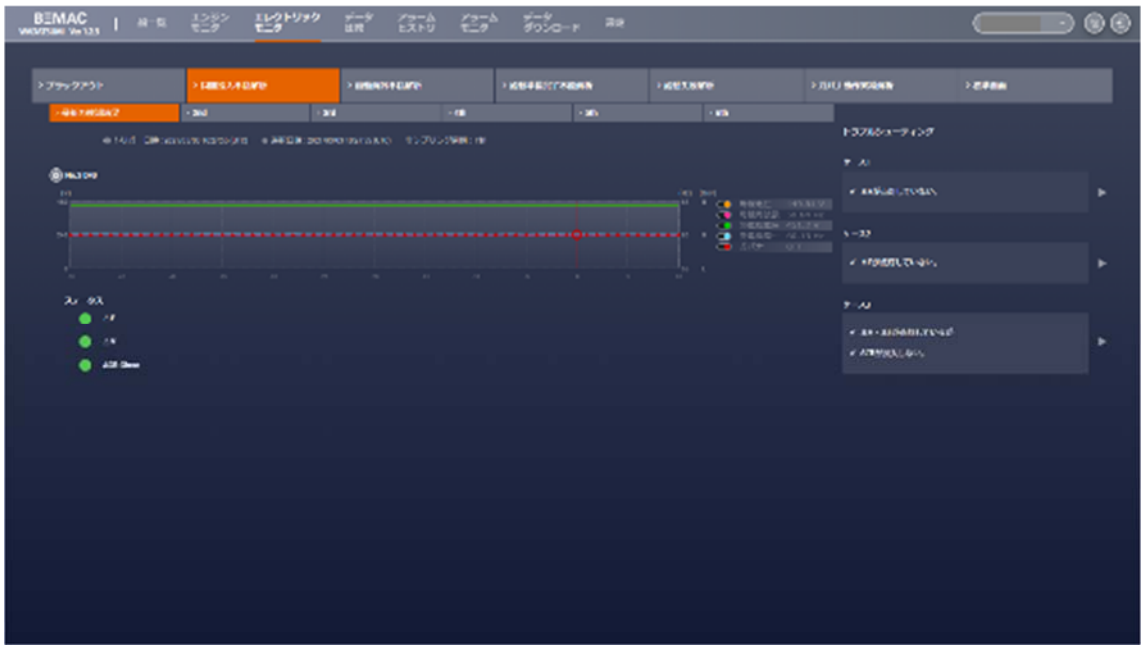


図 59　状態監視アプリケーション（同期投入不良解析 従来）

トラブルシューティングを自動化する方法として、当初は単純に判定した結果のトラブルシューティング部分の色を変更して表示させる方向で検討していたが、トラブルシュートおよび不具合解決方法の提示をより分かりやすいようにするべきと考え、表示方法の再構築を実施することにした。表示方法の再検討の結果、図 60 のような表示とすることにした。

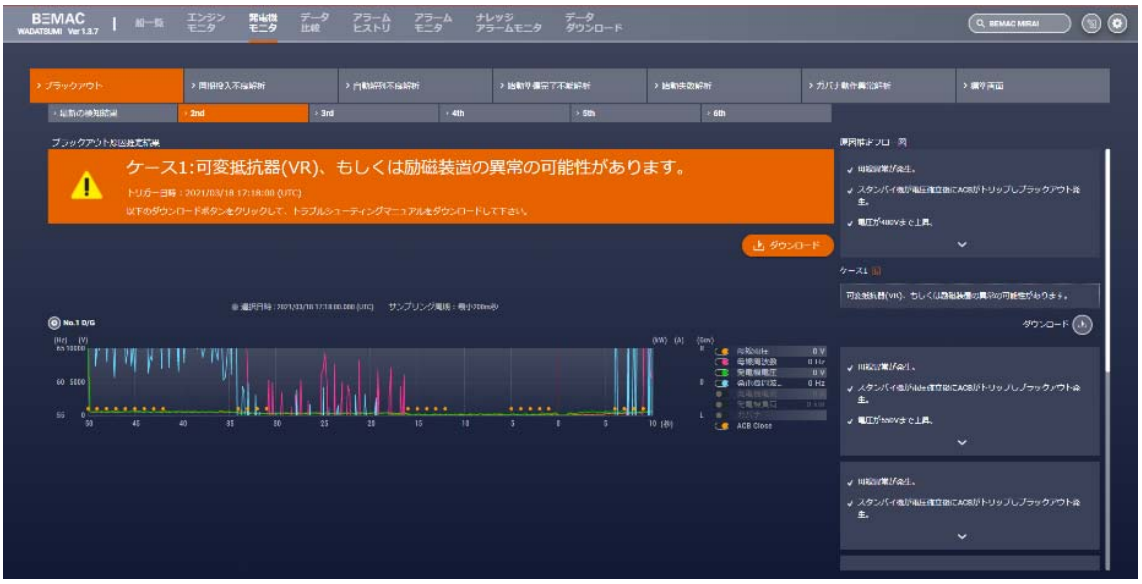


図 60　状態監視アプリケーション（トラブルシューティング自動化改良）

表示としては以下の変更を実施した。

- 自動化による推定原因結果を画面の一番見やすい先頭位置に配置するとともに、

注意を引く色にした。

- 推定原因結果の内容は、従来からあるトラブルシューティングから抜きだしたものだが、より直感的に判断できるようにするため文言の見直しを実施した。
- トラブル発生時の対処法のマニュアルダウンロードについては、従来トラブルシューティングから選択していく方法で、到達するまでに数回の操作が必要となっていたものを、原因結果の下にダウンロードボタンを表示させることによりすぐにアクセス可能となった。また、本画面は通常状態ではあまり操作されず、不具合発生時にのみ操作するため船員によっては操作方法を十分把握していない可能性があると想定されるので、画面内にマニュアルダウンロードに誘導する説明文を加えた。
- 推定原因の誤検知や想定外の要因の為に自動化機能が正常に動作しない可能性も鑑み、従来のトラブルシューティングおよびグラフ表示はそのままの状態で残すことにした。
- 従来は最上部のタブを選択した時、無条件でブラックアウト画面の最新の検知結果が表示されていたが、どの不具合が発生したかすぐわかるように、一番直近に発生した状態の画面に自動的に遷移するアルゴリズムを取り入れた。

始動準備完了解析機能については要因が意図的に実施したものか不具合なのかがデータのみでは判定できない状態の為、自動化機能の実装は実施していない。

図 61 から図 66 に改良した各解析機能画面を例示する。

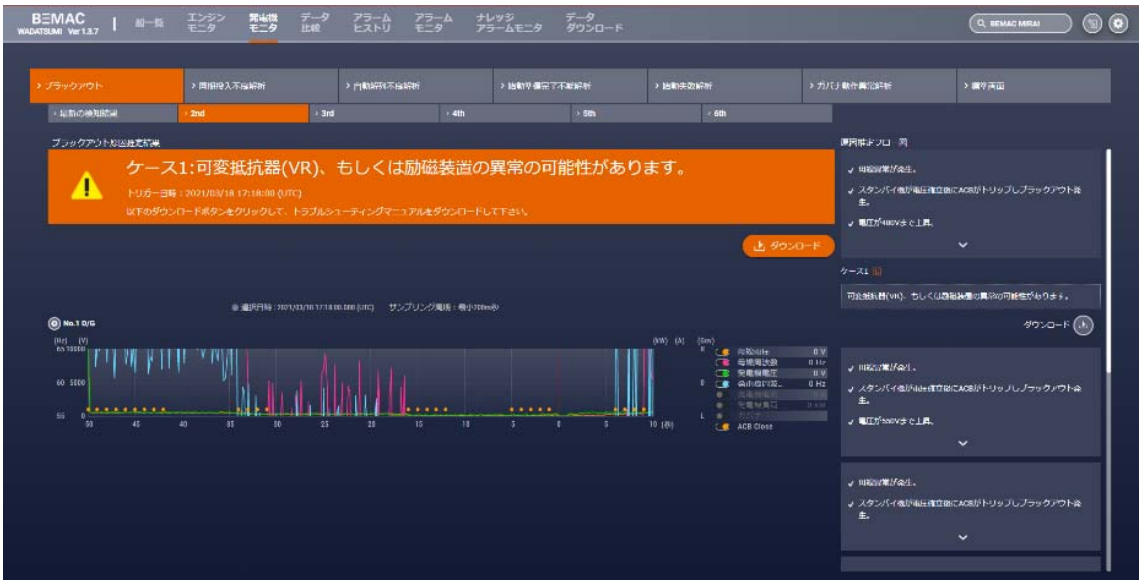


図 61　ブラックアウト解析（不具合要因ケース 1 自動判定）

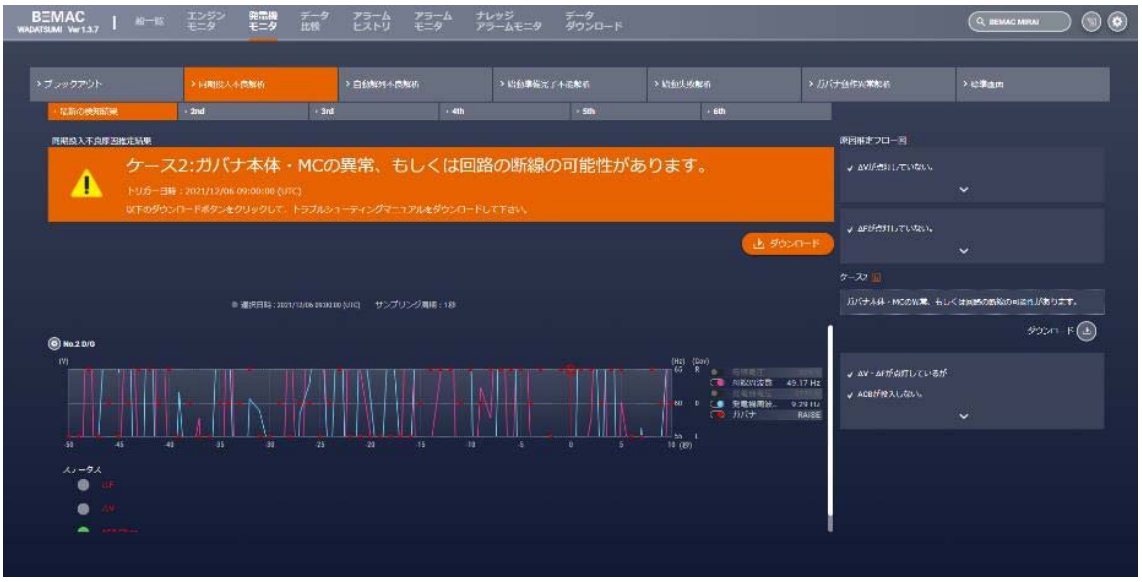


図 62　同期投入不良解析（不具合要因ケース 2 自動判定）

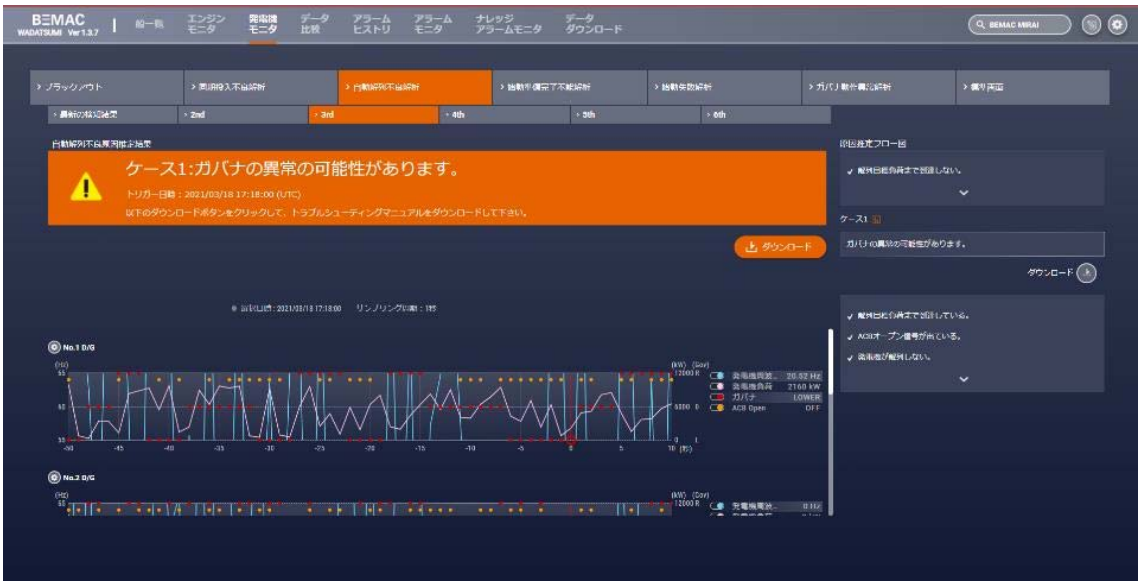


図 63　自動解列不良解析（不具合要因ケース 1 自動判定）

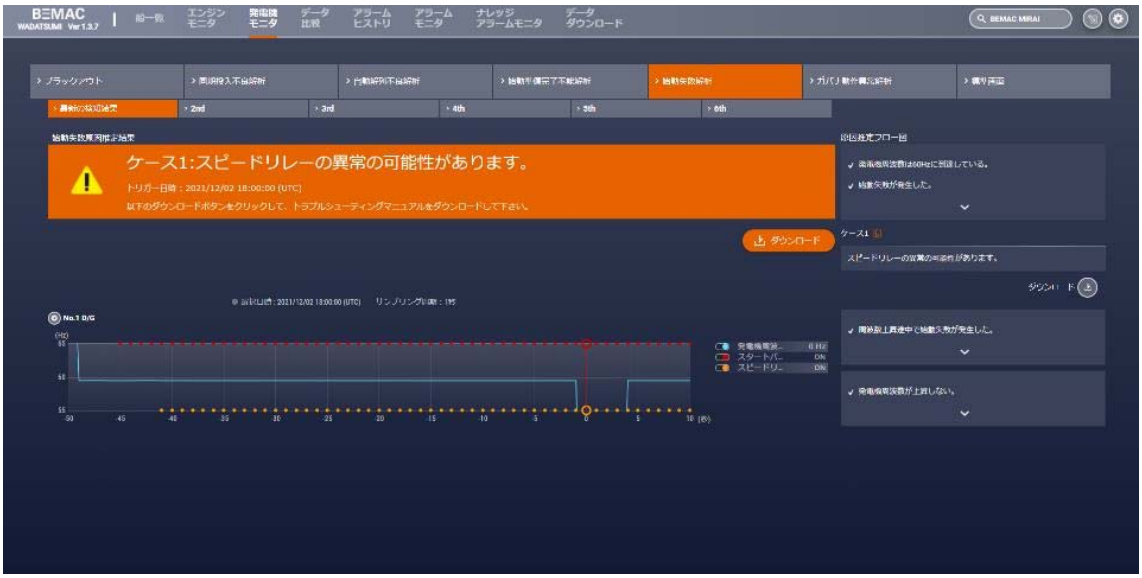


図 64　始動失敗解析（不具合要因ケース 1 自動判定）

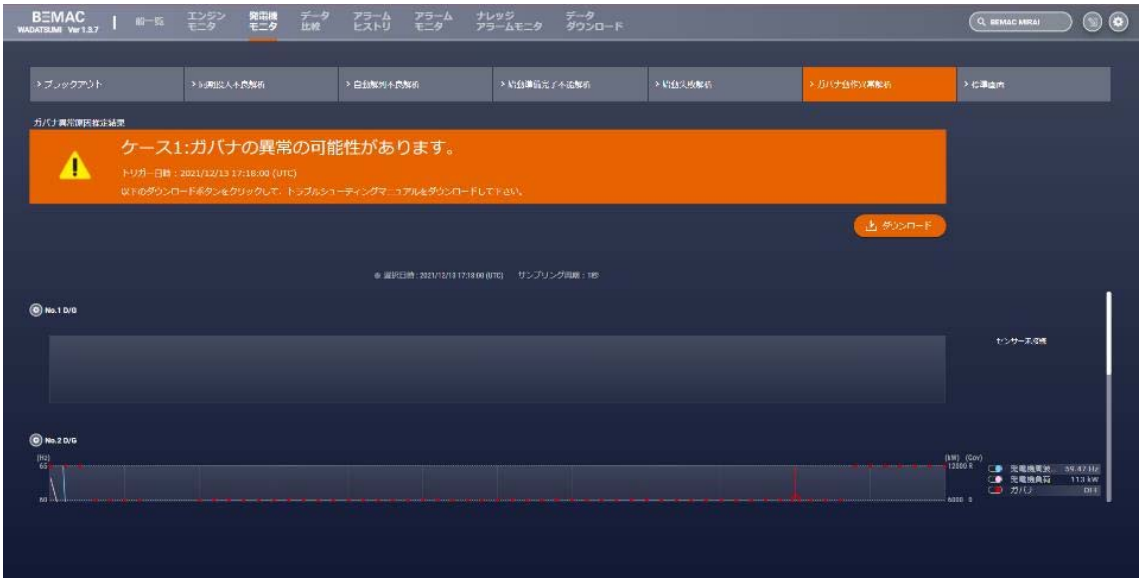


図 65　ガバナ動作不良（不具合要因ケース 1 自動判定）

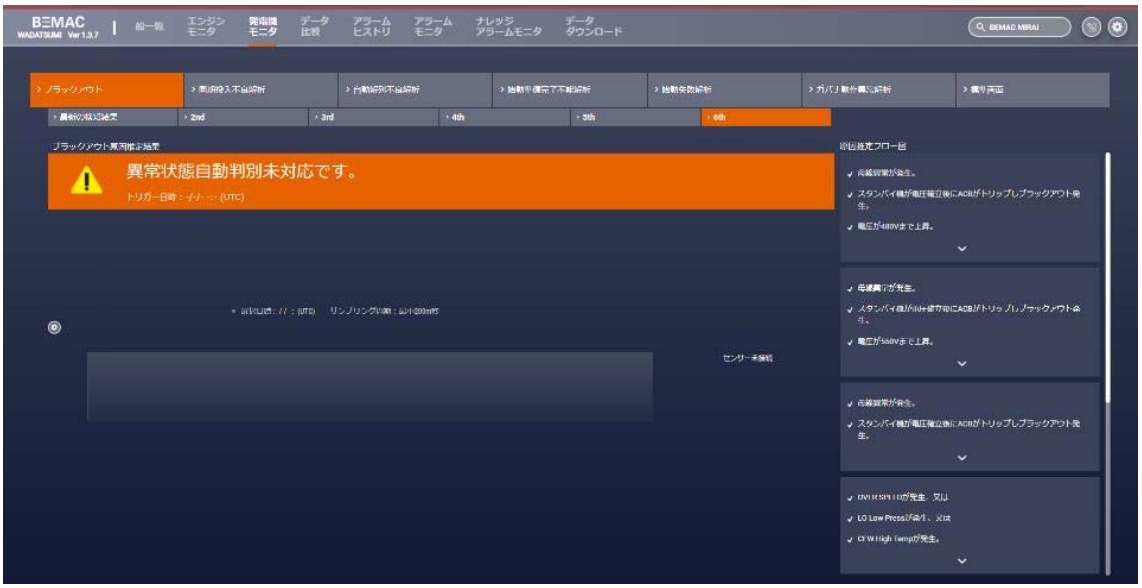


図 66　不具合要因判定不能

5.1 で開発した機能を状態アプリケーションで利用できるように IoT データサーバーへの組み込みを実施することで、予防保全アプリケーションを開発した。

5.3　実証実験

2021 年 3 月から収集している対象船の電気系統に関する本船データを利用して、予防保全アプリケーションで検知できた異常の一例を図 67 示す。これは対象船で 2021 年 5 月 22 日の#1_D/G_潤滑油入口圧力で検知したものであり、状態監視アプリケーションで手動設定されたトリガで検知できない異常、トラブルの予兆が予防保全アプリケーションで検知できていることが確認できた。更にこの結果を基に実施者がトラブルシューティングを実施したところ、センサー自体の異常であることが判明した。

図 67　異常検知の一例

（対象船の 2021 年 5 月 22 日の#1_D/G_潤滑油入口圧力）

また、誤検知数過多という問題も明らかになった。対象船では本船データの収集を開始した 2021 年 3 月 20 から主なトラブルが発生していないにも関わらず、ほぼ毎日いずれかの項目で誤検知されている（図 68）。これは機械学習のモデルがデータのみで構築されたものであり、船舶の電気系統に詳しいエンジニアのドメイン知識を取り入れていないことなどが原因の一つとして考えられる。

図 68　全項目における異常検知数の日毎の合計値

今後、実施者が予防保全アプリケーションから出力されるログデータを引き続き分析し、船舶の電気系統に詳しいエンジニアにその結果を共有、そこで得られたフィードバックを基に改善、というサイクルを繰り返すことで機械学習のモデルを精緻化していく予定である（図 69）。

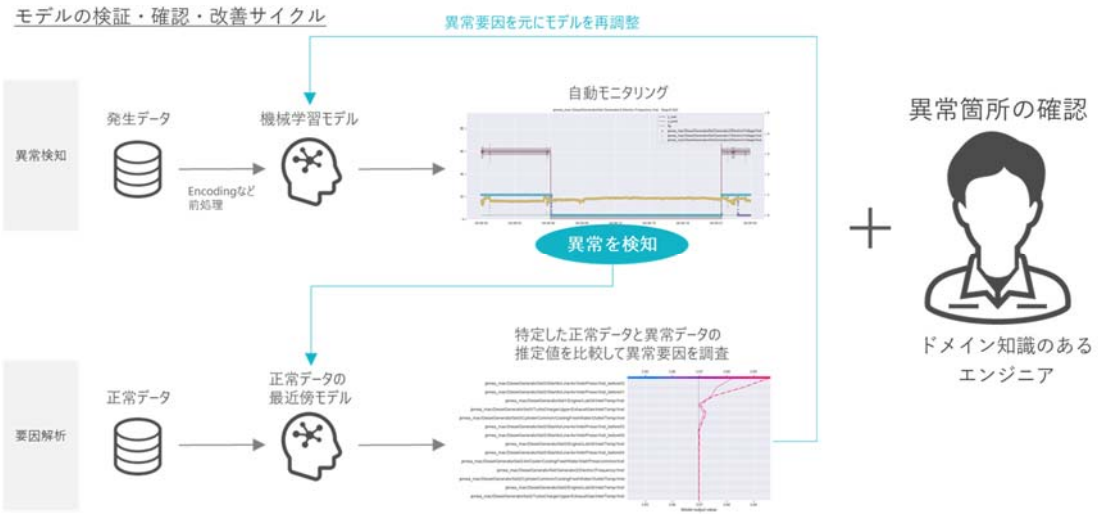


図 69　機械学習のモデルの改善サイクル

5.4　予防保全アプリケーションの機能検証

5.2 の予防保全アプリケーションについて機能検証を実施した結果について報告する。図 70 に検証環境を示す。

図 70　検証環境（発電機コントローラシステム一式）

①発電機シミュレーター一式

②データ収集ユニット（三菱電機 PLC、通信変換器等）

③IoT データサーバー

④汎用 PC（モニタリング/デバッグ用）

⑤ソフトウェア開発環境（Python、Visual Studio Code）

5.4.1　IoT データサーバーでの必要データの抽出機能の機能検証

予防保全アプリケーション用 CSV に追加される状態を、発電機コントローラデバックシステムを用いて実際に入力し、追加行に想定されるデータが正常に入ることを確認した。また、従来からあるデータが変更無く正常に入力されていることも併せて確認した。

結果例は前述の図 56 の自動同期投入不良診断 CSV データ（改良）に示すとおりである。

以上の確認により、追加されたデータと実動作させた発電機コントローラシステムの状態が合致することを確認でき、予防保全アプリケーション用のデータを仕様通り追加することができた。

5.4.2　IoT データサーバーへの自動化機能の組み込み機能検証

トラブルシューティング自動化機能と異常検知トリガ設定自動化機能の組み込みについて、まずは想定されるすべてのパターンの模擬データを作成し、ソフトウェアを動作させ想定される結果が導き出されることが確認できた。次に前項で生成された予防保全アプリケーション用 CSV を用いて検証を実施し、こちらも想定される結果を得られることを確認できた。図 71 から図 73 に開発環境上において模擬データを用いてソフトウェアを動作させた状況を示す。

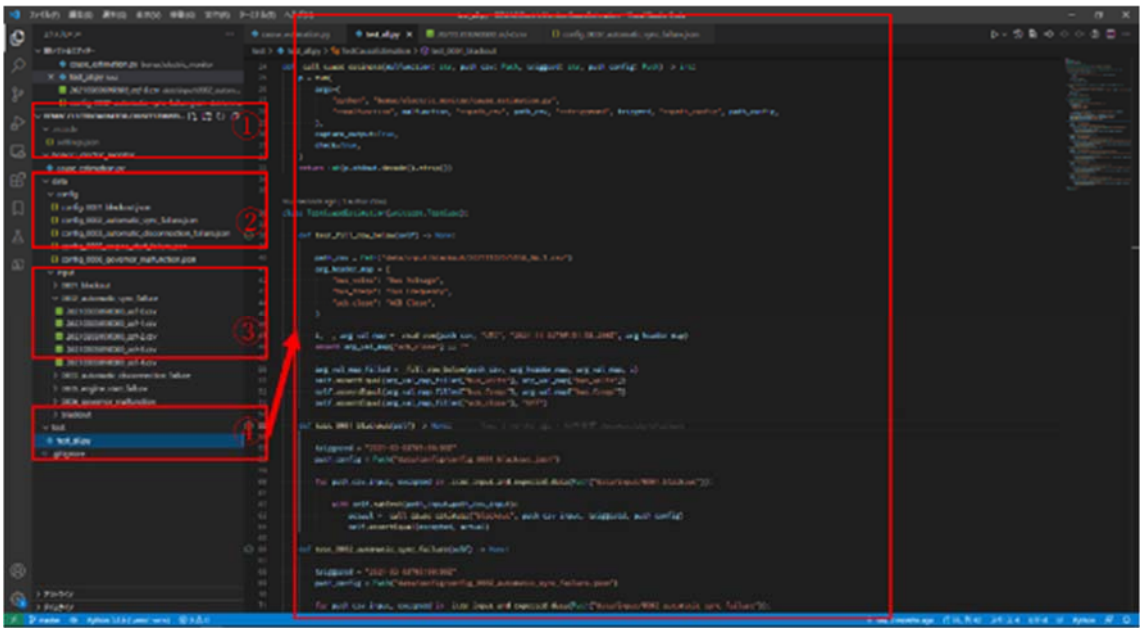


図 71　組み込みソフトウェア検証環境

上図の①がトラブルシューティング自動化機能と異常検知トリガ設定自動化機能を組み込んだソフトウェアモジュール、②が模擬 CSV に対応した項目設定ファイル、③が模擬 CSV データ、④が動作検証用ソフトウェアである。④の動作検証用ソフトウェア内に①～③のモジュール、ファイル、データが読み込まれて実行されることにより、自動解析結果番号が出力されるかを確認した結果、すべてのテストパターンで良好な出力を確認できた。

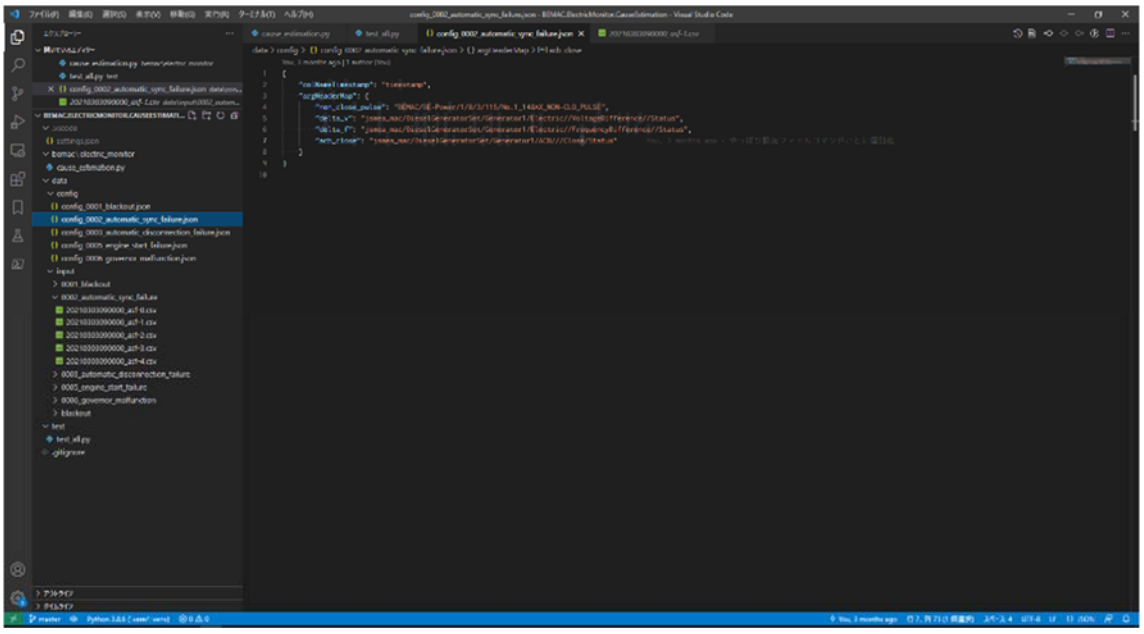


図 72　動作検証用項目設定ファイル（②の内部構成）

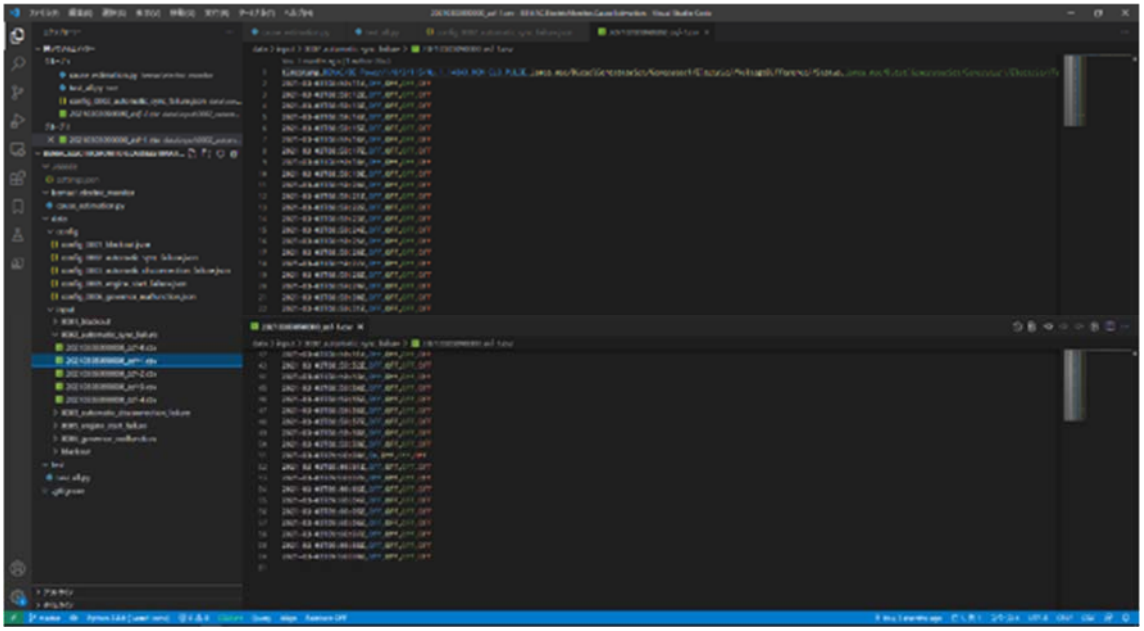


図 73　動作検証用模擬 CSV ファイル（③の内部構成）

5.4.3　予防保全アプリケーションへの自動解析結果の受け渡し機能検証

5.4.2 で設計・開発されたソフトウェアにより自動解析結果を導き出すことができるようになった。その結果を用いて受け渡しファイルが生成されるかを確認した。動作後、所定の位置に所定の名称、内容のファイルが生成され正常に受け渡しを行えることを確認した。生成されたファイル例を図 74 に示す。

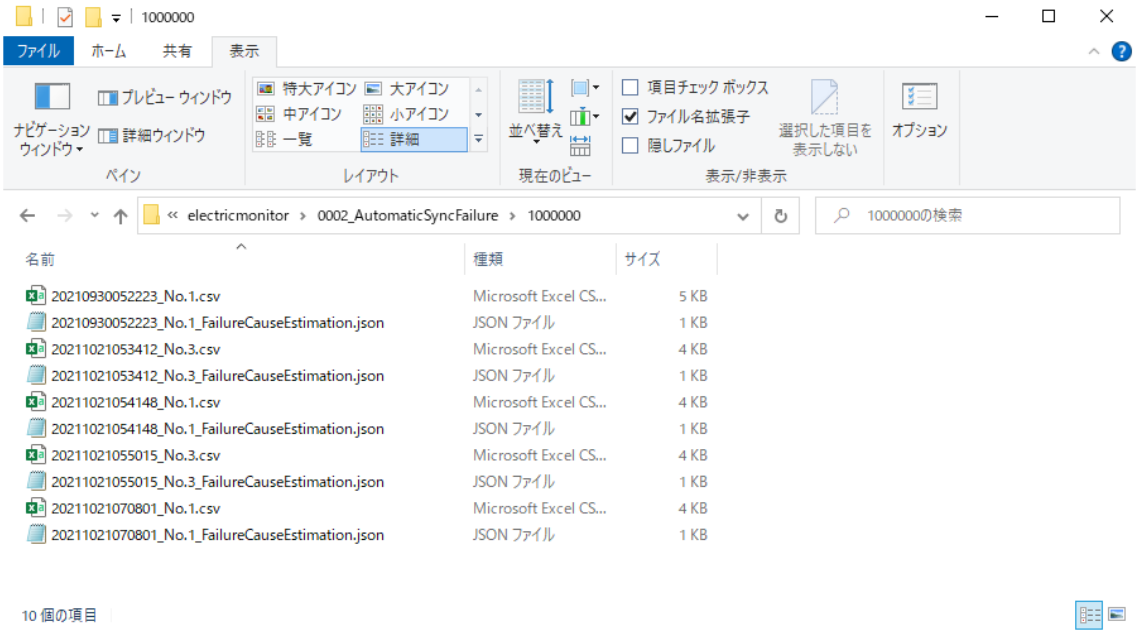


図 74　自動解析結果受け渡しファイル生成状態

5.4.4　予防保全アプリケーションの自動化機能検証

前項までの機能検証を基に予防保全アプリケーションの機能検証を実施した。

まず、単体の動作確認として、5.4.2 で確認した結果、受け渡しファイルについて、ファイルを模擬的に全パターン用意し、想定された表示になるか確認を実施した。結果すべてのパターンにおいて想定される表示が行われることを確認した。ファイルが無い場合は判定未対応画面が表示されることを確認した。また、追加機能として実装した、一番直近に発生した状態の画面に自動的に遷移するアルゴリズムについても、一番新しい日付のファイルを読み込んだ画面に自動的に遷移することが確認できた。

次に、発電機コントローラシステムに接続し実際に動作させ、正常に入力、表示が行われ、不具合を模擬した操作通りに結果が導き出される状態が表示されることを確認できた。

5.4.5　疑似データによる自動化機能検証

実施者のこれまでのトラブルシューティングを参照にして、本船取得データに 1,000 件の擬似的な異常の値を追加した（図 75）。電気的なトラブルを想定した急な値の変化、機械的なトラブルを想定した緩やかな値の変化の 2 パターンを各項目でランダムに追加した。これより予防保全アプリケーションは全ての異常を検知でき、想定した異常についての検知率は 100%となり、目標の 50%を達成することができた。尚、急な値な変化についてはそれが起きたタイミングで検知できており、緩やかな値の変化についてはそれの起こり始めで検知できていたことから、検知タイミングについても問題ないことが確認できた。但し、5.3 で述べたとおり毎日いずれかの項目で誤検知している状態である。これは、機械学習のモデルがデータのみから構築されたものであり、船舶の電気系統に詳しいエンジニアの知見を取り入れられていないことが原因の一つであると考え

られる。したがって、船舶の電気系統に詳しいエンジニアのドメイン知識を機械学習のモデルに取り込むなどで誤検知の数を抑える必要がある。

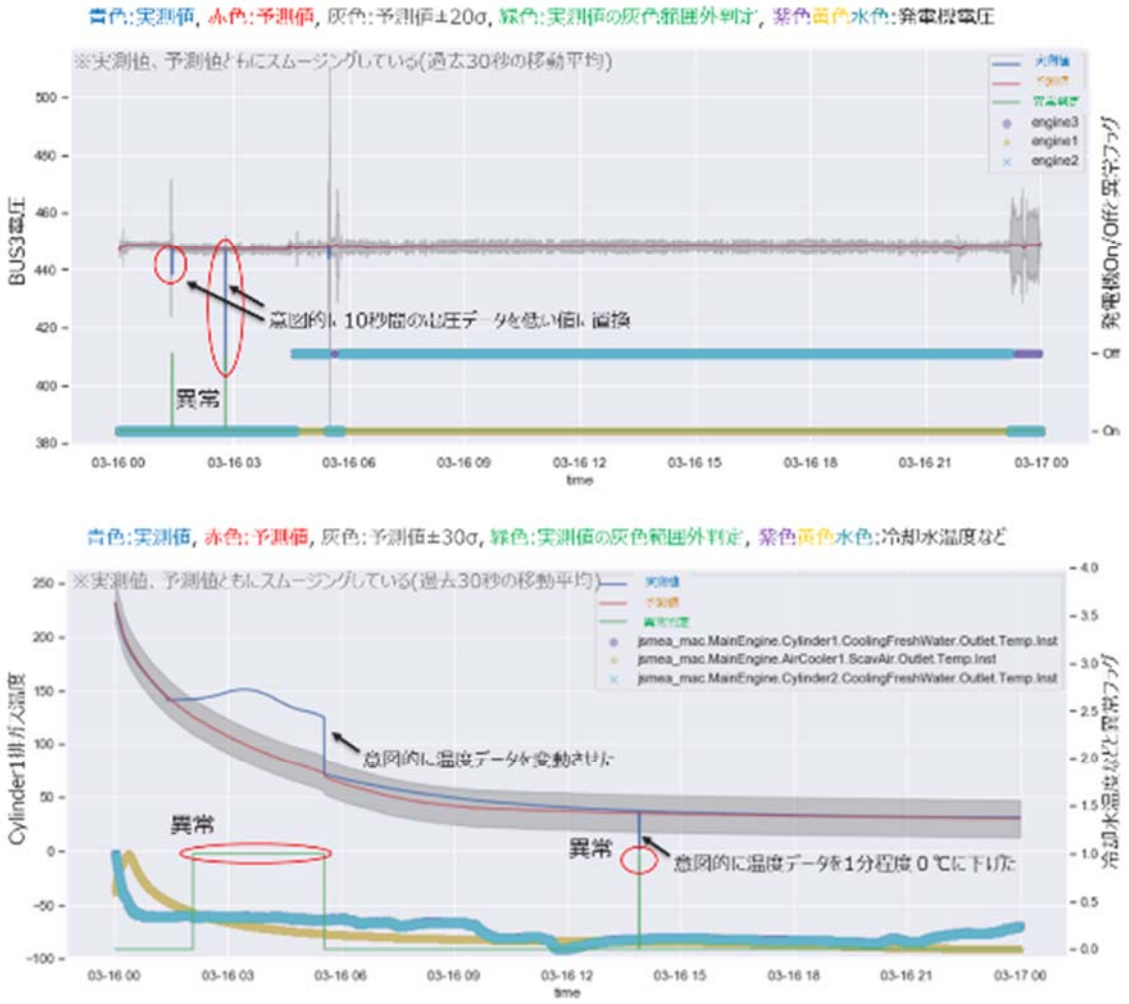


図 75　追加した異常の例

（上）電気的なトラブルを想定した急な値の変化（#3_D/G_電圧）

（下）機械的なトラブルを想定した緩やかな値の変化（#1_D/G_冷却清水集合出口温度）

5.3 で述べたとおり、既存の状態監視アプリケーションで手動設定されたトリガで検知できない異常、トラブルの予兆も検知できたことも確認済みである。今後、実施者が予防保全アプリケーションから出力されるログデータを引き続き分析し、船舶の電気系統に詳しいエンジニアにその結果を共有、そこで得られたフィードバックを基に改善、というサイクルを繰り返すことで機械学習のモデルを精緻化していく予定である。

６．目標の達成状況

船内で収集されたデータを基に異常検知とトラブルシューティングが自動化された機能を持ち合わせた機器の予防保全の支援を行うアプリケーションを開発した。実施者のこれまでのトラブルシューティングを参照にして、本船データに 1,000 件の擬似的な異常の値を追加することで機能検証を実施した。これより予防保全アプリケーションは追加した全ての異常を検知でき、検知率は 100%となり、目標の 50%を達成することができた。急な値の変化についてはそれが起きたタイミングで検知できており、緩やかな値の変化についてはそれの起こり始めで検知できていたことから、検知タイミングについても問題ないことが確認できた。また、既存の状態監視アプリケーションで手動設定されたトリガで検知できない異常、トラブルの予兆も検知できたことも確認できた。今後、実施者が予防保全アプリケーションから出力されるログデータを引き続き分析し、船舶の電気系統に詳しいエンジニアにその結果を共有、そこで得られたフィードバックを基に改善、というサイクルを繰り返すことで機械学習のモデルを精緻化していく予定である。

７．2021 年度の実施内容の概要

１）予防保全アプリケーションの設計

2020 年度に開発した状態監視アプリケーションにおいてフローチャートによるトラブルシューティングを自動化する「トラブルシューティング自動化機能」を検討・設計した。各トラブルに対応する発生事象と確認すべきデータ及び検知する異常との関係をフローチャートの形で整理を行った。更に船内で収集されたデータに機械学習の手法を利用して得た理論値と実測値の差異から異常検知のトリガ設定を自動化する「異常検知トリガ設定自動化機能」を検討・設計した。複数の機械学習手法を検討した結果、NGboost による区間推定を採用した。これにより、人が異常と認識していた事象だけでなく、人が異常と認識していない事象も把握することが可能になった。これらの機能を状態監視アプリケーションに追加することで、機器の予防保全の支援を行うアプリケーションを設計した。

２）予防保全アプリケーションの開発

１）にて設計したトラブルシューティング自動化機能と異常検知トリガ設定自動化機能を、Python を用いて実装した。また、実装したこれらの機能を状態監視アプリケーションで使用できるようにするため IoT データサーバーへの組み込みを実施した。

３）実証実験

2021 年 3 月から収集している対象船の電気系統に関する本船データを利用して、２）で開発した予防保全アプリケーションで状態の監視・トラブルの予兆を検知するという機能が正常に動作するかを確認した。状態監視アプリケーションで手動設定されたトリガで検知できない異常、トラブルの予兆が予防保全アプリケーションで検知できていることが確認できた。更にこの結果を基に実施者がトラブルシューティングを実施したところ、センサー自体の異常であることが判明した。

４）予防保全アプリケーションの機能検証

実施者のこれまでのトラブルシューティングを参照にして、計測データに 1,000 件の擬似的な異常の値を追加することで予防保全アプリケーションの機能検証を実施した。これより予防保全アプリケーションは追加した全ての異常を検知でき、検知率は 100%となり、目標の 50%を達成することができた。尚、急な値の変化についてはそれが起きたタイミングで検知できており、緩やかな値の変化についてはそれの起こり始めで検知できていたことから、検知のタイミングについても問題ないことが確認できた。また、既存の状態監視アプリケーションで手動設定されたトリガで検知できない異常、トラブルの予兆も検知できたことも確認できた。

５）報告書作成

陸上での状態監視・予防保全アプリケーションの機能検証の結果の報告及び本研究全体を通じた総括を踏まえた報告書を作成した。

８．今後の予定

2022 年度は、実施者が予防保全アプリケーションから出力されるログデータを引き続き分析し、船舶の電気系統に詳しいエンジニアにその結果を共有、そこで得られたフィードバックを基に改善、というサイクルを繰り返すことで機械学習のモデルを精緻化していく予定である。更に、各舶用機器メーカーとの協業により他機関への転用も検討する予定である。2022 年度中には既開発済みの予兆検知アプリケーションを販売することにより商品化を完了させ、その後年間 100 隻程度の受注を目指す。

９．まとめ

9.1　2020 年度

配電システムの IoT 化による状態監視アプリケーションの開発によって、データを船上サーバー標準規格で収集することにより、配電系統のモニタリング技術を高度化することができた。その結果、これまであまり重要視していなかった情報も入手可能となりトラブル発生時の原因推測が短期間で行うことが可能となった。船舶の安全性を向上させる一歩であり、来年度以降に開発する予防保全アプリケーション、電気系統のモデル化に生かしていく。

9.2　2021 年度

2020 年度に開発した状態監視アプリケーションにおいてフローチャートによるトラブルシューティングを自動化する「トラブルシューティング自動化機能」を検討・設計した。更に船内で収集されたデータに NGboost という機械学習の手法を利用することで、異常検知のトリガ設定を自動化する「異常検知トリガ設定自動化機能」を検討・設計した。これにより、人が異常と認識していた事象だけでなく、人が異常と認識していない事象も把握することが可能になった。これらの機能を状態監視アプリケーションに追加することで、機器の予防保全の支援を行うアプリケーションを開発した。

最後に本開発に関して公益財団法人日本財団からモーターボート競走共益資金による補助金を受けて実施しており、ここに記して厚く感謝申し上げる。

「この報告書は BOAT RACE の交付金による日本財団の助成金を受けて作成しました」

(一社)日本舶用工業会
〒105-0001
東京都港区虎ノ門一丁目１３番３号（虎ノ門東洋共同ビル）
電話：03-3502-2041　　FAX:03-3591-2206
http://www.jsmea.or.jp