# リアルタイム起動フレームワーク

ユーザーガイド

版 1.0

# 目次



# 概要

## なぜ、リアルタイム起動フレームワークが必要か

Blue Prism はデジタル従業員（Digital Workforce）という概念を提唱し、自動化対象の業務を専用の実行環境に集約して実行することを推奨している。このような形態をとることで、自動化対象業務をデジタル従業員がひっきりなしに実行することができ、費用を節約しつつセキュリティや監査対応を実現、運用管理を容易にし、業務改革につなげていくメリットを得ることができる。

ただ、一方で、「どうしてもリアルタイムに実行したい」という要望があることも事実である。Blue Prism が推奨するスケジューリングによる実行ではなく、ユーザーの働きかけをトリガーとして実行するという形態である。このような要望を実現するにあたって、最も簡単な方法は全員の PC にランタイムリソースを配ることだが、費用対効果、セキュリティ、監査、運用管理など様々な問題があるため、現実的ではない。

Blue Prism のリアルタイム起動をするための方式としては、他にも様々なものがあるが、それぞれ一長一短がある。

| 方式 | 長所 | 短所 |
|---|---|---|
| AutomateC.exe を実行 | ▪ 実行方法がシンプル | ▪ 起動する端末に Blue Prism をインストールする必要がある<br>▪ コマンドプロンプトの実行のため、非技術者にはなじみが薄い<br>▪ 処理結果の確認に手間がかかる |
| コントロールから実行 | ▪ 必要な情報が十分に確認可能なリッチな UI | ▪ 起動する端末に Blue Prism をインストールする必要がある<br>▪ 情報量が多く、非技術者が混乱する恐れがある |
| メールやファイル配置をトリガーに実行 | ▪ 実行方法がシンプル | ▪ メールやファイルの内容は自由に記述できるため、入出力のフォーマットを強制しにくい<br>▪ 進捗の確認や完了通知の視認性に課題があり、対策が必要 |

そこで、従来の方式の課題点をクリアするために、このアセット（リアルタイム起動フレームワーク）は、オブジェクトの公開を活用し、Web API（SOAP/WSDL）をインターフェースとしたリアルタイム起動方式のサンプル実装を提供する。

このアセットは、Blue Prism のドキュメントである

**Web サービス　ユーザーガイド**

https://portal.blueprism.com/documents/v6-user-guide-web-services-japanese

を考え方のベースとし、既に公開されているアセットである

**Process Dispatch Framework（英語）**

https://digitalexchange.blueprism.com/dx/entry/9648/solution/process-dispatch-framework

を実装のベースとしている。

### リアルタイム起動フレームワークのメリット

Blue Prism ユーザーは、このアセットを活用し、またユーザー固有の要件に基づいて拡張をすることで、以下のようなメリットを享受できると考えている。

1. Blue Prism をリアルタイムに起動することができる
2. シンプルな実行方法でリアルタイム起動が行える
3. リアルタイム起動時に、入出力のフォーマットを強制できる
4. リアルタイムに起動したプロセスの処理結果を通知できる

## リアルタイム起動フレームワークを構成する

### 前提条件

1. Blue Prism 6.7 で構成された、Blue Prism サーバと最低 2 つ以上（推奨は 3 つ以上）のパブリックなランタイムリソースが稼働している環境
2. .NET Framework 4.7.2 がインストール済みの端末（クライアント・アプリを動かすため）[1]
3. 以下のオブジェクト（VBO）
   A) Utility – Blue Prism Process Info[2]
   B) Utility – File Management[3]
   C) Data - SQL Server[4]

[1]クライアントアプリを作る際に .NET Framework 4.7.2 を選択しただけであり、4.7.2 固有の機能が使われている可能性は低い。もし、より古いバージョンの.NET Framework で動作させたい場合は、提供されているクライアントアプリのソースからビルドを試みることができる

[2] Blue Prism DX の当該ページ（https://digitalexchange.blueprism.com/dx/entry/9648/solution/process-information-utility-2）から入手できる

[3] Blue Prism DX またはインストールフォルダ内の「VBO」フォルダから入手できる

[4] Blue Prism DX またはインストールフォルダ内の「VBO」フォルダから入手できる

D) Utility - XML[5]

E) （オプション）リアルタイム起動フレームワーク用 Teams チャネル[6]

## アセットに含まれる成果物

このアセット（リアルタイム起動フレームワーク）には、以下の成果物が含まれる

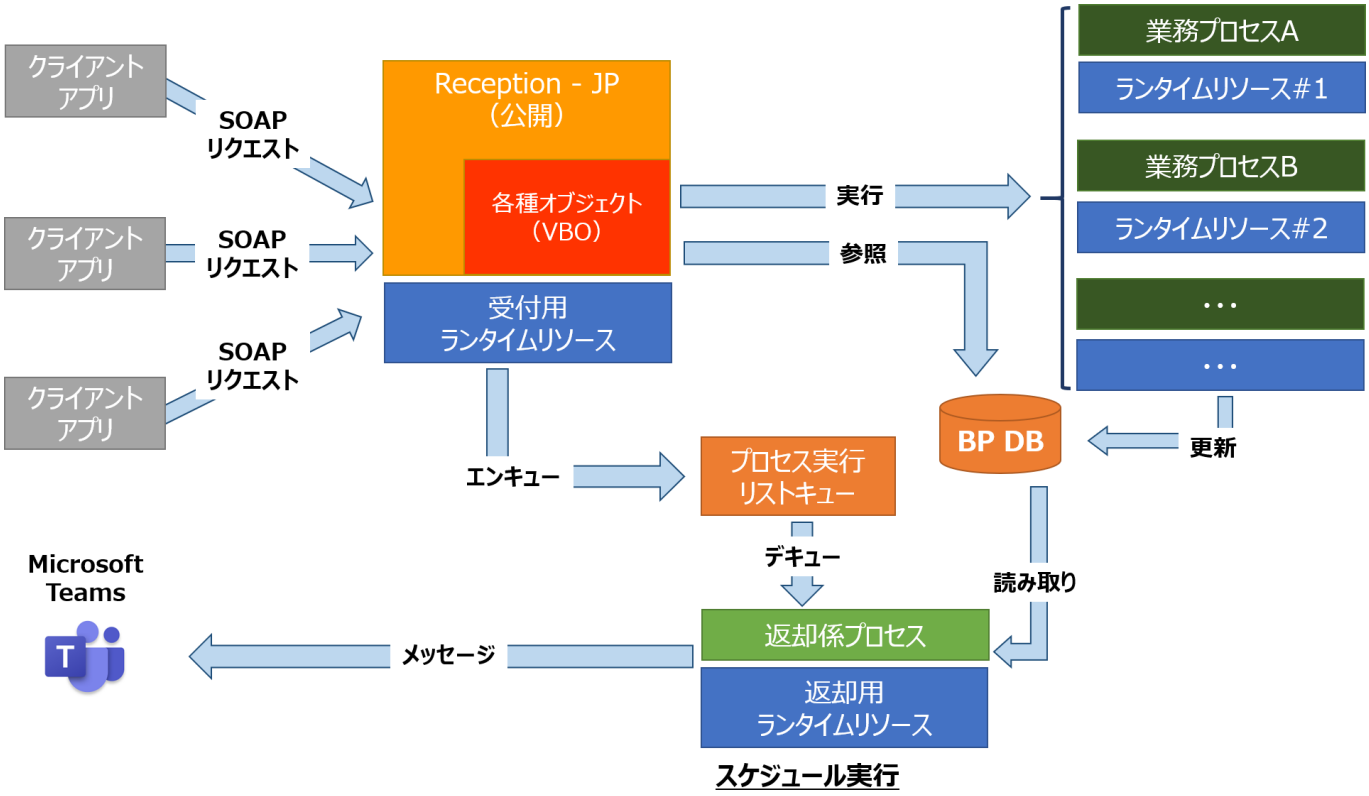
| 場所 | ファイル名 | 概要 |
|---|---|---|
| ドキュメント | リアルタイム起動フレームワーク ユーザーガイド 1.0.pdf | このドキュメント |
| YouTube | Blue Prism リアルタイム起動フレームワーク 完成版 | https://www.youtube.com/watch?v=wXnZ1f5aAWY<br>リアルタイム起動フレームワークのデモ動画。なお、この動画の 00:29 に出るテロップの「BP0598:8182」というのは「BP0598:8184」の誤りである |
| プログラム | リアルタイム起動フレームワーク ver 1.2.bprelease | リアルタイム起動フレームワークを構成するプロセスやオブジェクト（VBO）が含まれる |
| プログラム | リアルタイム起動フレームワークのクライアント（ソースコード）.zip | Microsoft Visual Studio で開くことが可能なプロジェクト |

## 各コンポーネントの関係図

リアルタイム起動フレームワークは、以下のような関係で各コンポーネントが動作する。

[5] Blue Prism Portal の Developer Jumpstart からダウンロードする（https://portal.blueprism.com/products/developer-jumpstart）

[6] Web API サービスとして、新規作成する。詳細は後述




## 各コンポーネントの配置手順

### bprelease のインポート

リアルタイム起動フレームワーク ver 1.2.bprelease を Blue Prism 6.7 にインポートする。

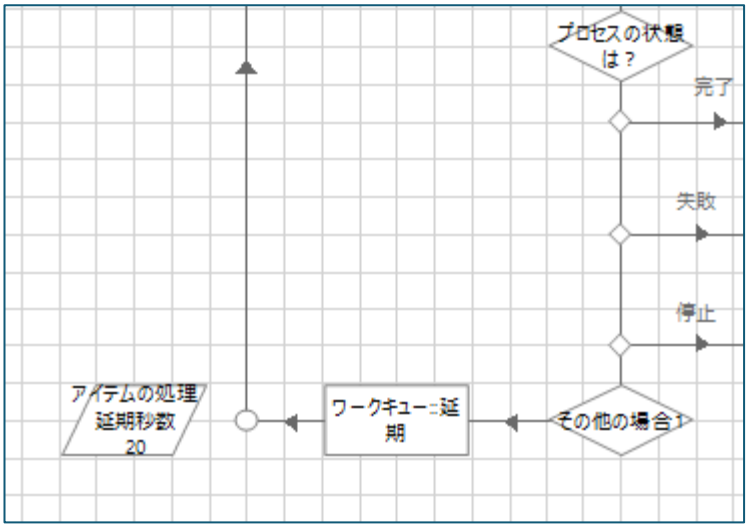
### （オプション）返却係の設定

返却係を定期実行するスケジュールを設定するか、返却係 – 常時実行型を半永久的に実行する。

定期実行の場合は、専用のランタイムリソースで、数分おきに実行するようにして、徐々にチューニングしていく。

半永久的に実行する場合は、コントロールで中止を要求すれば、いつでも安全に停止できる。

返却係の中には、キューの中身を確認する処理が含まれているが、ここで実行中だったキューのアイテムを遅延する処理がある。




これも、初期値として 20 秒の遅延をさせているが、チューニングが必要なポイントである。

また、返却係から Microsoft Teams に投稿するために、Web API サービスの作成が必要となる。

設定手順は以下の通り。

1. 「リアルタイム起動フレームワーク用 Teams チャネル」という名前の Web API サービスを作成する
2. Teams の適当なチームにチャネルを追加し、コネクタ > Incoming Webhook を設定し、URL をコピー
3. 「リアルタイム起動フレームワーク用 Teams チャネル」のベース URL にペースト
4. 「投稿」というアクションを作成し、「Body」というテキスト型のパラメータを作成して公開し、リクエストのメソッドを「POST」、本文のコンテンツに「[Body]」と入れる

### 返却係のための認証情報の設定

認証情報「BP admin」にユーザー名とパスワードを設定する。この認証情報は、返却係が、プロセスの完了状態を確認する際に利用している。

### オブジェクトの公開

オブジェクト「Reception - JP」を公開する

### クライアントアプリのインストール

リアルタイム起動フレームワークのクライアント（ソースコード）.zip を解凍し、Visual Studio でインストーラーを作成する（ビルド > リアルタイム起動フレームワークのクライアントの発行 など）。Visual Studio でのインストーラーの作成方法は、Visual Studio のマニュアルや関連ドキュメントを参考にしてほしい。

なお、クライアントアプリは Reception – JP が公開されたランタイムリソースを参照するようになっている。デフォルトでは localhost:8182 を参照しており、これを変更したい場合は、App.config の以下の箇所を修正し、インストーラーを作成する。

```
<client>
    <endpoint address="http://localhost:8182/ws/ReceptionJP"
binding="basicHttpBinding"
        bindingConfiguration="ReceptionJPSoapBinding"
contract="ReceptionJP.ReceptionJPPortType"
        name="ReceptionJPSoap" />
</client>
```

インストーラーを作成したら、クライアント端末に配布し、クライアントアプリのインストールを行う。

# 稼働確認

## クライアントアプリの操作

クライアントアプリの各項目の説明は以下の通り。


1. ユーザー名：Blue Prism のユーザー名を入力
2. パスワード：Blue Prism のパスワードを入力

3. 実行できるプロセスを取得：入力したユーザーが実行できるプロセス群を取得してプロセス名の候補に設定
4. プロセス名：実行したいプロセス名を候補から選択
5. 選択したプロセスを実行：選択したプロセス名を任意のランタイムリソースで実行

### 全体の動き

全体の動きは、デモ動画を参照。

https://www.youtube.com/watch?v=wXnZ1f5aAWY

# 拡張について

### 拡張の考え方

リアルタイム起動については、冒頭でも触れたように、Blue Prism ユーザーの要件はさまざまである。したがって、リアルタイム起動フレームワークはあくまでサンプル実装とし、オープンソースで提供している。

要件に応じた拡張については、たとえばクライアントアプリについては、画面項目の変更や追加が考えられる。プロセス起動時にパラメータを渡したければ、そのパラメータの入力欄や、入力バリデーションが必要になる。また、返却係は、このサンプル実装では Microsoft Teams に通知を行っていたが、通知はメールの方が良い、という要件もあるだろうし、通知の文言も変更したいという要件も考えられる。

このように、リアルタイム起動フレームワークには拡張のポイントがある、というよりは、拡張が想定される箇所の方が多い、と考えているが、以下にこのユーザーガイドを執筆している時点で想定可能な主な拡張のポイントを列挙しておく。

### 拡張のポイント

主な拡張のポイントは以下である。

1. SSO（シングルサインオン）への対応
2. ログインエージェントへの対応
3. 通信プロトコルに応じたクライアントアプリの改修（現在は、HTTP/認証ありを前提としている）
4. ランタイムリソースの選択アルゴリズムを増やす
5. 入出力パラメータの追加
6. AutomateC.exe のロケールを en-US に固定する（今後のバージョンアップで AutomateC.exe の出力文字列が変わった場合の対処）
7. 公開したオブジェクトに Web API アクセスが行われるたびに、監査ログが出力されるので、その抑制
8. いくつかのデータアイテムを環境変数にする

以上