



# リアルタイム起動フレームワーク

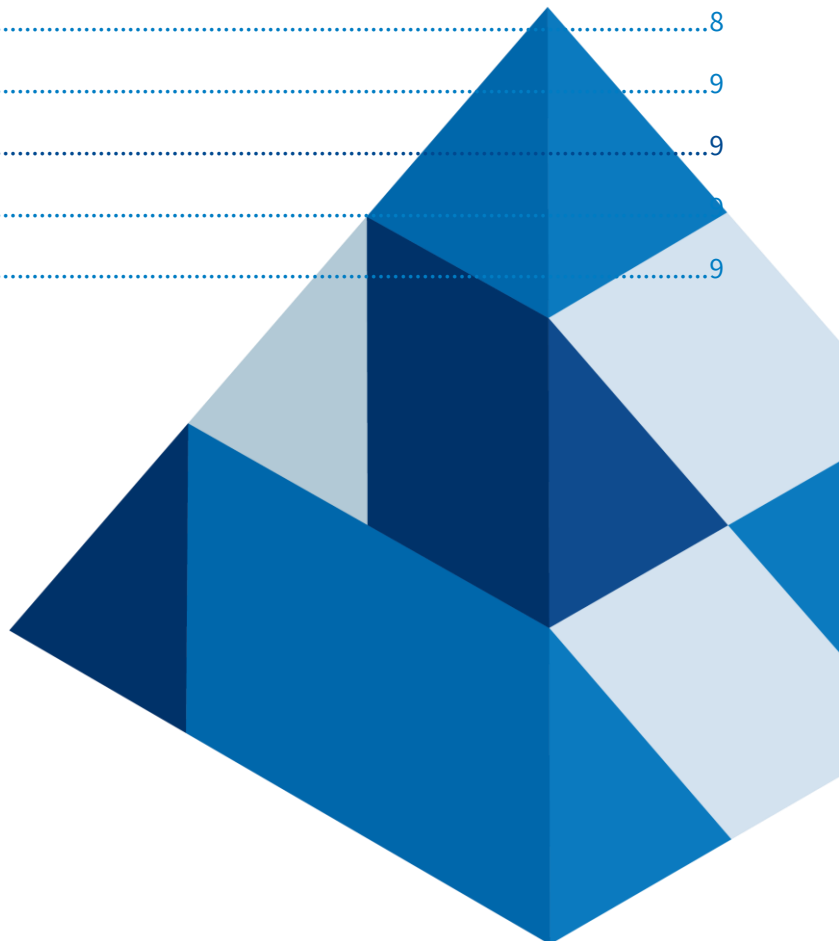
ユーザーガイド

版 1.0



## 目次

概要 .....	1
なぜ、リアルタイム起動フレームワークが必要か .....	1
リアルタイム起動フレームワークのメリット .....	2
リアルタイム起動フレームワークを構成する .....	2
前提条件 .....	2
アセットに含まれる成果物 .....	3
各コンポーネントの関係図 .....	3
各コンポーネントの配置手順 .....	4
稼働確認 .....	6
クライアントアプリの操作 .....	6
全体の動き .....	7
トラブル・シューティング .....	7
よくあるエラー .....	7
オブジェクトログを確認する .....	8
ランタイムリソースの再起動 .....	9
拡張について .....	9
拡張の考え方 .....	9
拡張のポイント .....	9



## 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	<ul style="list-style-type: none"><li>■ 実行方法がシンプル</li></ul>	<ul style="list-style-type: none"><li>■ 起動する端末に Blue Prism をインストールする必要がある</li><li>■ 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</li><li>■ 処理結果の確認に手間がかかる</li></ul>
コントロールから実行	<ul style="list-style-type: none"><li>■ 必要な情報が十分に確認可能なリッチな UI</li></ul>	<ul style="list-style-type: none"><li>■ 起動する端末に Blue Prism をインストールする必要がある</li><li>■ 情報量が多く、非技術者が混乱する恐れがある</li></ul>
メールやファイル配置をトリガーに実行	<ul style="list-style-type: none"><li>■ 実行方法がシンプル</li></ul>	<ul style="list-style-type: none"><li>■ 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</li><li>■ 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</li></ul>

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）<sup>1</sup>
3. 以下のオブジェクト（VBO）
  - A) Utility – File Management<sup>2</sup>
  - B) Data - SQL Server<sup>3</sup>
  - C) Utility - XML<sup>4</sup>

---

<sup>1</sup>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

<sup>2</sup> 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

<sup>3</sup> 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

<sup>4</sup> Blue Prism Portal の Developer Jumpstart からダウンロードする（<https://portal.blueprism.com/products/developer-jumpstart>）

D) (オプション) リアルタイム起動フレームワーク用 Teams チャンネル<sup>5</sup>

## アセットに含まれる成果物

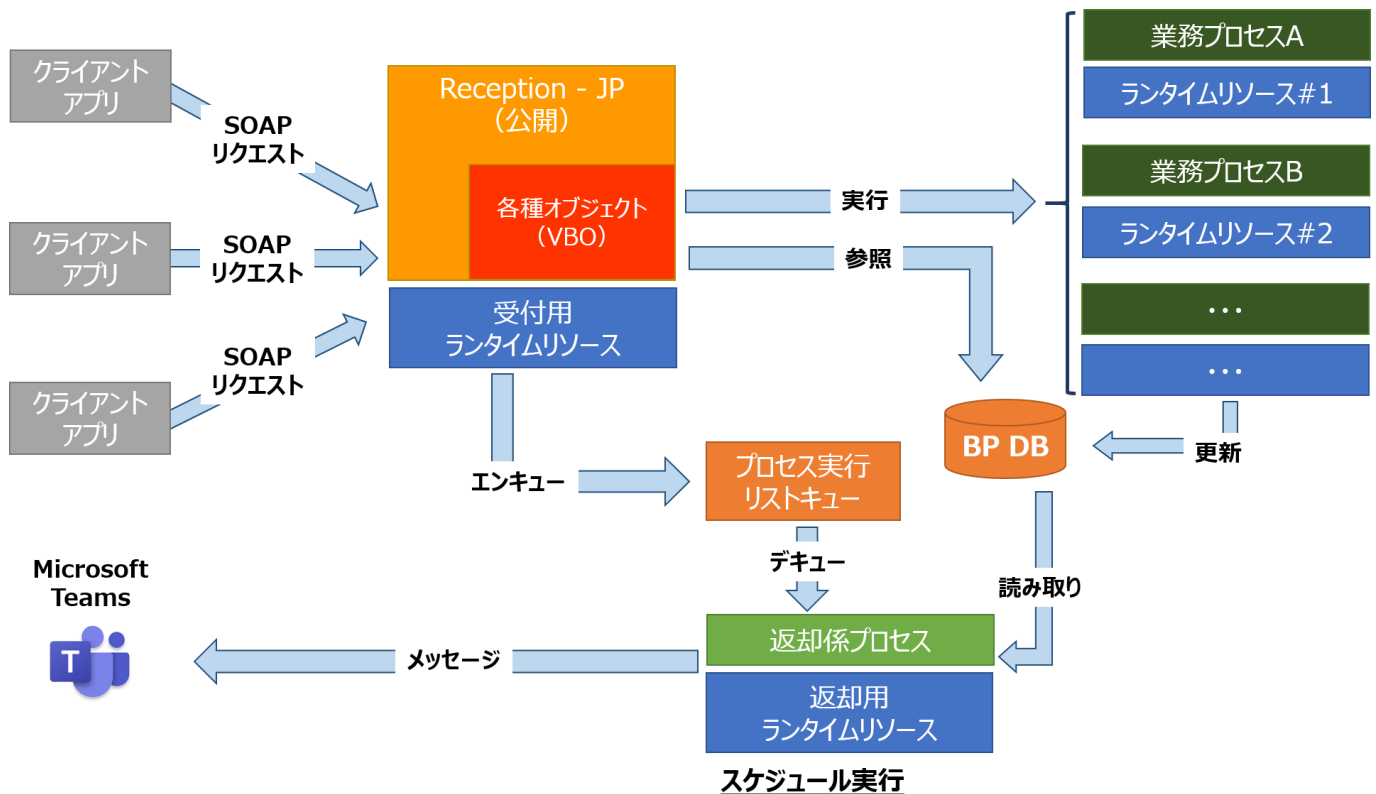
このアセット (リアルタイム起動フレームワーク) には、以下の成果物が含まれる

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	<a href="https://www.youtube.com/watch?v=wXnZ1f5aAWY">https://www.youtube.com/watch?v=wXnZ1f5aAWY</a> 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 1.5.1.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。

<sup>5</sup> 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 1.5.1.bprelease を Blue Prism 6.7 にインポートする。

### Dispatcher - JP の設定

初期化アクションにある以下のデータアイテムに初期値を設定する。

1. データベースサーバー名の初期値に、Blue Prism の DB が動いているホスト名（IP アドレス）を設定する。
2. データベース名の初期値に、Blue Prism のデータベース名を設定する。

### 認証情報「BP DB User」の設定

認証情報「BP DB User」に、Blue Prism の DB がある SQL Server のユーザー名とパスワードを設定する（ユーザーは SQL Server のユーザーで、Blue Prism のデータベースへの接続および表の読み取り権限を有すること）。

また、認証情報「BP DB User」のアクセス権を適切に設定する。

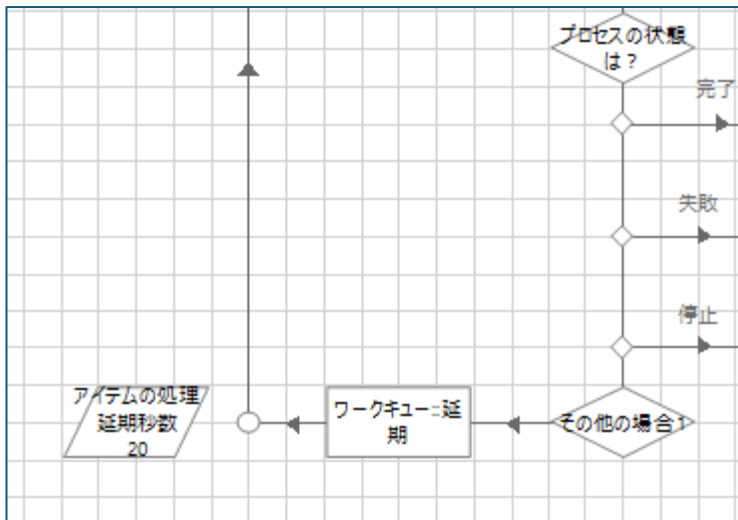
### （オプション）返却係の設定

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 チャンネル」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ンネルを追加し、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 チャンネル」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、

```
リアルタイム起動フレームワークのクライアント¥bin¥Debug¥リアルタイム起動フレームワークのクライアント.exe.config
```

の以下の箇所を修正し、

```
<client>

  <endpoint address="http://localhost:8182/ws/ReceptionJP"
binding="basicHttpBinding"

  bindingConfiguration="ReceptionJPSoapBinding"
contract="ReceptionJP.ReceptionJPPortType"

  name="ReceptionJPSoap" />

</client>
```

実行ファイルである

```
リアルタイム起動フレームワークのクライアント¥bin¥Debug¥リアルタイム起動フレームワークのクライアント.exe
```


を起動すればよい。

## 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。





A screenshot of a Windows dialog box titled "Blue Prism のプロセスを起動". The dialog has a blue background. It contains two input fields: "ユーザー名" (Username) with the value "admin" and "パスワード" (Password) with masked characters "\*\*\*\*\*". Below these is a button labeled "実行できるプロセスを取得" (Get executable processes). Underneath is a dropdown menu for "プロセス名" (Process name) with "sleep" selected. To the right of the dropdown is a button labeled "選択したプロセスを実行" (Execute selected process). At the bottom, a message reads "セッションID : 469a8227-8c05-4f00-83cb-daa113156745で受け付けました。" (Accepted with session ID: 469a8227-8c05-4f00-83cb-daa113156745).

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。

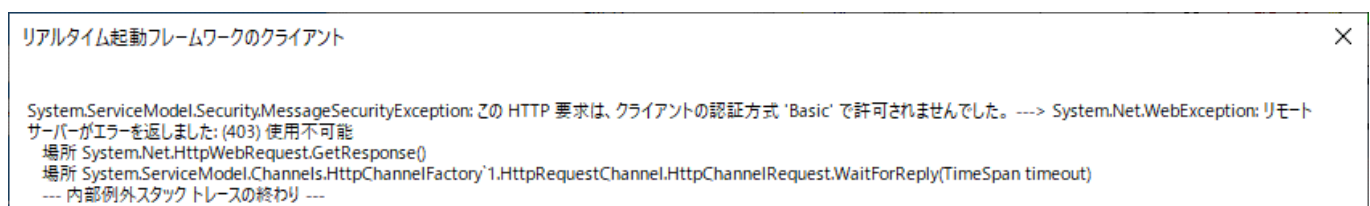
<https://www.youtube.com/watch?v=wXnZ1f5aAWY>

## トラブル・シューティング

### よくあるエラー

オブジェクト「Reception - JP」が公開されていない

クライアントアプリに以下が表示される（403）。



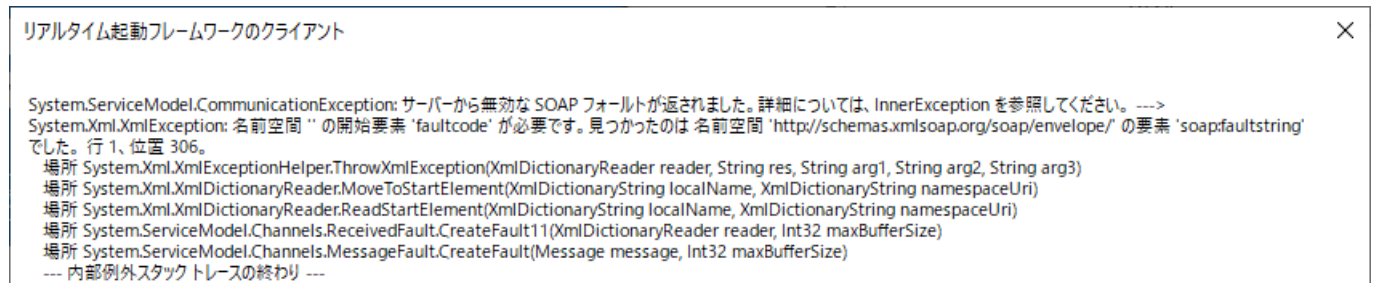
## ユーザー名、パスワードが間違っている

クライアントアプリに以下が表示される（401）。



## プロセス名を指定せずに起動しようとした

クライアントアプリには以下のように表示される。



後述するオブジェクトログには、以下のように表示される。



## オブジェクトログを確認する

クライアントアプリからは正しい処理を行っているのに、意図しない動作をしている場合は、監査 - オブジェクトログ（システム > 監査 > オブジェクトログ）を確認する。例えば以下のように、エラー内容が記録されている。

セッションログビューアー						
ファイル(E) 表示(V)						
プロセス: Reception - JP セッション開始日時: 2020/05/17 15:28:33						
検索						
ログ番号	ステージ名	プロセス	オブジェクト	アクション	結果	処理開始日時
1					セッションログを作成済み	2020/05/17 15:28:32
2	FAIL		Data - SQL Server	Raise Exception If Appr...	エラー: Action Failed: ユーザー 'dbuser' はログインできませんでした。...	2020/05/17 15:28:35
3	Get Process Info from DB		Utility - Blue Prism Proce...	Get Processes	エラー: Action Failed: ユーザー 'dbuser' はログインできませんでした。...	2020/05/17 15:28:35
4	Action Exception		Utility - Blue Prism Proce...	Get Processes	エラー: Action Failed: The following exception was encountered ...	2020/05/17 15:28:35
5	Utility - Blue Prism Process Info:Ge...		Dispatcher - JP	Process Info	エラー: Action Failed: The following exception was encountered ...	2020/05/17 15:28:35

1 の 1

表示タイプ: グリッド ページごとの行数: 1000

## ランタイムリソースの再起動

初期値を変えるなどした場合は、ランタイムリソースの再起動を行って、設定を反映する。

## 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. 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
7. いくつかのデータアイテムを環境変数にする

以上