DEVELOPER HANDBOOK

# Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 11/10/2017 | 1.0.0 | Changes to the original template | Cauê Engelmann |
| 27/10/2017 | 1.0.1 | Changes to the naming convention | Cauê Engelmann |
| 09/11/2017 | 1.0.2 | New section - Utilities and some punctual changes | Cauê Engelmann |
| 17/04/2018 | 1.1.0 | Change of process name to English | William Almeida |
| 13/07/2018 | 1.1.2 | Layout rules  Environment variables | Cauê Engelmann |

# Content

[Version Control 2](#_Toc519500757)

[Content 3](#_Toc519500758)

[1. Purpose 5](#_Toc519500759)

[2. Machine Configuration for Developers 6](#_Toc519500760)

[3. Naming Conventions 7](#_Toc519500761)

[3.1 Business Objects 7](#_Toc519500762)

[3.2 Process 8](#_Toc519500763)

[3.3 Variables 9](#_Toc519500764)

[3.4 Actions, Decision, Calculation… 10](#_Toc519500765)

[3.5 Queue 11](#_Toc519500766)

[3.6 Environment Variables 11](#_Toc519500767)

[3.7 Credentials 12](#_Toc519500768)

[3.8 Release 13](#_Toc519500769)

[3.8.1 Package Name / Release Name 13](#_Toc519500770)

[4. Best Practices 14](#_Toc519500771)

[4.1 General Layout 14](#_Toc519500772)

[4.1.1 Stages Name 14](#_Toc519500773)

[4.1.2 Line Spacing 14](#_Toc519500774)

[4.2 Object 15](#_Toc519500775)

[4.2.1 Application Modeler 15](#_Toc519500776)

[4.3 Actions 17](#_Toc519500777)

[4.4 Process 22](#_Toc519500778)

[4.5 Description 23](#_Toc519500779)

[4.6 Exceptions 24](#_Toc519500780)

[4.6.1 Exception Handling 24](#_Toc519500781)

[4.6.2 Types of Exception 24](#_Toc519500782)

[**Problems Example:** 25](#_Toc519500783)

[**Casting** 26](#_Toc519500784)

[5. Utilities 27](#_Toc519500785)

[5.1 Clean Work Queue 27](#_Toc519500786)

[5.2 Greetings 27](#_Toc519500787)

# Purpose

This document was designed to give instruction, information and advice developers to maintain a high quality of development and help improve the success of the software development (robot).

# Machine Configuration for Developers

The Windows Language must be configured as Portuguese.

The Office Language must be configured as Portuguese.

The date format must be configured as dd/MM/yyyy.

The SAP language must be configured as Portuguese (PT).

Another application should be set as Portuguese.

# Naming Conventions

## Business Objects

**Objects Name Structure:**

[Application Type]\_[Application ID]\_[Transaction]\*\_[Description]\*\_[ElementType]\*

|  |  |  |  |
| --- | --- | --- | --- |
| Structure | Definition | Examples | Required |
| Application Type (Folder) | Classification of the application | SAP SICALC WEB | X |
| Application ID (Folder) | Application ID for SAP is the product name and domain for web applications | ECC CCS FormulasCoge | X |
| Transaction\* | Transaction is only necessary on SAP applications | ME21N EA40 | X |
| Description\* | Element description | ClassCont Item | X |
| Element Type\* | Element Type is necessary on SAP applications when the screen contains too much elements | Tab Table |  |

\* Only for SAP applications

Examples:

SAP\_ECC\_ME21N\_ClassCont\_Tab

SAP\_ECC\_ME21N\_Item\_Table

SAP\_CCS\_EA40\_Item\_Tab

SICALC\_SelfService

WEB\_CCEE

## Process

**Process Name Structure: [Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\***

|  |  |  |  |
| --- | --- | --- | --- |
| **Structure** | **Definition** | **Examples** | **Required** |
| **Robot Nº** | Nº of the robot | R03 R04 R11 | X |
| **Business Unit** | Initials of the business unit | CSP CORP UND | X |
| **Company\*** | Name of the company if the business unit is UND | SP ES |  |
| **Activity** | Name of the digitalized activity or process | IrrfFolha Arrecadação ProtocoloPidSp | X |
| **Application Type/ID \*** | Classification of the application + name of the application for SAP or domain for web applications | SAPECC |  |

Examples:

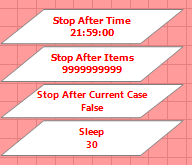
R03\_CSP\_IrrfPayroll

R04\_CORP\_ES\_Collection

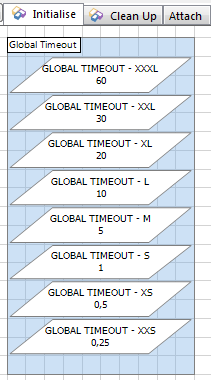
R06\_UND\_SP\_PidProtocol\_SAPECC

## Variables

Variables (or data items) must have the first word letter with upper case:

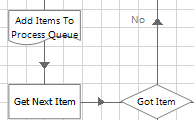


Upper case for constants:



The TIMEOUT values must be copied and pasted in each Business Object Initialise Page in order to simplify their use.

## Actions, Decision, Calculation…



Provide descriptions to Inputs, Outputs and Actions: this removes ambiguity and provides content for auto-generated Business Object Definition (BOD) document.

## Queue

The queue should have the same name as the process to recognize easily which process has each queue.

[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*

If the process has more than one work queue they must be distinguished by a unique attribute:

[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*\_[Unique Attribute]

[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*\_[Unique Attribute]

Example:

R21\_UND\_ES\_Sequencing\_Base

R21\_UND\_ES\_Sequencing\_Child

## Environment Variables

For the storage of the directory and e-mail data, environment variables must be created according to the structure below:

Note: For the other variants of environments variables that are necessary to create, you need request to the CER.

- Directory:

RootPath\_[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*

Example:

* RootPath\_R39\_UND\_SP\_AnomalyCollection;
* RootPath\_R32\_UND\_ES\_PIDPreliminaryAnalysis

- E-mail:

Email\_To\_[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*

Email\_Cc\_[Robot Nº]\_[Business Unit]\_[Company]\*\_[Activity]\_[Application Type/ID]\*

Example:

* Email\_To\_R07\_UND\_SP\_ReadingEquipmentReplacement
* Email\_Cc\_R34\_UND\_PIDRevenuesAndReturns

- SAP System:

System\_SAP\_[Application ID]\_[System ID]

Example:

System\_SAPECC\_E5Q

System\_SAPCCS\_E4Q

## Credentials

SAP: [Robot Nº]\_[Application Type/ID]\_[System ID]

Example:

R06\_SAPECC\_E5Q

R47\_SAPCCS\_E4Q

Another Applications: [Robot Nº]\_[Application Type]\_[Application ID]\_[Company Code\*]

Example:

R19\_Web\_FormulasCoge

R01\_Web\_Viana\_ESCE

R01\_Web\_Viana\_GHPC

\* If necessary

## Release

### Package Name / Release Name

[Robot Nº]\_[Description]

Example:

R03\_IrrfPayroll

# Best Practices

## General Layout

### Stages Name

All stages names must be in English and must use capital letters.

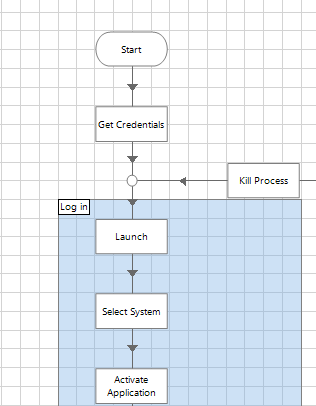
Example:

Select System

Activate Application

### Line Spacing

The vertical line spacing between two stages must be equal to 2:



## Object

### Application Modeler

The Application Modeler must be separated by section for each part of the screen. Naming convention is: {Element Type} - {Element Name}.



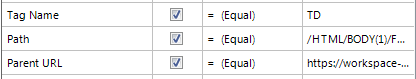
\* If the application is connected in another language, you should input the name of original field in ‘Description’

Example:

Name: Input - User Name  
Description: Usuário

|  |
| --- |
| Elements |
| Input (Field to input text) |
| Label (Fixed text description) |
| Combo Box (Elements in a list) |
| Check Box |
| Radio Button |
| Button |
| Window (Pop Up is also an Window) |
| Table |
| Link |
| Menu |
| Tab |
| Tree View |
| Scrollbar |

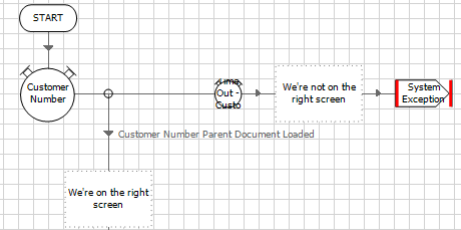
Do not use specific environment data because this will cause the process to fail when migrated. If required, make the value dynamic.



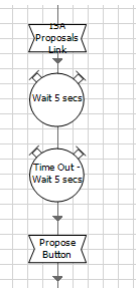


## Actions

* Add wait stage at start of each action: this will confirm the process is on the correct path and absorb system latency to increase the resilience of the process.
* Always throw exception on timeout: Throw the exception and let the process handle it.



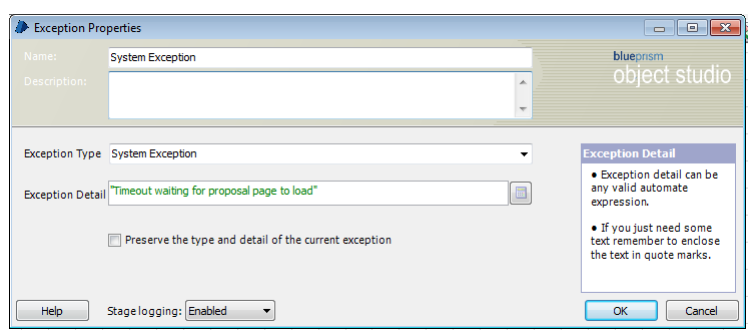
* Avoid using arbitrary waits: Arbitrary waits should only be used if a screen change cannot be waited for.

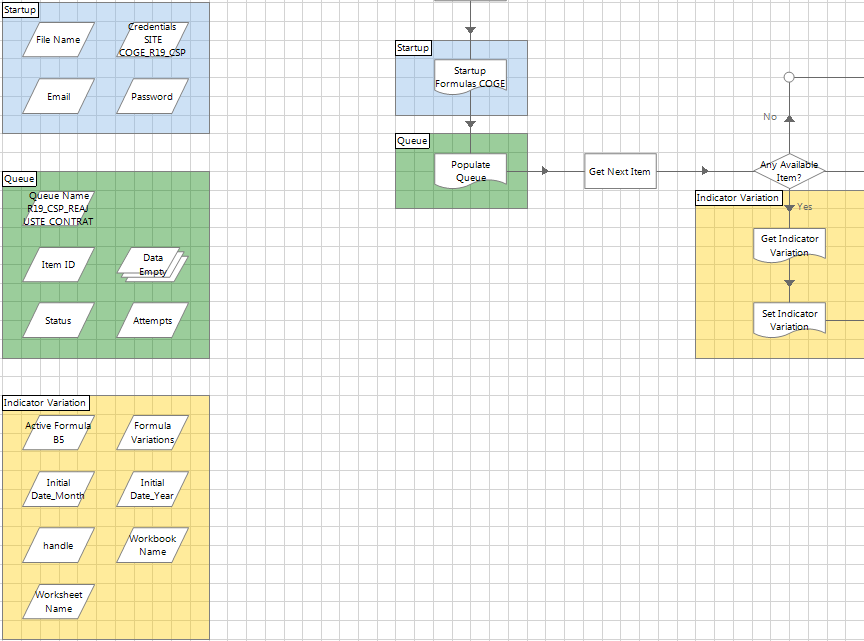


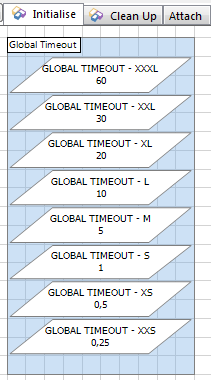
* Avoid using **Sleep**: Sleep must only be used if a screen change cannot be waited for.
* Always wait for the screen to change: Use wait stages after Navigate stages or any stage that causes the screen to update. It will ensure the process runs at its fastest.

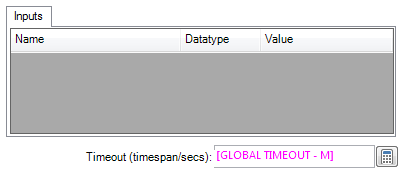


* Do not make business decisions in the object: business decisions must be made on the Processes.
* Provide appropriate Type and Detail for exceptions: select the type of exceptions and provide a detail.



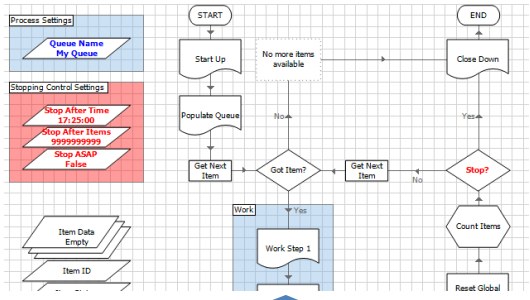
* Provide descriptions to Inputs, Outputs and Actions: this removes ambiguity and provides content for auto-generated Business Object Definition (BOD) document.
* Group data items relevant in blocks (the default color is blue).
* The maximum number of colors in the blocks is **three**.
* All the blocks must have their name according to their meaning
* Store global data items on the Initialise Page.
* Always create variables for timeout and use the Timeout Variables instead of using numbers on each wait stage.

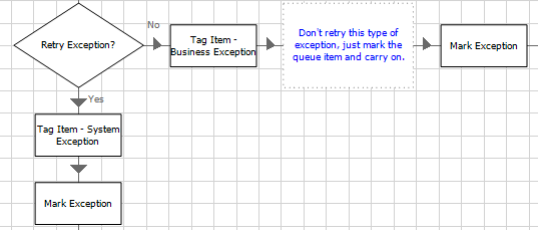




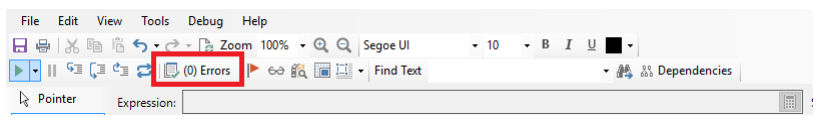
* Do not use Hard-Code data in Data Items: use input parameters and let the process provides the values. Always **clean** **the values** inside the Data items.
* Do not use Hard-Code data within Stages: **use Data Items.**

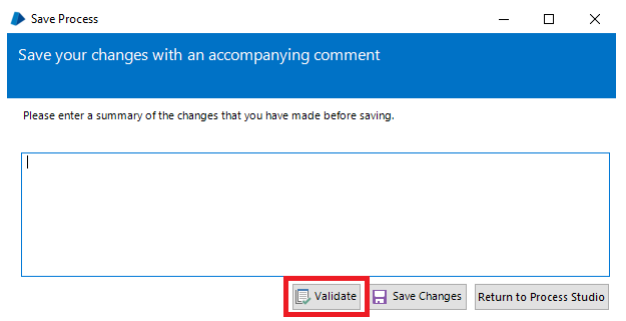
## Process

* All the Subpages names must be in English and must use capital letters.
* Use standard templates provided:
* Main page should contain high-level process steps: all process detail and decision should be divided into logical subpages.
* Always check for errors or validate when saving.
* Correct use of retries: Where possible retry system exception within the process. This may require special navigation or even a restart of the system. Do not retry business exceptions:

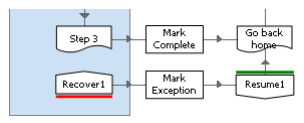


* Always check for errors or validate when saving:





* Always update the queue and move past the Resume stage before attempting to remedy the situation, because remedy the situation before update can cause the process to fail before the item has been updated.



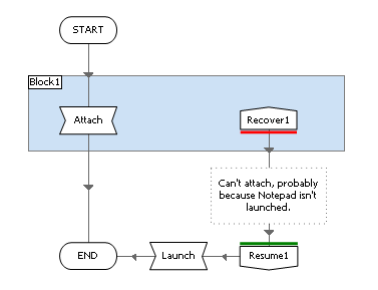
## Description

* Code documentation is important so it is always recommended use a description to build a self-documenting code where data items, actions, process, objects, packages, queues names must be meaningful and code-flow must be pure. It is also recommended that for very complex code sections, short comment lines may be included.

## Exceptions

### Exception Handling

* By default, a Recover stage will attract any exception occurring on its page and this can sometimes lead to an infinite loop.
* A Block is a mechanism for isolating exception handling to a specific area and is a good way to prevent an infinite loop:



* Always prevent an exception to occur in unprotected area. An exception occurring in this unprotected area will terminate the process, so care must be taken to assess the whole process for the risk of unhandled exceptions.

### Types of Exception

System Exception

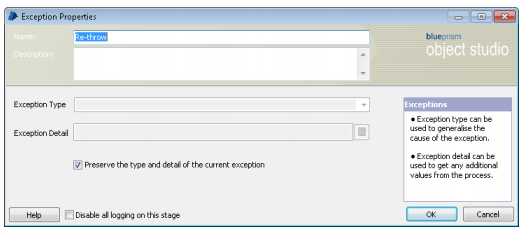
* Technical Issues: Blue Prism is used to automate other applications and as such is dependent on the behavior of these applications and the Blue Prism should expect some issues with the applications it is being automated and for this kind of application-based problem is system exception.

Business Exception

* A process may be set up deliberately disregard certain types of case. The exception for this kind of rules-based rejection is business exception.

Preserve the type and detail of the current exception

* When the decision to stop retrying is reached, it is important to understand the role of the ‘Preserve the type and detail of the current exception’ checkbox.
* When this checkbox is ticked, the exception stage simple ‘re-releases’ the current exception and allows it to bubble upwards as if it had never been recovered in the first place. When this checkbox is not ticked the exception stage generates a new exception, and importantly, keeps the current exception alive. This means that by misusing the ‘preserve’ checkbox can lead to problems.

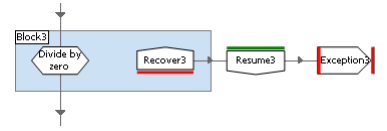


* The current exception is released when the ‘Preserve…’ checkbox is ticked.
* A new exception is generated when the ‘Preserve…’ checkbox is not ticked.

**Problems Example:**

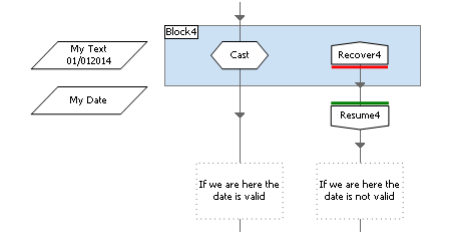


* In this example, the page falls into an exception loop. This is because the Resume stage has neutralized the original exception and the Exception stage generates a new one, which in turn is caught by the Recover stage.
* Add a block as shown below. See that it prevents the infinite loop by isolating the Recover stage so that is only has responsibility for exceptions originating inside the block.



**Casting**

* The problem with casting is that it can go wrong if there is no ‘fit’ between the data types, e.g. “£12.34CR” will not ‘fit into’ a number data item because of the non-digit characters. Blue Prism does provide a number of functions to test values (e.g. IsNumber) but often exception handling is used to test the compatibility of a value.



* Above, the failed cast operation is handled with more control. This kind of ‘exploratory’ casting is often a good way to validate data. Yes, there are other ways to check, but because the ‘fit’ required for a cast has to be exact, using exception handling like this is often the safest strategy.

# Utilities

## Clean Work Queue

This Utility could be used before populate the work queue in order to ensure that the queue does not have any pending item in it.

## Greetings

This Utility returns the formal greetings according the current local time.

Example:

‘Bom dia’, ‘Boa tarde’.