|  |
| --- |
| HEB: DEVELOPMENT STANDARDS |

1. **REVISION SUMMARY SHEET**

Document Approval

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision Identifier** | **Date** | **Autor** | **Revision Description** |
| 1.0 | 09-Sept-2014 | Noé Martinez Mtz | Initial version |
| 1.1 | 29- Jul -2015 | Gustavo Morales Luna | Update in the section “TIBCO error management standards” |
| 1.2 | 26-Nov-2015 | Gustavo Morrales Luna | Add section “TIBCO Alert management standards” |
| 1.3 | 04-Feb-2016 | Ainee Padilla | Create the section “TIBCO Audit management standards”.  Update the section “TIBCO error management standards”. |

**Reference**

|  |  |
| --- | --- |
| **Document Name** | **Document Type** |
| Error Management Developer’s Guide | Error Management |
| TDN\_Error\_Handler.docx | Error Handler Technical Specification |
|  |  |
|  |  |
|  |  |
|  |  |

**Table of Contents**

[1. REVISION SUMMARY SHEET 1](#_Toc524427860)

[2. INTRODUCTION 4](#_Toc524427861)

[2.1 General Information 4](#_Toc524427868)

[2.2 Audience 4](#_Toc524427869)

[2.3 Related Documents 4](#_Toc524427870)

[3. TIBCO COMPONENTS AND NAMING CONVENTIONS 5](#_Toc524427871)

[3.1 BW Project 5](#_Toc524427877)

[3.2 Folders 5](#_Toc524427878)

[3.3 Process Names 6](#_Toc524427879)

[3.2.1 Process Starter 7](#_Toc524427886)

[3.2.2 Main Process 7](#_Toc524427887)

[3.4 Shared Resources 7](#_Toc524427888)

[3.4.1 Schema 7](#_Toc524427892)

[3.4.2 Namespaces 8](#_Toc524427893)

[3.4.3 Service Activity 8](#_Toc524427894)

[3.4.4 WSDL 8](#_Toc524427895)

[3.4.5 WADL 9](#_Toc524427896)

[3.4.6 BW Variables 9](#_Toc524427897)

[3.5 Adapters 10](#_Toc524427898)

[3.5.1 Adapter Configuration Instances 10](#_Toc524427900)

[3.5.2 Adapter Configuration 10](#_Toc524427901)

[3.5.3 Adapter Service 10](#_Toc524427902)

[3.5.4 DSN’s 10](#_Toc524427903)

[3.5.5 Appendix A 11](#_Toc524427904)

[3.5.6 Appendix B 13](#_Toc524427905)

[3.6 EMS Objects 14](#_Toc524427906)

[3.6.1 EMS Queues and Topics 14](#_Toc524427908)

[4. TIBCO DEVELOPMENT STANDARS 16](#_Toc524427909)

[4.1 General Development Standards 16](#_Toc524427911)

[5. TIBCO AUDIT MANAGEMENT STANDARDS 20](#_Toc524427912)

[5.1 General Audit Management Standards 20](#_Toc524427914)

[5.2 General TrackLogger Management Standards 20](#_Toc524427915)

[6. TIBCO ERROR MANAGEMENT STANDARDS 22](#_Toc524427916)

[7.1 General Error Management Standards 22](#_Toc524427919)

[7.2 Error Handling Services 27](#_Toc524427920)

[7. TIBCO ALERT MANAGEMENT STANDARDS 31](#_Toc524427921)

[8.1 Alert Management Standards 31](#_Toc524427923)

[8.2 Alerts in a BW Process 31](#_Toc524427924)

[8.3 Alert Schema and Queue 31](#_Toc524427925)

[8.3.1 Alert Schema 32](#_Toc524427926)

[8.3.2 Alert Queue 32](#_Toc524427927)

[8. ANNEXES 33](#_Toc524427928)

1. **INTRODUCTION**

## General Information

The purpose of the document is to provide development guidelines and best practices for new and experienced TIBCO developers.

## Audience

This document is intended to be study by TIBCO Developers, Architects and Project Managers who are responsible for developing integrations within HEB. The document assumes a basic level of expertise in Tibco platform.

## Related Documents

The following additional development resources to support best practice and is recommended:

**Document Conventions**

|  |  |
| --- | --- |
| Convention | Description |
| [ ] | Optional keywords or values are enclosed in [ ]. Do not type the [ ] symbols in your own code. |
| < > | Mandatory elements will be enclosed in angle brackets. |
| / | A forward slash ( / ) separating two elements means either one of the elements can be used. |

1. **TIBCO COMPONENTS AND NAMING CONVENTIONS**

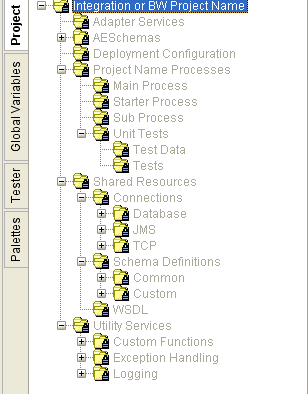
## BW Project

* Initial letter of each element shall be in upper case while other characters shall be in lower case (Upper camel-case style).
* All the characters in acronyms like SMS, PO should be handled the same (i.e. Sms, Po, etc.).
* Project Naming Rules/Guidelines based on different integration models:

|  |  |
| --- | --- |
| Integration Type | Description |
| P2P integrations | <Source/Domain><BusinessEntity>To<Destination/Domain>  Example: ExePoToPmm |
| Publisher Integrations | <Source/Domain ><BusinessEntity>Publisher  Example: PmmPoPublisher |
| Subscriber Integrations | <Destination/Domain><BusinessEntity>Subscriber  Example: BiCvSubscriber, StepPoSubscriber |
| SOAP WebService | Ws<BusinessEntity>Service  Example: WsProductService |
| REST WebService | Api<BusinessEntity>Service  Example: ApiProductService |

## Folders

Given below is the folder structure in the BW template:



The root folder name given as ‘Integration or BW Project Name’ in the above template will be changed to the project name while saving the project through TIBCO designer.

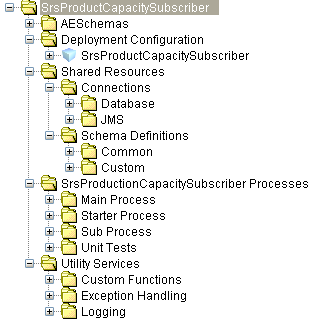
Repository:

mx02292@192.168.126.105:/git/IntegracionesTIBCO/Templates

The project name should be substituted for ‘Project Name Processes’ in the template above.

|  |  |
| --- | --- |
| ‘Project Name Processes’ folder name | |
| Process Folder Name: | <Project Name> Processes |
| Example: | StepProductHierarchySubscriber Processes |

An example of the project folder structure after renaming the ‘Project Naming Processes’ in the template is given below.



## Process Names

There are four types of processes, which are used in the template:

* Process Starter: Run the Main Processes.
* Main Processes: Error handler logic.
* Sub Processes: Process logic.
* Unit Tests: Test process.



### Process Starter

Standard Name for process starter:

|  |  |
| --- | --- |
| Process Starter Name | |
| Process Definition / Process Starter: | <BusinessEntity><ProcessingType>\_Starter |
| Example: | CustomerAddressSubscriber\_Starter |
| CustomerDataScheduler\_Starter |
| AsnPublisher\_Starter |

Each element of the process names are detailed in the table below:

|  |  |  |
| --- | --- | --- |
| Element | Description | Values |
| BusinessEntity | Refer sec 2.1 | Reference sec 2.1 |
| ProcessingType | Types of processing events. | Scheduler Subscriber Publisher |

### Main Process

Standard Name for Main Process:

|  |  |
| --- | --- |
| Main Process Name | |
| Process Definition / Main Process: | <Action>\_<BuisnessEntity>\_Main |
| Example: | Send\_CustomerData\_Main Update\_Asn\_Main |

## Shared Resources



### Schema

Class / Schema name.

|  |  |
| --- | --- |
| Classes / Schema: | <BusinessEntity>[List /Details/DetailList] |
| Example: | StoreList ( schema for the list of stores)  StoreDetails ( schema for the details of a particular store )  StoreDetailList ( schema for the list of stores and the details of each store )  RetailPriceChange |

### Namespaces

Namespaces definition:

|  |  |
| --- | --- |
| Namespace Value | |
| Namespace: | http://xmlns.heb.com/ei/<BusinessEntity> |
| Example: | [http://xmlns.heb.com/ei/RetailPriceChange http://xmlns.heb.com/ei/PurchaseOrder](http://xmlns.heb.com/ei/RetailPriceChangehttp://xmlns.heb.com/ei/PurchaseOrder) |

### Service Activity

Service Activity definition:

|  |  |
| --- | --- |
| Service Activity Name | |
| Service Activity: | <BusinessEntity>Service |
| Example: | MasterDataService ProductService StoreService |

Note: EndPoint name of the service activity should be same as the service activity name.

### WSDL

The WSDL should be named same as the Service activity.

|  |  |
| --- | --- |
| WSDL Name | |
| WSDL | <BusinessEntity>Service |
| Example: | HebCompetitorService  ProductService |

The request and reply messages should be named as follows.

#### Message name definition:

|  |  |
| --- | --- |
| Message Name | |
| Message | <Action>\_<BusinessEntity>[List/DetailList/Details]\_<Request/Reply> |
| Example: | Get\_BdmDetailList\_Request Get\_BdmDetailList\_Reply |

#### Port type name definition:

|  |  |
| --- | --- |
| Port Type Name | |
| PortType | <BusinessEntity>Service\_PortType\_[Version No] |
| Example: | HebCompetitorService\_PortType\_1 HebCompetitorService\_PortType\_2 ProductService\_PortType ProductService\_PortType\_1 ProductService\_PortType\_2 |

#### Operation name definition:

|  |  |
| --- | --- |
| Operation Name | |
| Operation | <Event><BusinessEntity><List/Details/DetailList>[By][MandatoryInputParameter] |
| Example: | getBdmDetailList getStoreDetailList getProductCasepackDetailsByItemId insertStoreTrendDetails updateStoreTrendDetails deleteStoreTrendInfo |

### WADL

The WADL should be named same as the Project Name.

### BW Variables

Name definition for BW Variables:

|  |  |  |
| --- | --- | --- |
| Category | Prefix | Comments |
| Varying with Environment (Migration Variables) | MIG\_ | These variables are assumed to change from one environment to another (.e.g. DEV, STAGE, PROD) such as the default global variable ENV, user names, connection strings. |
| Global Across Project | GLB\_ | These variables are assumed to be globally static across the project and don’t vary from one migration environment to another.  e.g. GLB\_MAX\_ROWS  e.g. GLB\_TIMEOUT |
| Configuration Variables | CFG\_ | These variables may or may not vary from one migration environment to other. They relate to configuration/optimization parameters. |
| Service Level Variables | SVC\_ | Set of values given in the properties file of the environment. |

## Adapters



### Adapter Configuration Instances

### Adapter Configuration

|  |  |
| --- | --- |
| Adapter Configuration Name | |
| Adapter Configuration: | <Source/Destination><BusinessEntity>\_<adapterType>\_Adapter |
| Example: | StepInventoryAdjustment\_TD\_Adapter  ExePo\_ADB\_Adapter  HebADomCustomerData\_ADB\_Adapter |

### Adapter Service

|  |  |
| --- | --- |
| Adapter Service Name | |
| Adapter Service | <BusinessEntityDetail>\_<Publication/Subscription> |
| Example: | CustomerAddress\_Publication  CustomerJobData\_Publication  PurchaseOrder\_Subscription |

Check appendix A and B.

### DSN’s

DSN name definition:

Data sources names. ODBC.

|  |  |
| --- | --- |
| DSN Name | |
| DSN | <Source/Destination> |
| Example: | EXE  PMM  STEP  ATENEA |

Check appendix A and B.

### Appendix A

|  |  |
| --- | --- |
| AP | Full Name |
| ADMVALES | SISTEMA DE ADMON DE VALES |
| BNSCENTER | BUSINESS CENTER |
| CFDBASQUE | CFD BASIQUE |
| CGSPLANNING | COGNOS PLANNING |
| CTARETIRO | CARTA RETIRO |
| CTRMORRALLA | CONTROL DE MORRALLA |
| DETALLISTA | DETALLISTA |
| HEBERNETICO | HEBERNÉTICO |
| HEBKIOSKO | HEBKIOSKO |
| HEBNET | HEBNET |
| KLYMA | KLYMA |
| KRONOS | KRONOS |
| MEGA | MEGA |
| NTSINGRESO | NOTAS DE INGRESO |
| PATTSY | PATTSY |
| PGOVALES | PAGO DE VALES |
| PSFFINANZAS | PROPLESOFT FINANZAS |
| PSFPROFILE | PEOPLE PROFILE |
| PSFRH | PEOPLESOFT RH |
| RELOCO | RELOCO |
| SCSFACTORS | SUCCESS FACTORS |
| SEGINTRANET | SEGURIDAD DE SISTEMAS INTRANET |
| SICAJAS | SICAJAS |
| SUPERTARJETA | SUPERTARJETA |
| UNIVIRTUAL | UNIVERSIDAD VIRTUAL |
| VSRFACTURAELECT | VISOR DE FACTURA ELECTRONICA |
| ATNDASHBOARD | ATENEA DASHBOARD |
| ATNPROCESS | ATENEA PROCESS |
| ATNREPORT | ATENEA REPORT |
| DATAMINING | DATAMINING |
| HPSMREPORT | HPSM REPORT |
| INFORMATICA | INFORMATICA |
| QLIKVIEW | QLIKVIEW |
| TIBCO | TIBCO |
| VNDNEGOTIATION | VENDOR NEGOTIATION |
| ACCVIATAG | ACCESS VÍA / TAG |
| AUTPOG | AUTORIZATION POG |
| CMPSIM | COMPARES SIM |
| DATMANAGER | DATA MANAGER |
| EFFITEMASSMNT | EFFICIENT ITEM ASSORTMENT |
| EVTESPECIALES | EVENTOS ESPECIALES |
| EVTPLANNING | EVENT PLANNING |
| FLRPLANNING | FLOOR PLANNING |
| HEBADOM | HEB A DOMICILIO |
| HEBIMEAT | H-E-B iMeat |
| HEBMEXICO | HEBMEXICO |
| HEBUSINESS | HEBUSINESS |
| LGXCOPIENT | LOGIX-COPIENT |
| PEM | PEM |
| PLANFAR | PLANFAR |
| PMM | PMM |
| SPCPLANNING | SPACE PLANNING |
| TDLAMISTAD | TORNEO DE LA AMISTAD |
| ARTXBOLSA | ARTICULOS POR BOLSA |
| AUTHENTIC | AUTHENTIC |
| CFDMANAGER | CFD MANAGER |
| CTLMARKET | CENTRAL MARKET ORDERING PORTAL |
| FDS | FDS |
| FRAXINTERNET | FACTURACION POR INTERNET |
| HEBPOS | HEB POS |
| INFACT | IN-FACT |
| KSKFACTURACION | KIOSKO DE FACTURACION |
| MRACONTINUA | MEJORA CONTINUA |
| PGOSERVICIOS | PAGO DE SERVICIOS |
| PGOVARPANADERIA | PAGO VARIABLE PANADERIA |
| SISAUDITORIAS | SISTEMA DE AUDITORIAS |
| SISBASCULAS | SISTEMAS DE BÁSCULAS |
| SISVERIFICADORES | SISTEMA DE VERIFICADORES |
| STEPTIENDA | STEP TIENDA |
| SYMBOL | SYMBOL |
| APRTIENDAS | APERTURA DE TIENDAS |
| DISTRYLOGSCA | DISTRIBUCION Y LOGISTICA |
| EDI | EDI |
| EXESTREORDRS | EXE-STORE ORDERS |
| JDADEMAND | JDA DEMAND |
| JDAFULFILLMENT | JDA FULFILLMENT |
| PO | PURCHASE ORDER |
| SICOI | SICOI COMERCIO INTERNACIONAL |
| VOCOLLET | VOCOLLET-STORE ORDERS |

### Appendix B

|  |  |
| --- | --- |
| BP | Description |
| **ACC** | Accounting |
| **ASR** | Assortment |
| **BIN** | Business Intelligence |
| **CHG** | Cargo |
| **CRF** | Customer Return - Refund |
| **CST** | Cost |
| **CUR** | Currency - Exchange Rate |
| **DMD** | Demand |
| **DSC** | Discount |
| **INC** | Invoice |
| **INV** | Inventory |
| **ORG** | Organization |
| **PAY** | Payment |
| **PMG** | Purchase Order |
| **PRC** | Price |
| **PRD** | Product |
| **RBT** | Rebate - Procurement Income |
| **RCV** | Reception |
| **RPL** | Replenishment |
| **RTV** | Return to Vendor |
| **SLE** | Sale |
| **TRF** | Transference |
| **TXS** | Tax |
| **USR** | User - Partner |
| **VPC** | Vendor |

## EMS Objects



### EMS Queues and Topics

* Use uppercase for naming.

|  |  |
| --- | --- |
| Name Definition | |
| Name: | <Company>.[Vertical].<Source/Destination>.<BusinessEntity>.[Purpose].[EndpointType].<Type> |
| Example: | HEB.COMMON.VENDOR.MAINTENANCE.TOPIC  HEB.MFG.AXAPTA.PURCHASEORDER.QUEUE  HEB.DMD.RXPRODUCT.MAINTENANCE.ADB.QUEUE  HEB.STORE. POS.PAYOUT.MAINTENANCE.TDB.QUEUE |

Each element of the queues and topics are detailed in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Element** | **Description** | **Values** | |
| <Company> | Defines the ownership of the data in the message. | HEB | |
| [Vertical] | Various business organization units within HEB grocery business |  | |
| **COMMON** | Apply to All Business Areas |
| **SC** | Supply Chain |
| **ME** | Merchandise |
| **SE** | Shopping Experience |
| **EN** | Enabling |
| **IC** | Information Center |
|  | |
| <Source / Destination> | The application which sends the message to downstream applications or receives the messages from source application. |  | |
| ADMVALES | SISTEMA DE ADMON DE VALES |
| BNSCENTER | BUSINESS CENTER |
| CFDBASQUE | CFD BASIQUE |
| CGSPLANNING | COGNOS PLANNING |
| CTARETIRO | CARTA RETIRO |
| A comprehensive list is given in Appendix A | |
| <BusinessEntity> | A name describing the business object. Wherever possible, object names should be based on the Enterprise Common Information Model (CIM). |  | |
| **ACC** | Accounting |
| **ASR** | Assortment |
| **BIN** | Business Intelligence |
| A comprehensive list of Busines Entities is given in Appendix B | |
| [Purpose] | Purpose gives the reason for sending the message. | e.g. Create, Update, Delete, Request, Response, Maintainenace, Status, Verify, InitialLoad. | |
|
|
| [EndpointType] | Adapters are endpoints to applications | ADB TDB  PS COM  LDAP FILE | |

1. **TIBCO DEVELOPMENT STANDARS**

## General Development Standards

* In the **TIBCO BusinessWorks and Adapters Deployment Repository Instance** section, under the Advanced tab of the deployment changed the Username/Password to **deployadm**. This is the user account which developers should use when deploying integration in DEV/CERT domains. If you do not use this account, your personal ID is used and will cause the integration not to start when your password changes.
* Using the source/target system and the integration type (P2P, Pub/Sub, Req/Reply), all code should be uploaded under the pertinent Application Management folder in the TIBCO Administrator domain based on the following rules:
  + 1. **Point-to-Point (P2P)** **:** code should reside under the folder of the source application (i.e. for STEPTransactionToPMM should reside under the “Mexico/STEP” folder).
    2. **Publish/Subscribe :** the publisher code should reside under the folder of the source system and the subscriber code should reside under the target application’s category folder. For example, for the PMMPurchaseOrderPublisher the code should reside under the “PMM” category folder. For HEBusinessPurchaseOrderSubscriber, the code should be stored under the “Mexico/HEBusiness” folder.
    3. **Request/Reply :** All SOAP/HTTP web services have a common location and should be stored under the “Mexico/Technical Services” folder.
  + Generate Service Instance name with meaningful name for deploy:
    1. Service Instance name: Project Name(view project name standard)
* For project with instances Stores name the Service Instances with the following nomenclature: [Project Name] **–** [Store Number]
  + 1. **High-Level Project Standards**
* Always use the latest version of the BW Project Template from XML Canon, which also contains the latest version of the ErrorProcess and AuditProcess. Do not use copies of other processes, as there is a chance that an existing process may not have been built off of the most recent BW project template. Once logged into XML Canon, click the “Browse” button and select “Categories”. You’ll find the BW template under the Mexico > Template category, according its design pattern:
* BW Project Name P2P or Publisher
* BW Project Name Subscriber
* BW Project Name Service
* Refer to the Naming Conventions and Standards documentation on naming Publishers and Subscribers. Need to be consistent in the naming of ALL publishers and subscriber BW projects. For example, if you are working on an Item Publisher and Subscriber, then use ProductPublisher and ProductSubscriber, respectively. Be careful when using application-specific naming in publishers, especially if the source may be more than one application. Always use application names in subscribers. For example, if there is a potential that multiple applications will eventually subscribe to the same entity of data being published, instead of simply naming the subscriber ProductSubscriber, use HEBusinessProductSubscriber, EXEProductSubscriber, STEPProductSubscriber, etc. Valid suffixes are Pub/Sub or Publisher/Subscriber and Request/Reply.
* Projects need to be validated with the BW Designer validation tool prior to delivery of code for deployment to ensure that no errors exist.
* Using the source/target system and the integration type (P2P, Pub/Sub, Req/Reply), all code should be uploaded under the pertinent category folder in XML Canon based on the following rules:

1. Point-to-Point (P2P) – code should reside under the category folder of the source application (i.e. for STEPTransactionToPMM should reside under the “Mexico/STEP” folder).
2. Publish/Subscribe – the publisher code should reside under the category folder of the source system and the subscriber code should reside under the target application’s category folder. For example, for the PMMPurchaseOrderPublisher the code should reside under the “Mexico/PMM” category folder. For HEBusinessPurchaseOrderSubscriber, the code should be stored under the “Mexico/HEBusiness” category folder.
3. Request/Reply – All SOAP/HTTP web services have a common location and should be stored under the “Mexico/Technical Services” category folder.
   * 1. **Using Global Variables**

* In global variables, change /Connections/Database/DatabaseName to the actual application that owns the database (i.e. /Connections/Database/PMM, /Connections/Database/HEBusiness, etc.)
* Remove any additional groups and variables that are not needed from the BW project template.
* Ensure that all queues are created under the /Destinations/Queues group, and all topics under the /Destinations/Topics group.
* Even though a variable named after a particular transaction may exist under two different groups, /Destinations/Queues and /Destinations/Topics, in the global variables, please distinguish each by using different naming (i.e. DisplayCountQueue, DisplayCountTopic).
* Topic names need to end with suffix, “.TOPIC”, just as queue names end with “.QUEUE”, and underscores ( \_ ) should not be used in queue names. Periods ( . ) should be used to separate wording.
* When using string variables, do not apply double quotes ( “ ) around the values. This is interpreted by BW as part of the actual string. Also, avoid using spaces.
* Remove all “no value” default settings for any string variables by changing it to an empty string.
* Global variables should always be defined as deployment level variables, by having the checkbox in the “Deployment” column enabled!
  + 1. **Working With Processes & Services**
* All projects should have a process Starter folder. The Starter should include the activity which invokes the process, call the main process, and then the sending activity, or simply the End activity. The Starter would also include the ErrorProcess activity. The Main process should perform the bulk of the work, which may call on smaller functions of one to many sub-processes.
* Please ensure that all process names follow a consistent naming convention where caps is used to reflect a new word, and fully name as much of the words in the naming as possible. You are free to name your processes based on the specific function performed.
* Acknowledge Mode in the JMS Queue Receiver of the Starter process needs to be set to “Client”. A Confirm activity needs to be added prior to the End activity to confirm the message to “JMS Queue Receiver”. If the service is request/reply, use Auto-Confirm.
* Label all transitions which are not Success by default.
* All explicit transitions should exist from an activity (i.e. “Success With Matching Condition”, “Error”, “Success”, “Otherwise”).
* All JMS and JDBC activities should contain global variables for the parameters which are identified in the template for all BW projects. For example, Timeout and Maximum Rows should be mapped from the global variables, using GLB\_TIMEOUT and GLB\_MAX\_ROWS, respectively.
* Ensure that the User Name and Password, in addition to the JNDI Username and JNDI Password in the JMS Connections are all mapped with the global variables, respectively, provided in the BW template global variables. Be sure to select the “Password” type in the global variables for variables that should be encrypted. These are different credentials, in some cases. The “Use JNDI for Connection Factory” checkbox needs to always be enabled in the JMS Connections for the deployment to initialize the process and connect to JMS.
* For any activity (i.e. HTTP Connection, TCP Connection, Timer, etc.) where global variables are applicable, please create and map them to the corresponding activity’s configurations, especially if values are subject to change throughout migration.
* In all Adapter Configurations, if used, global variables should be mapped where parameters may require change throughout migration, just as with JDBC parameters. There should be very few parameters, if any, that are hard-coded when applying configuration parameters for any type of connection.
* In all processes using a JMS Queue Receiver as the starter activity, under the Configuration tab, Message Type should be set to get messages as “Text” from the queue. If you are getting an XML message from the queue, you should receive the message as “Text”, and in the main process (to which the string is mapped) use the XML Parse activity to parse the message. This prevents any invalid XML messages from causing a bottleneck in the integration at the JMS Queue Receiver activity when an exception is thrown.

**IMPORTANT NOTE:**

For sending emails functionality from Mexico Integrations please consider these VERY IMPORTANT notes:

**DOMAIN :** The Domain to be used in Mexico Integrations must be **mail.heb.com.mx**, (this has been already approved by Infrastructure team).

**FROM :** For UNIT TESTING purposes “FROM” account must be the tester’s personal account.

For CERTIFICATION AND PRODUCTION environments “FROM” account must be **tibco@hebmex.com.**

**TO :** For UNIT TESTING purposes “TO” account must be the tester’s personal account.

For CERTIFICATION AND PRODUCTION environments “TO” account must be [IntegracionesTIBCO@hebmex.com](mailto:IntegracionesTIBCO@hebmex.com)

* + 1. **Managing Schema Definitions**
* The Error Management schemas should always be included, as they are contained within the template. Make sure that you do not re-create a schema with attributes common to an existing schema that may already exist under another name.

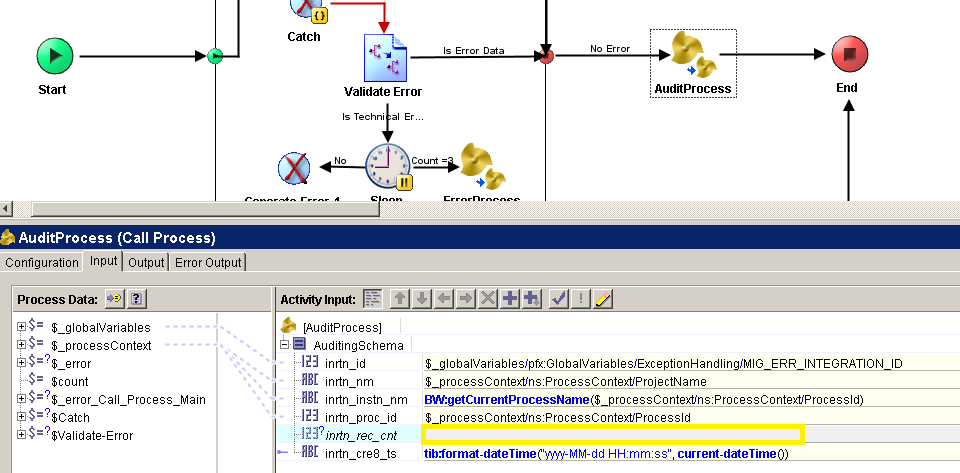
1. **TIBCO AUDIT MANAGEMENT STANDARDS**

## General Audit Management Standards

The Audit is the process of save a record of every message or data that pass throw the integration successfully. It allows to the support team identify specific data and dates of processing, in order to help in the solution of problems regarding to the integration.

It is strongly recommended to use the Audit process in integrations with transactional information, but it is optional and depends of what level of record is needed. It’s included by default in the BW project templates we use.

The audit schema data will be always mapped as following:

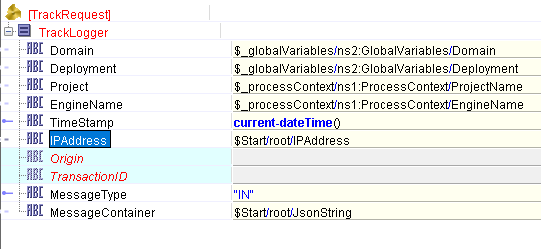


It’s important to fill in the field **inrtn\_rec\_cnt**, it will be the ID stored and the one we’ll use for look up. The audit process will send the messages to the audit queue (HEB.COMMON.EI.AUDIT.QUEUE).

## General TrackLogger Management Standards

Just as the Audit is the process of save a record of every message or data that pass throw the integration successfully the TrackLogger is the process of saving the message itself whatever the type of message. It allows to the support team identify the message that pass through in any given specific time/call, in order to help in the solution of problems regarding to the integration.

The TrackLogger schema data will be always mapped as following:



Where **IPAddress** is the Address of the machine that make the call.

Where **Origin** is the Source of the caller (PS, POS, PIP, etc.).

Where **TransactionID** is a unique value that identifies a specific call made to the service.

Where **MessageType** is a helper to identify the kind of message that we are going to save.

1. **TIBCO ERROR MANAGEMENT STANDARDS**

## 



## General Error Management Standards

Error Handler is the process of responding to the occurrence of anomalous or exceptional conditions requiring special processing.

When we are designing a TIBCO Integration to solve a business need, it is strongly recommended to consider the alternative use cases that might arise in a live production environment in order to properly handle these exceptions.

The following are examples of alternative use cases:

* Bad data inputs (string instead of numbers).
* Missing data inputs.
* Common infrastructure failures (network outages, non-responding data base engines).

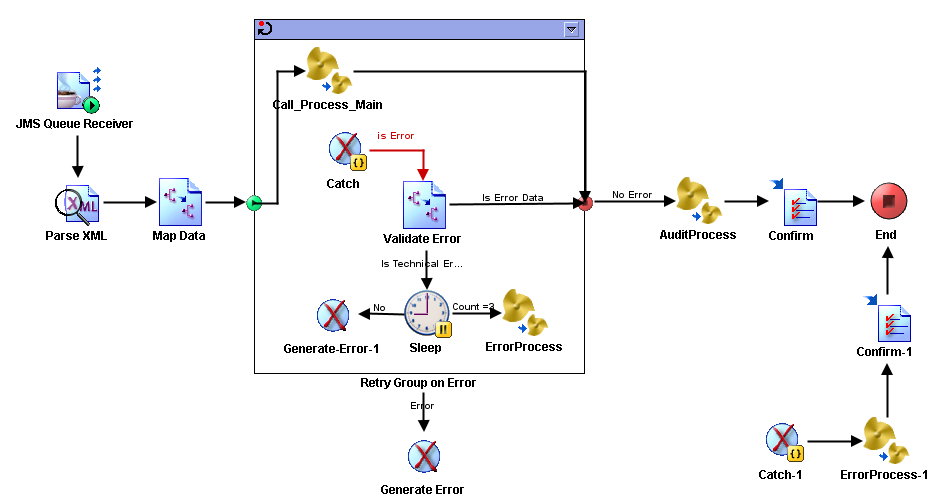
As a team, we have agreed to adopt the following design patterns to handle exceptions:

* **Publish/Subscribe/P2P BW Process Starter:**

**Objective**: Encapsulate the main error management logic in a “Repeat-On-Error-Until-True” group in order to try to complete the integration flow within a configured number of tries and known data errors. If any unhanded exception arises anywhere inside the “main process”, the integration will be restarted until it reaches its maximum number of tries. In the last loop, the error information will be sent to the exception queue (HEB.COMMON.EI.EXCEPTION.QUEUE) and the starter process will be suspended. If known data errors happened, it can continue processing and keep the integration running up.

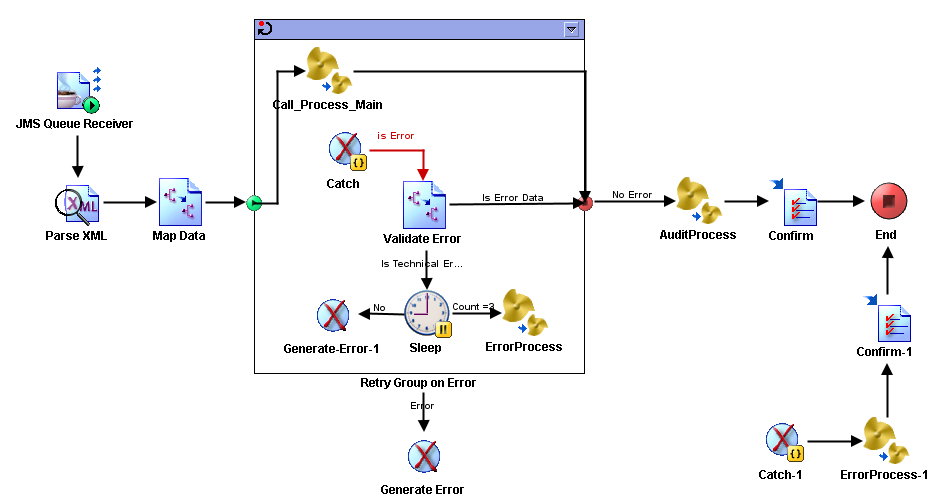
Every BW Process Starter that implements the exception control design pattern needs to have the following characteristics:

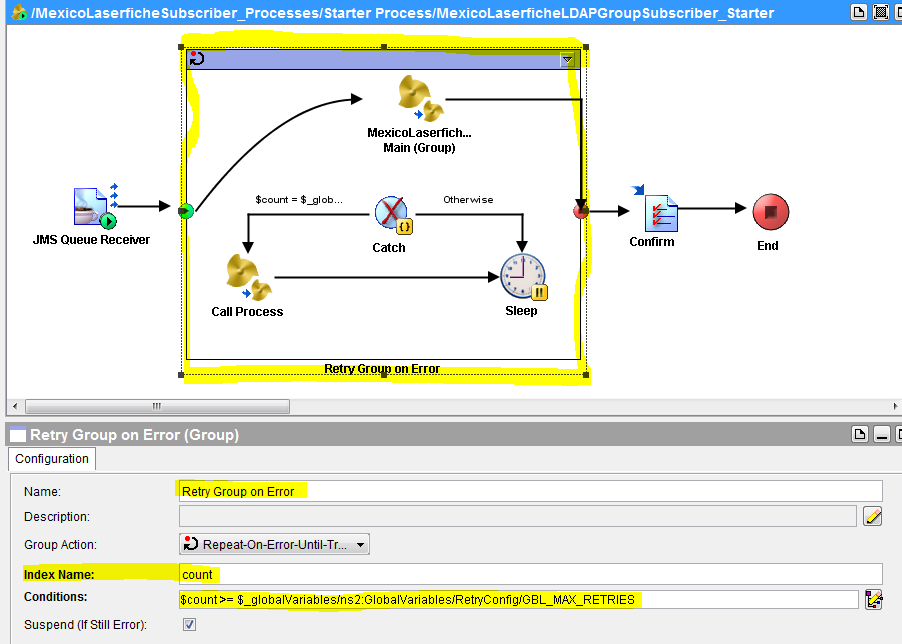
* The “Starter Process” will call to a “Main Process” that will contains the integration logic.
* The “Main Process” will be inside a “Repeat-On-Error-Until-True” group.



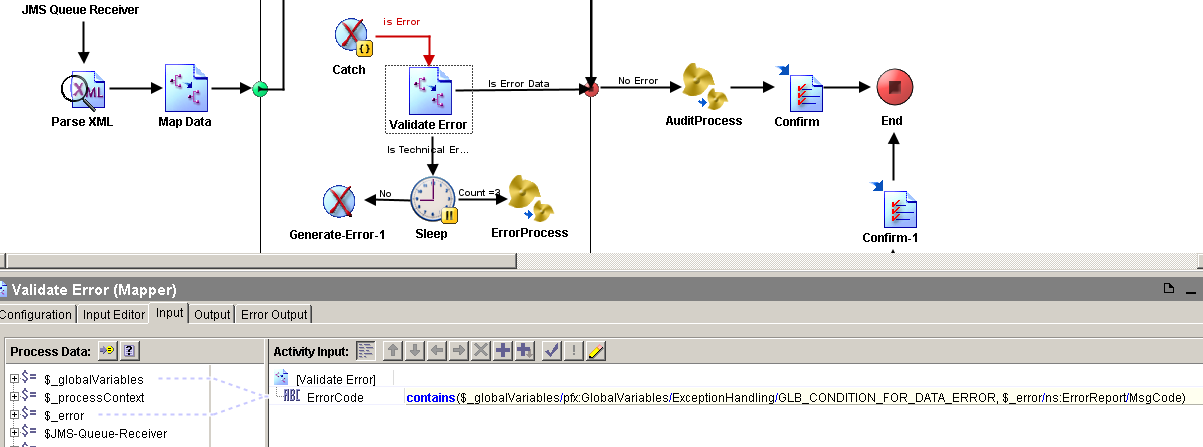
* The condition to stop the “Repeat-On-Error-Until-True” action will be when the “Index Name” reaches the maximum number of tries defined in a global variable. For example:

*$count >= $\_globalVariables/ns2:GlobalVariables/RetryConfig/GBL\_MAX\_RETRIES*

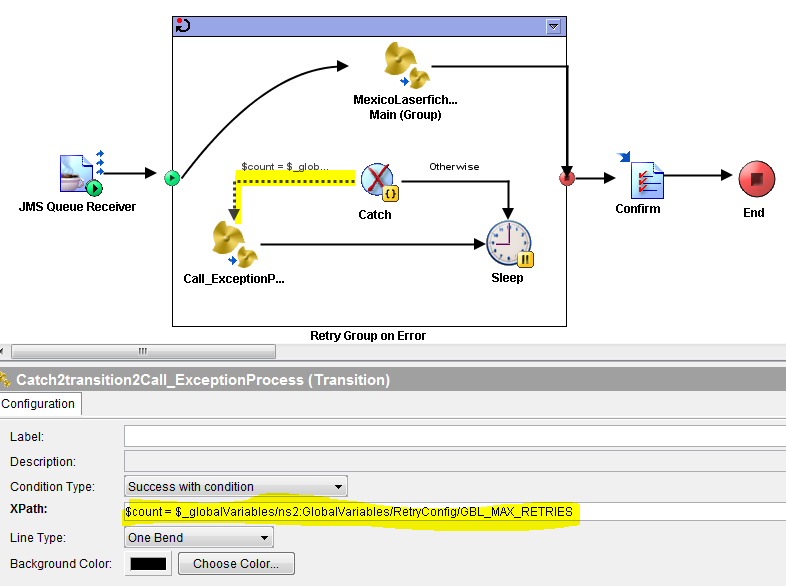
**



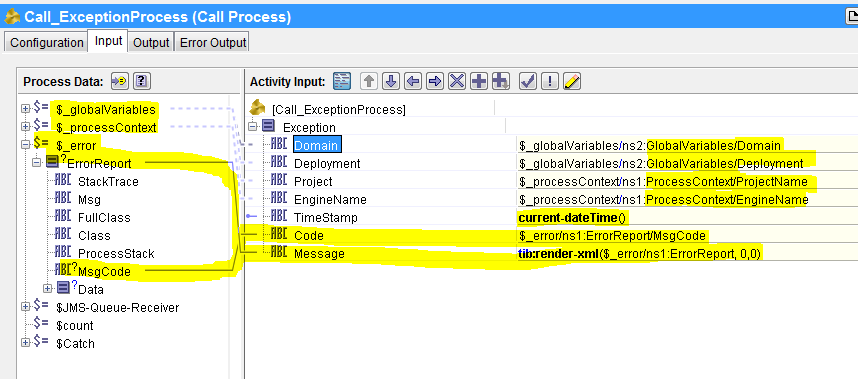
* Inside the “Repeat-On-Error-Until-True” will be a “Catch” activity that will listen to all unhandled exceptions that may arise in the execution of the “Main Process”
* The “Mapper” activity will test what kind of error happened: Technical or Data error. In Technical Error is expected to retry until Suspend the process if the problem persists. In Data Error is expected to continue processing and discard the message or data in process. There’s a global variable defined for save all error codes that we know are caused by Data Errors (as well we can add more error codes there, as needed):



* The “ErrorProcess” [[1]](#footnote-1) activity will be call when the maximum number of tries has been reached:

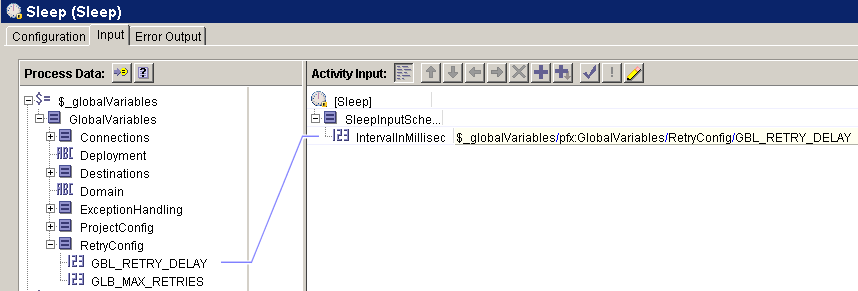


* The exception schema data will be always mapped as following:

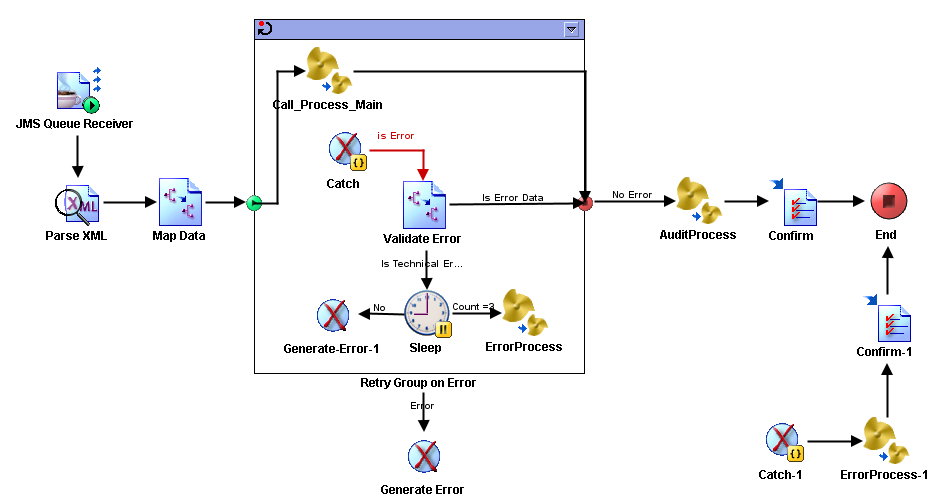


Please note that the “message” field is mapped to the “ErrorReport” data but it is rendered to a string with the usage of the “render-xml” string function. The “0,0” parameters in this function are defined to allow the function to include the XML headers and to disable the “pretty XML” format.

* The “Sleep” activity will be connected to the “Catch” with a “Success with no matching condition” transition. The IntervalInMillisec parameter of the “Sleep” activity will be mapped to a global variable:



* The “Catch” outside of the “Repeat-On-Error-Until-True” group will get all the errors that happen in that context.

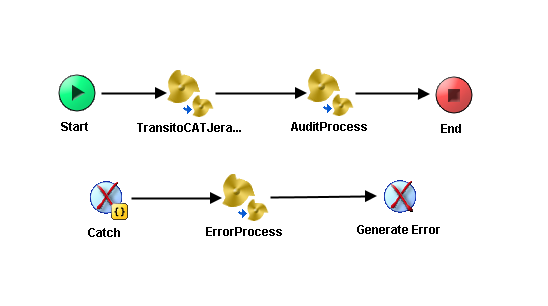


* **Soap Web Service BW Process:**

**Objective**: Allow the continuous operation of the BW process in case of any unhandled exception and response properly to the Soap Web Service requestor in a timely manner.

Due to the nature of the web services standard architecture, it is strongly recommended to implement a Fault Schema to respond the WS Requestor properly in case of an unhandled exception.

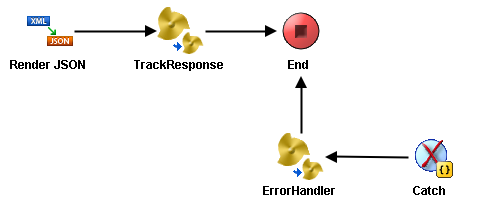
In the following example, there is a common web method called “InventarioCat” that can be requested by a particular process. If an unhandled exception arises within any particular process, the “Catch” will get the error information and the “ErrorProcess” will communicate the information to the Exception queue (HEB.COMMON.EI.EXCEPTION.QUEUE) according to the exception schema[[2]](#footnote-2) and the “Generate Error” will send a fault response to the client according to the fault schema defined in the web service message.

****

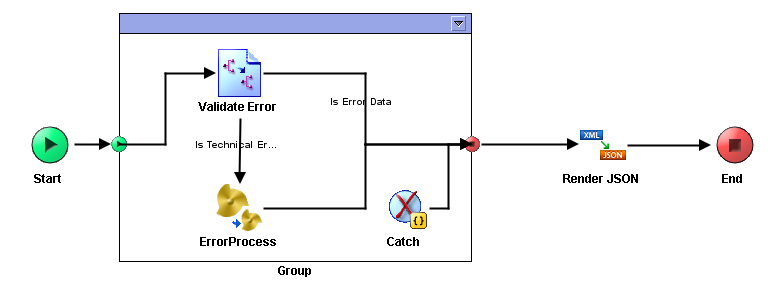
* **REST Web Service BW Process:**

**Objective**: Allow the continuous operation of the BW process in case of any unhandled exception and response properly to the REST Web Service requestor in a timely manner.

Main Process:



Error Handler Process



## Error Handling Services

This section describe how implements the error handling services, describe data base objects and services used. The services are included in the template for BE Projects in section: 3.2.

* Error types

The use of the error types are defined in the following table:

|  |  |
| --- | --- |
| **VALUE** | **DESCRIPTION** |
| Functional | Use when the service was stopped and the problem is a functional rule error that require some manual corrective action to keep on the data integrity. |
| Technical | Use when the service was stopped for any type of connection (eg. BD, FTP, HTTP, etc.) |
| Data | Use when the service was stopped when trying to process an incorrect data type (null, incorrect data types) |

* Error categories

The use of the error categories are defined in the following table:

|  |  |
| --- | --- |
| **VALUE** | **DESCRIPTION** |
| High | Use this category when the error require a high level of attention for resolve the error and the service cannot process correctly the information. |
| Medium | Use this category when the error require a medium level of attention to resolve the error. |
| Low | Use this category when the error require a low level of attention to resolve the error. |

* Error codes

|  |  |  |
| --- | --- | --- |
| **CODE** | **TYPE OF ERROR** | **ACTION** |
| BW-JMS | TECHNICAL | Send mail to Operator and Tibco Team. |
| BW-ENGINE | FUNCTIONAL | Send mail to correspondig Area.  eg. Support Supply Chain, Support Shopping, Support Merchandising y Support Enabling- |
| Others | DATA | Send mail to correspondig Area.  eg. Support Supply Chain, Support Shopping, Support Merchandising y Support Enabling. |

Error handling data base configuration:

Is necessary to configure the next database tables according to necessity for assign the errors to corresponding users or groups of users for solution, locate the tables in the server: SDI201080.

* TIB\_USER: this table contain the configuration for the users who receive a mail with the error or errors.

|  |  |
| --- | --- |
| Table name: [dbo].[TIB\_USER] | |
| Data types | Description |
| [ID\_USER] [int] IDENTITY(1,1) NOT NULL | User Id |
| [NAME] [varchar](50) NULL | User name |
| [EMAIL] [varchar](50) NULL | User email |
| CONSTRAINT [PK\_TIB\_USER] PRIMARY KEY CLUSTERED | Relation field |

* TIB\_GROUP: this table contain the configuration for the users groups who receive a mail with the error or errors.

|  |  |
| --- | --- |
| Table name: [dbo].[TIB\_GROUP] | |
| Data types | Description |
| [ID\_GROUP] [int] IDENTITY(1,1) NOT NULL | Group Id |
| [GROUP\_NAME] [varchar](50) NULL | Group name |
| [PROJECT\_NAME] [varchar](50) NULL | Proyect name |
| CONSTRAINT [PK\_TIB\_GROUP] PRIMARY KEY CLUSTERED | Relation field |

* TIB\_USER\_GROUP: this table contain the configuration for the association between user, groups and services.

|  |  |
| --- | --- |
| Table name:[dbo].[TIB\_USER\_GROUP] | |
| Data types | Description |
| [ID] [int] IDENTITY(1,1) NOT NULL | Identity id |
| [ID\_USER] [int] NULL | User id |
| [ID\_GROUP] [int] NULL | Group id |
| CONSTRAINT [PK\_TIB\_USER\_GROUP] PRIMARY KEY CLUSTERED | Relation field |

* TIB\_LOG: this table store the audit log.

|  |  |
| --- | --- |
| Table name: [dbo].[TIB\_LOG] | |
| Data types | Description |
| [Run\_Id] [int] IDENTITY(1,1) NOT NULL | Run Id |
| [INTEGRATION\_ID] [int] NULL | Integration Id |
| [INTEGRATION\_NAME] [varchar](250) NULL | Integration name |
| [INSTANCE\_NAME] [varchar](250) NULL | Instance name |
| [PROCESS\_ID] [varchar](250) NULL | Proces ID |
| [RECORD\_COUNT] [int] NULL | Number of process running |
| [RUN\_DATE][DateTime] NULL | Run date |

* TIB\_LOG\_ERROR: this table store the error log.

|  |  |
| --- | --- |
| Table name: [dbo].[TIB\_LOG\_ERROR] |  |
| Data type | Description |
| [Run\_Id] [int] IDENTITY(1,1) NOT NULL | Run Id |
| [Error\_Date] [smalldatetime] NULL | Error date |
| [Severity] [nvarchar](50) NULL | Error severity |
| [ServiceName] [varchar](250) NULL | Service name |
| [ErrorCode] [varchar](50) NULL | Error code |
| [DescriptionError] [varchar](4000) NULL | Error description |
| [MessageError] [text] NULL | Error message |
| [att1] [varchar](250) NULL | Additional data |
| [att2] [varchar](250) NULL | Additional data |

For more detail, check the document: TDN\_Error\_Handler.docx.

1. **TIBCO ALERT MANAGEMENT STANDARDS**

## Alert Management Standards

Alerts handling is the process to notify the data owners when certain (known) conditions arise in a live production environment and the usage of alternative use cases is required to properly finish the running processes.

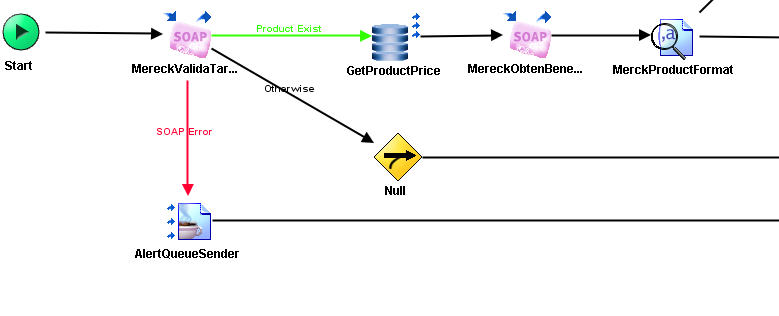
An example of known conditions that might require the usage of alerts are:

* Missing required data.
* Invalid data.
* Unavailability of services and /or infrastructure.
* Business rule violation.

## Alerts in a BW Process

**Objective**: Communicate the appropriate users (data owners) when certain known conditions are met that prevent the integration to complete its normal use case.

In the following example, an alert is sent to the process owner when there is an error consuming a web service.



## Alert Schema and Queue

## Alert Schema

<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">

<xs:element name="Alert">

<xs:annotation>

<xs:documentation>Comment describing your root element</xs:documentation>

</xs:annotation>

<xs:complexType>

<xs:sequence>

<xs:element name="Project" type="xs:string">

<xs:annotation>

<xs:documentation>Nombre del proyecto</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="EngineName" type="xs:string">

<xs:annotation>

<xs:documentation>Nombre de la instancia del servivio</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="TimeStamp" type="xs:string">

<xs:annotation>

<xs:documentation>Fecha y hora de la alerta</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="Message" type="xs:string">

<xs:annotation>

<xs:documentation>Descripcion de la alerta</xs:documentation>

</xs:annotation>

</xs:element>

<xs:element name="CompleteRecord">

<xs:annotation>

<xs:documentation>Mensaje o registro completo que originó la alerta</xs:documentation>

</xs:annotation>

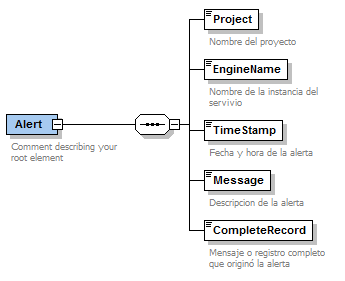
</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>



## Alert Queue

|  |  |
| --- | --- |
| EMS Server: | lpmextibadm01a.ad.heb.com.mx:7021 |
| Queue Name: | HEB.COMMON.EI.ALERT.QUEUE |

1. **ANNEXES**

1. The ErrorProcess is by default inside the BW Project Name P2P or Publisher/BW Project Name Subscriber template, you can find the last version in XMLCanon (<http://xmlcanon.heb.com:8080/Mexico/Template>) . [↑](#footnote-ref-1)
2. The ErrorProcess is by default inside the BW Project Name Service template, you can find the last version in XMLCanon (<http://xmlcanon.heb.com:8080/Mexico/Template>) [↑](#footnote-ref-2)