Flat view (UML)

Retry on SystemError
SystemError
BusinessRuleException
Success
Success
Success
No Data to Process
New Transaction
SystemError
Success
BusinessRuleException
SystemError

<<Framework Layer>>
wbMain

Global Variables

+ TransactionItem: QueueItem
+ TransactionData: DataTable
+ Config: Dictionary<string, object>
+ SystemError: Exception
+ BusinessRuleException: BusinessRuleException
+ wbMain_Type: String = in_wbType
+ wbMain_Level: Int32 = in_wbLevel
+ wbMain_Parent: String = in_wbParent
+ wbMain_Audit: List<Dictionary<string, object>> = New List(of Dictionary(Of String, Object))
+ wbMain_StartTime: DateTime = now()
+ SystemReserved: New Dictionary(Of String, Object) From { {"isQueueItem", False}, {"TransactionNumber", 1}, {"RetryNumber", 0}, {"InitRetryNumber", 0}, {"RobotFail", ""} }
+ wMain_FinalExec: Boolean = in_wbFinalExec
+ wbMain_Path = "Rethrow"
+ wbMain_HandleError = if(not(string.IsNullOrEmpty(in_wbPath)), in_wbPath+"l"+wbMain_Type, wbMain_Type)

Arguments

+ in_wbType: String = "MainTask"
+ in_wbLevel: Int32 = 0
+ in_wbParent: String = string.Empty
+ in_wbFinalExec: Boolean= True
+ in_wbParentStart: DateTime = now()
+ in_TaskEnv: String
+ in_wbPath: String = string.empty
+ io_TransactionData: List<DataRow>
+ io_Audit: List<Dictionary<string, object>>

<<State Machine - Initialization State>>
<<Framework Layer>>
wbInitialisation

Local variables:
+ wbInit_Type: String = in_wbType+", "+"Initialization State"
+ wbInit_Level: Int32 = wbMain_Level+1
+ wbInit_Parent: String = wbMain_Type
+ wbInit_Audit: List<Dictionary<string, object>> = New List(of Dictionary(Of String, Object))
+ wbInit_StartTime: DateTime = now()
+ wbInit_FinalExec: Boolean = true
+ wbInit_Path = wbMain_Path+"l"+wbInit_Type
+ wbInit_HandleError = "Failed"

Try executing Components:

System - reserved

+ **InitAllSettings** (in_ConfigFile:"Data\Config.xlsx", in_ConfigSheets:{"Settings", "Workblocks", "Tasks", "Constants"}, out_Config:Config)

System - reserved
<<Business Process Layer>>

+ **CloseAllApplications** (
in_Config:Config, in_wbType:Config.Item("wbCloseAppsRecover_Type").ToString, in_wbPar
in_wbLevel:wbInit_Level+1, io_Audit:wbInit_Audit, in_wbFinalExec: wbMain_FinalExec AND wbInit_FinalExec, in_wbHandleError: "Rethrow", in_wbPath: wbInit_Path)

Business Process Layer Interface
<<Business Process Layer>>

+ **InitAllApplications** (in_Config:Config, in_wbType: Config.Item("wbInitAllApplications_Type").ToString, in_wbParent:wbInit_Type, in_wbLevel:wbInit_Level+1, io_Audit:wbInit_Audit, in_wbFinalExec: wbMain_FinalExec AND wbInit_FinalExec
, in_wbHandleError: "Rethrow",in_wbPath: wbInit_Path)

Log success

+ **wbLogging**(in_wbType:wbInit_Type, in_wbLevel:wbInit_Level, in_wbParent:wbInit_Parent, in_LogLevel:"Info", in_LogMessage:"Work Block "+ wbInit_Type.toString + " successful", in_wbStart:wbInit_StartTime, in_ChildAudit:wbInit_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:if(Config.Item("wbInit_SuppressSuccessful").ToString.trim.ToLower = "true", true, false), in_wbStatusSuccessful:true, in_wbFinalExec: wbMain_FinalExec AND wbInit_FinalExec, in_wbExceptionType: string.Empty, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbInit_HandleError, in_wbPath: wbInit_Path)

Catch all exceptions, log them, treat or rethrow

System.Exception: Log failure

+ **wbLogging**(in_wbType:wbInit_Type, in_wbLevel:wbInit_Level, in_wbParent:wbInit_Parent, in_LogLevel:if (wbInit_FinalExec AND wbMain_FinalExec, "Fatal", "Warn"), in_LogMessage:exception.ToString, in_wbStart:wbInit_StartTime, in_ChildAudit:wbInit_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:false, in_wbStatusSuccessful:false, in_wbFinalExec:wbMain_FinalExec AND wbInit_FinalExec, in_wbExceptionType: exception.GetType.Name, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbInit_HandleError, in_wbPath: wbInit_Path)

<<State Machine - Get Transaction Data State>>
<<Framework Layer>>
wbGetTransactionData

Local variables:
+ wbGetSetTransactionData_Type: String = in_wbType+", "+"Get Transaction Data State"
+ wbGetSetTransactionData_Level: Int32 = wbMain_Level+1
+ wbGetSetTransactionData_Parent: String = wbMain_Type
+ wbGetSetTransactionData_Audit: List<Dictionary<string, object>> = New List(of Dictionary(Of String, Object))
+ wbGetSetTransactionData_StartTime: DateTime = now()
+ wbGetSetTransactionData_FinalExec: Boolean = true
+ ShouldStop: Boolean
+ wbGetSetTransactionData_Path = wbMain_Path+"l"+wbGetSetTransactionData_Type
+ wbGetSetTransactionData_HandleError = "Failed"

Try executing Components:

System - reserved

+ **OrchestratorStop** ()

Business Process Layer Interface
<<Business Process Layer>>

+ **GetTransactionData**
(in_TransactionNumber:cint(SystemReserved.Item("TransactionNumber,")), in_Config:Con
io_TransactionItem:TransactionItem, io_TransactionData:TransactionData, in_RetryNumber:cint(SystemReserved.Item("RetryNumber,")), in_wbType:wbGetSetTransactionData_Type, in_wbParent:wbGetSetTransactionData_Parent, in_wbLevel:wbGetSetTransactionData_Level+1, io_Audit:wbMain_Audit, in_wbFinalExec: wbMain_FinalExec AND wbGetSetTransactionData_FinalExec, in_wbParentStart: wbGetSetTransactionData_StartTime, in_wbHandleError: "Rethrow", in_wbPath: wbGetSetTransactionData_Path)

Log success

+ **wbLogging**(in_wbType:wbGetSetTransactionData_Type, in_wbLevel:wbGetSetTransactionData_Level, in_wbParent:wbGetSetTransactionData_Parent, in_LogLevel:"Info", in_LogMessage:"Work Block "+ wbGetSetTransactionData_Type.toString + " successful", in_wbStart:wbGetSetTransactionData_StartTime, in_ChildAudit:wbGetSetTransactionData_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:if(Config.Item("wbGetSetTransactionData_SuppressSuccessful").ToS
= "true", true, false), in_wbStatusSuccessful:true, in_wbFinalExec: wbMain_FinalExec AND wbGetSetTransactionData_FinalExec, in_wbExceptionType: string.Empty, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbGetSetTransactionData_HandleError, in_wbPath: wbGetSetTransactionData_Path)

Catch all exceptions, log them, treat or rethrow

System.Exception: Log failure

+ **wbLogging**(in_wbType:wbGetSetTransactionData_Type, in_wbLevel:wbGetSetTransactionData_Level, in_wbParent:wbGetSetTransactionData_Parent, in_LogLevel: "Fatal", in_LogMessage:exception.ToString, in_wbStart:wbGetSetTransactionData_StartTime, in_ChildAudit:wbGetSetTransactionData_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:false, in_wbStatusSuccessful:false, in_wbFinalExec:wbMain_FinalExec AND wbGetSetTransactionData_FinalExec, in_wbExceptionType: exception.GetType.Name, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbGetSetTransactionData_HandleError, in_wbPath: wbGetSetTransactionData_Path)

<<State Machine - End Process State>>
<<Framework Layer>>
wbMain

Local variables:

Try executing Components:

System - reserved

Business Process Layer Interface
<<Business Process Layer>>

+ **CloseAllApplications** (
in_Config:Config, in_wbType:Config.Item("wbCloseAllApplications_Type").ToString, in_wbPa
in_wbLevel:wbMain_Level+1, io_Audit:wbMain_Audit, in_wbFinalExec: wbMain_FinalExec, in_wbHandleError: "Rethrow", in_wbPath: wbMain_Path)

Log success

+ **wbLogging**(in_wbType:wbMain_Type, in_wbLevel:wbMain_Level, in_wbParent:wbMain_Parent, in_LogLevel:"Info", in_LogMessage:"Work Block "+ wbMain_Type.toString + " successful", in_wbStart:wbMain_StartTime, in_ChildAudit:wbMain_Audit, io_ParentAudit:processAudit, in_SuppressLogging:false, in_wbStatusSuccessful:true, in_wbFinalExec: wbMain_FinalExec, in_wbExceptionType: string.Empty, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbMain_HandleError, in_wbPath: wbMain_Path)

Catch all exceptions, log them, treat or rethrow

System.Exception: Log failure

+ **wbLogging**(in_wbType:wbMain_Type, in_wbLevel:wbMain_Level, in_wbParent:wbMain_Parent, in_LogLevel: "Fatal", in_LogMessage:exception.ToString, in_wbStart:wbMain_StartTime, in_ChildAudit:wbMain_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:false, in_wbStatusSuccessful:false, in_wbFinalExec:wbMain_FinalExec, in_wbExceptionType: exception.GetType.Name, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbMain_HandleError, in_wbPath: wbMain_Path)

<<State Machine - Process Transaction State>>
<<Framework Layer>>
wbProcess

Local variables:
+ wbProcessTransaction_Type: String
+ wbProcessTransaction_Level: Int32
+ wbProcessTransaction_Parent: String
+ wbProcessTransaction_Audit: List<Dictionary<string, object>> = New List(of Dictionary(Of String, Object))
+ wbProcessTransaction_StartTime: DateTime
+ wbProcessTransaction_FinalExec: Boolean = true
+ wbProcessTransaction_Path = wbMain_Path+"l"+wbProcessTransaction_Type
+ wbProcessTransaction_HandleError = "Failed"

Try executing Components:

System - reserved

Business Process Layer Interface
<<Business Process Layer>>

+ **Process** (io_TransactionItem:TransactionItem, in_Config:Config, in_TransactionNumber:SystemReserved.Item("TransactionNumber").ToString, in_RetryNumber:SystemReserved.Item("RetryNumber").ToString, in_wbType:Config.Item("wbProcess_Type").ToString, in_wbLevel:wbProcessTransaction_Level+1, in_wbParent:wbProcessTransaction_Type, io_Audit:wbProcessTransaction_Audit, in_wbFinalExec: wbMain_FinalExec AND wbProcessTransaction_FinalExec, in_wbParentStart: wbProcessTransaction_StartTime, in_wbHandleError: "Rethrow", in_wbPath: wbProcessTransaction_Path)

Log success

+ **wbLogging**(in_wbType:wbProcessTransaction_Type, in_wbLevel:wbProcessTransaction_Level, in_wbParent:wbProcessTransaction_Parent, in_LogLevel:"Info", in_LogMessage:"Work Block "+ wbProcessTransaction_Type.toString + " successful", in_wbStart:wbProcessTransaction_StartTime, in_ChildAudit:wbProcessTransaction_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:if(Config.Item("wbProcessTransaction_SuppressSuccessful").ToStrin
= "true", true, false), in_wbStatusSuccessful:true, in_wbFinalExec: wbMain_FinalExec AND wbProcessTransaction_FinalExec, in_wbExceptionType: string.Empty, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbProcessTransaction_HandleError, in_wbPath: wbProcessTransaction_Path)

Catch all exceptions, log them, treat or rethrow

System.Exception: Log failure

+ **wbLogging**(in_wbType:wbProcessTransaction_Type, in_wbLevel:wbProcessTransaction_Level, in_wbParent:wbProcessTransaction_Parent, in_LogLevel:if (wbProcessTransaction_FinalExec AND wbMain_FinalExec, "Error", "Warn"), in_LogMessage:exception.ToString, in_wbStart:wbProcessTransaction_StartTime, in_ChildAudit:wbProcessTransaction_Audit, io_ParentAudit:wbMain_Audit, in_SuppressLogging:false, in_wbStatusSuccessful:false, in_wbFinalExec:wbMain_FinalExec AND wbProcessTransaction_FinalExec, in_wbExceptionType: exception.GetType.Name, in_wbParentStart: wbMain_StartTime, in_wbHandleError: wbProcessTransaction_HandleError, in_wbPath: wbProcessTransaction_Path)

Legend	
blue	input argument
red	output argument
purple	io argument