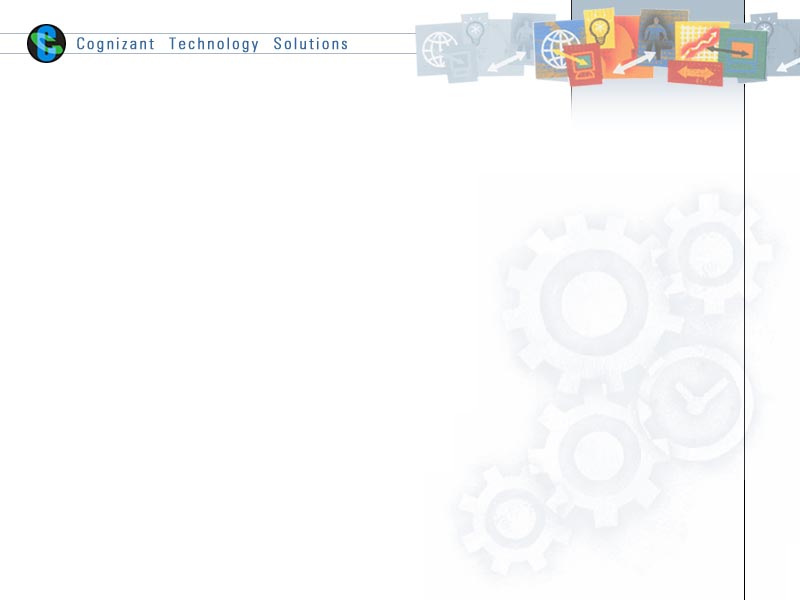
**ESB Integration**

**CLE12 (Logging & Exception Framework) Design Specification**

Version 1.0

****

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** |
| **Name** | Soumitra Bishnu |  |  |
| **Role** |  |  |  |
| **Signature** |  |  |  |
| **Date** | 20/Jan/2017 |  |  |

Table of Contents

[1.0 Executive Summary 6](#_Toc473106518)

[ Purpose and Audience 6](#_Toc473106519)

[ Document Scope 6](#_Toc473106520)

[ References 6](#_Toc473106521)

[ Definitions, Acronyms and Abbreviations 6](#_Toc473106522)

[2.0 Design Overview 7](#_Toc473106523)

[ Design Objectives 7](#_Toc473106524)

[ Assumptions and Dependencies 7](#_Toc473106525)

[ Rules of Thumb 7](#_Toc473106526)

[3.0 Application Design 8](#_Toc473106527)

[ Service Overview 8](#_Toc473106528)

[ Service Abstract 8](#_Toc473106529)

[ Service Architecture 9](#_Toc473106530)

[ Interfaces Overview 10](#_Toc473106531)

[4.0 CLE12LogManagement 01 11](#_Toc473106532)

[ Overview 11](#_Toc473106533)

[ Log Data Process Flow 11](#_Toc473106534)

[ Design Pattern and Transactional Scope 12](#_Toc473106535)

[ Performance 12](#_Toc473106536)

[ Availability 12](#_Toc473106537)

[ Authentication 12](#_Toc473106538)

[ Regulatory Requirements 12](#_Toc473106539)

[ Technical Components 12](#_Toc473106540)

[4..1 BW Components 12](#_Toc473106541)

[4..1.1. JMSChannel.process 12](#_Toc473106542)

[~~4..1.2.~~ ~~Handle\_Log\_Request Service~~ 13](#_Toc473106543)

[4..2 Data Structures 13](#_Toc473106544)

[4..2.1. Input 13](#_Toc473106545)

[4..2.2. Output 13](#_Toc473106546)

[4..3 Translation / Transformation 13](#_Toc473106547)

[ Exceptions 13](#_Toc473106548)

[4..1 System Exceptions 13](#_Toc473106549)

[4..2 User-defined Exceptions 14](#_Toc473106550)

[ Testing Requirements 14](#_Toc473106551)

[4..1 Unit Testing 14](#_Toc473106552)

[5.0 CLE12ExceptionManagement01 15](#_Toc473106553)

[ Overview 15](#_Toc473106554)

[ Exception Data Flow 16](#_Toc473106555)

[ Design Pattern and Transactional Scope 17](#_Toc473106556)

[ Performance 17](#_Toc473106557)

[ Availability 17](#_Toc473106558)

[ Authentication 17](#_Toc473106559)

[ Regulatory Requirements 17](#_Toc473106560)

[ Technical Components 17](#_Toc473106561)

[5..1 BW Components 17](#_Toc473106562)

[5..1.1. JMSChannel.process 17](#_Toc473106563)

[~~5..1.2.~~ ~~Handle\_Log\_Request Service.service~~ 18](#_Toc473106564)

[5..1.3. Scheduler.process 18](#_Toc473106565)

[5..1.4. MonitorResolutionEvents.process 18](#_Toc473106566)

[5..1.5. OnUpdateExceptionConfiguration.process 18](#_Toc473106567)

[5..1.6. OnGetExceptionRequest.process 18](#_Toc473106568)

[5..1.7. OnHandlingComplete.process 18](#_Toc473106569)

[5..1.8. NotificationConnector.process 18](#_Toc473106570)

[5..2 Data Structures 19](#_Toc473106571)

[5..2.1. Input 19](#_Toc473106572)

[5..2.2. Output 19](#_Toc473106573)

[5..3 Translation / Transformation 19](#_Toc473106574)

[ Exceptions 19](#_Toc473106575)

[5..1 System Exceptions 19](#_Toc473106576)

[5..2 User-defined Exceptions 20](#_Toc473106577)

[ Testing Requirements 20](#_Toc473106578)

[5..1 Unit Testing 20](#_Toc473106579)

[6.0 CLE12WorkFlow01 21](#_Toc473106580)

[ Overview 21](#_Toc473106581)

[ Workflow Data Flow 21](#_Toc473106582)

[ Design Pattern and Transactional Scope 22](#_Toc473106583)

[ Performance 22](#_Toc473106584)

[ Availability 22](#_Toc473106585)

[ Authentication 22](#_Toc473106586)

[ Regulatory Requirements 22](#_Toc473106587)

[ Technical Components 22](#_Toc473106588)

[6..1 BW Components 22](#_Toc473106589)

[6..1.1. StartExceptionWorkflow.process 22](#_Toc473106590)

[6..1.2. ReplayMessage.process 22](#_Toc473106591)

[6..2 Data Structures 23](#_Toc473106592)

[6..2.1. Input 23](#_Toc473106593)

[6..2.2. Output 23](#_Toc473106594)

[6..3 Translation / Transformation 23](#_Toc473106595)

[ Exceptions 24](#_Toc473106596)

[6..1 System Exceptions 24](#_Toc473106597)

[6..2 User-defined Exceptions 24](#_Toc473106598)

[ Testing Requirements 24](#_Toc473106599)

[6..1 Unit Testing 24](#_Toc473106600)

[7.0 CLE12Mediation01 25](#_Toc473106601)

[ Overview 25](#_Toc473106602)

[ Mediation Data Flow 25](#_Toc473106603)

[ Design Pattern and Transactional Scope 25](#_Toc473106604)

[ Performance 26](#_Toc473106605)

[ Availability 26](#_Toc473106606)

[ Authentication 26](#_Toc473106607)

[ Regulatory Requirements 26](#_Toc473106608)

[ Technical Components 26](#_Toc473106609)

[7..1 BW Components 26](#_Toc473106610)

[7..1.1. JMSChannel.process (*Log*) 26](#_Toc473106611)

[7..1.2. JMSChannel.process (*Exception*) 26](#_Toc473106612)

[7..2 Data Structures 27](#_Toc473106613)

[7..2.1. Input 27](#_Toc473106614)

[7..2.2. Output 27](#_Toc473106615)

[7..3 Translation / Transformation 27](#_Toc473106616)

[ Exceptions 27](#_Toc473106617)

[7..1 System Exceptions 27](#_Toc473106618)

[7..2 User-defined Exceptions 27](#_Toc473106619)

[ Testing Requirements 28](#_Toc473106620)

[7..1 Unit Testing 28](#_Toc473106621)

[~~8.0~~ ~~CLE12Administration01~~ 29](#_Toc473106622)

[~~~~ ~~Overview~~ 29](#_Toc473106623)

[~~~~ ~~CLE12Administration Data Process Flow~~ 30](#_Toc473106624)

[~~~~ ~~Design Pattern and Transactional Scope~~ 30](#_Toc473106625)

[~~~~ ~~Performance~~ 30](#_Toc473106626)

[~~~~ ~~Availability~~ 31](#_Toc473106627)

[~~~~ ~~Authentication~~ 31](#_Toc473106628)

[~~~~ ~~Regulatory Requirements~~ 31](#_Toc473106629)

[~~~~ ~~Technical Components~~ 31](#_Toc473106630)

[~~8..1~~ ~~BW Components~~ 31](#_Toc473106631)

[~~8..1.1.~~ ~~IntfUpdate\_ApplicationConfiguration-service~~ 31](#_Toc473106632)

[~~8..1.2.~~ ~~IntfCategory-service~~ 31](#_Toc473106633)

[~~8..1.3.~~ ~~IntfExceptionConfiguration-service~~ 32](#_Toc473106634)

[~~8..1.4.~~ ~~InterfaceConfigurationService~~ 32](#_Toc473106635)

[~~8..1.5.~~ ~~IntfProcedure-service~~ 32](#_Toc473106636)

[~~8..1.6.~~ ~~IntfRenderConfig-service~~ 32](#_Toc473106637)

[~~8..1.7.~~ ~~IntfSeverity-service~~ 33](#_Toc473106638)

[~~8..1.8.~~ ~~IntfType-service~~ 33](#_Toc473106639)

[~~8..1.9.~~ ~~IntfExceptions-service~~ 33](#_Toc473106640)

[~~8..1.10.~~ ~~Retrieve\_Log\_service~~ 33](#_Toc473106641)

[~~8..1.11.~~ ~~IntfACL-service~~ 33](#_Toc473106642)

[~~8..1.12.~~ ~~IntfChartInfoRetrieval-service~~ 34](#_Toc473106643)

[~~8..2~~ ~~Data Structures~~ 34](#_Toc473106644)

[~~8..2.1.~~ ~~Input~~ 34](#_Toc473106645)

[~~8..2.2.~~ ~~Output~~ 34](#_Toc473106646)

[~~8..3~~ ~~Translation / Transformation~~ 34](#_Toc473106647)

[~~~~ ~~Exceptions~~ 34](#_Toc473106648)

[~~8..1~~ ~~System Exceptions~~ 34](#_Toc473106649)

[~~8..2~~ ~~User-defined Exceptions~~ 35](#_Toc473106650)

[~~~~ ~~Testing Requirements~~ 35](#_Toc473106651)

[~~8..1~~ ~~Unit Testing~~ 35](#_Toc473106652)

[9.0 Audit and Logging 35](#_Toc473106653)

[10.0 Exception Handling 35](#_Toc473106654)

[11.0 Appendix A – Change Log 37](#_Toc473106655)

# Executive Summary

## Purpose and Audience

This contains the detail design specification of CLE12 framework services which will implement common logging exception handling framework in TIBCO BusinessWorks v6.x. This document is appropriate for the audience of the below groups.

* ESB Architect
* ESB Developers
* ESB Team

## Document Scope

Scope of this document is limited to design aspects of all implementations interacting with **CLE12 Framework.**

## References

|  |  |
| --- | --- |
| ***Document Title*** | ***Document Location*** |
| Requirements | N/A |
| Standards |  |
| Best Practices |  |

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| ***Term, Acronym*** | ***Definition*** |
| ESB | Enterprise service Bus |
| TIBCO | The Information Bus company |
| CLE | Common Logging and Exception |

# Design Overview

## Design Objectives

Below are few design objectives to implement **Common Logging and Exception framework (CLE12).**

* Handle Logs/exceptions in a standardized uniform structure across all enterprise applications and projects
* Provide Role based access to analyze/track the issue – application wise, process wise, and error wise.
* Support those applications who still use TIBCO CLE v2007.
* ~~Provide UI for easing off error search.~~
* ~~Provide dashboard to detail error information activity wise with graphical representation.~~

## Assumptions and Dependencies

|  |  |  |
| --- | --- | --- |
| S. No. | Assumptions | Dependencies |
| 1 |  |  |
| 2 |  |  |

## Rules of Thumb

|  |  |  |
| --- | --- | --- |
| S. No. | Focus Area | Rule of thumb |
| 1 | High availability | Load balanced instances over Active - passive setup |
| 2 | Integration pattern | Publish / Subscribe model to enable dynamic subscription of data. |

# Application Design

## Service Overview

Section describes list of services/interfaces involved for CLE 12 framework -

|  |  |  |  |
| --- | --- | --- | --- |
| ***Service Short ID*** | ***Service Name*** | ***Service Version*** | ***Service Type*** |
| CLE12LogMgmt01 | CLE12LogManagement | 01 | ~~SOAP/HTTP,~~ XML/JMS |
| ~~CLE12Admin\_01~~ | ~~CLE12Administration~~ | ~~01~~ | ~~SOAP/HTTP~~ |
| CLE12ExceptionMgmt01 | CLE12ExceptionManagement | 01 | ~~SOAP/HTTP,~~ XML/JMS |
| CLE12Workflow01 | CLE12Workflow | 01 | XML/JMS |
| CLE12Mediation01 | CLE12Mediation | 01 | XML/JMS |

## Service Abstract

In the enterprise level, application failures might occurs during runtime and leads to several risks or impacts to business down situation if they are not handled properly. Common Logging Error framework (CLE v12) addresses these issues by providing a common solution framework for logging and exception handling as enterprise services with a common standard structure format Exceptional Handling & Logging.

Enterprise applications use this CLE 12 framework services to manage Log and exception handling. For example, “CLE12LogManagement” service receives Log data in specific format from applications in either ~~SOAP/HTTP~~ or XML/JMS and then inserts into Log table to view/analyze application logs. Whereas “CLE12ExceptionManagement” receives exception data from applications and inserts into Exception table to view/debug application errors. Another service “CLE12WorkflowManagement” provides automated replay mechanism for error out transactions~~. “CLE12Administration” interface is used by CLE GUI application to view/manage log and exceptions. It also provides role based access to view/track errors. CLE12 provides dashboard functionality to provide detail information activity wise with graphical representation.~~

## Service Architecture



[**Note**: Currently in architectural landscape, primarily we are using JMS channel for logging and exception handling though framework supports both web service (SOAP/HTTP) and JMS channel.]

## Interfaces Overview

Section describes list of interfaces for CLE12 -

|  |  |  |
| --- | --- | --- |
| ***Interface Name*** | ***Interface Version*** | ***Description*** |
| CLE12LogManagement | 01 | Receives Log data from applications in SOAP/HTTP or XML/JMS format and inserts into CLE Log table |
| ~~CLE12Administration~~ | ~~01~~ | ~~Receives request from CLE12 GUI to perform following operations – Application Management, Managing Exception Category/Exception configuration/Exception procedures/Exception types/Exception severity~~ |
| CLE12ExceptionManagement | 01 | Receives Error data from applications in SOAP/HTTP or XML/JMS format and inserts into CLE Exception table |
| CLE12Workflow | 01 | Receives replay message from exception interface and sends back the message to applications(Both Store&Corp) for replay to happen |
| CLE12Mediation | 01 | Receives Log and exception data from applications in CLE v2007 format and transforms it to CLE v2012 format and send it to CLE12Exceptionmanagement process to process further to Exception DB. |

# CLE12LogManagement 01

## Overview

This interfaces under this module can receive log data in both ~~SOAP/HTTP~~ and XML/JMS format from applications. Once it receives Log data it invokes CLE DB Stored procedure to insert log data in CLE Log table. This log message can be viewed later for analysis purpose. Apart from Log table, this interface inserts data into “LOGALTKEY” table if AltKey value exists in the Log payload.

## Log Data Process Flow



***Fig: CLE Log Data Flow***

## Design Pattern and Transactional Scope

The interfaces under this module follows the request only pattern. The scope of its processing is receive Log data from applications through ~~SOAP/HTTP~~ or XML/JMS. The message processing used in CLE12LogManagement does guarantee if the message is not lost in the event of a failure when JMS channel interface is used.

## Performance

On a busy day, this JMS channel interface can receive around 100000 log message per hour from TIBCO applications (Corp&Store). To support this, JMS channel interface must be deployed in multiple instances where load will be balanced across instances as this has been built using EMS Queue.

## Availability

The interfaces under “CLE12LogManagement” module must be available 24 hours per day, 7 days per week, in a highly available environment. There will be no scheduled outages for this service.

## Authentication

N/A

## Regulatory Requirements

#### Data Protection

N/A

#### Data Retention

Log Data will persist in DB for only scheduled period to avoid overflowing table spaces in DB.

## Technical Components

### BW Components

### JMSChannel.process

This process receives XML/JMS Log data from applications. Basically applications publish Log data on a JMS Log topic (*<CompanyName>.ENTERPRISE.SR.CLE12.Log.PublishXml.V1\_0*) and then this topic is bridged to JMS Log queue (*<CompanyName>.ENTERPRISE.SR.CLE12.Log.PublishXml.V1\_0*). Now this CLE Log process receives Log data from this queue and invokes Log Stored Procedure “*INSERTLOG*” to insert Log data in Log table and along with it, “LOGALTKEY” also gets updated when Log data contains “AltKeyValues”.

### ~~Handle\_Log\_Request Service~~

~~This Log Web service can process Log data received in SOAP/HTTP format. So the applications directly invoke this Log service over SOAP/HTTP and then this process and invokes Log Stored Procedure “~~*~~INSERTLOG~~*~~” to insert Log data in Log table and along with it, “LOGALTKEY” table also gets updated when Log data contains “AltKeyValues”.~~

### Data Structures

### Input



### Output

* N/A

### Translation / Transformation

* N/A

## Exceptions

### System Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
|  |  |  |
|  |  |  |

### User-defined Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| CMN-05(XML validation error) | The Log XML data received from application doesn’t conform to Log schema | Check CLE “exceptionrec” table with application ID as “System” to get more details about XML validation and make sure application sends correct Log structure while sending log message |
| CMN-06(application ID not found) | The application ID in the log request doesn’t exist in the DB | Make sure the application id exists in “Applications” table |
| CMN-09(render id not found or invalid) | In the “InterfaceConfig” table, invalid value exists for corresponding “ApplicationID” in the log request | Make sure valid record exists in the “InterfaceConfig” table for corresponding “ApplicationdID” & “interaceID” |
| CMN-15(Interface ID not found) | No corresponding record found in the “InterfaceConfig” table for corresponding “ApplicationdID” and “interfaceID” in the log request. | Make sure valid record exists in “intefaceconfig” table corresponding “ApplicationdID” & “interaceID” mentioned in the log request |
| BW-JDBC-100039(database error) | DB connection timeout or DB driver not found or DB timed out | Make sure enough DB connections exists and DB is accessible from deployment machine |

## Testing Requirements

### Unit Testing



# CLE12ExceptionManagement01

## Overview

The interfaces under this module can receive error data in both ~~SOAP/HTTP~~ and XML/JMS format from applications. Once it receives error data it invokes CLE DB Stored procedure to insert error data in CLE Exceptionrec table. This error message can be viewed later for analysis purpose. Apart from Log table, this interface inserts data into “EXCEPTIONRECALTKEY” table if AltKey value exists in the Error payload.

## Exception Data Flow



***Fig: CLE Exception Data Flow***

## Design Pattern and Transactional Scope

The interfaces under this module follows the request only pattern. The scope of its processing is receive Error data from applications through ~~SOAP/HTTP~~ or XML/JMS. The message processing used in CLE12ExceptionManagement does guarantee if the message is not lost in the event of a failure when JMS channel interface is used.

## Performance

On a busy day, the JMS interface under this module can receive around 100000 error message per hour from <CompanyName> TIBCO applications (Store&Corp). To support this, JMS channel interface must be deployed in multiple instances where load will be balanced across instances as this has been built using EMS Queue.

## Availability

The interfaces under “CLE12ExceptionManagement” must be available 24 hours per day, 7 days per week, in a highly available environment. There will be no scheduled outages for this service.

## Authentication

N/A

## Regulatory Requirements

#### Data Protection

N/A

#### Data Retention

Exception Data will persist in DB for only scheduled period to avoid overflowing table spaces in DB.

## Technical Components

### BW Components

### JMSChannel.process

This process receives XML/JMS Error data from applications. Basically applications publish Error data on a JMS Exception topic (*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.PublishXml.V1\_0*) and then this topic is bridged to JMS Exception queue (*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.PublishXml.V1\_0*). Now this CLE Exception process receives error data from this queue and first retrieves *exceptionconfig* details from CLE DB for the exception received. Then it invokes exception stored procedure to insert exception data into *exceptionrec* table and *exceptionrecaltkey* tables with exception status as “*NEW*”. And then it sends email notification (through “*NotificationConnector*” process) and update exception status as “*NOTIFIED*”. Then it checks whether “*ResolvedDelayInterval*” and “*PhysicalprocedureName*” exists in the data fetched from exceptionconfig table. If exists then the process updates the exception status as “Scheduled” otherwise it invokes “*StartExceptionWorkflow*” process of *CLEWorkflow* module and update the exception status as “*PENDIND PROCEDURE*” .

### ~~Handle\_Log\_Request Service.service~~

~~This process works same way the above mentioned “JMSChannel” process executes but instead of receiving data in XML/JMS, this web service is consumer by applications to send Error data on SOAP/HTTP. The remaining data flow is same as mentioned above.~~

### Scheduler.process

This timer process executes at every specified interval to retrieve exception details with exception status as “*SCHEDULED*”. Then it fetches details from “*exceptionconfigview*” for each exception retrieved from earlier activity. Then it invokes “StartExceptionWorkflow” process of CLEWorkflow module and update the exception status as “*PENDIND PROCEDURE*”.

### MonitorResolutionEvents.process

This starter process receives event type as input and then retrieves exception with event type which came as input and exception status as “*SCHEDULED*”. Then it fetches details from “*exceptionconfigview*” for each exception retrieved from earlier activity. Then it invokes “*StartExceptionWorkflow*” process of *CLEWorkflow* module and update the exception status as “*PENDIND PROCEDURE*”.

### OnUpdateExceptionConfiguration.process

This starter process used to refresh Exception Configuration Cache which is used by CLE12Administration process.

### OnGetExceptionRequest.process

This starter process used to fetch exception details from “*excetptiondetailsview*” based on exception instance ID received as input from “*CLE12Administration*” processes.

### OnHandlingComplete.process

This starter process receives exception details (e.g. Exception instance ID, status as “Resolved” etc.) from “*StartExceptionWorkflow*” process of *CLE12Workflowmanagement* module and update exception status as resolved and then send exception details to replay workflow process(*ReplayMessage process*) before updating exception status as “REPLIED”.

### NotificationConnector.process

This process receives notification detail from notification queue from “JMSChannel” process to send error email notifications to users.

### Data Structures

### Input



### Output

* N/A

### Translation / Transformation

* N/A

## Exceptions

### System Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
|  |  |  |
|  |  |  |

### User-defined Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| XML validation error | The Exception XML data received from application doesn’t conform to Log schema | Check CLE exceptionrec table with application ID as “System” to get more details about XML validation and make sure application sends correct Exception structure while sending Exception message |
| DB Exception ("Records not found") | No corresponding value exists in “exceptionconfigview” for corresponding “applicationID” and “exception code” | Make sure valid record exists in “exceptionconfig” table for corresponding “applicationID” and “exceptioncode” in the exception request message. |
| DB Exception (BW-JDBC-100039” | Network Issue/DB time out/connection not found/driver not found/login time out | Check with network team/DB team and make sure DB is accessible from deployment machine |
| CMN-09(render id not found or invalid) | In the “InterfaceConfig” table, invalid value exists for corresponding “ApplicationID” in the log request | Make sure valid record exists in the “InterfaceConfig” table for corresponding “ApplicationdID” & “interaceID” |
| CMN-15(Interface ID not found) | No corresponding record found in the “InterfaceConfig” table for corresponding “ApplicationdID” and “interfaceID” in the log request. | Make sure valid record exists in “intefaceconfig” table corresponding “ApplicationdID” & “interaceID” mentioned in the log request |
| JMSConnectionError(CMN-14) | JMS Session Error/ JMS send exception | Make sure JMS server is up and running and has enough sessions for applications to connect. |
| Exception record not found( EXCP-02) | Error occurred while updating exceptionrec table with exception status for corresponding exceptionID | Verify exception id for which status in exceptionec table is going to be updated. |

## Testing Requirements

### Unit Testing



# CLE12WorkFlow01

## Overview

The interfaces under this module is used along with CLE12ExceptionManagement to replay errors for applications. It has two main starter process as mentioned below

## Workflow Data Flow



***Fig: CLE Workflow Data Flow***

## Design Pattern and Transactional Scope

The interfaces under this follow the request only pattern. The scope of its processing is receive replay data from applications through XML/JMS. The message processing used in CLE12Workflow does guarantee if the message is not lost in the event of a failure.

## Performance

On a busy day, the interfaces under this module can receive around 1000 error message per hour to replay. To support this, interface must be deployed in multiple instances where load will be balanced across instances as this has been built using EMS Queue.

## Availability

This interfaces under module “CLE12Workflow” must be available 24 hours per day, 7 days per week, in a highly available environment. There will be no scheduled outages for this service.

## Authentication

N/A

## Regulatory Requirements

#### Data Protection

N/A

#### Data Retention

N/A

## Technical Components

### BW Components

### StartExceptionWorkflow.process

This process receives XML/JMS Error procedure info data from JMS replay procedure queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Replay.Procedure.RequestXml.V1\_0*” published by application JMSChannel process of CLE12ExceptionManagement module. Then it transforms data to “*ProcedureCompletion*” schema and submit it on Error resolution queue.

### ReplayMessage.process

This process receives a *ResolutionReply* message from CLE12ExceptionManagement on queue “*<CompanyName>.ESB.Shared.Replay.V1\_0*”, examines the Shared Replay Config, and determines which one of these actions to take:

* + - Matching configuration found for a *Corp destination*
      * If replays have not been exhausted
        + Increments retry attempt count and places message back into the original CORP destination for replay
      * Otherwise, retries exhausted, places the message on the DLQ configured
    - Matching configuration found for a *Store destination*
      * Forwards to Store if retries not exhausted
      * Sends to CORP DLQ configured if retries exhausted
    - No matching configuration: message discarded

**Message Replayed**

1. **CORP: Message replayed to the Corp destination (or sent to the Corp Dead Letter Queue)**
   * The original CORP queue receiver gets the replayed message and re-processes it
2. **STORE: Message forwarded to the Store Replay Component (or sent to the Corp Dead Letter Queue)** 
   * The ResolutionReply is forwarded to the appropriate Store EMS via the routed topic ‘*<CompanyName>.ESB.CLE.Replay.PublishToStore.V1\_0’*

### Data Structures

### Input



### Output



### Translation / Transformation

* N/A

## Exceptions

### System Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
|  |  |  |
|  |  |  |

### User-defined Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| JMSConnectionError(CMN-14) | JMS Exception | Check error trace in both exception table and application log/exception email. Also verify payload to make sure correct data is being received. |
| InternalError (CMN-01) | Any generic exception | Check error trace in both exception table and application log/exception email. |

## Testing Requirements

### Unit Testing

# 

# CLE12Mediation01

## Overview

These interfaces provides another functionality where applications still use CLE2007 framework for logging and exception handling. These interfaces receive both Log and Exception request message (CLE2007 format) over JMS channel. This Log/Exception request message is then transformed to CLE Log/Exception format (CLE12Logmanagement/CLE12ExceptionManagement) and then message is published on Log/Exception queue to be processed further.

## Mediation Data Flow



***Fig: CLE12Mediation Data Flow***

## Design Pattern and Transactional Scope

These interfaces follows the request only pattern. The scope of its processing is to receive Log/Exception request data in CLE2007 format from applications through XML/JMS. The message processing used in CLE12Mediation does guarantee if the message is not lost in the event of a failure.

## Performance

On a busy day, the interfaces under this module can receive around 10000 Log/Error message per hour from applications. To support this, interfaces must be deployed in multiple instances where load will be balanced across instances as this has been built using EMS Queue for both Log and Exception.

## Availability

These interfaces under module “CLE12Mediation” must be available 24 hours per day, 7 days per week, in a highly available environment. There will be no scheduled outages for this service.

## Authentication

N/A

## Regulatory Requirements

#### Data Protection

N/A

#### Data Retention

N/A

## Technical Components

### BW Components

### JMSChannel.process (*Log*)

Applications using CLE v2007 publish log message on JMS topic “*<CompanyName>.ENTERPRISE.SR.CLE.Log.PublishXml.V1\_0*”. This log topic is bridged to CLE v2012 mediation queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Log.Mediation.PublishXml.V1\_0*”. Then this log process receives Log request message (in CLE2007 format) over XML/JMS from mediation queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Log.Mediation.PublishXml.V1\_0*” and then transforms to CLE12 Log format. It then publish back the transformed log message on CLE12 Log queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Log.PublishXml.V1\_0*”.

### JMSChannel.process (*Exception*)

Applications using CLE v2007 publish error message on JMS topic “*<CompanyName>.ENTERPRISE.SR.CLE.Exception.PublishXml.V1\_0*”. This exception topic is bridged to CLE v2012 mediation queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.PublishXml.V1\_0*”. This process receives Exception request message (in CLE2007 format) over XML/JMS from queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.Mediation.PublishXml.V1\_0*” and then transforms to CLE12 Exception format. It then publish back the transformed log message on CLE12 Exception queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.PublishXml.V1\_0*”.

### Data Structures

### Input



### Output



### Translation / Transformation

* N/A

## Exceptions

### System Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
|  |  |  |
|  |  |  |

### User-defined Exceptions

|  |  |  |
| --- | --- | --- |
| Potential Errors | Possible Reasons | Solution Strategy |
| CLE-QSEND-ERROR (Error trying to send ExceptionRequest to CLE JMS Exception Queue) | It could because of following JMS Error:-  InvalidInput  JMSMessageCreateException  JMSSessionCreateException  JMSSendException | Check application log stack to get more details about JMS errors and make sure that JMS serve is up and running and has enough connections to connect from applications |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Testing Requirements

### Unit Testing



# ~~CLE12Administration01~~

## ~~Overview~~

~~The module has been below mentioned services which is used by CLE12 GUI application to perform following various administrative tasks through GUI.~~

~~Services:-~~

* ~~Application Services~~
  + ~~Application Configuration service~~
  + ~~Category service~~
  + ~~Exception Configuration service~~
  + ~~Interface Configuration Service~~
  + ~~Procedure-service~~
  + ~~Render Configuration Service~~
  + ~~Exception Severity Service~~
  + ~~Exception Type Service~~
* ~~GUI Service~~
  + ~~Chart information retrieval service~~
* ~~Exception Service~~
  + ~~Exception Management Service~~
* ~~Log Services~~
  + ~~Log retrieval service~~
* ~~Security Service~~
  + ~~ACL Service~~

## ~~CLE12Administration Data Process Flow~~

~~~~

***~~Fig: CLE12Administration Data Flow~~***

## ~~Design Pattern and Transactional Scope~~

~~The interfaces under this module follows both request-response and request only pattern. The scope of its processing is to receive request from CLE12 GUI application as SOAP/HTTP message. The message processing through interfaces in CLE12Administration does not guarantee if the message is not lost in the event of a failure.~~

## ~~Performance~~

~~On a usual business day, the interfaces under this module can receive around 100 messages per hour from CLE12 GUI application. Henceforth interfaces are not required to be deployed in multiple instances.~~

## ~~Availability~~

~~The interfaces under this module must be available 24 hours per day, 7 days per week, in a highly available environment. There will be no scheduled outages for these interfaces.~~

## ~~Authentication~~

~~N/A~~

## ~~Regulatory Requirements~~

#### ~~Data Protection~~

~~N/A~~

#### ~~Data Retention~~

~~N/A~~

## ~~Technical Components~~

### ~~BW Components~~

### ~~IntfUpdate\_ApplicationConfiguration-service~~

~~This service implements following operations where application configuration request message come from CLE12 GUI application –~~

* ~~Create Application configuration~~
* ~~Delete Application configuration~~
* ~~Update Application configuration~~
* ~~Retrieve Application configuration List~~

### ~~IntfCategory-service~~

~~This service implements following operations where exception category data comes from CLE12 GUI application -~~

* ~~Create Exception Category~~
* ~~Delete Exception Category~~
* ~~Update Exception Category~~
* ~~Retrieve Exception Category List with System ID~~
* ~~Retrieve Exception Category Details~~

### ~~IntfExceptionConfiguration-service~~

~~This service implements following operations where exception configuration data comes from CLE12 GUI application -~~

* ~~Create Exception Configuration~~
* ~~Delete Exception Configuration~~
* ~~Update Exception Configuration~~
* ~~Retrieve Exception Configuration List~~
* ~~Retrieve Exception Configuration Details~~

### ~~InterfaceConfigurationService~~

~~This service implements following operations where interface configuration data comes from CLE12 GUI application -~~

* ~~Create Interface Configuration~~
* ~~Delete Interface Configuration~~
* ~~Update Interface Configuration~~
* ~~Retrieve Interface Configuration List~~
* ~~Retrieve Interface Configuration Details~~
* ~~Retrieve Parent Interface List~~

### ~~IntfProcedure-service~~

~~This service implements following operations where exception procedure data comes from CLE12 GUI application -~~

* ~~Create procedure~~
* ~~Delete procedure~~
* ~~Update procedure~~
* ~~Retrieve Specific procedure~~
* ~~Retrieve procedure list with System ID~~

### ~~IntfRenderConfig-service~~

~~This service implements following operations where transaction render configuration data comes from CLE12 GUI application -~~

* ~~Create RenderConfig~~
* ~~Delete RenderConfig~~
* ~~Update RenderConfig~~
* ~~Retrieve Application Specific RenderConfig~~

### ~~IntfSeverity-service~~

~~This service implements following operations where exception severity data comes from CLE12 GUI application -~~

* ~~Create Exception Severity~~
* ~~Delete Exception Severity~~
* ~~Update Exception Severity~~
* ~~Retrieve Specific Exception Severity~~
* ~~Retrieve Exception Severity list with System ID~~

### ~~IntfType-service~~

~~This service implements following operations where exception type data comes from CLE12 GUI application -~~

* ~~Create Exception Type~~
* ~~Delete Exception Type~~
* ~~Update Exception Type~~
* ~~Retrieve Specific Exception Type~~
* ~~Retrieve Exception Type list with System ID~~

### ~~IntfExceptions-service~~

~~This service implements following operations where exception data comes from CLE12 GUI application -~~

* ~~Delete Exceptions~~
* ~~Resolve Exceptions~~
* ~~Retrieve Exception Detail~~
* ~~Retrieve Exception Detail with rendering~~
* ~~Retrieve Exception list~~

### ~~Retrieve\_Log\_service~~

~~This service implements following operations where log data request comes from CLE12 GUI application -~~

* ~~Delete Logs~~
* ~~Retrieve Log Details~~
* ~~Retrieve Log List~~

### ~~IntfACL-service~~

~~This service implements following operations where ACL data comes from CLE12 GUI application -~~

* ~~AvailableRolesRetrieval~~
* ~~DeleteRoleFromACLTable~~
* ~~RetrieveACL~~
* ~~SetRoleACL~~
* ~~UserAuthentication~~
* ~~UserRolesApps~~
* ~~UserRolesRetrieval~~

### ~~IntfChartInfoRetrieval-service~~

~~This service implements following operation which are invoked from CLE12 GUI application to view data in dashboard chart -~~

* ~~ChartInfoRetrieval~~

### ~~Data Structures~~

### ~~Input~~

~~~~

### ~~Output~~

### ~~Translation / Transformation~~

* ~~N/A~~

## ~~Exceptions~~

### ~~System Exceptions~~

|  |  |  |
| --- | --- | --- |
| ~~Potential Errors~~ | ~~Possible Reasons~~ | ~~Solution Strategy~~ |
|  |  |  |
|  |  |  |

### ~~User-defined Exceptions~~

|  |  |  |
| --- | --- | --- |
| ~~Potential Errors~~ | ~~Possible Reasons~~ | ~~Solution Strategy~~ |
| ~~Exception Configuration not found.~~ | ~~Exception configuration data received from CLE12GUI doesn’t exist in DB~~ | ~~Verify input data given from GUI~~ |
| ~~Exception Record Not Found~~ | ~~Exception record data received from CLE12GUI doesn’t exist in DB~~ | ~~Verify input data given from GUI~~ |
| ~~Log records not found for the search criteria~~ | ~~Log record data received from CLE12GUI doesn’t exist in DB~~ | ~~Verify input data given from GUI~~ |
| ~~Could not find exception replay information~~ | ~~Exception replay data received from CLE12GUI doesn’t exist in DB~~ | ~~Verify input data given from GUI~~ |
| ~~Application ID not registered~~ | ~~Application ID received from CLE12GUI doesn’t exist in DB~~ | ~~Verify application ID entered from CLE12GUI~~ |
| ~~Application Record Not Found~~ | ~~Application ID received from CLE12GUI doesn’t exist in DB~~ | ~~Verify application ID entered from CLE12GUI~~ |
| ~~Interface ID not defined~~ | ~~Interface ID received from CLE12GUI doesn’t exist in DB~~ | ~~Verify Interface ID entered from CLE12GUI~~ |
| ~~Invalid Destination Exception~~ | ~~JMS destination doesn’t exist~~ | ~~Verify JMS destination name in EMS server~~ |
| ~~Cannot connect to EMS Server~~ | ~~EMS server may be down or connections exhausted~~ | ~~Verify connectivity with EMS server is fine.~~ |
| ~~Data Validation Error~~ | ~~Data received from CLE12GUI as inputs doesn’t conform to message structure~~ | ~~Verify data taken as inputs from CLE12GUI.~~ |

## ~~Testing Requirements~~

### ~~Unit Testing~~

~~<TBD>~~

# Audit and Logging

N/A

# Exception Handling

* For any interfaces under “CLE12LogManagement” and “CLE12WorkFlow” modules, if any error occurs related to XML validation error/application ID not found/render ID(or interface ID) not found, they send the error messages directly to exception queue “*<CompanyName>.ENTERPRISE.SR.CLE12.Exception.PublishXml.V1\_0*” which will be processed by “JMS Channel” process of “CLE12ExceptionManagement” module otherwise error will be logged in application logs.
* For interfaces under “CLE12Mediation”, “CLE12ExceptionManagement” and “CLE12Administration” modules, if any error occurs, it’ll be logged to application logs except for errors related to “applicationdID not found” or “interfaceID not found” or “renderID not found” or “exceptionconfig” table entries don’t exist.

# Appendix A – Change Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Version Number*** | ***Changes Made*** | | | |
| V0.1 | Initial baseline created on 16-Jun-2016 | | | |
|  |  | | | |
| **Section No.** | **Changed By** | **Effective Date** | **Changes Effected** |
| <Draft> | All | Soumitra Bishnu | 1/20/2017 |  |
| 1.0 | 4,5,6,8 | Soumitra Bishnu | 1/25/2017 | **1. For Log and Exception management modules, currently only JMS channel will be used in BW6.x migration.**  **2. For section # 6.9.2(i.e. in CLEWorkflow module), only CLE specific exception codes & messages have to be used.**  **3. CLE12Administraion module will not be migrated to BW6.x as of now.** |
|  |  |  |  |  |