cortex®
Intelligent Orchestration

CTX-Logging User Guide

Contents

Versions

Document Revisions

The following revisions have been made to this document

Date	Revision	Notes
04/12/2018	1.0	First release

Module Versions

The following revisions have been made to this document

Date	Revision	Notes
04/12/2018	1.0	Creation of: • Logging-CL-Cortex-Log

Preface

About this Manual

This document is a user guide for the CTX-Logging module.

Audience

The audience for this document is those wanting to understand how to use CTX-Logging module.

Related Material

None

Abbreviations used in this Document

SQL	Structured Query Language

Requirements

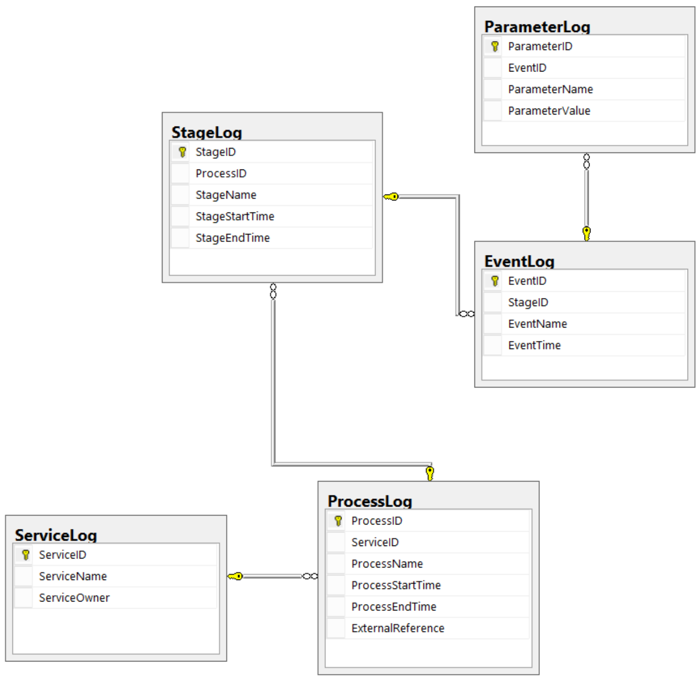
The CTX-Logging subtasks require the following:

- Minimum Cortex v6.4 installed on the Cortex Application Server
- SQL Cortex-Logging database installed

1 Overview

The Cortex logging module allows flow authors to easily log process information to the database in a structured manner to allow for complex reporting.

1.1 Data Model



1.2 Components

1.2.1 Service

A service is the top-level component and should be used as configuration to help with organisation. This table must be modified manually.

1.2.2 Process

A process can span across multiple flows, it has a start and end time and can have an external reference to link it to other systems.

1.2.3 Stage

A process can have multiple stages associated with it. Similarly to a process, a stage also has a start and end time, however stages shouldn’t span across multiple flows. There can only be one stage open at a time on a process.

1.2.4 Event

A stage can contain multiple events. Events occur at a single time, they don’t span over time periods like processes and stages.

1.2.5 Parameter

Events can have parameters associated with them such as error message on any other relevant information. Each parameter will have a name and a value.

1.3 Subtask logic summary

- If the Log-Handler structure is not passed in a new one will be created automatically
- If a Process is ended its child stage will be ended automatically
- If a stage is started without creating a process, the process will be created automatically with the same name as the stage
- If a stage is started for a process that already has an open stage, the currently open stage will be ended automatically.
- A stage can be linked to a process using either it's ID or external reference ID
- If an event is created without a stage or process, these items will be created automatically using the same name as the event
- After logs are committed the logs the log-handler structure will be cleared to prevent double commits.

2 Logging-CL-Cortex-Log

2.1 Inputs

Variable Name	Description
cl_i_Event-Name-To-Create	Name of the event that will be created
cl_i_Stage-Name-To-Create	Name of the stage that will be created
cl_i_Process-Name-To-Create	Name of the Process that will be created
cl_i_End-Process	Takes values ‘yes’ or ‘no’. Yes will end the process. ‘No’ will not end the process, this is the default behaviour if a value is not provided
cl_i_End-Stage	Takes values ‘yes’ or ‘no’. Yes will end the stage. No will not end the stage, this is the default behaviour if a value is not provided
cl_i_Log-Handler	Contains logging information, this variable should not be manually modified and should be passed in and out of all subtasks through the process
cl_i_Commit-Logs	Takes values ‘yes’ or ‘no’. ‘Yes’ will commit all the logs recorded by the ‘Log-Handler’. ‘No’ will continue to append the ‘Log-Handler’ with more logs, this is the default behaviour if a value is not provided.
cl_i_Connection-String	If ‘i_Commit-Logs’ is set to ‘yes’, a connection string for the database needs to be provided. Example: Server=localhost;Database=CortexLogging;Trusted_Connection=True;
cl_i_Parameters	Structure of name/value pairs of parameters to be added to an event. Note: the creation of an event is mandatory
cl_i_External-Reference	A reference to the process that offers another option to link a stage to a process
cl_i_Service-ID	Optional parameters that can be provided on the create of a process to create a link to the service
cl_i_Process-ID	Optional parameter that can be provided on the creation of a stage to create a link to the process

2.2 Outputs

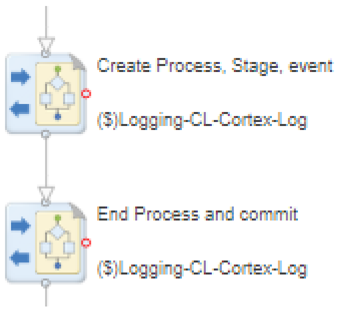
Variable Name	Description
cl_o_Flow-Reference	UUID of the flow, may be useful for process linking

cl_o_Log-Handler	Contains logging information, this variable should be passed out of every subtask

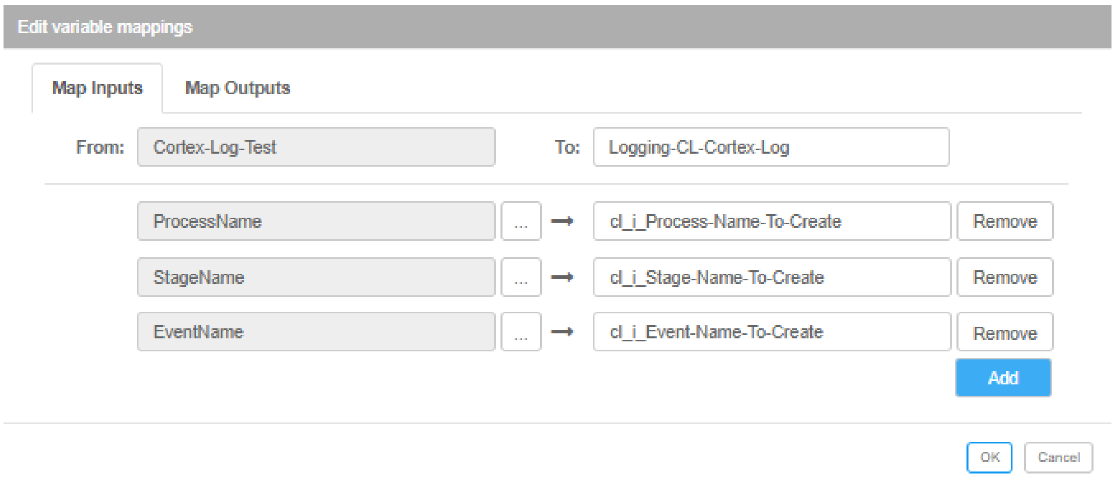
2.3 Examples

2.3.1 Basic end to end example

Create Process, Stage and Event then End Process and commit



Subtask 1 inputs

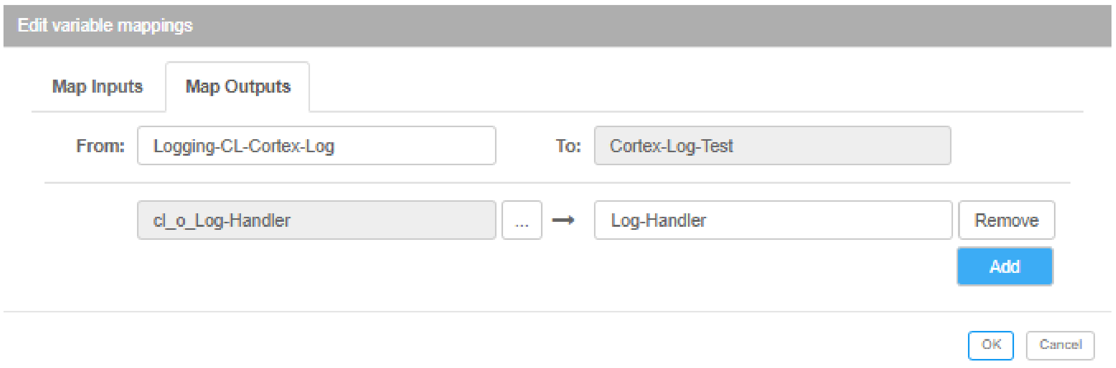


From: Cortex-Log-Test   To: Logging-CL-Cortex-Log

ProcessName	→	cl_i_Process-Name-To-Create
StageName	→	cl_i_Stage-Name-To-Create
EventName	→	cl_i_Event-Name-To-Create

'i_Log-Handler' is not required in the first instance of the subtask

Subtask 1 outputs

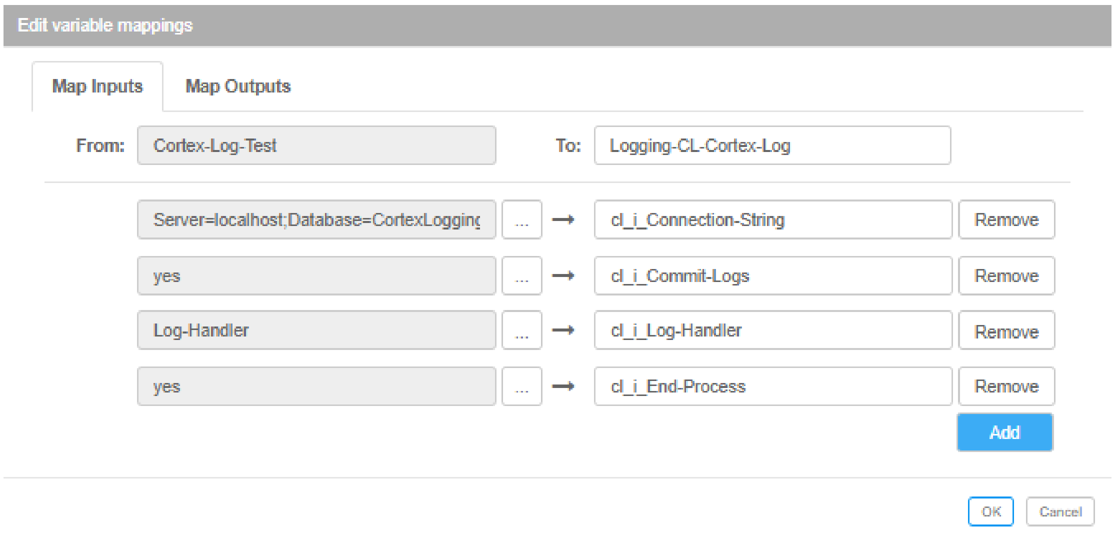


From: Logging-CL-Cortex-Log To: Cortex-Log-Test

cl_o_Log-Handler → Log-Handler

The log-handler needs to be passed out of the subtask as it contains uncommitted logging information

Subtask 2 Inputs



From: Cortex-Log-Test To: Logging-CL-Cortex-Log

Server=localhost;Database=CortexLogginç → cl_i_Connection-String

yes → cl_i_Commit-Logs

Log-Handler → cl_i_Log-Handler

yes → cl_i_End-Process

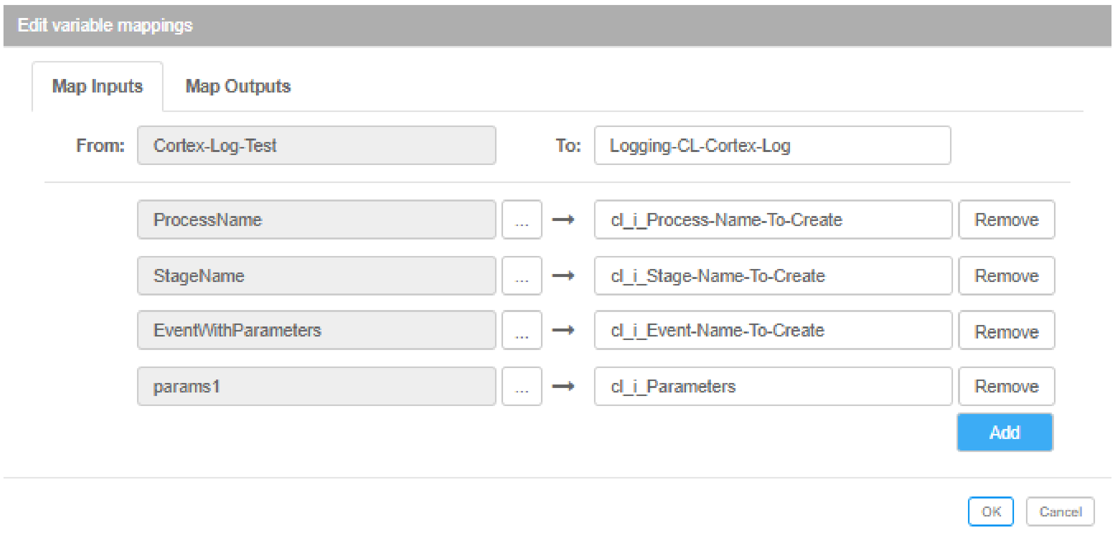
In this case the stage is not being ended explicitly, therefore it will be closed automatically when the process is ended

Subtask 2 Outputs

Because the logs have been committed no outputs are required

2.3.2 Create Process, stage and event with parameters

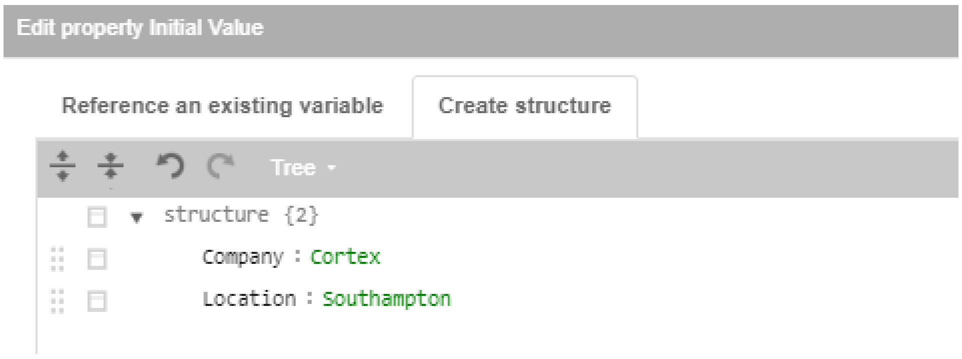
Subtask Inputs



From:	Cortex-Log-Test	To:	Logging-CL-Cortex-Log
ProcessName	→	cl_i_Process-Name-To-Create	Remove
StageName	→	cl_i_Stage-Name-To-Create	Remove
EventWithParameters	→	cl_i_Event-Name-To-Create	Remove
params1	→	cl_i_Parameters	Remove

In this case no log handler was passed in as it was the first instance of the subtask

Params1 variable example:



```
structure {2}
    Company : Cortex
    Location : Southampton
```

The amount of name value pairs isn't limited

2.4 Database Queries

2.4.1 Pivot tables

To access the parameters when reporting on the logging data pivot tables may be required, below is an example:

```
SELECT *
INTO #Temp
FROM
(
    SELECT Pa.ParameterName, Pa.ParameterValue, P.ProcessID
    FROM ProcessLog P
    INNER JOIN StageLog S
    ON P.ProcessID = S.ProcessID
    INNER JOIN EventLog E
    ON E.StageID = S.StageID
    LEFT JOIN ParameterLog Pa
    ON Pa.EventID = E.EventID
) SRC
PIVOT
(
    MAX(ParameterValue)
    FOR ParameterName IN ([Customer], [CRM_System]) --Input parameters here
) PIV

SELECT ProcessName, ProcessStartTime, ProcessEndTime, ExternalReference, T.*
FROM #Temp T
INNER JOIN ProcessLog P
ON P.ProcessID = T.ProcessID
```