## CC7940

|  |  |
| --- | --- |
| Short Description: | MPMS Watchlist reporter integration |
| Requested Date: | 10/02/2023 |
| Requested By: | Des Chapple |
| Client Contact: | Stephen Tomlinson |
| Client Name: | MPMS |
| Requirement Approved By: | Tracey Rose |
| Budget | |  |  |  | | --- | --- | --- | | Client Billable | Capex | Expense | |
| Estimate Type | |  |  | | --- | --- | | Ballpark | Detailed Estimate | |
| Planview ID | 0811409 |
| Cross Charge BUC | Provide Business Unit Code (BUC) to cross charge |
| Approval Probability | |  |  |  | | --- | --- | --- | | High | Medium | Low | |
| Client requested Implementation Date/Release: |  |

## Overview of Change Required

**Business Requirement Overview:**

MPMS Requirement

MPMS require a PEPs and Sanctions check to be performed on every new customer added to P1C and if a customer changes their name, address, or date of birth. This list could be increased so the solution should allow for more fields to trigger a scan request. It also runs checks daily for all customers on file.

Graphical user interface, diagram, application

Description automatically generated

**REAL-TIME SCANS**

When a customer is added to P1C, a PEPs check and sanctions check must be performed to prevent sanctioned or politically exposed persons from being granted account access. When a new customer is added online, P1C will need to format and send two new APIs to WLR to request the checks to be performed – a PEPs scan request, and a sanctions check request. Both requests differ by geography of the issuing bank hence the need to provide the country in the request for each call.

The same scans should be requested in the event specific customer demographic information changes (currently: name, address, date of birth).

*SubmitScan* (to be used for both PEPs and Sanctions scans):

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Required** | **Description** |
| **Version** | Yes | WLR Web Service API version. Set to 1 |
| **SystemID** | Yes | System Identity of the API. TBD |
| **ApplicationName** | Yes | Name of the application. TBD |
| **RequestId** | Yes | Unique identifier of the request |
| **ScanJobName** | No | Not required |
| **ScanJobRouting** | Yes | Should contain two parameters to indicate whether this is a request for a Sanctions or PEPs scan and the name of the “country of the Issuing Bank”  e.g.  param 1: Sanctions  param 2: Japan  or  param 1: PEPs  param 2: Australia |
| **ScanType** | Yes | Type of Scan as defined in the Scan Parameter.  Assume should always be “Full” |
| **ConfidenceThreshold** | No | Threshold limit of the Scan as defined in the Scan Parameter. |
| **ResultConfiguration** | Yes | Scan Result configuration and output settings. Usage TBD |
| **WatchList** | No | WatchList for Scan |
| **RequestProcessingType** | No | Process type of the System as passed to the originating system |
| **ScanRequest** | Yes | Payload request for the Scan. |

Payload for the scan (Customer information required for each call – one customer per request):

|  |  |  |
| --- | --- | --- |
| **Field** | **Required** | **Description** |
| **CustomerId** | Yes | Unique customer identifier |
| **CustomerName** | Yes | Customer name: First + Middle+Last |
| **FirstName** | No |  |
| **MiddleName** | No |  |
| **LastName** | No |  |
| **StreetAddress** | Yes | Customer’s street address |
| **City** | Yes | Customer’s city |
| **State** | Yes | Customer’s state |
| **Country code** | Yes | Customer’s country. 2-character ISO code |
| **Branch** | No | Branch associated with customer |
| **Department** | No | Department associated with customer |
| **PEPIndicator** | No | PEP (“politically exposed person”) Indicator |
| **TIN** | No | Tax Identification or Social Security Number |
| **PassportNumber** | No | Passport number |
| **PassportCountry** | No |  |
| **DateofBirth** | Yes | Date of birth: CCYYMMDD |
| **DriverLicenseNumber** | No | Driver license number |
| **DriverLicenseIssuingState** | No | Driver license issuing state |
| **CustomerType** | No | Customer type – Individual or Business |
| **TypeOfBusiness** | No |  |

See attached documents for full format / XML / further detail:



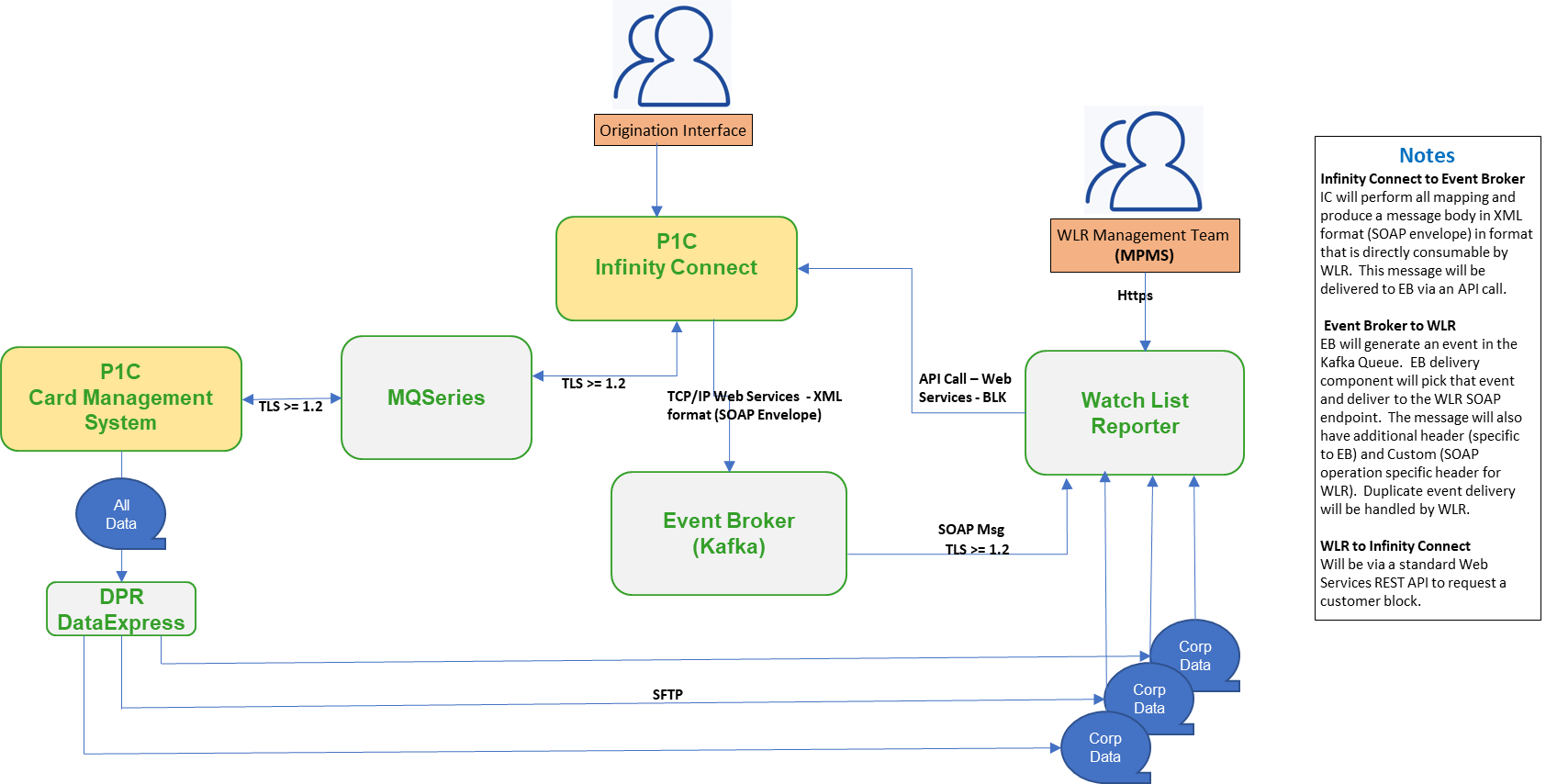
The PEPS check will be a ‘fire and forget’ message from P1C to WLR, via Event Broker. Any hits will generate a case in WLR to be worked by MPMS, but the message will have no feedback to P1C.

A Sanctions call will be an asynchronous call from P1C to WLR, via Event Broker, to perform a ‘scan’ (see FCM WLR Web API, above). If there is a ‘hit’ on a sanctions list, WLR will send an API showing the hit information, directly to P1C and bypassing Event Broker. P1C will process this API by blocking the **customer** (and all accounts associated to that customer) in P1C which will prevent the use of existing accounts and prevent any new accounts from being added in the future. It has been suggested to the WLR team that the BLK API will be modified to be keyed by customer ID and that upon receipt in P1C of this request, the customer, and all accounts associated to that customer will be blocked.

**BATCH SCANS**

Existing customers can be added to Sanctions lists at any time. It is required by MPMS that the customer data base be scanned daily for potential changes. WLR does not currently retain customer data for re-scanning. This feature is on a roadmap but is not prioritised in the next couple of years. Due to this issue, P1C will need to pass all customer details to WLR in full volume daily batch files.

Batch files will be sent from P1C to WLR via sFTP. There will be one file per ‘sanction region’ as this method helps WLR organize a more efficient way of scanning. The file segmentation also determines which regional lists need to be scanned as not all countries recognize the same sanctioning rules. WLR will ingest the batch files and will send an API (assumption is a customer ID modified version of BLK) to P1C if there is a ‘hit’. P1C will receive the API in the same manner as the real time method mentioned above.



**Sanctions Rescan**

All new/modified customer details need to be rescanned against the sanctions lists daily. All new/modified sanctions records will be scanned against all corresponding customer records.

Multiple files will be generated daily segmented by “Country of Issuing Bank”. This means a fixed set of lists will be scanned for each file generated.

Should there be a hit whilst processing WLR will API P1C immediately.

**PEPs Rescan**

All customers details need to be rescanned against the PEPs lists at least once in any 14-day period.

Multiple files will be generated segmented by “Country of Issuing Bank”. This means a fixed sit of lists can be scanned for each file generated. PEPs batch files should be generated once every 14 days.

The only outcome from PEPs rescans will be alerts that will be worked by the PEPs team.

The Customer Batch Data Specifications are standard file formats for completing batch scans within WLR. This specification is often referred to as the *Koder* File. The table below lists required fields and the order in which they must appear in each record.

Note: Each field must be delimited by the tilde (~) character.

A header record is also required in the file.

A full customer file must be sent each business day.

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Type** | **Mandatory Field (Yes/No)** |
| **Unique Customer ID** | varchar | Y |
| **Name (First + Middle + Last in this order or Company Name)** | varchar | Y |
| **FirstName** | varchar | N |
| **MiddleName** | varchar | N |
| **LastName** | varchar | N |
| **Street Address** | varchar | Y |
| **City** | varchar | Y |
| **State** | varchar | Y |
| **Country Code (2 character ISO code)** | varchar | Y |
| **Branch** | varchar | N |
| **Department** | varchar | N |
| **TIN** | varchar | N |
| **Passport Number** | varchar | N |
| **Passport Country** | varchar | N |
| **Date of Birth (CCYYMMDD)** | date | N |
| **Driver License Number** | varchar | N |
| **Driver License Issuing State** | varchar | N |
| **Customer Type (Individual or Business)** | varchar | N |
| **TypeOfBusiness (populated with occupation for personal customer)** | varchar | N |

FIS Delivery Team Comments

Any WatchList Reporter matches will initiate a ‘block **customer’** API to P1C. This block should block the customer (so that any accounts added for the is customer (e.g. via the NAC API) are rejected) and all accounts associated to that customer. It has been suggested to WLR that a modified version of the BLK API might be used for this – keyed by customer ID. WLR will populate this including the block / reclass value and send the API to P1C.

See attached WLR documentation:



Out of Scope

N/A

## High Level Design/Proposed Solution

**Solution Overview:**

P1C Solution:

Two new BC81 table entries will be defined for the following files:

* a daily extract file in Watch List Report (WLR) format for Sanctions screening
* a weekly extract file in WLR format for PEPs screening

The Non-Monetary Post process will be enhanced to create an extract file in WLR format if the BC81 parameter is active for the file.

The extract file will exclude the following accounts: stolen, transferred, charged-off, inventory cards (bulk ordered prepaid cards that haven’t been personalised, where the name 1 matches the default name from BC08), and accounts with a closed block as defined to BC72.

A new outgoing transmission process will be developed to transmit the file to Data Placement Manager   
(DPR). This process will support splitting by corporation, sorting by country, and generating the header and trailer records. The new outgoing transmission stream will be added to both the daily and weekly batch schedules. P1C will not be splitting the file by country. This work will be done by the Orion team on the PROCSRV.

To allow a ‘hit’ customer and all accounts associated with that customer to be rejected via the BLK API, the CDU service link will be enhanced to support the generation and execution of the BLK service link. The block and reclass fields will be added to the CDU service link receive data block.

P1C Infinity Connect Solution (For Flow #1 to #4)

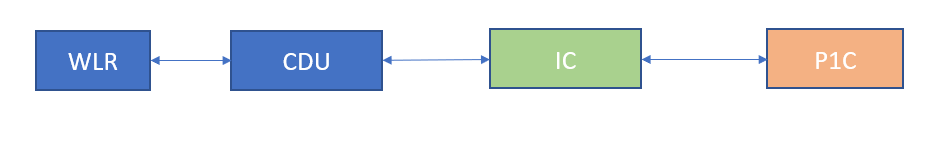
MPMS will be sending the CCA (new customer add) or CCU (customer update) Service Link JSON API request to P1C Infinity Connect (#1). Infinity Connect will respond back to MPMS once P1C responded (#2). If P1C returns successful response, then Infinity Connect will use AdminUI/ Data Mapper (#3) to call PEP Check and Sanction Check APIs via Event Broker (#4). These API calls simply send and forget to Event Broker. The complete solution and flow will be orchestrated through Infinity Connect AdminUI.

A picture containing text, screenshot, diagram, design

Description automatically generated

P1C Infinity Connect Solution (For flow #8 and #9)

WLR will send CDU JSON API request to apply the block to all accounts linked to the customer. A new version of the Infinity Connect CDU JSON API will be generated to include the new block and reclass fields in the receiver data block.



Proposed Changes:

|  |  |  |
| --- | --- | --- |
| Item # | Development Task | Estimate (days) |
| P1C Changes | | |
|  | The Non-Monetary Post process (PC510) will be enhanced to create an extract file in WLR format if the BC81 parameter is active for the file.  The extract file will exclude the following accounts: stolen, transferred, charged off, inventory cards (bulk ordered prepaid cards that haven’t been personalised, where the name 1 matches the default name from BC08) and accounts with a closed block as defined to BC72. | 12 days |
|  | A new outgoing transmission stream will be developed to transmit the extract file to DPR. This process will support splitting by corporation, sorting by country, and generating the header and trailer records.  This transmission stream will be added to both the daily and weekly batch schedules. | 6 days |
|  | Add the Block/Reclass fields to the CDU service link receive data block. Enhance the CDU service link program to support the BLK service link. | 10 days |
| Infinity Connect Changes | | |
| 4 | Orchestration Service via Admin UI (task for Flow #1 to #4).   * Analysis, design, research, and support. * Deploy the latest release in an existing IC development environment different than .140 GLBC. * Configure the new environment in Admin UI and IDP. * New GIT feature branch and IC workspace setup. * Create a new xpressng-json-services IC project. * The CCA/CCU JSON APIs service configuration will not be modified. These services will be called internally by a new orchestration API with its own REST endpoint. The version, corporation number, request code as well as header request fields passed in the orchestration API URL will be used to internally call either CCA or CCU JSON APIs. * Define a local IC HTTP inbound endpoint for internal API calls. * Define the Event Broker HTTP outbound endpoint for test and production environments. * Develop a component with logic to read from the exchange the CCA JSON response fields ‘SVL\_CCA\_S\_DATA\_BLOCK.SVL\_CCA\_S\_CUST\_PREFIX and SVL\_CCA\_S\_DATA\_BLOCK.SVL\_CCA\_S\_CUST\_ID’ and move the values to the CCA JSON request fields ‘SVL\_CCA\_R\_DATA\_BLOCK.SVL\_CCA\_R\_CUST\_PREFIX and SVL\_CCA\_R\_DATA\_BLOCK.SVL\_CCA\_R\_CUST\_ID’ respectively. Also make the updated request available in the exchange to be used by the transformation files. Component will only be executed for successful CCA/CCU responses and before calling the transformation files that generate the Event Broker request messages. * Use the API Builder and Mediations screens to create a new orchestration REST API and define the following mediation flow:   + Accepts CCA/CCU JSON APIs request and response messages.   + No request/response validation and transformation. As this will be done by each service.   + Design the service orchestration logic using the Route builder and Orch Designer screens.   + Internally call CCA/CCU existing JSON APIs via REST route or by the new local IC HTTP endpoint.   + If the CCA/CCU JSON response is successful (HTTP status code 200) then     - Execute the new component logic to move CCA customer id and prefix values into the respective CCA/CCU JSON request fields.     - Prepare the PEPs scan SOAP request using the CCA/CCU JSON request fields and transformation files.     - Call the PEPs scan SOAP webservice using the new Event Broker outbound endpoint configuration. IC will not wait for a response.     - Prepare the Sanctions scan SOAP request using the CCA/CCU JSON request fields and transformation files.     - Call the Sanctions scan SOAP webservice using the new Event Broker outbound endpoint configuration. IC will not wait for a response.     - Any exceptions thrown while trying to communicate with the Event Broker endpoint must be handled and the failed SOAP requests sent must be saved in a separate log in the filesystem. * Return the CCA/CCU JSON response. * Create the orchestration service Swagger file using the latest version of the CCA/CCU schemas. * Create unit tests for the generated orchestration service API. Integration tests in the development environment. Test successful and unsuccessful scenarios. * Verify that failed requests sent to Event Broker are being logged. | 20 days |
| 5 | PEPs Scan Service XSLT Generation via Data Mapper   * Create a new Data Mapper project to generate transformation file required for the PEPs scan SOAP service. * Map the Infinity Connect CCA/CCU API request JSON schema with the XML ‘SubmitScan’ request schema. * All required fields listed in page 3 and 4 must be mapped. Define constants for Version, SystemID, ApplicationName, RequestId and ScanJobRouting fields. * SOAP envelope and required namespaces must be included in the message transformation file. * Create data mapping file documentation. * Save, generate, and test transformations for all request mapped fields. * Validate transformed PEPs scan SOAP request message with WSDL. | 4 days |
| 6 | Sanctions Scan Service XSLT Generation via Data Mapper   * Create a new Data Mapper project to generate transformation file required for the Sanctions scan SOAP service. * Map the Infinity Connect CCA/CCU API request JSON schema with the XML ‘SubmitScan’ request schema. * All required fields listed in page 3 and 4 must be mapped. Define constants for Version, SystemID, ApplicationName, RequestId and ScanJobRouting fields. * SOAP envelope and required namespaces must be included in the message transformation file. * Create data mapping file documentation. * Save, generate, and test transformations for all request mapped fields. * Validate transformed Sanctions scan SOAP request message with WSDL. | 4 days |
| 7 | Workspace Scripts Update   * Export the new API service definition, xslt files, orchestration and endpoints configuration built in Admin UI development environment. * Import the configuration in the Infinity Connect workspace, review, define and externalise variables, update Oracle and PostgreSQL incremental script files to use variables for all the test and production environments. * Configure Even Broker HTTP endpoint keystore and truststore in the database scripts. * Test database scripts and verify that all the imported data is correctly packaged and configured per environment. | 4 days |
| 8 | Logging Enhancement:   * PEP and Sanction API requests should be stored in a logging file when IC are unable to connect with Event Broker. * Only for MPMS, create a new daily rolling file appender ‘eventBrokerAppender’ to be used to log unsuccessful SOAP requests to Event Broker. Enabled by default. * Externalise log appender configuration variables in all MPMS user properties. * Overwrite the ‘xpressNgLog4j.xml’ file via client configuration files. * Unit and integration testing with the new orchestrated API. * Verify that the updated log4j configuration file is included in the client configuration files. | 5 days |
| 9 | CDU Service Link JSON API changes for new Block/Reclass fields.   * New CDU API version will be generated. * Documentation Update (Wiki) for versioning. * Create a new IC WLR credential per client environment and one common ‘WLR’ channel. Include them in the Oracle and PostgreSQL scripts and encrypt the password. * Add credentials to xesDocs. * Unit and Integration Testing. * Verify that CDU JSON and V2XML APIs can be called with all the new credentials. | 4 days |
| 10 | Event Broker Connectivity (TLS1.2):   * New TLS certificates will be added in P1C Infinity Connect servers in Test and Prod to connect Event Broker * Connectivity testing in Test and Prod. * The Encrypted Passwords for certificates will be kept under CyberArk. * Documentation (Wiki) and share the knowledge with Support Team. | 3 days |
| 11 | Integration Testing (End to End) | 3 days |
| 12 | QA support for Testing | 3 days |
| 13 | Release related tasks   * Merge, code reviews, code roll. * Release builds. * Veracode and Black Duck scans. * Update IC wiki with release deployment steps and meetings. | 3 days |

## Assumptions

Infinity Connect:

MPMS will send CCA/CCU request in Infinity Connect JSON API format to the new orchestration API endpoint. Changes are not expected in both APIs so the existing versions will be used.

IC will not retry to send the API requests when the Event Broker is down, or not reachable, but IC can keep those API payloads in log file by masking the PAN details.

Unsuccessful PEPs or Sanction scan responses from Event Broker will not be processed by Infinity Connect since these are not expected to be received. Tasks for processing and validating Even Broker responses are not included in the estimate.

If P1C responds failure response for CCA/CCU or P1C under maintenance, then there will not be an Event Broker call for the given request, and these transactions will not be kept under log file.

Expected almost no code change for flow #1 to #4, all should be handled through AdminUI and Data Mapper, and these SQL and configuration configurations will be manually included in the respective MPMS IC workspace. The only component to be developed as part of the orchestration is the one mentioned in item #1.

North America CIO team are expected to support network connectivity, manage, and share TLS certificates with the client to connect Event Broker.

Event Broker team are expected to share the PEP and Sanction API WSDL and Test/Prod endpoint Information.

IC assumes that PEP and Sanction API will use the same endpoint URL and WSDL operation/method ‘*SubmitScan’* as mentioned in page 2. The provided WSDL file (page 4) only contains a different method named ‘PerformScan’. To start items 5, 6 and complete item 4 IC requires the correct WSDL file to generate the <SubmitScan> XML tag expected by Event Broker. Refer to page 40 of the ‘WatchList Reporter Web Service System APIs’ documentation for a ‘SubmitScan’ sample request.

MPMS must provide the correct values to set in the PEPs and Sanctions Scan SOAP API request fields: SystemID, ApplicationName, ScanJobName, ScanJobRouting and ResultConfiguration. If values are not provided, then IC will set the default values specified in the API documentation (for required fields) and will not send unmapped request fields defined as not required.

Infinity Connect will send the API calls to Event Broker and will not wait for response (send and forget). Also, IC will not call Event Broker to get the status of any PEPs or Sanctions scans requested.

IC assumes that consuming Event Broker web services does not require WS-Security implementation or adding credentials in the request payload. Only the request fields defined in the WSDL service operation ‘SubmitScan’ will be sent in the SOAP request message.

WLR are expected to send the API request (CDU) in P1C Infinity Connect JSON format.

WLR team are responsible to connect P1C Infinity Connect endpoints, will be created separate P1C Infinity Connect credentials for WLR to avoid security related issues.

The IC RqUID will be sent in the PEPs and Sanctions check SOAP request ‘RequestId’ field.

Item

Admin UI service authoring functions required to complete items 4, 5 and 6 are expected to be available, working after the 3.17 framework upgrade and capable of generating the new orchestration service, service routes, JSON to XML message transformation, and service configuration files. Items 4, 5 and 6 can be delayed if:

* The IC team finds application related issues while using Admin UI, IdP and these require Admin UI team support to resolve them. This also applies to IC application issues found while integrating the services, routing logic and components in the Camel framework used for orchestration purposes.
* The provided WSDL and schemas cannot be parsed and/or loaded in the Data Mapper UI tool. If the IC team cannot load the file as is, then the transformation files will be generated manually without Data Mapper which will take more time to complete.
* Admin UI authoring functions are not capable of generating the required orchestration service and supporting files so these configuration files and code will have to be created manually.
* The IC orchestration components do not support calling the Event Broker SOAP web services out of the box. A new component will be created to consume SOAP web services.

## Disclaimers/Exclusions

P1C will not split the file by country. This work will be done by the Orion team on the PROCSRV.

### Risk Assessment

High

|  |  |
| --- | --- |
| Critical | * High customer visibility (e.g. authorisations, correspondence, internet banking) * Difficult recovery * Complex changes to existing functionality * Date driven mandatory change * Extensive interface changes * 3rd party vendor changes required * Hardware/Infrastructure changes |
| High | * Some customer visibility * Some interface changes * New complex functionality * Back office and/or call centre changes |
| Medium | * Minor customer visibility * Minor interface changes (e.g. new fields in existing filler) * New low impact functionality * New reports or files |
| Low | * No customer visibility * Internal report layout changes |

## Solution Estimates

Details a breakdown of costs associated with this estimate request.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Estimate | Technical Days | Business Days | Total  Days | Cost |
| Design/Analysis | 12 | 8 | 20 |  |
| Code/Unit Test | 69 |  | 69 |  |
| Business Test |  | 16 | 16 |  |
| Manuals/Help Text |  |  |  |  |
| Set up |  |  |  |  |
| Parameter/Mapping |  |  |  |  |
| Project Management/Support |  |  |  |  |
| Implementation/Upgrade |  |  |  |  |
| Other Expenses |  |  |  |  |
| Total |  |  |  |  |
| GST |  |  |  |  |
| Total (GST inclusive) | **81** | **24** | **105** |  |

### Additional Batches

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Estimate | Number | Cost per Batch/Month | Total | Cost |
| Batches |  |  |  |  |
| Machine Cost |  |  |  |  |
| Once off (*described below*) |  |  |  |  |
| Other (*described below*) |  |  |  |  |
| Total |  |  |  |  |
| GST |  |  |  |  |
| Total (GST inclusive) |  |  |  |  |

### Additional Pricing

The following pricing will be effective on the production implementation of this enhancement.

|  |  |  |  |
| --- | --- | --- | --- |
| Invoice Component | Amount | GST | Amount  (GST inclusive) |
| License Cost |  |  |  |
| Annual maintenance |  |  |  |

### Other Estimates/Pricing/Cost

None.

## Estimate Totals

### One off Costs

|  |  |  |  |
| --- | --- | --- | --- |
| Solutions Estimate Component | Amount | GST | Amount  (GST inclusive) |
| Solution Estimate |  |  |  |
| Additional Batches |  |  |  |
| Additional Pricing |  |  |  |
| Other Estimates/Pricing/Costs |  |  |  |
| Totals |  |  |  |

### Ongoing Costs

|  |  |  |  |
| --- | --- | --- | --- |
| Solutions Estimate Component | Amount | GST | Amount  (GST inclusive) |
| Additional Pricing |  |  |  |
| Other Estimates/Pricing/Costs |  |  |  |
| Totals |  |  |  |