International Data Exchange Agent

High Level Design

2016-12-21

Version 0.4

# Document Information and Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Created / Updated | Revision Notes |
| V0.1 | Aug 3, 2016 | Tatiana Stepourska | Initial draft |
| V0.2 | Sep 27, 2016 | Tatiana Stepourska | First round of updates |
| V0.3 | Nov 23 ,2016 | Tatiana Stepourska |  |
| V0.4 | Dec 21, 2016 | Tatiana Stepourska | Modified to accommodate requirement for generating MessageRefId and feeding it back to Infodec along with PSNs after file has been accepted by other jurisdiction |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table Of Contents

[Document Information and Revision History 2](#_Toc470174423)

[Development Objective 3](#_Toc470174424)

[Solution Summary 3](#_Toc470174425)

[Application Architecture 5](#_Toc470174426)

[Business Model 5](#_Toc470174427)

[1.1 Outbound – Prepare Payload 5](#_Toc470174428)

[1.2 Outbound – Prepare Package 5](#_Toc470174429)

[2.1 Inbound – Preliminary 6](#_Toc470174430)

[2.2 Inbound – Unpackage 6](#_Toc470174431)

[3 CASD Data Access 7](#_Toc470174432)

[Logical Model 8](#_Toc470174433)

[Implementation Model 9](#_Toc470174434)

[Application Infrastructure 10](#_Toc470174435)

[Application Domain 11](#_Toc470174436)

[Metadata 11](#_Toc470174437)

[Process statistics 12](#_Toc470174438)

[Status and error codes 13](#_Toc470174439)

[Design 15](#_Toc470174440)

[Java Class Diagrams 16](#_Toc470174441)

[Batch Process 16](#_Toc470174442)

[Task Manager 17](#_Toc470174443)

[Task 18](#_Toc470174444)

[Components 20](#_Toc470174445)

[Batch Process 20](#_Toc470174446)

[Batch Process 20](#_Toc470174447)

[Web Application 20](#_Toc470174448)

[Job Configuration and Task Management 20](#_Toc470174449)

[Appendix A. XML Schema Comparison 21](#_Toc470174450)

[Appendix B. References 22](#_Toc470174451)

# Development Objective

IDX (International Data Exchange) Agent application described in this document is designed to provide an interim solution, capable of handling the exchange of tax information between the Canada Revenue Agency (CRA) data providers, such as Infodec for Common Reporting Standard (CRS), or CDC for Country by Country Reporting (CbC), and the Common Transmission System (CTS), while following international standards for data transformation, encryption, transmission, confidentiality and security, until permanent solution produced by the designated vendor is put in place

# Solution Summary

Solution is based on the [Use Cases and Functional Requirements document](file:///\\Omega.dce-eir.net\natdfs\CRA\HQ\ITB\ITB_014\GV\Solutions\CDMD\TRIC\EEI\1%20Projects\Common%20Reporting%20Standard%20(CRS)\1%20Planning\PM-BA%20Processes-Documents\3%20HLBR-DBR\UseCases_FunctionalRequirements_v1.0.docx).

The system will be deployed on eBCI infrastructure platform.

To be able to discover incoming and outgoing data packages, system will run batch process which utilizes Java 1.7, EJB, JAXB, XML under WebLogic 12, and Batch Initiator RCCI component developed within CRA. Batch process(s) able to run at specified time intervals or ad-hoc, will be checking inbound and outbound directories for newly loaded data and status message files and taking care of saving them for CASD access.

File transfer process will be handled by (S)FTP through drop zones between CTS and CRA data provider (i.e. Infodec, or other)

To provide the Client with ability to access data in transition, to trigger further data processing, and to view historical status messages and transmission statistics, web user interface will be built using Java 1.7, Struts 2, WET 4, EJB, JSP, XML, JAXB, RCCI components, Weblogic 12.

System components will have access to the application database for saving metadata, status messages content and statistics during data processing. Web application will access application database to provide views of historical status messages and statistics.

Web application will have a screen for accessing data and status messages in transition for CASD to review data during preparation process and trigger further data processing.

When the tax information is accessed, the following Treaty stamp must be visible on the screen display, as well as being printed on all hard copies made5:

This information is furnished under the provisions of an income tax treaty with a foreign government. Its use and disclosure must be governed by the provisions of that treaty.

Les renseignements sont donnés en vertu des articles d’une convention fiscale conclue avec un Gouvernement étranger ces renseignements doivent être utilises et divulgues selon les articles de cette convention.

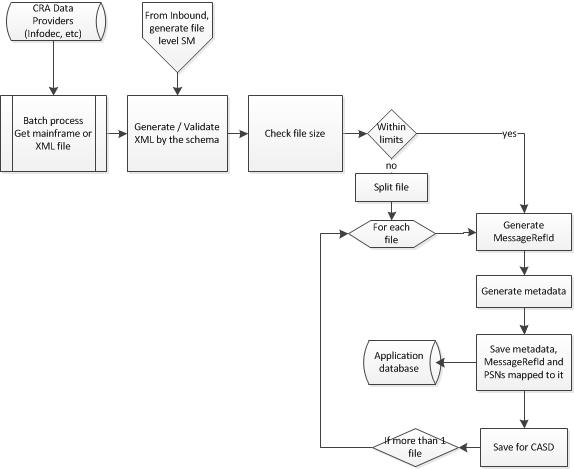
For conversion of data and status messages from flat files (Cobol copybooks) to XML format, application will select appropriate XML schema at run time based on data provider and message type. Java classes for data types must be pre-generated during development: for Cobol - from Cobol copybooks provided by Infodec using cob2java program, for XML - from XML schema(s) using JAXB.

# Application Architecture

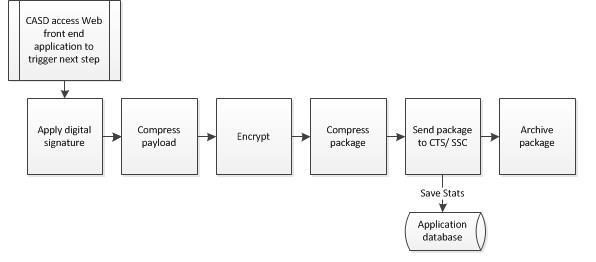
## Business Model

These diagrams give an overview of business processes implemented by the system.

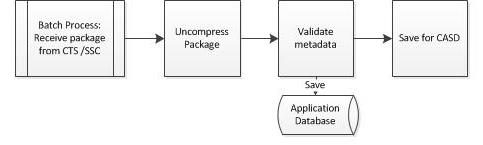
### Outbound – Prepare Payload



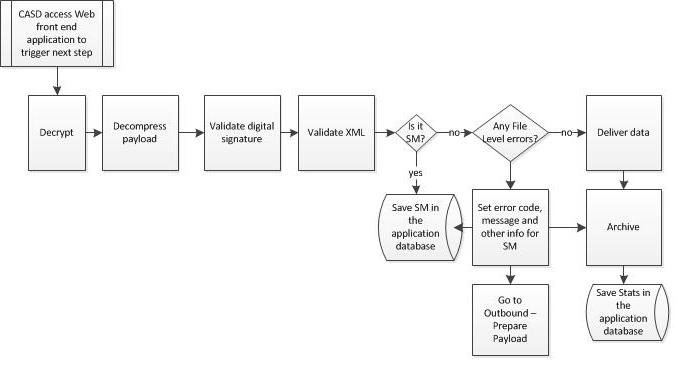
### 1.2 Outbound – Prepare Package



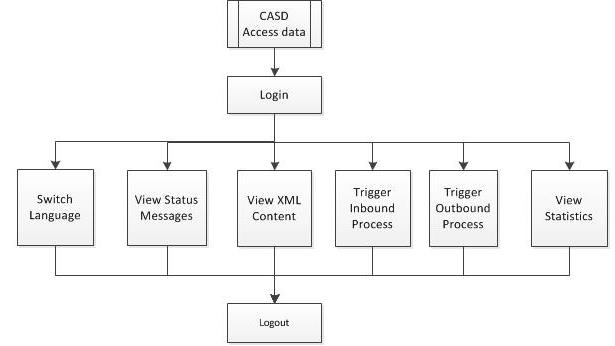
### 2.1 Inbound – Preliminary



### 2.2 Inbound – Unpackage

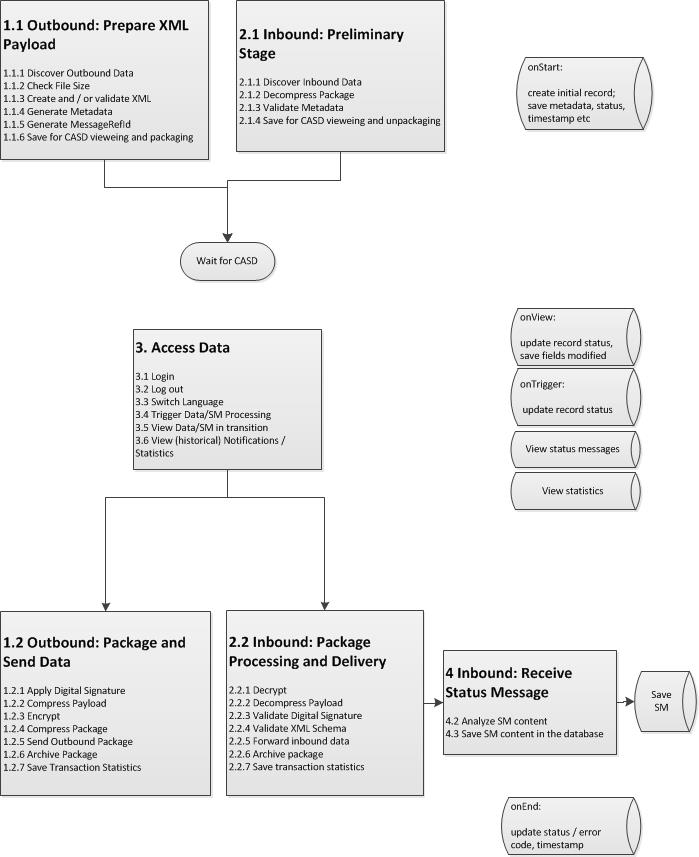


### 3 CASD Data Access



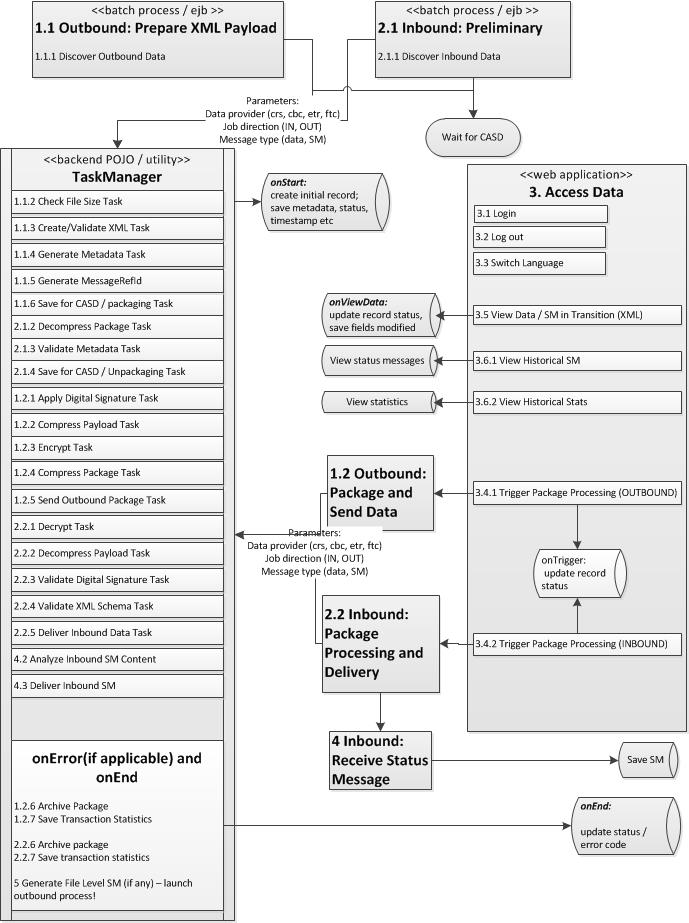
## Logical Model

This view presents business processes aligned according to the execution flow from start to end, with use cases grouped by business process.



## Implementation Model

This view presents an implementation model where actual physical components / applications responsible for execution of a certain process stage are defined. Sub-tasks are grouped together for better access and management.

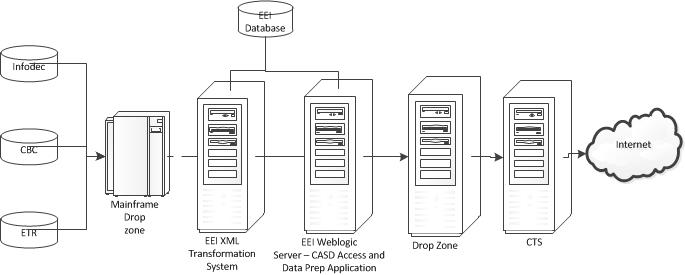


# Application Infrastructure

Application takes advantage of the CRA electronic Business Computing Infrastructure (eBCI) environment.

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Notes** |
| OS | Linux |  |
| RAM | 2 Gb allocated to the JVM |  |
| Disc Space |  |  |
| Web server | Weblogic 12 |  |
|  |  |  |
|  |  |  |

Infrastructure diagram



# Application Domain

## **Metadata**

Metadata provides information on the data file being transformed, prepared and transmitted between data provider and CTS. The following table lists metadata fields1

|  |  |  |
| --- | --- | --- |
| **Name** | **Example Value** | **Comment** |
| CTSSenderFileMetadata |  | Root element |
| CTSSenderCountryCd | CA | Identifies the jurisdiction of the Competent Authority transmitting the file. |
| CTSReceiverCountryCd | GB | Indicates the jurisdiction of the Competent Authority or Authorities that are the intended recipient of the file. |
| CTSCommunicationTypeCd | CRS, CRSStatus, ETR, ETRStatus, CBC, CBCStatus | Specifies the type of message transmitted. Only the values CRS, CRSStatus, ETR, ETRStatus, CBC and CBCStatus are currently allowed. |
| SenderFileId | <messageRefId>? | A free text field to capture the file name or ID created by the sender. The element helps both the sender and receiver to track and monitor a specific message. |
| FileFormatCd | XML | Specifies the file format of message transmitted, the only allowable value being XML. |
| BinaryEncodingSchemeCd | NONE | Identifies the type of encoding scheme for the transmission payload. If sending an XML file, the value should be ‘NONE’. |
| FileCreateTs | YYYY-MM-DD HH24:MI:SS UTC | Identifies the timestamp for the transmission payload created by the sender application. |
| TaxYear | 2015 | Optional; allows specifying the tax year to which the file relates. |
| FileRevisionInd | true  false | A Boolean field to indicate if the file is a revised message. The only allowable values are “true” or “false”. |
| OriginalCTSTransmissionId | ?  IDES format is  124FRNuzJUYnV0s5eYw1LOvrx1G10LcX | A free text field to reference the unique original CTS transmission ID. The identifier helps both the sender and receiver to track and monitor messages. |
| SenderContactEmailAddressTxt |  | A free text field to identify the sender email address. |

## **Process statistics**

The summary below is an example of statistics info. Full information on statistics is in

[CRS\_CBC Information to be stored in the database.docx](file:///\\Omega.dce-eir.net\natdfs\CRA\HQ\ITB\ITB_014\GV\Solutions\CDMD\TRIC\EEI\1%20Projects\COMMON%20TO%20ALL%20EOI%20Projects%20(EOI)\3%20Development\Phases\Design-Build\Data%20Modelling\CRS_CBC%20Information%20to%20be%20stored%20in%20the%20database.docx)

|  |  |  |
| --- | --- | --- |
| **Field** | **Example value** | **Comment** |
| Data provider | Crs, etr, cbc |  |
| Input file | IP.AIP5S182.CAUS.A14.S0000001 |  |
| Input file size, bytes | 22298 |  |
| XML file | CA-14-US-099999920160913T103322U6.xml |  |
| XML file size, bytes | 66800 |  |
| Schema validation result | VALID / INVALID / UNKNOWN |  |
| PKG-REF-ID Update File | IP.AIP5SMR.CAUS.A14.S0000001.X0000001 | ??? |
| Transmitting country | CA |  |
| Receiving country | GB |  |
| Message type |  |  |
| Warning |  |  |
| Contact |  |  |
| MessageRefId |  |  |
| Reporting period |  |  |
| Start Timestamp |  | When metadata is created? |
| End Timestamp |  | When package status is updated |
|  |  |  |
| Number of elements? |  |  |
| Number of FIs |  |  |
| Number of AccountReports |  |  |
| Number of DocRefIds |  |  |
|  |  |  |

Application uses database for storing metadata, status message information and transaction details.

## Status and error codes

By CRS Status Message Schema and User Guide, codes 50000-59999 are reserved for file level errors

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Code** | **Error Message** | **IN / OUT** | **Error Description** | **Action Requested** |
|  | success |  |  |  |
| **File Validations** | | | |  |
| 50001 | Failed to download | IN | The receiving Competent Authority could not download the referenced file. | Please resend the file. |
| 50002 | Failed decryption | IN | The receiving Competent Authority could not decrypt the referenced file. | Please re-encrypt the file with a valid key and resend the file. |
| 50003 | Failed decompression | IN | The receiving Competent Authority could not decompress the referenced file. | Please compress the file (before encrypting) and resend the file. |
| 50004 | Failed Signature Check | IN | The receiving Competent Authority could not validate the digital signature on the referenced file. | Please re-sign the file with the owner’s private key using procedures [as defined in the context of the CTS]. |
| 50005 | Failed Threat Scan | IN | The receiving Competent Authority detected one or more potential security threats within the decrypted version of the referenced file. Such threats include but are not limited to hyperlinks, Java script, and executable files. | Please scan the file for known threats and viruses, remove all detected threats and viruses prior to encryption and re-encrypt and resend the file. |
| 50006 | Failed Virus Scan | IN | The receiving Competent Authority detected one or more known viruses within the decrypted version of the referenced file. | Please scan the file for known threats and viruses, remove all detected threats and viruses prior to encryption, and re-encrypt and resend the file. |
| 50007 | Failed schema validation | IN | The referenced file failed validation against the CRS XML Schema. | Please re-validate the file against the CRS XML Schema, resolve any validation errors, and re- encrypt and resend the file. |
| 50008 | Invalid messageRefId format | IN | The structure of the MessageRefID is not in the correct format, as set out in the CRS User Guide.  The CRS User guide indicates that the MessageRefID can contain whatever information the sender uses to allow identification of the particular report but should start with the sending country code as the first element for Competent Authority to Competent Authority transmission, then the year to which the data relates, then the receiving country code before a unique identifier (e.g. FR2013CA123456789). | Please ensure the MessageRefID follows structure defined in the CRS User guide, and resend the file. |
| 50009 | MessageRefId has already been used | IN | The referenced file has a duplicate MessageRefID value that was received on a previous file.  Please do not submit a request to correct or delete any of the records in this file until you receive a CRS Status Message that this file has been received as valid (Status is Accepted). | Please replace the MessageRefID field value with a unique value (not containing all blanks), and resend the file. |
| 50010 | File Contains Test Data for Production Environment | IN | The referenced file contains one or more records with a DocTypeIndic value in the range OECD10-OECD13, indicating test data. As a result, the receiving Competent Authority cannot accept this file as a valid CRS file submission.  For more information on the DocTypeIndic data element, please consult the CRS User Guide. |  |
| 50011 | Error generating metadata | OUT |  |  |
| 50012 |  |  |  |  |
|  | data waiting | IN / OUT | For CASD |  |
|  | Error digitally signing | OUT |  |  |
|  | Error validating digital signature | IN / OUT |  |  |
|  | Invalid digital signature | IN / OUT |  |  |
|  | Error compressing payload | OUT |  |  |
|  | Error encrypting | OUT |  |  |
|  | Invalid certificate | IN / OUT |  |  |
|  | Error compressing package | OUT |  |  |
|  | Error decompressing package | IN |  |  |
|  | Invalid metadata | IN |  |  |
|  | Error validating metadata | IN |  |  |
|  | Error decrypting | IN |  |  |
|  | Error decompressing payload | IN |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Design

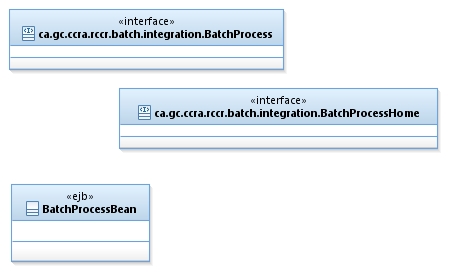
To make sure that the data transformation/preparation solution is flexible, maintainable and scalable, the following guidelines are recommended for design and implementation:

* Every system component should be independent and easily replaceable, in anticipation of the future integration of the COTS tool.
* In order to trigger different stages of data transformation, validation and preparation process, batch processes and web application access common collection of Java classes (tasks)
* Tasks are responsible for execution of data transformation, validation and preparation.
* Each use subcase related to data transformation and preparation process is implemented as a separate concrete Java class
* Each task class implements common interface ITask and extends the common abstract class AbstractTask, in order to implement methods common to all tasks and to be able to manipulate tasks without knowledge of each concrete type of task
* During execution tasks are organized into workflows, or jobs. Designated Java object will manage tasks, load and run jobs, depending on what component triggers the job, data provider and message type.
* To implement efficiently cases where inbound/outbound tasks share some methods, or task execution depends on data provider, see recommended design depicted in Java class diagram for the Task.

# Java Class Diagrams

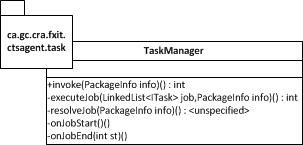
## Batch Process

To implement batch process on eBCI environment, a stateless session bean needs to be created, that runs within a WebLogic container and take advantage of RCCI component architecture. Application groups that want to develop and deploy a batch component must implement the batch Enterprise JavaBeans (EJB) using the following RCCI interfaces:



## Task Manager

Task Manager is a Java object dedicated to job /workflow management and configuration. Each job consists of the list of tasks to be executed in a certain order on a data package or status message. Depending on a data source and type of the package Task Manager selects and applies an appropriate job. Available jobs are configured in the configuration file and loaded at the application start.



Job types:

|  |  |  |  |
| --- | --- | --- | --- |
| **Type** | **Description** |  |  |
| ***IN\_PRE*** | receives data from CTS/SSC, uncompresses package, validates metadata and saves files in temp location for CASD, updates db stats |  |  |
| ***IN\_UPK*** | Updates db stats, decrypts payload, validates digital signature, validates XML and sends to the drop zone, updates db stats |  |  |
| ***OUT\_PL*** | gets data from Batch Job Initiator, transforms data to XML, checks file size, generates MessageRefId, generates metadata, saves MessageRefId, PSNs mapping to MessageRefId and metadata to the EEI application database, then saves the message file for further processing |  |  |
| ***OUT\_PK*** | Db, signs, encrypts and zips released XML and metadata files and sends the package to the drop zone for SSC/CTS, updates db |  |  |
| ***IN\_ PRE\_ SM*** | Receives notification in the XML format, validates metadata, saves in the database, and sends to the internal CRA drop zone |  |  |
| ***IN\_UPK\_SM*** | Updates db stats, decrypts payload, validates digital signature, validates XML and sends to the drop zone, updates db stats |  |  |

## Task

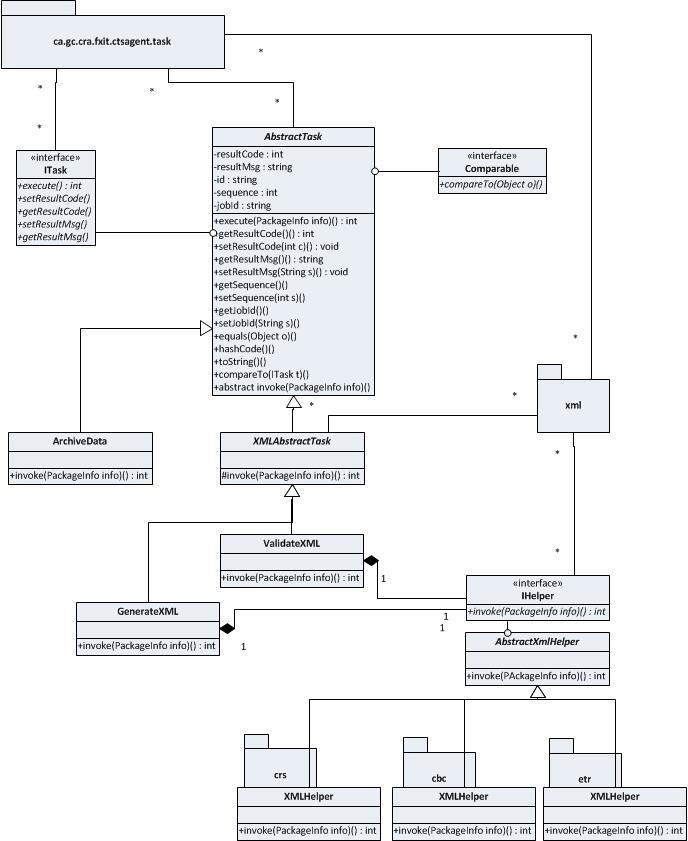
Each action that is performed on a data or data package is called Task. Each task implements an interface and extends an AbstractTask class which has methods common for all tasks. Each task can only access the PackageInfo object that is passed by the TaskManager to each task, or shared static utility methods, constants and application properties. Task is not aware of any other task, or TaskManager.

ArchiveTask in the diagram below is an example of simple task used in either inbound or outbound job.

Recommended implementation for tasks related to generation and validation of XML is depicted under XMLAbstractTask.

It is used when some or all methods are common for inbound and outbound jobs or are called in a different order by inbound and outbound jobs. Concrete class acts as a placeholder in the job and calls methods needed and in appropriate order from the parent class.

If the task is executed differently for different data providers, one of the recommended implementation options is for each concrete task class to use a Helper object. At runtime task loads Helper dynamically, based on the data provider prefix value.



# Components

## Batch Process

* Discovering of inbound files

## Batch Process

* Discovering outbound files

## Web Application

* Login action, screen(s), web configuration of authentication and authorization access
* Logout action, screen(s)
* Switch Language action, screens
* View inbound XML files (data and status messages) in transition: actions and screen(s) for view and update XML
* View outbound XML files (data and status messages) in transition: actions and screen(s) for view and update XML
* Trigger inbound data/status message processing: action, screen(s)
* Trigger outbound data/status message processing: action, screen(s)
* View historical statistics data: action, screen, integrate data access object(s)
* View historical status messages content: action, screen, integrate data access object(s)

## Job Configuration and Task Management

* Task Manager object and task framework
* Tasks Process Direction
  + Apply Digital Signature OUT
  + Archive Inbound Data IN / OUT
  + Compress Package OUT
  + Compress Payload OUT
  + Check File Size OUT
  + Decrypt IN
  + Deliver Inbound Data IN
  + Deliver Outbound Data OUT
  + Encrypt OUT
  + Generate MessageRefId OUT
  + Generate Metadata OUT
  + Generate XML OUT
  + Save Prepared Inbound Data IN
  + Save Prepared Outbound Data OUT
  + Uncompress Package IN
  + Uncompress Payload IN
  + Validate Digital Signature IN
  + Validate Metadata IN
  + Validate XML IN
  + Save Statistics IN / OUT
  + Generate File Level Error SM IN to OUT

# Appendix A. XML Schema Comparison

This comparison is provided for a generic overview only. Please refer to the appropriate XML schema and User Guide for exact rules.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Element Description** | **FATCA** | **CRS** | **CbCR** | **ETR** |  |
| Root | FATCA\_OECD | CRS\_OECD | CBC\_OECD | ETR\_OECD |  |
| Root attributes | X | X | X | X |  |
| Root children | ftc:MessageSpec  ftc:FATCA(1…oo) | crs:MessageSpec  crs:CrsBody(1…oo) | cbc:MessageSpec  cbc:cbcBody(1…oo) | etr:MessageSpec  etr:EtrBody(1…oo) |  |
| MessageSpec fields | ***sfa:MessageSpec\_Type***  sfa:SendingCompanyIN  sfa:TransmittingCountry  sfa:ReceivingCountry  sfa:MessageType  sfa:Warning  sfa:Contact  sfa:MessageRefId  sfa:CorrMessageRefId(0…oo)  sfa:ReportingPeriod  sfa:Timestamp | ***crs:MessageSpec\_Type***  crs:SendingCompanyIN  crs:TransmittingCountry  crs:RecevingCountry  crs:MessageType  crs:Warning  crs:Contact  crs:MessageRefId  crs:MessageTypeIndic  crs:CorrMessageRefId(0…oo)  crs:ReportingPeriod  crs:Timestamp | ***cbc:MessageSpec\_Type:***  cbc:SendingEntityIN  cbc:TransmittingCountry  cbc:ReceivingCountry (0…oo)  cbc:MessageType  cbc:Language  cbc:Warning  cbc:Contact  cbc:MessageRefId  cbc:MessageTypeIndic  cbc:CorrMessageRefId  cbc:ReportingPeriod  cbc:Timestamp | ***etr:MessageSpec\_Type***  etr:TransmittingCountry  etr:ReceivingCountry  etr:MessageType  etr:Language  etr:Warning  etr:Contact  etr:MessageRefId  etr:MessageTypeIndic  etr:Timestamp |  |
| Body | ftc:Fatca\_Type:  ftc:ReportingFI  ftc:ReportingGroup(1…oo) | crs:CrsBody\_type:  crs:ReportingFI  crs:ReportingGroup(1) | cbc:CbcBody\_Type:  cbc:ReportingEntity  cbc:CbcReports(0…oo)  cbc:AdditionalInfo(0…oo) | etr:EtrBody\_Type:  etr:TaxPayer(0…oo)  etr:RulingReports(0…oo) |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Appendix B. References

1 COMMON APPROACH TOWARDS CTS FILE PREPARATION AND ENCRYPTION Version approved by the ESG OECD Conference Centre, Paris June 2016

2 34592 - CRS - Notifications - CRA to OJ V1.0 2016-10-04

3 <http://www.oracle.com/technetwork/articles/java/batch-1965499.html>

4 [G:\EEI\1 Projects\Common Reporting Standard (CRS)\1 Planning\PM-BA Processes-Documents\3 HLBR-DBR\CTSDataPrep\_UseCases\_FunctionalRequirements\_v1.0.docx](file:///G:\EEI\1%20Projects\Common%20Reporting%20Standard%20(CRS)\1%20Planning\PM-BA%20Processes-Documents\3%20HLBR-DBR\CTSDataPrep_UseCases_FunctionalRequirements_v1.0.docx)

<<END OF DOCUMENT>>